



HAL
open science

Méthodes statistiques pour la différenciation génotypique des plantes à l'aide des modèles de croissance

Gautier Viaud

► **To cite this version:**

Gautier Viaud. Méthodes statistiques pour la différenciation génotypique des plantes à l'aide des modèles de croissance. Mathématiques générales [math.GM]. Université Paris Saclay (COmUE), 2018. Français. NNT : 2018SACLC020 . tel-01772414

HAL Id: tel-01772414

<https://theses.hal.science/tel-01772414v1>

Submitted on 20 Apr 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Méthodes statistiques pour la différenciation génotypique des plantes à l'aide des modèles de croissance

NNT : 2018SACLC020

Thèse de doctorat de l'Université Paris-Saclay
préparée à CentraleSupélec

École doctorale n°573 Interfaces
Spécialité de doctorat : Mathématiques appliquées

Thèse présentée et soutenue à Gif sur Yvette, le 22 janvier 2018, par

Gautier Viaud

Composition du jury :

Paul-Henry COURNÈDE MICS, CentraleSupélec, Université Paris-Saclay	Directeur de thèse
Jean-Louis FOULLEY IMAG, Université de Montpellier, CNRS	Examinateur
Janine ILLIAN CREEM, University of Saint Andrews	Rapporteuse
Estelle KUHN MaIAGE, INRA	Examinatrice
Olivier LOUDET IJPB, INRA	Examinateur
Frédéric MORTIER UR Forêts et Sociétés, CIRAD	Rapporteur
Jean-Christophe PESQUET CVN, CentraleSupélec, Université Paris-Saclay	Président du jury

Thèse de doctorat de l'Université Paris-Saclay
préparée à CentraleSupélec

École doctorale n°573 Interfaces
Spécialité de doctorat : Mathématiques appliquées

Gautier Viaud

Méthodes statistiques pour la différenciation génotypique des plantes à l'aide des modèles de croissance

Thèse présentée et soutenue à Gif sur Yvette, le 22 janvier 2018

Composition du jury :

Paul-Henry COURNÈDE MICS, CentraleSupélec, Université Paris-Saclay	Directeur de thèse
Jean-Louis FOULLEY IMAG, Université de Montpellier, CNRS	Examinateur
Janine ILLIAN CREEM, University of Saint Andrews	Rapporteuse
Estelle KUHN MaIAGE, INRA	Examinatrice
Olivier LOUDET IJPB, INRA	Examinateur
Frédéric MORTIER UR Forêts et Sociétés, CIRAD	Rapporteur
Jean-Christophe PESQUET CVN, CentraleSupélec, Université Paris-Saclay	Président du jury

Remerciements

Bien qu'il en soit de coutume, il me tient à cœur de remercier en premier lieu Paul-Henry Cournède, le directeur de cette thèse mais aussi et avant tout la personne qui a bien voulu laisser leur chance aux errances qui m'ont guidé jusqu'à son équipe de recherche. J'ignore s'il est encore nécessaire de souligner, après tant de thèses encadrées avec brio, ses qualités de chercheur et de directeur de thèse. Au-delà du scientifique, j'ai également découvert une personne éminemment cultivée et surprenante par de nombreux aspects. Pour toutes ces raisons et bien d'autres, j'ai été sincèrement honoré de réaliser cette thèse sous sa direction.

Je suis également particulièrement reconnaissant envers les rapporteurs de ma thèse, Frédéric Mortier et Janine Illian (dont l'irréprochable français m'a ôté tout scrupule de rédiger ces paragraphes dans ma langue natale), pour leur relecture et leurs suggestions d'amélioration qui ont permis à ce document d'acquérir une dimension nouvelle, ainsi que les examinateurs Estelle Kuhn, Jean-Louis Foulley et Jean-Christophe Pesquet pour leurs remarques pertinentes, les perspectives de continuation de ce travail qu'ils m'ont fournies, et certaines conversations passionnées et passionnantes. Je n'oublie évidemment pas Olivier Loudet, dernier membre de mon jury de thèse, à qui je tiens à exprimer en sus ma plus sincère gratitude pour le don désintéressé des nombreuses données d'images sans lesquelles la cohérence globale de ce travail n'aurait tout simplement pas été.

Comment aborder le délicat sujet du laboratoire MICS, dont je sais qu'aucun de mes efforts ne gratifiera à leur juste valeur l'ensemble de ses membres ? Sylvie, qui fut pour moi l'âme du laboratoire, et au bureau de laquelle commençait radieusement chacune de mes journées, que je me dois de remercier pour sa bonne humeur, son sens de l'humour, des discussions beaucoup trop longues, sans oublier nos projets de vie : à bientôt pour le bar à tapas, Sissi. Benoît, qui est probablement la personne de qui j'aurai le plus appris : pour tout ce qui concerne de près ou de loin l'informatique, c'est une évidence, dans un nombre incalculable d'autres domaines impossibles à lister de façon exhaustive, c'était plus inattendu. Merci donc avant tout pour les Bayolades, cette hyperactivité transdisciplinaire et ces innombrables références de toutes sortes. Laurent, avec qui j'ai pu partager en toute causticité mes problèmes liés à Julia et au mésocentre, des débats politiques cyniques mais révélateurs, nos passions communes pour qui la sophrologie, qui l'éducation positive, qui l'écriture inclusive. Ceux qui ont partagé mon bureau avec tant de patience : Pierre, pour sa considérable culture cinématographique et musicale ainsi que mon initiation aux échecs, Jean-Christophe, pour ses indispensables cours de mécanique ; avoir remporté haut la main le jeu de la boîte de balles de tennis restera probablement l'un des plus grands accomplissements de ma thèse (l'histoire est-elle écrite par les vainqueurs ou ceux qui en

prennent la peine ?). D'autres encore, que leur existence laborantine ait inclus la mienne – sur le chemin du million, Ludovic – ou inversement – dommage que les Vosges t'aient tant manqué, Julien. Que soient loués également ceux qui ont été les principaux acteurs de l'époque insouciante et jubilatoire de ma première année de thèse : Alexandre, Benjamin, Charlotte, Marion et Pierre-André. Désireux de garder ces remerciements plus succincts que le reste de ce document, je remercie avec une concision qui ne leur fait certainement pas honneur le reste des membres de l'équipe Biomathematics, les anciens (Robert et Yuting), les confirmés (Sarah et Véronique), la relève (Antonin, Chloé, Mahmoud, Mathilde, Walid, Xiangtuo), les éloignés (Étienne et Thomas) ainsi que certains autres du laboratoire (Adrien, Brice, Kevin, Marcus, Rémi) pour l'ensemble de tous ces bons moments passés au cours de ces années.

Il m'est impossible de passer sous silence le rôle salvateur qu'ont exercé mes amis parisiens afin de me préserver des affres de l'isolement social doctoral : merci donc à Anaïs et Édouard ainsi qu'à Clarisse, Eliott, Flore, Jonathan, Kevin, Lucile et Marine pour ces années dont je garderai de merveilleux souvenirs. Moins présents, essentiellement car plus éloignés, je n'en oublie pas pour autant mes amis d'enfance que je regrouperai ici sous l'appellation des Caennais. Et parce que les remerciements ne devraient pas être l'apanage des lettrés, mentionnons également Neva, Hubert, Bonzini, l'Octomore, les Cohiba, les rives du canal de l'Ourcq, la terrasse du fort d'Issy ainsi que l'appartement de Plan Peisey.

J'aimerais conclure en remerciant très logiquement ma famille : mon frère, tout d'abord, pour ces souvenirs mémorables que nous laisseront le Steaking et cette escapade éclair à l'Alpe d'Huez, et mes parents, enfin, sans qui ce travail n'aurait probablement pas été possible, pour leur soutien, leur patience et leur compréhension durant l'intégralité de cette thèse, et plus particulièrement la période de rédaction.

Encore une fois, à tous, merci.

G.

Résumé

L'un des principaux objectifs des modèles de croissance de plantes pour l'agriculture concerne la prédiction de quantités d'intérêt telles que le rendement des cultures, mais ils peuvent également constituer des outils essentiels pour l'évaluation de la variabilité génotypique au sein d'une population de plantes. Ce double usage est mis en évidence tout au long de ce travail, dont les contributions peuvent être décomposées en cinq parties.

Tout d'abord, le modèle Log-Normal Allocation and Senescence (LNAS) pour la betterave est rappelé et reformulé. C'est un modèle compartimental qui vise à décrire uniquement les principaux processus biologiques pour le bilan de biomasse au cours de la croissance. Le modèle LNAS pour le blé, plante plus complexe, est ensuite proposé comme une adaptation de ce modèle compartimental couplé à un modèle de sol. Enfin, une adaptation de GreenLab, modèle à l'échelle de l'organe, est également conçu pour *Arabidopsis thaliana*.

Une nouvelle plate-forme de calcul pour la modélisation et l'inférence statistique (ADJUSTIN') a été développée tout au long de cette thèse. Elle est fondée sur l'utilisation du récent langage Julia qui présente de nombreuses caractéristiques attrayantes : ses performances élevées ainsi qu'une parallélisation simple des différents algorithmes ont permis de rendre ADJUSTIN' particulièrement efficace, son expressivité permet une écriture aisée des différents modèles et algorithmes et ses capacités de métaprogrammation en font un excellent outil pour une gestion générique de ces modèles et algorithmes. ADJUSTIN' permet donc la modélisation et la simulation des modèles de croissance de plantes considérés ainsi que l'utilisation de techniques d'estimation de pointe telles que, entre autres, les méthodes de Monte Carlo par chaînes de Markov (MCMC) ou les méthodes de Monte Carlo séquentielles (SMC).

L'inférence statistique au sein des modèles de croissance de plantes étant de première importance pour des applications concrètes telles que la prédiction de rendement, les méthodes d'estimation de paramètres et d'états au sein de modèles à espaces d'états et dans un cadre bayésien furent tout d'abord étudiées, et plusieurs cas d'étude pour les trois modèles de croissance de plantes considérés sont ensuite analysés pour le cas d'une plante individuelle. Notamment, différents scénarios pour l'assimilation de données pour le blé, en présence de données rares et hétérogènes, sont examinés afin de souligner la performance du filtre particulaire régularisé et la robustesse de l'approche bayésienne. Une méthode MCMC particulière est également utilisée pour le modèle LNAS appliqué à la betterave pour une estimation précise des paramètres et états cachés et une estimation subséquente des bruits de modélisation et d'observation.

La caractérisation de la variabilité au sein d'une population de plantes est envisagée à travers les distributions des paramètres de population au sein de modèles hiérarchiques bayésiens. Cette approche requiert l'acquisition de nombreuses données pour chaque individu de la population. Un algorithme de segmentation-suivi pour l'analyse d'images d'*Arabidopsis thaliana*, obtenues grâce au Phénoscope, une plate-forme de phénotypage à haut rendement de l'INRA Versailles, est proposé. Ces deux étapes de segmentation et suivi tirent parti de caractéristiques spécifiques à cette espèce et il est démontré, avec l'exemple de quatre individus appartenant à des génotypes différents, comment cette procédure globale permet d'obtenir des données nombreuses et précises.

Enfin, l'intérêt de l'utilisation des modèles hiérarchiques bayésiens pour la mise en évidence de la variabilité au sein d'une population de plantes est discutée. D'abord par l'étude de différents scénarios sur des données simulées à partir du modèle GreenLab, et enfin en utilisant les données expérimentales obtenues pour une population d'*Arabidopsis thaliana* comprenant 48 individus à partir de l'analyse d'images.

Contents

Remerciements

Résumé

Introduction	1
1 Mathematical framework	9
1.1 General state space models	10
1.1.1 Main equations	10
1.1.2 Hidden Markov models	11
1.1.3 Structure of the observations	12
1.1.4 Observation model	14
1.1.5 Extensions	14
1.2 Generic probability distributions	15
1.2.1 Transition probability density function	15
1.2.2 Observation probability density function	17
1.3 Population models	18
1.3.1 Motivation	18
1.3.2 Hierarchical models	20
1.3.3 State space models for populations	21
1.3.4 Intraindividual variability	21
1.3.5 Interindividual variability	22
2 Plant growth models	25
2.1 History	25
2.2 Log-Normal Allocation and Senescence for <i>Beta vulgaris</i>	28
2.3 Log-Normal Allocation and Senescence for wheat	32
2.3.1 Soil model	33
2.3.2 Plant model	36
2.4 GreenLab for <i>Arabidopsis thaliana</i>	40
3 Estimation of parameters and hidden states for a single individual	47

3.1	General principle	47
3.1.1	Data for calibration	48
3.1.2	Criteria	49
3.2	Sensitivity analysis	50
3.2.1	Sobol method	50
3.2.2	Structural and correlative sensitivity analysis	51
3.3	Frequentist vs. Bayesian estimation	54
3.4	A frequentist method: generalized least squares	56
3.5	Bayesian methods	57
3.5.1	Markov chain Monte Carlo methods	58
3.5.2	Sequential Monte Carlo methods	65
3.5.3	Particle Markov chain Monte Carlo	78
4	Estimation of parameters within population models	83
4.1	Frequentist approach	84
4.2	Bayesian approach	85
4.2.1	Full conditional distributions	86
4.2.2	Choice of the prior distributions	87
4.2.3	Non-linear models	91
4.2.4	Metropolis–Hastings for sampling the individual parameters	93
4.2.5	Individual adaptive scheme	94
5	Adopting Julia for statistical inference	97
5.1	Why Julia?	99
5.2	Modelling and simulation	100
5.2.1	Mathematical framework	100
5.2.2	Model types	101
5.2.3	Simulation core	104
5.2.4	Flexible modelling	106
5.3	Platform organization	106
5.4	Operations on system observations	109
5.5	Probability density functions	110
5.5.1	Sampling	110
5.5.2	Prior distribution	111
5.5.3	Transition and observation distributions	112
5.6	Algorithms	112
5.6.1	Sobol method	113
5.6.2	Adapted Metropolis within Gibbs	114
5.6.3	Regularized particle filter	115
5.6.4	Particle marginal Metropolis–Hastings	119

5.6.5	Adaptive hybrid Metropolis–Hastings–Gibbs	120
5.7	Discussion	120
6	Estimation in state space models: application	123
6.1	Comparison of frequentist and Bayesian approaches for the GreenLab model	124
6.1.1	Sensitivity analysis	124
6.1.2	Data simulation	125
6.1.3	Parameter estimation	127
6.2	RPF-based data assimilation for the LNAS wheat model	130
6.2.1	Sensitivity analysis	131
6.2.2	Data simulation	132
6.2.3	Uncertainty analysis and data assimilation	133
6.2.4	Influence of the number of observations	134
6.2.5	Influence of the observation noise	135
6.2.6	Influence of the prior	137
6.2.7	Influence of the number of estimated parameters	141
6.3	Parameter and state estimation using the PMMH sampler for the LNAS sugar beet model	142
6.3.1	Sensitivity analysis	142
6.3.2	Data simulation	144
6.3.3	Comparison of filters within PMMH	144
6.3.4	Estimation of process and observation noises	148
7	Image analysis	153
7.1	Phenoscope	154
7.2	Data analysis	156
7.2.1	Analysis of angles	157
7.2.2	Analysis of areas	159
7.3	Segmentation	160
7.4	Tracking	163
7.5	Preliminary results on four individuals	165
7.6	Results on a large data set	168
7.7	Discussion	171
8	Estimation within population models: application	175
8.1	Simulated data	176
8.1.1	Simulation of synthetic data	176
8.1.2	Initialization of the prior distributions	177
8.1.3	Details of implementation	181
8.1.4	First results	182
8.1.5	Influence of the number of individuals	183

8.1.6	Influence of the precision	185
8.1.7	Influence of the prior	188
8.1.8	Influence of the number of estimated parameters	190
8.2	Real data	192
8.2.1	Estimation on 24 individuals	192
8.2.2	Integration of other individuals	196
Discussion and perspectives		201
Appendices		211
A	Calculation of full conditional distributions	211
B	LNAS model for sugar beet in ADJUSTIN'	215
C	UKF algorithm in ADJUSTIN'	217
D	Database and results directories	219
E	Parallel computation of mean and covariance	221
F	Exchange of particles between processes	223
Bibliography		225

Introduction

Context

THE CHALLENGE OF PREDICTING THE PHENOTYPE OF PLANTS based on both their genotype and the environment they evolve in is a critical and topical issue in plant science. Such a prediction has its share of difficulties because the plant's phenotype can be deeply affected by small genome or environment variations but also by the strong genotype by environment interactions that affect the plant's phenotype. These effects have long been known in the plant breeding community (see [Allard and Bradshaw, 1964], [Hill, 1975] or [El-Soda et al., 2014] for a more recent review in the specific case of *Arabidopsis thaliana*).

In order to predict plant phenotypic performance, statistical models are usually built based on linear mixed-effect models for integrative variables [Bustos-Korts et al., 2016]. Their strength is that they take advantage of large repetitions of trials in very diverse environmental conditions since they necessitate only a restricted amount of data, but they offer poor perspectives in terms of interpretation and extrapolation. On the contrary, since they rely on the mechanistic description of growth processes, plant growth models have opened promising perspectives for the description and prediction of genotype by environment interactions.

Over the last decades, many plant growth models have been proposed in order to predict a plant's phenotype under a wide range of varying environments, by describing the main physical processes of the plant's growth [Martre et al., 2011], [Chapman, 2008]. Since the latter highly depend on the genotype and its interaction with the environment [Kusano et al., 2007], [Kromdijk et al., 2014], calibration of such models is often specific to one genotype. This not only restricts the validity range of the model but prevents a proper assessment of the variability of the model parameters across a population of plants. One of the issues of interest in this thesis is how to highlight the variability of model parameters in a given population of plants. Hierarchical models provide an interesting framework in order to answer such a question, allowing to decompose a plant's phenotype as the result of two levels, one taking into account the genotypic variability of the related model parameters and the other being the actual model whose evolution is driven by these parameters.

Mathematically speaking, if we consider the system of interest as the plant in its environment (or a population of plants, or a specific part of the plant for models at smaller scales) plant growth models could formally be represented in the very generic following form:

$$y = f(\theta, u) \tag{1}$$

where:

- y represents all the phenotypic traits of interest, and is generally a real-valued function of space and time. In most models at the whole plant scale, this function is generally discretized in both space and time and y is a vector, for example representing the daily values of organ masses and dimensions (see for example most functional structural plant models (FSPMs) and a recent review by [Vos et al., 2009]);
- f represents the functional equations (usually dynamical, see for example the description of plant growth models as dynamic state space models and hidden Markov models in [Cournède et al., 2013]). In literature, most mathematical formalisms can be found at this stage. Mostly, we find ordinary differential equations or finite difference equations in time, and we follow the evolution of a vector of state variables related through nonlinear equations. These equations are often deterministic, but efforts have been recently pursued to achieve a better statistical evaluation of the model with stochastic versions of the dynamical system [Cournède et al., 2013]. A particularity for plant growth is that the evolution of the structure usually necessitates to consider organogenesis submodels. When only the dynamics of organ appearance is concerned, generative formal grammars of the L-Systems type are generally used, see for example [Prusinkiewicz and Lindenmayer, 1990], [Kurth, 1994]. However, when the morphogenesis process is considered, partial differential equations or partial differential integro-differential equations are used [Dupuy et al., 2008], [Beyer et al., 2014], [Beyer et al., 2015];
- θ represents all the parameters of the model and can be considered to be a real-valued vector, $\theta \in \mathbb{R}^p$. Some of them are of biophysical relevance but some are only empirical parameters of descriptive functions. As we will detail later, the estimation of these parameters is a key issue in plant growth modelling;
- u represents all the external variables for the system and mostly corresponds to environmental variables. At the global scale, the main variables generally correspond to radiation, temperature, potential evapotranspiration, soil content in water and nutriment. The agricultural, horticultural and forestry practice can also be represented in the environmental variables.

Models differ with respect to the phenomenon of interest and the studied species. However, some pretty generic frames were introduced such as STICS [Brisson et al., 2003] for crop models or GreenLab [Yan et al., 2004] for FSPMs.

For a given species and a given model, the parameters should ideally be able to characterize genotypes. As stated by [Tardieu, 2003], the application associating its model parameter vector to each genotype should be injective. In such an ideal situation, we could imagine very concrete applications: for instance, for a given environment u , comparing the performances of two genotypes characterized by two different parameter sets θ_1 and θ_2 . Conversely, if the parameter set is stable for one genotype in a large range of environmental conditions, we can optimize some traits of interest with respect to the environmental conditions – see examples for maize [Qi et al., 2010], sunflower [Lecœur et al., 2011] or peach [Quilot-Turion et al., 2012] – leading

to potential decision aid tools. One example would be the optimization of water supply under logistic and availability constraints [Wu et al., 2012].

If the ecophysiological parameters characterize a given genotype, then we could also imagine to decompose the genetic variation of model parameters into individual quantitative trait loci, or conversely to design a predictive model determining this parameter set from the plant genetics, that is to say to write $\theta = h(g)$ where g represents the individual plant genotype, either the genomic sequence or a representation of it with quantitative trait loci markers – see for example [Reymond et al., 2003], [Quilot et al., 2005], [Hammer et al., 2006], [Letort et al., 2008], [Xu et al., 2011], [Des Marais et al., 2016]. Several strategies are possible. The most accessible one seems to use a quantitative genetics model to determine the parameter values through combination matrices giving the relative influences of the allelic values and representing both the pleiotropy and epistasis phenomena [Letort et al., 2008]. Other methods could involve approaching h with machine learning techniques, a technique that is common in "omics" science to determine other outputs of interest from the genomic sequence and has recently been tested in plant growth modelling [Migault et al., 2017].

As described by [Yin and Struik, 2010] or [Baldazzi et al., 2016], the tendency is to complicate the mechanistic description of biophysical processes, by linking ecophysiology to "omics" sciences as an attempt to fully comprehend the regulatory networks from which plant robustness and plasticity is supposed to emerge [Hirai et al., 2004] whilst the related robustness appears to be difficult to achieve at the cell or tissue level. The modelling of plant growth and development lends itself to such an integrative approach. Several models for various component systems of plants are constantly developed [Hodgman et al., 2009]. However, the road is still long to achieve such an ambitious objective, resulting in a predictive model from the genes to the whole plant phenotype in a large range of environmental conditions. The more complex the models, the more troublesome their parameterization and the assessment of the estimate uncertainty [Ford and Kennedy, 2011], specifically due to costly experimentations and the large number of unknown parameters to consider. Likewise, local environmental conditions (in terms of climatic and soil variables, as well as biotic stresses) and initial conditions in specific fields are also very delicate to characterize. Consequently, the propagation of uncertainties and errors, which are related to parameters and inputs of these dynamic models, may result in poor prediction of the plant-environment interaction in real situations [Chen, 2014].

A good compromise between the mechanistic description of plant growth processes and the level of details in the data necessary for their parameterization has recently emerged with a new paradigm for plant ecophysiological modelling, namely functional-structural plant modelling [Vos et al., 2009]. It combines the ecophysiology of plant growth to its architectural development. One of their fundamental properties is that their parameterization does not rely on the same type of information as classical ecophysiological models: architectural traits have the property to integrate the whole history of plant functioning, and a large information (in the Fisher sense) on model parameters can be inferred from the observation of the architectural traits. The key point that we aim at taking advantage of in this work is that architectural traits can potentially be measured efficiently by automatic image analysis in high-throughput phenotyping platforms. These have recently gained increasing interest, both in fields [Araus and Cairns, 2014] and laboratories [Tisné et al., 2013], thanks to their capacity to automatically measure many morphological and physiological traits for a

large number of plant genotypes in various environmental conditions. However, although these measurements are potentially very detailed in time, they usually concern integrative traits (masses, total leaf area, height, etc.) and are again classically analyzed with descriptive statistical (multifactorial) models (see for example [Granier and Vile, 2014]).

The main objectives of this work will hence be to develop a statistical method to identify the interindividual variability of a population of plants composed of several genotypes. We will do so by describing the functioning of the plants using mechanistic growth models. Throughout this thesis, whether it be for single individuals or populations, the estimation of functional parameters or noise parameters, we will indeed adopt a Bayesian approach which is believed to be the most appropriate to the study of the biological systems that are plants for it not only allows to integrate prior biological knowledge into the estimation of parameters but also appears to be best suited for handling scarce and heterogeneous data as is often the case in plant applications. In the proposed model-based and Bayesian approach, the variability in question will be described by the posterior distribution of the parameters of the plant growth models considered in a hierarchical framework. This is in fact the first time that such a combination is considered in plant growth modelling: previous works in the field have considered the use of Bayesian hierarchical models [Schneider et al., 2006] but with relatively simple models describing the state variables of the plant, some others have considered using plant growth models combined to hierarchical models to explain the variability of a population of plants [Baey, 2014] but in a frequentist paradigm.

This ambitious final objective requires preliminary works in several matters. First, the design of plant growth models in a rigorously defined mathematical framework allowing to take into account the diverse inputs for biological systems, such as the environment, the initial state of the system and the process and observation noises as well as the description of the evolution of this system through hidden states and observations. Such models should be applicable to either a single plant or a population of such plants. Within this framework, mechanistic plant growth models will be proposed for different species. Although the ultimate aim of this work is the study of a population of plants, a first step in the apprehension of the dynamics and calibration of these models involves a single individual, which also allows to tackle common practical issues arising in agriculture and plant science. This requires the design and the practical and generic implementation of state-of-the-art algorithms for parameter and state estimation. To fulfil the expectation of evidencing the variability of a population of plants using Bayesian hierarchical models, one requires a data set comprising many individuals with many observations per individual.

For this purpose, we will also propose a method adapted to high-throughput phenotyping platforms by developing an algorithm of image analysis to segment plant individual leaves in order to compute their areas, but also to individually track each leaf across the different images. We will apply our algorithm to numerous time series of images of *Arabidopsis thaliana* individuals. This work will therefore lie at the intersection of many disciplines including mathematical modelling, plant science, statistical inference, computer science and image analysis.

Outline

The applications considered in this thesis will therefore be decomposed into two parts, each of which will address different issues. The first one will concern estimation techniques in the case of a single individual as an illustration of the dynamics of the designed plant growth models and the main problems that need to be dealt with in plant science and agriculture. This will constitute a first step fulfilling the necessity of understanding what happens with a single individual, before the study of genotypic variability among a population of plants. This second part will specifically focus on parameter estimation in Bayesian hierarchical models.

This thesis will hence be composed of eight different chapters. In Chapter 1, we will introduce the mathematical framework that will be used throughout this work. This comprises notably general state space models and the formulation of their transition and observation distributions. When dealing with a population of individuals, hierarchical models constitute an especially suited framework, particularly in the context of Bayesian estimation.

The three main plant growth models that are going to be considered throughout this work are presented in Chapter 2. After a brief history of plant growth models that justifies the use of functional-structural models for genotypic differentiation, we introduce three different models for sugar beet, wheat and *Arabidopsis thaliana*. The LNAS model for sugar beet first introduced in [Cournède et al., 2013] and [Chen, 2014] will serve as an illustrative model comprising the two possible kinds of noise considered in this thesis (process and observation noise) for the tasks of parameter and state estimation in the most general framework for plant growth models. The LNAS model for wheat was designed in this thesis; it was inspired by the main principles of the LNAS model for sugar beet for the modelling of the allocation and senescence processes and also introduces an original water budget model. Last but not least, we present a new version of a GreenLab-based model for *Arabidopsis thaliana*. It describes the plant at the scale of the organ in order to incorporate as much information on the functioning of the whole plant growth as possible for a better differentiation between individuals than if global variables, such as the total leaf area, were used. All these models are presented within the mathematical framework of Chapter 1.

Chapters 3 and 4 deal with the estimation techniques used for the calibration of the models. Throughout this work, we decided to adopt a Bayesian point of view. Chapter 3 focuses on the algorithms used for parameter and state estimation for a single individual, with a variety of techniques ranging from Markov chain Monte Carlo methods to sequential Monte Carlo methods and their combination. This review of state-of-the-art estimation methods represents a preliminary step to the study of how inference (for parameters and states) can be performed for plant growth models in the case of a single individual.

Chapter 4 introduces the convenient methods of Bayesian estimation for hierarchical models. Notably, emphasis is put upon the choice of the Bayesian perspective. A generic methodology adapted to dynamic state space models based on Gibbs sampling is proposed. These methods will therefore be used for the estimation of both individual and population parameters using experimental data sets for a population of *Arabidopsis thaliana* individuals. The ultimate motivation being to show how the different parameters of the GreenLab model vary among this population.

The application of these methods requires efficient computing tools: this motivated the design of a whole new platform for statistical inference written using the Julia language that is both fast and easy to use. This platform is described in detail in Chapter 5 together with how the estimation techniques fit in and perform. It therefore allowed to consider practical applications of the theoretical issues presented so far.

In particular, the estimation techniques considered in the case of a single individual are applied in Chapter 6. We notably demonstrate and compare the use of different sequential Monte Carlo methods for the joint estimation of parameters and states within a Markov chain Monte Carlo procedure. It is also the occasion to present some of the concrete applications of interest in plant science. The first case study involves the Green-Lab model for *Arabidopsis thaliana* which is used for a first model calibration and a comparison between a frequentist approach (based on generalized least squares) and a Bayesian one (using a Markov chain Monte Carlo method) for parameter estimation in terms of the estimates they provide and the related uncertainties. The second case study deals with data assimilation, a common application in plant science and agriculture, using the LNAS model for wheat, for which yield prediction are of primary importance. A sequential Monte Carlo method, the regularized particle filter, is used for an on-the-fly estimation of the parameters and the hidden states. It is notably shown that data assimilation using the regularized particle filter provides much better predictions than a simple uncertainty analysis. Several scenarios are studied, notably how the number of observations available, the measurement error, the prior distributions or the number of estimated parameters influence the quality of the predictions. Finally, the last case study relates to a precise joint estimation of the hidden states for the LNAS model for sugar beet, comprising both process and observation noise, using a particle Markov chain Monte Carlo method and how it paves the way for the estimation of the noise parameters. We notably study how the choice of the sequential Monte Carlo algorithm within the particle Markov chain Monte Carlo method influences the quality of the estimates and how a trade-off with computing time can be found.

The image analysis algorithm that is used to obtain experimental data from images of *Arabidopsis thaliana* is then introduced in Chapter 7. The Phenoscope, a high-throughput phenotyping platform, allows us to obtain large amounts of data that are necessary for parameter estimation in hierarchical models. Its ability to automatically compute the projected area of the whole plant is rather limited to explain the whole history of a plant's functioning and accurately differentiate between individuals, which is why an image analysis procedure had to be developed in order to obtain a finer segmentation of each plant at the scale of the leaf during the entire plant growth. This two-step segmentation-tracking procedure takes advantage of a careful analysis of data specific to *Arabidopsis thaliana*, showing very promising results.

Finally, the retrieval of sufficient data allows for the calibration of Bayesian hierarchical models in Chapter 8. The estimation of the individual and population parameters is first studied in the case of simulated data to investigate different issues concerning the convergence of our algorithm. Notably, we study how the number of individuals in the population, the measurement error, the choice of the prior distributions or the number of estimated parameters influence the estimation of the different population parameters. The case of real data is then investigated. This last chapter therefore takes advantage of the different aspects of this work by combining modelling, Bayesian estimation for hierarchical models, high performance computing and real

data obtained thanks to image analysis in order to achieve the assessment of the intergenotypic variability in a population of *Arabidopsis thaliana* of different genotypes.

Chapter 1

Mathematical framework

PLENTY OF REAL-WORLD APPLICATIONS require the prediction of some quantities, which can be achieved by first defining a system characterized by state variables and second designing a model using equations that aim at representing the underlying physical processes. These equations usually involve defining many parameters that specify these equations to the system under study, and a crucial point is therefore to have good estimates of these parameters' values. Real data, also referred to as experimental data, is thus utilized and compared to the output of the designed model for parameter inference. A very common mathematical framework for this task is that of general state space models (SSMs) [Doucet et al., 2001]. SSMs were first introduced for finite state spaces in the 1960s, jointly with the development of the theory of Gaussian linear models. Popularized by Kalman [1960] in his work on the so-called Kalman filter, SSMs can describe either deterministic or stochastic dynamical systems. This mathematical framework, being very general, has been used extensively in very diverse fields such as ecological population dynamics [Newman et al., 2009], [Hosack et al., 2012], biochemistry [Golightly and Wilkinson, 2008], [Golightly and Wilkinson, 2011], finance [Elliott and Hyndman, 2007], [Mamon and Elliott, 2014], neuroscience [Paninski et al., 2010], [Chen and Brown, 2013] and plant science [Yan et al., 2004], [Cournède et al., 2013], just to name a few recent examples.

This chapter is devoted to the introduction of the mathematical models that will be used in the rest of this thesis. The mathematical framework of general SSMs is first described in Section 1.1, notably the main system of equations characterizing them. The link with hidden Markov models, which is nothing more than another formulation of the same model, is briefly discussed. Emphasis is put upon the structure of the observations of a system because of its importance in the estimation procedure and its potentially scarce and non-homogeneous nature. Then the choice of the observation models as well as possible extensions of models are presented. A critical aspect of this mathematical framework is that of the definition and the calculation of the transition and observations probability density functions, as these functions are involved in most of the estimation algorithms of Chapter 3 and constitute one of the angular stones of the design of the computing platform introduced in Chapter 5 – this point being notably discussed in detail in Section 5.5. Last but not

least, Section 1.3 is dedicated to the introduction of hierarchical models that will further be used for the analysis of the genotypic variability of a population of plants.

1.1 General state space models

Usually the term state space models refers to linear state space models, whereas the term general state space models is used for their nonlinear equivalent. For convenience, general state space models will simply be referred to as state space models in what follows. The state space of such models can be either discrete or continuous. The SSMs considered throughout this document will be continuous-valued and discrete in time. If the equations modelling a system are continuous in time, they first need to be discretized for numerical simulation.

1.1.1 Main equations

Starting from initial state variables (initial conditions) at time step $n = 0$, the system variables are updated at each time step $n \in \llbracket 1, T \rrbracket$ where T denotes the last time step of the simulation. For plant models, this usually means that variables are updated daily, as is the case in the Log-Normal Allocation and Senescence model for *Beta vulgaris* (Section 2.2) or wheat (Section 2.3), or hourly, for instance in the GreenLab model for *Arabidopsis thaliana* (Section 2.4). At each time step n , a system of two equations summarizes the evolution of the state variables and the observations of the system respectively. In their most general form, they read:

$$\begin{cases} x_{n+1} &= f_n(x_n, u_n, \theta, \eta_n), \\ y_n &= g_n(x_n, \theta, \xi_n), \end{cases} \quad (1.1)$$

where the evolution of the system is considered between the initial time $n = 0$ and the final time $n = T \in \mathbb{N}^*$, and where at time step $n \in \llbracket 0, T \rrbracket$:

- $x_n \in \mathbb{R}^{d_x}$ represents the state variables of the model, x_0 therefore denotes the initial state of the system. Since these variables are a priori not accessible to measurement, they are also called hidden states;
- $y_n \in \mathbb{R}^{d_{y_n}}$ represents the observations on the system. It is worth noting here that the dimension of the vector of observations depends on the time step n , this will be detailed in Section 1.1.3;
- $u_n \in \mathbb{R}^{d_u}$ represents the external variables influencing the system, for example control variables: in plant science, these are typically environmental conditions in which the system evolves, such as temperature, radiation, water resources or nutrients;
- $\theta \in \mathbb{R}^{d_\theta}$ represents the functional parameters, which intervene in the functional equations and can either have a systemic meaning – they can originate from biology, for instance – or simply be parameters of empirical descriptive functions used because they constitute a convenient and sensible way of modelling a physical process;

- $\eta_n \in \mathbb{R}^{d_\eta}$ are process noises – or equivalently modelling noises – and are the values taken at each time step by a random vector representing the stochastic factors that aim to account for either possible model limitations or imperfections;
- $\xi_n \in \mathbb{R}^{d_\xi}$ are observation noises: since the observed data is most of the time measured with some uncertainty, observation noises are the values taken at each time step by the random vector defined so as to reproduce this measurement error;
- f_n is the transition function, it drives the evolution of the state variables from one time step to another;
- g_n specifies how the system is observed and what the observations are in terms of the hidden states.

The fact that both the transition and observation functions are allowed to depend on the time index n is referred to as non-homogeneous transitions. This is often the case in plant growth models since the plant has different evolution stages (which mostly depends on the thermal time) where its behaviour can be drastically different.

1.1.2 Hidden Markov models

In their stochastic formulation with random vectors defining the process and observation noises, SSMs are equivalent to hidden Markov models (HMMs) [Rabiner, 1989] where x_n represents the hidden states, y_n the observations and where:

$$\begin{cases} x_0 \sim p(x_0) & \text{is the initial distribution,} \\ x_{n+1} \sim p(x_{n+1}|\theta, x_n) & \text{is the transition distribution,} \\ y_n \sim p(y_n|\theta, x_n) & \text{is the observation distribution.} \end{cases} \quad (1.2)$$

The transition and observation distributions represented above by conditional probability density functions can be rewritten using the process and observation noises as will be detailed in Section 1.2. It has to be noted that each one of these distributions can be taken as a Dirac distribution. An important case is that of a model without process noise, in which case Equation 5.2 reduces to:

$$\begin{cases} x_{n+1} = f_n(x_n, u_n, \theta), \\ y_n = g_n(x_n, \theta, \xi_n). \end{cases} \quad (1.3)$$

In this case, the transition distribution is equivalent to a Dirac distribution so that $p(x_{n+1}|\theta, x_n)dx_{n+1}$ is replaced by $\delta_{f_n(x_n, u_n, \theta)}(dx_{n+1})$ and for given parameters θ , initial state x_0 and external variables $(u_n)_{n \in \llbracket 1, T \rrbracket}$, all state variables at all time steps $(x_n)_{n \in \llbracket 1, T \rrbracket}$ are deterministically defined. A less common case would be that of a model without observation noise, where $p(y_n|\theta, x_n)dy_n$ would be replaced by $\delta_{g_n(x_n, \theta)}(dy_n)$, which would be such that every measurement contains perfect information on the system. This scenario, however, finds very few practical applications as most models deal with the measurement of continuous valued variables and thus necessarily involves some uncertainty.

1.1.3 Structure of the observations

At a given time step n , the observed variables can be of different nature: integers (the number of phytomers of a plant), real numbers (the biomass of a leaf compartment), vectors (the areas of the different individual leaves) or even matrices in some cases. Most of the time and for practical reasons in real world applications, observations do not come up at every time step and the data available at a given time might not be the same at another. Let us consider the example of a plant model for, say, sugar beet: the biomass of the leaves might be available on days 10, 20 and 35 whereas the biomass of the roots might be available on days 12, 20, 33. For an organ-scale plant model where observations on the different organs are obtained via image analysis, as is the case of leaf areas, the latter might not be available on the same days because an algorithm deemed its classification confidence to be insufficient. The size and content of the observations might therefore not be the same through time, which poses no problem whatsoever as long as one knows what variables are observed at what time. The merged experimental timeline, representing time steps at which any experimental data is available, will be denoted by:

$$\mathcal{O} = (t_k)_{k \in \llbracket 1:O \rrbracket} \in \mathbb{N}^O \text{ with } 1 \leq t_k < t_{k+1} \leq T \text{ for } k \in \llbracket 1, O \rrbracket, \quad (1.4)$$

where $O \geq 1$ is the total number of experimental time steps. More formally, the observations at a given time step n can be seen as a dictionary where keys would be the different observed variables through the experiment and the values would be the actual observations of the corresponding variables. The operation of converting a dictionary of observations into a vector and its inverse will be presented in more detail in Section 5.4 when discussing the practical implementation of the storage of observations using the computing platform. If ℓ_n denotes the total number of variables observed at time step n , $v_n = (v_n^\ell)_{\ell \in \llbracket 1, \ell_n \rrbracket}$ said variables and y_n^ℓ denotes the observation relative to variable v_n^ℓ , then the vector of observations at time n can be defined as the concatenation:

$$y_n = (y_n^\ell)_{\ell \in \llbracket 1, \ell_n \rrbracket} \in \mathbb{R}^{d_{y_n}}. \quad (1.5)$$

Equivalently, all the observation vectors at each time in the experimental timeline can be concatenated and the following notations are introduced:

$$\begin{cases} x_{1:T} &= (x_n)_{n \in \llbracket 1:T \rrbracket}, \\ y_{1 \rightarrow T} &= (y_{t_k})_{k \in \llbracket 1:O \rrbracket}. \end{cases} \quad (1.6)$$

Some algorithms (least squares algorithms for example) require to deal with vector observations and it is therefore important to be able to work with observations of such a nature. In fact, $y_{1 \rightarrow T}$ could also be seen as a matrix of observations where each row would correspond to a time step and each column to a type of observation, and elements of this matrix could be missing (since all variables are not observed at all times). All the information about what variables are observed at what time is actually stored in the sequence of observation functions $(g_n)_{n \in \llbracket 1, T \rrbracket}$. More details on how this is implemented in the computing platform can be found in Sections 5.2.3 and 5.3 with examples. For now, one can assume that at each time step n , y_n is a vector of observations, that are not necessarily the same at different times, and that one knows what these vector observations contain and how to exploit them.

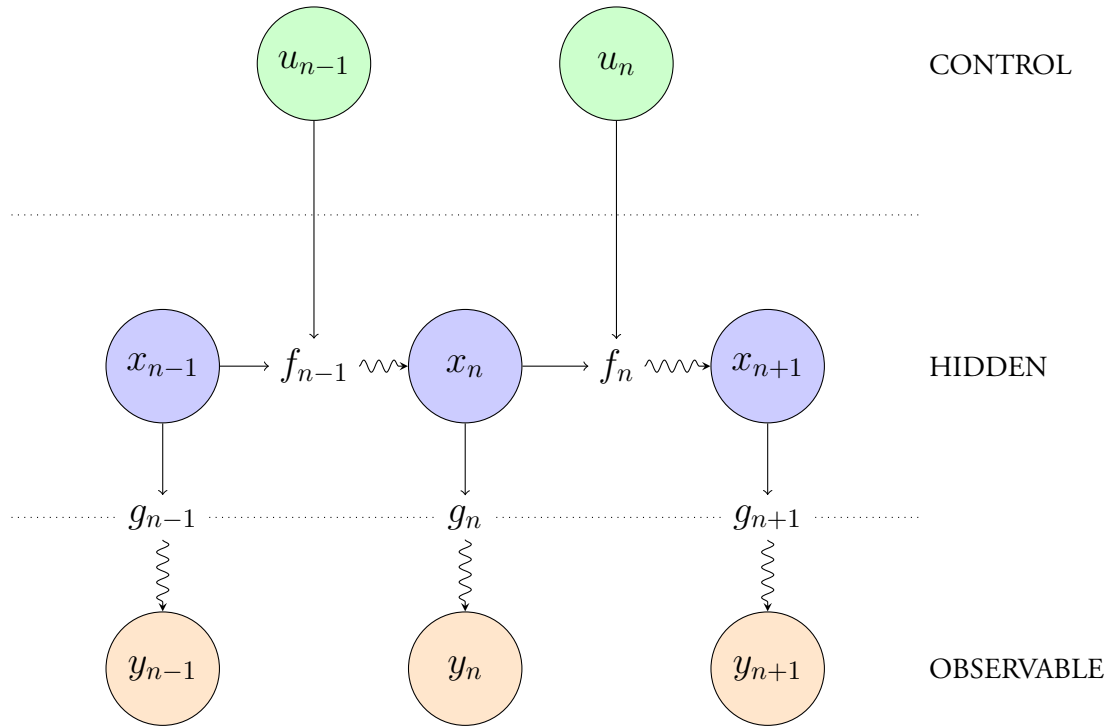


Figure 1.1: Representation of a general state space model. Wavy curves in the hidden layer (respectively observable layer) represent the randomness introduced by the process noise (respectively observation noise). The external (or control) variables u_n are known at every time step, the hidden states x_n are unknown in real experiments but are accessible when the corresponding model is simulated, and the observations y_n are both known in real experiments and can also be simulated provided that an observation model error has been defined.

Process and observation noises are of stochastic nature, and the underlying parameters constant throughout a model simulation are typically the mean or standard deviation of the statistical distribution from which they are sampled. We distinguish the random variables η and ξ , from their realizations at a given time η_n and ξ_n such that:

$$\begin{cases} \eta & : \Omega^\eta \rightarrow \mathbb{R}^{d_\eta} \\ \omega^\eta & \rightarrow \eta_n \equiv \eta(\omega^\eta) \end{cases} \quad (1.7)$$

and:

$$\begin{cases} \xi & : \Omega^\xi \rightarrow \mathbb{R}^{d_\xi} \\ \omega^\xi & \rightarrow \xi_n \equiv \xi(\omega^\xi) \end{cases} \quad (1.8)$$

where Ω^η and Ω^ξ are appropriate sample spaces. A simulation of the model can therefore be summarized as:

$$y_{1 \rightarrow T} = M(x_0, u, \theta, \eta, \xi). \quad (1.9)$$

where the model M contains the information on the sequences of transition and observation functions $(f_n)_{n \in \llbracket 1, T \rrbracket}$ and $(g_n)_{n \in \llbracket 1, T \rrbracket}$. Sometimes, one might abbreviate the output of the model as $y \doteq y_{1 \rightarrow T}$. It is worth noting that the use of such a stochastic formulation for plants is not very common and dates back to less than 20 years [Makowski et al., 2004], [Chen and Cournède, 2012], [Trezvas and Cournède, 2013]. A graphical representation of a general SSM is shown on Figure 1.1.

1.1.4 Observation model

Considering a plant model, the biomass produced by the whole plant on a given day is not directly measurable, it is thus considered to be a hidden state, whereas the biomass of green leaves on the same day is a measurable data, making it an observation. As previously said, a measure on a system is very often inexact, and this is almost always the case when dealing with continuous-valued models. For instance, measuring biomass can be done using either destructive or non-destructive methods: in the first case, biomass is removed from the plant and weighed while the second case is based on digital image analysis. The cutting point, the weighing process, or imperfections of algorithms are as many factors implying that some measurement error is made. The true value of the biomass is therefore never exactly measured, and a distinction must be made between the hidden state of the biomass, which remains unknown, and its corresponding observation. A given measurable variable will frequently be defined both as a hidden state and as an observation. In the case of a biomass denoted by q , this would translate into:

$$q_n \in x_n \text{ and } \tilde{q}_n \in y_n. \quad (1.10)$$

How these two values are related constitute a model for the measurement error. A standard approach is to consider that, on average, the hidden state is measured with some white noise following a normal distribution. If the noise is proportional to the value of the hidden state – for instance if the greater the biomass, the greater the measurement error – one might want to consider a multiplicative noise:

$$\tilde{q}_n = q_n (1 + \xi_n), \text{ with } \xi_n \sim \mathcal{N}(0, (\sigma^q)^2) \text{ and } \sigma^q > 0, \quad (1.11)$$

so that:

$$\tilde{q}_n \sim \mathcal{N}\left(q_n, (\sigma^q q_n)^2\right), \quad (1.12)$$

whereas if the noise does not depend on the value of the hidden state, one might want to consider an additive noise:

$$\tilde{q}_n = q_n + \xi_n, \text{ with } \xi_n \sim \mathcal{N}\left(0, (\sigma^q)^2\right) \text{ and } \sigma^q > 0, \quad (1.13)$$

so that:

$$\tilde{q}_n \sim \mathcal{N}\left(q_n, (\sigma^q)^2\right). \quad (1.14)$$

Obviously, there can be situations where measured values are always underestimated or overestimated, in which case these two measurement error models might not be relevant anymore. In the rest of this thesis, the values that will be measured will be either biomasses or leaf areas, and it is therefore assumed that multiplicative normal noises are the most adapted to such situations. However, more observation models have been considered as will be emphasized in Chapter 5 when discussing the computing platform.

1.1.5 Extensions

The popularity of such models has generated many extensions. A common one concerns k -th order Markov process, where $k > 1$: in this case the hidden state x_{n+1} does not depend only on x_n but on $(x_{n-j+1})_{j \in [1, k]}$.

This can happen in plant growth models – such as in the STICS model [Brisson et al., 1998] for instance – even though from a mathematical point of view it is always possible to redefine the system state as $x'_n = (x_{n-j+1})_{j \in \llbracket 1, k \rrbracket}$ to define a standard SSM again.

In Markov-switching models (also called Markov jump systems), at time step n the observation y_n depends not only on the hidden state x_n but also on the previous observation y_{n-1} (and possibly on even older observations) [Cappé et al., 2005]. The sequence of observations $(y_n)_{n \in \llbracket 1, T \rrbracket}$ can therefore be seen, conditional on the sequence of hidden states $(x_n)_{n \in \llbracket 1, T \rrbracket}$, as a non-homogeneous Markov chain. Although this kind of model has a lot in common with standard HMMs, their statistical analysis is much more complicated because the observed sequence $(y_n)_{n \in \llbracket 1, T \rrbracket}$ is not directly related to that of the unobservable one $(x_n)_{n \in \llbracket 1, T \rrbracket}$. Although Markov-switching models are not considered within this thesis, two reasons behind their potential uses must be mentioned. First, when measurements are performed on a plant or in a field, the observer might be influenced by the previous obtained results: if the biomass of an organ is surprisingly much lower than that of a previous time, one might be tempted to correct the current measurement upwards. Second, the image analysis algorithm used to estimate the individual leaf areas (see Chapter 7) does so by taking into account the whole history of a given leaf. For the image of a given day, the decision to classify a given segment (i.e. a set of connected pixels that was considered to represent a leaf occurrence) – whose area contains some observation noise intrinsic to the segmentation algorithm – as belonging to a particular leaf of the plant depends on the whole history of the said leaf, hence on previous observations. Nevertheless, this effect is considered to be of minimal importance in the present case: when it is uncertain whether a segment belongs to a leaf, it is considered as not being observed. Classification-related errors (and time-induced) will therefore be minimal in the pool of actually observed data collected.

1.2 Generic probability distributions

As will be detailed in Chapter 3, parameter and state estimation algorithms require the use of the transition and observation probability density functions (pdfs). The algorithms should be designed so as to be easily used with different models. It therefore requires the calculation of the transition pdf $p(x_{n+1} | \theta, x_n)$ and the observation pdf $p(y_n | \theta, x_n)$. What is more, the pdf of all the observations conditional to the parameters and the hidden states $p(y_{1:T} | \theta, x_{1:T})$ can then easily be deduced from the observation pdf. For this aim, a generic expression of the latter is derived and, as explained in Chapter 5, this will allow to automatically compute their values provided that models are written using a predefined template.

1.2.1 Transition probability density function

In the models considered, it is always possible to arrange the state variables by their order of computation at a given time step. The hidden state is therefore decomposed into $(x_n^j)_{j \in \llbracket 1, d_x \rrbracket}$ where for all $j \in \llbracket 1, d_x - 1 \rrbracket$, x_n^j is computed before x_n^{j+1} . In particular, a variable x_{n+1}^j can depend in practice on all variables computed before, $(x_{n+1}^k)_{k \in \llbracket 1, j-1 \rrbracket}$ – which have been expressed as functions of x_n themselves – without breaking the

dependence on only x_n from a mathematical point of view. The transition pdf can therefore be expressed in a hierarchical fashion:

$$p(x_{n+1}|\theta, x_n) = \prod_{j \in \llbracket 1, d_x \rrbracket} p(x_{n+1}^j|\theta, x_n, x_{n+1}^{1:j-1}). \quad (1.15)$$

In the absence of process noise, the dynamics of the system is entirely deterministic, as a consequence $p(x_{n+1}^j|\theta, x_n, x_{n+1}^{1:j-1})dx_{n+1}^j = \delta_{m_{n+1}^j(x_n, x_{n+1}^{1:j-1}, \theta)}(dx_{n+1}^j)$ where $m_{n+1}^j(x_n, x_{n+1}^{1:j-1}, \theta)$ prescribes how x_{n+1}^j is computed within the model. In a model without process noise, the transition pdf therefore becomes such that:

$$p(x_{n+1}|\theta, x_n)dx_{n+1} = \prod_{j \in \llbracket 1, d_x \rrbracket} \delta_{m_{n+1}^j(x_n, x_{n+1}^{1:j-1}, \theta)}(dx_{n+1}^j). \quad (1.16)$$

In particular, one can choose to introduce as many intermediary variables in the hidden state x_n without fundamentally changing the model formulation in terms of transition distribution. In view of this remark, when some process noise is involved, the only non-Dirac terms in Equation 1.15 are those affected by the process noise and the corresponding random vector can also be arranged by order of use in the model $\eta_n = (\eta_n^j)_{j \in \llbracket 1, d_\eta \rrbracket}$. For now, we assume that all the process noises of the model are unidimensional.

Let $m_\eta : \llbracket 1, d_\eta \rrbracket \rightarrow \llbracket 1, d_x \rrbracket$ be an application such that $m_\eta(\llbracket 1, d_\eta \rrbracket)$ represents the sets of indices of the state variables on which are set the process noises, i.e. there exists some function ϕ_j such that:

$$x_n^{m_\eta(j)+1} = \phi_j(x_n^{m_\eta(j)}, \eta_n^j), \text{ for } j \in \llbracket 1, d_\eta \rrbracket. \quad (1.17)$$

For a particular process noise of index j and in the case of an additive normal noise, this would translate into:

$$x_n^{m_\eta(j)+1} = x_n^{m_\eta(j)} + \eta_n^j \text{ with } \eta_n^j \sim \mathcal{N}(0, (\sigma^j)^2). \quad (1.18)$$

It is possible to express the transition pdf by choosing only the d_η state variables on which are set the process noises:

$$p(x_{n+1}|\theta, x_n)dx_{n+1} = \prod_{j=1}^{d_\eta} p(x_{n+1}^{m_\eta(j)+1}|\theta, x_{n+1}^{m_\eta(j)}) dx_{n+1}^{m_\eta(j)+1} \times \prod_{j \notin m_\eta(\llbracket 1, d_\eta \rrbracket)} \delta_{m_{n+1}^j(x_n, x_{n+1}^{1:j-1}, \theta)}(dx_{n+1}^j)$$

where we recall that $m_{n+1}^j(x_n, x_{n+1}^{1:j-1}, \theta)$ is the value computed for the state variable x_{n+1}^j within the model considered. In what follows, since the variables that are deterministic functions of the stochastic variables are computed directly by closure relationships in the simulation program, the Dirac distributions take values 1, hence will be omitted in the following. In particular, we can restrain ourselves to the noised variables and replace the transition pdf by:

$$p(x_{n+1}^{m_\eta(j)+1}, \dots, x_{n+1}^{m_\eta(d_\eta)+1}|\theta, x_n) = p(\eta_n|\theta) = \prod_{j=1}^{d_\eta} p(x_{n+1}^{m_\eta(j)+1}|\theta, x_{n+1}^{m_\eta(j)}). \quad (1.19)$$

For the sake of simplicity and with a slight abuse of notation, in what follows we will denote $p(x_{n+1}|\theta, x_n) = p(x_{n+1}^{m_\eta(j)+1}, \dots, x_{n+1}^{m_\eta(d_\eta)+1}|\theta, x_n)$, implicitly assuming Dirac distributions for the variables computed in a deterministic fashion by model closure. This formulation generalizes in fact very well to multidimensional noises. The decomposition of the state variable $x_n = (x_n^j)_{j \in \llbracket 1, d_x \rrbracket}$ can be performed in such a way that x_n^j can be a multidimensional quantity such as a vector or a matrix and so that $d_x^j \leq d_x$.

Similarly, $\eta_n = (\eta_n^j)_{j \in \llbracket 1, d'_\eta \rrbracket}$ can contain multidimensional quantities. For instance, the random vector of process noises in the Kalman filter [Kalman, 1960] is drawn from a multivariate normal distribution with non-zero off-diagonal components (i.e. this cannot be simplified to the use of unidimensional process noises), which for some j would translate into:

$$x_n^{m_\eta(j)+1} = x_n^{m_\eta(j)} + \eta_n^j \text{ with } \eta_n^j \sim \mathcal{N}(0, \Sigma) \quad (1.20)$$

and where Σ can potentially be a full matrix.

1.2.2 Observation probability density function

Equivalently, one can decompose the observation noises as $\xi_n = (\xi_n^j)_{j \in \llbracket 1, d'_\xi \rrbracket}$ and define an application $m_\xi : \llbracket 1, d'_\xi \rrbracket \rightarrow \llbracket 1, d'_x \rrbracket$ such that $m_\xi(\llbracket 1, d'_\xi \rrbracket)$ represents the set of indices of the state variables on which are set the observation noises, i.e. there exists some function ψ_j such that:

$$y_n^j = \psi_j(x_n^{m_\xi(j)}, \xi_n^j), \text{ for } j \in \llbracket 1, d'_\xi \rrbracket. \quad (1.21)$$

Again, for a particular observation noise of index j and in the case of a unidimensional multiplicative noise, this would translate into:

$$y_n^j = x_n^{m_\xi(j)} (1 + \xi_n^j) \text{ with } \xi_n^j \sim \mathcal{N}(0, (\sigma^j)^2). \quad (1.22)$$

It can also happen that observations on the system require multidimensional noises. This is notably the case in [Baey et al., 2016] where the biomasses of the different organs are observed with correlation, this would mean that:

$$y_n^j = x_n^{m_\xi(j)} \cdot (1 + \xi_n^j) \text{ with } \xi_n^j \sim \mathcal{N}(0, \Sigma). \quad (1.23)$$

where it is understood that in the case of multidimensional noises, operations on vectors such as \cdot or $/$ are performed element-wise, and with Σ having some of its off-diagonal components non-zero. The observation pdf can be expressed in the same way as for the transition pdf:

$$p(y_n | \theta, x_n) = p(\xi_n | \theta) = \prod_{j=1}^{d'_\xi} p(y_n^j | x_n^{m_\xi(j)}) \quad (1.24)$$

where the product runs on all indices j for which experimental data y_n^j is available and, again, possibly contains multidimensional noises. The generic expression of the pdf of the observations conditional to the parameters and the hidden states $p(y_{1 \rightarrow T} | \theta, x_{1:T})$ naturally follows from that of the observation pdf since:

$$p(y_{1 \rightarrow T} | \theta, x_{1:T}) = \prod_{k=1}^O p(y_{t_k} | \theta, x_{t_k}) = \prod_{k=1}^O \prod_{j=1}^{d'_\xi} p(y_{t_k}^j | \theta, x_{t_k}^{m_\xi(j)}). \quad (1.25)$$

Equation 1.19 makes it possible to compute the transition pdf as long as are specified the nature of the noises (are they additive or multiplicative, sampled from a normal, a log-normal or a uniform distribution?) and lists of labels corresponding to the parameters necessary to compute the values of these pdfs. Likewise, Equation 1.24 makes it possible to compute the observation pdf. The complete mechanism for the practical computation of such values of the transition and observation pdfs are described in detail in Section 5.5.

Examples of theoretical transition and observation pdfs are given in Chapter 2 in the case of two plant growth models for *Beta vulgaris* and *Arabidopsis thaliana*.

1.3 Population models

In this section, we first describe the reasons behind the use of the population approach and emphasize its importance in the context of genotypic variability. We then move on to introduce the two types of variability in the context of plants and introduce hierarchical models, statistical models that are well-suited for dealing with this dual variability for a population of plants, and finally show how the SSMs considered in Section 1.1 fit in the mathematical description of population models before describing the complete two-stage hierarchical models that will further be used for parameter inference in a population context.

1.3.1 Motivation

Most biological and physical phenomena observed within a set of different individuals exhibit variability: they might manifest similar global behaviour and dynamics but with some variations. This is of practical use in many fields such as biology, agronomy, econometrics, environmental and human sciences. For instance, in pharmacometrics, one needs to develop models where different patients would react differently to the same disease and the same drug. [Lavielle \[2014\]](#) gives the example of the effect of genetically modified corn on the health of rats.

As far as plants are concerned, there exists a strong genetic variability even within individuals of the same species, which notably allows for better resistance to diseases or bugs and provides stronger adaptation to a wide set of environmental conditions. [Brouwer et al. \[1993\]](#) showed that soil and crop growth micro-variability in the semi-arid tropics of West Africa contributes to increased yield in case of droughts since parts of the field, more resistant to water stress, could compensate for other parts performing poorly, meaning a satisfactory level of assured production. The genetic variability of switchgrass was studied by [Hopkins et al. \[1995\]](#) in order to develop improved populations, which highlighted the importance of genotype-environment interactions for traits such as forage yield at heading, vegetative in vitro dry matter digestibility and heading date. For maize, [Maiti et al. \[1996\]](#) showed that both genotypic variability and soil content were of highly significance for the resistance to drought and salinity at the seedling stage. This study also allowed to speculate about the effect of higher root growth under saline stress in some of the genotypes as a mechanism of resistance in maintaining osmo-regulation. [Isfan \[1993\]](#) suggested that the index of physiological efficiency of absorbed nitrogen may be used in order to identify the likely high yielding oat genotypes and those capable of exploiting nitrogen input most efficiently.

All these examples indicate both the importance of such a genetic variability and the necessity to integrate it within plant growth models. Similarly, in populations of plants in fields or forests, interindividual variability can result from differences in the local environmental conditions of individual plants. The first plant growth

models integrating this variability tried to simulate the growth of each individual with a competition index between plants [Fournier and Andrieu, 1999], [Cournède et al., 2008].

Most of the time in plant applications, repeated measurements on different individuals of a population are available, and a population approach is particularly suited to characterize and explain this kind of data. The mathematical model used therefore needs to incorporate a growth model that depicts the dynamics of the different state variables of the plant considered – biomasses mainly – and a statistical model that explains the variations of this typical dynamics between the different individuals. There are in fact two sources of variability to account for:

- the intraindividual variability, which refers to how the state of a single individual might vary, because of random processes and measurement errors. This kind of variability was introduced in Section 1.1 under the form of process and observation noises;
- the interindividual variability, which arises because of differences between the genotypes or environments of different individuals. This corresponds to the variability of the different individuals' curves around a mean population curve.

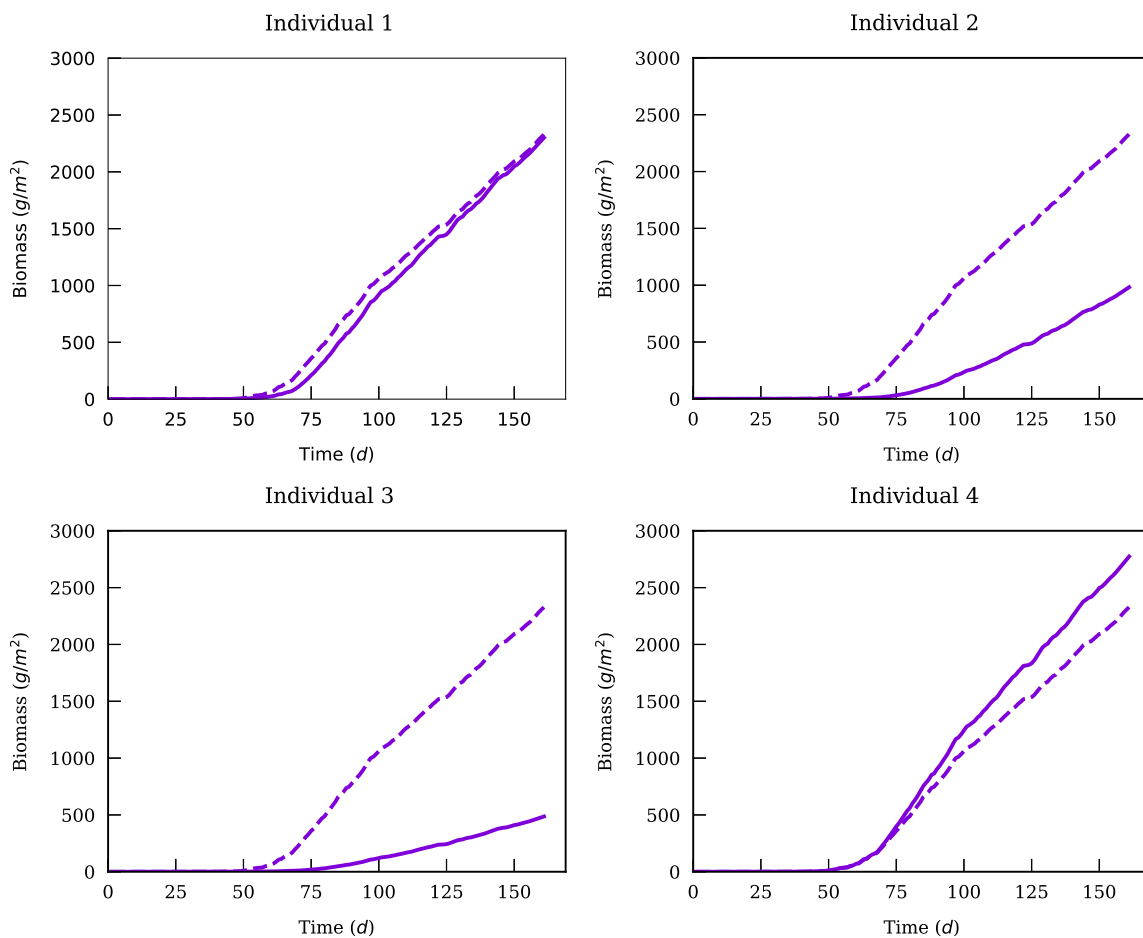


Figure 1.2: Yield curves of four different individuals. The individual curves are displayed with solid lines and the mean curve (similar for all graphs) is displayed with dotted line.

1.3.2 Hierarchical models

Hierarchical models provide a very suitable way of modelling this dual variability. This type of statistical models have long been used in pharmacokinetics, epidemiology or ecology, although their use in dynamic plant growth models is rather recent [Baey et al., 2013]. Broadly speaking, they are statistical models of parameters that vary at several levels. First introduced in the context of linear regression, multilevel models then consisted in doing a regression in which the parameters were given a probability model, and this second-level model had parameters of its own, called hyperparameters, and all these parameters were inferred from the same data.

Many names can be found for such models: multilevel models may be the more common, nested data models, random effects models (mixed effects models when some parameters are random and some are fixed in the population) are also widely used. The hierarchical term seemed the most appropriate to our case since a hierarchy is clearly established between the different stages of plant growth modelling where individual parameters are first derived from a given probability distribution describing the population, and these individual parameters then drive the growth of a given plant via a noisy nonlinear SSM.

The two most simple and direct methods of parameter estimation in a population are known as complete pooling and no pooling [Davidian and Giltinan, 1993]:

- in complete pooling, differences between individuals are ignored, all are treated equally and the data coming from different individuals is used for the estimation of parameters supposed to represent all individuals. Disregarding the diversity of the sources for the data represents a significant oversimplification, not only will it fail to provide accurate predictions but it also misses variations in the population, which is the main objective of many studies;
- in no-pooling, data coming from different sources are analyzed separately. This procedure ignores the information related to the diversity of the individuals in the population and can lead to unsatisfactorily variable inferences for the model parameters, in particular when little data is available. It must be noted, though, that individual estimates can still be of use for the initialization of prior of hyperparameters in a Bayesian estimation procedure as will be described in Section 8.1.2.

Hierarchical modelling manages to combine the information provided by different individuals and overcome the limitations introduced by these two simplistic methods.

The first stage of the model corresponds to the intraindividual variability and aims to explain how an individual evolves given a set of parameters supposed to represent it. The dynamics of each individual i within the population is then described by the same parametric model with a different set of parameters θ_i . Generalized linear models are often used for convenient and straightforward inference, however, the highly nonlinear nature of plant growth models suggested to proceed otherwise, and this first stage will be represented by the dynamics of general SSMs described in Section 1.1. The typical dynamics of an individual is thus well described with a parametric model such as those presented in Chapter 2 and a mean population dynamics y is assumed to be obtained with a set of parameters θ . The second stage of the model corresponds to the

interindividual variability and depicts the common probability distribution of the different sets of parameters that control the time dynamics of the individuals. The interindividual variability of the curves around the typical population curve can therefore be explained by the parameter variability around the mean population parameters as can be observed on Figure 1.2 for the yield curves of 4 different individuals. The parameters of the model can be considered either fixed or random. The random parameters are assumed to follow a statistical distribution parameterized by the typical population parameters and individual parameters are therefore random variables following the same population distribution. In a Bayesian framework, a third stage is finally added for the prior distribution of the population parameters.

1.3.3 State space models for populations

For the sake of consistency with literature, the mathematical notations used in the population approach will slightly differ from the single individual case. A population is made up of N different individuals indexed by $i \in \llbracket 1, N \rrbracket$. If $\theta_i \in \mathbb{R}^{d_\theta}$ denotes the set of parameters for individual i , the state space Equations 1.1 become in their most general form:

$$\begin{cases} x_{i,n+1} &= f_n(x_{i,n}, u_{i,n}, \theta_i, \eta_{i,n}), \\ y_{i,n} &= g_n(x_{i,n}, \theta_i, \xi_{i,n}). \end{cases} \quad (1.26)$$

Hopefully, some simplifications can be made. In the population approach, the nature of variability is two-fold and can be assumed, for the sake of simplicity, to encompass randomness arising from both process and observation noises. The process noise $\eta_{n,i}$ will therefore be ignored. As all applications of this thesis will be done within controlled environments, the control variables $u_{i,n}$ will be the same for all individuals and can be omitted. The transition part of Equation 1.26 can thus be simplified to:

$$x_{i,n+1} = f_n(x_{i,n}, \theta_i) \quad (1.27)$$

which allows, by induction, to rewrite more simply for all n :

$$x_{i,n+1} = h_n(x_{i,0}, \theta_i), \quad (1.28)$$

or in an even simpler form by incorporating the initial state into the h_n function – or the parameter vector if it is unknown:

$$x_{i,n+1} = h_{i,n}(\theta_i). \quad (1.29)$$

1.3.4 Intraindividual variability

As previously mentioned, when dealing with population models in the rest of this thesis, it will always be assumed that all observed variables follow a multiplicative normal observation model, which means that the measurement error associated to a given state is proportional to the latter. Note however that this assumption is made without loss of generality from a methodological point of view. For each individual, there are n_i measurements indexed by $j \in \llbracket 1, n_i \rrbracket$ and y_{ij} therefore denotes the j -th measurement on individual i , although index j does not designate time per se. The observations for the different individuals need not

be at the same times. In the context of population models, the vector of all observations from $n = 1$ to $n = T$ for a given individual i is denoted by $y_i = (y_{ij})_{j \in \llbracket 1, n_i \rrbracket} \in \mathbb{R}^{n_i}$. Although the notations look similar, there is usually little confusion possible between the observations y_n at time n for a single individual and the whole vector of observations y_i from $n = 1$ to $n = T$ for a given individual i . Given the complexity of the observations in the case of the GreenLab model for *Arabidopsis thaliana* within a population approach, details about the conversion between the observations at each time and the concatenated vector of all observations for the whole simulation are provided in Section 2.4. Assuming that the hidden state corresponding to the j -th observation of the i -th individual is simulated and denoted by $h_{ij}(\theta_i)$, then the observation model considered implies that:

$$y_{ij} = h_{ij}(\theta_i) \times (1 + \xi_{ij}) \text{ with } \xi_{ij} \sim \mathcal{N}(0, \sigma^2). \quad (1.30)$$

More elaborate observation models specifying heterogeneous observation-related variances, such as the one proposed in [Duval et al., 2009] could also be considered. It is worth noting that the standard deviation associated to the observation noise depends neither on the individual nor on the observations within a given individual. With the standard notations of hierarchical models, this becomes:

$$y_{ij} \sim \mathcal{N}(h_{ij}(\theta_i), \sigma^2 h_{ij}(\theta_i)^2) \quad (1.31)$$

for $i \in \llbracket 1, N \rrbracket$ and $j \in \llbracket 1, n_i \rrbracket$. Another way of rewriting this model with the vector of all observations of the i -th individual y_i reads:

$$y_i = \begin{pmatrix} y_{i1} \\ \vdots \\ y_{in_i} \end{pmatrix} = \begin{pmatrix} h_{i1}(\theta_i) + \xi_{i1} h_{i1}(\theta_i) \\ \vdots \\ h_{in_i}(\theta_i) + \xi_{in_i} h_{in_i}(\theta_i) \end{pmatrix} \sim \mathcal{N}(h_i(\theta_i), \tau^{-1} \Omega_i) \quad (1.32)$$

where $h_i(\theta_i) \in \mathbb{R}^{n_i}$ is the vector of the hidden states corresponding to the experimental data for the i -th individual (given by the model), and $\tau = \sigma^{-2} \in \mathbb{R}^{*+}$ is called the precision. The reason for using the precision instead of the standard deviation will become apparent when prior distributions for the estimation of parameters in a Bayesian framework are discussed in Chapter 4. For the multiplicative observation model considered, it is straightforward that Ω_i reduces to:

$$\Omega_i = \text{diag}\{h_i(\theta_i)^2\}. \quad (1.33)$$

More complicated covariance matrices could be used in practice, but this is not relevant to the case considered thereafter: each measurement is considered independent of the others, whence the diagonal matrix.

1.3.5 Interindividual variability

As the first stage of the model describes the intraindividual variability, the second stage of the model deals with the interindividual variability and prescribes how the individual parameters θ_i are distributed within the population. One of the simplest yet most sensible way to do so is to consider that the individual parameters $\theta_i \in \mathbb{R}^{d_\theta}$ follow a normal distribution:

$$\theta_i \sim \mathcal{N}(\eta, \Sigma) \quad (1.34)$$

where $d_\theta \in \mathbb{N}^*$ is the number of parameters to be estimated within the model, $\eta \in \mathbb{R}^{d_\theta}$ is the population mean vector and $\Sigma \in \mathcal{M}_{d_\theta}(\mathbb{R})$ the population covariance matrix. The two-stage hierarchical model thus reads:

$$\begin{aligned} \text{First stage:} \quad y_i &\sim \mathcal{N}(h_i(\theta_i), \tau^{-1}\Omega_i), \\ \text{Second stage:} \quad \theta_i &\sim \mathcal{N}(\eta, \Sigma). \end{aligned} \tag{1.35}$$

Parameter inference therefore amounts to jointly estimate the individual and the population parameters $\theta = \{(\theta_i)_{1:N}, \eta, \Sigma, \tau\}$. In a Bayesian approach, the population parameters will be given appropriate prior distributions that will extend the model with another stage. As far as genotypic differentiation is concerned, the main objective is to obtain reliable estimates of the population parameters so as to be able to describe which possess a significant role within the population, notably through hypothesis tests about the nullity of their respective variances. These issues will be discussed in Chapter 4.

Chapter 2

Plant growth models

THERE HAS BEEN A GREAT DEAL OF MODELS proposed for plant growth since the 1970s, with various objectives such as yield forecast or a better understanding of biological processes underlying plant functioning and with different modelling scales, both in space and time. This chapter starts off by providing brief historical reminders on the birth of plant growth models in Section 2.1. After emphasizing their importance in a global agricultural and environmental context, we focus on the mechanistic models for plant growth and see how the structural and functional approaches first disjoint were reconciled within functional-structural plant models (FSPMs). Section 2.2 is then devoted to the presentation of the compartmental LNAS model for *Beta vulgaris*. Sections 2.3 and 2.4 are dedicated to two models that were specifically designed in the context of this thesis: the first one is a model for wheat which is inspired from the LNAS model for sugar beet, the second one is a new organ-scale model for *Arabidopsis thaliana* that will be used in the context of population models for genotypic differentiation. The main biological processes considered in each of these models are first described, their whole formulation in terms of the mathematical framework of general SSMS introduced in Chapter 1 as well as first graphs of their simulations – obtained using the computing platform presented in Chapter 5 – are then presented.

2.1 History

Plants contribute to many aspects of human society: they are not only sources of foods and beverages, oxygen and energy, but they are also used for clothes, in construction and medicine and of course, for scientific research, which makes them a field of study of primary importance. Over the past decades, the growing concern on environmental protection also put the focus on agriculture because of its diverse impacts on the environment and notably on climate change:

- agriculture indeed contributes to climate change through the emissions of greenhouse gases and the conversion of non agricultural land such as forests into agricultural land. It is estimated to contribute up to 25% to global annual emissions [Blanco et al., 2014];

- conversely, climate change affects agriculture as pest insects and plant diseases together with extreme environmental conditions such as droughts tend to become much more frequent.

The effects of agriculture on the environment extend far beyond this and are also related to issues such as water resources, chemical pollution or the preservation of biodiversity. It raises many practical questions, just as in any other scientific field, but maybe even more since it deals with food production:

- can we predict the yield of a particular species – such as wheat or sugar beet – in France for the next year by taking into account the past climate and different future climate scenarios?
- how can water supplies be optimized in terms of yield, financial cost and with water use constraints?
- what variety of a given species is best adapted to particular environmental conditions (humidity, temperature, soil content, etc.)?
- how different genotypes, in interaction with their environment, can lead to drastically different phenotypes and results in the field?

All these questions are of primary importance for farmers and governments. To address them properly, mathematical models allowing the quantitative prediction of the evolution of a plant in given conditions must be designed. Their simulations via a programming implementation should be easy and reflect as much as possible the plant behaviour through time.

Two main approaches currently coexist for such problems. The first and oldest is to design mechanistic models that take into account biological knowledge. Biological processes are modeled with equations and parameters, and the global dynamics of the plant is inferred from all these processes. Such models can be considered at various scales: some are designed at the cell level, some others at the field level. The other approach is much more recent, purely data-based and has to do with machine learning. If enough data (biomasses, soil content, temperature, etc.) can be collected from the field, adequate machine learning models (such as neural networks, support vector machine, etc.) can be trained thanks to all these data and further be used for prediction in new conditions. The drawback of such an approach is that it requires a lot of data rarely available nowadays in the field of agriculture because of the complexity to obtain data on a regular basis, the different formats used by different experimenters, etc. What is more, it does not allow to approximate complex biological phenomena whose interactions could give rise to new dynamics in different conditions.

The first plant growth models date back to the 1970s, at the intersection between botanic, agronomy and computer science [De Reffye et al., 2009]. The continual progress of computer science since then have allowed for ongoing greater refinement and complexity. The first attempts to formally describe plant models came from botanists Hallé and Oldeman who, in 1970, published *An essay on the architecture and dynamics of growth of tropical trees* [Hallé and Oldeman, 1970] integrating the botanic qualitative knowledge of vegetal architecture. From a botanic point of view, the architectural analysis of trees aimed at interpreting their global structure and understanding the morphological mechanisms that gave birth to them. They proposed a classification allowing to split all the species of trees into 23 categories with respect to the type of their growth and ramification or their morphological differentiation. However, this first step towards a structural

classification of the plants did not allow for a refined systemic description of the plant architecture, which is why [Édelin \[1977\]](#) and then [Barthélémy et al. \[1989\]](#) later defined the concept of architectural unit characterizing the elementary architecture of trees of a same species. Alongside, progress in computer science in the 1980s, with even more powerful capabilities, allowed to generate increasingly complex arborescent structures [[Smith, 1984](#)], [[Prusinkiewicz et al., 1988](#)], [[Prusinkiewicz and Lindenmayer, 1990](#)], [[Françon, 1991](#)]. At this point, the use of L-systems, introduced by [Lindenmayer \[1968\]](#), and their stochastic extensions [[Kurth, 1994](#)] allowed for more elaborate growth rules; however most of these plant growth models arose from computer science and lacked the inclusion of crucial biological knowledge. The AMAP (Atelier de Modélisation de l'Architecture des Plantes) models [[De Reffye et al., 1988](#)], [[De Reffye et al., 1991](#)] tried to overcome these limitations by setting a complete data analysis routine integrating the morphological and architectural observation of the plant, measurements of the growing plants and the calculation of parameters related to the functioning of meristems. Yet, despite a faithful representation of the plant's architecture, they did not allow to take into account interactions with its functioning.

Jointly to the development of these architectural models originating from botany, other process-based models stemming from agronomy began to appear in the 1980s. These were more concerned with, for instance, the estimation of the average biomass production per square meter in a field in given environmental conditions. In this kind of models, the architecture of the plant does not matter and the latter is considered as a set of compartments for the different organs: roots, stems, leaves, fruits. The radiation intercepted by the plant is usually inferred from the Beer–Lambert law, from which the biomass produced can be deduced. Such models were developed for several plants, among which maize [[Jones et al., 1986](#)]. Some of them are developed in a generic manner so that they can be tuned for many species. This is the case of PILOTE [[Mailhol et al., 1996](#)] and STICS [[Brisson et al., 1998](#)]. One of the main criticism addressed to these process-based models is that, as previously mentioned, they do not include the architecture of the plant. Many authors advocated for its integration in order to obtain increased performance because of the strong interactions existing between the structure and the functioning of the plant [[Le Roux et al., 2001](#)], [[Kurth, 1994](#)].

The reconciliation of these two approaches happened at the end of the 1990s, when models combining the structural approach of botany and the functioning approach of agronomy started to appear, the so-called functional-structural plant growth models. This combination was done either by extending architectural models by integrating the biological processes representative of the plant's functioning, or by refining the process-based models in order to take into account its architecture. The first approach for example saw the evolution of the AMAP model towards the integration of functioning [[De Reffye et al., 1997](#)], [[De Reffye and Houllier, 1997](#)] and further led to the design of the GreenLab model by [De Reffye and Hu \[2003\]](#). The second one gave rise to the creation of the LIGNUM model [[Perttunen et al., 1996](#)].

In this thesis, three models for *Beta vulgaris* (sugar beet), wheat and *Arabidopsis thaliana* will mostly be used. The Log-Normal Allocation and Senescence (LNAS) model for sugar beet is a compartment model. This means that it does not take into account the architecture of the plant and the values of the variables of interest (biomasses mainly) will be given per square meter. A similar model was developed for wheat as it was particularly suitable for practical applications, notably data assimilation. As far as *Arabidopsis thaliana* is

concerned, a model based on previous GreenLab models was developed for the purpose of this thesis. The motivation behind these choices is simple: the LNAS model is a simple compartment model used extensively in the Biomathematics team, it is simple in the sense that it does not involve too many parameters, allows for quite an easy calibration and has been proved to provide accurate predictions in real case scenarios; the GreenLab model for *Arabidopsis thaliana* was developed in order to take advantage of the data available thanks to the images provided by the Phenoscope platform so as to build a model taking into consideration the interactions between the plant's architecture and functioning and in order to highlight the genotypic differentiation of *Arabidopsis thaliana*. These two models will therefore be presented using the formalism introduced in Chapter 1.

2.2 Log-Normal Allocation and Senescence for *Beta vulgaris*

Beta vulgaris, most commonly known as sugar beet, is primarily cultivated for the production of sugar. It is also used for the production of alcohol and ethano fuel. Some figures are provided to highlight the importance of this plant. Worldwide, 4.5 million hectares of land are devoted to the culture of sugar beets and 270 million tons were produced in 2014. Since 1875, France has been the first producer of sugar beets with 370,000 hectares (2,1% of the country's agricultural land) and 37.8 million tons produced in 2014, which accounts for 14% of the world production.

Sugar beet is a biennial plant and consists of roots and a rosette of leaves. During the first year, it is in a vegetative phase where the establishment and the main production of the plant happens: after germination of the glomerules, the foliage grows, sugar is formed in the leaves and then stored in the roots. The roots can then be harvested in autumn. The second year constitutes the reproductive phase: stems grow and the plant draw sugar on its reserves in the roots to produce an inflorescence which yields, after flowering and wind pollination, to the production of seeds. In practice, only the sugar contained in the roots is easily extractable, and since sugar beets are cultivated mainly for their sugar content, they are usually not cultivated for more than a year.



Figure 2.1: Sugar beets

The Log-Normal Allocation and Senescence (LNAS) model for sugar beet was first introduced by [Cournède et al. \[2013\]](#) as a simplification of the GreenLab model [[Lemaire et al., 2009](#)]. The organs of the plant are not considered individually but as a single compartment, whence its denomination of a compartmental model. In the case of sugar beet for which it was first designed, only two compartments are considered: the leaves and the roots. The idea of modelling the plant as a set of compartments is easily transposed to other plants and this is what inspired the design of the LNAS model for wheat detailed in Section 2.3 where more compartments are considered.

There is no distinction between petioles and limbs and leaf biomass is considered as a whole. On day n , the biomass of the roots is denoted q_n^r and that of the foliage is $q_n^\ell = q_n^{g\ell} + q_n^{s\ell}$, where $q_n^{g\ell}$ is the biomass of green leaves and $q_n^{s\ell}$ is the biomass of senescent leaves. All biomasses are expressed in $\text{g} \cdot \text{m}^{-2}$. The model is discretized according to a daily time step and the environmental variables of day n such as the temperature t_n ($^\circ\text{C}$) and the photosynthetically active radiation r_n ($\text{MJ} \cdot \text{m}^{-2}$) are daily averages. To emphasize the variables on which are set process noises, deterministic variables are denoted with a superscript ^{det} whereas their stochastic equivalent are denoted with a superscript ^{sto}.

The growth of the plant is driven by thermal time τ_n , which represents the accumulated daily temperature above a certain threshold called the base temperature – taken as $t^b = 0$ $^\circ\text{C}$ in the case of sugar beet – since germination, corresponding to day $i = 1$, of the plant such that:

$$\int_0^n \max(0, t(s) - t^b) ds \approx \sum_{i=1}^n \max(0, t_i - t^b) = \tau_n. \quad (2.1)$$

The thermal time must be higher than a certain value τ^{init} for the plant to reach the emergence stage: as soon as $\tau_n > \tau^{\text{init}}$, the plant starts to intercept light and as a consequence to produce biomass through photosynthesis. The biomass produced on day n per unit surface area is denoted by q_n^{det} and is assumed to follow a Beer–Lambert law [[Marcelis et al., 1998](#)]:

$$q_n^{\text{det}} = r_n \mu (1 - \exp(-k q_n^{g\ell}/e)), \quad (2.2)$$

where μ ($\text{g} \cdot \text{MJ}^{-1}$) is the radiation use efficiency, k (dimensionless) is the Beer–Lambert extinction coefficient and e ($\text{m}^2 \cdot \text{g}^{-1}$) is the leaf mass per area and is defined as the ratio of dry mass to leaf area. Hence, $q_n^{g\ell}/e$ represents the area of the foliage and $1 - \exp(-k q_n^{g\ell}/e)$ the fraction of radiation intercepted by the latter. To account for some inaccuracies of the Beer–Lambert law and the difficulty of assessing the local plant environment, the variable production of biomass is rendered stochastic by multiplying its deterministic value by a multiplicative normal noise such that:

$$q_n^{\text{sto}} = q_n^{\text{det}} (1 + \eta_n^q), \quad (2.3)$$

where $\eta_n^q \sim \mathcal{N}(0, (\sigma^q)^2)$. Although this kind of noise could theoretically lead to, for instance, negative values for a produced biomass, this is of no practical concern for the cases of interest of this work. The biomass produced on day n is distributed between the foliage and root system compartments according to an empirical function γ whose deterministic value is given by:

$$\gamma_n^{\text{det}} = \gamma^0 + (\gamma^\ell - \gamma^0) F_{\log\mathcal{N}(\mu^a, \sigma^a)}(\tau_n), \quad (2.4)$$

where we recall that τ_n here is the thermal time of the plant since germination, $\gamma^0 \geq 0$ and $\gamma^\ell \leq 1$ are respectively the initial value and the limit of biomass allocated to the leaves when the thermal time goes to infinity and $F_{\log\mathcal{N}(\mu^a, \sigma^a)}$ is the cumulative distribution function (cdf) of a log-normal law parameterized by its median and its standard deviation:

$$F_{\log\mathcal{N}(\mu, \sigma)}(\tau) \doteq \frac{1}{2} \left(1 + \operatorname{erf} \left[\frac{\log(\tau/\mu)}{\sigma\sqrt{2}} \right] \right) \mathbb{1}(\tau \geq 0). \quad (2.5)$$

Such a parameterization with the median is more convenient and meaningful than with the mean: once the thermal time has reached μ_a , the majority of the biomass is allocated to the root compartment. A process noise for the allocation is introduced since this allocation strategy highly depends on environmental conditions and is known to be rather plastic. Once again, a multiplicative normal noise is chosen and:

$$\gamma_n^{\text{sto}} = \gamma_n^{\text{det}} (1 + \eta_n^\gamma), \quad (2.6)$$

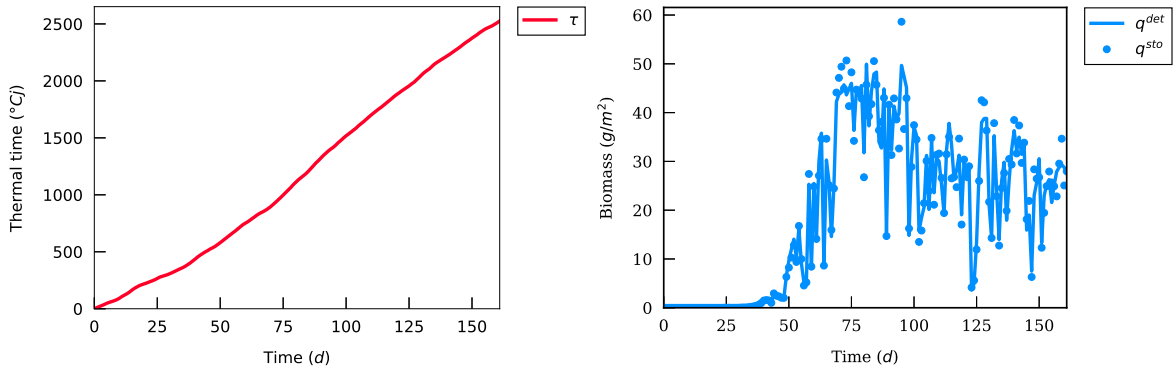


Figure 2.2: Left: example of thermal time dynamics for real data. Right: production of biomass, the continuous line represents the deterministic values predicted by the Beer–Lambert law, and the filled circles the corresponding noised values.

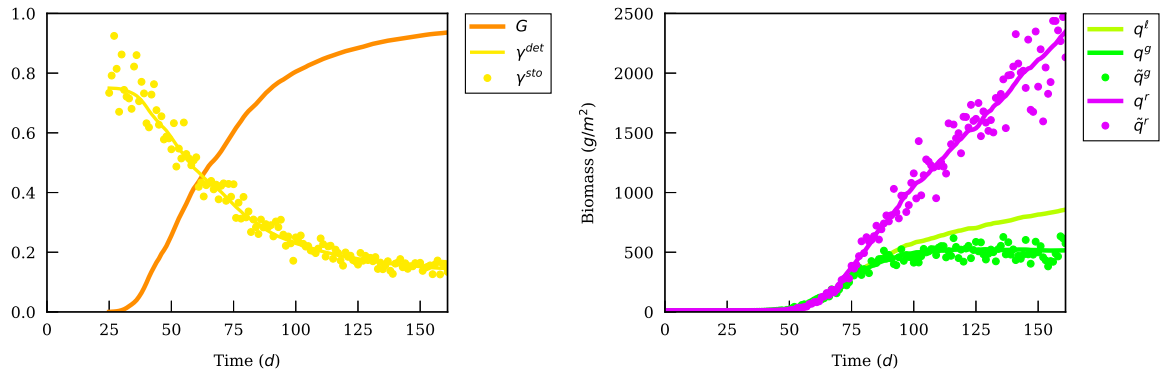


Figure 2.3: Left: allocation of the produced biomass to the different compartments; the orange curve depicts the log-normal cdf $F_{\log\mathcal{N}(\mu^a, \sigma^a)}$ as a function of the thermal time, the yellow line represents the deterministic value of the allocation variable γ^{det} and the filled circles its corresponding stochastic value γ^{sto} . Right: the different biomasses, the dark green curve depicts the total leaf biomass, the light green line and filled circles represents the deterministic value and the corresponding observation for the green leaves' biomass; the same goes for the purple curve and circles for the biomass of roots.

with $\eta_n^\gamma \sim \mathcal{N}(0, (\sigma^\gamma)^2)$. The biomass of the whole foliage increases every day: it receives the proportion $\gamma_n^{\text{sto}} \in [0, 1]$ of the biomass produced on day n :

$$q_n^\ell = q_{n-1}^\ell + \gamma_n^{\text{sto}} q_n^{\text{sto}}. \quad (2.7)$$

The biomass of senescent leaves is calculated as a proportion $\rho^s = F_{\log\mathcal{N}(\mu^s, \sigma^s)}(\tau_n - \tau^s)$ of the foliage biomass,

$$q_n^{s\ell} = \rho_n^s q_n^\ell, \quad (2.8)$$

where $F_{\log\mathcal{N}(\mu^s, \sigma^s)}$ is again the cdf of a log-normal distribution parameterized by its median μ^s and its standard deviation σ^s . This process begins with some delay, once the thermal time has reached a certain threshold τ^s . The biomass of green leaves is therefore:

$$q_n^{g\ell} = q_n^\ell - q_n^{s\ell} = (1 - \rho_n^s) q_n^\ell. \quad (2.9)$$

Finally, the biomass of the roots is increased by what is not allocated to the foliage:

$$q_n^r = q_{n-1}^r + (1 - \gamma_n^{\text{sto}}) q_n^{\text{sto}}. \quad (2.10)$$

The standard formulation of the LNAS transition function is:

$$x_{n+1} = \begin{bmatrix} q_{n+1} \\ \gamma_{n+1} \\ q_{n+1}^\ell \\ q_{n+1}^{g\ell} \\ q_{n+1}^r \end{bmatrix} = \begin{bmatrix} r_n \mu (1 - \exp(k q_n^{g\ell}/e)) (1 + \eta_n^q) \\ (\gamma^0 + (\gamma^\ell - \gamma^0) F_{\log\mathcal{N}(\mu^a, \sigma^a)}(\tau_n)) (1 + \eta_n^\gamma) \\ q_n^\ell + \gamma_{n+1} q_{n+1} \\ (1 - F_{\log\mathcal{N}(\mu^s, \sigma^s)}(\tau_n - \tau^s)) q_{n+1}^\ell \\ q_n^r + (1 - \gamma_{n+1}) q_{n+1} \end{bmatrix} = f_n(x_n, u_n, \theta, \eta_n). \quad (2.11)$$

However, this formulation does not allow to explicitly compute the transition pdf $p(x_{n+1}|\theta, x_n)$ from the values contained in the state x_n , which will notably be detailed in Section 5.5. This is why it is reformulated with the dual values for q and γ as:

$$x_{n+1} = \begin{bmatrix} q_{n+1}^{\text{det}} \\ q_{n+1}^{\text{sto}} \\ \gamma_{n+1}^{\text{det}} \\ \gamma_{n+1}^{\text{sto}} \\ q_{n+1}^\ell \\ q_{n+1}^{g\ell} \\ q_{n+1}^r \end{bmatrix} = \begin{bmatrix} r_n \mu (1 - \exp(k q_n^{g\ell}/e)) \\ q_{n+1}^{\text{det}} (1 + \eta_n^q) \\ (\gamma^0 + (\gamma^\ell - \gamma^0) F_{\log\mathcal{N}(\mu^a, \sigma^a)}(\tau_n)) \\ \gamma_{n+1}^{\text{det}} (1 + \eta_n^\gamma) \\ q_n^\ell + \gamma_{n+1} q_{n+1} \\ (1 - F_{\log\mathcal{N}(\mu^s, \sigma^s)}(\tau_n - \tau^s)) q_n^\ell \\ q_n^r + (1 - \gamma_{n+1}) q_{n+1} \end{bmatrix} = f_n(x_n, u_n, \theta, \eta_n) \quad (2.12)$$

where $x_n = (q_n^{\text{det}}, q_n^{\text{sto}}, \gamma_n^{\text{det}}, \gamma_n^{\text{sto}}, q_n^\ell, q_n^{g\ell}, q_n^r)$ represents the hidden state, $u_n = (r_n, \tau_n)$ the external variables, $\theta = (\mu, k, e, \gamma^0, \gamma^\ell, \mu^a, \sigma^a, \tau^s, \mu^s, \sigma^s)$ the set of 10 parameters and $\eta_n = (\eta_n^q, \eta_n^\gamma)$ the process noises.

According to Section 1.19, the transition pdf can be written as:

$$p(x_{n+1}|\theta, x_n) = \frac{1}{\sigma^q q_{n+1}^{\det} \sqrt{2\pi}} e^{-\frac{(q_{n+1}^{\text{sto}} - q_{n+1}^{\det})^2}{2(\sigma^q q_{n+1}^{\det})^2}} \frac{1}{\sigma^\gamma \gamma_{n+1}^{\det} \sqrt{2\pi}} e^{-\frac{(\gamma_{n+1}^{\text{sto}} - \gamma_{n+1}^{\det})^2}{2(\sigma^\gamma \gamma_{n+1}^{\det})^2}}. \quad (2.13)$$

It is assumed that the biomass of green leaves q^{g^ℓ} and that of roots q^r are observed with respective measurement noises ξ^{g^ℓ} and ξ^r , that are multiplicative normal noises. The experimental data potentially available on day n can therefore be written:

$$y_n = \begin{bmatrix} \tilde{q}_n^{g^\ell} \\ \tilde{q}_n^r \end{bmatrix} = \begin{bmatrix} q_n^{g^\ell} (1 + \xi_n^{g^\ell}) \\ q_n^r (1 + \xi_n^r) \end{bmatrix} = g_n(x_n, \theta, \xi_n) \quad (2.14)$$

where $\xi = (\xi^{g^\ell}, \xi^r)$ represents the observation noises and, according to Equation 1.24 and when both biomasses are available on day n , the observation pdf can be written as:

$$p(y_n|\theta, x_n) = \frac{1}{\sigma^{g^\ell} q_{n+1}^{g^\ell} \sqrt{2\pi}} e^{-\frac{(\tilde{q}_{n+1}^{g^\ell} - q_{n+1}^{g^\ell})^2}{2(\sigma^{g^\ell} q_{n+1}^{g^\ell})^2}} \frac{1}{\sigma^r q_{n+1}^r \sqrt{2\pi}} e^{-\frac{(\tilde{q}_{n+1}^r - q_{n+1}^r)^2}{2(\sigma^r q_{n+1}^r)^2}}. \quad (2.15)$$

Figures 2.2 and 2.3 display the dynamics of the main variables of this model.

2.3 Log-Normal Allocation and Senescence for wheat

Being a central component of the western alimentation and one of the two most consumed cereals worldwide, the study of wheat is of special importance. There exist two main types of wheat: winter wheat, which is sowed during autumn and characterizes the mediterranean and temperate regions, and spring wheat which is sowed during spring and is mostly used in countries with harsher winters. The world production of all types of wheat amounted to 729 million tons in 2014, that is to say 100 kilograms per capita, making it the 4th most important crop in terms of volume behind sugar cane, corn and rice. France itself is the 5th producer of wheat worldwide and devotes more than 5 million hectares to its culture for a production of nearly 40 million tons, highlighting the significance of its culture.

The LNAS model for sugar beet served as an inspiration for the design of a similar compartmental model for wheat. The latter can be decomposed into two parts: a model for the soil and one for the plant itself. The soil model will allow to compute potential water stress that could limit the daily production of biomass. It is of notable value for modelling purposes since roots have truly a secret life: in an hectare of wheat, there may be up to 300,000 kilometers of roots feeding the plant with water and nutrients. A highly developed root system is the result of a well-structured soil and crucial in order to obtain high yields.

As in the LNAS model for sugar beet, all state variables will be considered per unit surface, even though the underlying processes are described at the plant level.

2.3.1 Soil model

Water is an indispensable resource to plant growth and it is notably obtained thanks to the roots of the plant. The importance of the soil model is motivated by the importance of variations in soil water content that can potentially lead to stress and slow down plant growth. The soil is assumed to have total depth z^{soil} . Beyond the latter, it is considered to be more stony, that water is not retained by the soil and that plant roots cannot grow. In the practical cases to which this model is applied, it is typically of the order $z^{\text{soil}} \approx 150\text{cm}$. The evolution of the soil water content between days n and $n + 1$ is given by the balance equation:

$$\rho_{n+1} = \rho_n + w_n - E_n - T_n - d_n \quad (2.16)$$

where ρ_n is the soil water content, w_n is the daily input of water into the system – it comprises both precipitations and irrigation and is a measurable environmental variable – and E_n and T_n are the water lost by evaporation from the soil and by transpiration from the plant respectively, finally d_n represents a drain function which accounts for the fact that the soil cannot withhold a quantity of water greater than its field capacity ρ^{max} . All the variables of this balance equation are expressed in mm. The expression of the soil water content can be described as an integral over the whole soil depth:

$$\rho_n = \int_{z=0}^{z^{\text{soil}}} \theta_n(z) dz, \quad (2.17)$$

where $\theta_n(z)$ ($\text{mm} \cdot \text{mm}^{-1}$) is the soil humidity at depth z . The maximum soil humidity corresponding to the field capacity is denoted $\theta^{\text{max}}(z)$ and such that:

$$\rho^{\text{max}} = \int_{z=0}^{z^{\text{soil}}} \theta^{\text{max}}(z) dz. \quad (2.18)$$

Below some threshold called the wilting point ρ^{min} , the plant is not able to extract water through the roots, the corresponding soil humidity at depth z is denoted $\theta^{\text{min}}(z)$ and identically:

$$\rho^{\text{min}} = \int_{z=0}^{z^{\text{soil}}} \theta^{\text{min}}(z) dz. \quad (2.19)$$



Figure 2.4: Field of wheat.

The normalized humidity can then be defined as:

$$h_n(z) = \max \left(0, \frac{\theta_n(z) - \theta^{\min}(z)}{\theta^{\max}(z) - \theta^{\min}(z)} \right) \in [0, 1]. \quad (2.20)$$

The daily potential evaporation E_n^{pot} and transpiration T_n^{pot} can be calculated respectively as:

$$E_n^{\text{pot}} = \kappa^E e_n \exp \left(-k q_n^{g\ell} / e \right) \quad (2.21)$$

and:

$$T_n^{\text{pot}} = \kappa^T e_n \left(1 - \exp \left(-k q_n^{g\ell} / e \right) \right) \quad (2.22)$$

where e_n is the potential evapotranspiration that can be estimated using the Penman–Monteith formula [Monteith, 1965] and is considered as an environmental variable, κ^E and κ^T are cultural coefficients specific to the species and the soil type, k is the Beer–Lambert coefficient, e the leaf mass per area and $q_n^{g\ell}$ the biomass of green leaves. Just as for the Beer–Lambert law for light interception, the transpiration process occurring between the green foliage and the air is assumed to be proportional to $1 - e^{-k q_n^{g\ell} / e}$. On the contrary, the evaporation process occurring between the soil and the air is taken proportional to $e^{-k q_n^{g\ell} / e}$. The maximal evaporation E^{max} and transpiration T^{max} can be integrated over the soil depth as:

$$E^{\text{max}} = \int_0^{z^{\text{soil}}} \left(\min(\theta_n(z), \theta^{\min}(z)) + \frac{\max(0, \theta_n(z) - \theta^{\min}(z))}{\max \left(1, \exp\left(-\frac{z}{z^{\text{surf}}}\right) + \exp\left(-\frac{z}{z_n^r}\right) \right)} \right) \exp \left(-\frac{z}{z^{\text{surf}}} \right) dz, \quad (2.23)$$

and:

$$T^{\text{max}} = \int_0^{z^{\text{soil}}} \min \left(1, \frac{h_n(z)}{h^c} \right) \frac{\max(0, \theta_n(z) - \theta^{\min}(z))}{\max \left(1, \exp\left(-\frac{z}{z^{\text{surf}}}\right) + \exp\left(-\frac{z}{z_n^r}\right) \right)} \exp \left(-\frac{z}{z_n^r} \right) dz. \quad (2.24)$$

The surface depth z^{surf} depends on the nature of the soil and is the characteristic length for evaporation. The first term in the integrand $\min(\theta_n(z), \theta^{\min}(z))$ ensures that for this particular term the evaporation is no greater than the minimum soil water content at this depth. In the case where $\theta_n(z) > \theta^{\min}(z)$, an additional contribution is provided by the second term of the integrand.

As far as transpiration is concerned, the characteristic length is given by the depth of the roots z_n^r . The dynamics of the root horizon is assumed to be proportional to the root biomass, though it cannot go farther than the soil depth:

$$z_n^r = \min(z^{\text{soil}}, \lambda q_n^r) \quad (2.25)$$

and the evolution of the roots' biomass q_n^r is detailed in the plant model. An additional reduction factor is considered: if the normalized humidity at depth z is below a certain threshold h^c , then water extraction becomes more difficult and transpiration is therefore reduced. To account for the fact that the sum of the maximum evaporation and transpiration cannot exceed the water reserve in any scenario, a normalization by:

$$\max \left(1, \exp\left(-\frac{z}{z^{\text{surf}}}\right) + \exp\left(-\frac{z}{z_n^r}\right) \right) \quad (2.26)$$

is introduced. In practice to update the soil variables, particularly those linked to the evapotranspiration

processes, one performs a discretization of the soil from $z = 0$ to $z = z^{\text{soil}}$ in n_z layers of height $\delta_z = z^{\text{soil}}/n_z$. Each layer $i \in \llbracket 1, n_z \rrbracket$ thus starts at depth $z^i = (i - 1)\delta_z$. All the values in layer i are denoted with a superscript i . For the sake of simplicity, the functions giving the minimal and maximal soil water content at depth z will be taken constant so that $\theta^{\text{max}}(z) = \theta^{\text{max}}$ and $\theta^{\text{min}}(z) = \theta^{\text{min}}$, but it can easily be adapted if layers of different natures and characteristics are observed in the soil. The maximum evaporation and transpiration therefore become respectively:

$$E^{\text{max}} = \sum_{i=1}^{n_z} \left(\min(\theta_n^i, \theta^{\text{min}}) + \frac{\max(0, \theta_n^i - \theta^{\text{min}})}{\max\left(1, \exp\left(-\frac{z^i}{z^{\text{surf}}}\right) + \exp\left(-\frac{z^i}{z_n^r}\right)\right)} \right) \exp\left(-\frac{z^i}{z^{\text{surf}}}\right) \delta_z, \quad (2.27)$$

and:

$$T^{\text{max}} = \sum_{i=1}^{n_z} \min\left(1, \frac{h_n^i}{h^c}\right) \frac{\max(0, \theta_n^i - \theta^{\text{min}})}{\max\left(1, \exp\left(-\frac{z^i}{z^{\text{surf}}}\right) + \exp\left(-\frac{z^i}{z_n^r}\right)\right)} \exp\left(-\frac{z^i}{z_n^r}\right) \delta_z. \quad (2.28)$$

The drain value is calculated as:

$$d_n = \max(0, r_n + w_n - \theta^{\text{max}} z^{\text{soil}}) \quad (2.29)$$

and if it is positive, the soil is then completely flooded up until the field capacity in all layers:

$$\theta_n^i = \theta^{\text{max}} \text{ and } z^{\text{water}} = z^{\text{soil}}. \quad (2.30)$$

Otherwise, the layers are filled up to the field capacity from top to bottom until there is no water left:

$$\text{while } w^{\text{avail}} - \delta_z(\theta^{\text{max}} - \theta_n^i) > 0, \theta_n^i = \theta^{\text{max}} \text{ and } w^{\text{avail}} -= \delta_z(\theta^{\text{max}} - \theta_n^i). \quad (2.31)$$

At some layer of index j , there does not remain enough water to fill it up to the maximum humidity and it is filled with the remaining water:

$$\theta_n^j += \frac{w^{\text{avail}}}{\delta_z}. \quad (2.32)$$

The actual evaporation and transpiration are given by:

$$E_n = \min(E_n^{\text{pot}}, E_n^{\text{max}}) \quad (2.33)$$

and:

$$T_n = \min(T_n^{\text{pot}}, T_n^{\text{max}}) \quad (2.34)$$

and the maximal evaporation and transpiration in layer i are given by the terms in the sums of Equations 2.27 and 2.28 respectively. In a similar process as for the water input, the actual evaporation E_n and transpiration T_n are allocated to each layer from top to bottom, taking into account the maximum evaporation and transpiration in each layer as defined above, until all the transpiration and evaporation have been allocated. The evaporation and transpiration depths z^E and z^T are deduced from the last layer which is allocated a part of the total evaporation and transpiration respectively.

The soil can further be decomposed into several horizon layers for which measurements of the soil parameters θ_{min}^h and θ_{max}^h experimental data can be available, they are indexed by $h \in \llbracket 1, n_h \rrbracket$ where n_h is the total number of horizon layers (typically $n_h \in \llbracket 3, 10 \rrbracket$). Horizon layer h is comprised between $z = z^h$ and

$z = z^{h+1}$, and in each of them the soil water content θ_n^h can be calculated by averaging over the discretization layers that it contains:

$$\theta_n^h = \frac{\delta_z}{z^{h+1} - z^h} \sum_{i|(i-1)\delta_z \in [z^h, z^{h+1}[} \theta_n^i. \quad (2.35)$$

A crucial aspect of plant growth models is to consider the influence of stress on the functioning of the plant. In the present case, three different stresses that potentially limit the production of biomass are considered. The first one is a thermal stress, it is defined as:

$$s_n^t = \max \left(0, \min \left(1, \frac{t_n - t^b}{t^o - t^b} \right) \right), \quad (2.36)$$

where t^b is the base temperature and t^o an optimal temperature. Essentially, when the daily temperature t_n is below the base temperature, $s_n^t = 0$ and the plant will not grow anymore. On the contrary, when t_n is greater than the optimal temperature, $s_n^t = 1$ and the plant functioning is not thermally limited. Between these two values for the daily temperature, $s_n^t \in]0, 1[$ partially limits the production of biomass as will be detailed later. Another form of thermal stress occurs through a biological process called vernalization during which plants are exposed to a prolonged cold. Only after vernalization can winter wheat grow and flower. The related thermal time is accumulated as:

$$\tau_{n+1}^{\text{vern}} = \tau_n^{\text{vern}} + \max(0, \min(1, 1 - e^{\text{vern}} |t_n - t^{\text{vern}}|)) \quad (2.37)$$

where t^{vern} is the optimal vernalizing temperature and e^{vern} is a coefficient characteristic of the range of the vernalizing temperatures. In essence, the closer to t^{vern} the temperature, the quicker the vernalization-related thermal time accumulates, and the corresponding stress index is defined as:

$$s_n^{\text{vern}} = \frac{\tau_n^{\text{vern}}}{\tau_{\text{vern,opt}}^{\text{vern}}}. \quad (2.38)$$

The second kind of stress taken into account is of hydric nature: when the transpirable water reserve in the soil falls below the daily requirement, it diminishes the production of biomass and induces the closure of the plant's stomata which increases the plant's temperature. Hydric stress is defined by the stomatal stress index:

$$s_n^\tau = \frac{\tau_n}{\tau_n^{\text{pot}}} \quad (2.39)$$

and it can also be considered, for instance, to favour root growth during the biomass allocation process described in Equation 2.45.

2.3.2 Plant model

The plant is considered to be composed of several compartments: roots r , stem s , yellow stem ys , green leaves gl , yellow leaves yl , and grain g . The total biomass of the plant on day n is denoted by q_n^{tot} and the biomasses of each compartment are denoted by q_n^r , q_n^s , q_n^{ys} , q_n^{gl} , q_n^{yl} and q_n^g respectively. The time evolution

of the total biomass is described as:

$$q_{n+1}^{\text{tot}} = q_n^{\text{tot}} + q_n^p - q_n^\ell \quad (2.40)$$

where q_n^p is the amount of biomass created via photosynthesis and q_n^ℓ is the amount of lost biomass as yellow leaves fall from the plant. The biomass available for redistribution on day n is denoted by q_n and comprises the produced biomass q_n^p and the biomass remobilized from yellow leaves and the stem, q_n° . The production of biomass via photosynthesis q_n^p is described as usual by the Beer–Lambert law:

$$q_n^p = \min(s_n^\tau, s_n^{\text{vern}}, s_n^t) r_n \mu \left(1 - \exp\left(-k q_n^{g\ell}/e\right)\right). \quad (2.41)$$

What is new compared to the sugar beet production equation is that this production is affected by the thermal and hydric stress defined in the soil model. The pool of biomass on day n available for redistribution is therefore:

$$q_n = q_n^p + q_n^r. \quad (2.42)$$

On day n , q_n is allocated to the roots, the stem, the green leaves and the grain with respective relative quantities α_n^r , α_n^s , $\alpha_n^{g\ell}$, α_n^g , where τ_n is the thermal time on day n . As a matter of fact, for all $n > 0$:

$$\alpha_n^r + \alpha_n^s + \alpha_n^{g\ell} + \alpha_n^g = 1. \quad (2.43)$$

Motivated by the fact that the only change in the grain biomass is the allocation and considering experimental data, α_n^g was defined as a log-normal cdf of the thermal time with delay τ^g :

$$\alpha_n^g = F_{\log\mathcal{N}(\mu^g, \sigma^g)}(\tau_n - \tau^g) \quad (2.44)$$

where we recall that the expression of the log-normal cdf $F_{\log\mathcal{N}(\mu, \sigma)}$ is given by Equation 2.5. The rest of the available biomass is then allocated between all the other compartments as follows:

$$\begin{cases} \alpha_n^s &= (1 - \alpha_n^g) s_n^\tau \frac{a_n^s}{a_n^s + a_n^{g\ell} + a_n^r} \\ \alpha_n^{g\ell} &= (1 - \alpha_n^g) s_n^\tau \frac{a_n^{g\ell}}{a_n^s + a_n^{g\ell} + a_n^r} \\ \alpha_n^r &= (1 - \alpha_n^g) \left(1 - s_n^\tau \frac{a_n^s + a_n^{g\ell}}{a_n^s + a_n^{g\ell} + a_n^r}\right) \end{cases} \quad (2.45)$$

so that the sum of all coefficients indeed equals 1 and the coefficients a_n^s , $a_n^{g\ell}$ and a_n^r are allocation coefficients. The water stress s_n^τ is included so as to favour allocation of biomass towards roots when water resources become limited. Biomass allocation to the stem in wheat only occurs after a given thermal time called montaison, the allocation function to the stem can therefore be taken as a delayed log-normal:

$$a_n^s = a^s F_{\log\mathcal{N}(\mu^s, \sigma^s)}(\tau_n - \tau^s), \quad (2.46)$$

and the allocation functions for leaves and roots are chosen constant, $a_n^{g\ell} = a^{g\ell}$ and $a_n^r = a^r$. The stem and senescent leaves are assumed to remobilize a fraction of their biomasses on each day n , so that:

$$q_n^\circ = q_n^{\circ, y\ell} + q_n^{\circ, s} \quad (2.47)$$

with $q_n^{\circ, y\ell}$ and $q_n^{\circ, s}$ the remobilized biomasses from yellow leaves and the stem respectively. The quantity of remobilized biomass, say, for the yellow leaves is a fraction of the biomass of the yellow leaves, and this

fraction is again parameterized by the cdf of a log-normal distribution with a given delay:

$$q_n^{\circ,gl} = \beta_n^{gl} q_n^{gl} \doteq \eta^{gl} F_{\log\mathcal{N}(\mu^\circ, \sigma^\circ)}(\tau_n - \tau^\circ) q_n^{gl}. \quad (2.48)$$

The same is assumed for the stem and:

$$q_n^{\circ,s} = \beta_n^s q_n^s \doteq \eta^s F_{\log\mathcal{N}(\mu^\circ, \sigma^\circ)}(\tau_n - \tau^\circ) q_n^s. \quad (2.49)$$

The parameters η^{yl} and η^s are introduced here because the same dynamics will be assumed for the senescence process and, keeping in mind that the total number of model parameters should be kept relatively low, they allow to avoid redefining new log-normal cdf entirely parameterized by new variables. The remobilization and senescence processes therefore share the same parameterization with μ° , σ° and τ° .

On each day n , part of the green leaves become senescent and turn into yellow ones. The corresponding biomass is denoted by $q_n^{gl \rightarrow yl}$. All the same, some yellow leaves fall from the plant; this is described through the loss of yellow leaves biomass $q_n^l \doteq q_n^{yl \rightarrow fl}$. The same dynamics as for remobilization is assumed, and the quantity of senescence biomass for green leaves and the stem is therefore:

$$q_n^{gl \rightarrow yl} = \gamma_n^{gl} q_n^{gl} \doteq (1 - \eta^{gl}) F_{\log\mathcal{N}(\mu^\circ, \sigma^\circ)}(\tau_n - \tau^\circ) q_n^{gl}, \quad (2.50)$$

and:

$$q_n^{s \rightarrow ys} = \gamma_n^s q_n^s \doteq (1 - \eta^s) F_{\log\mathcal{N}(\mu^\circ, \sigma^\circ)}(\tau_n - \tau^\circ) q_n^s. \quad (2.51)$$

Finally, in a similar manner, the biomass of yellow leaves that fall on day n is given by:

$$q_n^{yl \rightarrow fl} = \gamma_n^{yl} q_n^{yl} \doteq F_{\log\mathcal{N}(\mu^\dagger, \sigma^\dagger)}(\tau_n - \tau^\dagger) q_n^{yl}. \quad (2.52)$$

with this time different parameters for the log-normal cdf as this biological process happens later in the growth cycle. When a sufficient thermal time τ^{init} is reached on day n^{init} , the biomasses of the different compartments are initiated using the biomass of the seed. The latter is exclusively distributed between the roots and the green leaves using their respective allocation parameters:

$$q_{n^{\text{init}}}^r = \frac{a^r}{a^r + a^{gl}} q^{\text{seed}} \quad \text{and} \quad q_{n^{\text{init}}}^{gl} = \frac{a^{gl}}{a^r + a^{gl}} q^{\text{seed}}, \quad (2.53)$$

all other biomasses being set to zero. From then on, the time evolution of each compartment of the plant is given by:

		<i>Remob.</i>	<i>Alloc.</i>	<i>Senesc.1</i>	<i>Senesc.2</i>
q_{n+1}^r	$=$	q_n^r	$+$	$\alpha_n^r q_n$	
q_{n+1}^s	$=$	$q_n^s - \beta_n^s q_n^s$	$+$	$\alpha_n^s q_n$	$- \gamma_n^s q_n^s$
q_{n+1}^{ys}	$=$	q_n^{ys}		$+$	$\gamma_n^s q_n^s$
q_{n+1}^{gl}	$=$	$q_n^{gl} - \beta_n^{gl} q_n^{gl}$	$+$	$\alpha_n^{gl} q_n$	$- \gamma_n^{gl} q_n^{gl}$
q_{n+1}^{yl}	$=$	$q_n^{yl} - q_n^{r,yl}$		$+$	$\gamma_n^{gl} q_n^{gl} - \gamma_n^{yl} q_n^{yl}$
q_{n+1}^g	$=$	q_n^g	$+$	$\alpha_n^g q_n$	

and summing the four equations indeed yields Equation 2.40.

The general SSM transition equation for this model would be too cumbersome to write here. Let us mention nevertheless that the environmental variables considered are $u_n = (t_n, r_n, w_n, e_n)$ and that there is a total of 33 parameters, 10 for the soil model:

$$(\theta^{\min}, \theta^{\max}, \kappa^E, \kappa^T, h^c, \lambda, z^{\text{soil}}, z^{\text{surf}}, \rho^{\text{init}}, t^0) \quad (2.55)$$

and 23 for the plant model:

$$(t^b, \mu, e, k, a^r, a^s, a^{gl}, \tau^g, \mu^g, \sigma^g, \tau^s, \mu^s, \sigma^s, \tau^{\odot}, \mu^{\odot}, \sigma^{\odot}, \tau^{\downarrow}, \mu^{\downarrow}, \sigma^{\downarrow}, \eta^s, \eta^{gl}, q^{\text{seed}}, \tau^{\text{init}}). \quad (2.56)$$

Of course, all these parameters need not be estimated from scratch and most of them can be set to sensible values originating from either biological knowledge or the environment. It is also worth mentioning that this model does not comprise process noise in its current formulation, although they could be added in the allocation or production processes.

For wheat, experimental data often include biomasses of the different compartments q_n^{gl} , q_n^r , q_n^s and q_n^g and

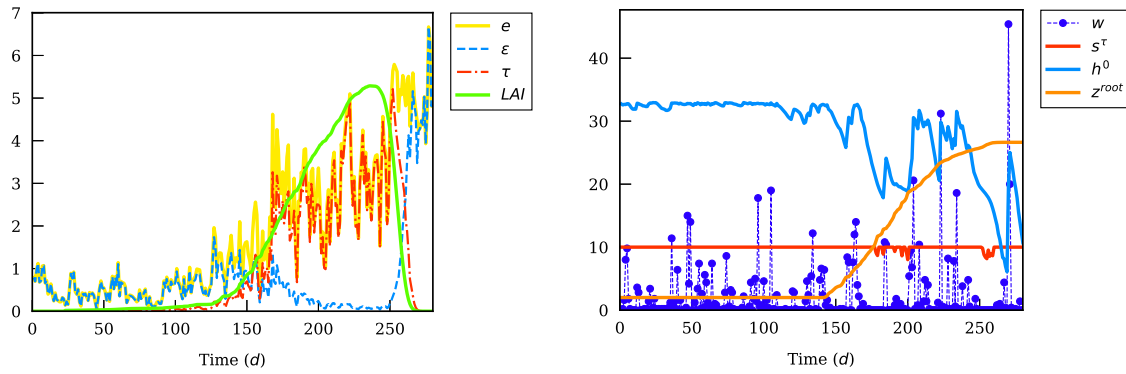


Figure 2.5: Left: dynamics of the evaporation and transpiration processes, the evapotranspiration e (mm) is known at all times as an environmental variable, E and T (mm) are the actual evaporation and transpiration; the leaf area index (dimensionless) is also provided to witness the evolution of the plant's growth. Right: water dynamics, the precipitations w (mm) are given as an environmental variable, the water stress index s_n^{τ} (dimensionless) has been multiplied by 10 for clarity, h_0 ($\text{cm} \cdot \text{m}^{-1}$) denotes the humidity in the first horizon layer (between $z = 0\text{cm}$ and $z = 30\text{cm}$) and the root horizon is also provided to follow plant growth in the soil.

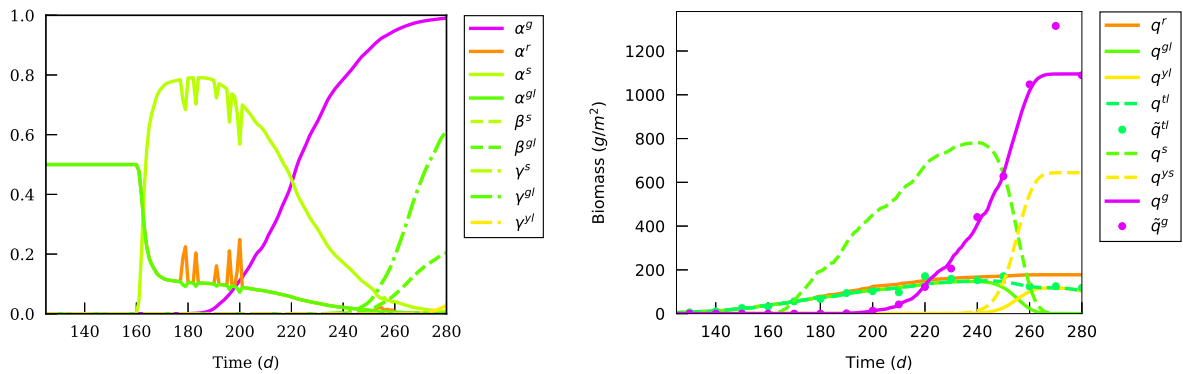


Figure 2.6: Left: evolution of the different coefficients involved in the allocation, remobilization and senescence process. Right: evolution of the different biomasses for all compartments. Continuous lines correspond to hidden states while filled circles denote observations.

the water reserve r_n . All these quantities are as usual assumed to be observed with some uncertainty and a multiplicative normal noise is assumed for all of them:

$$y_n = \begin{bmatrix} \tilde{q}_n^{g^\ell} \\ \tilde{q}_n^r \\ \tilde{q}_n^s \\ \tilde{q}_n^g \\ \tilde{r}_n \end{bmatrix} = \begin{bmatrix} q_n^{g^\ell} (1 + \xi_n^{g^\ell}) \\ q_n^r (1 + \xi_n^r) \\ q_n^s (1 + \xi_n^s) \\ q_n^g (1 + \xi_n^g) \\ r_n (1 + \xi_n^r) \end{bmatrix} = g_n(x_n, \theta, \xi_n). \quad (2.57)$$

The dynamics of the soil and plant models are displayed on Figures 2.5 and 2.6 respectively.

2.4 GreenLab for *Arabidopsis thaliana*

Arabidopsis thaliana, most commonly known as thale cress, is a small annual flowering plant from the mustard family (Brassicaceae) originating from Europe, Asia, and northwestern Africa. Its height can reach up to 25 cm. It possesses a single primary root growing vertically downwards, from which originate smaller lateral roots. The leaves form a rosette at the base of the plant and a few leaves also grow on the flowering system. White flowers grow on the stem and the fruits are siliquas containing approximately between 20 and 30 seeds. The entire growth cycle of *Arabidopsis thaliana* is usually of 6 weeks.

From the beginning of the 20th century, *Arabidopsis thaliana* began to be used for research purposes. Several factors make it particularly suitable for its use in research: its small size, its rapid lifecycle, its resistance and its efficiency in self-pollination are major attractive features. It is now widely used for the study of plant sciences, including genetics, evolution, population genetics, and plant development. It was the first genome of a plant to be sequenced in 2000 because of its relatively small size: it comprises 157 million base pairs distributed across 5 chromosomes.

More precisely, in the context of this work, our interest for this plant stems from the large amount of data that the Phenoscope, a high-throughput phenotyping platform, can provide for this plant in particular. Time series of images of *Arabidopsis thaliana*, typically over a period of 21 days, can be obtained for many different individuals, and such experiments yield the data necessary for the application of hierarchical modelling to a population of plants. More details will be given in Chapter 7 on the nature of the data obtained from the Phenoscope for *Arabidopsis thaliana*.

The GreenLab model [Yan et al., 2004] is a typical functional-structural model in the sense that it combines the description of plant architectural development and ecophysiological functioning. A version has been developed for the full cycle of *Arabidopsis thaliana* growth in [Christophe et al., 2008]. Basically, a developmental submodel predicts organ appearances while source-sink dynamics is simultaneously described:



Figure 2.7: *Arabidopsis thaliana*.

biomass production is computed via radiation interception by leaf area and the produced biomass is allocated between all growing organs according to individual sink strengths. Individual leaf areas are then deduced from leaf masses. In our study, only the rosette stage of *Arabidopsis thaliana* growth is considered, which particularly simplifies the organogenesis submodel and the number of competing sinks. Furthermore, at this stage, the senescence process has not started yet.

As detailed in Section 7.2.1, leaves first appear in pairs (the 1st and 2nd leaves together, then the 3rd and 4th leaves) before the following ones start appearing rhythmically. It should be noted that, for the sake of clarity, we have numbered leaves including the 2 cotyledons, so that the first true leaf is actually leaf 3 in our numbering. The time span between the appearances of two successive leaves is called the phyllochron [Wilhelm and McMaster, 1995]. It is mostly driven by thermal time, however, the GreenLab model for *Arabidopsis thaliana* will only be used for experiments in controlled environment, and under constant thermal conditions it amounts to considering the calendar time as the main driver of organogenesis. For a better understanding of the source-sink dynamics in this first stage of study, we consider that the leaf appearance times of the first 4 leaves are known, whereas those of the subsequent leaves are such that their difference is always the phyllochron ϕ (h).

As usual, one considers that plant growth starts at germination. At this time, the biomass is supposed to be that of the seed q_0 . To take into account the photoperiod and the differences in temperature between day and night, the time step is taken to be the hour. The plant is assumed to grow only during the day, which lasts typically $n_s = 8$ h in the experimental conditions of our study. Once growth has started, the biomass produced at time step n is given by the usual Beer–Lambert law:

$$q_n = r_n \mu s \left[1 - \exp \left(-\frac{k}{s e} \sum_{v \in \llbracket 1, \nu_n \rrbracket} q_n^v \right) \right] \quad (2.58)$$

where r_n is the photosynthetically active radiation ($\text{MJ} \cdot \text{cm}^{-2}$), μ is the radiation use efficiency ($\text{g} \cdot \text{MJ}^{-1}$), s is related to the projected area of the plant (cm^2), k is the Beer–Lambert law coefficient of light extinction (dimensionless), e is the leaf mass per area ($\text{g} \cdot \text{cm}^{-2}$), ν_n is the number of leaves of the plant at time step n

and q_n^v is the biomass of the v -th leaf (g).

The pool of produced biomass is then allocated to the different organs of the plant. In the present case, that is to say the rosette stage, only the leaves are considered. It actually amounts to consider that a constant proportion α of the biomass produced at time step n is allocated to the root system, with thus a real radiation use efficiency $\mu^{\text{real}} = \mu/(1 - \alpha)$. If this assumption is known to be oversimplifying on the whole growth cycle of the plant, it remains reasonable during the rosette stage in the absence of water stress. The biomasses allocated to the different leaves are proportional to their respective demands, or sink strengths, which are functions of their expansion stage, i.e. the thermal time since appearance. In previous GreenLab models, Beta distributions were used for the sink functions. This was not judged to be the best option here since the expansion period of the leaves is not known and they did not yield optimal results. Instead, log-normal distributions were used as they allow for a similar growth dynamics with an ever ongoing growth. As was suggested by the analysis of the areas of the different leaves, two different functions were used for the first 4 (performed) leaves on the one hand and the leaves with rank higher than 5 on the other, the demand of the v -th leaf hence being:

$$d_n^v = \rho_1 \frac{f_{\log\mathcal{N}(\mu_1, \sigma_1)}(\tau_n - \tau^v)}{\|f_{\log\mathcal{N}(\mu_1, \sigma_1)}\|_\infty} \quad (2.59)$$

if $v \leq 4$ and:

$$d_n^v = \rho_2 \frac{f_{\log\mathcal{N}(\mu_2, \sigma_2)}(\tau_n - \tau^v)}{\|f_{\log\mathcal{N}(\mu_2, \sigma_2)}\|_\infty} \quad (2.60)$$

if $v > 4$. Dividing by the uniform norms ensures a proper normalization to avoid variations of the functions maximum with their parameters, though a coefficient ρ_2 allows for a different intensity of the two different kinds of leaves (ρ_1 is actually always equal to 1 and defined only for the sake of notations). An index $k(v)$ indicates whether a leaf belongs to the first 4 leaves or not, i.e. $k(v) = 1 + \mathbb{1}(v > 4)$. Here, $f_{\log\mathcal{N}(\mu, \sigma)}$ is the pdf of a log-normal distribution this time parameterized for convenience by its mean μ and standard deviation σ :

$$f_{\log\mathcal{N}(\mu, \sigma)}(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\log(x) - \mu)^2}{2\sigma^2}\right), \quad (2.61)$$

τ_n is the thermal time at time step n and τ_v is the accumulated thermal time of the v -th leaf since its emergence (both in $^\circ\text{C} \cdot \text{h}$). (μ_1, σ_1^2) and (μ_2, σ_2^2) are the parameters of the log-normal distributions for the performed leaves and those with rank higher than 5 respectively. The biomass allocated to a leaf is then the relative demand of the available produced biomass:

$$\delta q_n^v = \frac{d_n^v}{\sum_{w \in \llbracket 1, \nu_n \rrbracket} d_n^w} q_n, \quad (2.62)$$

which allows to update the cumulated biomass of each leaf:

$$q_n^v = q_{n-1}^v + \delta q_n^v, \quad (2.63)$$

and compute the related leaf areas:

$$a_n^v = e^{-1} q_n^v. \quad (2.64)$$

The GreenLab transition equations can be summarized as:

$$x_{n+1} = \begin{bmatrix} q_{n+1} \\ (d_{n+1}^v)_{v \in [1, \nu_n]} \\ (\delta q_{n+1}^v)_{v \in [1, \nu_n]} \\ (q_{n+1}^v)_{v \in [1, \nu_n]} \\ (a_{n+1}^v)_{v \in [1, \nu_n]} \end{bmatrix} = \begin{bmatrix} r_n \mu s [1 - \exp(-\frac{k}{s e} \sum q_n^v)] \\ \left(\rho_{k(v)} f_{\log \mathcal{N}(\mu_{k(v)}, \sigma_{k(v)})}(\tau_n - \tau^v) \| f_{\log \mathcal{N}(\mu_{k(v)}, \sigma_{k(v)})} \|_{\infty}^{-1} \right)_{v \in [1, \nu_n]} \\ \left(\frac{d_{n+1}^v}{\sum d_{n+1}^w} q_{n+1} \right)_{v \in [1, \nu_n]} \\ (q_n^v + \delta q_{n+1}^v)_{v \in [1, \nu_n]} \\ (e^{-1} q_{n+1}^v)_{v \in [1, \nu_n]} \end{bmatrix} = f_n(x_n, u_n, \theta)$$

where $x_n = (q_n, (d_n^v)_{1:\nu_n}, (\delta q_n^v)_{1:\nu_n}, (q_n^v)_{1:\nu_n}, (a_n^v)_{1:\nu_n})$ represents the hidden state, $u_n = (r_n, \tau_n)$ the external variables and $\theta = (\phi, \mu, s, e, k, \mu_1, \sigma_1, \mu_2, \sigma_2, \rho_2, q_0)$ the parameters. There is no process noise in this model.

The GreenLab observation equations are:

$$y_n = \begin{bmatrix} \tilde{a}_n^1 \\ \vdots \\ \tilde{a}_n^{\nu_n} \end{bmatrix} = \begin{bmatrix} a_n^1 \times (1 + \xi_{1,n}) \\ \vdots \\ a_n^{\nu_n} \times (1 + \xi_{\nu_n,n}) \end{bmatrix} = g_n(x_n, \theta, \xi_n) \quad (2.65)$$

where $\xi_n \sim \mathcal{N}(0, \sigma^2 I_{\nu_n, \nu_n})$ represents the observation noises. The observations associated to the whole growth cycle of a plant is a sequence of vectors of leaf areas. Since the number of leaves is not the same at each time, a step of post-processing is used to transform this set of observations into a matrix of size $T \times \nu_{\max}$, where ν_{\max} is the maximum number of leaves, i.e. the number of leaves at time step T . The values of leaf areas that are related to:

- either leaves not emerged yet (during the first time steps, no leaf area can be attributed to a leaf that has not emerged yet),
- or unobserved values (in particular, as will be detailed in Chapter 7, the first two leaves become quickly recovered by other leaves and undetectable via analysis of zenithal images),

will be considered and marked as \circ . An example of experimental data for this model would look like:

$$y \doteq (y_{nv})_{n \in [1, T], v \in [1, \nu_{\max}]} = \begin{pmatrix} 0.030 & 0.035 & \circ & \dots & \circ \\ 0.038 & 0.041 & 0.020 & \dots & \circ \\ \dots & \dots & \dots & \dots & \dots \\ \circ & \circ & 0.512 & \dots & 0.853 \end{pmatrix}. \quad (2.66)$$

Since the GreenLab model will be used in particular in the context of population models, some additional notations are introduced. If y_i denoted the observations related to an individual i , then the actual experimental data for the k -th leaf of this individual is:

$$y_i(k) = \{x \in [y_{nk}]_{n \in [1, T]} \mid x \neq \circ\}, \quad (2.67)$$

and the vectorized experimental data:

$$y_i = [y_i(1), y_i(2), \dots, y_i(\nu_{\max})] \doteq (y_{ij})_{j \in \llbracket 1, n_i \rrbracket} \in \mathbb{R}^{n_i} \quad (2.68)$$

where it is recalled that n_i represents the total number of observations for the i -th individual thus corresponding to observations at different times for each leaf. In the rest of this thesis, each leaf is identified by its rank with a specific colour as specified in Table 2.1. The dynamics of the demands and areas for each leaf are depicted on Figure 2.9 with this colour code. Only the evolution of the plant during the $n_s = 8$ h of the day is represented, and the n -th day corresponds to the n_s n -th hour on the graphs.

rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
colour	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●

Table 2.1: Colours attributed to each leaf of a specific rank for easier identification.

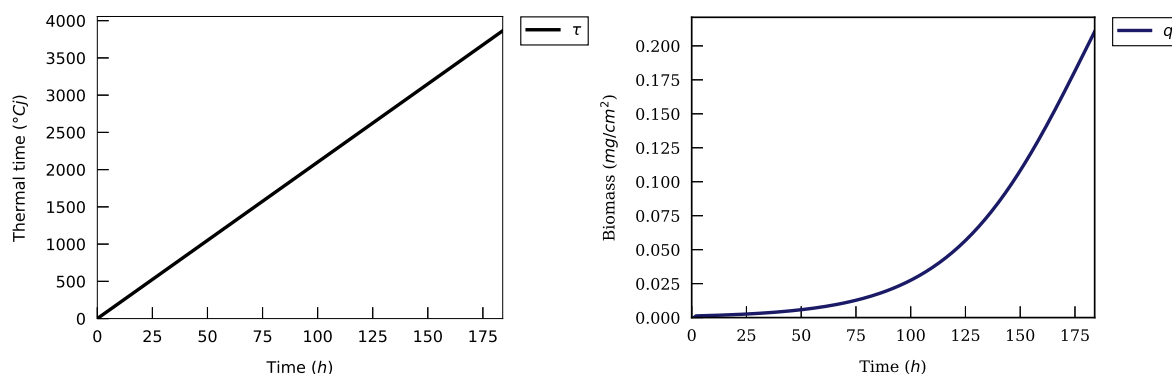


Figure 2.8: Left: example of thermal time dynamics for constant temperature. Right: production of biomass, no process noise is considered in this model for the production of biomass, contrarily to the model for sugar beet.

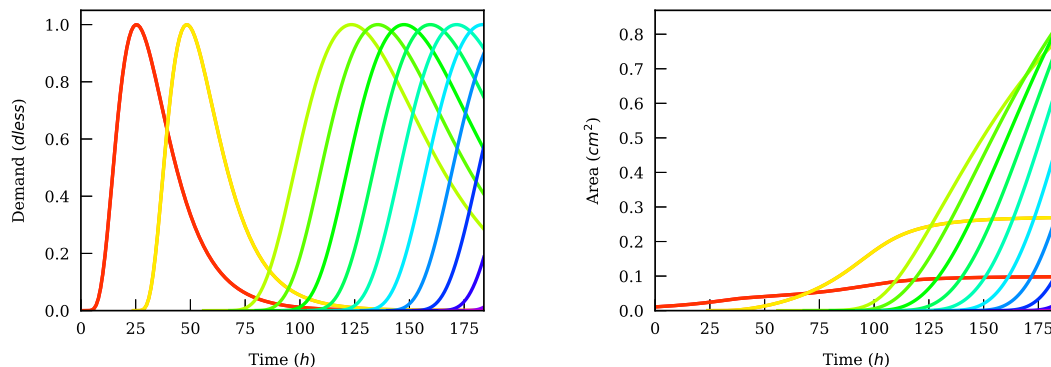


Figure 2.9: Left: demand curves d_v (dimensionless) for the different leaves, they have all been normalized with a maximum value of 1 for clarity so that the coefficients ρ_1 and ρ_2 do not play any role on this graph. Right: leaf area a_v (cm^2) for the different leaves.

Chapter 3

Estimation of parameters and hidden states for a single individual

ANY MODEL WORTHY OF THE NAME INVOLVES PARAMETERS that not only have a meaningful impact on the model outputs but whose values can be not well known a priori. The question of evaluating the most sensible values for parameters, and possible hidden states, within this model framework is therefore essential. The improvements in computing power and efficiency over the last decades allowed to consider the use and development of estimation methods for more and more complex models, notably those based on Monte Carlo approximation.

In the rest of this chapter, emphasis is put upon the estimation of a single individual in the context of general state space models (SSMs). The estimation problem in the context of populations will be presented in Chapter 4. We begin by presenting the general principle behind parameter and state estimation, notably by discussing the data used for model calibration in Section 3.1.1 and the criteria used in order to assess the accuracy of estimation in Section 3.1.2. Knowing which parameters have the most influence on the model outputs is crucial when the model comprises many parameters and is the concern of sensitivity analysis introduced in Section 3.2, where the classical Sobol method is presented before another approach considering the correlation between parameters. Once the most important parameters to estimate have been identified, estimation per se can occur. The two main statistical paradigms that are the frequentist and Bayesian approaches are introduced and discussed. Some of the frequentist methods are discussed in Section 3.4, while Bayesian methods, which constitute the major part of this work, are presented in Section 3.5.

3.1 General principle

Once a model has been designed to describe in the best possible way a system (in this case the growth of a plant), it needs to be calibrated, i.e. the model parameters need to be given realistic values and this is done using experimental data. Usually, only a subset $\theta_e \subset \theta$ of the model parameters is estimated, but for the sake

of simplicity and as long as there is no ambiguity, both the set of all model parameters and that of estimated parameters will be denoted θ .

System identification amounts to estimate the values of certain functional parameters θ jointly to the values of the hidden states $x_{1:T}$. Another subject of interest evoked at the end of this chapter is the estimation of parameters related to the process and observation noises η and ξ , e.g. the standard deviations of normal distributions from which noises are sampled at each time step. For now we focus on functional parameters. There are two main types of such parameters:

- some of them correspond to known physical or biological processes for which sensible values are already known. This is the case of k the extinction coefficient in the Beer–Lambert law, or e the leaf mass per area. Despite this prior knowledge on their values and because of calibration issues, it might still be necessary to estimate these parameters;
- some others are variables intervening in an a priori reasonable descriptive process parameterization. This is the case of the parameters of the log-normal distributions that are used in the two plant models considered here.

First and as mentioned earlier, some parameters can be considered known and fixed since they either are involved in well described physical processes or have very little influence on the model outputs. Second, the simultaneous estimation of all the remaining model parameters might be troublesome for several reasons: estimation procedures rely on numerical algorithms that hardly converge when the dimension of the state space to be explored is too important, a problem known as the curse of dimensionality. Identifiability problems may also arise which render the estimation of the parameters impossible but most importantly irrelevant from a modelling point of view – a trivial example would be the estimation of the two parameters k and e intervening as a ratio in Equation 2.58 for the produced biomass and as there is no experimental data to discriminate between the influence of the two parameters. Increasing the number of model parameters can have a significant effect on the mean square error; the latter can be decomposed into two terms of bias and variance:

$$\text{MSE} \left(y(\hat{\theta}) \right) = \mathbb{E} \left((y(\hat{\theta}) - y(\theta))^2 \right) = \text{Bias}^2 \left(y(\hat{\theta}) \right) + \mathbb{V} \left(y(\hat{\theta}) \right) \quad (3.1)$$

whose expressions depend on the observation model used. Models with low bias are often more complex and allow to represent the experimental data with more precision. However, they can also incorporate part of the process or observation noises intrinsic to the data used for calibration, their predictions being thus less precise despite an additional complexity. On the contrary, models with high bias are generally simpler but can provide predictions that are more robust. This is the reason why we try to keep the number of estimated parameters generally quite low.

3.1.1 Data for calibration

Classically, synthetic data are simulated using the model to calibrate from known values of parameters in order to validate the method used, which can further be tested on real data. Simulating a model with a

known set of parameters allows to know the values of all the hidden states and all the observations at all time steps. Some of the observations can then serve as experimental data to test different estimation procedures: not all data (all types of observations, values at all time steps) need to be kept. Since in this case both the real values of the parameters θ and those of the hidden states $x_{1:T}$ are known, their estimates can then be easily compared to the true values. However, experimental data originating from a real life process cannot be described exactly with a given model. This approximation is not taken into account when directly simulating data from the model. Nonetheless it remains interesting to study how the different estimation algorithms behave on this type of data in order to:

- make sure that the algorithms are correctly implemented,
- and study their performances and convergence properties.

The second type of data comes from real experiments and will be detailed later when used in the estimation procedures. We can already mention that for sugar beet, experimental data sets were obtained from the French Technical institute for sugar beet (ITB) in 2010: the biomasses of green leaves and that of roots are given at 14 different days. For *Arabidopsis thaliana*, data is obtained thanks to a phenotyping platform, the Phenoscope (Section 7.1), which outputs zenithal images of plants. Image analysis of these images is then performed (Sections 7.3 and 7.4) so as to obtain the areas of the different leaves of a plant on different days, thereby providing enough data for the organ-scale plant model that GreenLab is.

3.1.2 Criteria

Throughout this work, the accuracy of estimation will have to be assessed. In the case of simulated data, the true values of both the parameters and the hidden states are known, which allows a direct comparison with their estimates. In this scenario, we will consider the relative errors for a single parameter θ_i and a single hidden state at time step n x_n^i :

$$\delta_{\theta_i} = \frac{|\theta_i^{\text{true}} - \hat{\theta}_i|}{\theta_i^{\text{true}}} \quad \text{and} \quad \delta_{x_n^i} = \frac{|x_n^{i,\text{true}} - \hat{x}_n^i|}{x_n^{i,\text{true}}}. \quad (3.2)$$

In the case of real data, this comparison is not possible anymore as there do not exist such true values. The only available information are the observations on the system $y = (y_i)_{i \in \llbracket 1, d_y \rrbracket}$, and in this case, if we denote by $x = (x_i)_{i \in \llbracket 1, d_y \rrbracket}$ the corresponding hidden states either obtained via state estimation or with model simulation using an estimate of the model parameters, the most standard criterion [Wallach, 2006] for the evaluation of the predictive capabilities of a model and its estimates is the root mean square error of prediction (RMSEP):

$$\text{RMSEP} = \left(\frac{1}{d_y} \sum_{i=1}^{d_y} (y_i - x_i)^2 \right)^{\frac{1}{2}}. \quad (3.3)$$

The RMSEP should be computed for observations of the same nature so that the errors are of the same order of magnitude. The modelling efficiency (EF), also called the coefficient of determination, is also used as it is

a normalized indicator of the error and complements the information provided by the RMSEP:

$$EF = 1 - \frac{\sum_{i=1}^{d_y} (y_i - x_i)^2}{\sum_{i=1}^{d_y} (y_i - \bar{y}_i)^2} \quad (3.4)$$

where \bar{y}_i is the mean of the observations and again, the modelling efficiency should be computed for variables of the same kind. It is comprised between $-\infty$ and 1, and the closer it gets to 1, the better the fitting to the data.

3.2 Sensitivity analysis

3.2.1 Sobol method

To overcome the difficulties inherent to a high number of parameters to be estimated, sensitivity analysis is usually conducted. Its basic principle is to study how uncertainty in the model inputs influence the uncertainty in the model outputs. In the present case, model inputs are the model parameters and the model outputs are the hidden states corresponding to observed variables. Although no strict ranking can always be established – since parameters do not affect in the same way all the different outputs – at different time steps, the most influential parameters can still be selected most of the time. This allows to set all the others – the least influential – to fixed reasonable values without having too much of an effect on the global model behaviour. This procedure is called screening or factor fixing.

The general principles of sensitivity analysis [Saltelli et al., 2000], [Saltelli et al., 2004] are briefly recalled. There exist two families of methods. Local methods study only the effect of the local variation of a given factor around a nominal value with all the other factors being fixed. Global methods allow the simultaneous variations of all factors according to prescribed probability distributions. Because of the nonlinearities and the interactions between parameters that often exist in ecological models [Cariboni et al., 2007] and since, as will be seen in Chapter 5, we dispose of important computing resources, Sobol method will be used.

Although this analysis could be conducted on the control variables $u_{1:T}$ or the initial state x_0 , here we focus on sensitivity analysis on model functional parameters θ and the two former are fixed to known values. Obviously, the output of a model are also influenced by the process and observation noises. To isolate the influence of functional parameters on the model output only, all the standard deviations of the process and observation noises are set to zero, which is equivalent as not taking any into account. In these conditions, the dynamics of a model described in Equation 1.9 can be simplified as:

$$y_{1 \rightarrow T} = M(\theta) = M((\theta_i)_{i \in [1, d_\theta]}) \quad (3.5)$$

and in particular, the observation y_n^ℓ relative to variable v_ℓ at time step n is simply a function of the parameters:

$$y_n^\ell(\theta) = y_n^\ell((\theta_i)_{i \in [1, d_\theta]}) \quad (3.6)$$

where $(\theta_i)_{i \in \llbracket 1, d_\theta \rrbracket}$ are called the input factors and are considered random variables. They are given probability distributions which describe how these parameters are likely to vary within the state space. Sensitivity analysis results will therefore differ depending on the distributions used to sample these parameters.

The Hoeffding functional decomposition [Hoeffding, 1948] allows to decompose a model output y_n^ℓ as:

$$y_n^\ell((\theta_i)_{i \in \llbracket 1, d_\theta \rrbracket}) = f_{n\emptyset}^\ell + \sum_{u \in S^*} f_{nu}^\ell(\theta_u) \quad (3.7)$$

where S is the set of all subsets of $\llbracket 1, d_\theta \rrbracket$ and $S^* = S \setminus \emptyset$. In the case of independent input variables, this representation is unique insofar as:

$$\mathbb{C}(f_{nu}^\ell(\theta_u), f_{nv}^\ell(\theta_v)) = 0 \quad (3.8)$$

for $u \neq v$. Taking the variance of Equation 3.7 yields:

$$\mathbb{V}(y_n^\ell) = \sum_{u \in S^*} \mathbb{V}_u(f_{nu}^\ell(\theta_u)) \quad (3.9)$$

and dividing by the total variance yields:

$$1 = \sum_{u \in S^*} \frac{\mathbb{V}_u(f_{nu}^\ell(\theta_u))}{\mathbb{V}(y_n^\ell)} = \sum_{u \in S^*} S_{nu}^\ell \quad (3.10)$$

where S_{nu}^ℓ is the sensitivity index related to the multi-index u for variable v_ℓ at time n . For each $u \in S$, it is possible to define the total order sensitivity index $T_{nu}^\ell = \sum_{v, u \subset v} S_{nv}^\ell$, summing all the different contributions of θ_u in the output variance. For a single parameter θ_i with $i \in \llbracket 1, d_\theta \rrbracket$, the first order sensitivity index and the total order sensitivity index can therefore be written respectively as:

$$\begin{cases} S_{ni}^\ell &= \frac{\mathbb{V}_i(\mathbb{E}_{-i}(y_n^\ell | \theta_i))}{\mathbb{V}(y_n^\ell)}, \\ T_{ni}^\ell &= \frac{\mathbb{E}_{-i}(\mathbb{V}_i(y_n^\ell | \theta_{-i}))}{\mathbb{V}(y_n^\ell)}, \end{cases} \quad (3.11)$$

where i refers to taking into account only variable θ_i and $-i$ all variables θ_j for $j \neq i$. These expectations and variances are computed using a Monte Carlo procedure [Saltelli et al., 1993]. The convergence of the algorithm can be improved by using the permutation of certain values of parameters as suggested by Wu et al. [2012] and described in Algorithm 1.

3.2.2 Structural and correlative sensitivity analysis

The assumption that the model parameters are independent is in fact rather strong and restricting, all the more in complex systems, which is the case of most biological models, notably plant growth models: when a model is used for different genotypes of a given plant, the dependence between parameters is significant [Lecœur et al., 2011]. This is why it is important, particularly in the context of genotypic differentiation, to understand the influence of correlations between model parameters.

Li et al. [2010] introduced a generalization of the Sobol indices known as the structural and correlative sensitivity analysis. It is based on another decomposition of a generalized hierarchical Hoeffding–Sobol (GHHS)

Algorithm 1 Sobol algorithm for sensitivity analysis

Sample the first two different sets of parameters according to their prior distributions:

$$\begin{cases} (\theta^{s,j})_{j \in \llbracket 1, N \rrbracket} \sim p(\theta) \\ (\theta^{s',j})_{j \in \llbracket 1, N \rrbracket} \sim p(\theta) \end{cases}$$

Sample the relative model observations via model simulation, i.e. for all $j \in \llbracket 1, N \rrbracket$:

$$\begin{cases} y^{s,j} \sim p(y|\theta^{s,j}) \\ y^{s',j} \sim p(y|\theta^{s',j}) \end{cases}$$

Define the corresponding complementary sets of parameters as copies of the latter:

$$\begin{cases} (\theta^{c,j})_{j \in \llbracket 1, N \rrbracket} = (\theta^{s,j})_{j \in \llbracket 1, N \rrbracket} \\ (\theta^{c',j})_{j \in \llbracket 1, N \rrbracket} = (\theta^{s',j})_{j \in \llbracket 1, N \rrbracket} \end{cases}$$

For $i \in \llbracket 1, d_\theta \rrbracket$

Switch parameters θ_i between $(\theta^{c,j})_{j \in \llbracket 1, N \rrbracket}$ and $(\theta^{c',j})_{j \in \llbracket 1, N \rrbracket}$

Sample $(y^{c,j})_{j \in \llbracket 1, N \rrbracket}$ and $(y^{c',j})_{j \in \llbracket 1, N \rrbracket}$ via model simulation

For $\ell \in \llbracket 1, d_\ell \rrbracket$

For $k \in \llbracket 1, O \rrbracket$

Compute $S_{t_k i}^\ell$ and $T_{t_k i}^\ell$

End

End

End

decomposition which was formally demonstrated by [Chastaing et al. \[2012\]](#). Assuming that $y(\theta) \in H = L_{\mathbb{R}}^2(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^p), P_\theta)$ where P_θ denotes the probability measure of the parameters, we denote by H_u the subspace of H such that all its elements depend only on θ_u . We can recursively define the subspace family $(H_u^0)_{u \in \mathcal{S}}$ such that H_\emptyset^0 is the space of constant functions and:

$$H_u^0 = \{h_u \in H_u \mid \langle h_u, h_v \rangle = 0, \forall v \subset u, \forall h_v \in H_v^0\}. \quad (3.12)$$

The GHHS decomposition then reads:

$$\mathbb{V}(y) = \sum_{u \in \mathcal{S}^*} \mathbb{V}(f_u(\theta_u)) + \sum_{u \in \mathcal{S}^*} \sum_{\substack{v \in \mathcal{S}^* \\ u \cap v \neq \{u, v\}}} \mathbb{C}(f_u(\theta_u), f_v(\theta_v)) \quad (3.13)$$

with $f_u \in H_u^0$ and where we have dropped the indices related to the time step n and the observed variable ℓ for readability. In the case where the correlations between parameters are ignored, the second term in the previous decomposition vanishes and it simplifies to Equation 3.7 of the Sobol case. One can define the three indices of generalized sensitivity related to θ_u :

$$\begin{cases} S_u = \frac{\mathbb{C}(y, f_u(\theta_u))}{\mathbb{V}(y)}, \\ S_u^s = \frac{\mathbb{V}(f_u(\theta_u))}{\mathbb{V}(y)}, \\ S_u^c = \sum_{\substack{v \in \mathcal{S}^* \\ u \cap v \neq \{u, v\}}} \frac{\mathbb{C}(f_u(\theta_u), f_v(\theta_v))}{\mathbb{V}(y)}, \end{cases} \quad (3.14)$$

which are called the total, structural and correlated contributions respectively and are such that $S_u = S_u^s + S_u^c$ and $\sum_u S_u = 1$. As in the classical Sobol case, total indices can be defined as:

$$\left\{ \begin{array}{l} T_u = \sum_{v|u \subset v} S_v, \\ T_u^s = \sum_{v|u \subset v} S_v^s, \\ T_u^c = \sum_{v|u \subset v} S_v^c. \end{array} \right. \quad (3.15)$$

These indices do not share the same properties as those in the uncorrelated case since the indices S_u are not comprised between 0 and 1 anymore and only the structural contributions S_u^s are assured to be positive, which renders their interpretation all the less so straightforward. As far as single parameters are concerned (i.e. $|u| = 1$) [Sainte-Marie et al. \[2017\]](#) proposed to consider the 5 following cases:

- $T_i > T_i^c > T_i^s > 0$: the uncertainty related to θ_i is significant and mostly due to its correlations with other parameters;
- $T_i > T_i^s > T_i^c > 0$: the uncertainty related to θ_i is mostly due to its direct influence and amplified by correlations;
- $T_i^s > T_i > 0 > T_i^c$: the uncertainty related to θ_i is significant but reduced by its dependence to other parameters;
- $T_i = 0$ and $T_i^s = -T_i^c$: the direct influence of θ_i is tempered by its dependence to other parameters;
- $T_i^s > 0 > T_i > T_i^c$: the parameter θ_i is essentially reducing the contribution of other parameters but does not have a significant direct effect on the total variance per se.

The practical computation of the functions f_u in the GHHS decomposition is based on the hierarchically orthogonal Gram–Schmidt (HOGS) algorithm which provides an approximation of each term of the decomposition $f_u \in H_u^0$ by projecting it on subspaces of finite dimension. It first starts by the approximation of the subspaces H_u^0 for which $|u| = 1$ (corresponding to a single parameter) by considering p subspaces $H_u^{0,L}$ of dimension L . A standard choice is that of the bases composed of polynomials whose degree is comprised between 1 and L . The HOGS algorithm then proceeds to construct the approximation of the subspaces for multi-indices of higher cardinality. We can limit the degree of interactions considered to d . Then for $u \in S$ such that $2 \leq |u| \leq d$, a functional basis $(\phi_l^u)_{l \in \llbracket 1, L \rrbracket^{|u|}}$ is defined as:

$$\phi_l^u(\theta_u) = \prod_{i=1}^{|u|} \phi_{l_i}^{u_i}(\theta_{u_i}) + \sum_{\substack{v \in S^* \\ v \subset u}} \left[\sum_{k \in \llbracket 1, L \rrbracket^{|v|}} \lambda_{kl}^v \phi_k^v(x_v) \right] + C_l^u \quad (3.16)$$

where the λ_{kl}^v and C_l^u are determined by solving the hierarchical orthogonality conditions defining H_u^0 :

$$\langle \phi_l^u, 1 \rangle_N = 0 \text{ and } \langle \phi_l^u, \phi_k^v \rangle_N = 0, \forall v \subset u, \forall k \in \llbracket 1, L \rrbracket^{|v|} \quad (3.17)$$

where $\langle \cdot, \cdot \rangle_N$ denotes the empirical inner product. The HOGS algorithm therefore relies on 3 main parameters: the dimension $L \in \mathbb{N}^*$ of the bases of order 1, the number of samples $N \in \mathbb{N}^*$ and the maximum degree

of interaction d . Let us highlight the fact that the construction of such a functional basis does not depend on the studied model. Once this functional basis has been computed, each term in the GHHS decomposition can be approximated by projection on this basis:

$$f_u(\theta_u) \approx \hat{f}_u(\theta_u) = \sum_l \beta_l^u \phi_l^u(\theta_u) \quad (3.18)$$

where the β_l^u must be estimated. This can be achieved with a standard least squares algorithm:

$$(\beta_u^l) = \arg \min \|Y - \Phi B\|^2 + \lambda J(B) \quad (3.19)$$

where $Y = (y^j)_{j \in \llbracket 1, N \rrbracket}$, $\Phi = (\phi_l^u(\theta_u^j))$ and $B = (\beta_l^u)$, and where the second term $\lambda J(B)$ accounts for a possible penalization with $\lambda > 0$, $J(B) = \mu \|B\|_1 + (1 - \mu) \|B\|_2$ is characteristic of an elastic net procedure with $\mu \in [0, 1]$ and whose well-known special cases are ridge regression ($\mu = 1$) and lasso regression ($\mu = 0$).

3.3 Frequentist vs. Bayesian estimation

The opposition between these two paradigms may be one of the most important within the community of statistical inference. Although both are concerned with understanding statistical systems thanks to the estimation of model parameters using experimental data, the interpretations that they attribute to the nature of parameters are different.

The frequentist paradigm considers that the studied events exhibit a statistical stability and is based on the frequency of occurrences, i.e. their repetitions. The model parameters are assumed to have, true fixed values, and confidence intervals on the results of the estimation are usually provided.

The Bayesian formulation first appeared in the work of the Reverend Thomas Bayes *An essay towards solving a problem in the doctrine of chances* in 1763. From the 19th century, this statistical paradigm fell into oblivion and was resurrected during the second half of the 20th century by Alan Turing, Leonard Savage and Dennis Lindley among others, even though up until the 1990s it was left out by the academic community. With the appearance of more powerful computers, some of the Bayesian algorithms such as Markov chain Monte

Algorithm 2 Structural and correlative sensitivity analysis

Generate multi index list

Sample parameters according to their prior distributions $(\theta^{s,j})_{j \in \llbracket 1, N \rrbracket} \sim p(\theta)$

Evaluate the HOFD bases

Sample the relative model observations via model simulation, i.e. for all $j \in \llbracket 1, N \rrbracket$, $y^{s,j} \sim p(y|\theta^{s,j})$

For $\ell \in \llbracket 1, d_\ell \rrbracket$

For $k = 1 : O$

 Compute the vector of all observations for the different samples for ℓ at time t_k

 Compute the related HOFD coefficients β_u^l using a least squares algorithm with adequate penalization

 Compute $S_u^s, S_u^c, S_u, T_u^s, T_u^c$ and T_u .

End

End

Carlo or sequential Monte Carlo methods could be used more easily, which led in return to an increase of interest for the theoretical foundations of these methods.

In the Bayesian framework, the parameters do not have fixed values, instead they are considered to be random variables that follow distributions. Probabilities are interpreted more as a degree of belief rather than as the frequency of a phenomenon. Pre-estimation belief on the parameters distributions can be incorporated in a prior distribution, which is then updated thanks to the observations on the system. In the limit of an infinite number of observations, both approaches yield the same results.

Bayesian inference calculates the posterior probability of the parameters as a consequence of two factors, the prior distribution on the parameters and the likelihood specific to the statistical model considered for the observed data, using Bayes' theorem:

$$p(\theta|y_{1 \rightarrow T}) = \frac{p(y_{1 \rightarrow T}|\theta) p(\theta)}{p(y_{1 \rightarrow T})} = \frac{p(y_{1 \rightarrow T}|\theta) p(\theta)}{\int_{\theta} p(y_{1 \rightarrow T}|\theta) p(\theta) d\theta} \quad (3.20)$$

where $p(\theta)$ is the prior distribution and represent the original belief on the parameters before the experiment and taking into account the observations $y_{1 \rightarrow T}$, $p(y_{1 \rightarrow T}|\theta)$ is the likelihood, and $p(y_{1 \rightarrow T})$ is sometimes coined the marginal likelihood or evidence and can be rewritten by integrating over the whole parameter space. The posterior distribution $p(\theta|y_{1 \rightarrow T})$ thus describes the probability that the parameters take value θ after observation of the data $y_{1 \rightarrow T}$. The calculation of the evidence as an integral over the whole parameter space is the main issue in assessing the posterior distribution since it would potentially requires complex integrations over high dimension spaces.

The Bayesian approach seems more adapted to the case studies under consideration in the rest of this work for several reasons:

- first, it is more adapted to cases where there is a limited number of observations that are particularly noisy, as is often the case in plant growth models;
- integrating prior distributions on the parameters not only alleviates this problem and allows for the incorporation of biological background information and more modelling freedom (see [Illian et al., 2009] for instance); moreover, models of biological systems usually describe biophysical processes for which parameters have a clear interpretation and prior knowledge can be used to derive prior distributions;
- in the context of population models, the Bayesian approach allows for simpler analytical formulation in a Gibbs sampler (see Chapter 4);
- for the sake of philosophy and consistency, we consider mainly one paradigm in this whole work, in the cases of inference for both a single individual and a population.

3.4 A frequentist method: generalized least squares

In this section we briefly present the generalized least squares algorithm [Aitken, 1936]. It is very often used for parameter estimation within a frequentist paradigm as it is easy to implement and rather fast. It will notably be used for the comparison of frequentist and Bayesian estimation methods using the GreenLab model for *Arabidopsis thaliana*. We denote by $y \in \mathbb{R}^{d_y}$ the vector of all observations for a model and by $x(\theta) \in \mathbb{R}^{d_y}$ the vector of all the corresponding hidden states arising from model simulation. For the sake of simplicity, we assume for now that all the experimental data follow an additive normal observation model, and a much simpler formulation of the model reads:

$$y = x(\theta) + \xi \quad (3.21)$$

where $\xi \sim \mathcal{N}(0, \Sigma)$. The classical least squares estimator consists in minimizing the sum of all squared differences between the data and the model output:

$$\theta^{\text{LS}} = \arg \min_{\theta \in \Theta} (y - x(\theta))^T (y - x(\theta)). \quad (3.22)$$

Often, experimental data represents observed values with different orders of magnitude, and it becomes desirable to weight the components of y accordingly. This can be achieved by introducing a weight matrix $W \in \mathcal{M}_{d_y}(\mathbb{R})$ such that:

$$\theta^{\text{WLS}} = \arg \min_{\theta \in \Theta} (y - x(\theta))^T W (y - x(\theta)). \quad (3.23)$$

When the observation model is as in Equation 3.21, it is straightforward to show that the maximum likelihood estimator is equivalent to a weighted least squares estimator with weight matrix $W = \Sigma^{-1}$, which yields the generalized least squares estimator:

$$\theta^{\text{GLS}} = \arg \min_{\theta \in \Theta} (y - x(\theta))^T \Sigma^{-1} (y - x(\theta)). \quad (3.24)$$

It is of primary importance to assess the uncertainty on this estimator. In the case of maximum likelihood estimators, the asymptotical variance is given by the inverse of the Fisher information matrix:

$$I(\theta) = (\partial_{\theta} \log x(\theta))^2. \quad (3.25)$$

The variance on this estimator can be approximated in the case of the observation model 3.21 and considering that the covariance matrix Σ does not depend on θ , this yields:

$$\mathbb{V}(\theta^{\text{GLS}}) = J(\theta^{\text{GLS}})^T \Sigma^{-1} J(\theta^{\text{GLS}}) \quad (3.26)$$

where $J(\theta^{\text{GLS}}) = \partial_{\theta} x|_{\theta^{\text{GLS}}}$ is the Jacobian matrix. Most of the time, Σ is not known and must be estimated. A simple choice for the covariance matrix is that of heteroscedastic errors [Cournède et al., 2011], where all the observations of the same variable are grouped together. Assuming that the variables observed are $(v_{\ell})_{\ell \in \llbracket 1, d_{\ell} \rrbracket}$, then the vector of experimental data can be put under the form $y = (y^{\ell})_{\ell \in \llbracket 1, d_{\ell} \rrbracket}$ where $y^{\ell} \in \mathbb{R}^{d_{y^{\ell}}}$ comprises all observations of v_{ℓ} and the covariance matrix considered is:

$$\Sigma = \text{blockdiag}_{\ell \in \llbracket 1, d_{\ell} \rrbracket} \left\{ \sigma_{\ell}^2 I_{d_{y^{\ell}}} \right\} \quad (3.27)$$

where σ_ℓ^2 is the empirical variance of variable v_ℓ . A 2-stage procedure was proposed by Taylor [1977] where the parameters are first estimated using a generalized least squares algorithm, considering at each stage that the variance is known, and then updated with the new estimated parameters. The initial variance of each variable can be computed from the experimental data:

$$\begin{cases} (\sigma_\ell^{(1)})^2 &= \mathbb{V}(y^\ell), \\ \Sigma^{(1)} &= \text{blockdiag}_{\ell \in \llbracket 1, d_\ell \rrbracket} \left\{ (\sigma_\ell^{(1)})^2 I_{d_{y^\ell}} \right\}, \\ \hat{\theta}^{(1)} &= \arg \min_{\theta \in \Theta} (y - x(\theta))^T (\Sigma^{(1)})^{-1} (y - x(\theta)). \end{cases} \quad (3.28)$$

Once the first estimate $\hat{\theta}^{(1)}$ is obtained, the variance is updated as follows:

$$\begin{cases} (\sigma^{(2)})^2 &= \mathbb{V}(y - x(\hat{\theta}^{(1)})), \\ \Sigma^{(2)} &= \text{diag}((\sigma^{(2)})^2), \\ \hat{\theta}^{(2)} &= \arg \min_{\theta \in \Theta} (y - x(\theta))^T (\Sigma^{(2)})^{-1} (y - x(\theta)) \end{cases} \quad (3.29)$$

so as to obtain the final estimate $\hat{\theta}^{(2)}$. This procedure can be easily adapted in the multiplicative case and the second stage now reads:

$$\begin{cases} (\sigma^{(2)})^2 &= \mathbb{V}\left(\frac{y - x(\hat{\theta}^{(1)})}{x(\hat{\theta}^{(1)})}\right), \\ \Sigma^{(2)} &= \text{diag}((\sigma^{(2)})^2 x(\hat{\theta}^{(1)})^2), \\ \hat{\theta}^{(2)} &= \arg \min_{\theta \in \Theta} (y - x(\theta))^T (\Sigma^{(2)})^{-1} (y - x(\theta)). \end{cases} \quad (3.30)$$

The GLS algorithm is the only frequentist method that is going to be used in this work. It is still worth mentioning that, in the presence of process noise, one must resort to other methods such as the Expectation Maximization (EM) algorithm [Dempster et al., 1977] since the maximum likelihood estimator (MLE) cannot be derived explicitly in HMMs. The EM algorithm is briefly presented in Section 4.1 in the context of population models. Such an approach in the case of plant growth models has notably been discussed in [Trevezas and Cournède, 2013] and [Trevezas et al., 2014] but has not been considered in this work since, as previously explained, we adopt a Bayesian perspective throughout the whole thesis: the frequentist GLS algorithm will therefore be used only for the initialization of sensible parameter values within a global Bayesian framework.

3.5 Bayesian methods

There are two main families of methods for Bayesian inference, Markov chain Monte Carlo (MCMC) methods and sequential Monte Carlo (SMC) methods – which include Kalman-based filters as well as particle filters – whose purposes are slightly different: most of the time with MCMC methods, at each iteration, the model is simulated from time zero to the final time of interest, and to each MCMC iteration corresponds a potential candidate for the posterior distribution. With SMC methods, a set of many simulations are run simultaneously from one time step where observations are available to another, and at each of these time

steps, parameters and hidden states are corrected according to the current observation. SMC methods are therefore more suited for learning on the fly, i.e. for real applications where data come sequentially in time. Such examples are numerous in radar tracking, signal processing, finance and of course agriculture. Examples of past use and discussion of Bayesian methods for parameter and state estimation in plant science can be found in [Chen, 2014] and [Malefaki et al., 2014] for instance.

Remark: In the remaining of this work, the following abuse of notation might occur: sampling a variable z given y might be expressed as $z \sim p(z|y)$ and evaluating the value of the pdf of z given y might also be expressed as $p(z|y)$.

3.5.1 Markov chain Monte Carlo methods

Monte Carlo integration

Monte Carlo integration relies on the strong law of large numbers: let $(z^t)_{t \in \llbracket 1, M \rrbracket}$ be M i.i.d. random variables with law π and $h \in L^1(\pi)$, then:

$$\frac{1}{M} \sum_{t=1}^M h(z^t) \xrightarrow[a.s.]{M \rightarrow \infty} \int h(z) d\pi(z) = \mathbb{E}(h(z)). \quad (3.31)$$

It therefore becomes possible to approximate this integral by sampling many independent realizations z^t with respect to the law π . The precision on the estimate can be chosen arbitrarily high by choosing a sufficiently high number of samples. However, it may not be possible to sample directly from π . This is generally the case of the posterior distribution when models are complex.

When it is not possible to sample directly from π , it is still generally possible to evaluate it. A Markov chain can be generated so that its stationary distribution is precisely the target distribution from which one wants to sample. In practice, it is necessary to define a burn-in period during which the chain explores the state space. After a sufficiently long number of iterations, when the chain has reached its stationary distribution, samples obtained from the Markov chain are identically distributed. However, they are not independent, which prevents to use the strong law of large numbers. Nevertheless, under certain regularity conditions, a similar result holds for Markov chains: a Markov chain $(z^t)_{t \in \llbracket 1, M \rrbracket}$ with stationary distribution π is said ergodic if it is aperiodic, irreducible and positive recurrent, and in this case Equation 3.31 still holds.

Markov chains

The introduction of MCMC algorithms dates back to the middle of the last century, with the Metropolis–Hastings algorithm [Metropolis et al., 1953], [Hastings, 1970] and subsequently the Gibbs sampling algorithm [Geman and Geman, 1984], [Gelfand and Smith, 1990]. These two base algorithms were regarded as popular tools for the analysis of complex statistical models, by sampling values from an ergodic Markov chain for the estimation of the posterior distribution of both parameters and hidden states. However, these

Algorithm 3 Metropolis–Hastings algorithmChoose initial value z^0 **For** $t = 1 : M$ Sample $z^* \sim q(\cdot | z^t)$

Compute the acceptance probability:

$$\alpha(z^t, z^*) = 1 \wedge \frac{\pi(z^*) q(z^t | z^*)}{\pi(z^t) q(z^* | z^t)}$$

Set $z^{t+1} = z^*$ with probability α , set $z^{t+1} = z^t$ otherwise**End**

methods were extremely costly from a numerical point of view, and one had to wait until the beginning of the 1990s to witness a regain of interest for these methods with the increase in numerical power.

Metropolis–Hastings

Metropolis et al. [1953] introduced the first MCMC algorithm for the computation of integrals involving Boltzmann distributions and this procedure was generalized almost twenty years later by Hastings [1970]. It allows to generate samples with respect to a probability distribution for which direct sampling is not possible. This sequence can then be used either to approximate the target distribution or to compute an integral involving the latter. At each iteration t , a candidate z^* is generated with respect to a proposal distribution (also known as the instrumental distribution) q from which it is easy to sample from, e.g. a normal distribution. This candidate is either accepted or rejected – in which case the chain stagnates at its previous state – with an acceptance probability ensuring the convergence towards the target distribution. The Metropolis–Hastings algorithm is described in Algorithm 3.

The acceptance probability can be interpreted as made of two terms: on the one hand, it is desired that the algorithm head towards regions of high probability under the target distribution π and this is controlled by the ratio $\pi(z^*)/\pi(z^t)$, on the other hand the chain should not be stuck for too long in a region of high probability under the proposal distribution q , this is controlled by the ratio $q(z^t | z^*)/q(z^* | z^t)$. The acceptance probability ensures that if a candidate is sampled from the target distribution, then this will also be the case of the candidate sampled at the next iteration. This can easily be seen as the transition kernel for the Markov chain is such that:

$$p(z^{t+1} | z^t) = \alpha(z^t, z^{t+1}) q(z^{t+1} | z^t) + \mathbb{1}(z^{t+1} = z^t) \left(1 - \int q(z | z^t) \alpha(z^t, z) dz \right) \quad (3.32)$$

where the first term denotes acceptance of the candidate $z = z^{t+1}$ and the second term rejection of all other candidates. By definition of the acceptance probability:

$$\pi(z^t) q(z^{t+1} | z^t) \alpha(z^t, z^{t+1}) = \pi(z^{t+1}) q(z^t | z^{t+1}) \alpha(z^{t+1}, z^t) \quad (3.33)$$

which yields the detailed balance equation:

$$\pi(z^t) p(z^{t+1} | z^t) = \pi(z^{t+1}) p(z^t | z^{t+1}). \quad (3.34)$$

After integration of the last equation with respect to z^t :

$$\int \pi(z^t) p(z^{t+1}|z^t) dz^t = \pi(z^{t+1}). \quad (3.35)$$

The left-hand side represents the marginal distribution of z^{t+1} assuming that z^t has distribution π , and this marginal distribution is precisely equal to the target distribution π for z^{t+1} . This means that whenever a sample of the chain has distribution π , all the next candidates will also have distribution π . This simple argument yet does not constitute a rigorous proof which can be found in [Roberts, 1996].

As already mentioned, the acceptance probability involves only the evaluation of the target distribution: there is no need to directly sample from it nor is it necessary to compute the problematic normalizing constant $\int \pi(z) dz$, which constitutes one of the main strengths of the MCMC algorithms.

Proposal distributions

The nature of the proposal distribution has not been discussed yet: in principle it can have any form, the only requirement being that its support include the support of the target distribution, and the convergence towards the target distribution remains assured. However some choices will obviously lead to faster convergence towards the target distribution π because it drives the exploration of the state space [Robert and Casella, 1999].

Two main choices are classically considered: when the proposal distribution q does not depend on the current position of the chain z^t , the algorithm can be seen as an extension of accept-reject methods. In this particular case, the acceptance probability reduces to:

$$\alpha(z^t, z^*) = 1 \wedge \frac{\pi(z^*) q(z^t)}{\pi(z^t) q(z^*)} \quad (3.36)$$

Another popular choice for the proposal distribution is symmetric random walk. It depends only on the difference between the candidate and the previous state and is symmetric:

$$q(z|z') = q(z - z') = q(z' - z) \quad (3.37)$$

in which case the acceptance probability can thus be simplified to:

$$\alpha(z^t, z^*) = 1 \wedge \frac{\pi(z^*)}{\pi(z^t)}. \quad (3.38)$$

All the candidates that have a higher π value are then automatically accepted. One of the advantages of the symmetric random walk is that the computation of the acceptance probability requires to evaluate the target distribution only. Several choices for the random walk proposal distribution are possible though the most common one is to use a multivariate normal distribution with zero mean and general covariance matrix, the choice of which is of critical importance for the convergence of the chain. If it is too small, the difference between the current state of the chain and a candidate will be very small, whence a slow exploration of the state space. On the contrary, if it is too large, the candidates are often rejected and the chain stagnates too often.

Adaptive schemes

Adaptive methods have been proposed in order to overcome this problem. The acceptance rate is defined as the number of times a candidate has been accepted over the current iteration. It is a good criterion of how well the chain performs at exploring the state space and is of course related to the covariance matrix of the proposal distribution. When this ratio is too large, it most probably means that the chain moves too slowly and does not explore the state space as much as necessary. Contrarily, when it is too low, it may indicate that the chain moves too fast and possibly misses some regions of high probabilities for the target distribution.

To ensure a good exploration of the state space by the Markov chain, one might focus on the acceptance rate that has to be neither too low nor too high. In the case of the random walk Metropolis–Hastings algorithm when the proposal distribution is multivariate normal and when the dimension of the problem goes to infinity, the optimal acceptance rate has been proved to be approximately 0.234 [Roberts and Sahu, 1997], [Roberts and Rosenthal, 2001], [Bédard, 2008]. Moreover, Gelman et al. [1996] have shown that this optimal acceptance rate corresponds to an optimal covariance matrix equal to $(2.38^2/d) \Sigma_\pi$ where Σ_π is the covariance matrix of the target distribution. Several adaptive algorithms have been proposed to optimize the choice of the covariance matrix in the proposal distribution at each iteration and consequently the acceptance rate. Haario et al. [1998], Haario et al. [1999] estimated Σ_π at the end of each iteration t by considering the past realizations of the Markov chain:

$$\begin{cases} \mu^t &= \frac{1}{t} \sum_{u=1}^t z^u, \\ \Sigma^t &= \frac{1}{t} \sum_{u=1}^t (z^u - \mu^t)(z^u - \mu^t)^T. \end{cases} \quad (3.39)$$

This approach was further generalized by Andrieu and Thoms [2008] who proposed the integration of a Robbins–Monro update scheme [Robbins and Monro, 1951] based on stochastic approximation. At each iteration $t + 1$, a new candidate is generated according to the distribution $\mathcal{N}(z^t, \lambda^t \Sigma^t)$ where:

$$\begin{cases} \mu^t &= \mu^{t-1} + \gamma^t (z^t - \mu^{t-1}) \\ \Sigma^t &= \Sigma^{t-1} + \gamma^t [(z^t - \mu^t)(z^t - \mu^t)^T - \Sigma^{t-1}] \\ \lambda^t &= \lambda^{t-1} \exp(\gamma^t [\alpha(z^{t-1}, z^*) - \alpha^*]) \end{cases} \quad (3.40)$$

with a sequence (γ^t) such that $\sum_{t=1}^{\infty} \gamma^t = \infty$ and there exists $\nu > 0$ such that $\sum_{t=1}^{\infty} (\gamma^t)^{1+\nu} < \infty$, z^* is the candidate sampled at iteration t , $\alpha(z^{t-1}, z^*)$ is the acceptance probability and α^* is the optimal acceptance ratio, which we recall to be 0.234 in the multivariate case. It is obvious that when $\alpha(z^{t-1}, z^*) - \alpha^*$ is negative, the current acceptance rate is too low compared to the optimal one and that the variance – through λ^t – needs to be decreased. Conversely, when $\alpha(z^{t-1}, z^*) - \alpha^*$ is positive, the current acceptance rate is too high and the variance of the random walk is decreased.

Although this adaptive procedure can be adopted in order to enhance the acceptance rate and the convergence of the MCMC algorithms, it also has modified the nature of the chain, which is no longer Markovian as the current state now depends on the whole history of the chain and not only the previous state. It has been shown however that convergence is still ensured as long as the parameters of the adaptive schemes stay away

from poor values, see [Andrieu and Thoms, 2008] for more details.

Gibbs sampling

The Gibbs sampler was introduced by Geman and Geman [1984] and further developed by Gelfand and Smith [1990]. Its main idea is to decompose the sampling of a multidimensional variable of interest z into multiple ones of lower dimensionality. More precisely, if $z = (z_1, \dots, z_p)$ is decomposed into p blocks of variables (where each z_i has dimension comprised between 1 and that of z) and has distribution π , and if it is not possible to sample according to π but if it is easy to sample according to the full conditional distributions π_1, \dots, π_p , i.e.:

$$z_i | z_{-i} \sim \pi_i(\cdot | z_{-i}), \quad (3.41)$$

where z_{-i} denotes all variables but z_i , then the dimension of the problem can be reduced since it suffices to sample p times according to distributions of lower dimensionalities. An important and common special case is when all the blocks of z are unidimensional: one has thus to sample p times from univariate distributions. At each iteration, only one component z_i of z is therefore updated while all the others are held fixed. The proposal distribution for the i -th component is simply its full conditional distribution:

$$q(z_i^* | z_i^t, z_{-i}^t) = p_i(z_i^* | z_{-i}^t). \quad (3.42)$$

It can be proved [Robert and Casella, 1999] that the stationary distribution of the Markov chain thus generated is precisely the target distribution π . One immediate consequence of such a scheme is that all candidates are actually accepted because:

$$\begin{aligned} \alpha(z^t, z^*) &= 1 \wedge \frac{\pi(z^*) q(z^t | z^*)}{\pi(z^t) q(z^* | z^t)} \\ &= 1 \wedge \frac{\pi(z^*) \pi_i(z_i^t | z_{-i}^*)}{\pi(z^t) \pi_i(z_i^* | z_{-i}^t)} \\ &= 1 \wedge \frac{\pi(z^*) \pi_i(z_i^t | z_{-i}^t)}{\pi(z^t) \pi_i(z_i^* | z_{-i}^*)} \\ &= 1 \wedge \frac{\pi(z^*)}{\pi_i(z_i^* | z_{-i}^*)} / \frac{\pi(z^t)}{\pi_i(z_i^t | z_{-i}^t)} \\ &= 1 \wedge \frac{\pi(z_{-i}^*)}{\pi(z_{-i}^t)} \\ &= 1 \end{aligned} \quad (3.43)$$

since only the i -th component is updated, whence $z_{-i}^* = z_{-i}^t$. In particular, this means that no criterion based on acceptance rates can be used for the Gibbs sampler, contrarily to the Metropolis–Hastings algorithm.

One of the main advantages of this approach compared to the Metropolis–Hastings algorithm is that information contained in the target distribution π – through the full conditional distributions – is directly used and there is no need to rely on a proposal distribution arbitrarily chosen. Furthermore, it allows to decompose problems of high dimensions into problems of lower dimensions.

Algorithm 4 Gibbs algorithm

```

Choose initial value  $z^0$ 
For  $t = 1 : M$ 
  For  $i = 1 : p$ 
    Sample  $z_i^* \sim \pi_i(\cdot | z_{-i}^t)$ 
    Set  $z^{t+1} = (z_{1:i-1}^{t+1}, z_i^*, z_{i+1:p}^t)$ 
  End
End

```

Adapted Metropolis within Gibbs

The estimation of model parameters is not the only concern in general state space models as some of the hidden states $x_{1:T}$ might also be of interest. For plant growth models, one is often interested in knowing the values of some biomasses or leaf areas, for instance. Even if the exact value θ of a perfect model were known and because of the process and observation noises, different simulations of the model would lead to different outputs. A joint estimation of parameters and hidden states is therefore required and the target distribution is $\pi(\theta, x_{1:T} | y_{1 \rightarrow T})$. A natural approach is to update the parameters conditionally to the current value of the hidden states and reciprocally. However the efficiency of such a scheme is compromised as soon as there exists a strong correlation between parameters and hidden states as shown by Liu et al. [1994] and Roberts and Sahu [1997], which is most often the case in plant growth models.

To overcome this issue, Fearnhead [2011] proposed the joint update of the parameters and hidden states as follows:

$$q(\theta^*, x_{1:T}^* | \theta^t, x_{1:T}^t) = q(\theta^* | \theta^t) q(x_{1:T}^* | \theta^*). \quad (3.44)$$

This way, the changes in the parameters can easily be controlled via $q(\theta^* | \theta^t)$ and, what is more, the candidate hidden states $x_{1:T}^*$ will be consistent with the candidate parameters θ^* thanks to the proposal $q(x_{1:T}^* | \theta^*)$. A classical choice for the proposal distribution of the hidden states amounts to sampling from $p(x_{1:T}^* | \theta^*)$ – i.e. to perform a model simulation. An improvement would consist in taking into account observations to optimize state space exploration and sample from $\pi(x_{1:T}^* | \theta^*, y_{1 \rightarrow T})$ instead; this will be described in Section 3.5.3. The acceptance probability can therefore be computed as:

$$\begin{aligned}
\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t)) &= 1 \wedge \frac{\pi(\theta^*, x_{1:T}^* | y_{1 \rightarrow T})}{\pi(\theta^t, x_{1:T}^t | y_{1 \rightarrow T})} \frac{q(\theta^t, x_{1:T}^t | \theta^*, x_{1:T}^*)}{q(\theta^*, x_{1:T}^* | \theta^t, x_{1:T}^t)} \\
&= 1 \wedge \frac{p(\theta^*, x_{1:T}^* | y_{1 \rightarrow T})}{p(\theta^t, x_{1:T}^t | y_{1 \rightarrow T})} \frac{q(\theta^t | \theta^*)}{q(\theta^* | \theta^t)} \frac{q(x_{1:T}^t | \theta^t)}{q(x_{1:T}^* | \theta^*)} \\
&= 1 \wedge \frac{p(y_{1 \rightarrow T} | \theta^*, x_{1:T}^*) p(x_{1:T}^* | \theta^*) p(\theta^*)}{p(y_{1 \rightarrow T} | \theta^t, x_{1:T}^t) p(x_{1:T}^t | \theta^t) p(\theta^t)} \frac{q(\theta^t | \theta^*)}{q(\theta^* | \theta^t)} \frac{p(x_{1:T}^t | \theta^t)}{p(x_{1:T}^* | \theta^*)} \\
&= 1 \wedge \frac{p(y_{1 \rightarrow T} | \theta^*, x_{1:T}^*)}{p(y_{1 \rightarrow T} | \theta^t, x_{1:T}^t)} \frac{p(\theta^*)}{p(\theta^t)} \frac{q(\theta^t | \theta^*)}{q(\theta^* | \theta^t)}
\end{aligned} \quad (3.45)$$

which can further be simplified to:

$$\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t)) = 1 \wedge \frac{p(y_{1 \rightarrow T} | \theta^*, x_{1:T}^*)}{p(y_{1 \rightarrow T} | \theta^t, x_{1:T}^t)} \frac{p(\theta^*)}{p(\theta^t)} \quad (3.46)$$

if the proposal distribution for the parameters is symmetric, as is usually the case.

Beaumont [2003] proposed to generate multiple samples from $p(x_{1:T}^* | \theta^*)$, which would amount to many model simulations. When dealing with complex plant growth models, whose simulation can be time-consuming, it is preferable from a numerical point of view to generate only once the hidden states. It has also been shown [Chen, 2014] that the estimation of the hidden states improved when the functional parameters had first converged. The strategy is therefore to first estimate the posterior distribution of the functional parameters, from which can be deduced the mean $\hat{\theta}$ and the covariance $\hat{\Sigma}^\theta$. The hidden states $x_{1:T}$ are then estimated either by setting the parameters value to the estimated mean $\hat{\theta}$ or by drawing samples from this distribution, or for convenience from $\mathcal{N}(\hat{\theta}, \hat{\Sigma}^\theta)$.

During the second step, at each MCMC iteration, the hidden states are estimated sequentially from $n = 1$ to $n = T$. The proposal distribution for the hidden states is taken to be the transition pdf of the model. The acceptance probability for the hidden state x_n is:

$$\alpha(x_n^t, x_n^*) = 1 \wedge \frac{\pi(x_n^* | x_{1:n-1}^{t+1}, x_{n+1:T}^t, \hat{\theta}, y_{1 \rightarrow T})}{\pi(x_n^t | x_{1:n-1}^{t+1}, x_{n+1:T}^t, \hat{\theta}, y_{1 \rightarrow T})} \frac{q(x_n^t | \hat{\theta}, x_{1:n-1}^{t+1}, x_{n+1:T}^t)}{q(x_n^* | \hat{\theta}, x_{1:n-1}^{t+1}, x_{n+1:T}^t)} \quad (3.47)$$

and based on both the Markov chain property and the choice for the proposal distribution can be simplified to:

$$\alpha(x_n^t, x_n^*) = 1 \wedge \frac{p(x_{n+1}^t | \hat{\theta}, x_n^*)}{p(x_{n+1}^t | \hat{\theta}, x_n^t)} \frac{p(y_n | \hat{\theta}, x_n^*)}{p(y_n | \hat{\theta}, x_n^t)}. \quad (3.48)$$

As previously mentioned, the parameters set can be resampled as $\tilde{\theta}_n^t \sim \mathcal{N}(\hat{\theta}, \hat{\Sigma}^\theta)$ instead of being kept fixed at the mean value $\hat{\theta}$. In this case both the hidden states and the parameters are jointly accepted at each iteration. The Adapted Metropolis-within-Gibbs algorithm is summarized in Algorithm 5.

Burn-in period and stopping criteria

During the first iterations of the Markov chain, the latter has not reached its stationary distribution, the first samples generated therefore do not belong to the target distribution and must be discarded. The period during which the Markov chain has not reached its stationary distribution is called the burn-in period. Determining its length and from when to consider the samples of the Markov chain to represent the target distribution is rather difficult because it depends not only on the prior distribution assigned to the estimated parameters but also on the nature of the target distribution. In practice, the burn-in period is adjusted empirically and the samples are kept when the Markov chain is believed to have reached its equilibrium.

A pragmatic and empirical way of assessing the convergence of the MCMC algorithm is to run multiple chains independently with different initial values. It has the advantage to allow for a visual identification of both the burn-in period and, if the chains converge to the same distribution, the ideal length of the chain.

The Gelman–Rubin criterion is also based on the comparison between different chains using the between-chains and within-chains variances B and W . Gelman and Rubin [1992] has shown that under certain stationary conditions an unbiased estimator of the marginal posterior variance is:

$$V = \frac{M-1}{M}W + \frac{K+1}{KM}B \quad (3.49)$$

where we recall that M is the number of iterations of the chain. The potential scale reduction factor is defined as the ratio between V and W , and if all the chains have converged it should be close to 1. Brooks and Gelman [1998] corrected it to account for sampling variability as:

$$R_c = \frac{d+3}{d+1} \frac{V}{W} \quad (3.50)$$

where d is the estimated degrees of freedom for a Student- t approximation to the posterior inference based upon the simulations and advised, that when $R_c^{1/2} < 1.2$, convergence can be assumed.

In practice, for time-consuming applications, we would like to free ourselves from running multiple Markov chains independently and focus on other criteria instead. The acceptance rate can be such a criterion of whether the chain has converged or not: despite some optimal values being suggested in the literature, the actual optimal acceptance rate depends on the shape of the posterior distribution and it is considered that a value between 0.1 and 0.5 remains acceptable.

A standard stopping rule is based on the mean estimates: the last n_a overlapped batch means, where each batch mean contains n_b iterations and n_c iterations are overlapped, are computed for each parameter and denoted by θ_i^r for $i \in \llbracket 1, d \rrbracket$ and $r \in \llbracket 1, n_a \rrbracket$. If the relative changes in the batch means are small enough, i.e. such that:

$$\max_{i \in \llbracket 1, d \rrbracket} \left(\frac{|\theta_i^{r+1} - \theta_i^r|}{\theta_i^r + \delta_1} \right) < \delta_2 \quad (3.51)$$

then the chain is assumed to have converged. Searle et al. [1992] suggested that $\delta = 1 \cdot 10^{-03}$ and $\delta_2 = 5 \cdot 10^{-04}$ and we followed the values suggested by Chen [2014]: $n_a = 4$, $n_b = 3,000$ and $n_c = 1,000$.

3.5.2 Sequential Monte Carlo methods

Sequential Monte Carlo (SMC) methods [Doucet et al., 2001] [Del Moral, 2004], also known as particle filters in the context of dynamic systems, offer a good alternative to the MCMC methods for the estimation of parameters and hidden states within a Bayesian framework in SSMs. All the more so as MCMC algorithms are more influenced by ill-chosen prior distributions. There are also practical reasons behind the choice of SMC methods for inference. In particular, many real world problems require online estimation as data arrive sequentially in time.

A very general problem of interest in the context of HMMs is to estimate the posterior distribution of the hidden states x_n at time step n given a sequence of observations $y_{1:m}$. Three cases can then be considered:

- if $n > m$, it amounts to predict the values of the hidden states based on past observations, this is a prediction problem;

Algorithm 5 Adapted Metropolis-within-Gibbs algorithm

Choose initial values for the parameters θ^0 and the hidden states $x_{1:T}^0 = h(x_0, u, \theta^0, \eta)$

Initialize the adaptive schemes variables:

$$\begin{cases} \gamma^0 &= 1/2 \\ \mu^0 &= \theta^0 \\ \Sigma^0 &= \Sigma^{\theta^0} \\ \lambda^0 &= 2.38^2/n_\theta \end{cases}.$$

For $t = 0 : M_1 - 1$

Sample $\theta^* \sim \mathcal{N}(\theta^t, \lambda^t \Sigma^t)$

Sample $x_{1:T}^* = h(x_0, u, \theta^*, \eta)$

Compute the acceptance probability:

$$\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t)) = 1 \wedge \frac{p(y_{1:T} | \theta^*, x_{1:T}^*) p(\theta^*)}{p(y_{1:T} | \theta^t, x_{1:T}^t) p(\theta^t)}$$

Set $(\theta^{t+1}, x_{1:T}^{t+1}) = (\theta^*, x_{1:T}^*)$ with probability $\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t))$

Update the adaptive scheme variables:

$$\begin{cases} \gamma^{t+1} &= (t+1)^{-1} \\ \mu^{t+1} &= \mu^t + \gamma^{t+1}(z^{t+1} - \mu^t) \\ \Sigma^{t+1} &= \Sigma^t + \gamma^{t+1} [(z^{t+1} - \mu^{t+1})(z^{t+1} - \mu^{t+1})^T - \Sigma^t] \\ \lambda^{t+1} &= \lambda^t \exp(\gamma^{t+1}[\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t)) - \alpha^*]) \end{cases}$$

With a burn-in period of B , compute the estimated mean and covariance for the parameters:

$$\begin{cases} \hat{\theta} &= (M_1 - B)^{-1} \sum_{t=B+1}^{M_1} \theta^t \\ \hat{\Sigma} &= (M_1 - B - 1)^{-1} \sum_{t=B+1}^{M_1} (\theta^t - \hat{\theta})^T (\theta^t - \hat{\theta}) \end{cases}$$

End

For $t = M_1 : M_2 - 1$

For $n = 1 : T$

Sample $\tilde{\theta}_n^{t+1} \sim \mathcal{N}(\hat{\theta}, \hat{\Sigma}^\theta)$

Sample $x_n^* = f_n(x_{n-1}^{t+1}, u_n, \tilde{\theta}_n^{t+1}, \eta_n)$

Compute the acceptance probability:

$$\alpha(x_n^t, x_n^*) = 1 \wedge \frac{p(x_{n+1}^t | \tilde{\theta}_n^{t+1}, x_n^*) p(y_n | \tilde{\theta}_n^{t+1}, x_n^*)}{p(x_{n+1}^t | \tilde{\theta}_n^{t+1}, x_n^t) p(y_n | \tilde{\theta}_n^{t+1}, x_n^t)}$$

Set $x_n^{t+1} = x_n^*$ with probability $\alpha(x_n^t, x_n^*)$

End

End

- if $n < m$, both past and future observations are used for the estimation of the state at time step n , this is a smoothing problem;
- if $n = m$, the observations right up to the current time are used for the estimation, this is a filtering problem.

In the following, only the filtering problem is of concern. Its history dates back to the development of the Kalman filter [Kalman, 1960] for state estimation in the case of linear systems with normal noises. When dealing with complex non-linear systems, analytical expressions cannot be derived anymore. The first attempts to use SMC methods for filtering in the context of non-linear models date back to Handschin and Mayne [1969]. They led the foundations of such methods by using a sequential version of the importance sampling technique, which amounted to generate samples according to a proposal distribution and then attributing an importance weight to each of this sample. The weighted distribution therefore obtained was supposed to approximate the target distribution. This procedure was sequential in the sense that there were no need to regenerate the whole trajectory of the states whenever a new observation was available. Rather the samples were forwarded sequentially in time. This method suffered from a major drawback that was later identified by Gordon et al. [1993] known as the weight degeneracy problem. From then on and as the computing capabilities began to increase sufficiently enough for satisfying numerical approximations in complex non-linear models, other filters began to emerge as generalizations of the standard Kalman Filter such as the extended Kalman filter (EKF) [Anderson and Moore, 1979], [Sorenson, 1985], the unscented Kalman filter [Julier and Uhlmann, 1997] [Quach et al., 2007] or the ensemble Kalman filter [Evensen, 1994], [Evensen, 2009] were developed as well as other methods belonging to the family of particle filters such as the regularized particle filter [LeGland et al., 1998] [Musso et al., 2001] or the convolution particle filter [Campillo and Rossi, 2006].

In SMC methods, the set of parameters is not constant throughout time anymore, as it is updated when observations are available. The value of the parameters at time step n will therefore be denoted θ_n . Since both the parameters and the hidden states are estimated, an augmented state comprising the two is defined: $z_n = (\theta_n, x_n)$. A filtering procedure comprises two main steps:

- prediction, also known as time update, is concerned with the distribution of the hidden state x_{n+1} at time step $n + 1$ based on the former observations:

$$p(z_{n+1}|y_{1:n}) = \int p(z_{n+1}|z_n) p(z_n|y_{1:n}) dz_n, \quad (3.52)$$

- correction, also known as measurement update, uses the new observation at time step $n + 1$ to update the distribution, this can be decomposed using Bayes rule as:

$$p(z_{n+1}|y_{1:n+1}) = \frac{p(z_{n+1}|y_{1:n}) p(y_{n+1}|z_{n+1})}{\int p(z_{n+1}|y_{1:n}) p(y_{n+1}|z_{n+1}) dz_{n+1}}. \quad (3.53)$$

Kalman filter

The Kalman filter deals with the case of linear models, which means that both the transition and observation functions are assumed to be linear, and the equations of the general state space model presented in Section 1.1 can be written in this case as:

$$\begin{cases} x_{n+1} &= F_n x_n + B_n u_n + \eta_n, \\ y_n &= H_n x_n + \xi_n. \end{cases} \quad (3.54)$$

Both the process and observation noises are assumed to be drawn from centered multivariate normal distributions:

$$\begin{cases} \eta_n &\sim \mathcal{N}(0, Q_n), \\ \xi_n &\sim \mathcal{N}(0, R_n). \end{cases} \quad (3.55)$$

Several standard shorthand notations in the context of Kalman filters are introduced. At time step n conditionally to observations up to time step j , the state estimate is denoted $\hat{z}_{n|j}$, the error for the state estimate is $\tilde{z}_{n|j} = z_n - \hat{z}_{n|j}$, and the error covariance matrix for the state estimate is $\hat{\Sigma}_{n|j}^z$. Equivalent notations $\hat{y}_{n|j}$, $\tilde{y}_{n|j}$ and $\hat{\Sigma}_{n|j}^y$ are introduced for the observations.

Let us recall that in practice experimental data is only available at certain time steps and the merged experimental timeline is:

$$\mathcal{O} = (t_k)_{k \in [1:O]} \in \mathbb{N}^{\mathcal{O}}. \quad (3.56)$$

The dual prediction-correction step thus happens between two experimental times t_k and t_{k+1} . By convention, $t_0 = 0$. At these experimental times, the content of experimental data might not be the same and the correction step at time step t_k only involves the relevant hidden states corresponding to the available observations through the observation function g_{t_k} .

The prediction therefore consists in determining the estimate at time step t_{k+1} given the observations up to time step t_k . This is done using the evolution equation, Equation 3.54, to obtain the predicted state estimate:

$$\hat{z}_{t_{k+1}|t_k} = \mathbb{E}(z_{t_{k+1}} | y_{1:t_k}), \quad (3.57)$$

the predicted error covariance matrix:

$$\hat{\Sigma}_{t_{k+1}|t_k}^z = \mathbb{E}(\tilde{z}_{t_{k+1}} \tilde{z}_{t_{k+1}}^T | y_{1:t_k}), \quad (3.58)$$

the predicted observation:

$$\hat{y}_{t_{k+1}|t_k} = \mathbb{E}(y_{t_{k+1}} | y_{1:t_k}), \quad (3.59)$$

and the associated covariance matrix:

$$\hat{\Sigma}_{t_{k+1}|t_k}^y = \mathbb{E}(\tilde{y}_{t_{k+1}} \tilde{y}_{t_{k+1}}^T | y_{1:t_k}). \quad (3.60)$$

Last but not least, the correlation matrix between the errors related to the state and observations estimates:

$$\hat{\Sigma}_{t_{k+1}|t_k}^{zy} = \mathbb{E}(\tilde{z}_{t_{k+1}|t_k} \tilde{y}_{t_{k+1}|t_k} | y_{1:t_k}^T). \quad (3.61)$$

All these quantities can be easily calculated using Equation 3.54. Then comes the correction step where it is made use of the last observation, as the updated state estimate is assumed to be a linear combination of the predicted state estimate $\hat{z}_{t_{k+1}|t_k}$ and the observation $y_{t_{k+1}}$. The objective of the Kalman filter is to obtain an estimate that:

- is unbiased, i.e. such that $\mathbb{E}(\tilde{z}_{t_k|t_k}) = 0$,
- minimizes the error covariance matrix of the state estimate $\hat{\Sigma}_{t_k|t_k}^z$.

Under such assumptions, it can be shown that the updated state estimate and its associated covariance matrix are:

$$\begin{cases} \hat{z}_{t_{k+1}|t_{k+1}} &= \hat{z}_{t_{k+1}|t_k} + K_{t_{k+1}}(y_{t_{k+1}} - \hat{y}_{t_{k+1}|t_k}) \\ \hat{\Sigma}_{t_{k+1}|t_{k+1}}^z &= \hat{\Sigma}_{t_{k+1}|t_k}^z - K_{t_{k+1}} \hat{\Sigma}_{t_{k+1}|t_k}^{zy} K_{t_{k+1}}^T \end{cases} \quad (3.62)$$

where $K_{t_{k+1}}$ is the so-called Kalman gain:

$$K_{t_{k+1}} = \hat{\Sigma}_{t_{k+1}|t_k}^{zy} \left(\hat{\Sigma}_{t_{k+1}|t_k}^y \right)^{-1}. \quad (3.63)$$

The Kalman filter was designed for linear systems, and this assumption made it unsuitable for filtering in the case of non-linear systems. A basic and immediate extension is to linearize both the transition functions f_n and the observation functions g_n . This approach led to the development of the linearized Kalman filter (when a nominal trajectory is available) and the Extended Kalman filter. In the case of highly non-linear systems, these filters can still provide very poor performance, mostly because the covariance is propagated through linearization of the underlying non-linear system. In the following, two extensions of the Kalman filters aiming to overcome this issue are presented.

Unscented Kalman filter

This filter, introduced by [Julier and Uhlmann \[1997\]](#), makes use of the unscented transform. The latter allows to use a set of deterministically sampled points, the so-called sigma-points, that are propagated through the non-linear state space equations before providing updated estimates for the mean and the covariance matrix. This allows to use information of higher order that was discarded in linearized filters and usually provide better estimates [[Gustafsson and Hendeby, 2012](#)].

The unscented transform was introduced by [Uhlmann \[1995\]](#) as a novel method for calculating the statistics of a random variable that is modified via a non-linear transformation. If x denotes a d -dimensional random variable following a normal distribution $\mathcal{N}(\mu, \Sigma)$, f is a non-linear Borel function and $y = f(x)$, then the mean and the covariance of y can be well approximated by a set of $2d + 1$ sigma-point s_i : these sigma-points are propagated through the true non-linear function to obtain $y_i = f(s_i)$ and the mean and covariance of

Algorithm 6 Unscented Kalman filter

Choose initial values for the parameters and states $z_0 = (x_0, \theta_0)$ and the associated covariance Σ_0^z

Set $\hat{z}_{0|0} = z_0$ and $\hat{\Sigma}_{0|0}^z = \Sigma_0^z$

For $k \in \llbracket 0, O - 1 \rrbracket$

Define the augmented mean $\hat{z}_{t_k|t_k}^a$ and covariance $\hat{\Sigma}_{t_k|t_k}^a$ to account for the process noise η (whose related covariance matrix is J_{t_k}) as:

$$\begin{cases} \hat{z}_{t_k|t_k}^a &= (\hat{z}_{t_k|t_k}, 0_{d_\eta}) \\ \hat{\Sigma}_{t_k|t_k}^a &= \text{blockdiag}\{\hat{\Sigma}_{t_k|t_k}^z, J_{t_k}\} \end{cases}$$

Define the $2d + 1$ sigma-points s_i , the mean weights w_i^m and the covariance weight w_i^c for the normal distribution $\mathcal{N}(\hat{z}_{t_k|t_k}^a, \hat{\Sigma}_{t_k|t_k}^a)$

Propagate the sigma-points through the model to obtain the predicted states $s_{t_{k+1}|t_k}^i = f_{t_{k+1}}(s_{t_k|t_k}^i)$

Compute the estimated mean for the predicted state and observation:

$$\begin{cases} \hat{z}_{t_{k+1}|t_k} &= \sum_{i=0}^{2d} w_i^m s_{t_{k+1}|t_k}^i \\ \hat{y}_{t_{k+1}|t_k} &= \sum_{i=0}^{2d} w_i^m g_{t_{k+1}}(s_{t_{k+1}|t_k}^i) \end{cases}$$

and the associated covariance matrices:

$$\begin{cases} \hat{\Sigma}_{t_{k+1}|t_k}^z &= \sum_{i=0}^{2d} w_i^c (s_{t_{k+1}|t_k}^i - \hat{z}_{t_{k+1}|t_k})(s_{t_{k+1}|t_k}^i - \hat{z}_{t_{k+1}|t_k})^T \\ \hat{\Sigma}_{t_{k+1}|t_k}^y &= \sum_{i=0}^{2d} w_i^c (g_{t_{k+1}}(s_{t_{k+1}|t_k}^i) - \hat{y}_{t_{k+1}|t_k})(g_{t_{k+1}}(s_{t_{k+1}|t_k}^i) - \hat{y}_{t_{k+1}|t_k})^T \\ \hat{\Sigma}_{t_{k+1}|t_k}^{zy} &= \sum_{i=0}^{2d} w_i^c (s_{t_{k+1}|t_k}^i - \hat{z}_{t_{k+1}|t_k})(g_{t_{k+1}}(s_{t_{k+1}|t_k}^i) - \hat{y}_{t_{k+1}|t_k})^T \end{cases}$$

Compute the Kalman gain:

$$K_{t_{k+1}} = \hat{\Sigma}_{t_{k+1}|t_k}^{zy} \left(\hat{\Sigma}_{t_{k+1}|t_k}^y \right)^{-1}$$

Compute the updated estimate for the state and the covariance matrix using the Kalman formulas:

$$\begin{cases} \hat{z}_{t_{k+1}|t_{k+1}} &= \hat{z}_{t_{k+1}|t_k} + K_{t_{k+1}} (y_{t_{k+1}} - \hat{y}_{t_{k+1}|t_k}) \\ \hat{\Sigma}_{t_{k+1}|t_{k+1}}^z &= \hat{\Sigma}_{t_{k+1}|t_k}^z - K_{t_{k+1}} \hat{\Sigma}_{t_{k+1}|t_k}^y K_{t_{k+1}}^T \end{cases}$$

End

y are approximated using the y_i and appropriate weights w_i^m and w_i^c for the mean and covariance respectively. The sigma-points are defined as functions of the mean μ and the covariance matrix Σ to represent the distribution of x :

$$\begin{cases} s_0 &= \mu \\ s_i &= \mu + \left(\sqrt{(d + \lambda)\Sigma} \right)_i \\ s_{d+i} &= \mu - \left(\sqrt{(d + \lambda)\Sigma} \right)_i \end{cases} \quad (3.64)$$

where $\lambda = \alpha^2(d + \kappa) - d$ is a scaling parameter, α is representative of the spread of the sigma points around

the mean μ ; usual values are such that $10^{-4} \leq \alpha \leq 1$ and $\kappa \in 0, 3 - d$, see [Julier and Uhlmann, 1997] for more details, $\left(\sqrt{(d + \lambda)\Sigma}\right)_i$ is the i -th column of $\sqrt{(d + \lambda)\Sigma}$, and the corresponding weights are given by:

$$\begin{cases} w_0^m &= (d + \lambda)^{-1} \lambda \\ w_i^m &= 2^{-1}(d + \lambda)^{-1} \\ w_{d+i}^m &= 2^{-1}(d + \lambda)^{-1} \end{cases} \quad (3.65)$$

for the mean and by:

$$\begin{cases} w_0^c &= (d + \lambda)^{-1} \lambda + (1 - \alpha^2 + \beta) \\ w_i^c &= 2^{-1}(d + \lambda)^{-1} \\ w_{d+i}^c &= 2^{-1}(d + \lambda)^{-1} \end{cases} \quad (3.66)$$

for the covariance, which finally provides the estimate for the expectation of y :

$$\begin{cases} \mathbb{E}(y) &= \mathbb{E}(f(x)) \approx \sum_{i=0}^{2d} w_i^m f(s_i), \\ \mathbb{C}(y) &= \mathbb{C}(f(x)) \approx \sum_{i=0}^{2d} w_i^m (f(s_i) - \mathbb{E}(y))(f(s_i) - \mathbb{E}(y))^T. \end{cases} \quad (3.67)$$

In linearized versions of the Kalman filter such as the EKF, the state distribution is approximated by a normal distribution and then propagated through a first-order linearization of the model, leading to potentially large errors in the posterior mean and covariance. In the UKF, the state distribution is again approximated by a normal distribution but the carefully chosen sigma-points propagated through the true non-linear system are able to capture the posterior mean and covariance up to the second order whatever the non-linearities involved.

Ensemble Kalman filter

Another solution to the limitations of the linearized versions of the Kalman filter for non-linear systems is to use a Monte Carlo based approach. The ensemble Kalman filter (EnKF) was introduced by Evensen [1994] and, just as in the case of the UKF, it is based on an approximation of the state distribution by a Gaussian distribution. However, contrary to the UKF that uses a fixed set of deterministically sampled points to approximate the latter, the $2d + 1$ sigma-points, the EnKF uses a large number of samples randomly drawn from the normal distribution, whence the name of this filter, as the population of samples constitutes an ensemble of possible parameters and hidden states vectors. With the UKF, the accuracy of the estimates is fixed as both the number of sigma-points and their values is fixed as well. With the EnKF, this accuracy can be increased with the ensemble size. Of course, this also comes at the cost of a higher computing time. Unlike in the UKF, the estimations are not weighted and samples are simply averaged.

Algorithm 7 Ensemble Kalman filter

Choose initial values for the parameters and states $z_0 = (x_0, \theta_0)$ and the associated covariance Σ_0^z

Set $\hat{z}_{0|0} = z_0$ and $\hat{\Sigma}_{0|0}^z = \Sigma_0^z$

For $i = 1 : N$

Sample $z_0^i \sim \mathcal{N}(\hat{z}_{0|0}, \hat{\Sigma}_{0|0}^z)$

End

For $k \in \llbracket 0, O - 1 \rrbracket$

For $i = 1 : N$

Propagate the i -th sample through the model to obtain the predicted states $z_{t_{k+1}|t_k}^i = f_{t_{k+1}}(z_{t_k|t_k}^i)$

End

Compute the estimated mean for the predicted state and observation:

$$\begin{cases} \hat{z}_{t_{k+1}|t_k} &= N^{-1} \sum_{i=1}^N z_{t_{k+1}|t_k}^i \\ \hat{y}_{t_{k+1}|t_k} &= N^{-1} \sum_{i=1}^N g_{t_{k+1}}(z_{t_{k+1}|t_k}^i) \end{cases}$$

and the associated covariance matrices:

$$\begin{cases} \hat{\Sigma}_{t_{k+1}|t_k}^z &= (N-1)^{-1} \sum_{i=1}^N (z_{t_{k+1}|t_k}^i - \hat{z}_{t_{k+1}|t_k})(z_{t_{k+1}|t_k}^i - \hat{z}_{t_{k+1}|t_k})^T \\ \hat{\Sigma}_{t_{k+1}|t_k}^y &= (N-1)^{-1} \sum_{i=1}^N (g_{t_{k+1}}(z_{t_{k+1}|t_k}^i) - \hat{y}_{t_{k+1}|t_k})(g_{t_{k+1}}(z_{t_{k+1}|t_k}^i) - \hat{y}_{t_{k+1}|t_k})^T \\ \hat{\Sigma}_{t_{k+1}|t_k}^{zy} &= (N-1)^{-1} \sum_{i=1}^N (z_{t_{k+1}|t_k}^i - \hat{z}_{t_{k+1}|t_k})(g_{t_{k+1}}(z_{t_{k+1}|t_k}^i) - \hat{y}_{t_{k+1}|t_k})^T \end{cases}$$

Compute the Kalman gain:

$$K_{t_{k+1}} = \hat{\Sigma}_{t_{k+1}|t_k}^{zy} \left(\hat{\Sigma}_{t_{k+1}|t_k}^y \right)^{-1}$$

For $i = 1 : N$

Compute the updated state estimate for the i -th sample using Kalman formula:

$$\hat{z}_{t_{k+1}|t_{k+1}}^i = \hat{z}_{t_{k+1}|t_k}^i + K_{t_{k+1}} \left(y_{t_{k+1}} - g_{t_{k+1}}(z_{t_{k+1}|t_k}^i) \right)$$

End

Compute the updated estimate for the state and the covariance matrix:

$$\begin{cases} \hat{z}_{t_{k+1}|t_{k+1}} &= N^{-1} \sum_{i=1}^N \hat{z}_{t_{k+1}|t_{k+1}}^i \\ \hat{\Sigma}_{t_{k+1}|t_{k+1}}^z &= (N-1)^{-1} \sum_{i=1}^N \left(z_{t_{k+1}|t_{k+1}}^i - \hat{z}_{t_{k+1}|t_{k+1}} \right) \left(z_{t_{k+1}|t_{k+1}}^i - \hat{z}_{t_{k+1}|t_{k+1}} \right)^T \end{cases}$$

End

Particle filters

The SMC methods presented so far are all extensions of the original Kalman approach developed in 1960. For each of these methods, parameters and hidden states were propagated through the model allowing to compute predicted values as well as covariance and correlation matrices, which served to update the estimates through

linear algebra formulas. The approach developed by particle filters is different: they aim at providing better estimates using a population of particles where each is associated to a weight measuring its relevance with respect to the experimental data. Here, a particle again designates a set of parameters and hidden states at a given time.

Before describing the different particle filters, we first take a brief detour through importance sampling [Robert and Casella, 1999]. If p is a probability distribution from which it is not possible to sample but such that $p(x) \propto \pi(x)$ and π can easily be evaluated, and if $x^i \sim q(x)$ are sampled from a proposal distribution q whose support includes that of p , called the importance density, then a weighted approximation to p is given by:

$$p(x) \approx \sum_{i=1}^N w^i \delta(x - x^i) \quad (3.68)$$

where the normalized weight of the i -th sample is given by:

$$w^i \propto \frac{\pi(x^i)}{q(x^i)}. \quad (3.69)$$

Importance sampling thus allows to evaluate integrals such as:

$$\mathbb{E}(f(x)) = \int f(x)p(x)dx \quad (3.70)$$

when it is not possible to easily sample from p . Instead, one samples from the importance density and rewrite the last equation as:

$$\mathbb{E}(f(x)) = \frac{\int f(x) \frac{p(x)}{q(x)} q(x) dx}{\int \frac{p(x)}{q(x)} q(x) dx} \quad (3.71)$$

which can further be approximated as:

$$\mathbb{E}(f(x)) \approx \frac{\frac{1}{N} \sum_{i=1}^N f(x^i) \frac{p(x^i)}{q(x^i)}}{\frac{1}{N} \sum_{i=1}^N \frac{p(x^i)}{q(x^i)}} = \frac{\frac{1}{N} \sum_{i=1}^N f(x^i) \frac{\pi(x^i)}{q(x^i)}}{\frac{1}{N} \sum_{i=1}^N \frac{\pi(x^i)}{q(x^i)}}. \quad (3.72)$$

Defining the importance weight corresponding to the i -th sample:

$$w^i = \frac{\frac{\pi(x^i)}{q(x^i)}}{\sum_{j=1}^N \frac{\pi(x^j)}{q(x^j)}}, \quad (3.73)$$

Equation 3.72 yields the weighted formulation:

$$\mathbb{E}(f(x)) \approx \sum_{i=1}^N w^i f(x^i). \quad (3.74)$$

Coming back to particle filters, the most standard one is known as sequential importance sampling (SIS) and was first derived by Gordon et al. [1993]. It goes by other denominations such as, originally, survival of the fittest.

The population of particles is comprised of N samples, the i -th particle being $z_n^i = (\theta_n^i, x_n^i)$ and its associated weight w_n^i . This provides a discrete weighted estimate of the posterior density at time step n :

$$p(z_n|y_{1:n}) \approx \sum_{i=1}^N w_n^i \delta(z_n - z_n^i). \quad (3.75)$$

The weights w_n^i are determined using importance sampling: at time step n , the importance weight for the i -th particle is defined as:

$$w_n^i \propto \frac{p(z_{1:n}^i|y_{1:n})}{q(z_{1:n}^i|y_{1:n})} \quad (3.76)$$

where q is the importance density. How to choose q is of primary importance and will be detailed thereafter. It can indeed be taken advantage of the Markov structure of the general state space framework considered to define recursively the importance density and the corresponding weights at time step n . For this purpose, the importance density q is decomposed using another importance density, called the importance transition density and denoted by $\rho_n(z_{n+1}|z_n)$. It allows to propagate the particle from time step n to $n+1$. The importance density can therefore be defined such that:

$$q_{n+1}(z_{1:n+1}|y_{1:n}) = q_n(z_{1:n}|y_{1:n}) \rho_{n+1}(z_{n+1}|z_n) = \rho_0(z^0) \prod_{k=1}^n \rho_{k+1}(z_{k+1}|z_k). \quad (3.77)$$

Furthermore, the properties of general SSMs for the transition and observation pdf together with Bayes' rule allows to formulate the following formula for Bayesian recursive estimation:

$$\begin{aligned} p(z_{1:n+1}|y_{1:n+1}) &= \frac{p(y_{1:n+1}|z_{1:n+1}) p(z_{1:n+1})}{p(y_{1:n+1})} \\ &= \frac{p(y_{n+1}, y_{1:n}|z_{1:n+1}) p(z_{1:n+1})}{p(y_{n+1}, y_{1:n})} \\ &= \frac{p(y_{n+1}|y_{1:n}, z_{1:n+1}) p(y_{1:n}|z_{1:n+1}) p(z_{1:n+1})}{p(y_{n+1}|y_{1:n}) p(y_{1:n})} \\ &= \frac{p(y_{n+1}|z_{n+1}) \left[p(z_{1:n+1}|y_{1:n}) p(y_{1:n}) p(z_{1:n+1})^{-1} \right] p(z_{1:n+1})}{p(y_{n+1}|y_{1:n}) p(y_{1:n})} \\ &= \frac{p(y_{n+1}|z_{n+1}) p(z_{n+1}, z_{1:n}|y_{1:n})}{p(y_{n+1}|y_{1:n})} \\ &= \frac{p(y_{n+1}|z_{n+1}) p(z_{n+1}|z_{1:n}, y_{1:n}) p(z_{1:n}|y_{1:n})}{p(y_{n+1}|y_{1:n})} \\ &= p(z_{1:n}|y_{1:n}) \frac{p(y_{n+1}|z_{n+1}) p(z_{n+1}|z_n)}{p(y_{n+1}|y_{1:n})}. \end{aligned} \quad (3.78)$$

Hence, the expression of the weights can be simplified to:

$$\begin{aligned} w_{n+1}^i &= \frac{p(z_{1:n+1}|y_{1:n+1})}{q_{n+1}(z_{1:n+1}|y_{1:n+1})} \\ &= \frac{p(z_{1:n}|y_{1:n})}{q_{n+1}(z_{1:n}|y_{1:n})} \frac{p(y_{n+1}|z_{n+1}) p(z_{n+1}|z_n)}{p(y_{n+1}|y_{1:n}) \rho_{n+1}(z_{n+1}|z_n)} \\ &= w_n^i \frac{p(y_{n+1}|z_{n+1}) p(z_{n+1}|z_n)}{p(y_{n+1}|y_{1:n}) \rho_{n+1}(z_{n+1}|z_n)}. \end{aligned} \quad (3.79)$$

In the latter fraction, all elements can be easily calculated (the transition pdf and the observation pdf are known from the model used, and ρ_{n+1} , the importance transition density, has been explicitly chosen) except from the term $p(y_{1:n+1}|y_{1:n})$. However, since this term is the same for every particle, there is no need to calculate it as the weights are further normalized as:

$$w_{n+1}^i = \frac{\tilde{w}_{n+1}^i}{\sum_{j=1}^N \tilde{w}_{n+1}^j} \quad (3.80)$$

where:

$$\tilde{w}_{n+1}^i = w_n^i \frac{p(z_{n+1}|z_n)}{p(y_{n+1}|y_{1:n}) \rho_{n+1}(z_{n+1}|z_n)}. \quad (3.81)$$

A well-known problem with the SIS algorithm is that of weight degeneracy [Doucet et al., 2001] first highlighted by Gordon et al. [1993]. For a sufficient number of observations, particle weights will differ by orders of magnitude depending on the distance between the state of the particle and the observation since:

$$w_n^i \propto \prod_{k \in \llbracket 1, O \rrbracket | t_k < n} p(y_{t_k} | x_{t_k}^i), \quad (3.82)$$

and the variance of the weights will increase at every step. This suggests that a few particles (only one in the extreme case) will carry the weight and all others will have importance weights close to zero. A simple method to avoid this degeneracy is to resample particles from time to time with replacement probabilities w_n^i for particle i . Several strategies have been designed for resampling: one is to resample particles at every time step, another one is to resample only when some appropriate measure indicates to do so. One such indicator is the effective sample size (ESS) [Kong et al., 1994], defined as:

$$n_{\text{eff}} = \frac{\left(\sum_{i=1}^N w_n^i \right)^2}{\sum_{i=1}^N (w_n^i)^2} \quad (3.83)$$

which simplifies to:

$$n_{\text{eff}} = \frac{1}{\sum_{i=1}^N (w_n^i)^2} \quad (3.84)$$

with normalized weights. It is a measure of the effective number of particles; this is better seen with extreme case scenarios: if all particles have the same weight $1/N$ they have the same importance and $n_{\text{eff}} = N$, whereas if only one particle has a non-zero weight 1, then $n_{\text{eff}} = 1$. Therefore, when the effective sample size falls below a given threshold, one might decide to resample because the population is judged to be degenerate. The resampling frequency has been discussed in [Kong et al., 1994], [Kitagawa, 1996] and more recently in [Carmier et al., 2017].

The resampling procedure per se can be done in several ways. A comparison of the four main resampling schemes (multinomial, residual, stratified and systematic) has been done by Douc and Cappe [2005]. It is shown that in practical SMC applications, the different schemes provide comparable results, however, systematic resampling is preferred because of its numerical efficiency and since it is based on a deterministic algorithm.

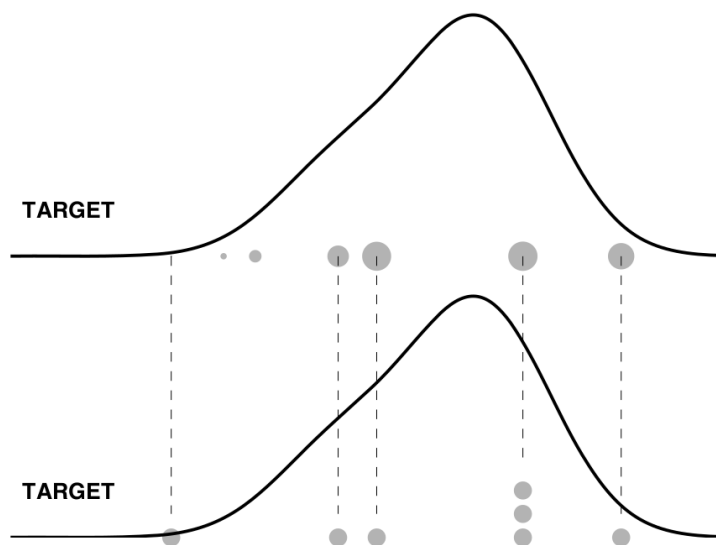


Figure 3.1: Principle of resampling. Top: the filled circles represent the samples drawn from the proposal distribution, their radii being proportional to the associated normalized importance weights (the target distribution is plotted in solid line). Bottom: after resampling, all points have the same importance weight, some of them being duplicated. Original image from [Cappé et al., 2005].

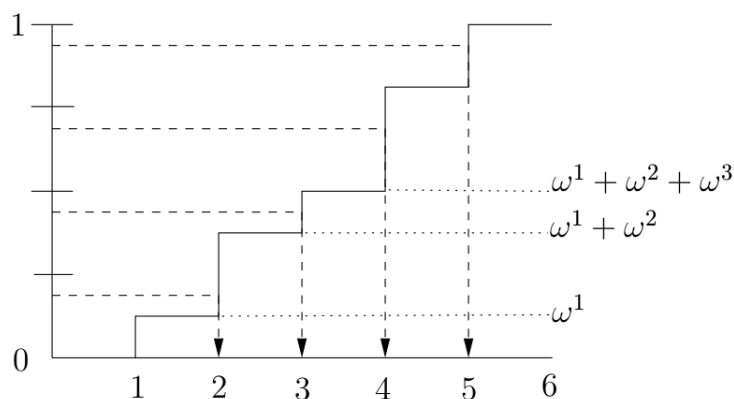


Figure 3.2: Systematic sampling: the unit interval is divided into N intervals $((i-1)/N, i/N]$ and one sample is selected from each of them as $U^i = U + (i-1)/N$ where $U \sim \mathcal{U}(0, 1/N)$ (only one random variable is therefore actually sampled). The U^i 's are represented with dashed lines. Indices are computed as $I^i = D^{\text{inv}}(U^i)$, where D^{inv} is the inverse of the cumulative distribution function associated with the w^i 's, i.e. $D^{\text{inv}}(u) = i$ for $u \in (\sum_{j=1}^{i-1} w^j, \sum_{j=1}^i w^j]$ and the bounds of this interval are represented with dotted lines. Original image from [Cappé et al., 2005].

The SIS algorithm combined to a resampling procedure constitutes the sequential importance resampling (SIR) algorithm, also known as the bootstrap filter. However, resampling from the discrete weighted estimate of Equation 3.75 only duplicates particles and leads to the so-called sample impoverishment.

Regularized particle filter

To tackle this issue, LeGland et al. [1998] proposed to regularize the discrete weighted estimate by using a Parzen–Rozenblatt kernel density estimator [Parzen, 1962]. The main idea is to smooth the discrete estimate by replacing the Dirac measure with another that is able to interpolate between all particles. A kernel k is an application that must be positive, bounded, symmetric, integrable and such that its integral on the whole space must be 1. This ensures that the kernel density estimator is a pdf. If k is a kernel, it is worth noting that:

$$\begin{cases} k_h & : \mathbb{R}^d \rightarrow \mathbb{R} \\ & x \rightarrow \frac{1}{h^d} k\left(\frac{x}{h}\right) \end{cases} \quad (3.85)$$

is also a kernel where $h > 0$ is a smoothing factor known as the bandwidth parameter. The regularized particle filter (RPF) uses this idea of kernel smoothing to turn the discrete estimate 3.75 into an absolutely continuous one. Details on the RPF can be found in [Musso and Oudjane, 1998], [Oudjane and Musso, 1999]. This regularization takes place after the correction step of the filter and the nature of the kernels is usually either Epanechnikov or Gaussian. The Epanechnikov kernel, which is nothing but a parabolic function, enjoys a good reputation because it is the optimal kernel in terms of minimization of the asymptotic mean integrated squared error [Epanechnikov, 1969], making it sometimes known as the optimal kernel. However, the gain is rather small in practice [Wand and Jones, 1994] and owing to its very convenient mathematical properties, its non-finite support, its more flexible variance and given how easy it is to sample from a normal distribution, the kernels used in the following will all be Gaussian. The regularized estimator for the posterior pdf can therefore be written as:

$$\hat{p}(z_n | y_{1:n}) = \sum_{i=1}^N w_n^i k_{h_n}(z_n - z_n^i) \quad (3.86)$$

where:

$$\begin{cases} w_n^i & \propto p(y_n | z_n^i) / \sum_{j=1}^N p(y_n | z_n^j), \\ k_{h_n} & = f_{\mathcal{N}(0, h_n^2)}. \end{cases} \quad (3.87)$$

If $p(y_n | z_n)$ is unknown, it can also be estimated using a kernel density approach [Chen, 2014]. This is the main motivation behind the design of the convolution particle filter (CPF) [Rossi and Vila, 2006], [Campillo and Rossi, 2006] which approximates:

$$\hat{p}(z_{n+1} | y_{1:n+1}) = \frac{\sum_{i=1}^N k_{h_{n+1}^x}(z_{n+1} - z_{n+1|n}^i) k_{h_{n+1}^y}(y_{n+1} - y_{n+1|n}^i)}{\sum_{i=1}^N k_{h_{n+1}^y}(y_{n+1} - y_{n+1|n}^i)} \quad (3.88)$$

with Parzen–Rozenblatt kernels k and bandwidths h_n^x and h_n^y for the particles and observations respectively. Equation 3.86 is thus nothing but a mixture of normal distributions, and resampling from this estimate

amounts to a two-step procedure:

- first the particles are resampled (when necessary in accordance with the effective sampling size) using systematic resampling;
- second, these particles are perturbed around the previous values and where the size of the perturbation is controlled by the bandwidth h_n , which reduces to sampling from a Gaussian kernel.

This procedure prevents the sample impoverishment problem since all particles generated in this manner are unique. In practice, in order to avoid a computationally expensive copy of all the parameters and states for the resampling, a list of indices $(j^i)_{i \in \llbracket 1, N \rrbracket}$ can be computed so that if the list of particles before resampling is $(z^i)_{i \in \llbracket 1, N \rrbracket}$, then $(z^{j^i})_{i \in \llbracket 1, N \rrbracket}$ is such that particles are resampled with the adequate probability.

The choice of the bandwidth at every time step h_n is rather important and has been discussed notably by [Musso et al. \[2001\]](#). They proposed an optimal bandwidth parameter when the kernels are Gaussian so as to minimize the mean integrated square error between the posterior distribution and the regularized weighted empirical measure:

$$h_n^2 = \alpha_h \Sigma_n = \left(\frac{4}{N(d+2)} \right)^{\frac{1}{d+4}} \Sigma_n. \quad (3.89)$$

Since the true covariance matrix Σ_n is unknown, in practice it is replaced with an estimate $\hat{\Sigma}_n$ based on the weighted samples:

$$\begin{cases} \hat{\mu}_{t_{k+1}} &= \sum_{i=1}^N w_{t_{k+1}}^i z_{t_{k+1}|t_k}^i, \\ \hat{\Sigma}_{t_{k+1}} &= \frac{N}{N-1} \sum_{i=1}^N w_{t_{k+1}}^i (z_{t_{k+1}|t_k}^i - \hat{\mu}_{t_{k+1}})^2. \end{cases} \quad (3.90)$$

More recently, [Carmier et al. \[2017\]](#) suggested a different strategy for the choice of the bandwidth. Observing that the RPF estimator exhibits a systematic deviation from the true posterior density increasing with the number of observations, they proposed an adaptive procedure with a sequence $\alpha_n = (n + \alpha_h^{-1})^{-1}$.

3.5.3 Particle Markov chain Monte Carlo

Estimation methods from the MCMC and SMC families have gained an increasing amount of attention over the past two decades, in particular for the estimation of hidden states, and have been expecting to deal with problems more complex and of higher dimensionality. The latter highlighted some of the limitations of the MCMC and SMC methods, and this saw the birth of a new family of methods combining the previous two, namely particle Markov chain Monte Carlo (PMCMC) methods that rely on their respective strengths [\[Andrieu et al., 2010\]](#). They aim at overcoming possible shortcomings of standard MCMC algorithms when the proposal distributions to explore the state space are poorly chosen or if highly correlated variables are updated independently. One of the key issues in MCMC methods is to manage to sample from a target distribution, this is achieved in practice by sampling from a proposal distribution, which becomes all the more crucial in the context of complex and high-dimensional models where such a choice might become difficult: indeed, there is always a compromise to find between ease of implementation and integrating the

Algorithm 8 Regularized particle filter**For** $i = 1 : N$ Initialize the i -th particle by sampling parameters and states from the prior distributions $\theta_0^i \sim p(\theta)$ and $x_0^i \sim p(x_0)$ and setting its weight to $w^i = N^{-1}$ **End****For** $k \in \llbracket 0, O - 1 \rrbracket$ **For** $i = 1 : N$ Propagate the i -th particle to estimate the predicted value $z_{t_{k+1}|t_k}^i \sim p(z_{t_{k+1}} | z_{t_k}^i)$ Sample the predicted observation $y_{t_{k+1}|t_k}^i \sim p(y_{t_{k+1}} | z_{t_{k+1}|t_k}^i)$ Compute the associated weight $\tilde{w}_{t_{k+1}}^i = w_{t_k}^i p(y_{t_{k+1}} | z_{t_{k+1}|t_k}^i)$ **End**

Normalize the weights:

$$w_{t_{k+1}}^i = \frac{\tilde{w}_{t_{k+1}}^i}{\sum_{j=1}^N \tilde{w}_{t_{k+1}}^j}$$

Compute the effective sample size:

$$n_{\text{eff}} = \left(\sum_{i=1}^N (w_n^i)^2 \right)^{-1}$$

If $n_{\text{eff}} < \lambda N$ Create the list of indices for the systematic resampling of particles with probability w_n^i

Resample particles by replacing parameters and states according to the list of indices

Compute the updated estimate for the mean and covariance matrix:

$$\begin{cases} \hat{z}_{t_{k+1}|t_{k+1}} &= N^{-1} \sum_{i=1}^N z_{t_{k+1}|t_{k+1}}^i \\ \hat{\Sigma}_{t_{k+1}|t_{k+1}}^z &= (N-1)^{-1} \sum_{i=1}^N (z_{t_{k+1}|t_{k+1}}^i - \hat{z}_{t_{k+1}|t_{k+1}})(z_{t_{k+1}|t_{k+1}}^i - \hat{z}_{t_{k+1}|t_{k+1}})^T \end{cases}$$

For $i = 1 : N$

Regularize particles by adding a small perturbation according to:

$$z_{t_{k+1}|t_{k+1}}^i += \alpha_h \zeta_n^i \text{ with } \zeta_n^i \sim \mathcal{N}(0, \hat{\Sigma}_{t_{k+1}|t_{k+1}}^z)$$

Reset weight $w_{t_{k+1}}^i = 1/N$ **End****End****End**

potentially complicated shape of the target distribution. PMCMC algorithms overcome these difficulties by providing a general sampling method that requires no special design from the user. The depletion drawback of the SMC methods, even though partially solved with the resampling and regularization steps, remains of importance in particular because the samples obtained at early time steps will never be resampled. PMCMC methods are not subject to depletion problems and do not necessitate that the SMC algorithms used inside provide an accurate estimate of $p(x_{1:T} | y_{1 \rightarrow T})$ but simply a sample from an approximation of the latter.

The practice of using MCMC methods for updating blocks of states between available experimental data, as in Algorithm 5, has its own limitations and it becomes difficult to sample from efficient proposal distributions as the observations become scarcer. A natural strategy to obtain such efficient proposal distributions is to use

an SMC method to sample from an approximation of $p(x_{1:T}|y_{1 \rightarrow T})$ at each iteration of an MCMC loop.

Assuming one is interested in sampling from $p(\theta, x_{1:T}|y_{1 \rightarrow T})$. We recall that a standard approach to jointly update θ and $x_{1:T}$ is the following decomposition:

$$q(\theta^*, x_{1:T}^*|\theta^t, x_{1:T}^t) = q(\theta^*|\theta^t) q(x_{1:T}^*|\theta^*). \quad (3.91)$$

In the AMWG algorithm, we previously considered that sampling states directly from $p(x_{1:T}|\theta, y_{1 \rightarrow T})$ was not directly feasible, which is why we chose to sample from $p(x_{1:T}|\theta)$ instead. Were it possible to sample from the former distribution, the corresponding acceptance probability would be:

$$\begin{aligned} \alpha &= \frac{p(\theta^*, x_{1:T}^*|y_{1 \rightarrow T})}{p(\theta^t, x_{1:T}^t|y_{1 \rightarrow T})} \frac{q(\theta^t, x_{1:T}^t|\theta^*, x^*)}{q(\theta^*, x_{1:T}^*|\theta^t, x_{1:T}^t)} \\ &= \frac{p(\theta^*, x_{1:T}^*|y_{1 \rightarrow T})}{p(\theta^t, x_{1:T}^t|y_{1 \rightarrow T})} \frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)} \frac{p(x_{1:T}^t|y_{1 \rightarrow T}, \theta^t)}{p(x_{1:T}^*|y_{1 \rightarrow T}, \theta^*)} \\ &= \frac{p(x_{1:T}^*|y_{1 \rightarrow T}, \theta^*)}{p(x_{1:T}^t|y_{1 \rightarrow T}, \theta^t)} \frac{p(\theta^*|y_{1 \rightarrow T})}{p(\theta^t|y_{1 \rightarrow T})} \frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)} \frac{p(x_{1:T}^t|y_{1 \rightarrow T}, \theta^t)}{p(x_{1:T}^*|y_{1 \rightarrow T}, \theta^*)} \\ &= \frac{p(\theta^*|y_{1 \rightarrow T})}{p(\theta^t|y_{1 \rightarrow T})} \frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)} \\ &= \frac{p(y_{1 \rightarrow T}|\theta^*)}{p(y_{1 \rightarrow T}|\theta^t)} \frac{p(\theta^*)}{p(\theta^t)} \frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)} \end{aligned} \quad (3.92)$$

ultimately showing that such an MCMC scheme essentially targets the marginal density $p(\theta|y_{1 \rightarrow T})$ which stems from its marginal Metropolis–Hastings (MMH) sampler, as already exploited in [Beaumont, 2003] and [Andrieu and Roberts, 2009]. A reasonable approximation to obtaining samples from $p(x_{1:T}|\theta, y_{1 \rightarrow T})$ is to use an SMC algorithm within this MMH sampler, whence the appellation particle marginal Metropolis–Hastings (PMMH) sampler. The adaptive scheme detailed in 3.40 is again used for efficient state space exploration and convergence and the proposal distribution for the parameters is again taken to be a multivariate normal distribution with mean θ^t and covariance $\lambda^t \Sigma^t$. The whole algorithm is detailed in Algorithm 9.

What has not come under scrutiny yet is the choice of the SMC algorithm used at each MCMC iteration. It must be sufficiently accurate to provide reliable samples from the target distribution $p(x_{1:T}|y_{1 \rightarrow T}, \theta)$, but one must not forget that such an SMC procedure is run at every iteration of the Markov chain, and the overall procedure can quickly become extremely expensive in terms of computing time and memory, in particular when dealing with complicated models that take a lot of time to simulate. This question will be the subject of a case study and will partly be answered in Chapter 6.

PMCMC algorithms continue to be of high interest in applications dealing with complex SSMs, large amounts of data and when the posterior distributions is highly multimodal. Some extensions have recently been proposed, such as interacting PMCMC algorithms [Rainforth et al., 2016], [Mingas et al., 2017] or by introducing new latent variables [Fearnhead and Meligkotsidou, 2016] for enhanced mixing rates. Their parallelization on CPU, GPU or field programmable gate arrays (FPGAs) is of primary importance given their computing cost.

Algorithm 9 Particle marginal Metropolis–Hastings algorithm

Choose initial values for the parameters θ^0

Run an SMC algorithm targeting $p(x_{1:T}|y_{1 \rightarrow T}, \theta^0)$ and sample $x_{1:T}^0 \sim \hat{p}(x_{1:T}|y_{1 \rightarrow T}, \theta^0)$

Initialize the adaptive schemes variables:

$$\begin{cases} \gamma^0 &= 1/2 \\ \mu^0 &= \theta^0 \\ \Sigma^0 &= \Sigma^0 \\ \lambda^0 &= 2.38^2/n_\theta \end{cases} .$$

For $t = 0 : M - 1$

Sample $\theta^* \sim \mathcal{N}(\theta^t, \lambda^t \Sigma^t)$

Run an SMC algorithm targeting $p(x_{1:T}|y_{1 \rightarrow T}, \theta^*)$ and sample $x_{1:T}^* \sim \hat{p}(x_{1:T}|y_{1 \rightarrow T}, \theta^*)$

Compute the acceptance probability:

$$\alpha = 1 \wedge \frac{p(y_{1 \rightarrow T}|\theta^*)}{p(y_{1 \rightarrow T}|\theta^t)} \frac{p(\theta^*)}{p(\theta^t)} \frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)}$$

Set $(\theta^{t+1}, x_{1:T}^{t+1}) = (\theta^*, x_{1:T}^*)$ with probability α

Update the adaptive scheme variables:

$$\begin{cases} \gamma^{t+1} &= (t+1)^{-1} \\ \mu^{t+1} &= \mu^t + \gamma^{t+1}(z^{t+1} - \mu^t) \\ \Sigma^{t+1} &= \Sigma^t + \gamma^{t+1} [(z^{t+1} - \mu^{t+1})(z^{t+1} - \mu^{t+1})' - \Sigma^t] \\ \lambda^{t+1} &= \lambda^t \exp(\gamma^{t+1}[\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t)) - \alpha^*]) \end{cases}$$

With a burn-in period of B , compute the estimated mean and covariance for the parameters:

$$\begin{cases} \hat{\theta} &= (M-B)^{-1} \sum_{t=B+1}^M \theta^t \\ \hat{\Sigma}^\theta &= (M-B-1)^{-1} \sum_{t=B+1}^M (\theta^t - \hat{\theta})(\theta^t - \hat{\theta})^T \end{cases}$$

and the hidden states:

$$\begin{cases} \hat{x}_{1:T} &= (M-B)^{-1} \sum_{t=B+1}^M x_{1:T}^t \\ \hat{\Sigma}^x &= (M-B-1)^{-1} \sum_{t=B+1}^M (x_{1:T}^t - \hat{x}_{1:T})(x_{1:T}^t - \hat{x}_{1:T})^T \end{cases}$$

End

Chapter 4

Estimation of parameters within population models

PARAMETER INFERENCE WITHIN A POPULATION CONTEXT requires different methods of estimation taking into account the intraindividual and interindividual variabilities of plants, as the estimation methods presented in the previous chapter for a single individual will not be sufficient to explain them. Bayesian hierarchical parameter estimation of non-linear models have gained some attention over the last decade and this has been witnessed in the field of plants, among others, for different purposes.

[Schneider et al. \[2006\]](#) used them in the context of competition between different individuals in populations of *Arabidopsis thaliana* to estimate functions called competition kernels which describe the dynamics of the neighbouring effects, thereby highlighting their relevance for the analysis of complex spatially dependent data. Bayesian hierarchical models appear particularly suitable for modelling spatial patterns of tree density [[Mortier et al., 2007](#)] as they notably allow to decompose complex biological data as a series of simpler conditional models, since it is a crucial issue in spatial modelling to take into account possible autocorrelation between the different observations. Still in the context of georeferenced data sets, they have also been used for the prediction of dependent non-Gaussian spatial fields [[Chagneau et al., 2011](#)]. [Boone et al. \[2008\]](#) used a Bayesian hierarchical modelling regression model in order to locate the QTL mapping associated with cotyledon opening in a population of *Arabidopsis thaliana*. [Patrick et al. \[2009\]](#) highlighted the utility of such a framework on a model of photosynthesis to leaf gas exchange for which the photosynthetic parameters were estimated. They were able to highlight the importance of the variations related to plants and species and showed that such a variability should be accounted for when using this kind of model for either prediction or inference. More recently, in the context of decentralized participatory plant breeding where different farms perform a selection experiment on its own with different designs, [Rivière et al. \[2015\]](#) showed that a Bayesian hierarchical modelling approach could lead to reliable mean comparisons between farms, which in practice allows for the evaluation of a wide range of diversity in the population of farms. Although they used a linear-bilinear model, [Jarquin et al. \[2016\]](#) took advantage of the Bayesian hierarchical framework for the evaluation of genotypic performance in different environmental conditions and the interpretation of genotype

by environment interaction for maize, which enabled them to use plant breeding data from different years and thus highlight genotype by environment patterns. Finally, [Wasson et al. \[2017\]](#) utilized this approach on field data for wheat genotypes to fit a soil model for the roots. They managed to obtain profile-specific traits representative of a genotype which could be used to link phenotypic traits with specific genotypes and further allow breeders to select the most appropriate root system for sustainable intensification and increased yields. All these works show the growing interest for this kind of model for parameter inference in the context of genotypic differentiation for plant growth models and the importance to study and understand them.

Some results originating from the frequentist point of view for population models and that were obtained in particular in the context of plant growth models are presented in Section 4.1. We then move on to introducing the Bayesian hierarchical model that will be used in this thesis from Section 4.2. In particular, how in the Bayesian paradigm the full conditional distributions of population parameters can be calculated as known statistical distributions by choosing appropriate prior distributions for these parameters (Sections 4.2.1 and 4.2.2). The case of linear models is considered before that of non-linear ones in Section 4.2.3 for which it is not possible to easily sample the individual parameters and alternative methods need to be devised (Section 4.2.4). The full adaptive hybrid Metropolis–Hastings–Gibbs algorithm that will be used for non-linear hierarchical models is finally presented in Section 4.2.5.

4.1 Frequentist approach

Many frequentist estimation methods for the estimation of mixed effect models have been proposed in the literature since the beginning of the 1980s. Notably, this kind of approach has already been applied to plant growth models, see [\[Baey, 2014\]](#) for instance. The first methods were based on the isolated estimation of each individual parameter set θ_i [\[Davidian and Giltinan, 1993\]](#). This can be done by applying (generalized) least squares methods to each individual which translates in the population framework to:

$$\theta_i^{\text{GLS}} = \arg \min_{\theta_i \in \Theta} (y_i - h_i(\theta_i))^T (\tau^{-1} \Omega_i)^{-1} (y_i - h_i(\theta_i)). \quad (4.1)$$

A standard approach to non-linear problems consists in linearizing the function h_{ij} for $j \in \llbracket 1, n_i \rrbracket$ and apply methods available in the linear case. [Beal and Sheiner \[1982\]](#) proposed to use a first-order Taylor development allowing to express the conditional distribution $p(y_i | \theta_i, \tau)$ as a multivariate normal distribution with mean and covariance matrix being linear functions of θ_i , and since the population distribution $p(\theta_i | \eta, \Sigma)$ is also normal, the likelihood of the model become analytically tractable. However, this approximation proved to yield unsatisfying and biased results [\[Vonesh, 1992\]](#), [\[Davidian and Giltinan, 2003\]](#). An improvement was suggested by [Lindstrom and Bates \[1990\]](#) where the Taylor development was performed around the best unbiased linear predictor in the case of the linear model. Although this method brought better estimates [\[Vonesh, 1992\]](#), it could still lead to mediocre results when the normal approximation hypothesis was aberrant.

In the face of the difficulties encountered with linear approximations, so-called exact methods were developed. Their main advantage is that they rely on numerical approximations of exact expressions and not on analytical expressions of the model. This was the case of Gauss quadrature approximations of the likelihood or importance sampling based methods [Pinheiro and Bates, 1995], even though these remained computationally costly and time-consuming. The majority of modern frequentist estimation methods for population models are now based on the Expectation–Maximization algorithm [Dempster et al., 1977], [Foulley, 2002]. It is particularly adapted to the subject of genotypic differentiation, as it has notably been used for the study of variations of residual and genetic variances in animal breeding [Foulley and Quaas, 1995]. It is an iterative algorithm aiming at maximizing the likelihood of a model with missing data, where the latter here represents the individual parameters. Denoting the whole population parameters by $\Phi = (\eta, \Sigma, \tau)$, each iteration is divided into two main steps:

- the first one deals with the calculation of the conditional expectation of the log-likelihood of the complete data under the law of missing data given the observations at the current step:

$$Q(\Phi, \Phi^t) = \mathbb{E}(\log p(y, (\theta_i)_{i \in [1, N]} | \Phi) | (y_i)_{i \in [1, N]}, \Phi^t), \quad (4.2)$$

- the second one aims at maximizing this conditional expectation to update the value of the parameters at iteration $t + 1$:

$$\Phi^{t+1} = \arg \max_{\theta} Q(\Phi, \Phi^t). \quad (4.3)$$

The EM algorithm notably ensures that the value of the log-likelihood is increased at each iteration. Since the E step requires the calculation of a conditional expectation that is usually intractable, it must therefore be approximated. This can be done via either an MCMC algorithm (MCEM) [McCulloch, 1994] or via a stochastic approximation method (SAEM) [Delyon et al., 1999], [Kuhn and Lavielle, 2005] where samples from previous EM iterations are reused with a weight depending on the distance between the iteration of the reused sample and the current one. These two approaches were compared by Baey et al. [2013] who found that they both yield satisfactory and comparable results, even though the SAEM algorithm was more efficient from a computational point of view. The SAEM algorithm was notably used in the context of restricted maximum likelihood method for the estimation of variance parameters in non-linear mixed effects models [Meza et al., 2007].

4.2 Bayesian approach

It is interesting to note that, by introducing the prior in the Q function of the EM algorithm (i.e. $p(y, (\theta_i)_{i \in [1, N]} | \Phi)$ is replaced by $p(y, (\theta_i)_{i \in [1, N]} | \Phi)p(\Phi)$), we straightforwardly get an algorithm to obtain the maximum a posteriori estimate. However, this approach does not fully qualify as Bayesian since it does not allow to derive the full posterior distribution.

The first attempt to estimate parameters within the population framework and a Bayesian paradigm can probably be traced back to [Gelfand and Smith \[1990\]](#). The case of linear models was first addressed for two main reasons: first such models were still en vogue in many scientific fields, and second as will be detailed later it is possible to derive analytical expressions for the full conditional distributions of all the random variables of interest, making the implementation of an MCMC algorithm straightforward.

In the Bayesian approach, a third stage is added to the hierarchical model where prior distributions are given to the population parameters. The hierarchical model introduced in Chapter 1 now reads:

$$\begin{aligned}
 \text{First stage:} \quad & y_i \sim \mathcal{N}(h_i(\theta_i), \tau^{-1}\Omega_i) \\
 \text{Second stage:} \quad & \theta_i \sim \mathcal{N}(\eta, \Sigma) \\
 \text{Third stage:} \quad & \left\{ \begin{array}{l} \eta \sim p(\eta) \\ \Sigma \sim p(\Sigma) \\ \tau \sim p(\tau) \end{array} \right.
 \end{aligned} \tag{4.4}$$

and adequate prior distributions for the population mean $p(\eta)$, covariance $p(\Sigma)$ and precision $p(\tau)$ remain to be chosen. The objective is to estimate both the population parameters $\Phi = (\eta, \Sigma, \tau)$ and the individual parameters $(\theta_i)_{1:N}$. Within such models, the standard Bayesian estimation of the posterior density for the population parameters $p(\Phi|y_{1:N})$ cannot be performed straightforwardly, precisely because of the hierarchical formulation. Furthermore, using Bayes' rule,

$$p(\Phi|y_{1:N}) = \frac{p(y_{1:N}|\Phi) p(\Phi)}{p(y_{1:N})} \tag{4.5}$$

and from Equation 4.4:

$$p(y_{1:N}|\Phi) = \prod_{i=1}^N p(y_i|\Phi) = \prod_{i=1}^N \int_{\theta_i} p(y_i|\theta_i) p(\theta_i|\eta, \Sigma) d\theta_i \tag{4.6}$$

one is again confronted to the calculation of complex and intractable integrals. The Gibbs sampler introduced in Section 3.5.1 overcome this difficulty and is particularly convenient in the context of hierarchical models: indeed, their formulation allows for simple analytical expressions of most of the required full conditional distributions from which to sample.

4.2.1 Full conditional distributions

The power and simplicity of the Gibbs sampler is first demonstrated on a hierarchical normal-linear model introduced by [Lindley and Smith \[1972\]](#). The output of the model is given as a linear function of the parameters and conditionally independent homoscedastic errors are assumed. The first stage of the model can therefore be written as:

$$y_i \sim \mathcal{N}(X_i\theta_i, \tau^{-1}I_{n_i}) \tag{4.7}$$

where $X_i \in \mathcal{M}_d(\mathbb{R})$ and where we recall that the problem dimension d here is that of the number of estimated parameters d_θ . With the objective to estimate the posterior density of all the parameters $p(\theta_{1:N}, \eta, \Sigma, \tau | y_{1:N})$ via the Gibbs sampler, it is required to calculate all the full conditional distributions:

$$\begin{cases} p(\theta_i | \theta_{j \neq i}, \eta, \Sigma, \tau, y_{1:N}), \text{ for } i \in \llbracket 1, N \rrbracket, \\ p(\eta | \theta_{1:N}, \Sigma, \tau, y_{1:N}), \\ p(\Sigma | \theta_{1:N}, \eta, \tau, y_{1:N}), \\ p(\tau | \theta_{1:N}, \eta, \Sigma, y_{1:N}). \end{cases} \quad (4.8)$$

For the sake of simplicity, the full conditional distribution of random variable z conditional to all other variables will be abbreviated as $p(z | \dots)$. The joint distribution of the model is:

$$p(y_{1:N}, \theta_{1:N}, \eta, \Sigma, \tau) = \prod_{i=1}^N p(y_i | X_i \theta_i, \tau^{-1} I_{n_i}) \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\eta) p(\Sigma) p(\tau). \quad (4.9)$$

To obtain the full conditional distribution of a given random variable z , it suffices to use Bayes' formula $p(z | \dots) = p(\dots)^{-1} p(z, \dots)$, and collecting all terms involving z in $p(z, \dots)$ will therefore lead to the desired distribution. This yields:

$$\begin{cases} p(\theta_i | \dots) \propto_{\theta_i} p(y_i | X_i \theta_i, \tau^{-1} I_{n_i}) p(\theta_i | \eta, \Sigma) & = f_{\mathcal{N}}(y_i, X_i \theta_i, \tau^{-1} I_{n_i}) f_{\mathcal{N}}(\theta_i, \eta, \Sigma) \\ p(\eta | \dots) \propto_{\eta} \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\eta) & = \prod_{i=1}^N f_{\mathcal{N}}(\theta_i, \eta, \Sigma) p(\eta) \\ p(\Sigma | \dots) \propto_{\Sigma} \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\Sigma) & = \prod_{i=1}^N f_{\mathcal{N}}(\theta_i, \eta, \Sigma) p(\Sigma) \\ p(\tau | \dots) \propto_{\tau} \prod_{i=1}^N p(y_i | h_i(\theta_i), \tau^{-1} \Omega_i) p(\tau) & = \prod_{i=1}^N f_{\mathcal{N}}(y_i, X_i \theta_i, \tau^{-1} I_{n_i}) p(\tau) \end{cases} \quad (4.10)$$

and it can be shown that the full conditional distribution for the individual parameters can be put under the form:

$$\begin{cases} \theta_i \sim \mathcal{N}(\eta_i^*, \Sigma_i^*), \\ \eta_i^* = (\tau X_i^T X_i + \Sigma^{-1})^{-1} (\tau X_i^T y_i + \Sigma^{-1} \eta), \\ \Sigma_i^* = (\tau X_i^T X_i + \Sigma^{-1})^{-1}. \end{cases} \quad (4.11)$$

4.2.2 Choice of the prior distributions

So far, the choice of the prior distributions has not been discussed but Equation 4.10 makes it appear of crucial importance to obtain an analytical expression of the full conditional distributions. Adequate choices of the priors arise from the work of [Raiffa and Schlaifer \[1961\]](#) on what they coined conjugate priors. Going back to Bayes' theorem with random variables z and observations y :

$$p(z|y) = \frac{p(y|z) p(z)}{p(y)} = \frac{p(y|z) p(z)}{\int p(y|z) p(z) dz}, \quad (4.12)$$

it is clear that different choices of the prior distribution $p(z)$ will lead to both different analytical expressions for $p(y|z) p(z)$ and more or less difficult integral $p(y)$. In many situations and given the form of the likelihood, which is typically given by the model considered, an adequate choice of the prior distribution

$p(z)$ will yield a posterior distribution $p(z|y)$ that is in the same family of probability distribution as the prior. This framework can be applied to the full conditional distributions of the population parameters. For the mean η , the likelihood takes the form of a multivariate normal distribution with known covariance Σ ; in this case, the conjugate prior for η is also a multivariate normal distribution. For the covariance matrix Σ , the likelihood is a multivariate normal distribution, but this time it is the mean η that is known instead of the covariance; the conjugate prior for Σ is then an inverse Wishart distribution, or equivalently, the prior for the inverse of the covariance matrix Σ^{-1} is a Wishart distribution. The use of Σ^{-1} will be preferred in the following for easier and more conventional notations. Finally, as far as the precision τ is concerned, the likelihood takes the form of a univariate normal distribution with known mean; the conjugate prior for τ is therefore a Gamma distribution. This leads us to the final formulation of the hierarchical model:

$$\begin{aligned}
 \text{First stage:} \quad & y_i \sim \mathcal{N}(h_i(\theta_i), \tau^{-1}\Omega_i) \\
 \text{Second stage:} \quad & \theta_i \sim \mathcal{N}(\eta, \Sigma) \\
 \text{Third stage:} \quad & \left\{ \begin{array}{l} \eta \sim \mathcal{N}(\lambda, \Lambda) \\ \Sigma^{-1} \sim \mathcal{W}(q, \Psi) \\ \tau \sim \mathcal{G}(\alpha, \beta) \end{array} \right.
 \end{aligned} \tag{4.13}$$

where \mathcal{N} , \mathcal{W} and \mathcal{G} denote the normal, Wishart and Gamma distributions respectively. Some precisions shall be given on these prior distributions. The multivariate normal distribution is a well known probability distribution, parameterized by its mean λ and its covariance matrix Λ . Its probability density function is nothing but:

$$f_{\mathcal{N}}(x, \lambda, \Lambda) = |2\pi\Lambda|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x - \lambda)^T \Lambda^{-1}(x - \lambda)\right). \tag{4.14}$$

The Gamma distribution admits two main parameterizations, with shape and scale (k, θ) or shape and rate (or inverse scale) $(\alpha, \beta) = (k, \theta^{-1})$. The former seems slightly more standard in the statistics community (it is notably used in the Julia package `Distributions`¹), although in the context of hierarchical models and Bayesian update the latter is always preferably used, probably because it yields a clearer update of the parameters. Its probability density function reads:

$$f_{\mathcal{G}}(x, \alpha, \beta) = x^{\alpha-1} \beta^\alpha \Gamma(\alpha)^{-1} \exp(-\beta x) \tag{4.15}$$

and its mean and variance are given by:

$$\left\{ \begin{array}{l} \mathbb{E}(X) = \frac{\alpha}{\beta}, \\ \mathbb{V}(X) = \frac{\alpha}{\beta^2}. \end{array} \right. \tag{4.16}$$

The Wishart distribution can finally be seen as a multivariate generalization of the Gamma distribution. It arose from the study of the scatter matrix of normal variables [Wishart, 1928]. More precisely, if $X_i \sim \mathcal{N}(0, \Psi)$ where $\Psi \in \mathcal{M}_d(\mathbb{R})$, let $X = [X_1, \dots, X_q]$ be the matrix made of all the samples X_i and $S = X^T X$ be the scatter matrix, then $S \sim \mathcal{W}_d(q, \Psi)$, where q is called the number of degrees of freedom.

¹<https://github.com/JuliaStats/Distributions.jl>

Its probability density function is defined on $\mathcal{S}_d^+(\mathbb{R})$ as:

$$f_{\mathcal{W}_d}(X, q, \Psi) = 2^{-\frac{qd}{2}} |\Psi|^{-\frac{q}{2}} \Gamma_d\left(\frac{q}{2}\right)^{-1} |X|^{-\frac{q-d-1}{2}} e^{-\frac{1}{2}\text{tr}(\Psi^{-1}X)} \quad (4.17)$$

and the mean and variance for each of its components are:

$$\begin{cases} \mathbb{E}(X_{ij}) &= q\Psi_{ij}, \\ \mathbb{V}(X_{ij}) &= q(\Psi_{ij}^2 + \Psi_{ii}\Psi_{jj}). \end{cases} \quad (4.18)$$

In the following, we often simplify $\mathcal{W}_d(q, \Psi)$ as $\mathcal{W}(q, \Psi)$. The inverse Wishart distribution is such that if $X \sim \mathcal{W}(q, \Psi)$, then $X^{-1} \sim \mathcal{IW}(q, \Psi^{-1})$, so that for the population covariance $\Sigma \sim \mathcal{IW}(q, \Psi^{-1})$.

The parameters of the prior distributions are called hyperparameters. In this particular case, these prior distributions therefore do not arise because they represent some prior knowledge on the population parameters; rather, they are used for convenience and to obtain analytical expressions of the full conditional distributions for a straightforward sampling in a Gibbs framework. It is, however, possible and desirable to tune the parameters of these distributions to integrate reasonable prior knowledge: the means and variances of such distributions should be used to choose the parameters of these distributions that provide distributions centered on the regions of interest as will be discussed in Chapter 8.

Some care must be taken for the choice of the Gamma and Wishart hyperparameters. The multiplicative observation noise in plant growth models is expected to be such that the standard deviation is of the order of $\sigma = 0.1$, which is equivalent to a precision of $\tau = 100$. The hyperparameters of the Gamma distribution can be expressed as functions of the mean m and standard deviation s , from Equation 4.15:

$$\begin{cases} \alpha &= \frac{m^2}{s^2}, \\ \beta &= \frac{m}{s^2}. \end{cases} \quad (4.19)$$

On Figure 4.1, several Gamma distributions are plotted with mean 100 and different standard deviations. Informative Gamma priors such as the one with $s = 10$ resemble normal distributions, but non-informative Gamma priors such as the one with $s = 200$ does not look like a priori information one would like to

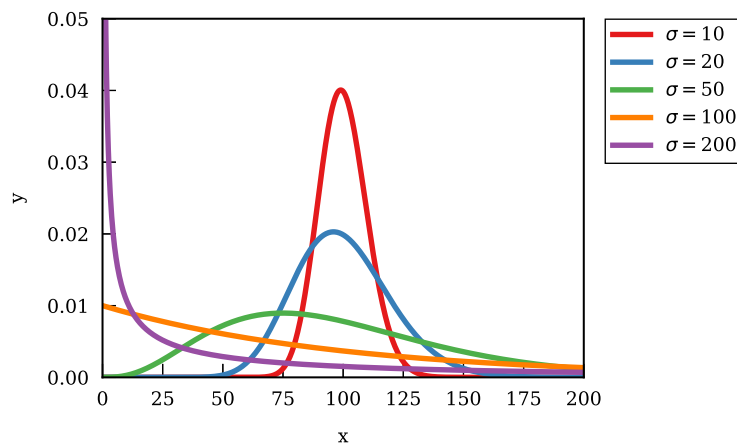


Figure 4.1: Probability density functions of several Gamma distributions with mean 100 and different standard deviations ranging from $s = 10$ to $s = 200$.

incorporate into the precision distribution.

Taking the example of a 2-dimensional Wishart distribution $\mathcal{W}(3, \Psi)$ without correlation in Ψ , it is shown on Figure 4.2 how the individual components of the sampled matrices behave. Assuming a diagonal inverse covariance matrix $\Sigma^{-1} \sim \mathcal{W}(q, \Psi)$, the parameters of the Wishart distribution can be expressed in terms of expected means $(m_i)_{1:p}$ and standard deviations $(s_i)_{1:p}$ of the diagonal components of the inverse covariance matrix:

$$\begin{cases} q &= 2 \frac{m_i^2}{s_i^2}, \\ \Psi_{ii} &= \frac{s_i^2}{2m_i}. \end{cases} \quad (4.20)$$

Maybe unsurprisingly since the Gamma distribution is a particular case of a Wishart distribution, the same conclusions apply: strongly non-informative priors with very high variances lead to undesirable pdf shapes.

The full conditional distributions of the population parameters can therefore be calculated (see Appendix A). We find that for the population mean:

$$\begin{cases} \eta &\sim \mathcal{N}(\cdot | \lambda^*, \Lambda^*), \\ \lambda^* &= (N\Sigma^{-1} + \Lambda^{-1})^{-1}(N\Sigma^{-1}\theta^* + \Lambda^{-1}\lambda), \\ \Lambda^* &= (N\Sigma^{-1} + \Lambda^{-1})^{-1}, \end{cases} \quad (4.21)$$

for the population covariance matrix:

$$\begin{cases} \Sigma^{-1} &\sim \mathcal{W}(\cdot | q^*, \Psi^*), \\ q^* &= q + N, \\ \Psi^* &= (\Psi^{-1} + \sum_{i=1}^N (\theta_i - \eta)(\theta_i - \eta)^T)^{-1}, \end{cases} \quad (4.22)$$

and finally for the population precision:

$$\begin{cases} \tau &\sim \mathcal{G}(\cdot | \alpha^*, \beta^*), \\ \alpha^* &= \alpha + n_{\text{tot}}/2, \\ \beta^* &= \beta + \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i)). \end{cases} \quad (4.23)$$

Equation 4.11 and Equations (4.21)–(4.23) therefore provide all the necessary material for the implementation of a Gibbs sampler.

In order to robustify the model to outlier individuals, Wakefield et al. [1994] proposed to replace the population normal assumption with a Student distribution. They also suggested that it be possible to replace the first stage of the model with a Student law in order to better deal with data outliers. Such an extension with fixed degrees of freedom for the Student's distribution ν and covariance matrix $\nu/(\nu - 2)\Sigma$ is straightforward since such a distribution can be expressed as a scale mixture of normals [Andrews and Mallows, 1974], [Racine-Poon, 1992].

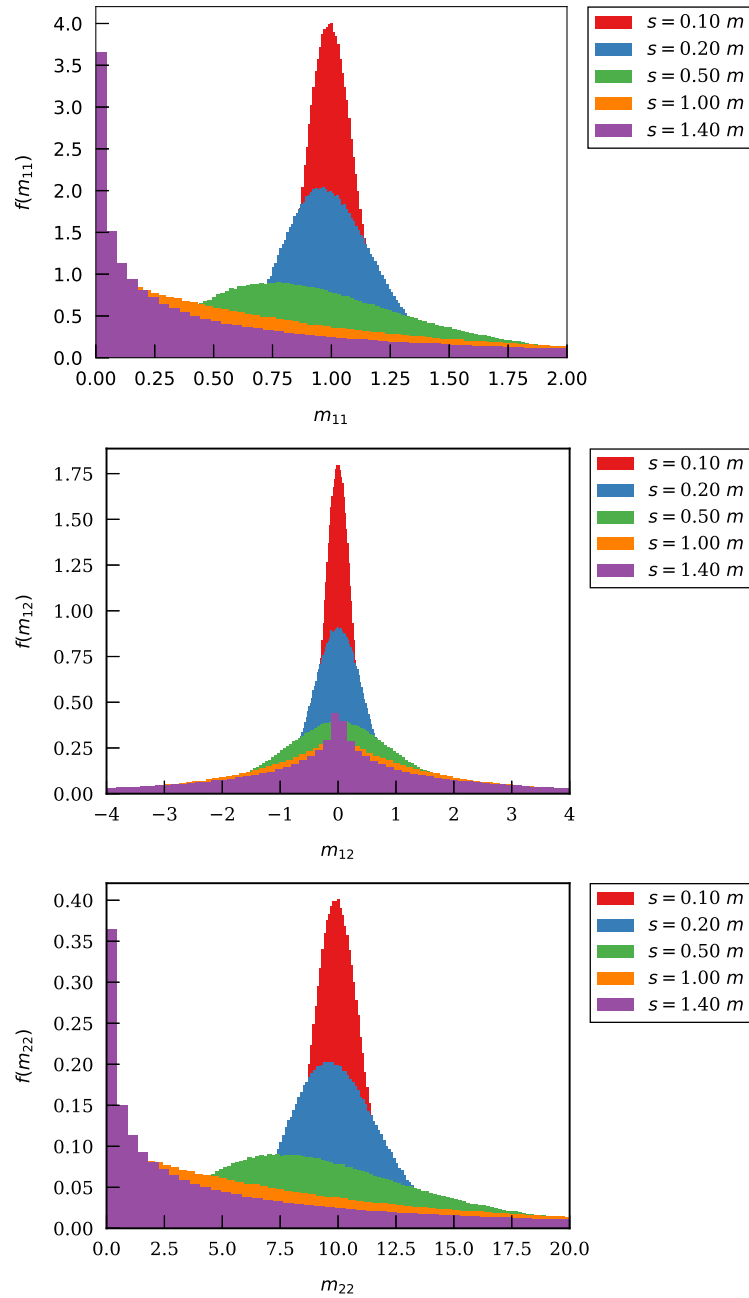


Figure 4.2: Histograms corresponding to the pdfs of the first diagonal component m_{11} , the off-diagonal component m_{12} and the second diagonal component m_{22} obtained from $N = 100,000$ samples of the Wishart distribution $\mathcal{W}(3, \text{diag}(1, 10))$.

4.2.3 Non-linear models

When dealing with non-linear hierarchical models, the full conditional distribution of the individual parameters cannot be analytically expressed anymore. Indeed, going back to the general formulation with multiplicative normal noise:

$$\begin{aligned}
 p(\theta_i | \dots) &\propto_{\theta_i} p(y_i | h_i(\theta_i), \tau^{-1} \text{diag}\{h_i(\theta_i)^2\}) p(\theta_i | \eta, \Sigma) \\
 &= f_{\mathcal{N}}(y_i, h_i(\theta_i), \tau^{-1} \text{diag}\{h_i(\theta_i)^2\}) f_{\mathcal{N}}(\theta_i, \eta, \Sigma).
 \end{aligned} \tag{4.24}$$

Algorithm 10 Gibbs sampler for linear population models

Choose values for the hyperparameters (λ, Λ) , (q, Ψ) and (α, β) .

For $t = 1 : M$

For $i = 1 : N$

 Update η_i^* and Σ_i^* according to Equation 4.11

 Sample individual parameters $\theta_i^t \sim \mathcal{N}(\eta_i^*, \Sigma_i^*)$

End

 Update λ^* and Λ^* according to Equation 4.21

 Sample population mean $\eta^t \sim \mathcal{N}(\lambda^*, \Lambda^*)$

 Update q^* and Ψ^* according to Equation 4.22

 Sample population covariance $\Sigma^t \sim \mathcal{W}(q^*, \Psi^*)$

 Update α^* and β^* according to Equation 4.23

 Sample population precision $\tau^t \sim \mathcal{G}(\alpha^*, \beta^*)$

End

With complex models, such as the GreenLab plant growth model, finding the analytical expression of $p(\theta_i | \dots)$ as a known distribution from which one could sample is not feasible. Several MCMC strategies have therefore been designed to overcome this issue and sample the individual-specific parameters θ_i in the case of non-linear models. Wakefield et al. [1994] sample single components of the θ_i vectors with the method of extended ratio-of-uniforms [Wakefield et al., 1991]. This method not only involved too many maximizations per random variable generation, but it was also showed [Bennett et al., 1996] that the convergence can be improved if all the components of the parameters vector θ_i are updated simultaneously instead of component-wise. In the case of a generalized linear model, Zeger and Karim [1991] tried rejection sampling from an approximate rejection envelope but their approach was deemed unreliable because of the potential negative effects of a non-dominating envelope. Gilks et al. [1995] used a univariate adaptive rejection sampling scheme in the context of non-linear hierarchical models, the rejection sampling method originating from Ripley [1987]. For an additive observation model, the likelihood for individual i would read:

$$\ell_i(\theta_i, \tau) = p(y_i | \theta_i, \tau) = \prod_{j=1}^{n_i} \left(\frac{\tau}{2\pi} \right)^{\frac{1}{2}} \exp \left(-\frac{\tau}{2} (y_{ij} - h_{ij}(\theta_i))^2 \right) \quad (4.25)$$

and the corresponding log-likelihood, ignoring the terms not depending on θ_i :

$$\log \ell_i(\theta_i, \tau) = -\frac{\tau}{2} \sum_{j=1}^{n_i} (y_{ij} - h_{ij}(\theta_i))^2. \quad (4.26)$$

It is assumed that this function can be maximized when τ is fixed, the maximum likelihood estimate (MLE) is therefore:

$$\hat{\theta}_i = \arg \max \log \ell_i(\theta_i, \tau). \quad (4.27)$$

Bennett et al. [1996] proposed that the rejection envelope for the sampling of the individual parameters be $p(\theta_i | \eta, \Sigma)$, since:

$$p(\theta_i | \dots) \propto \ell_i(\theta_i, \tau) p(\theta_i | \eta, \Sigma) \quad (4.28)$$

the rejection sampling scheme takes the form given by Algorithm 11.

Algorithm 11 Accept-reject sampling scheme for individual parameters

```

while A == False
  Sample  $\theta_i^* \sim \mathcal{N}(\theta_i|\eta, \Sigma)$ 
  Sample  $u \sim \mathcal{U}(0, 1)$ 
  If  $u \leq \frac{\ell_i(\theta_i, \tau)}{\ell_i(\hat{\theta}_i, \tau)}$ 
    Set  $\theta_i^t = \theta_i^*$  and A = True
  End
End while

```

The main advantage of this procedure is, as can be seen from Equation 4.26, that $\hat{\theta}_i$ does not depend on τ and needs to be evaluated only once prior to the MCMC run. Generation of the individual samples from a normal distribution is also straightforward: overall this method is simple and costless.

However, when adopting a multiplicative observation model, as was chosen for the application of population models in this work, the likelihood becomes:

$$\ell_i(\theta_i, \tau) = p(y_i|\theta_i, \tau) = \prod_{j=1}^{n_i} \left(\frac{\tau}{2\pi h_{ij}(\theta_i)^2} \right)^{\frac{1}{2}} \exp \left(-\frac{\tau}{2} \left(\frac{y_{ij} - h_{ij}(\theta_i)}{h_{ij}(\theta_i)} \right)^2 \right) \quad (4.29)$$

and the corresponding log-likelihood, again ignoring the terms not depending on θ_i , reads:

$$\log \ell_i(\theta_i, \tau) = - \sum_{j=1}^{n_i} \log(h_{ij}(\theta_i)) - \frac{\tau}{2} \sum_{j=1}^{n_i} \left(\frac{y_{ij} - h_{ij}(\theta_i)}{h_{ij}(\theta_i)} \right)^2 \quad (4.30)$$

and the maximization of the log-likelihood becomes not only more complicated but now depends on τ , which prevents from performing it only once.

4.2.4 Metropolis–Hastings for sampling the individual parameters

Another family of MCMC methods for sampling the individual parameters involves using a Metropolis–Hastings step. Hybrid MCMC algorithms where some parameters are updated from full conditional distributions by Gibbs sampling and some others with a Metropolis–Hastings step are discussed in [Tierney, 1994]. A natural choice is to use a random walk procedure with a multivariate normal proposal distribution so that parameters are updated altogether. Some popular choices are mentioned in the following, a first one being:

$$\theta_i^* \sim \mathcal{N}(\theta_i^t, c I_{\hat{\theta}_i}) \quad (4.31)$$

where $c > 0$ is a constant and $I_{\hat{\theta}_i}$ is the information matrix evaluated in the MLE estimate for individual i :

$$I_{\hat{\theta}_i} = - \left[\frac{\partial^2 \log \ell_i(\theta_i, \tau)}{\partial \theta_i \partial \theta_i^T} \right]_{\theta_i = \hat{\theta}_i}^{-1}. \quad (4.32)$$

However, this implies that the covariance matrix stays constant all along the MCMC algorithm. Other possibilities include an independence sampler where this time the mean of the proposal distribution is fixed

to the MLE estimate:

$$\theta_i^* \sim \mathcal{N} \left(\hat{\theta}_i, -c \left[\frac{\partial^2 \log \ell_i(\theta_i, \tau)}{\partial \theta_i \partial \theta_i^T} \right]_{\theta_i = \hat{\theta}_i}^{-1} \right) \quad (4.33)$$

or, by approximating the likelihood as $\ell_i(\theta_i, \tau) \approx \mathcal{N}(\hat{\theta}_i, I_{\hat{\theta}_i})$ and sampling from the associated approximate conditional distribution:

$$\theta_i^* \sim \mathcal{N}(\hat{\theta}_i - I_{\hat{\theta}_i} (I_{\hat{\theta}_i} + \Sigma)^{-1} (\hat{\theta}_i - \eta), (I_{\hat{\theta}_i}^{-1} + \Sigma^{-1})^{-1}). \quad (4.34)$$

A comparison of the different strategies was made by [Bennett et al. \[1996\]](#). The Metropolis–Hastings were found to be much more computationally efficient than the rejection methods, with a single evaluation of $p(\theta_i | \dots)$ per MCMC iteration compared to methods where sampling must be done until acceptance or where maximizations are involved. The Metropolis–Hastings methods based on the proposal distributions presented in Equations 4.33 and 4.34 were found to perform moderately well compared to that based on Equation 4.32, although the latter suffered from drawbacks such as tuning the algorithm so as to find the optimal value for the constant c .

4.2.5 Individual adaptive scheme

Because plant growth models are complex and as a lot of data per individual is involved in the application of Chapter 8, a computationally efficient approach was chosen. To overcome the state space exploration issues raised by [Bennett et al. \[1996\]](#), an adaptive scheme as presented in Section 3.5.1 was used to ensure an automatic tuning of the covariance matrix in the proposal distribution, individual by individual. The parameters of the adaptive scheme relative to individual i at iteration t will be denoted γ_i^t , μ_i^t , Σ_i^t and λ_i^t . In particular, there should be no confusion between the individual covariance matrix for the Metropolis–Hastings step of individual i , Σ_i^t , and the population covariance matrix Σ^t . The full conditional distribution of the individual parameters provide the acceptance probability for the Metropolis–Hastings step:

$$\alpha = 1 \wedge \frac{p(y_i | \theta_i^*, \tau) p(\theta_i^* | \eta, \Sigma)}{p(y_i | \theta_i^t, \tau) p(\theta_i^t | \eta, \Sigma)}. \quad (4.35)$$

Initial values for the individual parameters were found by running a GLS algorithm (see Section 3.4) for each individual, therefore providing a reasonable first value for the MCMC chain instead of randomly sampling from the population distribution $\mathcal{N}(\eta, \Sigma)$. The final algorithm used in the context of non-linear hierarchical models for the GreenLab model is presented in Algorithm 12.

Algorithm 12 Adaptive hybrid Metropolis–Hastings–Gibbs algorithm for non-linear hierarchical models

Choose values for the population mean hyperparameters λ, Λ

Sample $\eta^0 \sim \mathcal{N}(\lambda, \Lambda)$

Choose values for the population covariance hyperparameters q, Ψ

Sample $\Sigma^{-1,0} \sim \mathcal{W}(q, \Psi)$

Choose values for the population precision hyperparameters α, β

Sample $\tau^0 \sim \mathcal{G}(\alpha, \beta)$

For $i = 1 : N$

 Compute $\theta_i^0 = \theta_i^{\text{GLS}}$

 Initialize individual adaptive scheme parameters:

$$\begin{cases} \gamma_i^0 &= 1/2 \\ \mu_i^0 &= \theta_i^0 \\ \Sigma_i^0 &= \Sigma^0 \\ \lambda_i^0 &= 2.38^2/n_\theta \end{cases} .$$

End

For $t = 0 : M - 1$

For $i = 1 : N$

 Sample $\theta_i^* \sim \mathcal{N}(\theta_i^t, \lambda_i^t \Sigma_i^t)$

 Compute $h_i(\theta_i^*)$ via model simulation

 Compute the acceptance probability:

$$\alpha_i = 1 \wedge \frac{p(y_i | h_i(\theta_i^*), \tau^t) p(\theta_i^* | \eta^t, \Sigma^t)}{p(y_i | h_i(\theta_i^t), \tau^t) p(\theta_i^t | \eta^t, \Sigma^t)}$$

 Set $\theta_i^{t+1} = \theta_i^*$ with probability α_i , set $\theta_i^{t+1} = \theta_i^t$ otherwise

 Update parameters of the individual adaptive scheme:

$$\begin{cases} \gamma_i^{t+1} &= (t+1)^{-1} \\ \mu_i^{t+1} &= \mu_i^t + \gamma_i^{t+1}(\theta_i^{t+1} - \mu_i^t) \\ \Sigma_i^{t+1} &= \Sigma_i^t + \gamma_i^{t+1} [(\theta_i^{t+1} - \mu_i^{t+1})(\theta_i^{t+1} - \mu_i^{t+1})^T - \Sigma_i^t] \\ \lambda_i^{t+1} &= \lambda_i^t \exp(\gamma_i^{t+1}[\alpha_i - \alpha^*]) \end{cases} .$$

End

 Update λ^* and Λ^* according to Equation 4.21

 Sample population mean $\eta^{t+1} \sim \mathcal{N}(\lambda^*, \Lambda^*)$

 Update q^* and Ψ^* according to Equation 4.22

 Sample population inverse covariance $\Sigma^{-1, t+1} \sim \mathcal{W}(q^*, \Psi^*)$

 Update α^* and β^* according to Equation 4.23

 Sample population precision $\tau^{t+1} \sim \mathcal{G}(\alpha^*, \beta^*)$

End

Chapter 5

Adopting Julia for statistical inference

THE MATHEMATICAL FRAMEWORK OF GENERAL STATE-SPACE MODELS introduced in Chapter 1 is particularly suited to the study of plant growth models. Incidentally, it was shown in Chapter 2 how two main plant growth models for the study of *Beta vulgaris* and *Arabidopsis thaliana* could be formulated as such, with each its peculiarities: the first one has process noise originating from environmental uncertainty, the second one has vector observations and will be used within a population context. The calibration of these models require specific algorithms, ranging from sensitivity analysis to estimation of parameters and possibly hidden states. In both cases, the latter are envisioned within a Bayesian paradigm and are based on sequential Monte Carlo or Markov chain Monte Carlo methods.

Therefore, many issues need to be resolved practically and to answer the statistical issues that we face, we need to be able to:

- efficiently simulate plant growth models formulated as complex state space models;
- adopt a population approach so as to evidence the genotypic differentiation existing within a plant population;
- use complex sensitivity analysis and estimation methods in the case of both an individual and a population approach;
- do all the above mentioned tasks efficiently: since we are dealing with large amounts of data and that the estimation algorithms often require either a large number of particles or iterations, the computing time and the memory used quickly skyrocket, rendering such procedures unusable in practice.

Several libraries or software for statistical inference in such state space models have been proposed. Some of the most renowned examples include Stan [Carpenter et al., 2017], Bugs [Lunn et al., 2000], [Lunn et al., 2009], [Lunn et al., 2012], Jags [Plummer, 2003] and LibBi [Murray, 2015] just to name a few. However, all these tools do not gather all the functionalities and requirements that are usually needed in the model design process, in terms of both model implementation ease and statistical methods necessary for a proper analysis and evaluation methodology referred to as good modelling practice in [Van Waveren et al., 1999].

Previous works in order to achieve these targets already existed in the Biomathematics team. Two versions of a C++ platform gathering the theoretical framework and a majority of the required estimation algorithms have been developed under the name of PyGMAlion [Cournède et al., 2013], whose architecture was designed and implemented by Bayol [2016]. The most recent version also integrates a domain specific language (DSL) in order to ease model designing. Nevertheless, although these platforms have proved to be invaluable tools for the design of models and their calibration, it was decided for this thesis to create a new platform based on the Julia language. The main motivation behind such a choice arises from different factors. First, in the research area and maybe particularly during a thesis, many adjustments need to be brought to the programs used in order to fix bugs, explore new methods or tune parameters. Modifying code in C++ has proved to be more complicated and especially much more time-consuming than in Julia. Several other convenient aspects of the Julia language that will be presented afterwards, such as metaprogramming or code generation, not available or much harder to obtain in C++ played a key role in this decision. As far as genericness is concerned, the automatic computation of the transition and observation probability density functions, a crucial feature for the different estimation procedures, was not doable. Last but not least, designing such a platform was not for the sole purpose of this thesis. It was kept in mind that this could serve as a main tool for the whole research team. Obviously and because of the simplicity, the conciseness and the efficiency of the Julia language, it is much easier to dive into code written in Julia rather than C++.

Complex, nonlinear models with external variables, specific process and observation noises should also be easy to design. The implementation of state of the art algorithms and their probable modifications in order to test many different strategies for convergence or space exploration should be easy for any model. This motivated the need for an automatic calculation of the transition and observation distribution of said models. Dealing with high-dimensional problems and complex models, all computations should be very efficient. These reasons led to the development of ADJUSTIN', which stands for Adopting Julia for Statistical Inference.

The organization of the rest of this chapter is as follows: we will first explain the main reasons behind the choice of the Julia language for a highly-efficient statistical computing platform. We then move on to specify the mathematical and programming framework underlying models in ADJUSTIN' and how it allows to design models easily (Section 5.2), then see how both the organization and the syntax of the platform lead to straightforward simulations (Section 5.3). An interesting aspect of Julia, code generation, will be discussed for the structures storing observations (Section 5.4). Its metaprogramming capabilities will then be highlighted for the automatic computation of the transition and observation pdfs (Section 5.5). Some of the algorithms introduced in Chapter 3 and 4 and developed in the platform, ranging from sensitivity analysis, Markov chain Monte Carlo and sequential Monte Carlo methods, will be discussed in the context of the platform (Section 5.6).

5.1 Why Julia?

The choice of Julia¹ [Bezanson et al., 2012], [Bezanson et al., 2014] as the main language for the ADJUSTIN⁷ platform was motivated by several factors. As previously mentioned, being a research platform used by academic researchers as well as engineers, it needs to be easy to use. Julia is a general purpose language, free, open-source, platform independent and user-friendly in many respects. First, it has a very clean and concise syntax as well as a high-expressiveness, allowing to design algorithms with many less characters than with other languages such as C++ or even Python. In particular, it was not considered necessary to design a DSL for the design of models as Julia is simple enough for the task (Section 5.2). It has a well-designed library for scientific computing, the Distributions package² makes sampling from classical distributions straightforward, and it also incorporates a lot of statistical routines in the standard library. It comes with a very handy read-eval-print loop (REPL) for interactive use which, as for instance in Python, allows to test code rapidly. Furthermore, Julia makes use of just-in-time (JIT) compilation, which means that there is no explicit need to compile a project, which would be compulsory with highly efficient languages such as C or Fortran.

A very important aspect of Julia concerns its metaprogramming capabilities that are extensively used within the ADJUSTIN⁷ platform for generic purposes: in particular, to make algorithms work regardless of the model considered (Section 5.6), to compute probability density functions automatically (Section 5.5), to design models whose structure can depend on certain hyperparameters (Section 5.2.4), or to automatically generate operations on certain structures (Section 5.4). It displays a dynamic, nominative and parametric type system and has excellent support for functional programming. It is also worth mentioning its tool supports, including documentation, testing, and Atom, a good integrated development environment (IDE). High-performance computing is an absolute necessity when dealing with computationally expensive algorithms for either sensitivity analysis or parameter and state estimation in complex statistical models with large state spaces. Julia responds well to such needs: it is fast and efficient, its LLVM-based compiler allows Julia to get performances close to those obtained with C: the computing times for 7 different popular algorithms were tested³ for Julia and 11 other languages, highlighting that Julia performs significantly better than other languages commonly used in the research community such as Python, R or Matlab. For this particular benchmark, it is actually slightly slower than C and compares to Fortran. This trend was confirmed by the implementation of an MCMC algorithm in plain C++, Julia and R [Viaud et al., 2015]. It is possible to call a variety of languages, which makes it very practical in a variety of situations. Bash commands can be executed within a Julia file, the PyCall package makes it possible to call Python functions, and the function `ccall` allows to call C and Fortran code very easily. Lots of popular libraries from other languages have been interfaced to be used in Julia, this is the case of Matplotlib (with the PyPlot package⁴), which allows excellent data visualization, or MPI and CUDA for parallel computing. The MPI package⁵ notably made it possible to parallelize several algorithms in the ADJUSTIN⁷ platform for high-performance computing as will be shown

¹<https://julialang.org/>

²<https://github.com/JuliaStats/Distributions.jl>

³<http://julialang.org/benchmarks/>

⁴<https://github.com/JuliaPy/PyPlot.jl>

⁵<https://github.com/JuliaParallel/MPI.jl>

in Section 5.6 for the convolution particle filter. Last but not least, the Julia community, though rather small in comparison with those of languages established for longer, is very active. For all these reasons, Julia seemed to be the best compromise between numerical efficiency, design, genericness and ease of use.

5.2 Modelling and simulation

The ADJUSTIN' platform adopts the mathematical framework presented in Chapter 1. The state variables can be continuous but the models are discrete in time. Note however that continuous in time models, such as ODEs or SDEs, can be cast into the framework by integration between the discrete time steps. In the ADJUSTIN' platform, models originating from ecology, epidemiology, biological regulatory networks, finance, and plant science coexist. Throughout this article, some basic knowledge (most notably base types, arrays, functions and syntax) of the Julia language is assumed. The official website of the language provides a great deal of excellent resources for learning the basics of Julia⁶. We follow the Julia convention according to which only types begin with an uppercase character and functions whose name ends with an exclamation mark modifies at least one of their arguments. A `Label` can be seen as an `AbstractString`, and the suffix `List` is added to type names to refer to arrays of said types, for instance:

$$\text{LabelList} \equiv \text{Array}\{\text{Label}\} \equiv \text{Array}\{\text{AbstractString}\}. \quad (5.1)$$

For the sake of conciseness, programming arrays will often be denoted with square brackets `[]`. Sometimes, `"..."` is used in code environment to indicate voluntarily omitted code.

5.2.1 Mathematical framework

Let $\mathbb{K} = (\cup_{k \geq 1} \mathbb{N}^k) \cup (\cup_{k \geq 1} \mathbb{R}^k)$. From a computing point of view, an element of \mathbb{K} can be an `Int`, a `Float64`⁷, an `Array{Int}` or an `Array{Float64}`. Arrays of integers or floats are useful in many models where, for instance, the size of some vector-valued variable is not known in advance and evolves as the model simulation advances. To account for this numerical flexibility, structures essentially are elements of \mathbb{K}^n for a given $n > 0$, even though they can also be regarded as elements of $\mathbb{R}^{n'}$ with $n' \geq n$, in a more standard approach. For instance, a structure containing two real variables and one 3-dimensional variable $(a, b, (x, y, z))$ can be seen as either an element of \mathbb{K}^3 from a computing point of view or \mathbb{R}^5 from a mathematical point of view.

We recall the main system of equations summarizing the evolution of a state space model in the ADJUSTIN' platform:

$$\begin{cases} x_{n+1} &= f_n(x_n, u_n, \theta, \eta_n), \\ y_n &= g_n(x_n, \theta, \xi_n), \end{cases} \quad (5.2)$$

where here all the structures are considered to belong to \mathbb{K}^d for some d , i.e.:

- the hidden states $x_n \in \mathbb{K}^{n_x}$,

⁶<https://julialang.org/learning/>

⁷The equivalent of a C++ `double`.

Listing 1 Type Extern for the LNAS model.

```

type Extern
    vec_t::Vector    ## vector of temperatures for all days
    vec_r::Vector    ## vector of radiations for all days
end

```

Listing 2 Type Parameters for the LNAS model for sugar beet.

```

type Parameters
    q_0::Float64    ## initial biomass
    tau_init::Float64    ## initiation thermal time
    mu::Float64     ## radiation use efficiency
    k::Float64      ## Beer-Lambert coefficient
    e::Float64      ## leaf mass per area
    gamma_0::Float64    ## initial leaf allocation coefficient
    gamma_l::Float64    ## final leaf allocation coefficient
    mu_a::Float64      ## allocation median
    sigma_a::Float64   ## allocation standard deviation
    mu_s::Float64      ## senescence median
    sigma_s::Float64   ## senescence standard deviation
    tau_s::Float64     ## senescence thermal time delay
end

```

- the observations $y_n \in \mathbb{K}^{n_y}$,
- the external variables $u_n \in \mathbb{K}^{n_u}$,
- the functional parameters $\theta \in \mathbb{K}^{n_\theta}$,
- the process noise $\eta_n \in \mathbb{K}^{n_\eta}$,
- the observation noise $\xi_n \in \mathbb{K}^{n_\xi}$,
- the transition function f_n ,
- the observation function g_n .

The formalism of the whole modelling system of ADJUSTIN' will be presented in the next sections. All along the way, bits of code for the LNAS model (see Section 2.2) will serve as an illustration of the theoretical programming concepts explained for easier understanding.

5.2.2 Model types

From a computing point of view, the state variables, the external variables and the parameters are stored in Julia's types. The definition of types `State` (to store an x_n), `Extern` (to store all external variables throughout the simulation $[u_n]_{1:T} \in \mathbb{K}^T$) and `Parameters` (to store θ) hence constitute the basis of any model. It has to be noted that external variables are stored for all times in the same type because the values are known for all time steps in advance before the simulation begins, which is not the case of the state variables.

The process and observation noises are stored in different structures because of statistical needs for genericness. A type `ProcessNoise` is defined, the fields of which are of type `Noise`. We will illustrate the principle

Listing 3 Type State for the LNAS model for sugar beet.

```

type State
  q_det::Float64    ## deterministic produced biomass
  q_sto::Float64    ## stochastic produced biomass
  gamma_det::Float64 ## deterministic leaf allocation coefficient
  gamma_sto::Float64 ## stochastic leaf allocation coefficient
  q_l::Float64      ## leaf biomass
  q_gl::Float64     ## green leaf biomass
  q_r::Float64      ## root biomass
end

```

underlying process noises with the LNAS model. We recall that, on day n , the production of biomass q_n^{det} is given by an empirical Beer–Lambert law as in Equation 2.2:

$$q_n^{\text{det}} = r_n \mu (1 - \exp(k q_n^{\text{gl}}/e)). \quad (5.3)$$

To account for the randomness inherent to this biological process, a stochastic value q_n^{sto} is defined such that:

$$q_n^{\text{sto}} = q_n^{\text{det}} (1 + \eta_n^q), \text{ with } \eta_n^q \sim \mathcal{N}(0, (\sigma^q)^2) \quad (5.4)$$

where $\sigma^q \in \mathbb{R}_+^*$, the standard deviation, is the parameter characterizing this process noise. Obviously, this has the apparent drawback of necessitating the storage of two variables at each time step for the seemingly same quantity, q^{det} and q^{sto} within the `State`. However, it allows to automatically compute the transition distribution $p(x_{n+1}|\theta, x_n)$ as will be emphasized in Section 5.5.

A given `Noise` η^* is itself composed of 4 different fields and can be formally defined as $\eta^* = (\lambda_{\text{det}}, \lambda_{\text{sto}}, \ell_d, \theta_d)$ where λ_{det} is a `LabelList` about the deterministic (unnoised) variables in the `State` for this particular process noise, λ_{sto} is the `LabelList` for the corresponding stochastic (noised) variables in the `State`, ℓ_d is a `Label` specifying the kind of noise used (both the distribution d and the way it is applied) and θ_d the parameters of d . When d is univariate, λ_{det} and λ_{sto} contain only one element. The different possibilities for ℓ_d include so far:

$$\left\{ \begin{array}{l} \text{"additive-uniform"}, \\ \text{"additive-normal"}, \\ \text{"additive-lognormal"}, \\ \text{"multiplicative-uniform"}, \\ \text{"multiplicative-normal"}, \\ \text{"multiplicative-lognormal"}. \end{array} \right. \quad (5.5)$$

Noises can be defined for both process and observation noises since, even though they do not play the same role within the model, their programming structure is identical. The `ProcessNoise` of a model is therefore defined as a list of `Noises`, $\eta = [\eta^j]_{1:n_\eta}$, and the same goes for the `ObservationNoise` $\xi = [\xi^j]_{1:n_\xi}$.

Once the `State`, `Extern`, `Parameters`, `ProcessNoise` and `ObservationNoise` types have been defined, it remains to write the transition and the observation functions f_n and g_n . In order to compartmentalize the model as much as possible, several modules can be defined before the transition function. A module m has

Listing 4 Types ProcessNoise and ObservationNoise for the LNAS model for sugar beet.

```

type ProcessNoise
    q::Noise          ## process noise for the production on q
    gamma::Noise      ## process noise for the allocation on gamma
end

type ObservationNoise
    q_gl::Noise       ## observation noise for green leaf biomass
    q_r::Noise        ## observation noise for root biomass
end

```

Listing 5 Module for the production of biomass in the LNAS model for sugar beet.

```

function production!(n, xn, u, p, xnplus1)
    xnplus1.q_det = u.vec_r[n] * p.mu * (1 - exp(-p.k * xn.q_gl / p.e))
end

```

signature:

$$m(n, x_n, u_n, \theta, x_{n+1}) \rightarrow \emptyset \quad (5.6)$$

and it modifies fields of x_{n+1} without returning anything. This is made possible in Julia because x_{n+1} is a mutable State type and Julia exhibits a pass-by-sharing behavior. In practice, modules can be used to isolate equations of different natures. In the case of the LNAS model, we could define different modules for biomass production and biomass allocation for instance.

The transition function can therefore be written using such modules (although not necessarily) and has signature:

$$f(n, x_n, u_n, \theta, \eta) \rightarrow x_{n+1}. \quad (5.7)$$

It is worth noting that the full ProcessNoise structure is passed to this function and not only its realizations as denoted in Equation 5.2. A new State is initialized within the transition function, and either modules are called to modify its field values or explicit modifications occur. The only operation worth to be detailed is that of the noised variables.

Considering the previous example with the produced biomass, there is no explicit update of the stochastic value q^{sto} . Rather, a function $\text{noise}(x_n, \eta^*)$ is called for each Noise η^* and uses the information contained in the latter to modify the values whose Labels are specified within λ_{sto} from those that are specified within λ_{det} in an appropriate way, determined by ℓ_d and θ_d . This function has to be called right after the deterministic values relative to this Noise have been set within the transition function. Julia's reflection makes it possible to access and modify the value of a type, in this case within State. The different fields of a type, such as those of the State, can be accessed using the `getfield` function, which requires the specification of a Symbol. The function `Symbol` allows to convert an AbstractString into a Symbol, such that if `x` is an instance of State, which has fields `q_det` and `q_sto`, the value of the former can be accessed with `getfield(x, Symbol("q_det"))` and the value of the latter can be set with `setfield!(x, Symbol("q_sto"), v)` with some value `v`. This approach allows to access and modify the deterministic and stochastic variables of the

Listing 6 Transition function for the LNAS model for sugar beet.

```

function f(n, xn, u, p, mn)
    xnplus1 = State(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
    thermal_time!(n, xn, u, p, xnplus1)
    production!(n, xn, u, p, xnplus1)
    noise!(xnplus1, mn.q)
    allocation!(n, xn, u, p, xnplus1)
    noise!(xnplus1, mn.gamma)
    senescence!(n, xn, u, p, xnplus1)
    update_total_leaf_biomass!(n, xn, u, p, xnplus1)
    update_green_leaf_biomass!(n, xn, u, p, xnplus1)
    update_root_biomass!(n, xn, u, p, xnplus1)
    return xnplus1
end

```

State specified in λ_{det} and λ_{sto} of a noise. The whole point of such an approach is multiple:

- there is no need to hardcode the noising process or even the Symbols relative to the variables affected by a Noise, the noising process is always the same for all models;
- such LabelLists can be defined within external files different from those of the model or algorithms assuming that all models follow the same syntax;
- such a Noise structure will allow, as will be seen later, to automatically compute the transition and observation pdfs, whatever the model or the process and observation noises.

5.2.3 Simulation core

An observation is the measurement of a given variable at a given time. For a given simulation, observations are stored in a structure referred to as SystemObservation. It is composed of a LabelList containing the name of all the variables observed. Two kinds of variables can actually be observed, i.e. registered within a SystemObservation, during a numerical simulation:

- the ones belonging to the State (hidden states in the HMM formalism);
- those corresponding to actual experimental data (observations in the HMM formalism) and involving observation noises.

Intuitively, one would expect that only observations would be stored inside a SystemObservation, but storing the hidden states proves practical for analyzing the system or computing some pdfs for estimation algorithms. A SystemObservation so can therefore be formalized as:

$$\text{so} = \left(\lambda_o, \left[\left[(t_k, v_k) \in \mathbb{N} \times \mathbb{K} \right]_{1:n_o,i} \right]_{1:n_o} \right). \quad (5.8)$$

The Labels of all the observed variables are stored in the LabelList λ_o , which is of size n_o . To each observed variable λ_o^j corresponds a series of $n_{o,j}$ observation tuples (t, v) where t is the time of the observation and v the value taken by this variable. It is worth noting that, here again, $v \in \mathbb{K}$ and can therefore be an Int, a Float64, an Array{Int} or an Array{Float64} depending on the nature of the observed variable, allowing

for modelling freedom. The function to run a simulation of the model is:

$$\text{simulate}(f, g, x_0, u, \theta; \eta, \xi) \rightarrow \text{so}. \quad (5.9)$$

Here, the process and observation noises η and ξ are key-valued arguments, which means that they need not necessarily be specified, in which case they take on a `Void` default value for the sake of user-friendliness: if a model does not make use of either process noise or observation noise, one needs not worry about them. The observation function g specifies what variables are observed and when. Its exact structure reads:

$$g = [(\ell_j, \tau_j, o_j)]_{1:n_o} \quad (5.10)$$

where ℓ_j is the `Label` of the observed variable, $\tau_j = [t_{j,k}]_{1:n_o,j}$ is the corresponding timeline (time steps at which the variable is observed), and o_j is an observer function whose signature is $o_j(x) \rightarrow v \in \mathbb{K}$ if the observed variable is a hidden state or $o_j(x, \xi) \rightarrow v \in \mathbb{K}$ for real observations. The former returns nothing but `getfield(x, Symbol(1))`, and it would be rather heavy to force the model-writer to define them all. Considering the LNAS model, on day n , the hidden state biomass of leaves $q_n^\ell \in x_n$ is computed and the corresponding observed value is noised to reproduce the error relative to the measurement done by separating plant organs and weighing biomass. We recall that the observation \tilde{q}_n^ℓ corresponding to the hidden state q_n^ℓ is given by:

$$\tilde{q}_n^\ell = q_n^\ell (1 + \xi_n^\ell), \text{ with } \xi_n^\ell \sim \mathcal{N}(0, (\sigma^\ell)^2) \quad (5.11)$$

where $\sigma^\ell \in \mathbb{R}_+^*$, the standard deviation, is the parameter characterizing this observation noise. The noising process is almost the same as for process noises: a function $\text{noise}(x, \xi) \rightarrow v \in \mathbb{K}$ handles the computation of the observed value. This time, instead of modifying values inside `State` x , motivated by the fact that both the deterministic and stochastic values for a process noise belong to the `State`, the observed value is simply returned to be later stored in a `SystemObservation`. The observer function for an observation is therefore nothing but $o_i(x, \xi) = \text{noise}(x, \xi)$. In both cases, defining the observer functions does not require more information than that contained in the `State` x and the observation noises ξ , which is why at the end of the model definition, a call to a macro `@observation_function()` handles the creation of all these functions for further model simulation. This macro is defined in the main module with the following structure:

```
macro observation_function()
    return esc(quote
        ...
    end)
end
```

This apparently unnecessarily complicated formulation was driven by the organization of the whole platform. For historical, practical and workflow reasons, the models are separated from the simulation core, encapsulated in a module, in which `@observation_function()` is defined, the whole point being to prevent the model-designer to write all the observer functions. However, `@observation_function()` makes use of types not defined within the module, such as `State`. The advantage of macros is that they return a compiled expression which is not evaluated at runtime, making possible to refer to the type `State` although the latter is

not defined yet, since models are loaded after the main module. Placing the code necessary for the generation of the observer functions inside a quote `... end` block ensures that the object returned by the macro is of type `Expr`. Since, in Julia, identifiers produced by a macro are renamed to avoid clashes with other variable names, the use of the `esc()` function is required.

The behavior of the `simulate` function is rather intuitive. From the observation function g , the maximum time and the `LabelList` of variables to be observed are deduced. The transition function is then called for each time step t and for each (ℓ, τ, o) in g , the variable ℓ is observed if t is in τ . Depending on the type of the observed variable, the signature of o can vary. Thanks to Julia's introspection capabilities, the signature of a function o can be easily determined. It has to be noted that the way of accessing the latter depends on the version of Julia. Fortunately, version-based conditional statements are very easy and it is straightforward to maintain the platform for different versions of Julia with statements such as:

```

if VERSION >= v"0.4.5"
    sig = methods(o).defs.sig.parameters
elseif VERSION >= v"0.5.0"
    sig = methods(o).mt.defs.sig.parameters
end

```

allowing to call the function with the correct arguments whatever the type of the observer function. All the results are stored in a `SystemObservation` `so`.

The complete formulation of the LNAS model for sugar beet in ADJUSTIN' can be found in Appendix B. The type specification of the fields of the types `State`, `Parameters`, `ProcessNoise` and `ObservationNoise`, indicated with the `::` operator, are not compulsory so that a simpler syntax could be used by the model-designer. Contrarily to other libraries or programs, no DSL was developed as the design of models was judged to be simple enough with the combination of ADJUSTIN's formalism and Julia's concise syntax.

5.2.4 Flexible modelling

An interesting aspect of the Julia language is the flexibility it allows in terms of defining objects, using expressions for example. For instance in [Sainte-Marie et al., 2017], it appeared necessary to define models whose parameters represented coefficients of quadratic polynomials of several dimensions. For benchmarking purposes, multiple polynomials had to be used. Luckily, such code generation is rather straightforward in Julia, it suffices to create the `AbstractString` corresponding to the `Parameters` needed and then make it into an expression, as exemplified by Listing 7. Such code generation requires less than 10 lines of concise code, which would be much more difficult to achieve in most other languages, C++ in particular.

5.3 Platform organization

The model definition does not comprise any initialization of the types, and in particular does not contain any information on the kinds of noises used inside `ProcessNoise` and `ObservationNoise`. The actual

Listing 7 Generation of generic types Parameters.

```

function generate_parameters(d::Int)
    s = "type Parameters\n"
    for i = 1:d s *= "x" * string(i) * "::Float64\n" end
    s *= "end\n"
    ## s is now a String representing raw Julia code defining a type Parameters
    eval(parse(s))
    ## the String is first parsed which returns an Expr
    ## this Expr is then evaluated, thus interpreting actual Julia code
end

## here a type Parameters with 5 Float64 fields is created
generate_parameters(5)
## and can then be instantiated
p = Parameters(1, 2, 3, 4, 5)

```

simulation of a model is illustrated in Listing 8. The module `Adjustin` containing the simulation core must first be loaded with the `using` instruction. It provides all the functions necessary for the simulation without the use of any potential `Adjustin.` namespace. A model can be loaded simply with its name because of the file organization, detailed on Listing 9, where all the models are stored in a specific directory. This actually “copy-paste”s the code defined in the model file, which implies that all the types and functions defined in different model files must have the same name for further reusability: it would not make sense to load a model where `Parameters` is called `Parameter` instead and try to instantiate a `Parameters` afterwards. The whole platform therefore relies on a strict identification of objects by name.

The function `create_observation_function` allows the creation of `g` as long as it is provided an Array of `Tuple{Label, Timeline}`, specifying at what times a variable is observed. If the two variables `q_g1` and `q_r` are to be observed, for instance:

```
g = create_observation_function([("q_g1", [50,70,90]), ("q_r", [60,80,90])])
```

If several observed variables share a common `Timeline`, this can be used to shorten again the argument as:

```
g = create_observation_function([("q_g1,q_r", [50,70,90])])
```

So far only hidden states were observed, if one wants to register data for both the hidden state and the observation of the same variable, one could do so with:

```
g = create_observation_function([("q_g1,obs_q_g1", [50,70,90]), ("q_r,obs_q_r", [60,80,90])])
```

and the keyword `all` provides a very short way to register all the hidden variables in `State` by looping over the elements of `fieldnames(State)`:

```
g = create_observation_function([("all", 1:161)])
```

which is very practical for the automatic registration of heavy models, that is to say containing many states as in the STICS [Brisson et al., 1998] or LNAS Wheat (Section 2.3) models, for further analysis. Loading all the data necessary for a model simulation can easily be accomplished thanks to the function `load_state`, `load_extern`, `load_parameters`, `load_process_noise` and `load_observation_noise` provided that their

Listing 8 Example of a simulation for the LNAS model in ADJUSTIN⁷

```

## calling the main module
using Adjustin

## loading model, in particular this instantiates the transition function f
load_model("lnas")

## defining the observation function
g = create_observation_function(["all",1:161], ("obs_q_gl,obs_q_r",20:20:161))

## loading files from the database required for the simulation
load_state("x0-bourgogne-2011")
load_extern("u-bourgogne-2011")
load_parameters("p-bourgogne-2011")
load_process_noise("pn-0.02")
load_observation_noise("on-0.1")

## simulating
so = simulate(f, g, x0, c, p; pn = pn, on = on)

```

Listing 9 Main structure of the root of the platform

```

algorithms/
applications/
database/
models/
adjustin/
results/

```

argument corresponds to a file in the database of the platform. More details on the platform organization are provided in Appendix D. For instance, a process noise is defined in the file "database/lnas/process-noise/pn_0.02.jl" which contains:

```

pn_q = Noise(["q_det"], ["q_sto"], "multiplicative-normal", (0.0, 0.02))
pn_gamma = Noise(["gamma_det"], ["gamma_sto"], "multiplicative-normal", (0.0, 0.02))
pn = ProcessNoise(pn_q, pn_gamma)

```

Rewriting the same piece of code for every model would seem a bit redundant, the user is therefore provided the opportunity to launch a simulation via command line using already-written applications, stored in the equivalent directory at the root platform. Here the `simulate` application given as an example is rather simple but this proves even more useful in the case of more complicated applications. The Julia package `ArgParse`⁸ allows to pass options to such programs. The simulation of Listing 8 could be done automatically with:

```

./simulate -m lnas -x x0-bourgogne-2011 -u u-bourgogne-2011 -p p-bourgogne-2011 \
--pn pn_0.02 --on on_0.1 -d sim-lnas-bourgogne-2011 \
-o '(["all", 1:161],("obs_q_gl,obs_q_r", 20:20:161))'

```

where `sim-lnas-bourgogne-2011` refers to the name of a directory in which all the results will be stored. More details on this aspect can be found again in Appendix D.

⁸<https://github.com/carlobaldassi/ArgParse.jl>

Listing 10 Macro generating operations for two `ObservationTuples`

```

macro operator_obs_obs(op)
  opn, dopn = symbol("$op"), symbol(".$op")
  return quote
    function $(esc(opn))(obs1::ObservationTuple, obs2::ObservationTuple)
      return (first(obs1), $(esc(dopn))(last(obs1), last(obs2)))
    end
    $(esc(dopn))(obs1::ObservationTuple, obs2::ObservationTuple) =
    $(esc(opn))(obs1::ObservationTuple, obs2::ObservationTuple)
  end
end

```

Listing 11 Macro generating operations for two `SystemObservations`

```

macro operator_so_so(op)
  opn, dopn = symbol("$op"), symbol(".$op")
  return quote
    function $(esc(opn))(so1::SystemObservation, so2::SystemObservation)
      so = SystemObservation(so1.oml)
      for i = 1:length(so1.oml)
        so.otl1[i] = $(esc(dopn))(so1.otl1[i], so2.otl1[i])
      end
      return so
    end
  end
end

```

5.4 Operations on system observations

Results of a model simulation are stored in a `SystemObservation`. Many statistical applications require to be able to make many different operations on such structures (such as computing the sum, the difference, the quotient, the mean or the variance). The Julia language makes it possible to generate all these operations with very concise code, using macros that will generate such functions for the different operations. First these operations are defined on the `ObservationTuples` $(t, v) \in \mathbb{N} \times \mathbb{K}$ (see Listing 10), these operations can then be reused to be defined on `SystemObservations` (see Listing 11).

Here the `esc` function ensures that the function created carries the exact same name as the operator, and is not replaced with a name chosen by the macro-expander for name-collision avoidance. In the case of the `+` operator, `opn` would denote the operator `+` and `dopn` the operator `.+`, which is an operator per se and cannot be retrieved with only `.$op`, whence the conversion to an `AbstractString` then `Symbol` for these operators.

Once these macros have been defined, they can be used to generate any operation as:

```

opl = [+ , - , * , /]
for op in opl
  @eval @operator_obs_obs($op)
  @eval @operator_so_so($op)
end

```


where, since the macro expansion happens before the loop runs, at which time the `op` operator is not defined yet, the `@eval` macro needs to be called with `$op` to prevent Julia from complaining about an unexistent `op`. The different operators `+`, `-`, `*`, `/` are then passed to the macro and the functions:

```
+(obs1::ObservationTuple, obs2::ObservationTuple)
+(so1::SystemObservation, so2::SystemObservation)
...
```

are then automatically generated. It then becomes possible to perform operations such as:

```
so_1 = simulate(f, g, x0, u, p; mn = mn, on = on)
so_2 = simulate(f, g, x0, u, p; mn = mn, on = on)
so_sum = so_1 + so_2
```

This way of defining all these functions presents several advantages: not only does it avoid code duplication thus reducing significantly the whole size of the code, it also makes it possible to adapt very easily the internal structure of a `SystemObservation`. Being a research platform, ADJUSTIN' is very likely to be regularly modified, which is why the ability to change easily the internal structure of the main types is of primary importance.

In the same manner, if `so` is a `SystemObservation`, it is possible to easily define among others `sqrt(so)`, `so^2` which makes the computation of the mean and variance of a `SystemObservationList` straightforward. Another approach to compute such mean and variance would be to first convert the `SystemObservationList` to a `Matrix` before computing its mean and covariance. However, the information on the variables and times of observations would be lost. In general, working in ADJUSTIN' proves much easier when using `SystemObservations`.

5.5 Probability density functions

Statistical distributions play a crucial role in HMMs. As described in Chapter 3, in order to generate actual values for parameters or hidden states, one needs to explicitly sample from standard distributions. Furthermore, for statistical inference within this type of models, actual values of some pdfs (namely for the prior, transition and observation distributions) need to be calculated.

5.5.1 Sampling

The sampling procedure is described for parameters since it is exactly the same for states. The prior distribution on a given group of parameters is contained in a `SamplingRule` $s = (\lambda, d)$ where λ is a `LabelList` specifying which parameters are sampled (to account for possible multivariate distributions) and d is the distribution to sample from. A `SamplingRuleList` $[s_j]_{1:n_r}$ is a set of `SamplingRule` allowing to sample several parameters following different distributions at once, and n_r is the number of rules involved. The function

`sample(θ^0 , $[s_j]_{1:n_r}$, N) \rightarrow $[\theta^i]_{1:N}$` allows to generate N Parameters from θ^0 where the parameters specified in each λ_j of the different `SamplingRules` s_j have been replaced with their sampled values. This is easily done as it suffices for $i \in \llbracket 1, N \rrbracket$ to make a copy θ^i of θ^0 , sample a value or vector $v_j^i \in \mathbb{K}$ according to the distribution of each sampling rule s_j , and to replace the corresponding value in θ^i with `fill!(θ^i , v_j^i , λ_j)`.

5.5.2 Prior distribution

The pdf related to the prior is:

$$p(\theta) = \prod_{j=1}^{n_r} p(\theta_j) \quad (5.12)$$

where each parameter to be estimated θ_j follows a distribution \mathcal{D}_j . We recall that $\theta_j \in \mathbb{K}$, hence \mathcal{D}_j can be either univariate or multivariate. For the case of the LNAS model for sugar beet, if the two parameters $(\mu, e) \subset \theta$ were to be estimated, and assuming that their prior distributions are $\mu \sim \mathcal{N}(m_\mu, s_\mu^2)$ and $e \sim \mathcal{U}(a_e, b_e)$, then the prior pdf becomes:

$$p(\theta) = p(\mu) p(e) = \frac{1}{s_\mu \sqrt{2\pi}} \exp\left(-\frac{(\mu - m_\mu)^2}{2s_\mu^2}\right) \frac{\mathbb{1}(e \in [a_e, b_e])}{b_e - a_e}. \quad (5.13)$$

It therefore becomes possible to compute the prior pdf as `pdf(θ , $[s_j]_{1:n_r}$)` as the product of the different `pdf(θ , s_j)` where `pdf(θ , s_j)` can be computed by evaluating, for a `SamplingRule` $s_j = (\lambda_j, d_j)$ the pdf of the distribution d_j at the point in θ_j specified by the variable λ_j . In ADJUSTIN', this would translate into:

```
load_model("lnas")
load_parameters("p-bourgogne-2011")

m_mu, s_mu = 3.7, 0.5
a_e, b_e = 0.6, 0.8
srl = [SamplingRule(["mu"], "normal", (m_mu, s_mu)),
       SamplingRule(["e"], "uniform", (a_e, b_e))]

## assuming that in p-bourgogne-2011, mu = 3.8 and e = 0.65
## the following line returns as expected 3.9 = 0.78 * 5.0
pdf = prior_pdf(p, srl)
```

Possible choices for an automatic computation of the prior pdf currently include the following distributions:

$$\left\{ \begin{array}{l} \text{"normal",} \\ \text{"uniform",} \\ \text{"poisson",} \\ \text{"multivariate-normal",} \\ \text{"gamma",} \\ \text{"inverse-gamma",} \\ \text{"wishart",} \\ \text{"inverse-wishart".} \end{array} \right. \quad (5.14)$$

5.5.3 Transition and observation distributions

We recall from Section 1.2 that the transition and observation pdfs can be written as:

$$p(x_{n+1}|\theta, x_n) = p(\eta_n|\theta) = \prod_{j=1}^{d_\eta} p(x_{n+1}^{m_\eta(j)+1}|\theta, x_{n+1}^{m_\eta(j)}) \quad (5.15)$$

and:

$$p(y_n|\theta, x_n) = p(\xi_n|\theta) = \prod_{j=1}^{d_\xi} p(y_n^j|x_n^{m_\xi(j)}) \quad (5.16)$$

respectively. Just as `SamplingRules` allow the computation of the prior pdf, the definition of `ProcessNoise` and `ObservationNoise` jointly with Equations 5.15 and 5.16 permit the computation of the transition pdf as `transition_pdf(x, η)` as the product of the different `pdf(x, ηj)`, where `pdf(x, ηj)` exploits the information contained in the `Noise ηj`. If a given `Noise` is given as:

$$\eta = (\lambda_{\text{det}}, \lambda_{\text{sto}}, \ell_d, \theta_d) \doteq (\text{ll_det}, \text{ll_sto}, \text{kind}, \text{vl}) \quad (5.17)$$

in programming notations, if `kind` is "multiplicative-normal" then the underlying distribution is univariate and the deterministic and stochastic values are retrieved from the `State x` for the `Noise n` with:

```
x_det = getfield(x, Symbol(n.ll_det[1]))
x_sto = getfield(x, Symbol(n.ll_sto[1]))
```

and the associated value of the pdf is:

```
pdf = normal_pdf(x_sto, x_det * (1 + n.vl[1]), abs(x_det * n.vl[2]))
```

where `normal_pdf(x, μ, σ)` computes the pdf of the normal distribution with mean μ and standard deviation σ at point x . All the same, if `kind` = "additive-uniform", the associated pdf is:

```
pdf = uniform_pdf(x_sto, x_det + n.vl[1], x_det + n.vl[2])
```

where `uniform_pdf(x, α, β)` computes the value of the pdf of the uniform distribution with lower bound α and upper bound β at point x . Looping over the different `Noises` of a `ProcessNoise` therefore allows to compute the whole transition pdf. The principle is almost the same for the observation pdf where `observation_pdf(so, x, ξ)` is computed as the product of the different `pdf(x, ξj)`. This formalism allows to compute the pdfs of multivariate distributions as well, which is of interest for models where different observations are thought to be correlated.

5.6 Algorithms

ADJUSTIN' is not restricted to a particular field of study and aims at being a platform for a full statistical analysis of general SSMs. It therefore includes routines for sensitivity analysis, parameter and state estimation (both from frequentist and Bayesian points of view), model selection, data assimilation and hierarchical

models, with the objective of performing all the usual steps of model design process as described for instance in [Van Waveren et al., 1999].

The combination of Julia's concise syntax and the genericness of the modelling paradigm adopted throughout the platform makes the design of algorithms relatively easy. When designing models comprising a great number of parameters, it is of primary importance to identify the most influential ones. Typically, a sensitivity analysis method will be run in order to rank the different model parameters according to their impact on the variance of the model output. Once the most important model parameters have been identified using sensitivity analysis, they need to be estimated from experimental data. Such an estimation procedure can be performed in either a frequentist or a Bayesian approach. Frequentist algorithms implemented in ADJUSTIN' include the least squares family, in particular GLS or algorithms such as the Expectation-Maximization (EM) algorithm [Dempster et al., 1977]. Estimation techniques from a Bayesian perspective is achieved with algorithms belonging to the MCMC and SMC families. The latter comprises several different algorithms such as the UKF, the EnKF and the RPF. We will now describe how some of these algorithms work inside the ADJUSTIN' platform.

5.6.1 Sobol method

We recall from Section 3.2.1 that for a single parameter θ_j , the first order sensitivity index and the total order sensitivity index on the value of variable ℓ at time step n are defined as:

$$\begin{cases} S_{nj}^\ell &= \frac{\mathbb{V}_j(\mathbb{E}_{-j}(y_n^\ell|\theta_j))}{\mathbb{V}(y_n^\ell)}, \\ T_{nj}^\ell &= \frac{\mathbb{E}_{-j}(\mathbb{V}_j(y_n^\ell|\theta_{-j}))}{\mathbb{V}(y_n^\ell)}, \end{cases} \quad (5.18)$$

where j refers to taking into account only variable θ_j and $-j$ all variables θ_k for $k \neq j$. These expectations and variances are computed using a Monte Carlo procedure [Saltelli et al., 1993] and the convergence of the algorithm can be improved by using the permutation of certain values of parameters [Wu et al., 2012]. Four different sets of parameters and corresponding model outputs are considered, with respective identifiers s for sampling, s' for resampling, c for complementary sampling and c' for complementary resampling. Four lists of Parameters $[\theta^{s,i}]_{1:N}$, $[\theta^{s',i}]_{1:N}$, $[\theta^{c,i}]_{1:N}$, $[\theta^{c',i}]_{1:N}$ are first sampled using $[\theta^i]_{1:N} = \text{sample}(\theta^0, [s_j]_{1:n_r}, N)$ where the SamplingRuleList $[s_j]_{1:n_r}$ prescribes the sampling distribution on each parameter, and N is the number of samples. Assuming that two Parameters $p1$ and $p2$ need to exchange a particular variable of name `label`, this can be done by accessing once again the fields of the type with:

```
function switch!(p1::Parameters, p2::Parameters, label::Label)
    s = Symbol(label)
    v1 = getfield(p1, s)
    v2 = getfield(p2, s)
    setfield!(p1, s, v2)
    setfield!(p2, s, v1)
end
```

The corresponding model outputs are then generated with:

$$[y^j]_{1:N} = \text{simulate}(f, g, x_0, u, [\theta^j]_{1:N}, \eta, \xi) \quad (5.19)$$

for the four parameters lists. The output of a simulation being a `SystemObservation`, computing the sensitivity index for a given observation requires the extraction of all its values. The value of the observation related to the variable ℓ at time n can be retrieved using $v_s = \text{to_vec}([y^{s,i}]_{1:N}, \ell, n)$, and similarly for the other lists. These four `Array{Float64}`s can then be used to perform all vector operations needed to compute the sensitivity indices.

The main steps of the Sobol algorithm therefore amount to parameter sampling following a set of prior rules, model simulation with a given set of parameters, parameter switching and computation on arrays, which are made very easy thanks to both Julia's concise syntax and ADJUSTIN's formalism. The generalized version of the Sobol algorithm [Sainte-Marie et al., 2017] described in Section 3.2.2 has also been implemented in ADJUSTIN' and integrated very well in the modelling framework.

5.6.2 Adapted Metropolis within Gibbs

MCMC algorithms have attracted a lot of attention [Gilks, 2005] [Berg and Billoire, 2007] for the analysis of complex statistical models in a Bayesian perspective since they allow for the evaluation of complex and high-dimensional integrals, whence for the estimation of parameters and hidden states. Unsurprisingly, many libraries are devoted to such algorithms.

In this section, we try to highlight how the ADJUSTIN' platform allows for easy modelling of MCMC algorithms by considering the special case of the Adapted Metropolis within Gibbs (AMWG) algorithm detailed in Algorithm 5.

Each algorithm in ADJUSTIN' owns a `Configuration` in which are stored all the variables specific to the algorithm in question. For the AMWG algorithm, this includes `SamplingRuleLists` for the prior distributions of the parameters and hidden states, and the number of total iterations as well as the burn-in length for the two MCMC chains generated. The initial value of the parameters are as usual sampled according to its prior distribution prescribed in the initial `SamplingRuleList` $[s_j]_{1:n_r}$: $\theta^0 = \text{sample}(\theta, [s_j]_{1:n_r})$. One of the arguments passed to the function is a `SystemObservation` representing the experimental data $y_{1 \rightarrow T}$ used for the estimation. A first step is to create the observation function for this specific set of observations, taking into account which variables are observed at what times. This can be achieved using:

```
g = create_observation_function(get_simulation_observer_model_list(so_exp, on))
```

Thanks to the structure of the `ObservationNoise` `on`, we are able to determine which hidden states correspond to the observations and therefore construct the observation function that will be used for the model simulation inside the MCMC loop. The hidden states corresponding to θ^0 can then be simulated using $x_{1:T}^0 = \text{simulate}(f, g, x_0, u, \theta^0; \eta, \xi)$ where the `ProcessNoise` η and the `ObservationNoise` ξ have been

provided to the algorithm. All the variables involved in the adaptive scheme are simply initialized with the values given in Algorithm 5. Inside an MCMC iteration, a candidate for the parameters is sampled according to a multivariate random walk with `SamplingRule(λ_θ , "mvnormal", $(\theta^t, \lambda^t \Sigma^t)$)` together with the `sample` function and the corresponding hidden states simulated from this candidate. We recall that the first acceptance probability for the acceptance of a candidate θ^* is:

$$\alpha((\theta^*, x_{1:T}^*), (\theta^t, x_{1:T}^t)) = 1 \wedge \frac{p(y_{1 \rightarrow T} | \theta^*, x_{1:T}^*)}{p(y_{1 \rightarrow T} | \theta^t, x_{1:T}^t)} \frac{p(\theta^*)}{p(\theta^t)}. \quad (5.20)$$

At iteration t , only the numerator of this expression is computed as the denominator has already been at the previous iteration. For numerical stability, the logarithm of this numerator is actually computed using:

```
log_likelihood_observation_pdf(so_sim, on, so_exp) + log_pdf(p, srl)
```

where `so_sim` and `so_exp` correspond to the `SystemObservations` from the model simulation and the experimental data respectively and `p` to the candidate parameter. The first term represents the pdf of all observations given the parameters and the associated hidden states $p(y_{1 \rightarrow N} | \theta^*, x_{1:N}^*)$ and the second term is related to the prior distribution $p(\theta^*)$. The exact value of the acceptance probability α can then be inferred and the candidate accordingly accepted or rejected. The value of the chain is stored in a `Matrix` of size $d \times N$. Finally, the variables of the adaptive schemes are straightforwardly updated using Equation 3.40. At the end of the MCMC loop, the mean and the covariance for the parameters can be easily computed from the results matrix.

Once an estimate for the parameters have been computed, a similar procedure can be undertaken for the second part of the algorithm. The acceptance probability now reads:

$$\alpha(x_n^t, x_n^*) = 1 \wedge \frac{p(x_{n+1}^t | \tilde{\theta}_n^t, x_n^*)}{p(x_{n+1}^t | \tilde{\theta}_n^t, x_n^t)} \frac{p(y_n | \tilde{\theta}_n^t, x_n^*)}{p(y_n | \tilde{\theta}_n^t, x_n^t)} \quad (5.21)$$

and it is computed with the transition and observation pdfs using the mechanism described in Section 5.5.

In practice, running an AMWG algorithm does not require much and is illustrated in Listing 12. Just as each algorithm owns a `Configuration` type, each algorithm also owns a `Result` type to store all the results specific to it. For AMWG, one can notably choose to save the estimates of the means and covariances for the parameters and hidden states, the acceptance ratio for each iteration and the matrix of all results if necessary for further analysis (this is not done automatically and this can be computationally expensive according to circumstance). This saving procedure is entirely handled by the `save` function which saves in the right format the different fields (whether they be `Vectors`, `Parameters` or `SystemObservations`) of the `Result` of an algorithm in the results directory provided in argument.

5.6.3 Regularized particle filter

Another approach to parameter and state estimation is the use of filters. One of the simplest ones used in ADJUSTIN' is the UKF, whose code is given in Appendix C in order to illustrate the ease of implementation of such algorithms in ADJUSTIN'. In the following, we will focus on the RPF described in Algorithm 8 and

Listing 12 Example of an AMWG algorithm being run for the LNAS model in ADJUSTIN'

```

using Adjustin

## creating a results directory
str_dir = create_results_directory()

## loading data and algorithm configuration
load_model("lnas")
load_state("x0-bourgogne-2011")
load_extern("u-bourgogne-2011")
load_parameters("p-bourgogne-2011")
load_process_noise("pn-0.02")
load_observation_noise("on-0.1")
load_configuration("cfg-bourgogne-2011")

## running algorithm
r = amwg(f, g, x0, u, p, so_exp, cfg; pn = pn, on = on, full_results = false)

## saving all its results in the results directory
save(r, str_dir * "result.txt")

```

dive a bit into the details of its implementation, all the more so since it can be almost fully parallelized. The MPI Julia package provides a convenient wrapper for the portable message passing system Message Passing Interface (MPI) [Gabriel et al., 2004]. In MPI, the same program is launched on different processes and these processes can communicate between each other to exchange information. The RPF is an algorithm particularly suited for such a parallel implementation since, as will be detailed later, only a possible resampling operation is not fully parallelized.

The RPF aims at jointly estimating parameters and hidden states of the model by propagating particles from experimental time to experimental time. Again, the observed variables need not be the same at all times. Let $p \in \llbracket 1, n_p \rrbracket$ represent the index of a parallel process where n_p is the total number of processes. A variable relative to a particular process p will be denoted with the superscript (p) . Let N be the number of particles (samples) and $N^{(p)} = N/n_p$ (assumed to be an integer) be the number of particles relative to a given process.

The initial population of particles is first sampled according to a prior distribution for both the parameters and the states. This translates into:

$$\begin{cases} [\theta^{i,(p)}]_{1:N^{(p)}} = \text{sample}(\theta^0, [s_j]_{1:n_r}, N^{(p)}) \\ [x^{i,(p)}]_{1:N^{(p)}} = \text{sample}(x^0, [s'_j]_{1:n'_r}, N^{(p)}) \end{cases} \quad (5.22)$$

where θ^0 and x^0 are initial parameters and state, $[s_j]_{1:n_r}$ and $[s'_j]_{1:n'_r}$ are the `SamplingRuleLists` for the estimated parameters and hidden states respectively, and the corresponding weights are initialized uniformly as $[w^{i,(p)}]_{1:N^{(p)}} = [1/N]_{1:N^{(p)}}$. Notice that all these lists contain $N^{(p)}$ elements so that the union of these sets over the different processes contain N elements, and that the sum of all the weights over all processes is 1. These particles are then propagated through the model in order to predict the hidden states at the next experimental time:

$$[y^{i,(p)}]_{1:N^{(p)}} = \text{next_prediction!}(f, g, [x^{i,(p)}]_{1:N^{(p)}}, u, [\theta^{i,(p)}]_{1:N^{(p)}}, \eta, \xi, t_k). \quad (5.23)$$

This function starts the simulation of the model from time step t_k and stops as soon as it encounters an experimental time stored in the observation function g , which has been constructed from the experimental data, denoted y^{exp} in this section. The transition function f and the observation function g are obviously the same for all the simulations since they are part of the model considered. The external variables u are assumed to be identical for the different simulations as well. Here, the parameters defining the process and observation noises η and ξ are considered known. The list of parameters $[\theta^{i,(p)}]_{1:N^{(p)}}$ is constant throughout the model simulation between t_k and t_{k+1} . Contrarily, the input list of states $[x^{i,(p)}]_{1:N^{(p)}}$ represent the values of the States for the different particles at time t_k , and `next_prediction!` modifies it so that at the end it contains the States at time t_{k+1} .

Each weight in $[w^{i,(p)}]_{1:N^{(p)}}$ is then multiplied with a new weight equal to $p(y^{\text{exp}}|\theta^{i,(p)}, x^{i,(p)})$. The function allowing to compute the latter, `compute_weight`($\xi, y, y^{\text{exp}}, t_{k+1}$), has the following behaviour: looping over the different fields of the `ObservationNoise` ξ , if the observed variable y^j relative to this Noise is observed at time t_{k+1} in y^{exp} , then the weight is multiplied by $p(y_{t_{k+1}}^{\text{exp},j}|x_{t_{k+1}}^{m_\xi(j)})$, see Equation 5.16. The weights are multiplied to keep track of the history of a particle until there is a resampling step, at which point all weights are uniformly reset. In order to compute the effective sampling size, the weights need to be normalized. The total sum of the weights must be computed, which amounts to computing the sum of the weights on each process and compute the sum of these sums over the different processes with:

```
w_sum = MPI.Allreduce(sum(wl_p), MPI.SUM, MPI.COMM_WORLD)
```

where `wl_p` is the list of weights on process p . The `Allreduce` operation ensures that the sum of the local sums is distributed among all processes. The local weights can then be divided by this value to ensure that $\sum_{p=1}^{n_p} \sum_{i \in r_p} w^{i,(p)} = 1$, where $r_p = \llbracket 1 + (p-1)N^{(p)}, pN^{(p)} \rrbracket$ is the range of indices in the population relative to process p . The weighted mean and weighted covariance are then computed in parallel, as detailed in Appendix E.

Since the weights have been normalized, the effective sampling size [Liu and Chen, 1995] reduces to:

$$n_{\text{eff}} = \left(\sum_{p=1}^{n_p} \sum_{i \in r_p} (w^{i,(p)})^2 \right)^{-1} \quad (5.24)$$

and can thus be obtained by computing the local sum of the weights squared and reducing once again over all processes, it is nothing but:

```
n_eff = 1 / MPI.Allreduce(sum(wl_p .^ 2), MPI.SUM, MPI.COMM_WORLD)
```

If n_{eff} falls under a given threshold, say half the total number of particles, then resampling occurs. This operation is not fully parallelized, but it has to be noted that it does not happen at every experimental time, only at those when the effective sample size is too low. If it cannot be avoided, it can still be done in a way that minimizes the transfer of information between the different processes; this is detailed in Appendix F.

The regularization step allows to convert the discrete approximation of the filtering density into an absolutely continuous approximation by using kernels. Whenever resampling occurs, the particles are sampled from

the smoothed distribution:

$$p(z|y_{0 \rightarrow T}) = \sum_{p=1}^{n_p} \sum_{i \in r_p} w^{i,(p)} k_{h_n}(z - z^{i,(p)}), \quad (5.25)$$

where we recall that $z = (\theta, x)$ is the augmented state. For a Gaussian kernel, it amounts to perturbate the different particles as:

$$\begin{pmatrix} \theta^{i,(p)} \\ x^{i,(p)} \end{pmatrix} += \begin{pmatrix} \delta\theta^{i,(p)} \\ \delta x^{i,(p)} \end{pmatrix} \text{ where } \begin{pmatrix} \delta\theta^{i,(p)} \\ \delta x^{i,(p)} \end{pmatrix} \sim \mathcal{N}(0, \Sigma). \quad (5.26)$$

The matrix Σ is function of the covariance matrix of all the particles which had already been computed in parallel before. The regularization step can therefore be performed on each process in a parallel manner.

The strong scalability of the implementation has been evaluated: a RPF algorithm is run on n_p processes, where the total number of particles is kept constant, the number of particles per process being $10^6/n_p$. Just as in the MCMC case, simulated data was obtained, this time for $t = 50:150$, with $\mu = 3.6$, $\gamma^0 = 0.75$ and $\mu^a = 600$; the priors given in the RPF configuration were $\mu \sim \mathcal{N}(4.32, 1.44)$, $\gamma^0 \sim \mathcal{N}(0.6, 0.3)$ and $\mu^a \sim \mathcal{N}(550, 75)$. Results for the speed-up t_1/t_{n_p} are displayed on Figure 5.1, where t_{n_p} denotes the walltime for n_p processes. The continuous red line represents the ideal scalability for which the elapsed time decreases linearly with the number of processes. For a number of processes smaller than 96, the speed-up is slightly above the ideal scalability because of parallel overhead due to communications between processes (particularly during the resampling step), whereas for 168 processes, the speed-up is below the ideal scalability. This local behaviour could be partly explained by cache effects.

The second case aims at appraising the weak scalability of our implementation. Once again, a RPF algorithm is run on n_p processes; this time the number of particles on each process is kept constant at 10^4 . Results for the normalized time t_p/t_1 are also displayed on Figure 5.1. The continuous red line indicates the ideal weak scalability, where the walltime would remain constant. Two phases can be observed: when tests are executed on a single node ($n_p \in \{1, 2, 4, 8, 12, 24\}$), the available memory bandwidth saturates as the number of processes increases, which prevents an efficient scalability. When tests are carried out on multiple fully loaded nodes ($n_p \in \{24, 48, 96, 168\}$), the walltime scales much better with the number of processes, although there is an expected minor additional overhead due to communications between nodes, much less significant than the one during the first phase.

These results show that the RPF algorithm has successfully been parallelized for concrete applications, where a number of particles of 10^6 is sufficient for accurate parameter and state estimation within such state spaces. Computing times were efficiently scaled with the number of cores and significantly reduced: for instance, with a sequential time of $t_1 = 3\text{h}40\text{m}04\text{s}$, we managed to achieve an actual computing time of $t_{168} = 1\text{m}35\text{s}$ on 168 cores, which represents a crucial practical improvement for the everyday user. At times when computing clusters are easily accessible, it is of primary importance to be able to take advantage of such resources in an optimal manner, and this demonstrates that Julia is perfectly suited to efficient computing for concrete applications using state of the art algorithms.

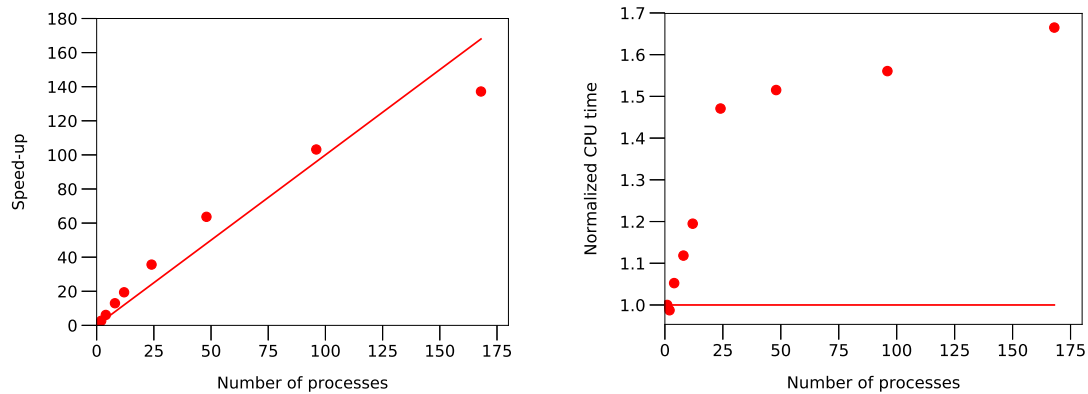


Figure 5.1: Left: performance of the parallel implementation of the RPF, displaying the speed-up t_1/t_{n_p} w.r.t. the number of processes (total number of particles across the different processes fixed at 10^6). Right: normalized time t_p/t_1 necessary to perform the RPF when each process has a constant number of particles, fixed at 10^4 .

5.6.4 Particle marginal Metropolis–Hastings

As described in Section 3.5.1, MCMC and SMC algorithms can be combined to design PMCMC algorithms. In particular, a particle marginal Metropolis–Hastings (PMMH) sampler has been implemented in ADJUSTIN³: basically, it suffices to design a standard MCMC algorithm and to call a particle filter at each iteration for state estimation. The design of the platform allows to easily switch between different filters by prescribing which one to use in the configuration of the PMCMC algorithm: a filter such as the UKF would be fast but with a few particles, whereas filters such as the EnKF or the RPF can use more particles, improving the precision of the estimation but requiring higher computing times. This switching procedure can easily be implemented as follows:

```
function filter_function(name)
    if name == "ukf"
        return ukf
    elseif name == "enkf"
        return enkf
    elseif name == "rpf"
        return rpf
    else
        error("Wrong name of filter.")
    end
end
```

and inside the MCMC procedure, the estimates for the hidden states come straightforwardly as:

```
filter = filter_function(cfg.str_filter)
rf = filter(f, g, x0, u, p, so_exp, cfg.cfg_filter; mn = mn, on = on)
```

where the Configuration `cfg` of the PMMH algorithm contains both the name of the filter used (in `str_filter`) and the own Configuration of this filter (in `cfg_filter`). The results returned by either

of the filters are then stored in `rf` and can be further used for other operations.

5.6.5 Adaptive hybrid Metropolis–Hastings–Gibbs

Last but not least, the population approach has been integrated into the platform. We discuss it rather briefly as most of the aspects of the adaptive hybrid Metropolis–Hastings–Gibbs (AHMHG) algorithm (see Algorithm 12) have already been presented for the other algorithms. What remains worth mentioning is that it is perfectly suited to a parallel implementation as the Metropolis–Hastings for each individual can be performed independently. We assume that as many MPI processes as individuals can be used, which was the case in practice for this thesis. This means that, on each process, one can sample the individual parameters for a single individual, compute the corresponding hidden states and accordingly deduce the acceptance probability for this candidate. After acceptance or rejection, the variables related to the adaptive schemes of this individual can be updated. Once this has been performed for all individuals, it remains to sample population parameters. The hyperparameter Λ^* used for sampling a new value of η requires the mean of the individual parameters which can be computed from the different processes efficiently as already seen for the RPF. Sampling new values for Σ and τ requires to compute at iteration t :

$$\left\{ \begin{array}{l} \sum_{i=1}^N (\theta_i^t - \eta^t)(\theta_i^t - \eta^t)^T \\ \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i)) \end{array} \right. \quad (5.27)$$

in order to update Ψ^* and β^* respectively, which can be easily done provided that the shape of the data be carefully handled when vectorizing and devectorizing it as needed by the `MPI.Allgather` function. Once the hyperparameters of the distributions of the population parameters have been updated, only one occurrence of η , Σ and τ needs to be sampled on, say, the 1st process, and broadcast to all other processes so that the same chains are generated from each individual's point of view.

5.7 Discussion

A new platform for the statistical analysis of hidden Markov models was introduced. The importance of such a tool is all the more important since HMMs have a tremendous number of applications. Far from being redundant with other existing tools, ADJUSTIN' makes use of both a precise mathematical formalism and the recently developed Julia language.

A generic approach was designed to handle randomness within HMMs, notably concerning process and observation noises. It allows automatic computations of prior, transition and observation pdfs for the estimation of parameters and hidden states. A wide variety of algorithms have thus been implemented in ADJUSTIN' for the analysis of HMMs. For each of them, easily customizable graphical outputs are automatically generated, which eases the interpretation of results and thereby workflow.

ADJUSTIN⁹ takes advantage of the conciseness and the practicality of the Julia language. Some elegant aspects of the latter (versioning, code generation, scripting, etc.) were highlighted. Furthermore, its refined syntax makes it suitable for model design, making the platform a unified Julia tool, with no need to rely on a domain specific language. Being also very efficient (at once for sequential and parallel programs), Julia seems particularly suited for applications in the academic research, though it does not get the attention it deserves yet.

Some improvements are under consideration and development, such as optimal experimental design, see [Rasch and Bücker, 2010] for an example within an interactive environment. To achieve even better performance, it might be interesting to implement some algorithms in CUDA rather than MPI, using the CUDArt⁹ package.

⁹<https://github.com/JuliaGPU/CUDArt.jl>

Chapter 6

Estimation in state space models: application

THIS CHAPTER FOCUSES ON SEVERAL CASE STUDIES in the context of general state space models (SSMs) for a single individual, the population approach being considered in Chapter 8. Three plant growth models have been presented in Chapter 2 using the mathematical framework of general SSMs introduced in Chapter 1 and each of these models exhibit special features:

- the GreenLab model for *Arabidopsis thaliana*, which is an organ-scale plant model, aims at describing the evolution of the area of each leaf in order to take advantage of the whole history of plant growth for genotypic differentiation. We will proceed to a first simple calibration of the model in order to acquire a preliminary understanding of this model using two estimation methods, generalized least squares and adapted Metropolis within Gibbs, which is the occasion to compare the estimates that the two algorithms provide and assess the relevance of each method on simulated data;
- the LNAS model for wheat, involving more functional parameters but no process noise, may be more difficult to calibrate, particularly in concrete applications where data from wheat fields are particularly scarce and heterogeneous. This is actually the first time that the LNAS model for wheat is used and issues related to its calibration need to be studied carefully. In particular, this model is aimed at being used for data assimilation in order to improve model predictions of variables of interest by taking advantage of early data being available and where the parameters and hidden states are corrected at each observation time;
- the LNAS model for sugar beet, a compartmental model, involved a relatively few number of functional parameters, but is designed with both process and observation noise, and this dual randomness may hinder the estimation of parameters and hidden states. Particle Markov chain Monte Carlo algorithms were designed to overcome this kind of difficulties and obtain an accurate estimation of hidden states. Furthermore, the latter is an essential prerequisite to the estimation of process and observation noise.

The Bayesian estimation methods presented in Chapter 3 will hence allow to tackle issues specific to each of them. These case studies were all performed using the ADJUSTIN' computing platform (see Chapter 5). It is hoped that its diversity, efficiency and ease of use will be highlighted through these different case studies.

6.1 Comparison of frequentist and Bayesian approaches for the GreenLab model

The GreenLab model for *Arabidopsis thaliana*, which will further be used in the context of Bayesian hierarchical modelling in Chapter 8, is investigated using two popular estimation methods: a frequentist approach, the generalized least squares (GLS) algorithm, which represents one of the most standard frequentist approaches used for model calibration (see Section 3.4), and a Bayesian one, the adapted Metropolis within Gibbs (AMWG) algorithm (see Section 3.5.1). If the former provides punctual estimates for the parameters coupled with a variance covariance matrix for the errors, the latter provides a posterior distribution from which a mean and a standard deviation can be inferred. In order to compare the results that these two approaches yield in the case of a single *Arabidopsis thaliana* individual, we generate realistic data from a known set of parameters and estimate some of the most influential parameters, determined thanks to a sensitivity analysis method.

6.1.1 Sensitivity analysis

We first start by using Sobol method, described in Section 3.2.1, in order to identify the most influential parameters of the GreenLab model for *Arabidopsis thaliana*. The external variables were, as will be the case all along this section, fixed to those used for the real experiments described in Chapter 7. We chose to consider the global output of the total leaf area $q_n^{\text{tot}} = \sum_{v \in \llbracket 1, \nu_n \rrbracket} q_n^v$ from time step $t = 1$ h (at the very beginning of plant growth) up until $t = 184$ h which corresponds to the 23rd day on the phenotyping platform for the plant (see Chapter 7) and every ten time steps inbetween.

Both the Beer–Lambert extinction coefficient k and the initial biomass of the seed q_0 were considered known.

The remaining model parameters were sampled from uniform distributions:

$$\left\{ \begin{array}{l} \phi \sim \mathcal{U}(9.000 \cdot 10^{+00}, 1.500 \cdot 10^{+01}), \\ \mu \sim \mathcal{U}(2.362 \cdot 10^{+00}, 3.937 \cdot 10^{+00}), \\ s \sim \mathcal{U}(3.750 \cdot 10^{+00}, 6.250 \cdot 10^{+00}), \\ e \sim \mathcal{U}(1.271 \cdot 10^{-03}, 2.119 \cdot 10^{-03}), \\ \mu_1 \sim \mathcal{U}(3.306 \cdot 10^{+00}, 5.511 \cdot 10^{+00}), \\ \sigma_1 \sim \mathcal{U}(3.445 \cdot 10^{-01}, 5.742 \cdot 10^{-01}), \\ \mu_2 \sim \mathcal{U}(4.013 \cdot 10^{+00}, 6.689 \cdot 10^{+00}), \\ \sigma_2 \sim \mathcal{U}(2.994 \cdot 10^{-01}, 4.990 \cdot 10^{-01}), \\ \rho_2 \sim \mathcal{U}(5.851 \cdot 10^{-02}, 9.752 \cdot 10^{-02}), \end{array} \right. \quad (6.1)$$

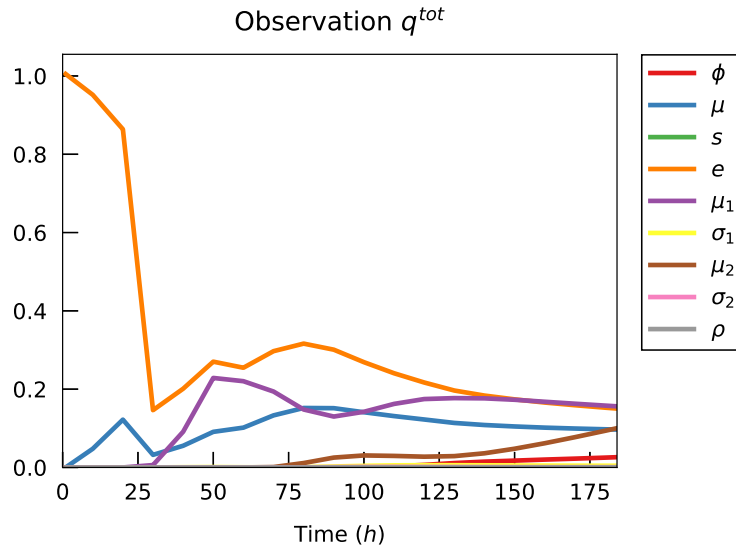


Figure 6.1: Sobol indices for all the model parameters when considering the total leaf area q^{tot} as output variable w.r.t. time.

and 10,000 samples were used to obtain the estimates of Sobol indices. The results are displayed on Figure 6.1.

The most influential parameter all along the growth of the plant is clearly e , the leaf mass per area. Although the radiation use efficiency μ comes second at the very early stages of the simulation, it seems that the log-normal mean for the leaves of the first kind μ_1 is more influential overall. Finally, in the late stages of the growth, the phyllochron ϕ and the log-normal mean μ_2 start to have a minor influence (which is understandable for the latter as the leaves of the second kind had not emerged before). All the other parameters seem to play a very minor role. The 5 most influential parameters by order of importance for the GreenLab model are therefore retained to be $(e, \mu_1, \mu, \mu_2, \phi)$.

However, the joint estimation of parameters e and μ is not doable in practice owing to the absence of experimental data on mass. It remains, however, strictly equivalent to estimate one parameter or the other and we will therefore keep μ fixed and estimate e . A shortcoming worth mentioning of this approach is that it is thus hard to confer a biological value to these parameters. As far as the phyllochron ϕ is concerned, it is seldom estimated in practice as it can be easily deduced from the days of appearance of the different leaves by averaging the differences between two such events. These reasons led us to consider the joint estimation of 5 parameters for the GreenLab model: $\theta = (e, \mu_1, \sigma_1, \mu_2, \sigma_2)$.

6.1.2 Data simulation

We chose to simulate data for a single individual using a set of parameters that has been calibrated using real data. This offers the advantage to provide simulated data that is consistent with the kind of data with which

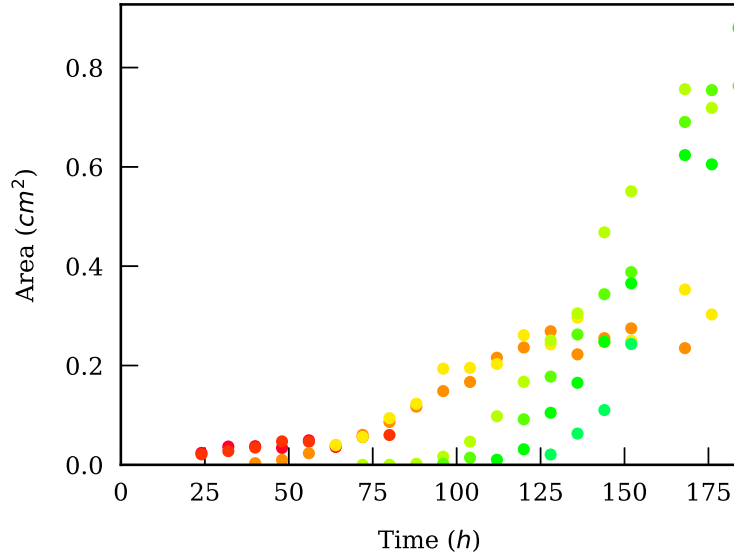


Figure 6.2: Data for leaf area simulated using a set of parameters previously calibrated on real data. The area of each leaf is not observed at all times but in a realistic manner as suggested by the results obtained via image analysis (see Chapter 7).

we are likely to be confronted in practice. The set of parameters in question was:

$$\left\{ \begin{array}{l} \phi^{\text{true}} = 1.200 \cdot 10^{+01}, \\ \mu^{\text{true}} = 3.150 \cdot 10^{+00}, \\ s^{\text{true}} = 5.000 \cdot 10^{+00}, \\ e^{\text{true}} = 1.695 \cdot 10^{-03}, \\ k^{\text{true}} = 7.000 \cdot 10^{-01}, \\ \mu_1^{\text{true}} = 4.408 \cdot 10^{+00}, \\ \sigma_1^{\text{true}} = 4.593 \cdot 10^{-01}, \\ \mu_2^{\text{true}} = 5.351 \cdot 10^{+00}, \\ \sigma_2^{\text{true}} = 3.992 \cdot 10^{-01}, \\ \rho_2^{\text{true}} = 7.802 \cdot 10^{-02}, \\ q_0^{\text{true}} = 3.808 \cdot 10^{-05}. \end{array} \right. \quad (6.2)$$

Identically, the observation function was taken to be that of a real case scenario: the areas of the different leaves are not observed at all time steps. We recall that the GreenLab model is simulated by considering a photoperiod of 8h per day. In the practical cases described in Chapter 7, data is only observed once per day from day 3, i.e. every 8 n , for $n \in \llbracket 3, 23 \rrbracket$. We also chose a multiplicative normal observation noise with standard deviation $\sigma = 0.1$. The simulated data can be observed on Figure 6.2. The areas of the first two leaves are observed quite well during the first eight days and then vanish as these leaves become entirely covered by younger ones. We denote by $\tau^v = (t_1^v, \dots, t_{n^v}^v)$ the timeline for the v -th leaf, $\tilde{\alpha}^v = (\tilde{a}_1^v, \dots, \tilde{a}_{n^v}^v)$ the corresponding observations on leaf area at these time steps. The concatenated experimental data vector is denoted y , and the vector of the corresponding hidden states obtained via model simulation from a set of parameters θ is denoted $x(\theta)$.

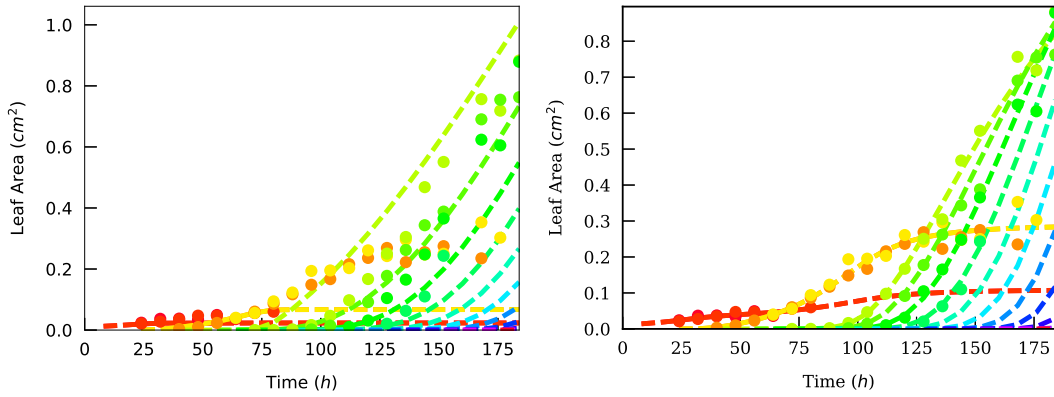


Figure 6.3: Left: hidden states (dashed lines) for the different leaf areas obtained with the initial value of the parameters' set vs. simulated observations (filled circles). Right: hidden states (dashed lines) for the different leaf areas obtained with the GLS estimates for the parameters vs. simulated observations (filled circles).

6.1.3 Parameter estimation

These simulated data were first used in a GLS algorithm for the estimation of the five parameters retained $\theta = (e, \mu_1, \sigma_1, \mu_2, \sigma_2)$. A standard Gauss–Newton algorithm was used for the minimization procedure with a maximum of 1,000 iterations. The initial values chosen for the different parameters were:

$$\begin{cases} e &= 1.5 e^{\text{true}}, \\ \mu_1 &= 0.7 \mu_1^{\text{true}}, \\ \sigma_1 &= 0.8 \sigma_1^{\text{true}}, \\ \mu_2 &= 1.1 \mu_2^{\text{true}}, \\ \sigma_2 &= 1.3 \sigma_2^{\text{true}}, \end{cases} \quad (6.3)$$

and the initial covariance matrix was taken to be heteroskedastic with a covariance relative to each leaf:

$$\Sigma^{(0)} = \text{blockdiag}_{\mathbf{g}_v \in \llbracket 1, \nu_{\max} \rrbracket} \{ \mathbb{V}(\tilde{\alpha}^v) I_{n^v} \}. \quad (6.4)$$

A first set of parameters $\hat{\theta}^{(0)}$ was obtained by minimizing the GLS criterion and the standard deviation of the observation noise is deduced as:

$$\hat{\sigma} = \mathbb{V} \left(\left[\begin{array}{c} y_i - x(\hat{\theta}^{(0)})_i \\ x(\hat{\theta}^{(0)})_i \end{array} \right]_{i \in \llbracket 1, d_y \rrbracket} \right)^{\frac{1}{2}}. \quad (6.5)$$

A new covariance matrix was defined as follows:

$$\Sigma^{\text{GLS}} = \hat{\sigma}^2 \text{diag} \left([y_i^2]_{i \in \llbracket 1, d_y \rrbracket} \right). \quad (6.6)$$

and the final estimate for the parameters $\hat{\theta}^{\text{GLS}}$ was inferred from the minimization of the GLS criterion using this new covariance matrix. In practice, the initial set of parameters chosen for an estimation routine is such that it gives realistic growth curves. In this case, the initial set of parameters provided growth curves for the different leaves very far from the actual data as can be seen from Figure 6.3, thereby highlighting the fact that such initial parameters can be considered far enough from the true ones.

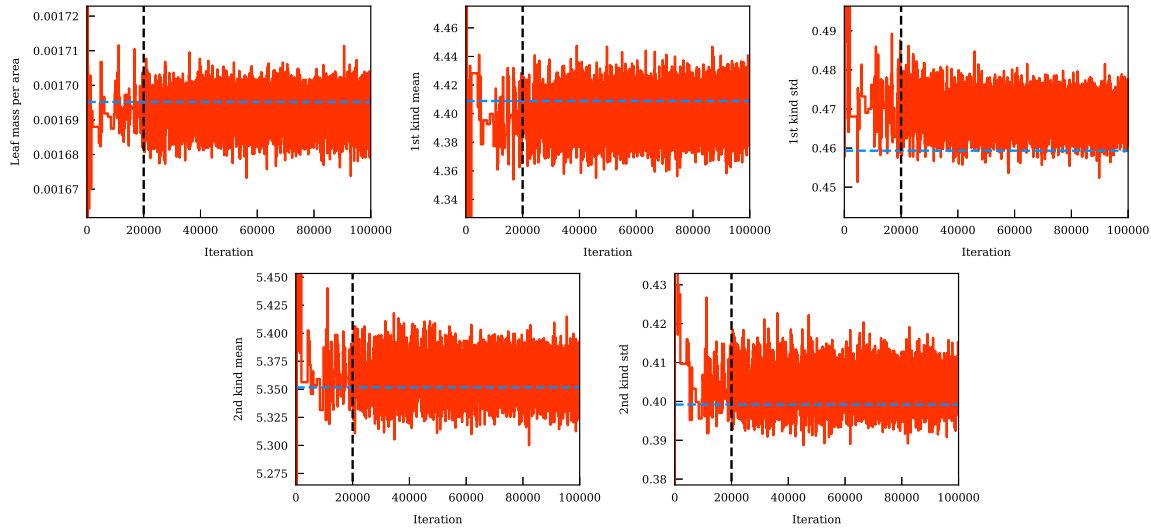


Figure 6.4: Markov chains generated by the AMWG algorithm for the estimated parameters $\theta = (e, \mu_1, \sigma_1, \mu_2, \sigma_2)$. The true values of the parameters are indicated by dashed blue lines and the dashed black lines indicate the end of the burn-in period.

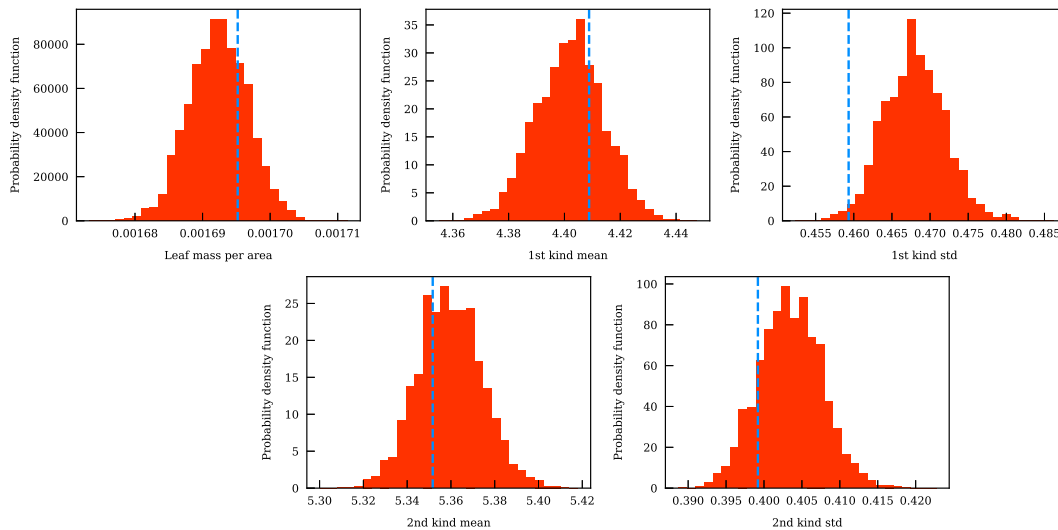


Figure 6.5: Histograms of the posterior distributions generated by the AMWG algorithm for the estimated parameters $\theta = (e, \mu_1, \sigma_1, \mu_2, \sigma_2)$ (burn-in period discarded).

For the AMWG algorithm, we chose normal prior distributions centered on the same values as the initial values for the GLS algorithm and a standard deviation equal to two fifths of the mean. The total number of MCMC iterations was $M = 100,000$ and the burn-in period was of $B = 20,000$ iterations. The Markov chains for each individual parameter are displayed on Figure 6.4 and the corresponding histograms on Figure 6.5, for which the samples of the burn-in period were discarded. The means and standard deviations for all parameters were then computed from these posterior distributions.

From Figure 6.4, it appears that the burn-in period could not have been lower than 20,000 iterations as the chains had not converged to the stationary distributions before. Despite initial values sampled from the prior distributions not very close to the truth, the chains still rapidly converge towards the true value of the parameters. From Figure 6.5, the posterior distributions are almost all centered on the true values, except for

	e	μ_1	σ_1	μ_2	σ_2
θ^{true}	$1.695 \cdot 10^{-03}$	$4.408 \cdot 10^{+00}$	$4.593 \cdot 10^{-01}$	$5.351 \cdot 10^{+00}$	$3.992 \cdot 10^{-01}$
$\hat{\theta}^{\text{GLS}}$	$1.694 \cdot 10^{-03}$	$4.405 \cdot 10^{+00}$	$4.650 \cdot 10^{-01}$	$5.357 \cdot 10^{+00}$	$3.997 \cdot 10^{-01}$
$\hat{\theta}^{\text{AMWG}}$	$1.692 \cdot 10^{-03}$	$4.402 \cdot 10^{+00}$	$4.680 \cdot 10^{-01}$	$5.359 \cdot 10^{+00}$	$4.035 \cdot 10^{-01}$
$\hat{\sigma}^{\text{GLS}}$	$5.900 \cdot 10^{-04}$	$6.806 \cdot 10^{-04}$	$1.241 \cdot 10^{-02}$	$1.121 \cdot 10^{-03}$	$1.253 \cdot 10^{-03}$
$\hat{\sigma}^{\text{AMWG}}$	$1.693 \cdot 10^{-03}$	$1.536 \cdot 10^{-03}$	$1.882 \cdot 10^{-02}$	$1.457 \cdot 10^{-03}$	$1.068 \cdot 10^{-02}$
$\hat{\sigma}^{\text{GLS}}$	$3.309 \cdot 10^{-05}$	$1.538 \cdot 10^{-01}$	$9.097 \cdot 10^{-02}$	$2.560 \cdot 10^{-01}$	$1.827 \cdot 10^{-01}$
$\hat{\sigma}^{\text{AMWG}}$	$4.363 \cdot 10^{-06}$	$1.215 \cdot 10^{-02}$	$3.985 \cdot 10^{-03}$	$1.470 \cdot 10^{-02}$	$4.145 \cdot 10^{-03}$
$\hat{\sigma}^{\text{GLS}}/\theta^{\text{true}}$	$1.952 \cdot 10^{-02}$	$3.489 \cdot 10^{-02}$	$1.981 \cdot 10^{-01}$	$4.784 \cdot 10^{-02}$	$4.577 \cdot 10^{-01}$
$\hat{\sigma}^{\text{AMWG}}/\theta^{\text{true}}$	$2.574 \cdot 10^{-03}$	$2.756 \cdot 10^{-03}$	$8.676 \cdot 10^{-03}$	$2.747 \cdot 10^{-03}$	$1.038 \cdot 10^{-02}$

Table 6.1: Comparison of the results provided by the GLS and AMWG algorithms for the estimation of 5 parameters of the GreenLab model for *Arabidopsis thaliana*.

that of σ_1 which does not capture extremely well the true value.

The results of the two estimation algorithms are summarized in Tables 6.1 and 6.2. The true values used for data simulation are denoted by θ^{true} . For GLS, $\hat{\theta}^{\text{GLS}}$ denotes the punctual estimate provided by the algorithm and $\hat{\sigma}^{\text{GLS}}$ the corresponding standard deviation (i.e. the square root of the diagonal components of the covariance matrix, see Equation 3.26). For AMWG, even though the results for the parameters are posterior distributions, we still denote by $\hat{\theta}^{\text{AMWG}}$ and $\hat{\sigma}^{\text{AMWG}}$ their means and standard deviations respectively. Finally, the relative error on the mean is $\delta^A = |\hat{\theta}^A - \theta^{\text{true}}|/|\theta^{\text{true}}|$ with $A \in \{\text{GLS}, \text{AMWG}\}$.

The first thing to note is that both GLS and AMWG yield excellent estimates for the parameters. The relative errors on each of the latter never exceeds $2 \cdot 10^{-02}$. The GLS estimates are always better than those provided by AMWG, sometimes by a very short margin, sometimes by a factor of 10. As far as the standard deviations given by the two estimates are concerned, AMWG always gives lower values by at least a factor of 10, sometimes of around 40 (for σ_2). In spite of better punctual estimates in the present case for GLS, the associated uncertainty is much higher than in the case of AMWG. The different standard deviations are also normalized in the last two lines of Table 6.1, which highlights that GLS provides particularly high variances for the two parameters σ_1 and σ_2 . These two parameters are also the ones that are associated with higher variances for AMWG, though with a lesser effect than in the case of GLS.

We also computed the RMSEP and EF coefficients for each leaf in the case of the two algorithms. The results are summarized in Table 6.2. The RMSEP coefficients obtained with GLS are lower for all leaves but the last one. The modelling efficiency are also all very close to the optimal value 1, which emphasizes the excellent fitting capabilities of both algorithms.

As a conclusion, we compared a frequentist and a Bayesian estimation method in the case of the GreenLab model for *Arabidopsis thaliana* and it appears that if the punctual estimates yielded by the GLS are closer to the true values for model parameters than the means of the posterior distributions obtained with AMWG, the

		1	2	3	4	5	6	7	8
RMSEP	GLS	$2.108 \cdot 10^{-08}$	$5.924 \cdot 10^{-08}$	$7.978 \cdot 10^{-05}$	$1.009 \cdot 10^{-04}$	$7.044 \cdot 10^{-05}$	$4.546 \cdot 10^{-05}$	$2.021 \cdot 10^{-05}$	$4.469 \cdot 10^{-06}$
	AMWG	$2.748 \cdot 10^{-08}$	$1.886 \cdot 10^{-07}$	$1.517 \cdot 10^{-04}$	$1.918 \cdot 10^{-04}$	$1.481 \cdot 10^{-04}$	$1.099 \cdot 10^{-04}$	$6.635 \cdot 10^{-05}$	$2.794 \cdot 10^{-06}$
EF	GLS	$9.999 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$	$9.995 \cdot 10^{-01}$	$9.989 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$	$1.000 \cdot 10^{+00}$	$9.998 \cdot 10^{-01}$
	AMWG	$9.999 \cdot 10^{-01}$	$9.998 \cdot 10^{-01}$	$9.990 \cdot 10^{-01}$	$9.979 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$	$9.999 \cdot 10^{-01}$

Table 6.2: RMSEP and EF coefficients calculated separately for each leaf in the case of the GLS and AMWG algorithm.

errors on the parameters provided by the variance covariance matrix are significantly higher than the standard deviations of the posterior distributions.

6.2 RPF-based data assimilation for the LNAS wheat model

Data assimilation is one of the key applications in agriculture. Let us assume that a model has already been calibrated using a given experimental data set in some specific conditions – this can include the year during which plants were grown, the location of the field the genotype used. Such experimental data provided an estimate of the model parameters. If another experimental data set is available for the same plant but in slightly different conditions (different year, location or genotype), estimates of the model parameters might not be optimal for the latter. Data assimilation therefore consists in using a few early observations in these new conditions in order to enhance the parameter and state estimates and provide more accurate predictions for state variables of interest, yield for instance [Launay and Guerif, 2005]. This is typically achieved by using online filtering procedures (such as the UKF, EnKF or RPF algorithms introduced in Section 3.5.2). The parameter estimates previously obtained in other conditions still remain relevant and are used as prior information in these SMC methods. In this regard, the Bayesian approach appears rather convenient as it allows to incorporate previously acquired knowledge. Indeed, since only a few observations are available in such a scenario, it is crucial that the particles be given initial values that are not too far from the optimal ones. In a first step, all the particles are first propagated and corrected according to the early observations. In a second step when no observations on the system are available anymore, they are propagated through the model to forecast the values of state variables of interest and the related uncertainty at future times.

The efficiency of data assimilation using a convolution particle filter for yield forecast has been described by Chen and Cournède [2014] with the LNAS model for sugar beet and the STICS model for winter wheat. In this section, we investigate the use of the RPF algorithm to the newly designed LNAS model for wheat for data assimilation. This model is indeed aimed in the future at providing yield predictions in scenarios where relatively few data is available. The main objective of this case study is to compare the model predictions with and without assimilation of data available from the early stages of plant growth.

Since real data was not available yet for this investigation, we generated simulated data from the model. This also conveniently allows us to compare the predictions of the model to the true hidden states. In practice, and this is one of the main difficulties with applications to wheat, very few measurements are available. We assumed that observations were available for 4 state variables that are:

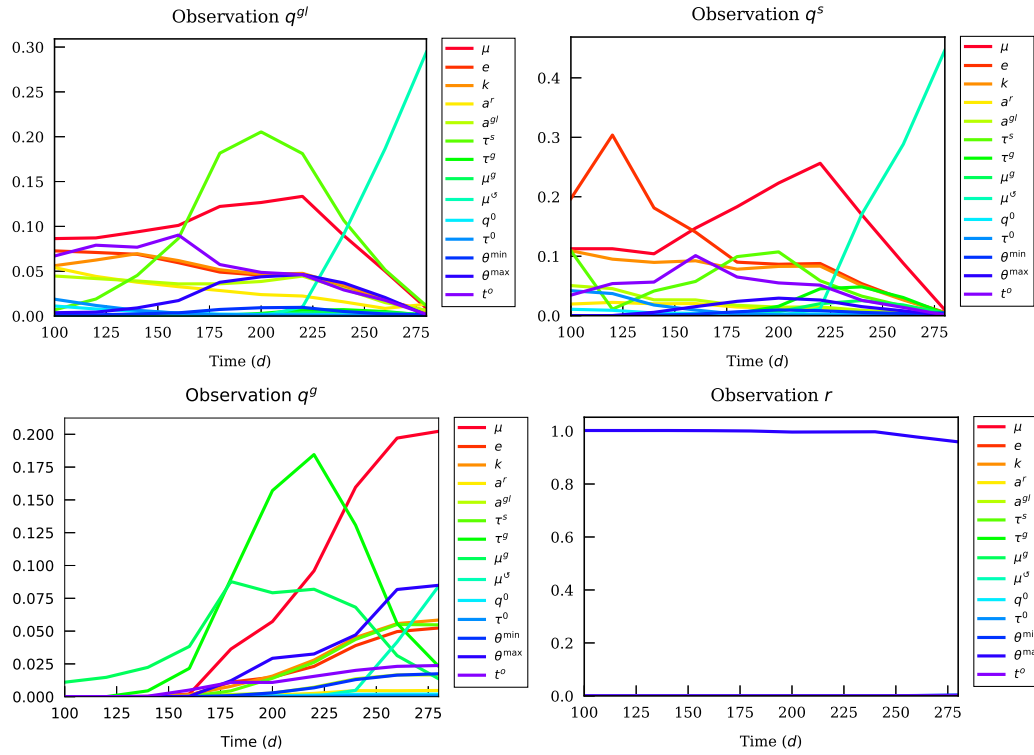


Figure 6.6: Sobol indices for the state variables q^{gl} , q^s , q^g , and r from $n = 100$ to $n = 280$.

- the biomass of green leaves q^{gl} ,
- the biomass of the stem q^s ,
- the biomass of the grain q^g ,
- the water reserve r .

6.2.1 Sensitivity analysis

As always for such estimation problems, a first step consists in performing a sensitivity analysis method to select the most influential parameters and reduce the dimension of the problem. We chose to compute Sobol indices for these 4 variables, as they are the ones used for model calibration. The results are illustrated on Figure 6.6.

Only 14 parameters were displayed on the different figures, for both the sake of readability and as all others play a role even less significant in the variances of these outputs. For the biomass of green leaves, the two most influential parameters by far are τ^g and μ , though at the very end of the plant's growth and as the remobilization process starts, μ° begins to play an even more significant role. Parameter μ remains the most influential in the early stages of the growth for the biomass of the stem and again μ° Sobol first order index takes off as soon as the remobilization process occurs. As far as the biomass of the grain is concerned, the two parameters of the related allocation distribution τ^g and μ^g clearly dominates the early stages while μ again takes in an important part in the late stages. Finally, the water reserve variance is vastly ruled by θ^{\max} .

Because the experimental data in the case of data assimilation concerns early stages of the growth and since there is hardly any water stress during the data assimilation period (so that the hydric reserve is close to saturation), we decided to estimate the three most influential parameters over this period, and retained $\theta = (\mu, e, \mu^g)$.

6.2.2 Data simulation

Experimental data was simulated from a set of parameters providing realistic data for biomasses. The graphs for the different variables observed can be seen on Figure 6.7. We actually simulated 3 different data sets with different observation noises: their standard deviation σ were respectively set for all the different variables to 0.1, 0.3 and 0.5. From day $n = 0$ to day $n = 100$, the biomasses of the different compartments remain very low and we considered that:

- the biomass of green leaves q^{gl} and the water reserve r were available from $n = 120$,
- the biomass of the grain q^g and that of the stem q^s were available from $n = 180$.

The final time for possible observations was set to $n = 200$, at which point prediction of the different state variables of interest begins up until $n = 280$ (as we used real environmental data for the year 2012 during which the crop cycle was of 280 days). The ideal case is to have at our disposal observations at every time steps, which is almost never the case in practice as far as wheat is concerned. Still, it remains interesting to see how data assimilation using the RPF can improve the predictions with a large number of observations.

It is assumed that the parameters have been previously estimated using another experimental data set, and the values of the parameters $(\theta_i^{\text{true}})_{i \in \llbracket 1, d \rrbracket}$ used for the simulation of the new data set for data assimilation were shifted by 5%. A standard deviation of 5% was also used. This means that the prior distributions for

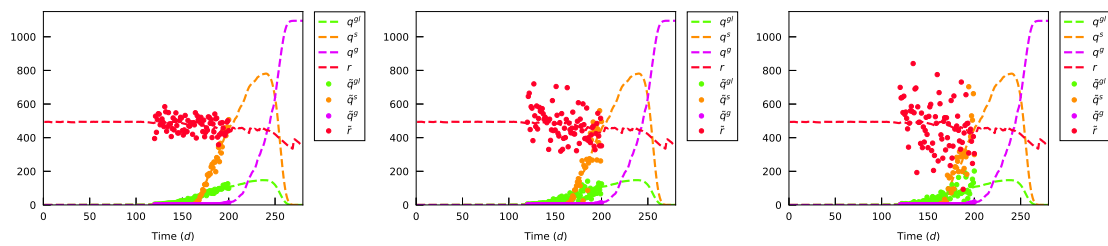


Figure 6.7: Three different data sets have been simulated from the same set of parameters θ^{true} but with different observation noises: with $\sigma = 0.1$ (left), $\sigma = 0.3$ (center) and $\sigma = 0.5$ (right). Observations were generated for all times between $n = 120$ and $n = 200$, but the experimental data sets actually used in the following will most of the time be filtered so that observations only occur every $\Delta n \in \{1, 10, 20\}$ days. The dispersion of the observations (filled circles) around the hidden state (dashed line) clearly increases with the observation noise.

the RPF were taken as:

$$\begin{cases} \mu & \sim \mathcal{N}\left(0.95 \mu^{\text{true}}, (0.05 \mu^{\text{true}})^2\right), \\ e & \sim \mathcal{N}\left(0.95 e^{\text{true}}, (0.05 e^{\text{true}})^2\right), \\ \mu^g & \sim \mathcal{N}\left(0.95 \mu^{g,\text{true}}, (0.05 \mu^{g,\text{true}})^2\right). \end{cases} \quad (6.7)$$

6.2.3 Uncertainty analysis and data assimilation

Uncertainty analysis (UA) is first performed to evaluate the predictions that would have been obtained without data assimilation. $N = 10,000$ particles are sampled from the prior distribution and propagated through the model without taking into account any experimental data set. At each time, these particles represent an ensemble of hidden states from which can be computed the mean and a 95% credible interval obtained by calculating the corresponding quantiles in the population of hidden states. These results will then be compared to those obtained with data assimilation.

Data assimilation (DA) using the RPF algorithm is then performed. Again, $N = 10,000$ particles are sampled from the prior distribution. They are then propagated through the model but this time are corrected whenever an observation is available. Notice, as mentioned previously, that the observations need not be at the same time for the different state variables which allows for more freedom in applications: this is the case here since observations on the biomass of the grain and that of the stem arrive later than for the biomass of green leaves or the water reserve. As in the case of uncertainty analysis, the particles provide at each time an estimate of the mean and a 95% credible interval.

To assess the performance of the estimates, we will look at different values. For the parameters, we simply calculate the relative error δ^{θ_i} between the parameters estimated with the RPF $\hat{\theta}_i$ compared to the true value θ_i^{true} . This is also applied to the hidden states for which we compute similar values at the times of interest for the different variables $\delta_{240}^{q^{\ell}}$, $\delta_{240}^{q^s}$ and $\delta_{280}^{q^g}$. From Figure 6.7, we indeed decided to take an interest in the values of the different biomasses at the peak of their growth curves, which are $n = 240$ for the stem and green leaves and $n = 280$ for the grain. To also assess the dispersion of the population for the hidden state x at time step n with respect to the corresponding true hidden state, we also calculate two values:

$$\begin{cases} \rho_n^x & = \left| \frac{x_n^{95\%} - \hat{x}_n}{\hat{x}_n} \right|, \\ \alpha_n^x & = \left| \frac{x_n^{\text{true}} - \hat{x}_n}{x_n^{95\%} - \hat{x}_n} \right|, \end{cases} \quad (6.8)$$

where x_n^{true} is the true value of the hidden state, \hat{x}_n is its estimate by the RPF (the mean of the population of particles), and $x_n^{95\%}$ is the boundary of the 95% credible interval closer to the true value. The first one of these criteria ρ_n^x is an indicator of how spread the population of hidden states' estimates is, and the second one α_n^x assesses the position of the true value with respect to the 95% credible interval: in particular, $\alpha_n^x > 1$ means that the true value is outside of the credible interval. Ideally, both ρ_n^x and α_n^x should be as low as possible.

6.2.4 Influence of the number of observations

The first case that we consider is that of data generated using a low observation noise (with standard deviation $\sigma = 0.1$). We used three experimental data sets: the first one comprising the observations at each time between $n = 120$ and $n = 200$ for $q^{g\ell}$ and r and $n = 180$ and $n = 200$ for q^g and q^s , the second one with observations every 10 days, and the last one with observations every 20 days. We denote by Δn the difference of time steps between two consecutive observations. The first scenario is of rather low interest in practice, and even the second one seems extremely optimistic as data is more likely to be collected as in the case of the last scenario. However, with the increasing availability of observations obtained from satellite images or captors in fields, a better quality of data to be assimilated can be expected in a near future.

The results for the estimated parameters are displayed on Figure 6.8 for the three scenarios and the three parameters. In every case, the final value is much closer to the truth than the initial one. In the case of $\Delta n = 1$, the parameters rapidly converge towards the region of the true parameters and the noised observations then makes it oscillate. For $\Delta n = 10$ and $\Delta n = 20$, the parameters still manage to converge towards a good estimate of the true values despite very few observations. Results from Table 6.3 show that the relative errors on the parameters with data assimilation have been decreased in all cases by at least one order of magnitude.

The graphs from Figure 6.9 provide a graphical representation of how well UA (in blue) and DA (in red) respectively perform as far as biomass predictions are concerned. The first thing to note is that even though the parameters were slightly modified (by only 5%), uncertainty analysis performs rather poorly and underestimates all biomasses significantly. When data assimilation with the RPF is used, predictions become much better not only during the observation phase but also until the end of the growth. This was expected with many observations as in the case of $\Delta n = 1$, but it works surprisingly well even in the case of $\Delta n = 10$ and $\Delta n = 20$.

For $\Delta n = 1$, the hidden state is very well estimated and the whole population of particles is near the true value. This is best seen by looking at the values in Table 6.3: the relative errors δ for the states are never higher than $2 \cdot 10^{-02}$ and are less than between 10 to 30 – depending on the variable – compared to the case of UA. The dispersion of the population is also much less than in the case of UA as can be seen from the values of the coefficients ρ which are roughly 10 times less in the case of DA. Even though the populations are very confined, the true hidden states are well captured inside the 95% credible interval as the coefficients α are less than 1. In the cases of $\Delta n = 10$ and $\Delta n = 20$, the estimates are still very accurate as the coefficients δ are never higher than $1.5 \cdot 10^{-02}$. The populations are more dispersed – because of the fewer observations – than in the case $\Delta n = 1$ as the coefficients ρ can take on higher values (up to $1.2 \cdot 10^{-01}$). Two of the coefficients δ are lower than in the case of $\Delta n = 1$ which, combined to more dispersed estimates, lead to very low α coefficients.

Overall, hidden states estimates with data assimilation are excellent compared to those provided by uncertainty analysis, even in the case of very few observations. This is extremely encouraging, in particular for the last case as the LNAS model for wheat is likely to be used in the context of scarce data, where $\Delta n = 20$ represents a reasonable frequency for the observations. This amounts to using only 14 observation data in

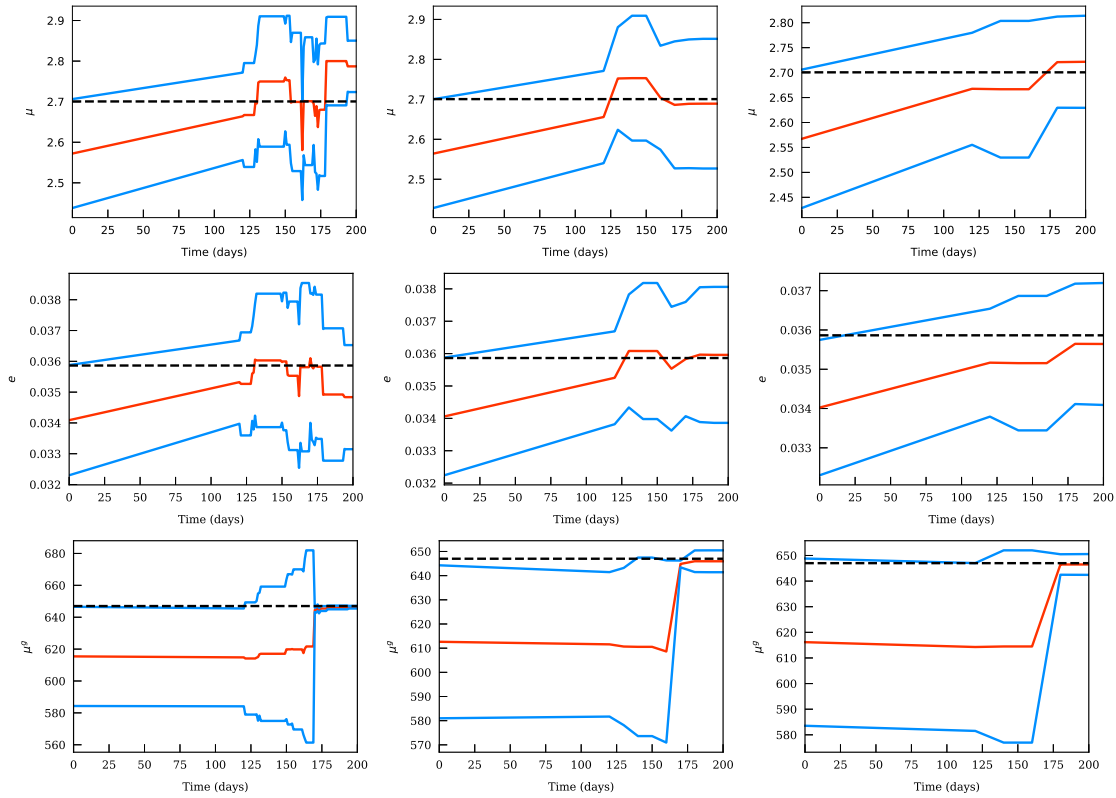


Figure 6.8: Evolution of the different estimated parameters as they are corrected when observations are available: with $\Delta n = 1$ (left), $\Delta n = 10$ (middle) and $\Delta n = 20$ (right). If the population of particles for parameter θ_i has mean m and standard deviation s , the red curve represents m and the blue lines $m \pm s$. The true value used for data simulation is indicated with a black dashed line. We recall that data assimilation starts on day 120.

order to calibrate the model and improving the predictions. As this is the case of highest interest to us, we will consider that $\Delta n = 20$ in the following.

6.2.5 Influence of the observation noise

In the previous experiments, the observation noise was rather low, such that its standard deviation was $\sigma = 0.1$. In real applications, one might obtain data that is even more noisy, and the study of the influence of the observation noise turns out to be crucial. In order to assess the performance of data assimilation with the RPF in such scenarios, we considered three different cases where the standard deviation of the observation noise takes values:

$$\begin{cases} \sigma = 0.1, \\ \sigma = 0.3, \\ \sigma = 0.5. \end{cases} \quad (6.9)$$

The prior distributions and the number of particles remain the same as before and, as previously mentioned, $\Delta n = 20$. The results are displayed on Figure 6.10. From these graphs, one could assume that for $\sigma = 0.5$ the observations are still very close to the curve, but two things must be accounted for: first, when the values of

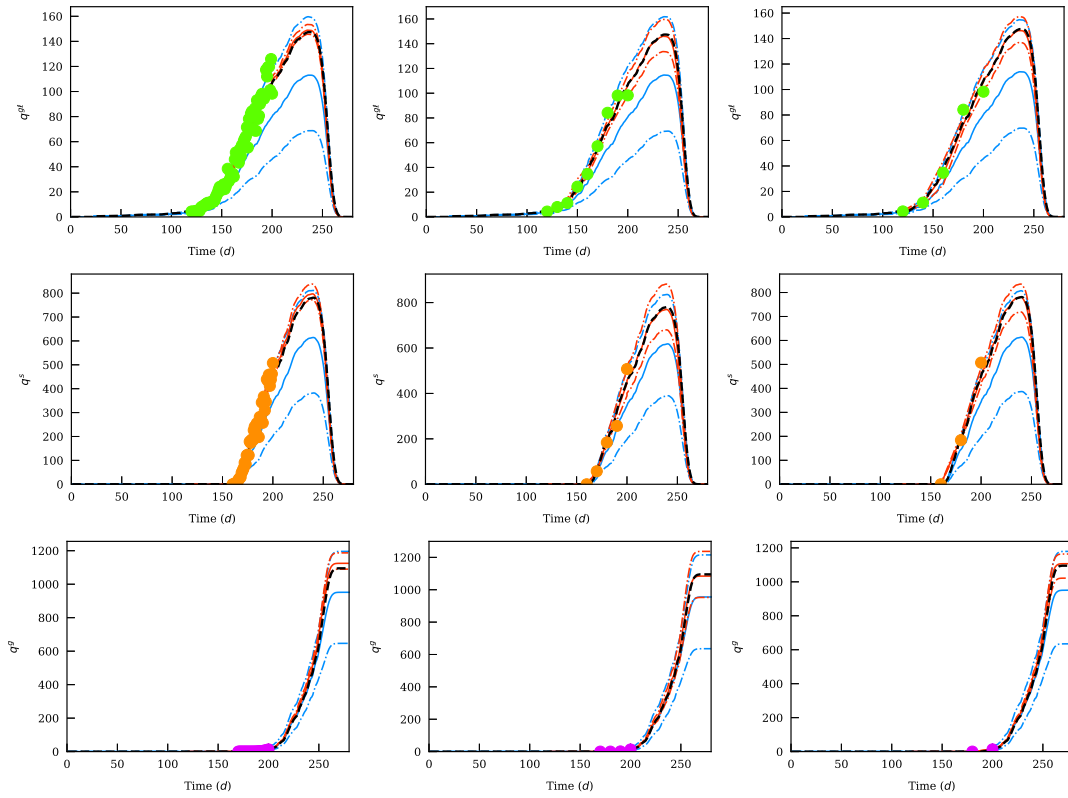


Figure 6.9: Predictions of the different biomasses in the case of uncertainty analysis (UA, blue lines) and data assimilation (DA, red lines). The 95% CI boundaries are indicated with dashed lines. The black lines correspond to the true hidden states and the observations used for data assimilation are represented by filled circles. The three cases considered here are $\Delta n = 1$ (left), $\Delta n = 10$ (center) and $\Delta n = 20$ (right).

	$\Delta n = 1$		$\Delta n = 10$		$\Delta n = 20$	
	UA	DA	UA	DA	UA	DA
δ^μ	$5.000 \cdot 10^{-02}$	$3.202 \cdot 10^{-02}$	$5.000 \cdot 10^{-02}$	$4.222 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$7.823 \cdot 10^{-03}$
δ^e	$5.000 \cdot 10^{-02}$	$2.863 \cdot 10^{-02}$	$5.000 \cdot 10^{-02}$	$2.701 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$6.147 \cdot 10^{-03}$
δ^{μ^g}	$5.000 \cdot 10^{-02}$	$1.100 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$1.632 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$7.847 \cdot 10^{-04}$
$\delta_{240}^{q^{g\ell}}$	$2.332 \cdot 10^{-01}$	$7.278 \cdot 10^{-03}$	$2.224 \cdot 10^{-01}$	$1.340 \cdot 10^{-02}$	$2.272 \cdot 10^{-01}$	$1.082 \cdot 10^{-02}$
$\rho_{240}^{q^{g\ell}}$	$3.905 \cdot 10^{-01}$	$2.143 \cdot 10^{-02}$	$3.939 \cdot 10^{-01}$	$8.509 \cdot 10^{-02}$	$3.873 \cdot 10^{-01}$	$6.459 \cdot 10^{-02}$
$\alpha_{240}^{q^{g\ell}}$	$7.479 \cdot 10^{-01}$	$3.372 \cdot 10^{-01}$	$6.958 \cdot 10^{-01}$	$1.427 \cdot 10^{-01}$	$8.245 \cdot 10^{-01}$	$1.481 \cdot 10^{-01}$
$\delta_{240}^{q^s}$	$2.135 \cdot 10^{-01}$	$1.965 \cdot 10^{-02}$	$2.086 \cdot 10^{-01}$	$1.522 \cdot 10^{-02}$	$2.142 \cdot 10^{-01}$	$3.150 \cdot 10^{-04}$
$\rho_{240}^{q^s}$	$3.782 \cdot 10^{-01}$	$3.090 \cdot 10^{-02}$	$3.697 \cdot 10^{-01}$	$1.167 \cdot 10^{-01}$	$3.697 \cdot 10^{-01}$	$7.958 \cdot 10^{-02}$
$\alpha_{240}^{q^s}$	$8.464 \cdot 10^{-01}$	$6.238 \cdot 10^{-01}$	$7.492 \cdot 10^{-01}$	$1.063 \cdot 10^{-01}$	$8.644 \cdot 10^{-01}$	$4.571 \cdot 10^{-03}$
$\delta_{280}^{q^g}$	$1.309 \cdot 10^{-01}$	$2.716 \cdot 10^{-02}$	$1.277 \cdot 10^{-01}$	$9.887 \cdot 10^{-03}$	$1.318 \cdot 10^{-01}$	$1.063 \cdot 10^{-02}$
$\rho_{280}^{q^g}$	$3.206 \cdot 10^{-01}$	$3.088 \cdot 10^{-02}$	$3.345 \cdot 10^{-01}$	$1.216 \cdot 10^{-01}$	$3.323 \cdot 10^{-01}$	$7.680 \cdot 10^{-02}$
$\alpha_{280}^{q^g}$	$5.864 \cdot 10^{-01}$	$8.563 \cdot 10^{-01}$	$5.379 \cdot 10^{-01}$	$7.090 \cdot 10^{-02}$	$6.312 \cdot 10^{-01}$	$2.035 \cdot 10^{-01}$

Table 6.3: Results for the different relative errors and other coefficients calculated in order to identify the quality of the estimates provided by uncertainty analysis (UA) and data assimilation (DA) with different frequencies of observations $\Delta n = 1$, $\Delta n = 10$ and $\Delta n = 20$.

the observations are small, the multiplicative error might still be important and second, what is not displayed here is the observations for the water reserve that are particularly noisy.

The case with $\sigma = 0.1$ corresponds to one previously discussed and is there only for comparison. For $\sigma = 0.3$, the relative errors on the parameters for DA are two times lower than the initial ones for μ and e , which admittedly constitutes an improvement, although a subtle one. Only the relative error on μ^g is greatly improved, by a factor of 20. This effect becomes even more pronounced when $\sigma = 0.5$, with a relative error for μ higher than in the case of uncertainty analysis, and the one for e is slightly decreased. Once again, the only parameters whose estimation is significantly improved is μ^g . Even though this is not illustrated here, we can say that the improvements for this parameter occur with the last two sets of observations being assimilated, highlighting a possible importance of the observations on the biomass of the grain q^g .

As far as hidden states estimates are concerned, DA predictions are still acceptable as the true states remain within the 95% credible interval for $\sigma = 0.3$ (by a narrow margin for $q^{g\ell}$). This is best seen by the values indicated in Table 6.4 where the α coefficients are less than $6 \cdot 10^{-01}$ for q^s and q^g and $8 \cdot 10^{-01}$ for $q^{g\ell}$. The dispersion of the particles in this case is still moderate, which is not the case anymore for $\sigma = 0.5$. Indeed, the ρ coefficients are not higher than $2 \cdot 10^{-02}$ which is rather low. The concentration of the particles is also obvious from Figure 6.10. What likely happens is that since the observations are very noisy, the first observation to strongly deviate from the hidden state curve will ensure that the weight is carried by very few particles and the population of particles, even though perturbed for a better state space exploration, will get stuck in a region of the state space that is not optimal. This, in turn, implies that the true hidden states is not well captured by the 95% credible interval as demonstrated by the large values for the α coefficients (up to more than 4 for $q^{g\ell}$).

Despite these issues in the case of large observation errors, data assimilation provides good state estimates and appears as an excellent and much more robust tool for model prediction than uncertainty analysis with, it should be remembered, only 14 observations over the whole plant growth and with potentially very high observation noise.

6.2.6 Influence of the prior

The prior distribution that is used in the RPF algorithm is also of interest. In the present case, it represents the knowledge that we have acquired by calibrating the model in another situation. If the two situations in which the plants grow highly differ, it might be possible that the optimal values of the parameters differ significantly as well. To study this effect, we chose 3 different prior distributions $\theta_i \sim \mathcal{N}\left((1 + (-1)^i \epsilon) \theta_i^{\text{true}}, (\epsilon \theta_i^{\text{true}})^2\right)$ with:

$$\begin{cases} \epsilon = 0.05, \\ \epsilon = 0.10, \\ \epsilon = 0.20, \end{cases} \quad (6.10)$$

where we recall that θ_i^{true} is the true parameter used for data simulation and that the prior used corresponds to the posterior distribution obtained from the calibration from a previous experiment. The factor $(-1)^i$ ensures

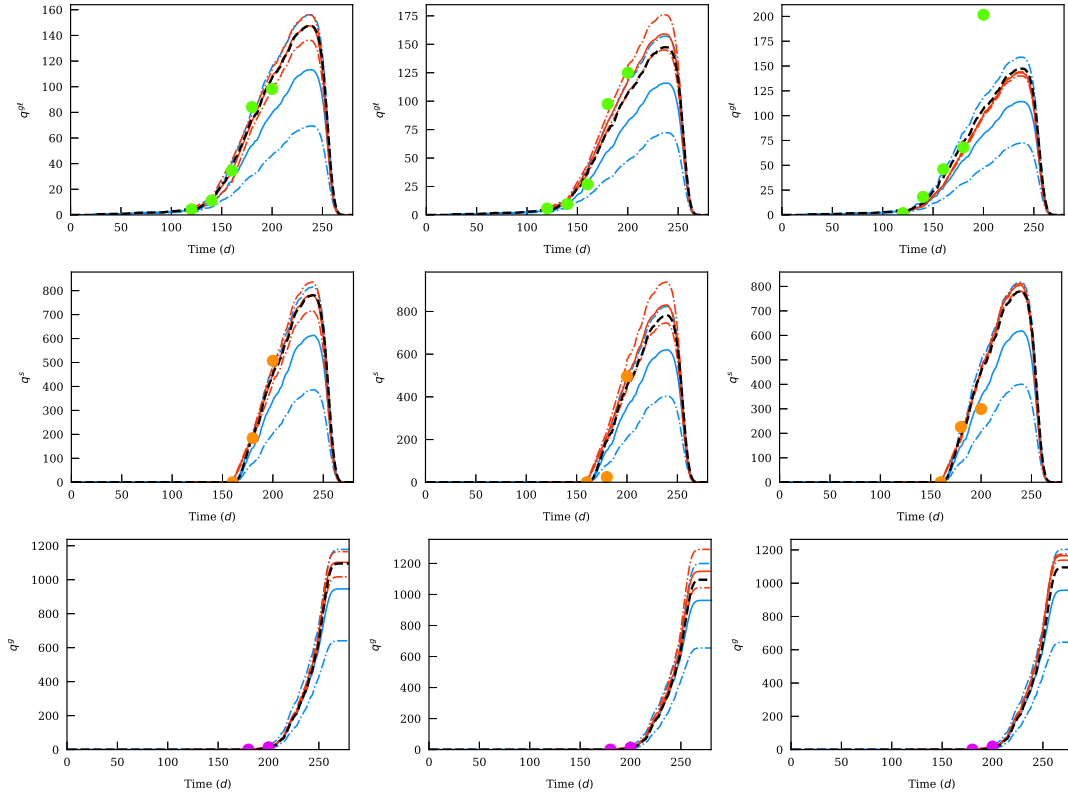


Figure 6.10: Predictions of the different biomasses in the case of uncertainty analysis (UA, blue lines) and data assimilation (DA, red lines). The 95% CI boundaries are indicated with dashed lines. The black lines correspond to the true hidden states and the observations used for data assimilation are represented by filled circles. The three cases considered here are $\sigma = 0.1$ (left), $\sigma = 0.3$ (center) and $\sigma = 0.5$ (right).

	$\sigma = 0.1$		$\sigma = 0.3$		$\sigma = 0.5$	
	UA	DA	UA	DA	UA	DA
δ^μ	$5.000 \cdot 10^{-02}$	$4.605 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$2.162 \cdot 10^{-02}$	$5.000 \cdot 10^{-02}$	$6.132 \cdot 10^{-02}$
δ^e	$5.000 \cdot 10^{-02}$	$8.105 \cdot 10^{-04}$	$5.000 \cdot 10^{-02}$	$2.769 \cdot 10^{-02}$	$5.000 \cdot 10^{-02}$	$4.034 \cdot 10^{-02}$
δ^{μ^g}	$5.000 \cdot 10^{-02}$	$1.109 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$2.183 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$3.437 \cdot 10^{-03}$
$\delta_{240}^{q^g \ell}$	$2.322 \cdot 10^{-01}$	$2.177 \cdot 10^{-03}$	$2.136 \cdot 10^{-01}$	$7.472 \cdot 10^{-02}$	$2.253 \cdot 10^{-01}$	$3.080 \cdot 10^{-02}$
$\rho_{240}^{q^g \ell}$	$3.855 \cdot 10^{-01}$	$7.649 \cdot 10^{-02}$	$3.756 \cdot 10^{-01}$	$8.829 \cdot 10^{-02}$	$3.647 \cdot 10^{-01}$	$2.210 \cdot 10^{-02}$
$\alpha_{240}^{q^g \ell}$	$7.965 \cdot 10^{-01}$	$3.761 \cdot 10^{-02}$	$7.691 \cdot 10^{-01}$	$7.874 \cdot 10^{-01}$	$7.440 \cdot 10^{-01}$	$4.574 \cdot 10^{+00}$
$\delta_{240}^{q^s}$	$2.146 \cdot 10^{-01}$	$2.307 \cdot 10^{-03}$	$2.058 \cdot 10^{-01}$	$6.258 \cdot 10^{-02}$	$2.083 \cdot 10^{-01}$	$3.045 \cdot 10^{-02}$
$\rho_{240}^{q^s}$	$3.705 \cdot 10^{-01}$	$8.609 \cdot 10^{-02}$	$3.494 \cdot 10^{-01}$	$1.014 \cdot 10^{-01}$	$3.521 \cdot 10^{-01}$	$3.022 \cdot 10^{-02}$
$\alpha_{240}^{q^s}$	$8.293 \cdot 10^{-01}$	$3.435 \cdot 10^{-02}$	$7.903 \cdot 10^{-01}$	$5.807 \cdot 10^{-01}$	$8.157 \cdot 10^{-01}$	$9.778 \cdot 10^{-01}$
$\delta_{279}^{q^g}$	$1.362 \cdot 10^{-01}$	$6.623 \cdot 10^{-03}$	$1.217 \cdot 10^{-01}$	$4.925 \cdot 10^{-02}$	$1.255 \cdot 10^{-01}$	$6.382 \cdot 10^{-02}$
$\rho_{279}^{q^g}$	$3.228 \cdot 10^{-01}$	$7.742 \cdot 10^{-02}$	$3.190 \cdot 10^{-01}$	$9.178 \cdot 10^{-02}$	$3.260 \cdot 10^{-01}$	$2.285 \cdot 10^{-02}$
$\alpha_{279}^{q^g}$	$6.412 \cdot 10^{-01}$	$1.155 \cdot 10^{-01}$	$5.597 \cdot 10^{-01}$	$5.114 \cdot 10^{-01}$	$5.596 \cdot 10^{-01}$	$2.625 \cdot 10^{+00}$

Table 6.4: Results for the different relative errors and other coefficients calculated in order to identify the quality of the estimates provided by uncertainty analysis (UA) and data assimilation (DA) with different standard deviations of observation noise $\sigma = 0.1$, $\sigma = 0.3$ and $\sigma = 0.5$.

that parameters are neither all underestimated nor all overestimated. We then performed an RPF algorithm on the scarce data, $\Delta n = 20$, with an observation noise such that $\sigma = 0.1$. The results are displayed on Figure 6.11.

We witness that in all three cases, the mean estimates provided by the RPF perfectly align with the true hidden states. The uncertainty analysis estimates remain reasonable with prior distributions close to the truth, their quality quickly degrades with distancing prior distributions. Even with prior distributions rather close to the truth, $\epsilon = 0.05$, UA significantly underestimates the different biomasses. As ϵ further increases, this becomes even more noticeable: in the case of $\epsilon = 0.10$, the different biomasses estimated with UA are almost half of their real values and when $\epsilon = 0.20$ they are at least 4 times lower than the real values. The values of the parameters estimated with the RPF are also much better at the final observation time as can be seen from Table 6.5.

For $\epsilon = 0.05$, despite the initial closeness of the parameters to the truth, the relative errors are divided by 8, 12 and 40 for the first, second and third parameters respectively. The relative errors on the different state variables of interest are also significantly reduced, by a factor of 40 to 60 depending on the variables. The coefficients α and ρ also both decreased, which indicates that the population of the DA estimates is not too dispersed and that the true values lies at the centre of the credible interval, which can also be seen on Figure 6.11.

What is more of interest concerns the case with $\epsilon = 0.10$ and $\epsilon = 0.20$. Indeed, in spite of further prior distributions, the hidden states' estimates with data assimilation are still excellent, as evidenced by the δ coefficients that are all less than $1.5 \cdot 10^{-02}$ in the case of $\epsilon = 0.10$ and $6 \cdot 10^{-03}$ in the case of $\epsilon = 0.20$, which is much better than in the case of UA. It is worth noting that the DA estimates provided by the case $\epsilon = 0.20$ are even better than those for the case of $\epsilon = 0.10$. The dispersion of the estimates is also reasonable and the true hidden states are always perfectly captured by the 95% credible interval. The worst case being for the very early stages of q^s : as prior distributions are chosen far from the true values, the population needs time to adapt and correct the values of the particles accordingly.

The predictions with data assimilation and ill-chosen priors are excellent in all cases: strikingly, when $\epsilon = 0.20$, the 95% credible intervals of UA and DA do not even overlap, which sums up the strong gain in performance for the predictions. To emphasize this point, the posterior distributions for the different state estimates $q_{240}^{g\ell}$, q_{240}^s and q_{280}^g are displayed on Figure 6.12. This confirms what was previously mentioned: the UA posterior distributions significantly underestimate the true values even with $\epsilon = 0.05$ and become plainly wrong as ϵ further increases. On the contrary, the DA posterior distributions are always centered on the true values, with rather low variances. For $\epsilon = 0.20$, the posterior distributions of UA and DA barely overlap. This illustrates a strong robustness of the RPF to ill-chosen priors and highlights the strength of the approach considered here to great changes, whether they be due to environment or genotype.

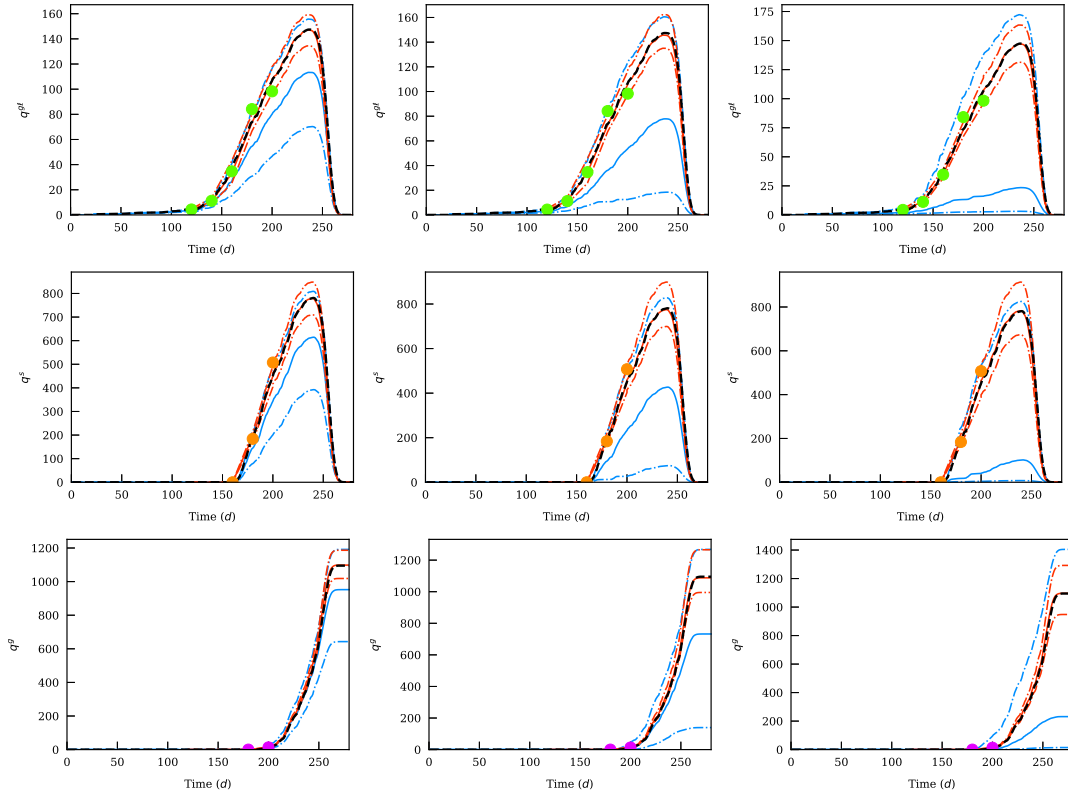


Figure 6.11: Predictions of the different biomasses in the case of uncertainty analysis (UA, blue lines) and data assimilation (DA, red lines). The 95% CI boundaries are indicated with dashed lines. The black lines correspond to the true hidden states and the observations used for data assimilation are represented by filled circles. The three cases considered here are $\epsilon = 0.05$ (left), $\epsilon = 0.10$ (center) and $\epsilon = 0.20$ (right).

	$\epsilon = 0.05$		$\epsilon = 0.10$		$\epsilon = 0.20$	
	UA	DA	UA	DA	UA	DA
δ^μ	$5.000 \cdot 10^{-02}$	$6.664 \cdot 10^{-03}$	$1.000 \cdot 10^{-01}$	$3.577 \cdot 10^{-03}$	$2.000 \cdot 10^{-01}$	$5.936 \cdot 10^{-03}$
δ^e	$5.000 \cdot 10^{-02}$	$4.243 \cdot 10^{-03}$	$1.000 \cdot 10^{-01}$	$2.444 \cdot 10^{-03}$	$2.000 \cdot 10^{-01}$	$1.217 \cdot 10^{-03}$
δ^{μ^g}	$5.000 \cdot 10^{-02}$	$1.154 \cdot 10^{-03}$	$1.000 \cdot 10^{-01}$	$8.309 \cdot 10^{-04}$	$2.000 \cdot 10^{-01}$	$1.196 \cdot 10^{-03}$
$\delta_{240}^{q^g \ell}$	$2.307 \cdot 10^{-01}$	$6.167 \cdot 10^{-03}$	$4.719 \cdot 10^{-01}$	$1.490 \cdot 10^{-02}$	$8.409 \cdot 10^{-01}$	$5.680 \cdot 10^{-03}$
$\rho_{240}^{q^g \ell}$	$3.795 \cdot 10^{-01}$	$8.518 \cdot 10^{-02}$	$7.625 \cdot 10^{-01}$	$7.288 \cdot 10^{-02}$	$8.681 \cdot 10^{-01}$	$1.069 \cdot 10^{-01}$
$\alpha_{240}^{q^g \ell}$	$8.081 \cdot 10^{-01}$	$7.498 \cdot 10^{-02}$	$8.447 \cdot 10^{-01}$	$2.076 \cdot 10^{-01}$	$8.387 \cdot 10^{-01}$	$5.110 \cdot 10^{-02}$
$\delta_{240}^{q^s}$	$2.134 \cdot 10^{-01}$	$3.238 \cdot 10^{-03}$	$4.535 \cdot 10^{-01}$	$8.567 \cdot 10^{-03}$	$8.704 \cdot 10^{-01}$	$1.411 \cdot 10^{-03}$
$\rho_{240}^{q^s}$	$3.616 \cdot 10^{-01}$	$8.889 \cdot 10^{-02}$	$8.255 \cdot 10^{-01}$	$9.746 \cdot 10^{-02}$	$9.235 \cdot 10^{-01}$	$1.390 \cdot 10^{-01}$
$\alpha_{240}^{q^s}$	$8.569 \cdot 10^{-01}$	$3.658 \cdot 10^{-02}$	$8.840 \cdot 10^{-01}$	$8.866 \cdot 10^{-02}$	$9.389 \cdot 10^{-01}$	$1.014 \cdot 10^{-02}$
$\delta_{279}^{q^g}$	$1.304 \cdot 10^{-01}$	$2.999 \cdot 10^{-03}$	$3.312 \cdot 10^{-01}$	$6.598 \cdot 10^{-03}$	$7.886 \cdot 10^{-01}$	$1.374 \cdot 10^{-03}$
$\rho_{279}^{q^g}$	$3.249 \cdot 10^{-01}$	$7.255 \cdot 10^{-02}$	$8.095 \cdot 10^{-01}$	$8.514 \cdot 10^{-02}$	$9.373 \cdot 10^{-01}$	$1.355 \cdot 10^{-01}$
$\alpha_{279}^{q^g}$	$5.960 \cdot 10^{-01}$	$4.122 \cdot 10^{-02}$	$6.771 \cdot 10^{-01}$	$7.802 \cdot 10^{-02}$	$7.361 \cdot 10^{-01}$	$1.013 \cdot 10^{-02}$

Table 6.5: Results for the different relative errors and other coefficients calculated in order to identify the quality of the estimates provided by uncertainty analysis (UA) and data assimilation (DA) with different prior distributions $\epsilon = 0.05$, $\epsilon = 0.10$ and $\epsilon = 0.20$.

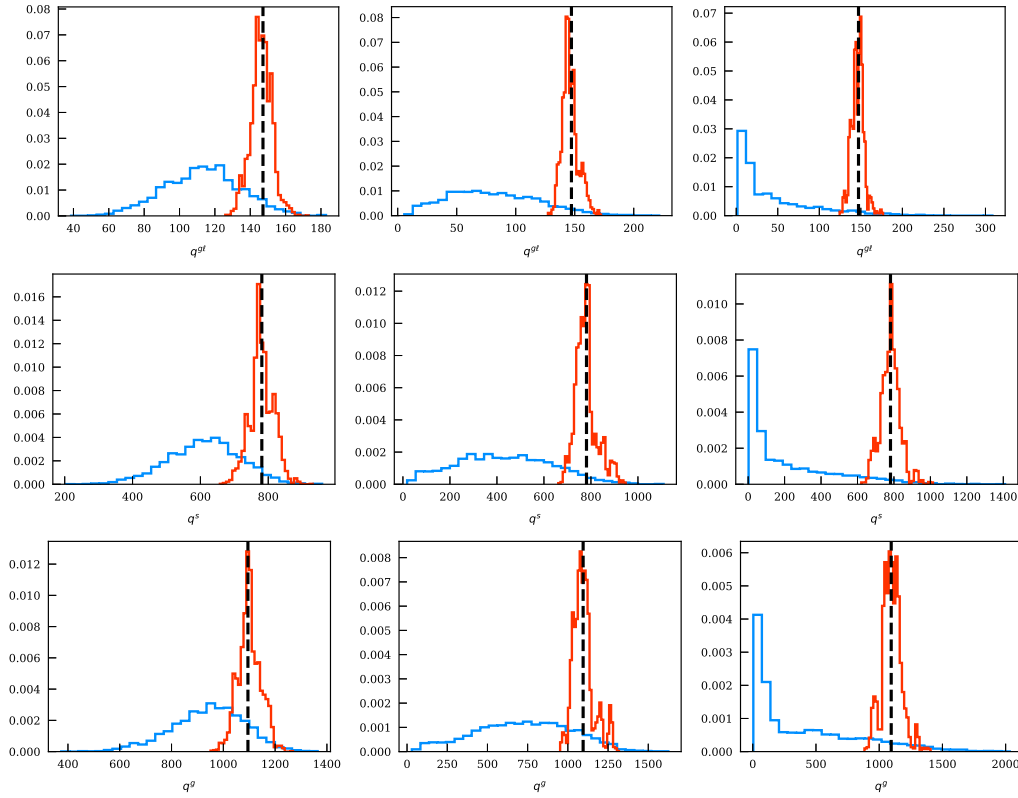


Figure 6.12: Prior (in blue, corresponding to UA's predictions) and posterior (in red, corresponding to DA's predictions) distributions for the different variables of interest q_{240}^{gl} (top), q_{240}^s (middle) and q_{280}^o (bottom) for different prior distributions $\epsilon = 0.05$ (left), $\epsilon = 0.10$ (center) and $\epsilon = 0.20$ (right). The black lines correspond to the true hidden states.

6.2.7 Influence of the number of estimated parameters

So far we have considered cases where relatively few parameters of the LNAS model were estimated. For real applications, one might need to estimate more than 3 parameters, and the difficulty related to the estimation problem with relatively few data becomes more challenging. To illustrate this aspect, we designed 3 cases where the estimated parameters were:

$$\begin{cases} \theta = (\mu, e, \mu^g), & \text{i.e. } d = 3, \\ \theta = (\mu, e, \mu^g, \mu^\odot), & \text{i.e. } d = 4, \\ \theta = (\mu, e, \mu^g, \mu^\odot, \tau^g), & \text{i.e. } d = 5. \end{cases} \quad (6.11)$$

The results are displayed on Figure 6.13. As the number of estimated parameters increases, we notice that the estimation of the hidden states during the data assimilation phase remains excellent, but degrades as time passes and there is no observation available to correct the particles. For $d = 4$, until $n = 230$ the estimated curves for q^{gl} and q^r follow perfectly those of the true hidden states, but slightly before the peak of the curve the estimated curves drop too early, which is not the case for $d = 3$. However, the three parameters estimated for $d = 3$ are even better estimated for $d = 4$ as can be seen from the values of the δ coefficients in Table 6.6. This means that the deviation from the true curves can only be attributed to the additional estimated parameters μ^\odot . This is all the more sensible that μ^\odot is heavily involved in the allocation of biomass during

the late stages of plant growth.

Still, the estimates for the state variables of interest remain rather good, with δ coefficients not exceeding $7 \cdot 10^{-02}$ for $d = 4$ and $2 \cdot 10^{-02}$ for $d = 5$. This divergence effect between the true curves and the estimated ones is indeed less pronounced in the case of $d = 5$. Although this might seem counter-intuitive at first glance, this is partly explained by a better estimation of μ^{\odot} in the case of $d = 5$ whose role during the last growth stages is significant, as was suggested by Sobol analysis illustrated on Figure 6.6. We must finally mention that the state estimates of $q^{g^{\ell}}$ and q^r would have probably been not as good if we had considered later times (after $n = 250$).

The dispersion of the curves remains reasonable and manages to capture the true values inside the credible interval, with values for the α coefficients below $5 \cdot 10^{-01}$ for $d = 4$ and $6 \cdot 10^{-01}$ for $d = 5$. All in all, data assimilation keeps on improving the predictions for all state variables with an increasing number of estimated parameters.

As a conclusion, the use of an efficient SMC method such as the regularized particle filter for the assimilation of data available in the early stages of plant growth appears to be an essential tool for accurate model predictions, which is crucial for agriculture applications. This has proved to work even in the case of scarce data with at most 14 observations on all possible model variables. It improves model predictions even in the case of highly noisy observations, and is very robust to prior distributions far from the optimal values for the parameters, which will undeniably be useful for model calibration in different contexts.

6.3 Parameter and state estimation using the PMMH sampler for the LNAS sugar beet model

The last case study we investigate is that of joint parameter and state estimation using the LNAS model for sugar beet. As explained in Chapter 3, both MCMC and SMC methods have their limitations, in particular when the model contains process noise as is the case of the one considered here. In order to obtain better estimates for the hidden states, and with the ultimate goal of estimating the parameters underlying the distributions of the process and observation noises, we will focus our attention on the PMMH algorithm introduced in Section 3.5.3.

6.3.1 Sensitivity analysis

We first start as usual by performing a sensitivity analysis routine in order to identify the most influential model parameters, whose results can be seen on Figure 6.14. Sobol indices were computed for the biomass of the green leaves $q^{g^{\ell}}$ and the biomass of the roots q^r . In the early stages of the growth, the allocation coefficient γ^0 is by far the most influential parameter as it controls how the biomass is divided between the different compartments. The radiation use efficiency μ also plays a significant role throughout the whole growth as it influences the biomass produced on each day. Finally the leaf mass per area e is also of importance. Other

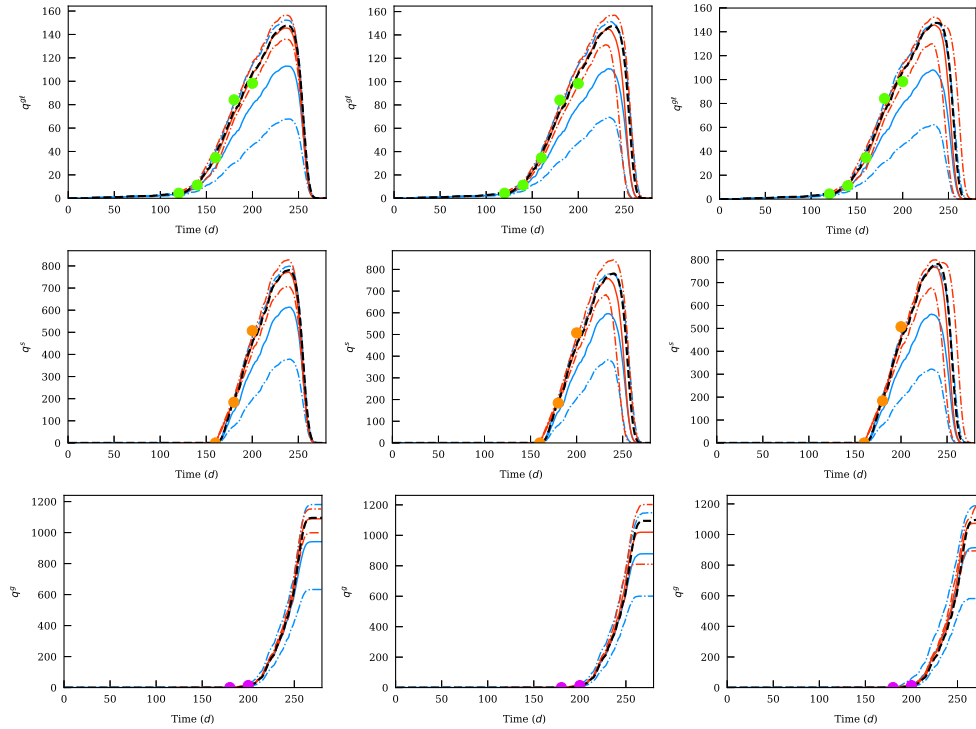


Figure 6.13: Predictions of the different biomasses in the case of uncertainty analysis (UA, blue lines) and data assimilation (DA, red lines). The 95% CI boundaries are indicated with dashed lines. The black lines correspond to the true hidden states and the observations used for data assimilation are represented by filled circles. The three cases considered here are $d = 3$ (left), $d = 4$ (center) and $d = 5$ (right).

	$d = 3$		$d = 4$		$d = 5$	
	UA	DA	UA	DA	UA	DA
δ^μ	$5.000 \cdot 10^{-02}$	$6.249 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$4.724 \cdot 10^{-04}$	$5.000 \cdot 10^{-02}$	$1.219 \cdot 10^{-02}$
δ^e	$5.000 \cdot 10^{-02}$	$1.421 \cdot 10^{-02}$	$5.000 \cdot 10^{-02}$	$8.293 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$2.233 \cdot 10^{-02}$
δ^{μ^g}	$5.000 \cdot 10^{-02}$	$1.973 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$1.368 \cdot 10^{-03}$	$5.000 \cdot 10^{-02}$	$3.468 \cdot 10^{-02}$
δ^{μ°	o	o	$5.000 \cdot 10^{-02}$	$4.857 \cdot 10^{-02}$	$5.000 \cdot 10^{-02}$	$3.376 \cdot 10^{-02}$
$\delta^{\mu^{\rightarrow g}}$	o	o	o	o	$5.000 \cdot 10^{-02}$	$1.047 \cdot 10^{-02}$
$\delta_{240}^{q^g \ell}$	$2.336 \cdot 10^{-01}$	$1.648 \cdot 10^{-02}$	$2.788 \cdot 10^{-01}$	$6.696 \cdot 10^{-02}$	$3.042 \cdot 10^{-01}$	$2.993 \cdot 10^{-02}$
$\rho_{240}^{q^g \ell}$	$3.994 \cdot 10^{-01}$	$6.567 \cdot 10^{-02}$	$3.873 \cdot 10^{-01}$	$2.570 \cdot 10^{-01}$	$4.349 \cdot 10^{-01}$	$2.322 \cdot 10^{-01}$
$\alpha_{240}^{q^g \ell}$	$8.814 \cdot 10^{-01}$	$2.211 \cdot 10^{-01}$	$9.726 \cdot 10^{-01}$	$5.074 \cdot 10^{-01}$	$1.022 \cdot 10^{+00}$	$5.880 \cdot 10^{-01}$
$\delta_{240}^{q^s}$	$2.143 \cdot 10^{-01}$	$1.252 \cdot 10^{-02}$	$2.646 \cdot 10^{-01}$	$6.269 \cdot 10^{-02}$	$3.033 \cdot 10^{-01}$	$2.584 \cdot 10^{-02}$
$\rho_{240}^{q^s}$	$3.833 \cdot 10^{-01}$	$8.322 \cdot 10^{-02}$	$3.646 \cdot 10^{-01}$	$2.538 \cdot 10^{-01}$	$4.337 \cdot 10^{-01}$	$2.198 \cdot 10^{-01}$
$\alpha_{240}^{q^s}$	$9.037 \cdot 10^{-01}$	$1.765 \cdot 10^{-01}$	$1.017 \cdot 10^{+00}$	$4.377 \cdot 10^{-01}$	$1.057 \cdot 10^{+00}$	$5.480 \cdot 10^{-01}$
$\delta_{279}^{q^g}$	$1.404 \cdot 10^{-01}$	$5.752 \cdot 10^{-03}$	$1.975 \cdot 10^{-01}$	$6.814 \cdot 10^{-02}$	$1.653 \cdot 10^{-01}$	$2.058 \cdot 10^{-02}$
$\rho_{279}^{q^g}$	$3.278 \cdot 10^{-01}$	$8.244 \cdot 10^{-02}$	$3.166 \cdot 10^{-01}$	$2.060 \cdot 10^{-01}$	$3.636 \cdot 10^{-01}$	$1.673 \cdot 10^{-01}$
$\alpha_{279}^{q^g}$	$6.412 \cdot 10^{-01}$	$9.899 \cdot 10^{-02}$	$8.027 \cdot 10^{-01}$	$4.101 \cdot 10^{-01}$	$6.603 \cdot 10^{-01}$	$1.825 \cdot 10^{-01}$

Table 6.6: Results for the different relative errors and other coefficients calculated in order to identify the quality of the estimates provided by uncertainty analysis (UA) and data assimilation (DA) with different numbers of estimated parameters $d = 3$, $d = 4$ and $d = 5$.

influential parameters include γ^ℓ and μ^a , although the former influences only the late growth stages and the latter does not have such an impact on the biomass of the roots. We still chose to include μ^a over e since experimental data often relates to the late growth stage and μ^a is less known from a biological perspective, the set of estimated parameters therefore being $\theta = (\mu, \gamma^0, \mu^a)$.

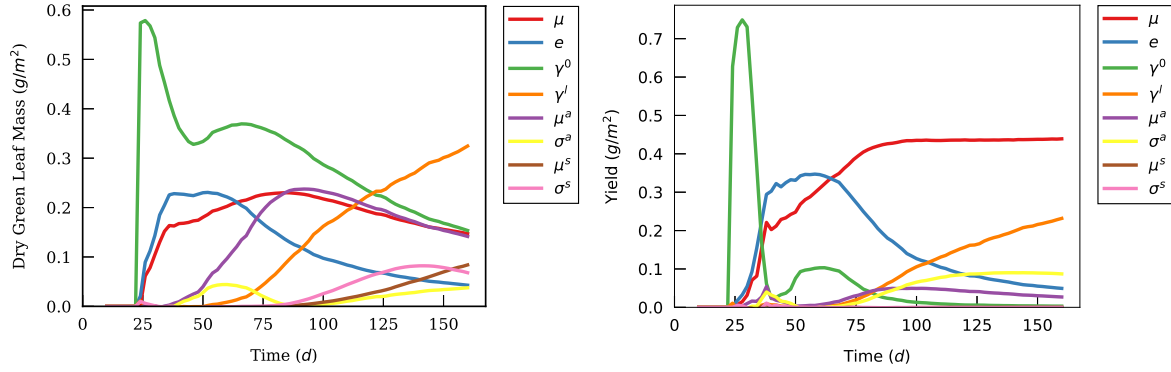


Figure 6.14: Sobol indices for the state variables $q^{g\ell}$ and q^r from $n = 1$ to $n = 160$.

6.3.2 Data simulation

Real experimental data sets used in previous works [Chen, 2014], [Baey, 2014] consisted of 14 different observations on the two variables $q^{g\ell}$ and q^r . We made the decision to also simulate data for these two variables on a similar number of days. Obviously, we could have used real data sets, but we would not have been able to compare the results of the estimation for the hidden states to their true values. We simulated data with the following parameters set:

$$\left\{ \begin{array}{l} \mu^{\text{true}} = 3.746 \cdot 10^{+00}, \\ k^{\text{true}} = 7.000 \cdot 10^{-01}, \\ e^{\text{true}} = 5.660 \cdot 10^{+01}, \\ \gamma^{0,\text{true}} = 7.525 \cdot 10^{-01}, \\ \gamma^{\ell,\text{true}} = 1.035 \cdot 10^{-01}, \\ \mu^{a,\text{true}} = 5.790 \cdot 10^{+02}, \\ \sigma^{a,\text{true}} = 9.500 \cdot 10^{+02}, \end{array} \right. \quad (6.12)$$

with a multiplicative normal observation noise having standard deviation $\sigma = 0.1$ and observed the values of the biomass of green leaves $\tilde{q}^{g\ell}$ and that of the roots \tilde{q}^r on 11 days:

$$\mathcal{O} = \{50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150\}. \quad (6.13)$$

6.3.3 Comparison of filters within PMMH

The aim of this case study is therefore to perform a joint estimation of the model parameters and hidden states $q^{g\ell}$ and q^r using the PMMH sampler, with different SMC algorithms. Indeed, at each iteration of the

PMMH, an SMC method must be used in order to accurately sample the hidden states. Different strategies can be considered. Using a simple filter such as the UKF (where the number of particles is fixed) or the EnKF with a low number of particles could provide decent samples, even though not optimal, with a rather low computing time. Another one is to use more refined filters at each iteration, such as EnKF or, even better, RPF algorithms with a high number of particles. It is awaited that they would provide better estimates as they overcome some drawbacks of simpler filters such as narrow state space exploration or weight degeneracy. A PMMH algorithm with $M = 30,000$ iterations was run on this simulated data. Different SMC algorithms were used: an unscented Kalman filter (UKF), two ensemble Kalman filters, one with 10^2 particles (EnKF-2) and the other with 10^3 particles (EnKF-3), and finally two RPF filters, one with 10^2 particles (RPF-2) and the other with 10^3 particles (RPF-3). The results were obtained by running 10 times the same application with the following prior distributions originating from [Chen, 2014]:

$$\begin{cases} \mu &= \mathcal{N}\left(0.961 \mu^{\text{true}}, (0.08 \mu^{\text{true}})^2\right), \\ \gamma^0 &= \mathcal{N}\left(0.997 \gamma^{0,\text{true}}, (0.13 \gamma^{0,\text{true}})^2\right), \\ \mu^a &= \mathcal{N}\left(1.036 \mu^{a,\text{true}}, (0.09 \mu^{a,\text{true}})^2\right). \end{cases} \quad (6.14)$$

	UKF	EnKF-2	EnKF-3	RPF-2	RPF-3
δ^μ	$1.023 \cdot 10^{-03}$	$8.603 \cdot 10^{-03}$	$1.091 \cdot 10^{-02}$	$8.414 \cdot 10^{-03}$	$7.508 \cdot 10^{-03}$
$\delta\gamma^0$	$1.027 \cdot 10^{-03}$	$1.042 \cdot 10^{-03}$	$5.065 \cdot 10^{-03}$	$1.104 \cdot 10^{-03}$	$3.204 \cdot 10^{-03}$
$\delta\mu^a$	$1.780 \cdot 10^{-02}$	$9.020 \cdot 10^{-03}$	$7.103 \cdot 10^{-03}$	$1.309 \cdot 10^{-02}$	$1.607 \cdot 10^{-02}$
$\sigma^\mu / \mu^{\text{true}}$	$1.383 \cdot 10^{-02}$	$1.295 \cdot 10^{-02}$	$1.489 \cdot 10^{-02}$	$1.340 \cdot 10^{-02}$	$1.212 \cdot 10^{-02}$
$\sigma\gamma^0 / \gamma^{0,\text{true}}$	$1.969 \cdot 10^{-02}$	$1.789 \cdot 10^{-02}$	$2.254 \cdot 10^{-02}$	$1.829 \cdot 10^{-02}$	$1.612 \cdot 10^{-02}$
$\sigma\mu^a / \mu^{a,\text{true}}$	$2.867 \cdot 10^{-02}$	$2.447 \cdot 10^{-02}$	$2.676 \cdot 10^{-02}$	$2.318 \cdot 10^{-02}$	$1.971 \cdot 10^{-02}$

Table 6.7: Relative errors on the parameters δ^{θ_i} and normalized standard deviations of the estimates $\sigma^{\theta_i} / \theta_i^{\text{true}}$ for the different SMC algorithms used within the PMMH sampler.

The relative errors on the estimated parameters as well as the related normalized standard deviations are displayed in Table 6.7. It shows that UKF yields the best results for μ and γ^0 , but performs slightly worse for μ^a . Indeed, even though the relative error on this parameter is ten times that on the first two, it seems that, in the case of all SMC algorithms, the PMMH struggles a bit more to estimate the true values of μ^a compared to μ and γ^0 . Rather surprisingly, EnKF-3 provides estimates not as good as EnKF-2 except for μ^a . Equivalently, RPF-3 performs a bit better than RPF-2 on the first parameter. Increasing the number of particles for a given filter does not seem to improve considerably the accuracy on the parameters' estimates. Still, the relative errors on the true values remain acceptable for all parameters and filters, but we note that UKF's performance is surprisingly good.

Concerning the standard deviations, they all are of the same orders of magnitude, comprised between $1.2 \cdot 10^{-02}$ and $1.5 \cdot 10^{-02}$ for μ , $1.6 \cdot 10^{-02}$ and $2.3 \cdot 10^{-02}$ for γ^0 , and $1.9 \cdot 10^{-02}$ and $2.9 \cdot 10^{-02}$ for

μ^a . The standard deviation related to μ^a is therefore almost the double of that for μ and overall, no filters significantly outperformed the others.

		UKF	EnKF-2	EnKF-3	RPF-2	RPF-3
RMSEP	$\hat{q}_{50:5:100}^{g^\ell}$	$9.778 \cdot 10^{+00}$	$4.999 \cdot 10^{+00}$	$5.071 \cdot 10^{+00}$	$4.690 \cdot 10^{+00}$	$4.263 \cdot 10^{+00}$
	$\hat{q}_{50:5:100}^r$	$6.402 \cdot 10^{+00}$	$1.868 \cdot 10^{+01}$	$2.187 \cdot 10^{+01}$	$1.835 \cdot 10^{+01}$	$1.766 \cdot 10^{+01}$
EF	$\hat{q}_{50:5:100}^{g^\ell}$	$9.965 \cdot 10^{-01}$	$9.992 \cdot 10^{-01}$	$9.992 \cdot 10^{-01}$	$9.993 \cdot 10^{-01}$	$9.994 \cdot 10^{-01}$
	$\hat{q}_{50:5:100}^r$	$9.999 \cdot 10^{-01}$	$9.993 \cdot 10^{-01}$	$9.991 \cdot 10^{-01}$	$9.994 \cdot 10^{-01}$	$9.994 \cdot 10^{-01}$

Table 6.8: Root mean square error prediction and modelling efficiency on the two hidden states of interest q^{g^ℓ} and q^r for the different SMC algorithms used within the PMMH sampler.

The choice of the SMC method might be felt more on the estimation of the hidden states. To assess the accuracy of the latter, we computed the RMSEP and the modelling efficiency for q^{g^ℓ} and q^r separately for each filter over the whole timeline. We must mention that here we compared estimated hidden states to true hidden states, and not to noised observations. The results are detailed in Table 6.8. The dissociation of the different variables appears particularly relevant in the case of this model. Indeed, UKF seems to perform worse for the estimation of q^{g^ℓ} than the other filters but much better as far as q^r is concerned. For q^{g^ℓ} , UKF has an RMSEP twice higher than the other filters, whereas it has an RMSEP three times lower for q^r . It is also worth noting that the RPF algorithms perform better than their EnKF counterparts for both variables. A closer look at the values of the modelling efficiency indicates that the EnKF and RPF algorithms perform very well overall (EF > 0.999) and in an identical manner for the two variables. On the contrary, UKF has a significantly better modelling efficiency for q^r (EF > 0.9999) whereas that for q^{g^ℓ} is the only one below the threshold of 0.999.

Still the estimation of the hidden states remain excellent for all SMC algorithms and the PMMH lives up to its reputation. This is better seen on Figures 6.15 and 6.16. On these two figures, the timeline of the estimated states are shifted – differently for each filter – for the sake of readability (otherwise, all error bars would overlap), which might be deceptive for the first figure that is mostly shown to see that the estimated hidden states follow the dynamics of the growth curves very well. The relative performance of all filters is best observed on Figure 6.16 where at each time, all values are divided by that of the true hidden state. The truth is therefore represented by the horizontal line $y = 1$ and the shifting of timelines does not matter anymore. First, it is worth noting that the hidden states are always underestimated, which might owe to the choice of the prior distributions for the parameters. The difference between UKF and the other filters on the two variables is particularly visible. Most importantly, the relative estimation of the hidden states becomes better with time (and the number of observations). This is all the more relevant that, in practice, one is particularly interested in the late values of the biomasses. The difference between the EnKF and RPF filters are negligible: RPF-2 and in particular RPF-3 seem to provide slightly better estimates during the late growth stages, despite further beginnings, which is in favour of the RPF.

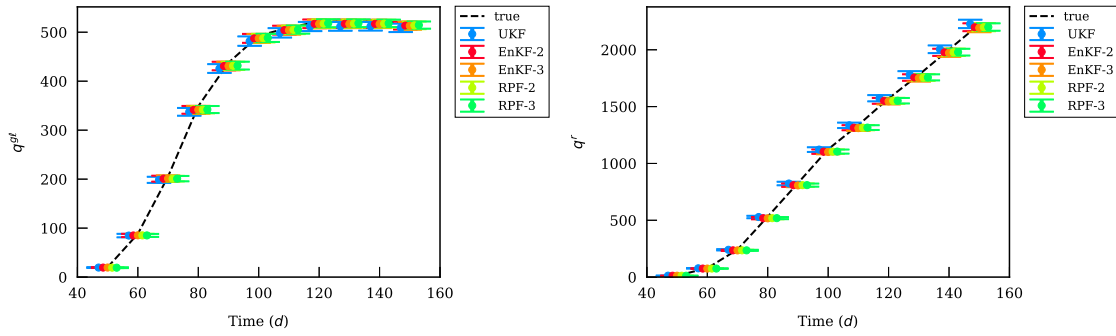


Figure 6.15: Error bars for the estimated hidden states of the biomass of green leaves q^{gl} and roots q^r for the different SMC algorithms used within the PMMH algorithm. The true hidden states are represented by the black dashed line and, for better readability, the error bars associated to each algorithm are shifted w.r.t. the x -axis.

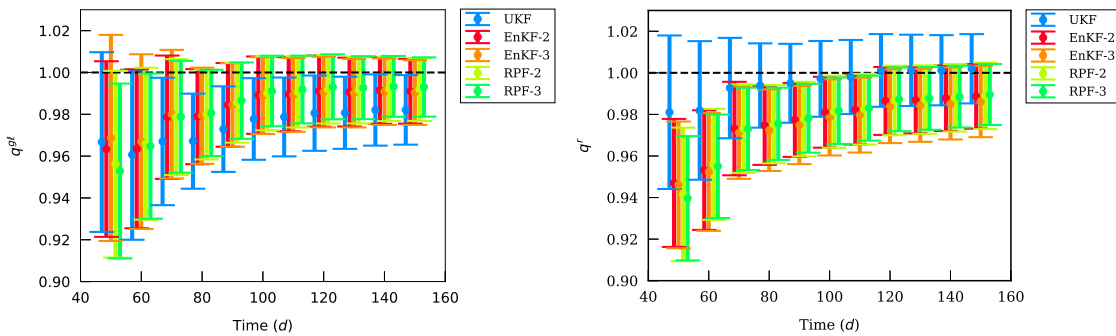


Figure 6.16: Error bars for the estimated hidden states of the normalized biomass of green leaves $q^{gl}/q^{gl,true}$ and roots $q^r/q^{r,true}$ for the different SMC algorithms used within the PMMH algorithm. The true hidden states are represented by the black dashed line and, for better readability, the error bars associated to each algorithm are a bit shifted w.r.t. the x -axis.

UKF	EnKF-2	EnKF-3	RPF-2	RPF-3
$2.253 \cdot 10^{+02}$	$8.923 \cdot 10^{+02}$	$8.783 \cdot 10^{+03}$	$8.549 \cdot 10^{+02}$	$7.711 \cdot 10^{+03}$

Table 6.9: Computing times (in seconds) necessary for running the different SMC algorithms used within the PMMH sampler.

The PMMH algorithms with these different SMC algorithms were all run sequentially. The related computing times can be found in Table 6.9. The UKF, which was run with only 13 particles, took only 3min45s, whereas filters EnKF-2 and RPF-2 took around 15min and 14min respectively, and EnKF-3 and RPF-3 took around 2h25min and 2h08min respectively.

From all these results, it appears always preferable to choose the RPF algorithm over its EnKF counterpart when the two have an identical number of particles, as it provides better parameter and state estimates overall and requires less computing time. As far as UKF is concerned, its strength lies in the very few number of particles it uses; it still manages to obtain very good estimates for both the parameters and the hidden states. It outclassed EnKF and RPF for two out of three parameters and one hidden state. Because of its low computing time and the good estimates it provides, it represents an excellent choice for the joint estimation of parameters and hidden states within a PMMH sampler. This echoes the recent results of [Sherlock et al.](#)

[2017] who investigated the performance of PMMH samplers and found that if the computational cost of the algorithm is proportional to the number of particles N of the SMC algorithm, it is often better to set N as low as possible.

Of course, all these filters could be parallelized (and, as a matter of fact, they have been in ADJUSTIN' as detailed notably in Section 5.6.3). The considerations of the different computing times therefore becomes irrelevant as soon as we dispose of more than 10^3 processes where all methods should require an equivalent computing time. In practice, this is rarely the case and even though UKF's computing time cannot be divided by more than a factor of 10, it will almost always remain the fastest. In conclusion, because of the relatively high cost in terms of memory and computing time of the PMCMC algorithms, using a PMMH sampler together with a UKF algorithm might appear as a very attractive strategy for the joint estimation of parameters and hidden states.

6.3.4 Estimation of process and observation noises

This paves the way for an accurate estimation of the process and observation noises within plant growth models as the joint estimation of the deterministic and stochastic values for a single state variable at different time steps can be used for the determination of the parameters underlying the distribution of the process and observation noises, such as standard deviations.

To investigate this issue, we first simulated data using the same values of parameters with observations for the biomasses of green leaves and roots from day 50 to day 150, $\tilde{q}_{50:150}^{g\ell}$ and $\tilde{q}_{50:150}^r$. Since it is expected that, in the real system under study, observation noises are of much higher intensity than process noises, the standard deviation of the process noises (for biomass production and allocation, see Equation 2.12) was set to $\sigma^q = \sigma^\gamma = 0.02$ and that of the observation noises (for the observation of the biomasses of green leaves and roots, see Equation 2.14) to $\sigma^{g\ell} = \sigma^r = 0.1$.

For the estimation procedure of the process and observation noise parameters we adopt a Bayesian approach where all parameters are given prior distributions. For the particular case under study, the full likelihood $\ell \doteq \ell(x_{1:T}, y_{1 \rightarrow t} | \theta, \sigma^{g\ell}, \sigma^r, \sigma^q, \sigma^\gamma)$ reads:

$$\ell \propto \prod_{k=1}^O \left(2\pi(\sigma^{g\ell} q_{t_k}^{g\ell})^2 \right)^{-1/2} e^{-\frac{(\tilde{q}_{t_k}^{g\ell} - q_{t_k}^{g\ell})^2}{2(\sigma^{g\ell} q_{t_k}^{g\ell})^2}} \prod_{k=1}^O \left(2\pi(\sigma^r q_{t_k}^r)^2 \right)^{-1/2} e^{-\frac{(\tilde{q}_{t_k}^r - q_{t_k}^r)^2}{2(\sigma^r q_{t_k}^r)^2}} \prod_{n=1}^T \left(2\pi(\sigma^q q_n^{\det})^2 \right)^{-1/2} e^{-\frac{(q_n^{\text{sto}} - q_n^{\det})^2}{2(\sigma^q q_n^{\det})^2}} \prod_{n=1}^T \left(2\pi(\sigma^\gamma \gamma_n^{\det})^2 \right)^{-1/2} e^{-\frac{(\gamma_n^{\text{sto}} - \gamma_n^{\det})^2}{2(\sigma^\gamma \gamma_n^{\det})^2}} \quad (6.15)$$

and the posterior distribution is therefore:

$$\begin{aligned} p(\theta, \sigma^{g\ell}, \sigma^r, \sigma^q, \sigma^\gamma | x_{1:T}, y_{1 \rightarrow T}) &\propto \ell(x_{1:T}, y_{1 \rightarrow T} | \theta, \sigma^{g\ell}, \sigma^r, \sigma^q, \sigma^\gamma) p(\theta, \sigma^{g\ell}, \sigma^r, \sigma^q, \sigma^\gamma) \\ &\propto \ell(x_{1:T}, y_{1 \rightarrow T} | \theta, \sigma^{g\ell}, \sigma^r, \sigma^q, \sigma^\gamma) p(\theta) p(\sigma^{g\ell}) p(\sigma^r) p(\sigma^q) p(\sigma^\gamma). \end{aligned}$$

As emphasized in Chapter 4, if the prior distributions for the process and observation noise parameters are appropriately chosen, the full conditional distributions of these parameters can be analytically derived. In particular, if each variance related to the noises follows an inverse Gamma distribution, then their full

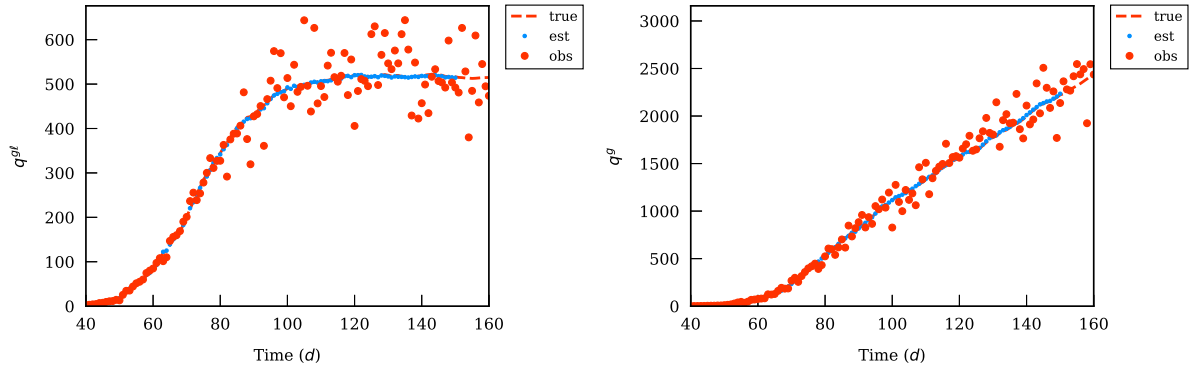


Figure 6.17: Estimation of the hidden states related to the observation noises q^{gl} (left) and q^r (right): the true hidden states are displayed as a dashed red line and the corresponding observations as filled red circles; the estimated hidden states are represented by the blue line.

conditional distributions will too. More precisely, taking the example of the observation noise related to the biomass of green leaves, if $(\sigma^{gl})^2 \sim \mathcal{IG}(\alpha^{gl}, \beta^{gl})$, then:

$$\begin{aligned} p((\sigma^{gl})^2 | \dots) &\propto \prod_{k=1}^O \left(2\pi(\sigma^{gl} q_{t_k}^{gl})^2 \right)^{-1/2} e^{-\frac{(\tilde{q}_{t_k}^{gl} - q_{t_k}^{gl})^2}{2(\sigma^{gl} q_{t_k}^{gl})^2}} \frac{(\beta^{gl})^{\alpha^{gl}}}{\Gamma(\alpha^{gl})} ((\sigma^{gl})^2)^{-\alpha^{gl}-1} e^{-\frac{\beta^{gl}}{(\sigma^{gl})^2}} \\ &\propto ((\sigma^{gl})^2)^{-(\alpha^{gl} + \frac{O}{2})-1} \exp \left[- \left(\beta^{gl} + \frac{1}{2} \sum_{k=1}^O \frac{(\tilde{q}_{t_k}^{gl} - q_{t_k}^{gl})^2}{(q_{t_k}^{gl})^2} \right) / (\sigma^{gl})^2 \right] \end{aligned} \quad (6.16)$$

and $(\sigma^{gl})^2$ follows an inverse Gamma distribution with updated parameters:

$$\begin{cases} \alpha^{gl,*} &= \alpha^{gl} + \frac{O}{2}, \\ \beta^{gl,*} &= \beta^{gl} + \frac{1}{2} \sum_{k=1}^O \frac{(\tilde{q}_{t_k}^{gl} - q_{t_k}^{gl})^2}{(q_{t_k}^{gl})^2}. \end{cases} \quad (6.17)$$

Equivalent formulas can be derived for the other noise parameters. For the estimation procedure, we adopt the following strategy summarized in Algorithm 13: we first run a PMMH in order to estimate the functional parameters θ and the hidden states $q_{50:150}^{gl}$ and $q_{50:150}^r$. From the latter we can accurately deduce the values of the standard deviation for the observation noises as:

$$\begin{aligned} (\sigma^{gl})^2 &\sim \mathcal{IG} \left(\alpha^{gl} + \frac{O}{2}, \beta^{gl} + \frac{1}{2} \sum_{k=1}^O \frac{(\tilde{q}_{t_k}^{gl} - q_{t_k}^{gl})^2}{(q_{t_k}^{gl})^2} \right), \\ (\sigma^r)^2 &\sim \mathcal{IG} \left(\alpha^r + \frac{O}{2}, \beta^r + \frac{1}{2} \sum_{k=1}^O \frac{(\tilde{q}_{t_k}^r - q_{t_k}^r)^2}{(q_{t_k}^r)^2} \right). \end{aligned} \quad (6.18)$$

As previously suggested by our initial study, we used a PMMH sampler where at each iteration an unscented Kalman filter is run and chose a total number of $M = 30,000$ iterations. The expectations of the prior distributions for the standard deviations of the process and observation noises are chosen higher than their expected values for a better state space exploration: $\mathbb{E}(\sigma^q) = \mathbb{E}(\sigma^r) = 0.05$ and $\mathbb{E}(\sigma^{gl}) = \mathbb{E}(\sigma^r) = 0.2$. The results of the first estimation for the observed states q^{gl} and q^r is displayed on Figure 6.17. Their values are very well estimated and this first PMMH algorithm allows a very precise estimation of the standard deviation for the observation noises as we obtained $\hat{\sigma}^{gl} = 0.1001$ and $\hat{\sigma}^r = 0.0942$.

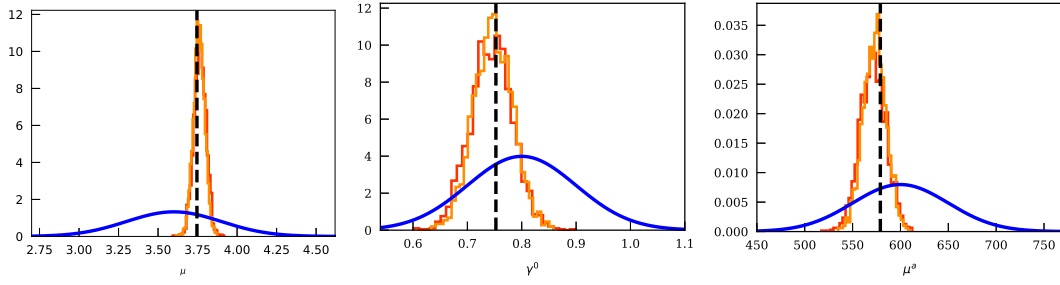


Figure 6.18: Comparison of the posterior distribution for the parameters μ (left), γ^0 (center) and μ^a (right) with underestimated process noise ($\sigma = 0.01$, red curve) and true process noise ($\sigma = 0.02$, orange curve).

With adequate estimates for the functional parameters θ and the observation standard deviations σ^{g^ℓ} and σ^r , these new quantities are used for a second PMMH algorithm aimed at refining the estimates on the parameters, hidden states and process standard deviations σ^q and σ^a . We used the posterior distributions for the functional parameters obtained with the first PMMH run as a prior distribution for the second one. We adopted this two-step strategy as the estimation of the observation noise parameters is extremely accurate and provide values close to the truth for a refined estimation, which is more efficient than updating all noise parameters at once.

The values of the observation standard deviations were hence fixed to those found at the end of the first PMMH algorithm, $\sigma^{g^\ell} = 0.1001$ and $\sigma^r = 0.0942$. The estimated hidden states comprised those related to observations q^{g^ℓ} and q^r and those related to the different process noises q^{det} , q^{sto} and γ^{det} , γ^{sto} . The process variances were then inferred from the hidden states as:

$$\begin{aligned} (\sigma^q)^2 &\sim \mathcal{IG} \left(\alpha^q + \frac{T}{2}, \beta^q + \frac{1}{2} \sum_{n=1}^T \frac{(q_n^{\text{sto}} - q_n^{\text{det}})^2}{(q_n^{\text{det}})^2} \right), \\ (\sigma^\gamma)^2 &\sim \mathcal{IG} \left(\alpha^\gamma + \frac{T}{2}, \beta^\gamma + \frac{1}{2} \sum_{n=1}^T \frac{(\gamma_n^{\text{sto}} - \gamma_n^{\text{det}})^2}{(\gamma_n^{\text{det}})^2} \right). \end{aligned} \quad (6.19)$$

The expectations of the posterior distributions that we obtained were underestimated as $\hat{\sigma}^q = 0.01236$ and $\hat{\sigma}^\gamma = 0.01178$, whereas the true values were $\sigma^q = \sigma^\gamma = 0.02$. One must remain careful, though, as these estimates seem to depend on the prior distributions of the standard deviations related to process noise and further investigation needs to be carried out. The difficulty to correctly estimate the standard deviations related to process noise might also owe to the fact that the observation noise are of much higher intensity than the process noise, whence identifiability problems for the process noise.

Still, our main objective in the presence of process noise remains an accurate estimation of the posterior distributions of the parameters, and these are barely affected by ill-chosen values of the process noise. On Figure 6.18, we compared the posterior distributions of the parameters obtained with either an underestimated value for the process noise $\sigma^q = \sigma^a = 0.01$ or their true values $\sigma^q = \sigma^a = 0.02$. It appears that the two posterior distributions resemble each other very closely and that highly overestimated standard deviations of the process noise does not prevent an accurate estimation of the posterior distribution.

Algorithm 13 Joint Bayesian estimation of the functional parameters, hidden states and noise parameters

Choose prior distributions for the functional parameters $p(\theta)$ (as described before) and inverse Gamma prior distributions for the noise variances:

$$(\sigma^{g^\ell})^2 \sim \mathcal{IG}(\alpha^{g^\ell}, \beta^{g^\ell}),$$

$$(\sigma^r)^2 \sim \mathcal{IG}(\alpha^r, \beta^r),$$

$$(\sigma^q)^2 \sim \mathcal{IG}(\alpha^q, \beta^q),$$

$$(\sigma^\gamma)^2 \sim \mathcal{IG}(\alpha^\gamma, \beta^\gamma)$$

Estimate functional parameters θ and hidden states $x_{1:N}$ given experimental data $y_{1 \rightarrow T}$, observation noise parameters $(\sigma^{g^\ell}, \sigma^g)$ and process noise parameters $(\sigma^q, \sigma^\gamma)$ using a PMMH-UKF sampler.

Update observation noise parameters as:

$$(\sigma^{g^\ell})^2 \sim \mathcal{IG} \left(\alpha^{g^\ell} + \frac{O}{2}, \beta^{g^\ell} + \frac{1}{2} \sum_{k=1}^O \frac{(\tilde{q}_{t_k}^{g^\ell} - q_{t_k}^{g^\ell})^2}{(q_{t_k}^{g^\ell})^2} \right)$$

$$(\sigma^r)^2 \sim \mathcal{IG} \left(\alpha^r + \frac{O}{2}, \beta^r + \frac{1}{2} \sum_{k=1}^O \frac{(\tilde{q}_{t_k}^r - q_{t_k}^r)^2}{(q_{t_k}^r)^2} \right)$$

and deduce the values $(\hat{\sigma}^{g^\ell}, \hat{\sigma}^r)$ to be used for the second PMMH algorithm.

Estimate functional parameters θ and hidden states $x_{1:N}$ given experimental data $y_{1 \rightarrow T}$, observation noise parameters $(\hat{\sigma}^{g^\ell}, \hat{\sigma}^g)$ and process noise parameters $(\sigma^q, \sigma^\gamma)$ using a PMMH-UKF sampler.

Update process noise parameters as:

$$(\sigma^q)^2 \sim \mathcal{IG} \left(\alpha^q + \frac{T}{2}, \beta^q + \frac{1}{2} \sum_{n=1}^T \frac{(q_n^{\text{sto}} - q_n^{\text{det}})^2}{(q_n^{\text{det}})^2} \right)$$

$$(\sigma^\gamma)^2 \sim \mathcal{IG} \left(\alpha^\gamma + \frac{T}{2}, \beta^\gamma + \frac{1}{2} \sum_{n=1}^T \frac{(\gamma_n^{\text{sto}} - \gamma_n^{\text{det}})^2}{(\gamma_n^{\text{det}})^2} \right)$$

Chapter 7

Image analysis

THE PREVIOUS CHAPTER AIMED AT HIGHLIGHTING THE VARIOUS ABILITIES of the ADJUSTIN' platform as far as parameter and state estimation of individuals are concerned through case studies for the different plant growth models. The final objective of this thesis remains to evidence the genotypic variability existing within a population of plants. Both a plant growth model for *Arabidopsis thaliana* (Chapter 2.4) within the mathematical framework of population models (Chapter 1.3) as well as estimation procedures of parameters of such models in a Bayesian hierarchical context (Chapter 4) have been introduced. We have laid all the foundations necessary for the estimation of the parameters of the GreenLab model in a population of plants, all that remains is a good data set. To obtain a significant population effect, one needs a large number of different individuals and for each of these individuals, in order to make the most of the benefits from the organ-scale model, one needs a sufficiently great amount of observations for several organs.

The obtention of such a data set has been made possible thanks to the Variation and Abiotic Stress Tolerance (VAST) laboratory of the French National Institute for Agricultural Research (INRA). Olivier Loudet¹ provided us with many time series of zenithal images of different *Arabidopsis thaliana* individuals obtained thanks to an automated phenotypic platform called the Phenoscope [Tisné et al., 2013]. As described in Chapter 2, a good compromise between mechanistic description of plant growth processes and the level of details in the data necessary for their parameterization is that of functional-structural plant models [Vos et al., 2009], which combines the ecophysiology of plant growth and its architectural development. One of their fundamental properties is that their parameterization does not rely on the same type of information as classical ecophysiological models: architectural traits have the property to integrate the whole history of plant functioning, and a large information, in the Fisher sense, on model parameters can be inferred from the observation of the architectural traits. The key point that we aim at taking advantage of is that architectural traits can potentially be measured efficiently by automatic image analysis in high-throughput phenotyping platforms such as the Phenoscope. These have recently gained increasing interest, both in fields [Araus and Cairns, 2014] and laboratories [Tisné et al., 2013], thanks to their capacity to automatically measure many

¹<http://www7.inra.fr/vast/>

morphological and physiological traits for a large number of plant genotypes in various environmental conditions. However, although these measurements are potentially very detailed in time, they usually concern integrative traits, such as masses, total leaf area or height, and are again classically analyzed with descriptive statistical (multifactorial) models [Granier and Vile, 2014]. More precisely, although the Phenoscope is able to capture images of *Arabidopsis thaliana* and has a daily estimate of the projected leaf area (PLA) for each individual, refined estimates for the area of every leaf in a given individual was required for an efficient calibration.

The main objective of this chapter is therefore to provide an image analysis methodology allowing to dynamically monitor surface areas of every individual leaf in *Arabidopsis thaliana* phenotypes. The Phenoscope platform that allowed to obtain the image series for many individuals from different genotypes is first described in Section 7.1. The image analysis methodology was calibrated on 4 individuals of different genotypes as a first step. This data is firstly analyzed in Section 7.2 as a preliminary step for the setting up of the image analysis methodology, as several traits of the output images are of particular importance for tracking. The full methodology for image analysis is then presented: it comprises two main steps, one concerns the segmentation of all the images in the time series of a given individual (Section 7.3), the other is the tracking part reconstructing the whole history of the different leaves for this individual (Section 7.4). The segmentation part has already been studied [Scharr et al., 2016], and the method we developed was inspired by Apelt et al. [2015]. However, most studies only consider static images and are not interested in the dynamic monitoring of leaf growth, which raises non-trivial problems in tracking. The results of the dynamic monitoring of individual leaf surface areas are presented for the 4 individuals of different genotypes in Section 7.5. With a successful calibration of the image analysis algorithm, the latter is applied in Section 7.6 to a new data set comprising several tens of individuals so as to obtain data for an entire population of plants. These results and further perspectives are discussed in Section 7.7.

7.1 Phenoscope

Images of *Arabidopsis thaliana* were acquired using the Phenoscope, an automated phenotyping platform, whose full description can be found in [Tisné et al., 2013]. Two Phenoscope tables can actually be seen side by side on Figure 7.1. It is made of an aluminum table on a steel structure and allows the simultaneous growth of 735 plants in individual pots that are displaced along guiding rails across the table to ensure that all plants are grown in the same environmental conditions on average. The Phenoscope comprises two stations: a watering station where each pot, when placed over it, is weighed and watered according to instructions with a specified nutrient solution, and an imaging station that captures zenithal images of the plant placed under the digital camera. The Phenoscope is equipped with its own image processing scripts, Phenospeed, that outputs images where the background and leaves from neighboring plants have been removed to keep only the main rosette with red, green and blue colour components. These images have height $m_1 = 1232$ pixels and width $m_2 = 1624$ pixels and 1cm^2 is considered equal to 28,900 pixels. Phenospeed automatically

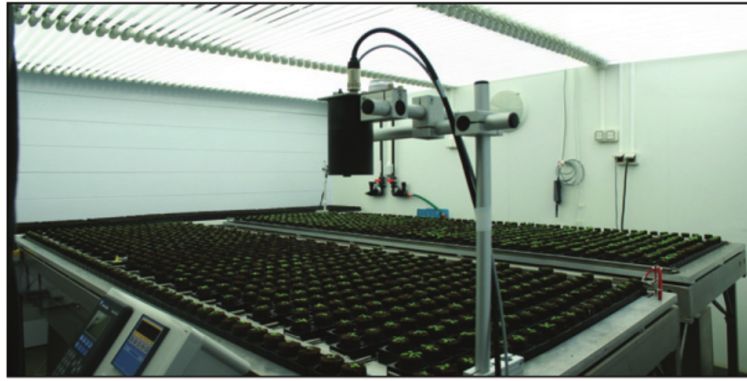


Figure 7.1: Two independent Phenoscope automated phenotyping platforms placed back to back in a 16 m² growth room. [Image from [Tisné et al., 2013]]

computes the total projected rosette area (in cm²). It cannot, however, compute the individual leaf areas necessary to exploit an organ-scale plant model.

The data set considered for assessing the performance of the image analysis procedure consists of a series of $T = 21$ images for one plant of each of the 4 genotypes Burren (Bur), Columbia (Col), Shahdara (Sha) and Tsushima (Tsu) denoted as the test individuals. The plants were all grown in the same environmental conditions. The photoperiod was of 8h, with a radiation of $350\mu\text{mol} \cdot \text{m}^{-2} \cdot \text{s}^{-1}$. The temperature was set to 21°C during the day and 18°C at night. The hygrometry was maintained constant at 65%RH. The series is composed of images taken on consecutive days from the 9th day after sowing (the day when the plants are installed on the robot) to the 29th day after sowing although, for the sake of clarity, the days of the image series will be identified from 1 to 21 in the following. On day 1 (from installation on the robot), the plants already have fully opened cotyledons (denoted as leaves 1 and 2). It should be noted that we have numbered leaves including the 2 cotyledons, so that the first true leaf is actually leaf 3. The full time series of images obtained for the Bur genotype is displayed on Figure 7.2 for the 21 days that lasted the experiment, and the genotypic variability among the 4 genotypes is highlighted on Figure 7.3 easily seen on the images.



Figure 7.2: Images output by the Phenoscope software on 21 different days for the Bur genotype. The images of the plant were obtained from the 1st day it was placed on the Phenoscope table, which takes place 8 days after sowing.



Figure 7.3: Comparison of the images obtained for the 4 different genotypes at 3 different times: $n = 1$ (left), $n = 11$ (middle) and $n = 21$ (right). In each panel, the Bur genotype is in the top left corner, the Col genotype in the top right corner, the Sha genotype in the bottom left corner and the Tsu genotype in the bottom right corner. For each day the images have been zoomed in equivalently between the genotypes.

7.2 Data analysis

To better understand the dynamics of the whole plant, a series of measurements was performed on each image for the 4 test individuals. Images are considered as elements of $\mathcal{M}_{m_1 m_2}(\mathbb{R})$. Each point P of an image can be defined as an ordered pair $P = (i, j) \in \mathcal{P}$, where $\mathcal{P} = \llbracket 1, m_1 \rrbracket \times \llbracket 1, m_2 \rrbracket$, representing its row and column. On day $n \in \llbracket 1, T \rrbracket$, an image has a set of ω_n visible leaves $L_n = \{\ell_{nu} | u \in \llbracket 1, \omega_n \rrbracket\}$, where $\ell_{nu} \subset \mathcal{P}$ is referred to as a leaf segment (or segment for short) and is a connected subset of the image $I_n \in \mathcal{M}_{m_1 m_2}(\mathbb{R})$. Over the whole timeline $\llbracket 1, T \rrbracket$, the plant has a set of N leaves $\mathcal{L} = \{L_v | v \in \llbracket 1, N \rrbracket\}$ indexed by their order of appearance. A particular leaf of the plant is therefore identified by as many occurrences as images in the series, $L_v = \{\ell_{nv} | n \in \llbracket 1, T \rrbracket, \ell_{nv} \in L_n \cup \emptyset\}$. A leaf can indeed have the empty set as a segment on certain days if its area is not available, because of overlappings for instance. The index u will therefore be reserved to segments, whereas the index v will be reserved to leaves. The leaf of rank v is the v -th to appear. Throughout this work, true segments, considered to be those manually extracted from the images, will be denoted ℓ_{nu} , while the segments found by the algorithm will be denoted $\tilde{\ell}_{nu}$. The same distinction applies to leaves. The problem can therefore be decomposed into two parts:

- segmentation: on each day $n \in \llbracket 1, T \rrbracket$, segment the image so as to find as many leaf segments as possible in \tilde{L}_n ;
- tracking: for each leaf of rank $v \in \llbracket 1, N \rrbracket$, for each day $n \in \llbracket 1, T \rrbracket$, find if there is an element of \tilde{L}_n susceptible to belong to \tilde{L}_v in order to reconstruct the whole history of the v -th leaf.

We will denote by $C \in \mathcal{P}$ the mass centre of the plant. The extremity of a segment ℓ_{nu} is defined as the furthest point from the mass centre of the plant, i.e.:

$$E_{nu} = \arg \max_{P \in \ell_{nu}} d(P, C) \quad (7.1)$$

where d is the Euclidean distance. This allows us to measure three variables for a given visible leaf segment ℓ_{nu} . Let $\overrightarrow{CE_{nu}}$ be the vector joining the mass centre of the plant to the segment extremity, then we define the maximum distance $e_{nu} = \|\overrightarrow{CE_{nu}}\|_1$ and the maximum angle $d_{nu} = (Oj, \overrightarrow{CE_{nu}})$ where Oj designates the horizontal axis. There are several ways to measure the angle of a segment, other possibilities would include to consider:

- the angle defined by the point minimizing the distance to the plant mass centre;
- the average angle of all points or the angle of the mass centre of the segment,

but these definitions are less stable and robust against overlappings. These two variables yield valuable information about the orientation and the distance of a given leaf throughout plant growth. A third variable is obviously leaf area, which was manually extracted from the images, potentially reconstructing the shape of the leaves partially hidden by others. It has to be noted that the insertion of the leaf was taken into account when extracting areas. They were manually acquired on all the images using Photoshop CS5, the Ruler tool for angles and distances, which allows easy measurements of distances and angles between two points as well as tab-delimited file export for post-processing, and the Eraser and the Magic Wand to isolate a segment and select all its pixels for the areas. The values obtained for the angles and leaf areas are displayed on Figure 7.4 and on Figure 7.5 respectively for each genotype.

We recall that each leaf is identified by its rank with a specific colour as specified in Table 7.1.
















rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
colour															

Table 7.1: Colours attributed to each leaf of a specific rank for easier identification.

7.2.1 Analysis of angles

As can be seen from Figure 7.4, the angle of a given leaf is not constant throughout the growth of the plant and there are two main reasons for this:

- there might be small displacements of the pot from day to day, both in translation and in rotation;
- leaves can be displaced or pushed by some others due to development competition.

In most cases, it is easy for a human observer to extract from all the points the occurrences of a given leaf, but sometimes it is very hard not to say impossible to choose between two points. For the Bur genotype, it suffices to consider the trajectories of the 2nd and 6th leaves that create a fork on day 8, or the 10th leaf whose trajectory overrides alternatively that of the 5th leaf and the 1st one. Similar scenarios can be found for the other genotypes.

Let d_{nv} denote the angle of the v -th leaf on day n , d_v^0 the angle of the v -th leaf on the first day it appeared and \bar{d}_v the angle of the v -th leaf averaged over all the days it exists. On day 1 on the robot, only the first two embryonic leaves (cotyledons) are visible. In fact, 4 leaves are already preformed in *Arabidopsis thaliana* but they might not be all visible from the very beginning of the image series. The first two leaves grow in opposite directions, i.e. $d_1^0 - d_2^0 \approx 180^\circ$. The 3rd and 4th leaves (the first true leaves) appear on the same day, more precisely on day 2 for Sha, on day 3 for Bur and Col, and on day 4 for Tsu. Similarly to the first two leaves, they grow in opposite directions such that $d_3^0 - d_4^0 \approx 180^\circ$. Furthermore, they grow in a direction very close to the bisector of (d_1^0, d_2^0) , i.e. for $i \in \{1, 2\}, j \in \{3, 4\}, |d_i^0 - d_j^0| > 40^\circ$, even though this might not be the case at the end of the growth because of competition, so that for $i \in \{1, 2\}, j \in \{3, 4\}, |\bar{d}_i - \bar{d}_j| > 40^\circ$ does not necessarily hold as can be seen for the Bur, Col and Sha genotypes. By convention and to distinguish

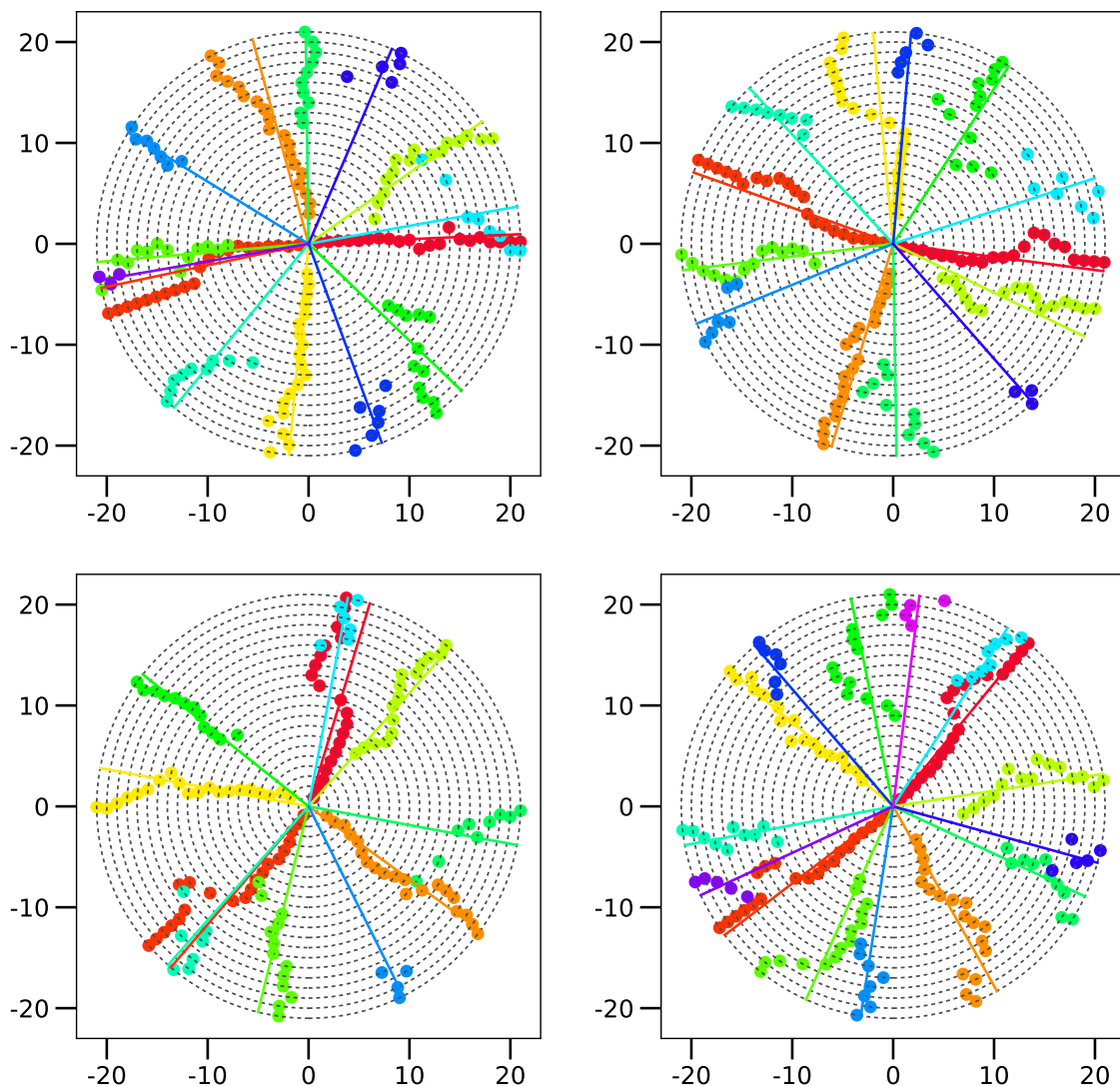


Figure 7.4: Evolution of the angles of the different leaves for genotypes Bur (top left), Col (top right), Sha (bottom left) and Tsu (bottom right). The angle α of the v -th leaf on day n is displayed on the circle centered in $(0, 0)$ and of radius n , i.e. have coordinates $(n \cos(\alpha), n \sin(\alpha))$. Straight lines indicate the mean angles for each leaf throughout their respective growth. Manually acquired data.

between the 1st and 2nd leaves on the one hand and the 3rd and 4th leaves on the other hand, the 1st and 3rd leaves are defined to have the closest averaged angle to that of the 5th leaf, that is $|\bar{d}_5 - \bar{d}_1| \leq |\bar{d}_5 - \bar{d}_2|$ and $|\bar{d}_5 - \bar{d}_3| \leq |\bar{d}_5 - \bar{d}_4|$.

The leaves appearing after the 4th one are not preformed and phyllotaxy underlies the direction of their growth. Phyllotaxy is a well-known phenomenon in *Arabidopsis thaliana* [Smith et al., 2006] which drives the growth direction of a leaf based on the growth direction of the leaf previously emerged. More precisely,

$$|d_{v+1}^0 - d_v^0| \approx d_p, \text{ where } d_p = 137.5^\circ \text{ is the golden angle.} \quad (7.2)$$

This phenomenon starts from $v = 4$ as it does not affect the preformed leaves and is either clockwise or counter-clockwise for a given plant. However this orientation cannot be predicted with certainty as it varies among plants: in this case study, it is counter-clockwise for the Bur individual and clockwise for the Col, Sha and Tsu individuals used here. The means and standard deviations of the difference of angles between two consecutive leaves from $v = 4$ are summarized in Table 7.2 for the 4 genotypes. This will be used in the classification algorithm to predict the direction of the leaves.

genotype	Bur	Col	Sha	Tsu
mean	136.52	136.01	136.15	136.12
std	6.91	17.92	15.27	10.32

Table 7.2: Mean and standard deviation of the phyllotaxy angle (in $^\circ$) for each genotype.

7.2.2 Analysis of areas

From the graphs of the areas on Figure 7.5, the growth behaviours of the different leaves appear to be similar to those of the angles: the 1st and 2nd leaves have an identical evolution, growing from day 1 to day 10 approximately, then reaching a plateau. The growth is initially linear. Alike, the 3rd and 4th leaves exhibit an identical behavior, appearing on the same day and with a growth curve resembling more a sigmoid than for the first two leaves. By the end of the series, they start to reach a plateau as well. From the 5th leaf, the leaves appear one by one, two leaves never having similar growth curves. The higher the rank of a leaf, the steeper its initial growth so that the area of the $v + 1$ -th leaf ends up (or would end up if the series were longer) to exceed that of the v -th leaf.

Since there is only one image per day, the phyllochron (that is to say the time interval between the appearance of two successive leaves) cannot be measured exactly, it is however obviously not the same across the different genotypes as can be seen from the number of emerged leaves on day 21: 14 for Bur, 13 for Col, 11 for Sha and 15 for Tsu. The phenotypic differences are well illustrated by, for instance, the area of the 7-th leaf which,

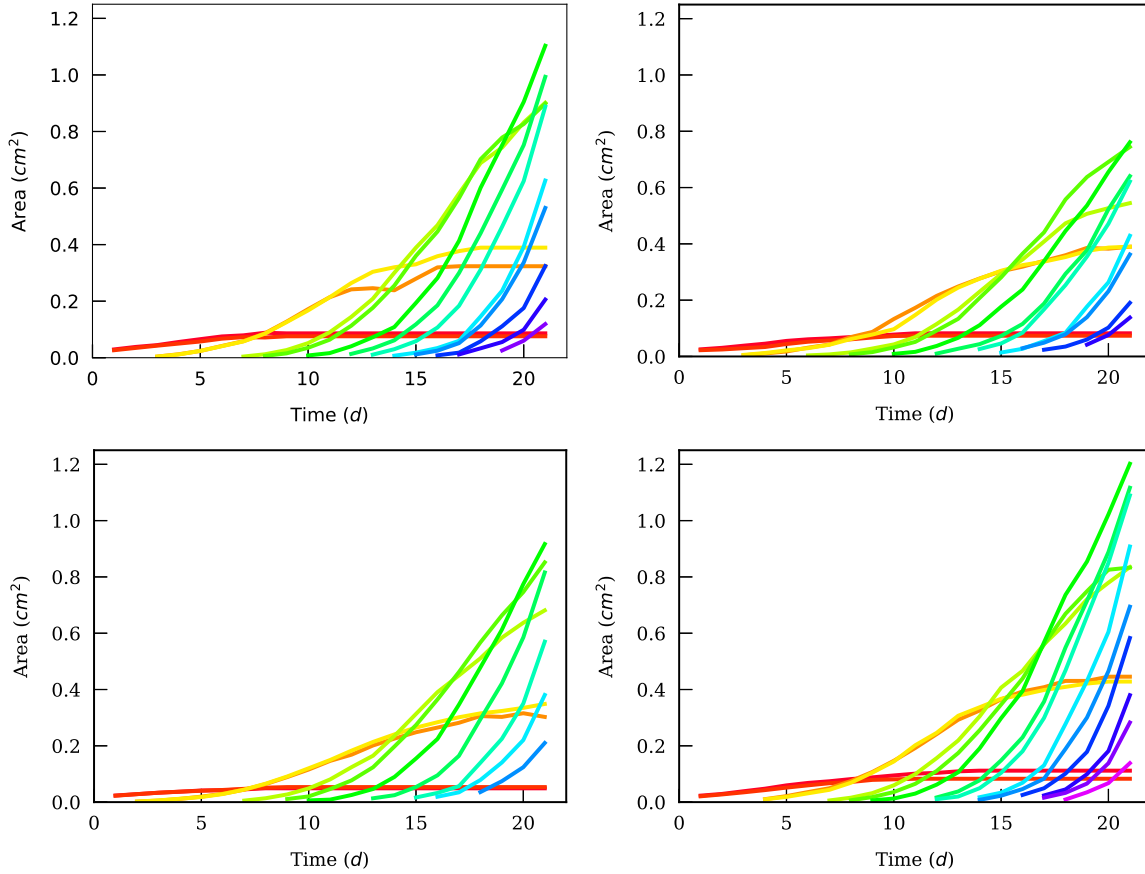


Figure 7.5: Evolution of the areas of the different leaves (in cm^2) with respect to time (in days) for genotypes Bur (top left), Col (top right), Sha (bottom left) and Tsu (bottom right). Manually acquired data.

on day 21, varies greatly among genotypes: 1.10cm^2 for Bur, 0.76cm^2 for Col, 0.92cm^2 for Sha, 1.20cm^2 for Tsu.

7.3 Segmentation

The objective of this part is to search for all possible segments of leaves and their corresponding areas on each image of the series. The segmentation problem has been approached in various ways, with recent contributions using ellipsoid leaf-shape models [Aksoy et al., 2015], Gaussian process shape models under a Bayesian approach [Simek and Barnard, 2015] or machine learning [Pape and Klukas, 2015]. This segmentation is achieved via a two-step algorithm for each day n :

- during the first step a segmentation routine, based notably on a watershed transformation, is applied and returns a first set of estimated segments $\tilde{L}_n^1 = \{\tilde{\ell}_{nu}^1 | u \in \llbracket 1, \tilde{\omega}_n \rrbracket\}$,
- during the second step, these estimates are refined by assessing the shape of the segments and comparing them to the usual ellipsoid shapes of leaves to obtain the final set of estimated segments $\tilde{L}_n^2 = \{\tilde{\ell}_{nu}^2 | u \in \llbracket 1, \tilde{\omega}_n \rrbracket\}$.

The segmentation step can be rendered difficult because of several aspects: some leaves might recover some others, making them difficult to detect, or can be bent, mainly because of their weights, thus not having the usual shape of a leaf. The approach undertaken here was inspired from [Apelt et al., 2015]. The image to be segmented on day n is denoted as I_n . First and foremost, the images are converted to grayscale via the usual transformation:

$$I_n^{\text{gray}}(i, j) = 0.2989 \times I_n^{\text{red}}(i, j) + 0.5870 \times I_n^{\text{green}}(i, j) + 0.1140 \times I_n^{\text{blue}}(i, j) \quad (7.3)$$

where I_n^{red} , I_n^{green} and I_n^{blue} denote the red, green and blue channels of image I_n respectively. Some of the images provided by the Phenoscope software still contained objects from the background, connected-segment labeling was hence first used to discard such objects from I_n^{gray} not belonging to the plant, considered to be the main connected subset. The mass centre of the plant is then computed, as it constitutes, once artifacts have been removed, a very good approximation of the stem location from where the leaves grow. Its coordinates (i_c, j_c) are given by:

$$\begin{cases} i_c &= (\sigma^{\text{gray}})^{-1} \sum_i \sum_j i I_n^{\text{gray}}(i, j), \\ j_c &= (\sigma^{\text{gray}})^{-1} \sum_i \sum_j j I_n^{\text{gray}}(i, j), \end{cases} \quad (7.4)$$

where $\sigma^{\text{gray}} = \sum_i \sum_j I_n^{\text{gray}}(i, j)$. We then move on to compute the radial profile that is used for the assessment of the distance of each point in the image to the mass centre:

$$I_n^{\text{rad}}(i, j) = (i - i_c)^2 + (j - j_c)^2. \quad (7.5)$$

Then a Canny edge detection filter [Canny, 1986] is applied to help detect strongly overlapping leaves to obtain I_n^{Canny} . The Euclidean distance transform of the plant can then be computed, it is defined as:

$$I_n^{\text{EDT}}(i, j) = \min_{(x, y) | I_n^{\text{Canny}}(x, y) = 0} (i - x)^2 + (j - y)^2 \quad (7.6)$$

and its local maxima are determined, as they are the points the furthest from the background, and are therefore likely to correspond to mass centres of leaves. The Euclidean distance transform and the radial profile are combined to obtain an image corresponding to a topographic relief suitable for a watershed segmentation. In essence the latter is a rather intuitive algorithm:

- it is initialized as if one progressively filled the basins (local minima) of the relief with water,
- when two basins meet, if their labels are identical, they are merged into one, otherwise a barrier is drawn between them so as to separates different segments.

This last operation returns the first set of connected segments susceptible to be leaves $\tilde{L}_n^1 = \{\tilde{\ell}_{nu}^1 | u \in \llbracket 1, \tilde{\omega}_n \rrbracket\}$. The image processing operations were all performed using Python 3.4.3, and the library scikit-image 0.12.3. An example of all the transformations applied to a particular image is presented in Figure 7.6.

This first step of the segmentation returns a set of segments which is different from the true set of segments found manually $L_n = \{\ell_{nu} | u \in \llbracket 1, \omega_n \rrbracket\}$. The main difficulty of segmenting the leaves of a plant classically owes to the fact that some leaves might partially overlap some others, hence leading to segments of the resulting image to be (i) either only parts of a leaf (ii) or several distinct leaves merged. To assess if a segment

$\tilde{\ell}_{nu}$ can be considered a true leaf segment, we define the ratio:

$$i_{nu} = \frac{A(H(B(\tilde{\ell}_{nu})))}{A(B(\tilde{\ell}_{nu}))} \quad (7.7)$$

where:

- $B : \tilde{L}_n^1 \rightarrow \mathcal{M}_{m_1 m_2}(\{0, 1\})$ transforms a segment into a binary image,
- $H : \mathcal{M}_{m_1 m_2}(\{0, 1\}) \rightarrow \mathcal{M}_{m_1 m_2}(\{0, 1\})$ denotes the convex hull operation,
- $A : \mathcal{M}_{m_1 m_2}(\{0, 1\}) \rightarrow \mathbb{R}$ gives the area of a segment.

If this ratio is greater than a given threshold, the segment is considered to be a leaf, i.e.:

$$\tilde{\ell}_{nu} \text{ is a leaf segment if } i_{nu} > i_0. \quad (7.8)$$

A value $i_0 = 0.9$ was retained throughout this work. In practice, the result of the first segmentation step is sometimes unable to discriminate between several leaves, grouping them into one single segment. To refine the segmentation, we used the following approach: if $\tilde{\ell}_{nu}$ is not a leaf segment, the local maxima M_{nu} of the Euclidean distance transform of $\tilde{\ell}_{nu}$ and the points achieving these maxima I_{nu} are computed:

$$\begin{cases} I_{nu} &= \{P \in \tilde{\ell}_{nu} \mid E_{\tilde{\ell}_{nu}}(P) \text{ is a local maximum of } E_{\tilde{\ell}_{nu}}\}, \\ M_{nu} &= \{E_{\tilde{\ell}_{nu}}(P) \mid P \in I_{nu}\}, \end{cases} \quad (7.9)$$

where $E_{\tilde{\ell}_{nu}} = E(B(\tilde{\ell}_{nu}))$ and $E : \mathcal{M}_{m_1 m_2}(\{0, 1\}) \rightarrow \mathcal{M}_{m_1 m_2}(\mathbb{R})$ denotes the Euclidean distance transform operation. We then define the greatest two maxima, likely to be mass centres of leaves:

$$\begin{cases} z_1 &= \max\{z \in M_{nu}\}, \\ z_2 &= \max\{z \in M_{nu} \mid z \neq z_1\} \end{cases} \quad (7.10)$$

their corresponding coordinates P_1 and P_2 , and two ellipses E_1 and E_2 with respective centres P_1 and P_2 , minor semi-axes a_1 and a_2 and major semi-axes b_1 and b_2 , where:

$$\begin{cases} a_k &= \phi \min_{P \in (CE_k)} d(P, E_k), \\ b_k &= \phi \min_{P \in (PE_k) \perp (CE_k)} d(P, E_k), \end{cases} \quad (7.11)$$

where $\phi = 1.05 > 1$ is defined so as to embrace the whole leaf segment. Two new segments are thus computed, $\tilde{\ell}_{nu}^{\cap} = \tilde{\ell}_{nu} \cap E_1$ and $\tilde{\ell}_{nu}^{\setminus} = \tilde{\ell}_{nu} \setminus \tilde{\ell}_{nu}^{\cap}$ and tested to be leaves again in a recursive manner. The results of this refinement step for a given image are illustrated on Figure 7.8.

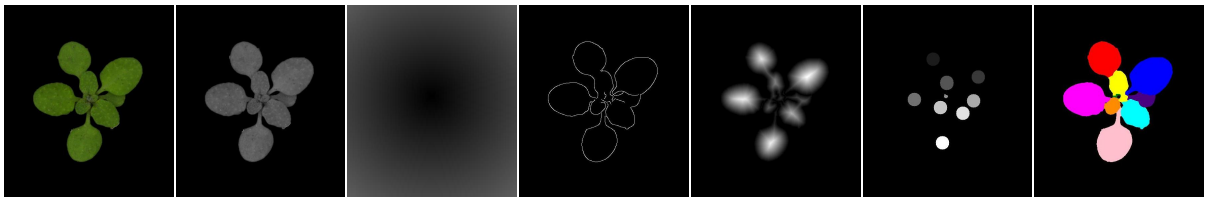


Figure 7.6: Series of some of the image transformation used during the first segmentation step. (a) Original image. (b) Gray scale image. (c) Radial profile. (d) Canny edge filter. (e) Euclidean distance transformation. (f) Local maxima of the EDT as starting basins of the watershed segmentation. (g) Results of the watershed segmentation.

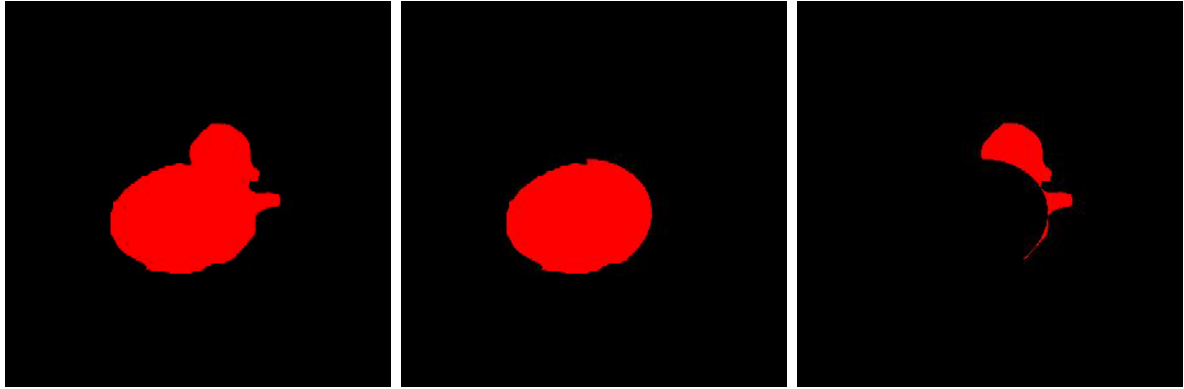


Figure 7.7: The ellipse refinement of the second segmentation step: when a segment $\tilde{\ell}_{nu}$ (a) originating from the first segmentation step is judged not to be a true leaf (in this case because two leaves are merged into the segment), it is decomposed into two new segments $\tilde{\ell}_{nu}^{\cap}$ (b) and $\tilde{\ell}_{nu}^{\setminus}$ (c).

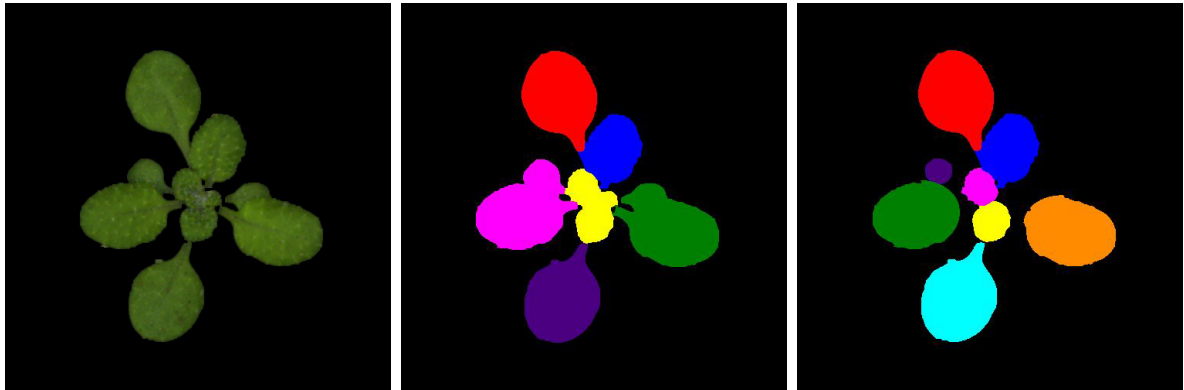


Figure 7.8: Example of images output by the Phenoscope (a) and after the first (b) and second (c) segmentation steps for the Col genotype on day 15.

7.4 Tracking

The segmentation step returns $\tilde{\mathcal{L}}$ such that:

$$\begin{cases} \tilde{\mathcal{L}} &= \{\tilde{L}_n | n \in \llbracket 1, T \rrbracket\}, \\ \tilde{L}_n &= \{\tilde{\ell}_{nu} | u \in \llbracket 1, \tilde{\omega}_n \rrbracket\}, \end{cases} \quad (7.12)$$

and the true set of leaves \mathcal{L} is such that:

$$\begin{cases} \mathcal{L} &= \{L_n | n \in \llbracket 1, T \rrbracket\}, \\ L_n &= \{\ell_{nu} | u \in \llbracket 1, \omega_n \rrbracket\}, \end{cases} \quad (7.13)$$

or equivalently:

$$\begin{cases} \mathcal{L} &= \{L_v | v \in \llbracket 1, N \rrbracket\}, \\ L_v &= \{\ell_{nv} | n \in \llbracket 1, T \rrbracket, \ell_{nv} \in L_n \cup \emptyset\}. \end{cases} \quad (7.14)$$

The objective of the tracking step is to assign each segment $\tilde{\ell}_{nu}$ found during the segmentation step to a leaf of a given rank so that $\tilde{\mathcal{L}} = \{\tilde{L}_v | v \in \llbracket 1, N \rrbracket\}$ with $\tilde{L}_v = \{\tilde{\ell}_{nv} | n \in \llbracket 1, T \rrbracket, \tilde{\ell}_{nv} \in \tilde{L}_n \cup \emptyset\}$ be as close as possible to \mathcal{L} . The data analysis helped us to better understand the growth dynamics of each leaf. The tracking of a

given leaf is based on its direction, its maximum distance and its area. To find the first occurrence of a leaf, only its direction and the day of possible first appearance are given. Let us recall that the first two leaves (cotyledons) always appear on day 1. Once an occurrence of the leaf has been found, it is searched on the following days according to this strategy: given a leaf of rank v whose k first segments have been tracked, $\tilde{L}_v = \{\tilde{\ell}_{nv} | n \in \llbracket 1, k \rrbracket\}$, the aim is to find the segment on day $k + 1$ the most probable to belong to the leaf of rank v . For each leaf $\tilde{\ell}_{k+1,u} \in \tilde{L}_{k+1}$, a score s_{kuv} is computed as:

$$s_{kuv} = \gamma_d s_{kuv}^d + \gamma_e s_{kuv}^e + \gamma_a s_{kuv}^a \quad (7.15)$$

where the superscripts d , e and a stand for direction, extremity and area respectively, $(\gamma_d, \gamma_e, \gamma_a) \in (\mathbb{R}^+)^3$ allows to weigh the different scores and:

$$\begin{cases} s_{kuv}^d &= 2\delta d f_{\mathcal{U}}(d_{k+1,u}, d_{kv} - \delta d, d_{kv} + \delta d) \\ s_{kuv}^e &= f_{\mathcal{N}}(e_{k+1,u}, \mu^e, (\sigma^e)^2) / \|f_{\mathcal{N}}(\cdot, \mu^e, (\sigma^e)^2)\|_{\infty} \\ s_{kuv}^a &= f_{\mathcal{N}}(a_{k+1,u}, \mu^a, (\sigma^a)^2) / \|f_{\mathcal{N}}(\cdot, \mu^a, (\sigma^a)^2)\|_{\infty} \end{cases} \quad (7.16)$$

where $f_{\mathcal{U}}$ and $f_{\mathcal{N}}$ are the pdfs of the uniform and normal distributions.

The d -score s_{kuv}^d favours the leaves that grow in the same direction of the last leaf segment $\tilde{\ell}_{k,v}$, with a tolerance of δd to account for pot rotations or competition of the growing leaves, as explained in Section 7.2.1. In practice, $\delta d = 30^\circ$. As can be seen from Figure 7.4, it does not seem useful to take into account the directions of the leaves $\tilde{\ell}_{nu}$ for $n < k$: the rotations being seemingly unpredictable, neither averaging nor interpolating seem of any help, and the last value is the one carrying the most information.

Data analysis of the areas and the distances of the extremities from the mass centre showed that their dynamics were sigmoid-like, which is why the means and standard deviations in the a -score and the e -score are obtained, whenever possible, by fitting a sigmoid using the last 4 segments of the leaf. Since segments might not be found on all images for a given leaf, the last 4 segments do not necessarily represent the last 4 days. More precisely, let $\{n_{k+1-i} | i \in \llbracket 1, 4 \rrbracket\}$ denote the days on which were registered the last 4 segments of the v -th leaf and let s be such that:

$$s_{\kappa}(x) = y_1 + \frac{y_0}{1 + \exp(-k(x - x_0))}, \quad \text{with } \kappa = (k, x_0, y_0, y_1). \quad (7.17)$$

For the prediction of the expected extremity on day $k + 1$, we define:

$$\begin{cases} \hat{\kappa}^e &= \arg \min_{\kappa} \sum_{i=1}^4 \|s_{\kappa}(n_{k+1-i}) - e_{n_{k+1-i},v}\|_2^2, \\ \mu^e &= s_{\hat{\kappa}^e}(k + 1). \end{cases} \quad (7.18)$$

All the same, for the prediction of the expected area:

$$\begin{cases} \hat{\kappa}^a &= \arg \min_{\kappa} \sum_{i=1}^4 \|s_{\kappa}(n_{k+1-i}) - a_{n_{k+1-i},v}\|_2^2, \\ \mu^a &= s_{\hat{\kappa}^a}(k + 1). \end{cases} \quad (7.19)$$

The standard deviations use only the last available value, $\sigma^e = e_{kv}/2$ and $\sigma^a = 2 a_{kv}$. When less than 4 occurrences are available, $\mu_e = 1.2 e_{kv}$ and $\mu_a = 2 a_{kv}$. Let \hat{u} be the segment index with the greatest score, $\hat{u} = \arg \max_u s_{kuv}$ and $\hat{s} = s_{k\hat{u}v}$ the corresponding score. The safety score s_0 is defined as a minimum score to achieve to be considered the next leaf segment: hence, if $\hat{s} > s_0$, the candidate segment $\tilde{\ell}_{k+1,\hat{u}}$ achieving

this best score is considered to be the segment of the leaf of rank v on day $k + 1$, and $\tilde{L}_v := \tilde{L}_v \cup \tilde{\ell}_{nu}$. In practice, typical values of the score weights and the safety score would be $(\gamma_d, \gamma_e, \gamma_a) = (10, 1, 1)$ and $s_0 = 11$, thereby prioritizing the orientation of a leaf over its length and area.

In order to take advantage of the phyllotaxy, the preformed leaves are first classified, the 1st and 2nd leaves first, in opposite directions, then the 3rd and 4th (the first 2 true leaves), in opposite directions and perpendicular to the first two. Classifying the 5th leaf will then yield the directions of the next leaves, which are the hardest to classify. The whole tracking strategy is summarized in Algorithm 14, where $d_p = 137.5^\circ$ is the phyllotaxy angle.

Algorithm 14 Classification strategy (leaves 1 and 2 are the cotyledons)

Track leaf 1 (randomly out of the two leaves found on day 1) for all days
 Look for leaf 2 in direction $d_1 + 180^\circ$ and track it
 Look for leaf 3 in directions such that $|d_3 - d_1| > 60^\circ$ and $|d_3 - d_2| > 60^\circ$ and track it
 Look for leaf 4 in direction $d_3 + 180^\circ$ and track it
 Track the 5th leaf to appear, whatever its growth direction
 Shuffle leaves 1 and 2 so that leaf 1 is closest to leaf 5 (convention)
 Shuffle leaves 3 and 4 so that leaf 3 is closest to leaf 5 (beginning of phyllotaxy)
For $j \geq 1$
 Look for leaf $5 + j$ in direction $d_5 + j \operatorname{sign}(d_5 - d_4) d_p$ and track it
End

7.5 Preliminary results on four individuals

The results for the first 8 leaves (including the 2 embryonic leaves) of the 4 test individuals are summarized on Figure 7.9, where the true results obtained with manual segmentation of the images, displayed as a continuous line, are compared to the results of the segmentation-tracking algorithm, displayed as filled circles. It has to be noted that the manual extraction of the leaf areas partly took into account the petiole, which is why the algorithmic results are on average slightly lower than the manual ones.

For the first two leaves (the cotyledons), no segments are found from day 15 – or even before sometimes – as these leaves are rapidly completely hidden by younger leaves. On some days, no segments were found: for instance, for the 6th leaf of the Bur genotype, the first segment is found on day 10 but from day 11 up to day 13, no segment is found. Such a situation arises because of overlappings and:

- either the segmentation step is unable to identify segments at all for this leaf on these days,
- or segments are found but they do not achieve a sufficient score to be considered as belonging to this leaf.

In the latter case, it is preferable to discard these segments so as not to introduce data that would be too noisy. Another scenario for missing data occurs when the growth curve of a leaf displays an unusual shape as is the case of the 4th leaf of the Col genotype, with a sharp increase after day 10. The predicted area

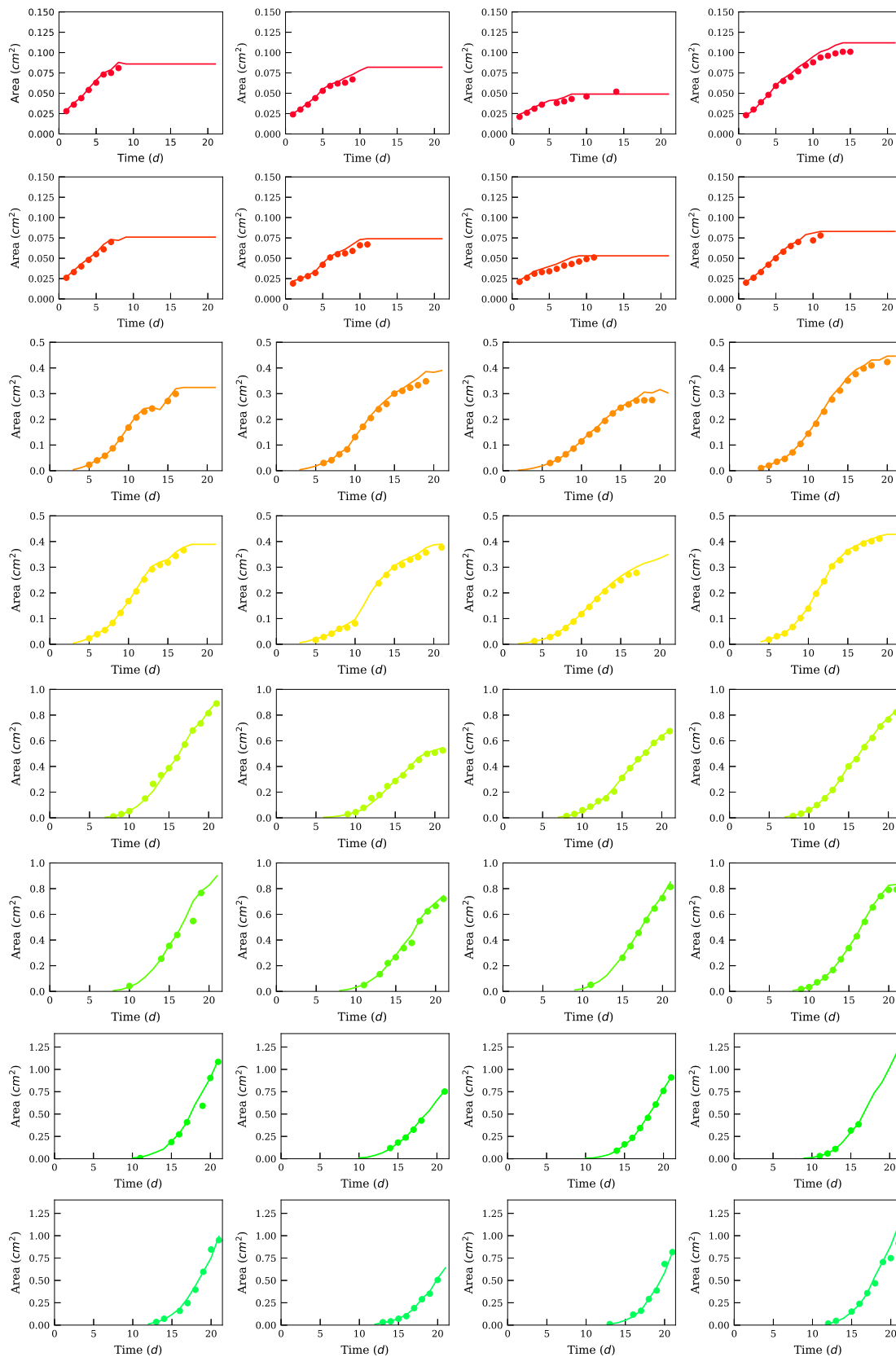


Figure 7.9: Results obtained manually (continuous line) and via the segmentation-classification approach (filled circles) for the first 8 leaves of the 4 genotypes (from left to right: Bur, Col, Sha, Tsu).

obtained by fitting a sigmoid using the last 4 segments of this leaf is therefore too low compared to that of the segment yielding an insufficient overall score to be accepted. In any case, this does not, fortunately, prevent the algorithm to find new segments on future days: starting from day 14 for the 6th leaf of the Bur genotype, or from day 13 for the 4th leaf of the Col genotype, therefore highlighting the robustness of the method towards missing data.

The evaluated leaf areas will be used to estimate the parameters of the GreenLab model presented in Section 2.4 by model inversion. For a given leaf, only conserving segments in which there is a very high level of confidence might seem overly cautious. However, since the architecture is in itself representative of the whole plant functioning, there is a lot of redundant information contained in leaf areas at close enough times. The daily image acquisition of the Phenoscope is thus in excess with respect to the time scale of *Arabidopsis thaliana* source-sink dynamics. Overall, all the accepted points have a very low error compared to the true values and the amount of very precise data this method yields for a given plant (more than 60 values for each genotype) will prove more than sufficient for an accurate parameter estimation of the model. To quantitatively assess the performance of the algorithm, the modelling efficiency and the accuracy (see Section 3.1.2) are inspected. Different such criteria were computed for completeness:

- one per individual leaf to account for the fact that the mean leaf area can be of different orders of magnitudes for different leaves,
- one per genotype for the different leaves,
- one per leaf for the different genotypes,
- and a global one taking into account all the data available.

The results are displayed in Tables 7.3 and 7.4. All the modelling efficiencies are greater than 0.93 except for the first two leaves of the Sha genotype, highlighting the overall excellent quality of the data extracted from the Phenoscope images for the leaf areas. It has to be noted that the accuracy is most often below 1, which means that the results are a bit underestimated on average. This is mostly due, as already mentioned, to taking into account the insertion of the leaf when extracting areas manually.

	1	2	3	4	5	6	7	8	all
Bur	0.9677	0.9617	0.9931	0.9951	0.9960	0.9379	0.9729	0.9801	0.9869
Col	0.9630	0.9455	0.9848	0.9942	0.9917	0.9873	0.9974	0.9941	0.9943
Sha	0.8231	0.7547	0.9830	0.9897	0.9979	0.9946	0.9986	0.9765	0.9955
Tsu	0.9588	0.9737	0.9944	0.9984	0.9992	0.9967	0.9958	0.9446	0.9874
all	0.9689	0.9430	0.9905	0.9955	0.9970	0.9857	0.9880	0.9683	0.9903

Table 7.3: Modelling efficiency for the different leaves and genotypes.

	1	2	3	4	5	6	7	8	all
Bur	0.9562	0.9524	0.9719	0.9668	1.0265	0.9852	0.8810	0.9832	0.9707
Col	0.9540	0.9329	0.9540	0.9374	1.0340	0.9629	0.9993	0.9838	0.9678
Sha	0.9249	0.8923	0.9679	0.9571	1.0149	0.9853	1.0000	1.0183	0.9685
Tsu	0.9551	0.9597	0.9554	0.9770	0.9846	0.9759	0.9988	0.9864	0.9712
all	0.9485	0.9318	0.9615	0.9598	1.0144	0.9759	0.9676	0.9919	0.9696

Table 7.4: Accuracy for the different leaves and genotypes.

Listing 13 PBS file for parallelization of the segmentation step

```
#!/bin/bash

#PBS -S /bin/bash
#PBS -l select=1:ncpus=24:mpiprocs=24
#PBS -l walltime=00:20:00
#PBS -J 1-21

module load anaconda3/4.3.1
time python segmentation.py 037 $PBS_ARRAY_INDEX
```

7.6 Results on a large data set

The results obtained in the previous section were very encouraging since they provided a lot of data for each leaf of each individual as evidenced by Figure 7.9. The estimated results were very close to the true values obtained manually, as can be seen from Tables 7.3 and 7.4 and our image analysis algorithm seems to work for several different individuals all belonging to different genotypes, which is of primary importance to deal with a larger population of plants. However, these results proved excellent for 4 individuals only, and there is no a priori guarantee that it will consistently work for all other individuals. Some refinements could be brought to the image analysis methodology as discussed in Section 7.7.

How will this algorithm behave with an entirely new data set for which there is no manually acquired data? First and foremost, one can obviously decide from a numerical point of view if the algorithm has failed by setting standard lower and upper bounds for the individual leaf area curves. With a very large amount of data there is, however, no way of being certain whether the results obtained for all individuals are relevant.

We obtained once again thanks to Olivier Loudet of the VAST laboratory a new data set comprising time series of 21 images for 64 individuals of *Arabidopsis thaliana* grown in the same experimental conditions as those described in Section 7.1. The 64 individuals belong to the five genotypes Burren (Bur), Columbia (Col), Cape Verde Islands (Cvi), Shahdara (Sha) and Tsushima (Tsu).

The image analysis algorithm has been implemented on the computer cluster of CentraleSupélec. Since the segmentation step can be done independently for all the $T = 21$ images of an individual, the algorithm has been parallelized rather easily using PBS job arrays. A node with 24 processes was reserved and on each of the 21 first processes was launched the segmentation routine for a particular day of the image series, see Listing 13.

Parallelization is particularly beneficial when dealing with this many data and it allowed to perform the entire segmentation step in $t_{\text{seg}} = 30\text{s}$ instead of $9\text{min}35\text{s}$ sequentially. The tracking step, however, because of its time-dependent nature, could not be parallelized, which is why the segmentation and the tracking algorithms needed to be run independently. The results of the segmentation step for a given day is a list of segments representing possible leaves. From a programming perspective, they are Python classes containing a lot of information on each segment such as distances, angles, area, actual position of the pixels belonging to the segments or values related to operations such as convex hull. In order to transmit this information to the tracking algorithm, an easy solution, though costly in terms of memory usage, was to convert these Python structures using the Python module `pickle` that allows to efficiently serialize such objects. The 21 sets of segments of a time series are then saved as `pickle` objects at the end of the segmentation step, which could amount to as much as 200MB of data. These objects are then loaded for the tracking step and immediately deleted so as to preserve disk usage. The tracking step takes approximately $t_{\text{cla}} = 40\text{s}$ per individual, thereby providing the approximate computing time required to process the entire data set of $N = 64$ individuals:

$$t_{\text{tot}} = N \times (t_{\text{seg}} + t_{\text{cla}}) \approx 64 \times (30 + 40) = 4,480\text{s} \approx 75\text{min} \quad (7.20)$$

which is very reasonable.

The results obtained for the 64 individuals were classified into 4 categories:

- **Category 1:** the algorithm worked very well and all the data obtained for each leaf have reasonable and sensible values, there is no need at all to discard data from the individuals of this category. It comprises 24 individuals;
- **Category 2:** the algorithm worked well but there are a few (between 1 and 4) absurd observations among all possible data obtained for the different leaves: for instance, the last data for the 6th and 7th leaves might be too large because segments have been wrongly allocated to these leaves on a particular day. These data could be filtered out but could also be kept without having much influence on the whole estimation as they represent a small percentage of all the data for a given individual. This category comprises 14 individuals;
- **Category 3:** the algorithm worked well for the major part but some leaves (between 1 and 4) among the 6th, 7th, 8th, 9th and 10th leaves are badly segmented and need to be removed entirely. This category comprises 10 individuals;
- **Category 4:** finally, there are individuals for which the algorithm simply failed, whether it be due to several plants growing together, pot rotations or a misdetection of the first occurrence of the 5th leaf. This category comprises 16 individuals.

We decided to focus first on the best individuals, belonging to category 1 and will reconsider individuals from the second and third categories in Section 8.2. The results obtained for these 24 individuals are displayed on Figure 7.10. Overall, the leaf area curves display reasonable dynamics: one can observe the standard behaviour of the first 2 leaves rapidly plateauing somewhere between 0.1cm^2 and 0.2cm^2 around day 10. There is no observation for these leaves afterwards since they are recovered by those of higher rank. As expected, the

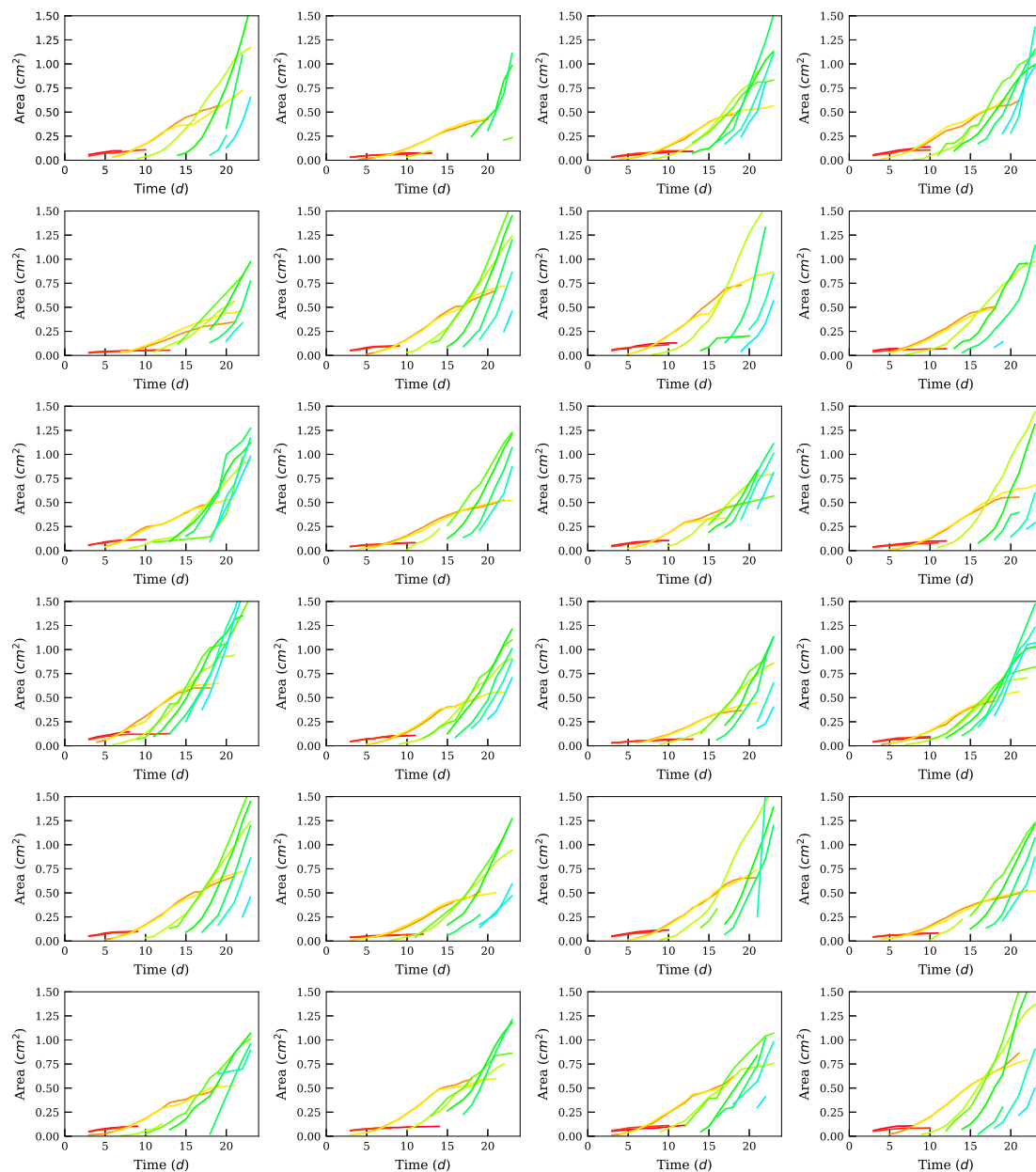


Figure 7.10: Graphs of the leaf area curves for 24 individuals of the data set of 64 individuals on which was automatically applied the image analysis algorithm. No data was acquired manually.

3rd and 4th leaves exhibit almost the same dynamics and their growth also starts to slow around day 15. Leaves of higher ranks then start to appear periodically. For several individuals, the tracking of some leaves ends early, probably because of the overcautious approach in discarding segments whose classification is not entirely certain.

7.7 Discussion

The Phenoscope, a high-throughput phenotyping platform, provided images of *Arabidopsis thaliana* for different genotypes grown in the same environmental conditions. The similarities and differences of some variables on these images for the different genotypes, such as orientation, distance from the mass centre or area of the different leaves, were highlighted. They helped build a two-step algorithm for leaf segmentation and tracking, allowing to reconstruct the whole history of the different leaves. The comparison of the results obtained for leaf area with the true results extracted from the images manually in the case of four plants belonging to different genotypes that exhibit diverse phenotypes showed that this procedure yields numerous and very precise data. This image processing routine was further tested on many more image series that the Phenoscope is able to provide [Tisné et al., 2013] for 5 genotypes with a dozen of individuals within each genotype. Although the results obtained for these series cannot be compared to true values, which would need countless hours of manual data extraction, the growth curves of most individuals seem very reasonable and encouraging. Having obtained such data for the different leaves of the plant makes it possible to use an organ-scale plant model such as the GreenLab model described in Section 2.4 to better understand the dynamics of leaf growth regulation and disentangle the effects of leaf growth and leaf emission rates [Tisné et al., 2010]. The experimental data obtained with the help of the segmentation-tracking algorithm can be used for parameter inference, notably in the context of population models. These results are indeed ideal for the use of the Bayesian hierarchical estimation procedures described in Chapter 4 and it is hoped that they will allow statistical testing of genotypic differentiation within *Arabidopsis thaliana* [Reymond et al., 2003].

Some technical difficulties have not been addressed yet, mostly by lack of time. For instance, pots containing plants might rotate throughout plant growth, changing the position of the mass centre and the direction of the leaves. Most of the time, these rotations are small in amplitude and the tracking algorithm handles them easily. However, in some cases, the pot might be rotated by 90° or even 180° because it has been displaced by an experimenter. In such cases, the current tracking algorithm will not be able to reconstruct the whole history of each leaf. We believe that such a difficulty can easily be overcome since, in this kind of scenarios, the algorithm will lose track of all leaves on the day $n \in \llbracket 1, T \rrbracket$ on which occurs this rotation. This can be easily detected and the segments of day n can be computed without taking into account the leaf orientation criterion. Hopefully, for several leaves, we will notice a change in their orientations between day $n - 1$ and n , which will provide the pot rotation angle.

Another problem that we encountered arose from the experiment procedure: in order to maximize the probability of plants growing, two plants might be planted in the same pot, leading to beginning of image series with two individuals on the image. The segmentation algorithm therefore fails to accurately compute the mass centre of the only plant that remains after a few days, the other one being removed by experimenters. Correctly interpreting the presence of two plants in the early stages of growth might not be this trivial, but again it is hoped that a preliminary step of computing the differences of consecutive images in the series might indicate when a plant is removed from the pot, say on day n , indicating that it must be ignored in the previous images of the series. The mass centre of the plant of interest can be accurately calculated on day n

and taken into account for the detection of leaves on this plant only for days before n .

Last but not least, one must mention the fact that the detection of the 5th leaf is a crucial step since it determines the direction of the next leaves. For a few series, the algorithm failed to properly detect the first occurrence of this 5th leaf and, although there were no false data retained because of the algorithm's prudence, a lot of leaf areas were missed. For these individuals, we input into the algorithm the day of first appearance and the orientation of the 5th leaf, allowing to obtain many more data for the late-emerging leaves. This was doable because it represented a small percentage of the image series, but it would be more convenient to not rely on such a manual step. All in all, this issue might only make us miss data for a few individuals and does not constitute a true pitfall to the overall image analysis procedure.

Another approach to obtain more data would be to not discard leaf areas that are regarded as exuberant because they imply a sudden change in the growth curve. Such unexpected changes in the growth curves can happen in real data and one could try to incorporate these possible changes in the GreenLab model. This problem was addressed by [Donnet et al. \[2010\]](#) who developed, in the context of repeated measurements of a continuous growth process over time in a population of individuals, a method taking into account unexpected changes in growth rates based on stochastic differential equations, the parameters of which being then inferred using Bayesian mixed effects models.

Environmental conditions such as altitude, rainfall or soil nutrient content influence leaf size [[McDonald et al., 2003](#)], [[Scoffoni et al., 2011](#)], but environmental influences, in particular light, can also influence leaf shape [[Tsukaya, 2005](#)], even though the underlying mechanisms are not yet fully understood. Therefore, it is all the more important to test whether more sophisticated shape models [[Herdiyeni et al., 2015](#)], [[Pape and Klukas, 2015](#)] could replace our relatively simple ellipse-based one and help estimate leaf areas more precisely or even increase the rate of leaf detection. Likewise, regarding leaf tracking, a promising approach would be to develop an iterative method in which, after a first step based on our original approach, the parameterized model thus obtained could be used instead of the sigmoid extrapolation to predict individual leaf areas at the next time step. In our original approach, there were cases in which leaf segments were discarded in the tracking step due to a low score resulting from a bad performance of the sigmoid growth function (for example because there were not enough segments of the corresponding leaf in the previous steps). Therefore, we expect that model-based predictions could significantly improve the number of detected segments for each leaf.

As underlined above, we used a very cautious approach and discarded data for which we did not get a very high level of confidence. Indeed, owing to the redundant information contained in the sequence of plant architectural descriptions, already pointed out in [[Godin et al., 1999](#)], we expected that it would not impact the quality of model identification. Our results seem to support this hypothesis. However, it would also be interesting to study more carefully the impact of the safety score s_0 on the rate of detection (both in terms of true positive rate and false positive rate), but also on the quality of model estimation. There are two aspects to this last question: how does the model estimation handle false data, and how does missing data impact the uncertainty in parameter estimates. Obviously, every refinement of the method that could help reduce such uncertainty will be profitable. Similarly, designing new data collection protocols for the Phenoscope that

are adapted to the model identification objective could also be considered (for example by measuring plant organ masses or using several individuals of the same genotype to take into account interindividual variability). Optimal experimental design methodology could help us for this purpose [[Pieruschka and Poorter, 2012](#)], [[Craufurd et al., 2013](#)].

Chapter 8

Estimation within population models: application

THIS CHAPTER IS DEVOTED TO THE ESTIMATION OF POPULATION PARAMETERS in the context of hierarchical models. One of the main purposes of this thesis was to evidence the genotypic differentiation within a population of plants. It was shown in Chapter 7 that the Phenoscope could provide an invaluable source of images for *Arabidopsis thaliana* populations. Coupled with an adequate image analysis algorithm, we were able to extract a great amount of organ-scale data for many individuals. Such a data set suits perfectly not only the mathematical framework of population models introduced in Chapter 1 but also the use of the Bayesian estimation algorithms developed in Chapter 4. We envision that the population approach and the estimation of the population mean vector and covariance matrix will be able to highlight the genotypic variability existing within a population of *Arabidopsis thaliana* used in the Phenoscope experiment. To this end, we use the ADJUSTIN' platform presented in Chapter 5 that was designed to tackle such estimation problems. We also highlight how the very nature of the Gibbs sampling scheme allows to efficiently parallelize algorithms and have efficient estimation procedures.

This application of the parameter estimation algorithms within population models on a real data set involves a lot of complex steps. First, the GreenLab model should reproduce correctly the dynamics of the real plant. It has been shown to be quite robust on individual estimations (see Section 6.1) even though some improvements could be made as far as the dynamics of the 3rd and 4th leaves are concerned. Second, the image analysis algorithm should obtain accurate estimates of leaf areas in many individuals and for many different genotypes, which amounts to many different phenotypes as shown on Figure 7.10. Even though it has been demonstrated in Section 7.5 that the preliminary results obtained on the 4 different individuals – belonging to 4 different genotypes – were excellent, and that the leaf area curves obtained for the new series of many individuals exhibit very promising shapes, there is no a priori guarantee that the results obtained on the new data set are as accurate as those obtained on the preliminary data set manually processed. Third, one must ensure that the parameter estimation algorithms are correctly implemented in the ADJUSTIN' platform, for there are a lot of steps and parameters involved, and that they effectively converge despite the

theoretical guarantees. For all these reasons, we will first investigate the case of synthetic data simulated directly from the GreenLab model. This ensures that there is no uncertainty related to either the plant model or the image analysis algorithm and that the Bayesian parameter estimation method effectively works.

8.1 Simulated data

Data were simulated using the GreenLab model presented in Section 2.4. The control variables were taken to be the same as for the experimental conditions described in Section 7.1. The number of hours per day was set to $n_s = 8\text{h}$, and the daily temperature to $t_n = 21^\circ\text{C}$. The photosynthetically active radiation was taken to be $r_n = 2.52 \cdot 10^{-5} \text{MJ} \cdot \text{cm}^2 \cdot \text{h}^{-1}$. Each leaf was attributed a specific time of appearance: the first two leaves appeared at $n = 1\text{h}$, the third and fourth leaves appeared at $n = 24\text{h}$. The time of appearance for the fifth leaf was set to $n_5 = 40\text{h}$. After that, the $5 + i$ -th leaf, for $i > 1$, appears at time $n_{5+i} = n_5 + i \phi$, where $\phi = 12\text{h}$ is the phyllochron. We focus on the estimation of the parameters $\theta = (e, \mu_1, \mu_2)$, even though more parameters will be estimated in Sections 8.1.8 and 8.2.

8.1.1 Simulation of synthetic data

The dimension of the estimation problem is therefore $d = 3$. All the other parameters were fixed at constant values throughout both the simulation of data and the estimation procedure. These values were:

$$\left\{ \begin{array}{l} \mu = 3.150 \cdot 10^{+00}, \\ s = 5.000 \cdot 10^{+00}, \\ k = 7.000 \cdot 10^{-01}, \\ \sigma_1 = 4.593 \cdot 10^{-01}, \\ \sigma_2 = 3.991 \cdot 10^{-01}, \\ \rho_2 = 7.801 \cdot 10^{-02}, \\ q_0 = 3.807 \cdot 10^{-05}. \end{array} \right. \quad (8.1)$$

The parameters varying in the population are given true values for their mean and covariance matrix. In the present case:

$$\eta^{\text{true}} = \begin{pmatrix} 1.558 \cdot 10^{-03} \\ 4.531 \cdot 10^{+00} \\ 5.390 \cdot 10^{+00} \end{pmatrix} \quad (8.2)$$

and:

$$\Sigma^{\text{true}} = \begin{pmatrix} 6.069 \cdot 10^{-09} & 0 & 0 \\ 0 & 5.131 \cdot 10^{-02} & 0 \\ 0 & 0 & 7.263 \cdot 10^{-02} \end{pmatrix} \quad (8.3)$$

and $\tau^{\text{true}} = 100$. The choice of the covariance matrix actually corresponds to $\Sigma^{\text{true}} = \text{diag}\{(\eta^{\text{true}}/20)^2\}$ and that of the precision amounts to a multiplicative normal observation noise with standard deviation $\sigma^{\text{true}} = (\tau^{\text{true}})^{-1/2} = 0.1$. Data were simulated for 24 individuals: this was done on a single process and then broadcasted to all others, the underlying motivation being that all processes obviously need to have the same experimental data for the different individuals. For the i -th individual, a set of true individual parameters was first sampled from the true population distribution:

$$\theta_i^{\text{true}} \sim \mathcal{N}(\eta^{\text{true}}, \Sigma^{\text{true}}) \quad (8.4)$$

and then used for the simulation of true leaf areas $(a_{i,n}^v)_{1:\nu_{\max}, 1:n_s:T}$ (hidden states) and their corresponding noised values $(\tilde{a}_{i,n}^v)_{1:\nu_{\max}, 1:n_s:T}$ (observations) where:

$$\tilde{a}_{i,n}^v = a_{i,n}^v (1 + \xi_{v,n}) \quad \text{with} \quad \xi_{v,n}^v \sim \mathcal{N}\left(0, (\tau^{\text{true}})^{-1}\right). \quad (8.5)$$

The areas of all leaves were observed on every day for 21 days which, taking into account that the phyllochron ϕ is fixed and given the rate of appearance of each leaf, amounts to observing 16 leaves over the whole growth and a number of observations of $n_i = 136$ for each individual i , i.e. a total number of observations of $n_{\text{tot}} = \sum_{i=1}^N n_i = 3264$. We denote by $\tau_i^v = (t_{i,1}^v, \dots, t_{i,n_i}^v)$ the timeline for the v -th leaf and by $\tilde{\alpha}_i^v = (\tilde{a}_{i,1}^v, \dots, \tilde{a}_{i,n_i}^v)$ the corresponding observations on leaf area at these time steps. The vector of all concatenated observations thus reads:

$$y_i = (\tilde{\alpha}_i^1, \dots, \tilde{\alpha}_i^{\nu_{\max}}) \in \mathbb{R}^{n_i} \quad \text{with} \quad n_i = \sum_{v=1}^{\nu_{\max}} n_i^v. \quad (8.6)$$

The calculations were performed on as many processes as individuals, i.e. on $n_p = 24$ processes. For reproducibility issues, the seed of the random number generator was fixed so that the same data was generated all the time. However, for the estimation part and as soon as the experimental data is loaded on the different processes, the seed is reset to a random value on each process.

The flattened vector of experimental data for all individuals is denoted as $Y = (y_1, \dots, y_N) \in \mathbb{R}^{n_{\text{tot}}}$. Examples of simulated data is displayed on Figures 8.1 and 8.2. On Figure 8.1, the leaf areas for each of the first eight leaves are displayed for the 24 individuals and on Figure 8.2 are displayed some examples of typical growth curves for the first eight leaves of several individuals. Despite the rather low population covariance used, there is still a high variability in the leaf areas. This is easily seen from Figure 8.2 when comparing, for instance, the 6th and 8th individuals, the latter having maximum leaf areas twice as high as the former.

8.1.2 Initialization of the prior distributions

Defining the prior distributions of the population parameters is of crucial importance. A first step consists in obtaining realistic values for the individual parameters θ_i . This can be done rather easily by performing a GLS procedure on each individual i . These first sets of parameters can be used to compute reasonable estimates of the population mean vector η and covariance matrix Σ . A first estimate of the individual set is therefore independently computed as:

$$\hat{\theta}_i^{\text{GLS}} = \arg \min_{\theta \in \Theta} (Y_i - h_i(\theta))^T \Sigma_i^{-1} (Y_i - h_i(\theta)), \quad (8.7)$$

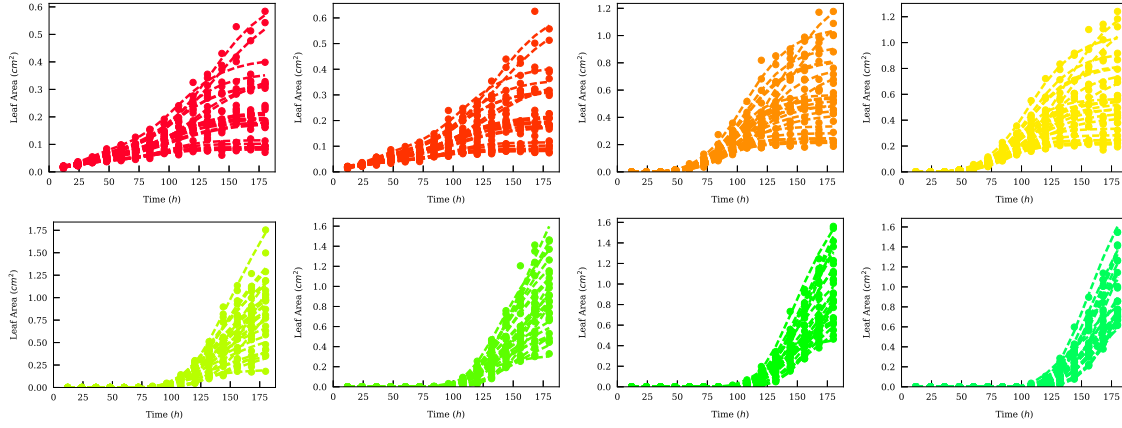


Figure 8.1: Leaf area variability for the first eight leaves. Each dashed line represents the true leaf area (hidden state) of a given individual and filled circles represent the corresponding noised data (observations).

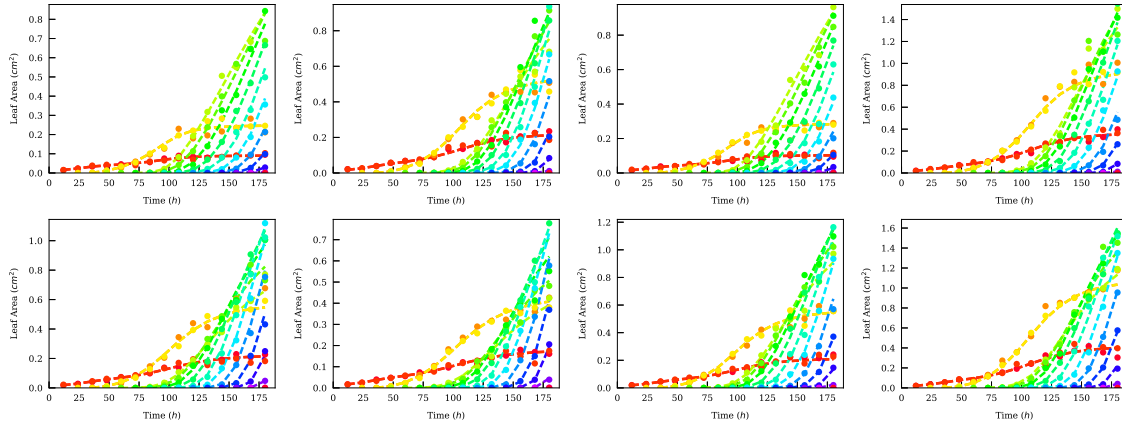


Figure 8.2: Leaf area for the first eight leaves for 8 different individuals. The hidden states are represented by dashed lines and observations by filled circles.

	e	μ_1	μ_2
$\mathbb{E}_i(\delta_i)$	$3.661 \cdot 10^{-03}$	$2.333 \cdot 10^{-03}$	$8.618 \cdot 10^{-04}$
$\max_i(\delta_i)$	$9.969 \cdot 10^{-03}$	$7.328 \cdot 10^{-03}$	$2.189 \cdot 10^{-03}$

Table 8.1: Average and maximum value of the relative errors for all individual parameters $(\theta_i)_{i \in \llbracket 1, N \rrbracket}$.

where $\Sigma_i = \text{blockdiag}_{v \in \llbracket 1, \nu_{\max} \rrbracket} \{ \mathbb{V}(\tilde{\alpha}_i^v) I_{n_i^v} \}$ is the heteroskedastic matrix of the variances for each leaf, and a Gauss–Newton optimization procedure with a maximum of 1,000 iterations was used. In practice, a couple of tens iterations suffice to reach a local minimum. To assess their relevance, one can simulate the hidden states from the GreenLab model using the GLS estimates $h(\hat{\theta}_i^{\text{GLS}})$ and these results can further be compared to the hidden states that would have been obtained without a GLS algorithm; such graphs are displayed on Figure 8.3. Since the true values of the parameters are known, it is possible to compare θ_i^{true} to $\hat{\theta}_i^{\text{GLS}}$. Relative errors were calculated for each parameter and each individual. The mean and maximum values of $(\delta_i^e)_{i \in \llbracket 1, N \rrbracket}$, $(\delta_i^{\mu_1})_{i \in \llbracket 1, N \rrbracket}$ and $(\delta_i^{\mu_2})_{i \in \llbracket 1, N \rrbracket}$ were calculated to illustrate the goodness of the GLS estimates.

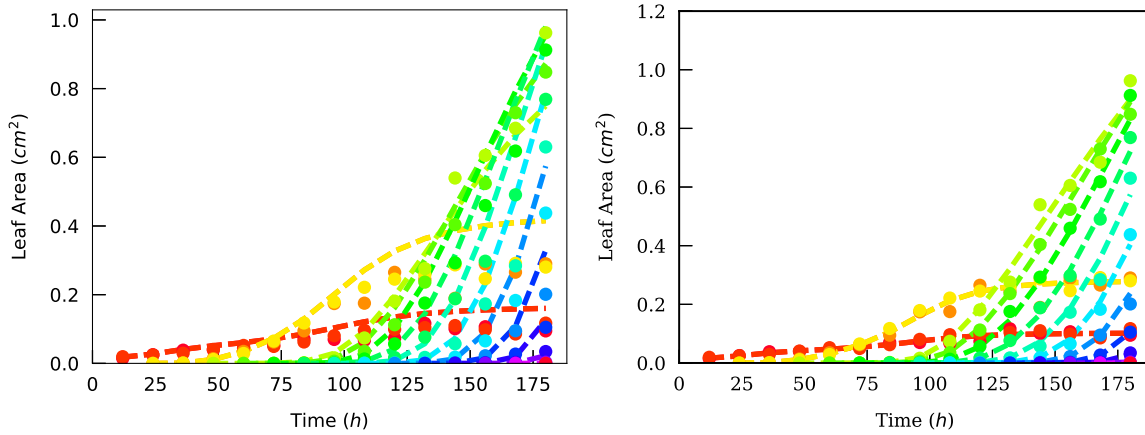


Figure 8.3: Example of GLS calibration for one particular individual. Hidden states (dashed lines) obtained via model simulation with the initial set of parameters (left) and with the GLS estimates (right) vs. observations (filled circles).

The individual parameters are therefore very precisely estimated. One must mention that this excellent estimation is partly helped by the fact that we considered simulated data and that the number of estimated parameters remains low. It nonetheless motivates the choice of such a strategy in the case of real data. The growth curves are then much better fitted after the GLS estimation than before as can be seen from Figure 8.3. This first crude step gives us a reasonable range of values that the parameters of GreenLab model for *Arabidopsis thaliana* can take and helps us define sensible prior distributions: even though they indirectly integrate observations, they remain slightly non-informative. Notably, the mean of the individual estimates can be used for the population mean prior, and the covariance matrix of the multivariate normal distribution for the population mean vector is chosen so that the standard deviations related to the different parameters are a fifth of their mean, i.e.:

$$\lambda = \hat{\theta}^{\text{GLS}} = \frac{1}{N} \sum_{i=1}^N \hat{\theta}_i^{\text{GLS}} \quad \text{and} \quad \Lambda = \text{diag}\{(\lambda/5)^2\}. \quad (8.8)$$

The covariance of the individual estimates can also be used for the initialization of the population inverse covariance matrix Σ^{-1} . We recall from Section 4.2.2 that the prior distribution for the population covariance matrix is an inverse Wishart distribution, i.e. $\Sigma \sim \mathcal{IW}(q, \Psi^{-1})$. Besides, the mean of the inverse Wishart distribution implies that:

$$\mathbb{E}(\Sigma) = (q - d - 1)^{-1} \Psi^{-1}. \quad (8.9)$$

We can therefore choose the hyperparameters q and Ψ as follows:

$$q = d + 3 \quad \text{and} \quad \Psi = \left(2 (q - d - 1) \mathbb{C} \left(\hat{\theta}_i^{\text{GLS}} \right) \right)^{-1}. \quad (8.10)$$

to obtain that $\mathbb{E}(\Sigma) = 2 \mathbb{C} \left(\hat{\theta}_i^{\text{GLS}} \right)$. Such choices of q and Ψ ensures that the initial population covariance matrices used in the Gibbs sampler will be, initially, larger than the covariance based on the individual estimates.

Finally, the hyperparameters α and β for the population precision $\tau \sim \mathcal{G}(\alpha, \beta)$ are chosen such that the mean and standard deviation of the prior distribution are $m = 9 \tau^{\text{true}}/10$ and $s = \tau^{\text{true}}/3$ respectively. We

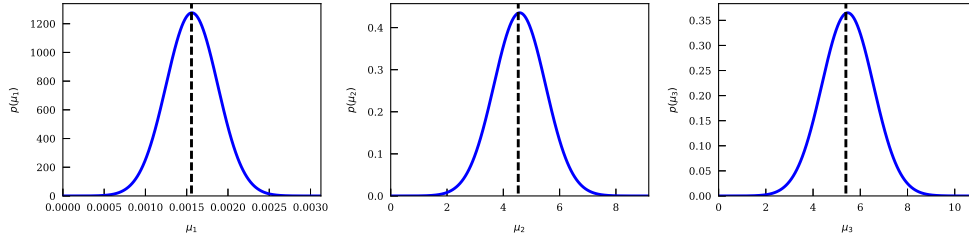


Figure 8.4: Prior distribution for the population mean vector components. The true values are represented by black dashed lines.

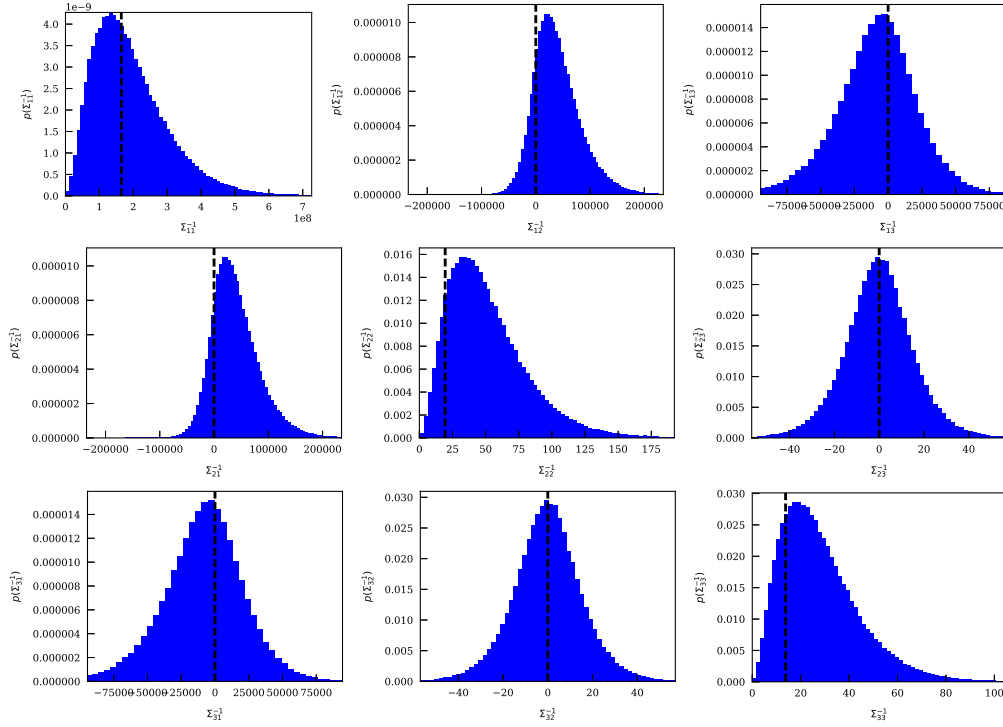


Figure 8.5: Prior distribution for the population inverse covariance matrix components. The true values are represented by black dashed lines.

recall from Section 4.2.2 that the hyperparameters of the Gamma distribution α and β can be expressed as functions of the mean and standard deviation m and s :

$$\begin{cases} \alpha = \frac{m^2}{s^2}, \\ \beta = \frac{m}{s^2}, \end{cases} \quad (8.11)$$

which leads us in the present case to choosing the following values for the hyperparameters of the Gamma distribution:

$$\alpha = \frac{(9 \tau^{\text{true}}/10)^2}{(\tau^{\text{true}}/3)^2} = 7.29 \quad \text{and} \quad \beta = \frac{9 \tau^{\text{true}}/10}{(\tau^{\text{true}}/3)^2} = 0.081. \quad (8.12)$$

The first calibration step using a GLS procedure hence allows to initialize reasonable prior distributions, as pictured on Figures 8.4 and 8.5. The mean of the prior distribution for η , λ , is already very close to the true values. The modes of the prior distributions for Σ^{-1} , are also near the true values, even though the diagonal components Σ_{22}^{-1} and Σ_{33}^{-1} are slightly overestimated.

8.1.3 Details of implementation

Initial values for the population parameters are then sampled from these prior distributions:

$$\begin{cases} \eta^0 & \sim \mathcal{N}(\lambda, \Lambda), \\ \Sigma^{-1,0} & \sim \mathcal{W}(q, \Psi), \\ \tau^0 & \sim \mathcal{G}(\alpha, \beta). \end{cases} \quad (8.13)$$

The Gibbs sampling scheme was performed for $M = 100,000$ iterations, in the sense that at each iteration all individual parameters $(\theta_i)_{i \in \llbracket 1, N \rrbracket}$ and the population parameters η , Σ and τ are updated. A burn-in period of $B = M/5 = 20,000$ iterations was set, so that all samples up to iteration B are discarded from the posterior distributions. For each individual, an adaptive scheme is initialized as described in Algorithm 12. At MCMC iteration $t + 1$, the individual parameters θ_i are the first to be updated for all $i \in \llbracket 1, N \rrbracket$. A new candidate is sampled from the proposal distribution:

$$\theta_i^* \sim \mathcal{N}(\theta_i^t, \lambda_i^t \Sigma_i^t) \quad (8.14)$$

and the corresponding hidden states for the leaf areas $h_i(\theta_i^*)$ are simulated. We recall that λ_i^t and Σ_i^t are defined by an adaptive scheme for each individual for an optimized state space exploration as described in Section 4.2.5. The acceptance ratio α is then computed, the candidate accepted or rejected, and the adaptive scheme updated. All these operations can be performed independently for each individual on a particular process, making this step of the Gibbs sampler fully parallelized.

The mean and covariance of the individual parameters are then computed in parallel (see Appendix E). Then, parameters λ^* and Λ^* are updated according to Equation 4.21 and a new population mean vector η^{t+1} is sampled. This operation is performed on a single process, and the updated η^{t+1} is broadcasted on all other processes, so that the population values of the chain are identical for all individuals. All the same, q^* and Ψ^* are updated according to Equation 4.22. The update of $q^* = q + N$ is straightforward (and can actually happen once during the whole algorithm). As far as the update of Ψ^* is concerned, on each process we computed the matrix:

$$M_p = \sum_{i \in r_p} (\theta_i^{t+1} - \eta^{t+1})(\theta_i^{t+1} - \eta^{t+1})^T, \quad (8.15)$$

where we recall that r_p is the range of individuals belonging to process p . In the present case and since there is only one individual per process, this reduces to:

$$M_p = (\theta_p^{t+1} - \eta^{t+1})(\theta_p^{t+1} - \eta^{t+1})^T. \quad (8.16)$$

These matrices are converted to vectors and all these vectors are broadcasted to a single process, which is now able to reconstruct the whole matrix Ψ^* . A new value of the inverse covariance matrix is then sampled on this process and broadcasted to all others for consistency as previously mentioned. Finally, $\alpha^* = \alpha + n_{\text{tot}}/2$ is straightforwardly updated and a similar procedure as for Ψ^* is used for β^* . It is worth noting that when the number of observations is not the same in the experimental data y_i and in the candidate $h_i(\theta_i^{t+1})$, the vector that contains the most elements is deprived from the additional values corresponding to a higher number

of leaves having emerged. This can happen, for instance, when the phyllochron ϕ is estimated and that the number of leaves at the final time of the simulation is not the same. A new value for τ^* is then sampled, and the observation noise relative to the model simulation is accordingly updated.

8.1.4 First results

The Markov chains generated for several individual parameters sets are displayed on Figure 8.6. Since the data was simulated with the same model used for the individual GLS estimates, the latter were very close to the true parameters, hence the $N = 24$ estimates provide excellent prior distributions for the population mean vector and covariance matrix as already discussed. This logically initializes values for the Gibbs sampler that are close to the truth as well and the Markov chains have already converged in the first few iterations.

The accuracy of the estimation will sometimes be assessed using three different values for variable z :

$$\begin{cases} \delta^z &= \frac{|\hat{z} - z^{\text{true}}|}{z^{\text{true}}}, \\ \rho^z &= \frac{\hat{\sigma}^z}{z^{\text{true}}}, \\ \alpha^z &= \frac{|\hat{z} - z^{\text{true}}|}{\hat{\sigma}^z}, \end{cases} \quad (8.17)$$

where δ^z is the relative error, ρ^z the normalized estimated standard deviation, and $\alpha^z = \delta^z / \rho^z$ measures how well the posterior distribution captures the true value.

	e	μ_1	μ_2
$\mathbb{E}_i(\delta)$	$2.009 \cdot 10^{-03}$	$6.296 \cdot 10^{-04}$	$3.399 \cdot 10^{-04}$
$\mathbb{E}_i(\rho)$	$9.676 \cdot 10^{-04}$	$4.478 \cdot 10^{-04}$	$2.874 \cdot 10^{-04}$
$\mathbb{E}_i(\alpha)$	$2.074 \cdot 10^{+00}$	$1.384 \cdot 10^{+00}$	$1.202 \cdot 10^{+00}$

Table 8.2: Average of the coefficients δ , ρ and α over the different sets of individual parameters.

	η_1	η_2	η_3	Σ_{11}	Σ_{22}	Σ_{33}	τ
δ	$2.498 \cdot 10^{-03}$	$3.472 \cdot 10^{-03}$	$1.438 \cdot 10^{-03}$	$6.343 \cdot 10^{-02}$	$1.108 \cdot 10^{-01}$	$7.359 \cdot 10^{-01}$	$9.428 \cdot 10^{-02}$
ρ	$9.868 \cdot 10^{-03}$	$1.074 \cdot 10^{-02}$	$1.341 \cdot 10^{-02}$	$2.756 \cdot 10^{-01}$	$3.288 \cdot 10^{-01}$	$5.108 \cdot 10^{-01}$	$3.077 \cdot 10^{-02}$
α	$2.532 \cdot 10^{-01}$	$3.233 \cdot 10^{-01}$	$1.073 \cdot 10^{-01}$	$2.302 \cdot 10^{-01}$	$3.371 \cdot 10^{-01}$	$1.440 \cdot 10^{+00}$	$3.064 \cdot 10^{+00}$

Table 8.3: Coefficients δ , ρ and α for the different components of the population mean vector η , covariance matrix Σ and precision τ .

The results of Table 8.2 indicate that the individual parameters are very well estimated with an average of relative errors of $2 \cdot 10^{-03}$ at most. The average of the ρ coefficients does not exceed 10^{-03} , which indicates very concentrated Markov chains. Table 8.3 displays the results for the estimation of η , and here again the

relative errors are very low, not exceeding $3 \cdot 10^{-03}$. The standard deviations of the posterior distributions are much higher than in the case of individuals, with ρ coefficients of the order of 10^{-02} . Since the α coefficients are rather low, with values under $4 \cdot 10^{-01}$, the true values will be within the posterior distribution. Results for the estimation of Σ are slightly less good, with higher relative errors on the mean estimates, up to $7.4 \cdot 10^{-01}$ for Σ_{33} , i.e. 74%, which is not satisfying. All these observations can also be noticed on Figure 8.7. Note that the asymmetric shape of the Wishart distribution induces a difference between its mode and its mean.

In the following we will try to highlight the influence that some variables may have on the posterior distributions of the population parameters, namely the number of individuals, the intensity of the observation noise (i.e. the precision), the prior distributions, the number of estimated parameters and finally the correlation between parameters. In each situation we consider different cases, the easiest one being represented with the colour blue and the hardest one with red. The total number of MCMC iterations is always set to $M = 100,000$ and all the Markov chains are assumed to have converged unless stated otherwise. In particular, this means that the results obtained cannot be improved with a greater M . Because of the onerousness of running such algorithms, only 5 scenarios at most are considered for each variable.

8.1.5 Influence of the number of individuals

Owing to the very nature of population models, the first influence variable that comes to mind is the number of individuals in the population. When only a few individuals are available, they might not represent the population distribution very well. To investigate this effect, we took advantage of the fully parallelized version of our algorithm as well as the capabilities of CentraleSupélec's computing cluster to generate and estimate populations of N individuals with $N \in \{12, 24, 48, 96, 192\}$. We were able to require as many processes as individuals in the population, which means that the overall computing time was the same for all cases (approximately 10 minutes), which represents a substantial time-saving trick: had the algorithm be written sequentially, the total computing time for $N = 192$ would have been of 32 hours.

It is worth mentioning that this is the single case study where the populations will differ. In the following sections, all the populations are composed of the exact same individuals, whether it be in terms of parameters or observations. Here, the individuals generated for a particular N also belong to the population of individuals generated for greater N 's.

On Figure 8.9, we displayed the posterior distributions of all the components of the population mean vector η_1 , η_2 and η_3 and on Figure 8.10 those of the diagonal components of the population covariance matrix Σ_{11} , Σ_{22} and Σ_{33} . On each graph, the true value, say z^{true} , is represented by a dashed black line and the magenta lines represent the $\pm 5\%$ values, $0.95 z^{\text{true}}$ and $1.05 z^{\text{true}}$. From Figure 8.9, the posterior distributions for the population mean vector components exhibit a slight bias for lower values of N . As N increases, their modes get closer to the true values, their variances decrease and the posterior distributions become more centered and narrowed. Still, even with a few individuals in the population, the posterior distribution for all components of η are mostly within the $\pm 5\%$ interval, thereby highlighting the accuracy of the estimates.

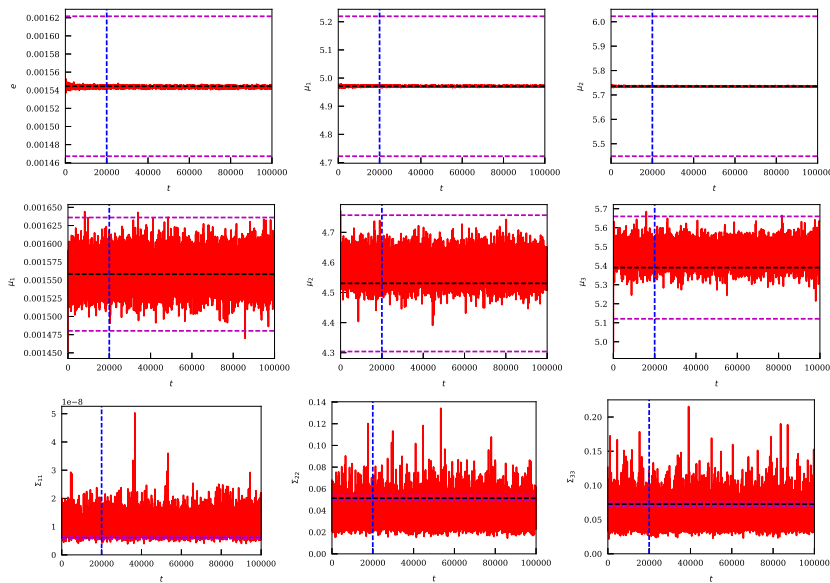


Figure 8.6: Markov chains for one individual θ_1 (top), the population mean η (middle) and the population covariance matrix Σ (bottom) for the three parameters e (left), μ_1 (center) and μ_2 (right).

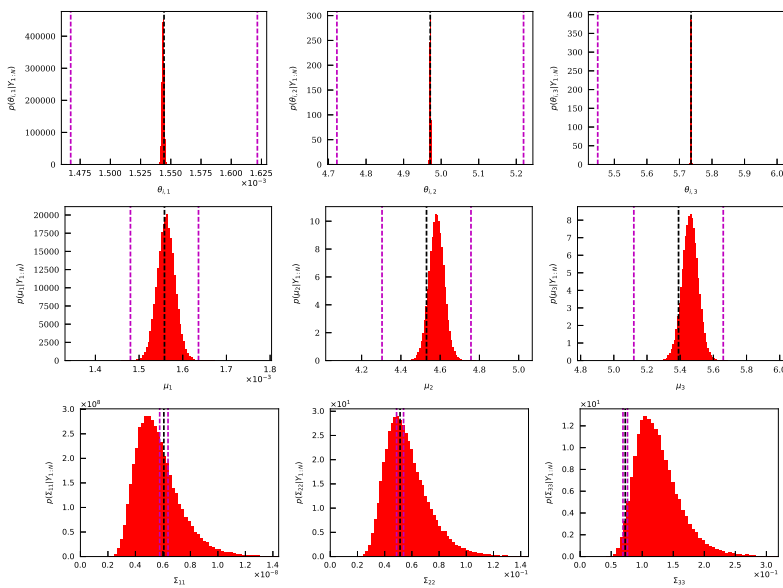


Figure 8.7: Posterior distribution for one individual θ_1 (top), the population mean η (middle) and the population covariance matrix Σ (bottom) for the three parameters e (left), μ_1 (center) and μ_2 (right).

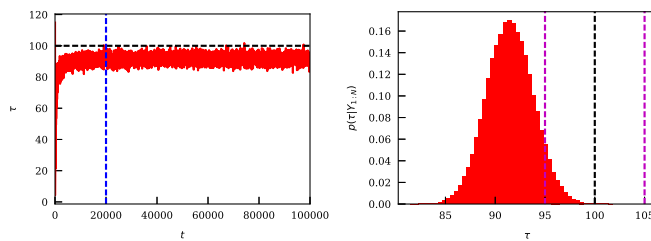


Figure 8.8: Markov chain and posterior distribution for the population precision τ .

As far as the covariance matrix Σ is concerned, the quality of the estimates for low values of N is not as good as for η . It is worth noting that Σ_{11} and Σ_{22} are most of the time underestimated whereas Σ_{33} is overestimated, rather strongly for low values of N , although Σ_{22} and Σ_{33} are of the same order of magnitude. Increasing the number of individuals in the population turns out to be particularly relevant for the covariance matrix: the estimates for Σ_{11} , Σ_{22} and Σ_{33} are greatly improved for $N = 96$ and $N = 192$, and the variance of the posterior distribution is even lower for $N = 192$. The estimation results for $N = 12$ are clearly insufficient, in particular for Σ_{22} and Σ_{33} , but the results significantly improve with $N = 24$ already. Ideally though, having populations with a large number of individuals ($N \gtrsim 100$) would be reassuring for the estimation of Σ .

Last, but not least, the posterior distributions for the population precision τ are displayed on Figure 8.11. Strikingly, all the posterior distributions have their modes around $\tau = 90$ and they fail to correctly estimate the true value of the precision. Increasing the number of individuals in the population has for only effect to lower the variance of the posterior distribution around this value and there will always exist a bias for the precision estimate whatever the value of N .

8.1.6 Influence of the precision

Seeing how the estimation of the precision might yield its share of difficulties, we focus in this section on observation noises of different intensities for a number of individuals $N = 24$. Basically, we consider standard deviations for the observation noise $\sigma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$, which corresponds to precisions

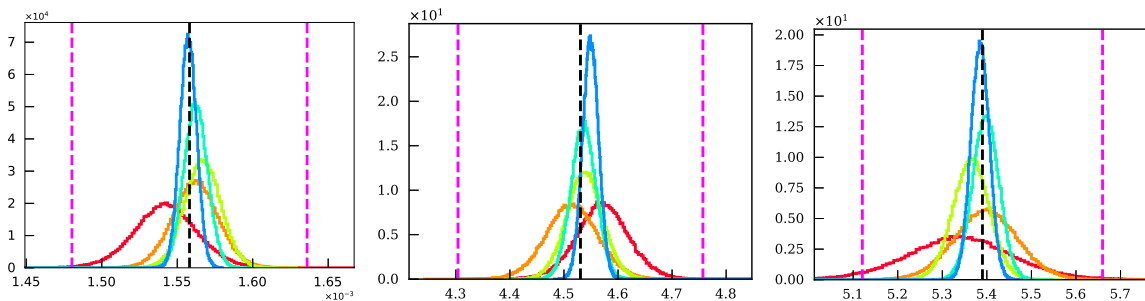


Figure 8.9: Posterior distributions of the components of η for different number of individuals (● $N = 12$ ● $N = 24$ ● $N = 48$ ● $N = 96$ ● $N = 192$).

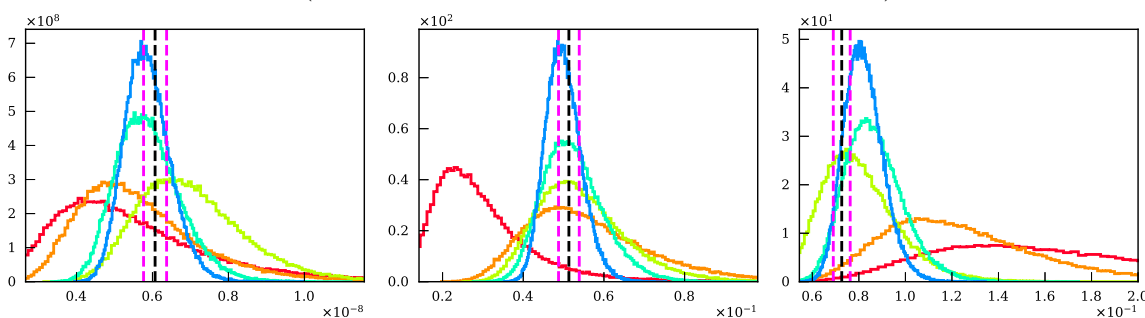


Figure 8.10: Posterior distributions of the diagonal components of Σ for different number of individuals (● $N = 12$ ● $N = 24$ ● $N = 48$ ● $N = 96$ ● $N = 192$).

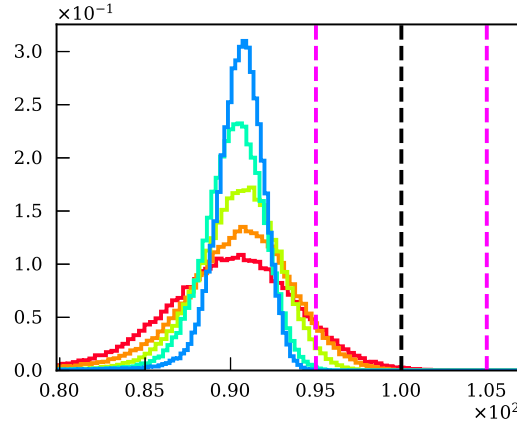


Figure 8.11: Posterior distribution of the precision τ for different number of individuals (● $N = 12$ ● $N = 24$ ● $N = 48$ ● $N = 96$ ● $N = 192$).

	$\tau = 100$	$\tau = 25$	$\tau = 11.11$	$\tau = 6.25$	$\tau = 4$
δ^τ	$9.428 \cdot 10^{-02}$	$3.624 \cdot 10^{-01}$	$5.041 \cdot 10^{-01}$	$4.810 \cdot 10^{-01}$	$4.044 \cdot 10^{-01}$
ρ^τ	$3.077 \cdot 10^{-02}$	$2.427 \cdot 10^{-02}$	$1.783 \cdot 10^{-02}$	$1.988 \cdot 10^{-02}$	$2.405 \cdot 10^{-02}$
α^τ	$3.064 \cdot 10^{+00}$	$1.493 \cdot 10^{+01}$	$2.827 \cdot 10^{+01}$	$2.419 \cdot 10^{+01}$	$1.681 \cdot 10^{+01}$

Table 8.4: Values of the coefficients δ , ρ and α for the different values of the precision.

$\tau = \sigma^{-2} \in \{100.0, 25.0, 11.11, 6.25, 4.0\}$. This set of values most probably includes some that are unrealistically high but that remain of interest for the study of how well the algorithm performs in the presence of high noise. For each value of τ , the population parameters η , Σ and all the individual parameters $(\theta_i)_{i \in [1, N]}$ were identical. In particular, for individual i , the hidden states for leaf area $h_i(\theta_i)$ did not change w.r.t. the value of τ used, and only the observations y_i were different.

The posterior distributions for η can be seen on Figure 8.12 and those for Σ on Figure 8.13. For both η and Σ , the estimation of the second and third parameters is barely affected by higher observation noises. Only the first parameter, which corresponds to e , improves with increasing τ . As far as Σ_{11} is concerned, even though the mode of the posterior distributions for $\tau = 25$ and $\tau = 100$ is not centered on the true value as is the case for other values of τ , the estimates obtained by computing the mean from the posterior distributions have lower relative errors (see Table 8.4) for $\tau = 25$ and $\tau = 100$ owing to the shape of the Wishart distribution. Overall, the results obtained for η and Σ remain excellent: the relative errors for η_2 and η_3 are always lower than $5 \cdot 10^{-03}$ for all cases, and the relative error for η_1 does not exceed $3 \cdot 10^{-02}$ in the worst case scenario ($\tau = 4$). Σ_{11} and Σ_{22} are well estimated, with relative errors always less than 10^{-01} in all cases. As already noticed, Σ_{33} is the only component of μ and Σ that is not well estimated, with relative errors $\delta^{\Sigma_{33}} \approx 7 \cdot 10^{-01}$.

The results for τ are displayed in Table 8.4. It is worth noting that all the chains for the precision had converged and that the results could not have been improved with a higher number of iterations. Even in the easiest case (i.e. $\tau = 100$), the relative error on the precision is of almost 10%. This error degrades to 36% for $\tau = 25$ and to 50% for $\tau = 11.11$. In fact, the precision τ is always underestimated which equates to the standard deviation of the observation noise σ being always overestimated.

We recall that if X follows a Gamma distribution $\mathcal{G}(\alpha, \beta)$, then $\mathbb{E}(X) = \alpha/\beta$ and $V(X) = \alpha/\beta^2$. Since the update of the full conditional distribution of the precision τ is based on $\alpha^* = \alpha + n_{\text{tot}}/2$ and $\beta^* = \beta + \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i))$, when the chains have converged the expectation of the precision is thus governed by:

$$\frac{\alpha + n_{\text{tot}}/2}{\beta + \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i))} \tag{8.18}$$

and one might therefore be tempted to adapt the values of α and β so as to raise this ratio. Here, $n_{\text{tot}}/2 = 1632$, and the initial choices for the hyperparameters of the Gamma distribution were $\alpha = 7.29$ and $\beta = 0.081$. A straightforward analysis of the orders of magnitude of each term involved reveals that the denominator of the ratio 8.18 is in the present case greatly dominated by $S = 1/2 \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i))$ and decreasing the value of β will hence not change much, whereas increasing α seems more meaningful.

For $\tau = 100$, we considered 5 different values for α , most of them were higher than the initial 7.29 in order to heighten the ratio 8.18 and the last one was taken very low, i.e. $\alpha = 0.01$. The motivation behind the last scenario is to evidence the lower bound on the estimate of τ . We therefore considered the following cases: $\alpha \in \{0.01, 30, 100, 300, 1000\}$.

The results that were obtained for η and Σ were identical for all the choices of α , which means that their estimation is robust to potentially ill-chosen priors for the precision. The posterior distributions of τ for the different cases are displayed on Figure 8.14. First, we notice that even for $\alpha = 0.01$ the results are close to those obtained so far. For such low values of α , the standard deviation of the observation noise during the

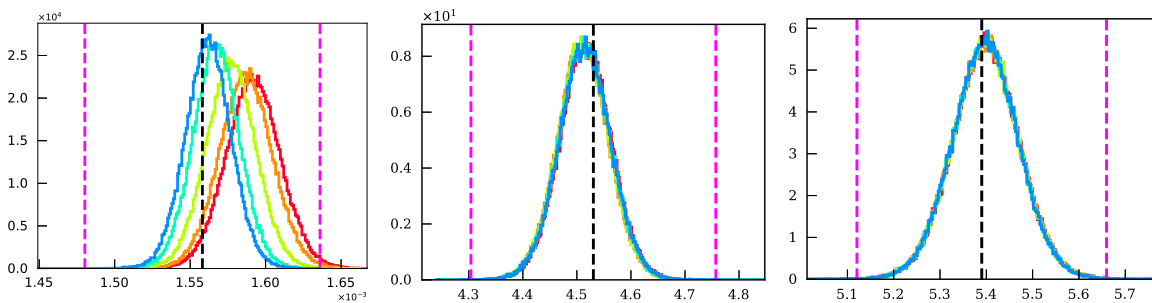


Figure 8.12: Posterior distributions of the individual components of η for different values of the precision (\bullet $\tau = 4$ \bullet $\tau = 6.25$ \bullet $\tau = 11.11$ \bullet $\tau = 25$ \bullet $\tau = 100$).

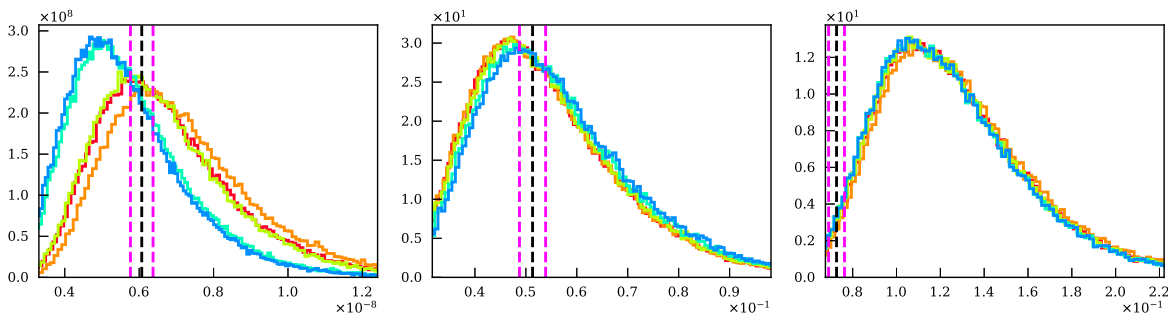


Figure 8.13: Posterior distributions of the diagonal components of Σ for different values of the precision (\bullet $\tau = 4$ \bullet $\tau = 6.25$ \bullet $\tau = 11.11$ \bullet $\tau = 25$ \bullet $\tau = 100$).

first iterations of the MCMC chain will be very high, which affects notably the acceptance probability (see Equation 4.35). In spite of these seemingly absurd values for the precision, the chain converges rather quickly to more sensible values for τ , but also η and Σ .

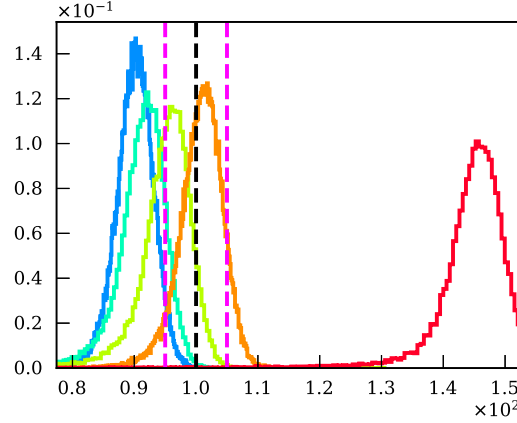


Figure 8.14: Posterior distribution of the precision τ ($\tau^{\text{true}} = 100$) for different prior distributions (\bullet $\alpha = 0.01$ \bullet $\alpha = 30$ \bullet $\alpha = 100$ \bullet $\alpha = 200$ \bullet $\alpha = 1000$).

As α increases, the posterior distribution starts to shift towards the true value, which is encouraging. However, there is a threshold beyond which the posterior of τ significantly overestimates the latter. This threshold can be roughly estimated in our estimation with:

$$\alpha^{\text{opt}} \approx \tau^{\text{true}} (\beta + S) - n_{\text{tot}}/2 \approx 200. \quad (8.19)$$

In practice though, τ^{true} is unknown, and S depends on the population mean η and the individual parameters $(\theta_i)_{i \in [1, N]}$, so this rough estimate only serves as illustrative purposes and should not be taken as a recipe for finding the optimal α . Still, the case $\alpha = 200$ is the one that yields the best posterior distribution for τ as illustrated on Figure 8.14. When α keeps on increasing, the posterior distribution is shifted even more to higher values of τ .

In conclusion, if β is negligible with respect to S , the estimate for τ is lower bounded by that obtained when $\alpha \rightarrow 0$. This careful strategy leads to relative errors that increase with lower τ . The optimal value for α when τ^{true} is known can be roughly assessed.

8.1.7 Influence of the prior

As mentioned previously, in the case of simulated data the GLS estimates provide excellent prior distributions. However, in the case of real data there are two additional sources of uncertainty since (i) the plant growth model that we have designed can never absolutely accurately represent real data and (ii) the image analysis algorithm introduces observation errors whose extent is hard to assess, whence less assurance that the prior distributions be so precise in the case of real data. In order to study this effect, we initialized the individual parameters and the population mean vector far from their true values. The convergence of the estimation

method is likely to be impacted by the initial values of all the parameters' sets. The prior distributions were set as follows:

$$\lambda_i = \hat{\theta}^{\text{GLS}} (1 + (-1)^i \epsilon) \quad (8.20)$$

and where $\epsilon \in \{0.01, 0.05, 0.10, 0.15, 0.20\}$. The factor $(-1)^i$ ensures that not parameters are neither all underestimated nor all overestimated.

For both η and Σ (see Figure 8.15 and 8.16) the results are very similar as long as $\epsilon < 0.15$. For $\epsilon = 0.20$, the posterior distributions start to be biased despite the convergence of the chains, even though the estimates remain decent. The estimation results for η and Σ therefore seem to be rather robust to moderately distant priors, which probably results from the relatively large number of available observations. The results for τ can be seen on Figure 8.17. The shape of the posterior distributions can be explained simply: this is one of those rare cases in these applications for which the chains for τ have not converged. The convergence of the chains for each case can be smoothed for better visualization by computing the mean value of the chain over each batch of 1,000 iterations for instance. Such results are displayed on Figure 8.18. It is not clear from the latter whether the chains will converge towards the same value that would have been obtained for $\epsilon = 0$.

To shed light on this issue, we ran an algorithm for $\epsilon = 0.10$ with a number of iterations of $M = 500,000$, for which the smoothed Markov chain for τ is shown on Figure 8.18. No definitive conclusion can be drawn from this curve, but it would seem that the chain would eventually converge towards the same value obtained for $\epsilon = 0$. Anyway, in practice it becomes unpractical to run the algorithm for so many iterations. Instead, we suggest that if the convergence of the Markov chain for τ is too slow, it is likely that the prior distributions

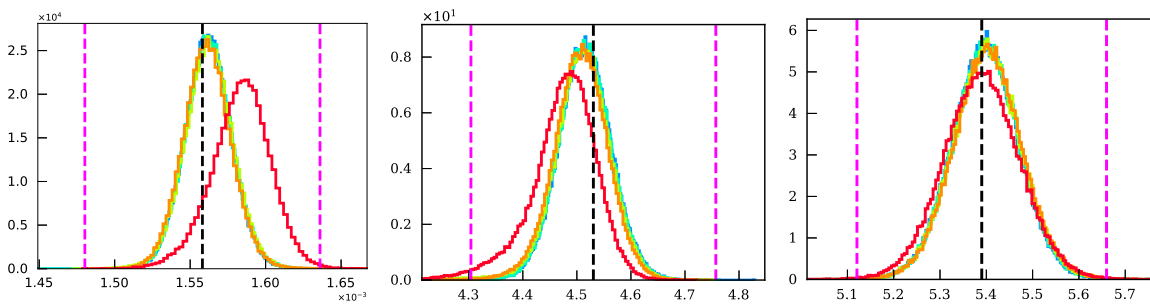


Figure 8.15: Posterior distributions of the individual components of η for different priors ($\bullet \epsilon = 0.01$ $\bullet \epsilon = 0.05$ $\bullet \epsilon = 0.10$ $\bullet \epsilon = 0.15$ $\bullet \epsilon = 0.20$).

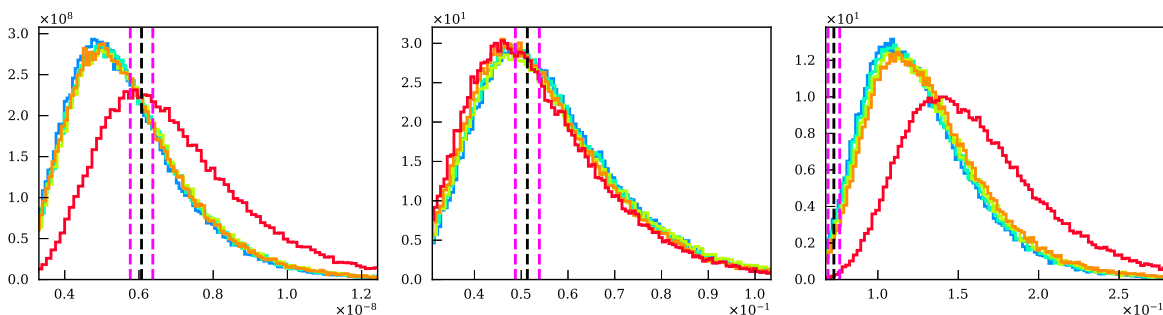


Figure 8.16: Posterior distributions of the diagonal components of Σ for different priors ($\bullet \epsilon = 0.01$ $\bullet \epsilon = 0.05$ $\bullet \epsilon = 0.10$ $\bullet \epsilon = 0.15$ $\bullet \epsilon = 0.20$).

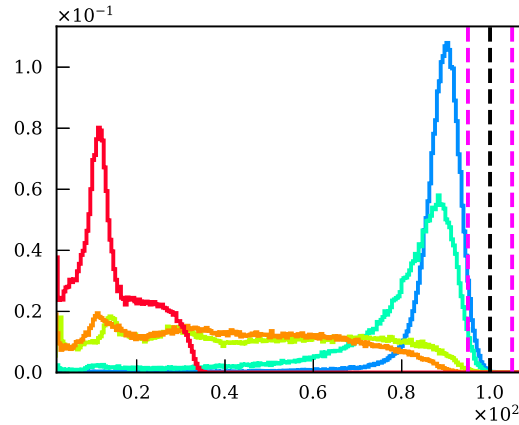


Figure 8.17: Posterior distribution of the precision τ for different priors ($\bullet \epsilon = 0.01$ $\bullet \epsilon = 0.05$ $\bullet \epsilon = 0.10$ $\bullet \epsilon = 0.15$ $\bullet \epsilon = 0.20$).

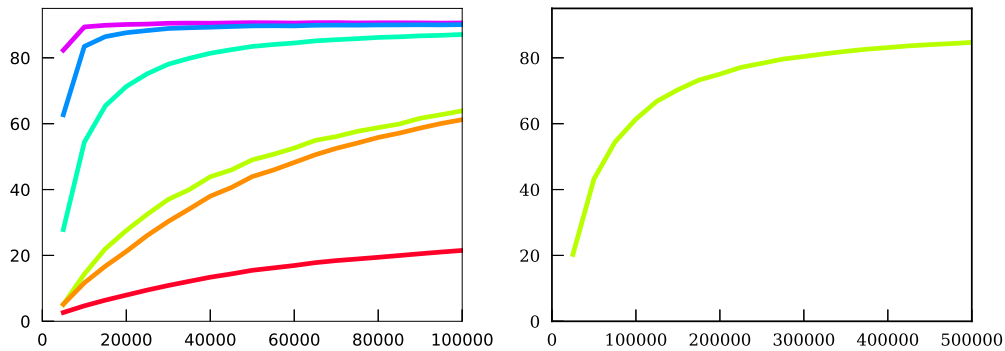


Figure 8.18: Left: smoothed Markov chains for the precision τ for different priors ($\bullet \epsilon = 0.00$ $\bullet \epsilon = 0.01$ $\bullet \epsilon = 0.05$ $\bullet \epsilon = 0.10$ $\bullet \epsilon = 0.15$ $\bullet \epsilon = 0.20$). Right: smoothed Markov chains for the precision τ with $\epsilon = 0.10$ and $M = 500,000$.

for the individual parameters or the population mean vector could be chosen differently for an improved convergence.

8.1.8 Influence of the number of estimated parameters

The last case study that we consider is that of the number of estimated parameters d . So far, only three parameters were estimated for convenience. For real case scenarios, more parameters need to be estimated in order to fit the experimental data correctly as will be detailed in Section 8.2. The study of how the estimation evolves with the dimension of the problem is therefore necessary, all the more so considering the additional difficulty of state space exploration due to the curse of dimensionality.

For the reasons given in Section 6.1, we considered the additional estimation of the parameters σ_1 and σ_2 . The three different cases that were designed were therefore:

$$\begin{cases} \theta = (e, \mu_1, \mu_2) & \text{i.e. } d = 3, \\ \theta = (e, \mu_1, \mu_2, \sigma_1) & \text{i.e. } d = 4, \\ \theta = (e, \mu_1, \mu_2, \sigma_1, \sigma_2) & \text{i.e. } d = 5. \end{cases} \quad (8.21)$$

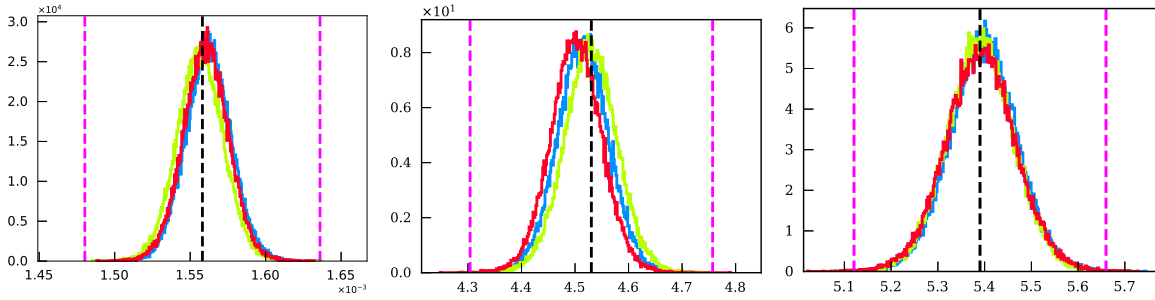


Figure 8.19: Posterior distributions of the individual components of η for different number of estimated parameters ($\bullet d = 3$ $\bullet d = 4$ $\bullet d = 5$).

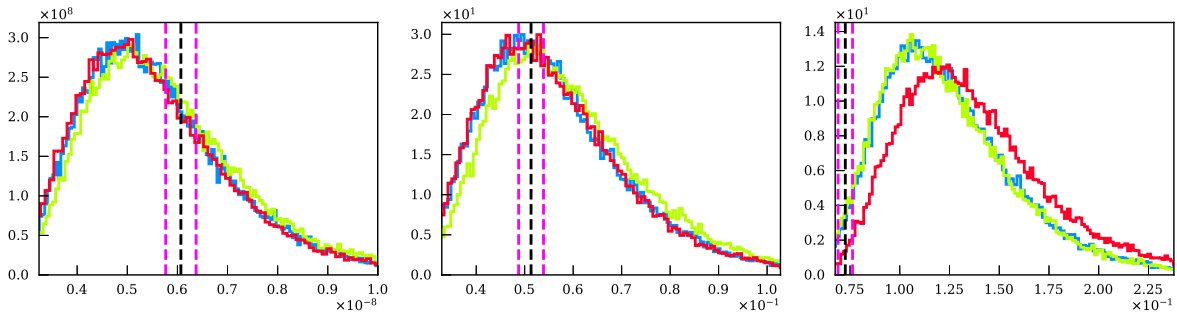


Figure 8.20: Posterior distributions of the diagonal components of Σ for different number of estimated parameters ($\bullet d = 3$ $\bullet d = 4$ $\bullet d = 5$).

The results for η and Σ are displayed on Figure 8.19 and Figure 8.20. Increasing d does not seem to significantly impact their estimation. Only the posterior distribution for Σ_{33} in the case of $d = 5$ is slightly worse compared to the cases $d \in \{3, 4\}$.

Again, real differences appear in the estimation of the population precision. The Markov chains are shown on Figure 8.21. The convergence of the chains slows down with increasing d 's. Since:

$$\beta^* = \beta + \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i)), \quad (8.22)$$

as the number of estimated parameters increases, the dimension of the state space implies slower convergence of the $h_i(\theta_i)$ towards the observations. A summary of the estimates of the individual parameters for the different cases is given in Table 8.5. Not only the mean relative errors on each individual parameters e , μ_1 and μ_2 indeed increases with d , but the addition of the other parameters σ_1 and σ_2 , whose mean relative errors are higher than for the first three estimated parameters, also contributes to a slower exploration of the state space, and hence a slower convergence for τ .

Previously, for ill-chosen prior distributions, it was still possible to accelerate the convergence of the chains by choosing more appropriate priors for the model parameters. There is no such obvious solution in the case of increasing dimension of the problem, which might be more problematic.

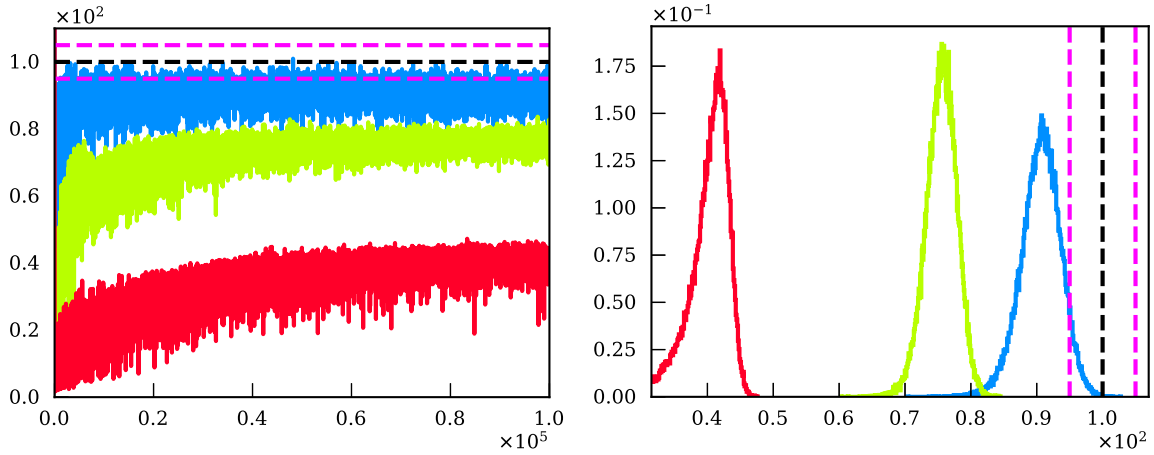


Figure 8.21: Posterior distribution (left) and Markov chains (right) of the precision τ for different number of parameters ($\bullet d = 3$ $\bullet d = 4$ $\bullet d = 5$).

	$d = 3$	$d = 4$	$d = 5$
$\mathbb{E}_i(\delta^e)$	$2.009 \cdot 10^{-03}$	$3.060 \cdot 10^{-03}$	$5.970 \cdot 10^{-03}$
$\mathbb{E}_i(\delta^{\mu_1})$	$6.296 \cdot 10^{-04}$	$2.579 \cdot 10^{-03}$	$5.328 \cdot 10^{-03}$
$\mathbb{E}_i(\delta^{\mu_2})$	$3.399 \cdot 10^{-04}$	$4.407 \cdot 10^{-04}$	$6.841 \cdot 10^{-03}$
$\mathbb{E}_i(\delta^{\sigma_1})$	\circ	$1.144 \cdot 10^{-02}$	$1.649 \cdot 10^{-02}$
$\mathbb{E}_i(\delta^{\sigma_2})$	\circ	\circ	$1.755 \cdot 10^{-02}$

Table 8.5: Average of the relative errors δ for all individual parameters $(\theta_i)_{i \in [1, N]}$.

8.2 Real data

8.2.1 Estimation on 24 individuals

Now that the estimation procedure has been detailed and shown to provide good estimates for both the individual and population parameters on synthetic data, we are going to apply it to real ones. Several key differences must be noted. Obviously, the true values for the parameters and the hidden states are not known anymore. Additionally, the experimental data sets obtained should be noisier than the simulated data for several reasons. First and foremost, the actual growth of *Arabidopsis thaliana* individuals does not follow exactly the dynamics of the GreenLab model. Second, the image analysis algorithm designed in Chapter 7 is suffused with additional noise that is hard to estimate: if the results in the case of the four individuals where manually extracted data was available proved excellent, this does not suffice for a generalization of the observation noise model, and as a matter of fact, our algorithm failed for some individuals out of the 64.

We therefore expect the individual observations to be noisier than before. On top of that, the number of observations will be less than previously. If we had considered that the areas of all leaves were observed at all times, this is not the case anymore. When the total number of observations was $n_{\text{tot}} = 3264$ earlier, it will now only be $n_{\text{tot}} = 1975$ for the same number of individuals $N = 24$, i.e. 1.65 times less. The data sets that we used is the one described in Section 7.6.

We recall from Section 7.6 that the 64 individuals were divided into 4 categories as a result of the image analysis procedure:

- **Category 1:** the algorithm worked very well and all the data obtained for each leaf have reasonable and sensible values, there is no need at all to discard data from the individuals of this category. It comprises 24 individuals;
- **Category 2:** the algorithm worked well but there are a few (between 1 and 4) absurd observations among all possible data obtained for the different leaves: for instance, the last data for the 6th and 7th leaves might be way too large because segments have been wrongly allocated to these leaves on a particular day. These data could be filtered out but could also be kept without having much influence on the whole estimation as they represent a small percentage of all the data for a given individual. This category comprises 14 individuals;
- **Category 3:** the algorithm worked well for the major part but some leaves (between 1 and 4) among the 6th, 7th, 8th, 9th and 10th leaves are badly segmented and need to be removed entirely. This category comprises 10 individuals;
- **Category 4:** finally, there are individuals for which the algorithm simply failed, whether it be due to several plants growing together, pot rotations or a misdetection of the first occurrence of the 5th leaf. This category comprises 16 individuals.

Among the 64 individuals on which we run our image analysis algorithm, only those 24 individuals from Category 1 were retained for a first estimation procedure. Indeed, we decided to adopt a cautious approach where first calibrations would be based on the cleanest data possible.

We chose to estimate as many parameters as was possible, which would allow for more individual freedom and to assess if some model parameters can be considered constant throughout the population based on the posterior distribution of their population variables, notably that of the covariance matrix Σ . Because of the technical difficulty inherent to the estimation of the phyllochron ϕ , which drives the appearance of the leaves, we decided to assess its value individually by averaging the rate of appearance of the leaves for each individual. The other parameters that were considered fixed were μ , s and k , the first two for identifiability reasons, and the last one because it is considered constant and well known from biological knowledge. The estimated parameters were therefore chosen to be:

$$\theta = (e, \mu_1, \sigma_1, \mu_2, \sigma_2, \rho_2, q^0). \quad (8.23)$$

Despite the differences with the case of simulated data, the estimation strategy remains the same as before:

- we start by calibrating the individual parameters by running a GLS algorithm for each individual,
- the prior distributions of the population mean and covariance matrix are initialized in the same way by using these GLS estimates,
- the core implementation of the algorithm respects the same principles as those described in Section 8.1.3.

It appeared that the Markov chains – in particular for the individual parameters and for the population mean vector – had not converged for a total number of iterations of $M = 100,000$. We therefore ran the algorithm for $M = 500,000$ iterations, for which all the different chains had reached their stationary distribution. The burn-in period was taken to be $B = 250,000$ to ensure that all transitional values were discarded.

On Figure 8.22, we displayed for the 24 individuals the experimental data sets vs. the results of the GreenLab model simulation using the estimated individual parameters. The individuals are ranked from top to bottom and from left to right. The 10th individual therefore appears in the third row and second column. For all the individuals, the leaf area curves are rather well estimated and the orders of magnitude for the different individuals are well respected: when the 9th individual had leaf areas as high as 1.75 cm at the end of the growth, the 11th individual struggled at 1.00 cm, and the growth curves simulated using the estimated individual parameters follow this trend. This is true not only for the values that the growth curves reach, but also for the rate of appearance of the different leaves: for instance, the 17th individual has its 15th leaf emerged by the end of the simulation, whereas the 10th individual only has 12 leaves emerged.

The estimation of the individual parameters hence allows to reproduce the diversity of the growth dynamics of the whole population. However, the dynamics of some leaves are not captured as well as for some others. For the majority of the individuals, the simulated growth curves of the 3rd and 4th leaves plateau rather rapidly, starting from around $t = 125\text{h}$, whereas the data obtained from image analysis suggest that even if the growth of these leaves start to decrease around this time as well, they do not reach a limit for all that.

In order to figure out how the individual estimates perform for each leaf, we computed the different modelling efficiencies summarized in Table 8.7. The modelling efficiencies for the first two leaves are surprisingly not very good (they are actually the leaves for which it is worse): their average leaf areas being much smaller than those of the other leaves, it was hardly discernible that their estimates were not very good from the graphs of Figure 8.22. These mixed results indicate that the dynamics of the first stages of the model could be refined, since the first two leaves are the less inclined to be affected with image analysis noise: they are indeed easily segmented given the simple architecture of the plant on the first images. The modelling efficiencies of the 3rd and 4th leaves are better than for the 1st and 2nd leaves but not optimal yet, with values around 0.73. This was expected as said before, the main problem probably being the dynamic constraints imposed upon the growth of these leaves by the GreenLab model via the demand functions. Optimizing the values of the model parameters ruling the log-normal demand functions does not seem to be enough for a perfect adequation with experimental data. The leaves of second rank ($v > 4$) exhibit modelling efficiencies much better than their counterparts of first rank, with values always higher than 0.8 and two of them very close to 0.9. Overall, these results remain encouraging as one might need to bring some improvements to the early stages of growth,

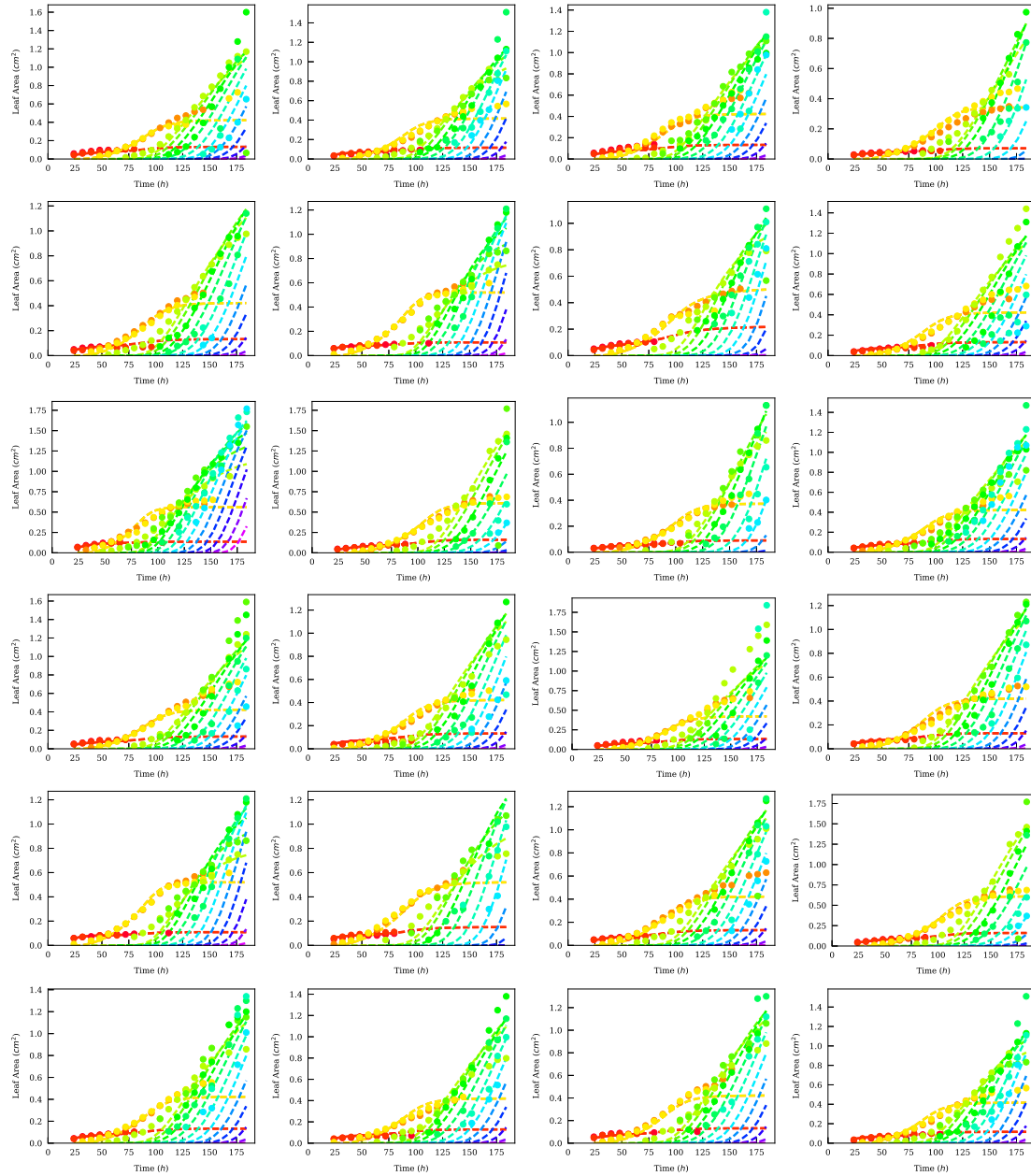


Figure 8.22: Experimental data (filled circles) obtained via image analysis vs. results of the predictions obtained by simulation of the model with the individual parameters θ_i inferred from Bayesian hierarchical estimation.

notably to biomass initiation and the allocation of biomass to the leaves of first rank. In particular, other demand functions could be proposed. We recall that the original Beta functions of the GreenLab model were not used in this study since they necessitate a time scaling parameter that was not easily available here.

The posterior distributions for η and Σ are displayed in blue on Figure 8.24 and look very much alike: the posterior distributions of the components of η corresponding to normal distributions and those for the diagonal components of Σ to Wishart ones. The mean and standard deviation for each of these posterior distributions were computed and are summarized in Table 8.6.

As far as the population precision is concerned, its chain rapidly converged to its stationary distribution.

	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7
$\mathbb{E}(\eta_i)$	$1.672 \cdot 10^{-03}$	$4.262 \cdot 10^{+00}$	$4.127 \cdot 10^{-01}$	$5.464 \cdot 10^{+00}$	$4.336 \cdot 10^{-01}$	$2.389 \cdot 10^{-02}$	$7.392 \cdot 10^{-05}$
$\sigma(\eta_i)$	$1.608 \cdot 10^{-05}$	$2.102 \cdot 10^{-02}$	$9.355 \cdot 10^{-03}$	$3.968 \cdot 10^{-02}$	$1.665 \cdot 10^{-02}$	$2.703 \cdot 10^{-03}$	$2.503 \cdot 10^{-06}$
$\mathbb{E}(\Sigma_{ii})$	$3.484 \cdot 10^{-09}$	$6.834 \cdot 10^{-03}$	$1.660 \cdot 10^{-03}$	$2.567 \cdot 10^{-02}$	$4.796 \cdot 10^{-03}$	$1.839 \cdot 10^{-04}$	$9.473 \cdot 10^{-11}$
$\sigma(\Sigma_{ii})$	$1.220 \cdot 10^{-09}$	$2.254 \cdot 10^{-03}$	$5.489 \cdot 10^{-04}$	$8.661 \cdot 10^{-03}$	$1.645 \cdot 10^{-03}$	$6.139 \cdot 10^{-05}$	$3.194 \cdot 10^{-11}$
$\mathbb{E}(\Sigma_{ii})/\mathbb{E}(\eta_i)$	$3.530 \cdot 10^{-02}$	$1.940 \cdot 10^{-02}$	$9.872 \cdot 10^{-02}$	$2.932 \cdot 10^{-02}$	$1.597 \cdot 10^{-01}$	$5.676 \cdot 10^{-01}$	$1.317 \cdot 10^{-01}$

Table 8.6: Mean and standard deviation of the posterior distributions for each component of the population mean vector η .

	Leaf 1	Leaf 2	Leaf 3	Leaf 4	Leaf 5	Leaf 6	Leaf 7	Leaf 8
EF	$2.487 \cdot 10^{-01}$	$3.722 \cdot 10^{-01}$	$7.304 \cdot 10^{-01}$	$7.349 \cdot 10^{-01}$	$8.941 \cdot 10^{-01}$	$8.331 \cdot 10^{-01}$	$8.929 \cdot 10^{-01}$	$8.273 \cdot 10^{-01}$

Table 8.7: Modelling efficiency computed over the 24 individuals for each leaf.

This is encouraging in the sense that this case does not exhibit the problematic behaviours that were witnessed in the case of ill-chosen prior distributions or with an increasing number of parameters. The estimated precision was $\mathbb{E}(\tau) = 5.378$, which corresponds to an observation noise with standard deviation of $\mathbb{E}(\sigma) = 1/\sqrt{\mathbb{E}(\tau)} = 0.431$. If this value might seem high at first glance, one has to remember the two main sources of uncertainty involved: first, the image analysis algorithm might have yielded data noisier than expected considering the preliminary results obtained on 4 individuals; second, the GreenLab model and notably the allocation process, specifically for the leaves of first rank, could probably be refined to allow for a better dynamics of the growth curves and a softer difference between the experimental data and the simulated states obtained from the individual parameters.

8.2.2 Integration of other individuals

This first procedure provides estimates for the population parameters η , Σ and τ , which can further be used as prior information for the estimation of a second batch of individuals. This second batch comprises the individuals from Categories 2 and 3. These individuals indeed provide data that is either in reduced quantity or of lesser quality.

There are several reasons for splitting the entire data set of 48 individuals (comprising the 24 individuals from Category 1, the 14 individuals from Category 2 and the 10 individuals from Category 3) into two distinct data sets for the estimation. First, considering the 48 individuals altogether would yield an estimate for the population precision that is lower than for the 24 individuals, and that is most probably due to the additional uncertainty introduced by the image analysis algorithm. Yet, estimating the precision using only the best individuals available does not prevent to incorporate the data that those from Categories 2 and 3 yield by performing a new estimation by fixing the precision to its estimated value and by setting the prior distributions of the population mean and covariance matrix to their posterior distributions obtained using the individuals from Category 1.

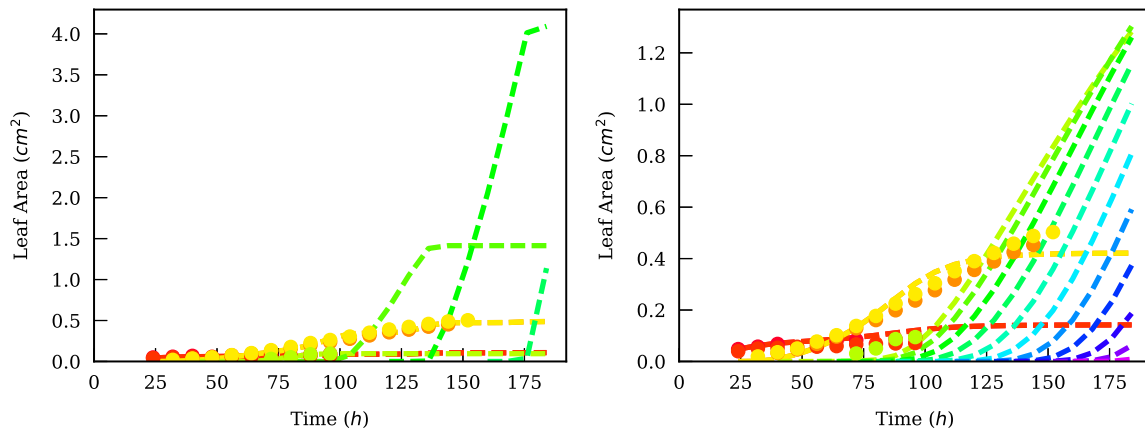


Figure 8.23: Example of model calibration for one particular individual of Category 3 obtained via GLS (left) or after Bayesian hierarchical estimation (right). The GLS highly overfit the data whereas the population approach provides a good compromise.

Furthermore, for individuals from Categories 2 and 3, the Markov chains for the individual parameters are also initialized from the population distribution $\theta_i \sim \mathcal{N}(\eta^{(1)}, \Sigma^{(1)})$ where $\eta^{(1)}$ and $\Sigma^{(1)}$ are the mean estimates of the posterior distributions obtained at the end of the first estimation procedure. For individuals of Categories 2 and 3, we indeed noticed that the GLS estimates were likely to highly overfit the model because of the data of lesser quality, whereas the population distribution would provide a reasonable compromise and fit of the model. This effect is illustrated on Figure 8.23 for an individual of Category 3.

In a nutshell, this two-step procedure allows the integration of more individuals in the pool of data, as it has the advantages of providing a more reasonable estimate for the population precision and more sensible initial values for individuals where a GLS procedure could fail.

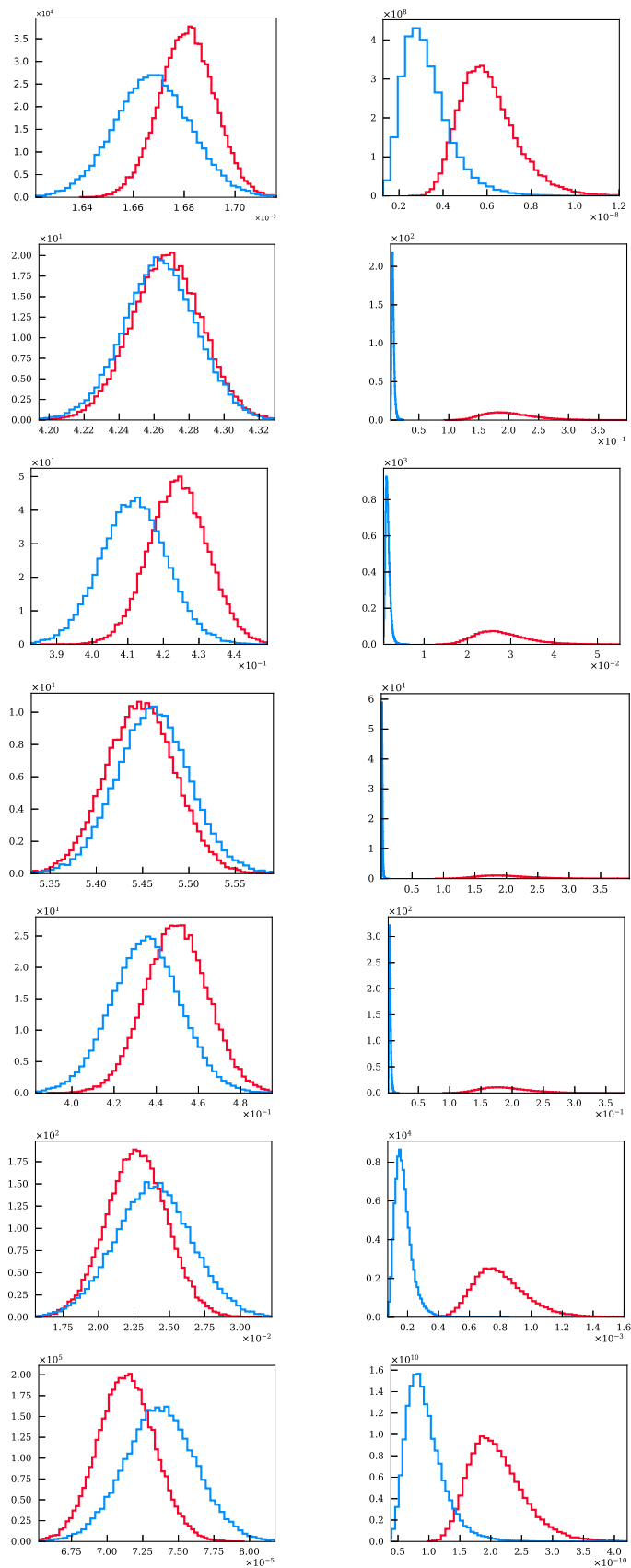


Figure 8.24: Posterior distributions of the components of the population mean vector η (left) and the diagonal components of the population covariance matrix Σ (right) after the first estimation run using individuals from Category 1 (blue) and the second estimation run with the integration of individuals from Categories 2 and 3 (red).

Discussion and perspectives

We endeavoured in this thesis to address different issues regarding the calibration of plant growth models in several situations, notably both in the case of single individuals and for populations of plants. Some elements of this work tried to shed some light on these questions but do not always constitute a definite answer, and sometimes raise new questions in return. In this last chapter, we summarize the different contributions of this work, which includes statistical analysis of plant growth models but also the necessary detours via topics such as computer science and programming or image analysis in order to obtain either the tools or data for our applications. We then move on to possible ways of improvements as well as topics that have not been addressed in this work but that constitute natural extensions.

Contributions and results

Design of plant growth models

The formulation of plant growth models as general state space models is a necessary preliminary step for the estimation of the parameters of these models. In particular, the presence of process and observation noises required to carefully formulate the expressions of the transition and observation probability density functions, which in turn allowed us to design generic algorithms for parameter and state inference. If the introduction of additional quantities for the deterministic and stochastic variables representing the same value, affected by either process or observation noise, might be considered redundant at first sight, it does not in fact complicate the model very much, and allows a generic formulation for all the models under study as far as the transition and observation probability density functions are concerned. The quantities in question can perfectly be multidimensional so that it is possible to design complex noise models. Population models were designed within the same framework where the dynamics of several individuals is driven by the same general state space model with different values of parameters.

In this computational framework, we designed the plant growth models on which rely our application test cases for three different species. The first one concerns sugar beet and had already been proposed by [Cournède et al. \[2013\]](#). We rewrote it in a slightly novel manner by introducing deterministic and stochastic values for the produced biomass and the allocation coefficient that involve process noise and derived the corresponding transition and observation probability density functions.

The principles of the LNAS model for sugar beet served as a basis for the design of an equivalent model for wheat. Log-normal distributions were again used for the parameterization of the allocation and senescence processes. The model is more complex than in the case of sugar beet, even though one would like to keep the number of model parameters as low as possible for an easier identifiability, and this is due to two reasons:

- first, it involves more compartments: when there are only two compartments in sugar beet, (roots, green leaves, senescent leaves), there are five compartments in wheat (roots, green leaves, yellow leaves, senescent leaves, stem and grain) because of its more complex structure. It therefore necessarily requires more parameters to describe the biological processes of interaction between the different compartments;
- second, the plant model is coupled to a soil model that is necessary for a proper modelling of the root system and to take into account hydric stress. This also complexified the model and increased the number of parameters significantly.

This model was designed with the final objective of predicting yield of wheat in different hydric conditions.

Finally, we also designed a new model for *Arabidopsis thaliana*. It was inspired by the GreenLab model approach that models the plant at the scale of the organ. The leaves are therefore not considered as a single compartment but the growth of each leaf is modelled individually and each organ has its own biomass demand function. The latter were parameterized with log-normal functions again, as they proved particularly suitable in the case of sugar beet and do not necessitate the tedious measurements of expansion durations as in the original GreenLab model [Jullien et al., 2010]. The motivation behind this organ-scale model was to make use of data potentially available for each leaf, thus summarizing information on the whole history of the functioning of the plant for a better characterization than if only the total leaf area was considered.

Bayesian inference in general state space models

If the final objective of this work was to study the variability within a population of plants, the use of these newly designed plant growth models was considered first in the case of a single individual. Adopting a Bayesian perspective, we investigated different methods for parameter and state inference, which could be separated in two main families, the Markov chain Monte Carlo (MCMC) and sequential Monte Carlo (SMC) methods. Particle Markov chain Monte Carlo (PMCMC) algorithms, combining the advantages of the two, were also considered for an accurate estimation of hidden states.

Motivated by the necessity to efficiently simulate potentially complex and heavy models as well as using estimation methods, we designed an entire computing platform for statistical modelling and inference, ADJUSTIN', written in Julia, which proves an excellent choice for the task because of its speed (competing with statically compiled languages such as C++), syntax, expressiveness and metaprogramming capabilities. It is based on the mathematical framework of general state space models, models are easy to design and require very few lines of code as most operations are handled by the simulation core and because we took advantage of Julia's capabilities for code generation. One of the main advantages of having designed our own platform

lies in the possibility to explore diverse strategies for the different algorithms such as the resampling scheme of filters or the adaptive schemes in MCMC algorithms. Most of the algorithms have been parallelized for efficient applications. This platform aims to be a unifying tool for most of the applications of interest in this work (and more globally in the Biomathematics team), which include model design, sensitivity analysis, parameter and state estimation, uncertainty analysis and data assimilation. It has already been reused by other members of the team (whether they be researchers or Ph.D. students) and should provide them with the adequate modelling and computing framework needed for their work and become a durable research tool.

ADJUSTIN⁷ being designed, we were able to explore some case studies for the different plant growth models considered in this work. Notably, we examined the use of the regularized particle filters (RPFs) for data assimilation using the newly design LNAS model for wheat. Since real data was not available yet, we worked with simulated data. It was shown that data assimilation with the RPF works very well even in the case of a few observations (which is probably going to be the case in practice). It also works rather well when the observation noise is important (i.e. when the measurements on the plant are very noisy) or when the prior distributions assigned to the model parameters are not well prescribed. In all cases, it represents a significant improvement compared to straight prediction with uncertainty analysis.

Another case study concerned the estimation of hidden states in the LNAS model for sugar beet, which involves both process and observation noises, where we compared the relevance of using different SMC methods within a particle marginal Metropolis–Hastings (PMMH) sampler for an accurate yet efficient inference. Three filters, namely the unscented Kalman filter (UKF), two ensemble Kalman filters and two regularized particle filters (with a different number of particles) were considered. Despite its relatively low number of particles, UKF performed well (slightly better than the other filters on one of the estimated states, slightly worse on some other). Its strength lies in the very low computing time and memory it requires (because of its low number of particles), and it appears to be a promising strategy for the estimation of hidden states in complex models that are time-consuming to simulate. When dealing with systems where the observation noise is much more important than the process noise, the former can be accurately estimated, whereas estimation of the process noise seems more problematic, mainly because it is drown in the higher values of the observation noises. Still, this has little impact on the posterior distributions of the model parameters, which remains our main concern.

Study of genotypic variability in a population of plants

The study of genotypic differentiation was considered through the use of Bayesian hierarchical models, where a Gibbs sampler coupled to individual adaptive schemes was used for the estimation of both individual and population parameters. Evidencing the variability within a population of plants required the acquisition of data for many individual plants. This could be achieved thanks to the Phenoscope platform of INRA Versailles [Tisné et al., 2013] which provided us with time series of zenithal images of *Arabidopsis thaliana* over 21 days for many individuals. On 4 individuals, we manually extracted data for key variables from each leaf, namely

their orientation, distance from the mass centre of the plant and area. They allowed us to better understand the growth dynamics of this species. We designed a two-step segmentation method in order to estimate the areas of the different leaves on each day of the time series. The history of each leaf was reconstructed thanks to a tracking step that took advantage of the specificities of *Arabidopsis thaliana*'s growth highlighted from data analysis. The performance of the algorithm was illustrated on these 4 individuals for which manually acquired data was available and it showed that our algorithm yielded numerous and very precise data. This algorithm was then applied to 64 individuals, providing the data necessary for the application of the Gibbs sampler designed for population models.

Bayesian hierarchical modelling was finally studied specifically in the case of the GreenLab model. Simulated data was first used in order to apprehend the behaviour of the estimation procedure for populations of plants. We studied how different variables of the context of the case study can influence the results of this estimation. Notably, how the number of individuals in the population helps refining the posterior distributions on the population parameters, the issues related to the bias for the estimation of the population precision and how the parameters of its prior distribution can be tuned. Furthermore, we highlighted how the slow convergence of the Markov chains for the precision is an indicator of ill-chosen prior distributions for the population mean and how the problem dimension affects the speed of convergence of the chains.

These studies were of great value in order to correctly apprehend the estimation of real data. The latter, obtained from image analysis, were used for the estimation of individual and population parameters within a two-step Bayesian procedure. The 24 best individuals served during a first phase for the estimation of the related individual parameters and the population parameters, and the inferred posterior distributions were then used as prior information for the estimation of 24 other individuals for which data were of lesser quality.

Perspectives

The different applications considered in this work suggested that there is still room for improvement in several respects.

Plant growth models

First, the GreenLab model for *Arabidopsis thaliana* could be refined for a better fit with experimental data. Model calibration, whether it be in the context of single individuals using generalized least squares or using Bayesian hierarchical models for a population of plants, highlighted the slight inadequacy between the resulting model simulation and experimental data for the late dynamics of the third and fourth leaves. Their growth curves reach limits too abruptly, which means that when the leaves of the second kind start to appear, too much biomass is allocated to them. The solution to this problem is not straightforward though, as we have already tried different functions to parameterize the demands of the different leaves, and the log-normal distributions used in this thesis remain the best compromise.

Data assimilation using the RPF has proved to enhance significantly model predictions for the different variables of interest. Nevertheless, its efficiency should be tested and confirmed on real data, for divergences between the LNAS model and the actual dynamics of the real plant might render the task more complicated. Ideally, one would need (at least) two real data sets: the first one would serve for an initial model calibration, and the second one for data assimilation, where only the early measurements in the second data set are used for data assimilation and the late ones for comparison with the corrected model predictions.

Computing platform

Some improvements can also be brought to the ADJUSTIN' platform, mainly from a computing point of view. Even though the efficiency of the Julia language combined to an efficient parallelization of most of the algorithms using the MPI standard has led to very low computing times, the latter could further be improved if the algorithms were designed using graphics processing units (GPU). There are several Julia packages available¹ to do so and computations could be launched on the GPU nodes of CentraleSupélec's computing cluster. In particular, very recent developments have led to the delivery of the `CUDAnative` package², which allows to write CUDA kernels directly in Julia in order to take advantage of its high-level language features with high GPU performance which compares to the CUDA C implementation (a slight overhead still exist because of argument passing). The transposition of all our algorithms to a GPU implementation should benefit from these aspects and not require the high development time that would be needed in C. One must remain careful however, as this is a relatively new implementation that still might contain its share of unexpected behaviours.

The development of very efficient GPU algorithms would be particularly interesting for cumbersome estimation methods in the case of complex models that require a lot of time to simulate, as can be the case for data assimilation using the LNAS model for wheat, or when applying PMCMC methods even to simple models. This would be all the more relevant for the population framework as the number of individuals in the population increases. If sufficient data becomes available in the future, ideally hundreds of individuals, the overall computing time for estimation within the population might become unsustainably high. Furthermore, as is emphasized at the end of this section, the use of mixture models adds an additional complexity which would further increase the computing time required for such applications. All these reasons call for the consideration of GPU implementations in ADJUSTIN'.

Image analysis

As previously mentioned, some technical difficulties related to image analysis should be solved for better automation of the method. This includes potentially large rotations of the pots and dealing with two plants growing side by side during the first days. However, these issues concerned only a fraction of all the individuals and remain anecdotal. The most crucial problem that needs to be addressed is that of the detection of the

¹<https://github.com/JuliaGPU>

²<https://github.com/JuliaGPU/CUDAnative.jl>

5th leaf: if this step fails, then chances are that data will only be available for the first four leaves. This already represents an important source of information but we fail to take advantage of observations on the system crucial for the estimation of model parameters intervening in the late growth stages, such as those parameterizing the demand functions for the leaves of the second kind. A current solution to this issue is to specify, for the problematic individuals, the day of emergence of the 5th leaf, and potentially its direction. We would like to free ourselves from this semi-automatic procedure with the prospect of applying this algorithm to possibly hundreds of individuals in a perfectly automatic way.

Bayesian hierarchical models for genotypic differentiation

In hierarchical population models, an interesting issue is to separate the fixed effects from the random effects in the model, that is to say the parameters that can be considered constant in the population from those that vary among individuals. The posterior distributions for the population parameters η and Σ constitute a first assessment of whether some model parameters can be considered constant throughout the population. More rigorous conclusions require the use of hypothesis testing. This issue has been recently tackled by [Baey et al. \[2017\]](#) in a frequentist approach when parameters are estimated with maximum likelihood. It would be interesting to see how to handle this issue in the Bayesian framework.

The choice of prior distributions for the population parameters also requires more insight. Notably, the hyperparameters of the Wishart distribution for the inverse covariance matrix could be set differently; several choices were discussed in [\[Bouriga and Féron, 2013\]](#). Furthermore, a number of limitations related to the prescription of inverse Wishart prior distributions have recently been illustrated (lack of flexibility for the different matrix components' uncertainty, dependency between the correlations and the variances) [\[Alvarez et al., 2014\]](#) and some other approaches, such as a hierarchical Half-t prior or a separation strategy based on the works of [Barnard et al. \[2000\]](#), should be investigated.

As far as MCMC methods are concerned, whether it be for the case of a single individual or a population, more efficient approaches should be studied, notably the use of Metropolis adjusted Langevin algorithm (MALA) [\[Roberts and Rosenthal, 1998\]](#) or Hamiltonian Monte Carlo (HMC) [\[Duane et al., 1987\]](#) [\[Betancourt, 2017\]](#) for optimized state space exploration. Such procedures become all the more interesting as the problem dimension increases. First attempts to combine HMC with hierarchical models has been done in [\[Girolami and Betancourt, 2015\]](#).

Towards mixture models

This work paves the way for the use of mixture models for genotypic differentiation in plant populations. The need for the use of mixture models emerges rather naturally when one expects that homogeneous subgroups or clusters are present in a population. They have become useful tools in many research areas such as genetic and medical research [\[Lewin et al., 2007\]](#), [\[White et al., 2012\]](#), ecology [\[Joseph et al., 2009\]](#) or image analysis [\[Layer et al., 2015\]](#) just to name a few examples.

So far, the individual parameters $(\theta_i)_{i \in [1, N]}$ were assumed to be sampled from a normal distribution with mean η and covariance matrix Σ . When studying a population of plants with different genotypes, this assumption might be insufficient as the parameters related to different genotypes may belong to different clusters. They are therefore better represented by a mixture of distributions $\theta_i \sim \sum_{k=1}^K w_k D_k$ where K is the number of distributions considered and $w = (w_k)_{k \in [1, K]}$ are the weights associated to each distribution and such that $\sum_k w_k = 1$. Usually, the distributions are chosen to be Gaussian with respective mean and covariance η_k and Σ_k . The mathematical framework introduced in Chapter 4 can therefore be generalized as follows:

$$\begin{aligned}
 \text{First stage:} \quad & y_i \sim \mathcal{N}(h_i(\theta_i), \tau^{-1} \Omega_i) \\
 \text{Second stage:} \quad & \theta_i \sim \sum_{k=1}^K w_k \mathcal{N}(\eta_k, \Sigma_k) \\
 \text{Third stage:} \quad & \begin{cases} \eta_k \sim \mathcal{N}(\lambda_k, \Lambda_k) \\ \Sigma_k^{-1} \sim \mathcal{W}(q_k, \Psi_k) \\ \tau \sim \mathcal{G}(\alpha, \beta) \\ w \sim \mathcal{D}(\gamma) \end{cases}
 \end{aligned} \tag{8.24}$$

where \mathcal{D} designates the Dirichlet distribution, indexed by $\gamma = (\gamma_k)_{k \in [1, K]}$, which is the conjugate prior for the multinomial distribution. The analysis of this mixture model now involves the calculation of the posterior distribution:

$$p(w_{1:K}, \eta_{1:K}, \Sigma_{1:K}, \tau | y_{1:N}). \tag{8.25}$$

The introduction of latent variables $z_{1:N}$ indicating to which distributions a parameter belongs allows to rewrite the problem in a more convenient form:

$$\begin{cases} \theta_i \sim \mathcal{N}(\eta_{z_i}, \Sigma_{z_i}) \\ z_i \sim \mathcal{C}(w) \\ w \sim \mathcal{D}(\gamma) \end{cases} \tag{8.26}$$

where \mathcal{C} designates the categorical distribution. Just as described in Chapter 4, it is possible to calculate the full conditional distributions of all these random variables but the individual parameters $(\theta_i)_{i \in [1, N]}$. The full conditional distribution of each of the η_k , Σ_k , τ and w belongs to the same family of distribution as their prior with updated hyperparameters whose expression can be found, for instance, in [Tatarinova and Schumitzky, 2015] and a Gibbs sampler can again be used for parameter inference.

With a number of components $K > 1$, new issues emerge [Havre et al., 2015], such as assessing the optimal number of components, label switching and trapping. It has been shown (see [Celeux et al., 2000] for instance) that even though the Gibbs sampler is supposed to converge in theory, since the MCMC chain is irreducible and should explore all the regions of the posterior distribution, this is seldom the case in practice as it gets stuck in one of the $K!$ modes and escaping from this mode might require a huge number of iterations. This effect is even more pronounced when one of the components has very few observations [Gilks et al., 1996] or when non-informative priors are used. Several methods have been designed to ensure proper mixing

in such multimodal situations such as parallel tempering [Earl and Deem, 2005].

Label switching arises because of the permutation invariance of the likelihood under relabelling of the mixture components [Stephens, 2000]. This has the following important consequence: the marginal posterior distributions of the parameters, weights and labels for the different components are identical, which can lead to highly symmetric and multimodal posterior distributions. Paradoxically, label switching is necessary for the convergence of the MCMC algorithm as it should ideally visit all the $K!$ symmetric modes of the posterior distribution. However, the standard Gibbs sampler may fail to visit all the modes, in particular when they are well separated, and becomes trapped in one of them. Several solutions have been proposed to resolve this problem, such as random permutations of the labels [Richardson and Green, 1997], [Frühwirth-Schnatter, 2006], minimizing a loss function [Nobile and Fearnside, 2007], [Grün and Leisch, 2009] or probabilistic relabelling [Yao, 2012].

Finally, assessing the optimal number of components K is essential for mixture models [Marin et al., 2005]. It might be achieved based on the reductive stepwise method [Sahu and Cheng, 2003] where K is iteratively reduced, by collapsing two of the K components at each iteration until data fitting is no longer satisfying, which can be assessed using a weighted Kullback–Leibler distance between two iterations [Tatarinova and Schumitzky, 2015], or Dirichlet process mixtures [Koutroumpas et al., 2016] that automatically select the best number of components.

Final remarks

This thesis represents an original attempt to a full Bayesian approach to analyze the intergenotypic or interindividual variability in populations of plants, which is achieved by using mechanistic models of plant growth. We believe that the Bayesian approach is the most relevant to study biological systems for which experimental knowledge is usually available and can be used to define adequate priors. Furthermore, the Bayesian approach allows a strict representation of estimation uncertainty which can be later used to properly master risk in the decision theory framework [Robert, 2007]. Finally, the potential applications of this research, which lies at the crossroads of several disciplines, are of different kinds.

The first one still derives from our Bayesian perspective: having identified the population parameters, when studying a new genotype (for example in breeding), the calibration process is simplified and the accuracy increased by incorporating the knowledge previously acquired on the population distribution into the prior. This fact was exemplified in our study when we used a two-step procedure by first using the best data to estimate the population distribution for 24 individuals and then use this distribution with 24 other individuals for which data were of lesser quality.

The second application concerns genetic improvement: by comparing the variability of parameters in a population, we are able to select the parameters for which there is more potential for improvement. Likewise, the knowledge of the full distributions also helps in the design of ideotypes by providing a better representation of the space to explore.

Last but not least, mixture models could help identify several subfamilies. As a consequence, we may for instance be able to improve phylogenetic knowledge so as to better understand the genetic mutations that have occurred. An example of such an application would be to identify subfamilies of wild genotypes originating from different habitats.

Appendix A

Calculation of full conditional distributions

The purpose of this appendix is to provide the details of the calculations leading to the formulas for the update of the population parameters: the population mean vector η , the population inverse covariance matrix Σ^{-1} and the population precision τ . We recall that there are N individuals, n_i observations for the i -th individual and that the total number of observations is denoted as n_{tot} and the dimension of the problem (i.e. the number of estimated parameters) as d . The hierarchical model reads:

$$\begin{aligned}
 \text{First stage:} \quad & y_i \sim \mathcal{N}(h_i(\theta_i), \tau^{-1}\Omega_i) \\
 \text{Second stage:} \quad & \theta_i \sim \mathcal{N}(\eta, \Sigma) \\
 \text{Third stage:} \quad & \begin{cases} \eta \sim p(\eta) \\ \Sigma^{-1} \sim p(\Sigma^{-1}) \\ \tau \sim p(\tau) \end{cases}
 \end{aligned} \tag{A.1}$$

and the associated joint distribution:

$$p(y_{1:N}, \theta_{1:N}, \eta, \Sigma, \tau) = \prod_{i=1}^N p(y_i | h_i(\theta_i), \tau^{-1}\Omega_i) \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\eta) p(\Sigma) p(\tau). \tag{A.2}$$

To obtain the full conditional distribution of a given random variable z , it suffices to use Bayes' formula:

$$p(z | \dots) = p(\dots)^{-1} p(z, \dots). \tag{A.3}$$

Collecting all terms involving z in $p(z, \dots)$ will therefore lead to the desired distribution. Appropriate conjugate prior distributions must be chosen to retrieve analytical expressions of known distributions.

A.1 Population mean vector

The full conditional distribution of the population mean vector is:

$$p(\eta | \dots) \propto_{\eta} \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\eta) = \prod_{i=1}^N f_{\mathcal{N}}(\theta_i, \eta, \Sigma) p(\eta). \tag{A.4}$$

The appropriate conjugate prior for a multivariate normal distribution with known covariance is a normal distribution denoted $\mathcal{N}(\lambda, \Lambda)$, i.e.:

$$p(\eta) = f_{\mathcal{N}(\lambda, \Lambda)}(\eta) = |2\pi\Lambda|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\eta - \lambda)^T \Lambda^{-1}(\eta - \lambda)\right). \quad (\text{A.5})$$

Then:

$$\begin{aligned} p(\eta | \dots) &\propto_{\eta} \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\eta) \\ &\propto_{\eta} \exp\left(-\frac{1}{2} \sum_{i=1}^N (\theta_i - \eta)^T \Sigma^{-1} (\theta_i - \eta) + \frac{1}{2} (\eta - \lambda)^T \Lambda^{-1} (\eta - \lambda)\right) \\ &\propto_{\eta} \exp\left(-\frac{1}{2} \eta^T \left(\Sigma^{-1} \left(\sum_{i=1}^N 1\right) + \Lambda^{-1}\right) \eta + \frac{1}{2} \eta^T \left(\Sigma^{-1} \left(\sum_{i=1}^N \theta_i\right) + \Lambda^{-1} \lambda\right) \right. \\ &\quad \left. + \frac{1}{2} \left(\Sigma^{-1} \left(\sum_{i=1}^N \theta_i\right) + \Lambda^{-1} \lambda\right)^T \eta\right) \end{aligned} \quad (\text{A.6})$$

and since $\sum_{i=1}^N 1 = N$ and defining $\theta^* = N^{-1} \sum_{i=1}^N \theta_i$, $\Lambda^* = (N\Sigma^{-1} + \Lambda^{-1})^{-1}$ and λ^* such that:

$$\eta^T \left(\Sigma^{-1} \left(\sum_{i=1}^N \theta_i\right) + \Lambda^{-1} \lambda\right) = \eta^T \Lambda^{*-1} \lambda^* \quad (\text{A.7})$$

i.e. $\lambda^* = \Lambda^*(N\Sigma^{-1}\theta^* + \Lambda^{-1}\lambda)$, then:

$$p(\eta | \dots) \propto_{\eta} \exp\left(-\frac{1}{2} (\eta - \lambda^*)^T \Lambda^{*-1} (\eta - \lambda^*)\right) \quad (\text{A.8})$$

which shows that:

$$\begin{cases} \eta &\sim \mathcal{N}(\cdot | \lambda^*, \Lambda^*), \\ \lambda^* &= (N\Sigma^{-1} + \Lambda^{-1})^{-1} (N\Sigma^{-1}\theta^* + \Lambda^{-1}\lambda), \\ \Lambda^* &= (N\Sigma^{-1} + \Lambda^{-1})^{-1}. \end{cases} \quad (\text{A.9})$$

A.2 Population inverse covariance matrix

The full conditional distribution of the population inverse covariance matrix is:

$$p(\Sigma | \dots) \propto_{\Sigma} \prod_{i=1}^N p(\theta_i | \eta, \Sigma) p(\Sigma) = \prod_{i=1}^N f_{\mathcal{N}}(\theta_i, \eta, \Sigma) p(\Sigma). \quad (\text{A.10})$$

The appropriate conjugate prior for a multivariate normal distribution with known mean is a Wishart distribution denoted $\mathcal{W}_d(q, \Psi)$, i.e.:

$$p(\Sigma) = f_{\mathcal{W}_d(q, \Psi)}(\Sigma) = 2^{-\frac{qd}{2}} |\Psi|^{-\frac{q}{2}} \Gamma_d\left(\frac{q}{2}\right)^{-1} |\Sigma|^{\frac{q-d-1}{2}} e^{-\frac{1}{2} \text{tr}(\Psi^{-1}\Sigma)}. \quad (\text{A.11})$$

Then:

$$\begin{aligned}
 p(\Sigma^{-1}|\dots) &\propto_{\Sigma^{-1}} \prod_{i=1}^N p(\theta_i|\eta, \Sigma) p(\Sigma) \\
 &\propto_{\Sigma^{-1}} |\Sigma|^{-N/2} \exp\left(-\frac{1}{2} \sum_{i=1}^N (\theta_i - \eta)^T \Sigma^{-1} (\theta_i - \eta)\right) \\
 &\quad 2^{-\frac{qd}{2}} |\Psi|^{-\frac{q}{2}} \Gamma_d\left(\frac{q}{2}\right)^{-1} |\Sigma^{-1}|^{\frac{q-d-1}{2}} \exp\left(-\frac{1}{2} \text{tr}(\Psi^{-1} \Sigma^{-1})\right).
 \end{aligned} \tag{A.12}$$

We recall that for a vector x and a matrix M , $x^T M x = \text{tr}(x x^T M)$, hence:

$$\begin{aligned}
 p(\Sigma^{-1}|\dots) &\propto_{\Sigma^{-1}} |\Sigma|^{-N/2} \exp\left(-\frac{1}{2} \sum_{i=1}^N \text{tr}((\theta_i - \eta)(\theta_i - \eta)^T \Sigma^{-1})\right) \\
 &\quad |\Sigma^{-1}|^{\frac{q-d-1}{2}} \exp\left(-\frac{1}{2} \text{tr}(\Psi^{-1} \Sigma^{-1})\right) \\
 &\propto_{\Sigma^{-1}} |\Sigma^{-1}|^{\frac{q+N-d-1}{2}} \exp\left(-\frac{1}{2} \text{tr}\left(\left(\Psi^{-1} + \sum_{i=1}^N (\theta_i - \eta)(\theta_i - \eta)^T\right) \Sigma^{-1}\right)\right)
 \end{aligned} \tag{A.13}$$

so that:

$$\left\{ \begin{array}{l} \Sigma^{-1} \sim \mathcal{W}(\cdot|q^*, \Psi^*), \\ q^* = q + N, \\ \Psi^* = (\Psi^{-1} + \sum_{i=1}^N (\theta_i - \eta)(\theta_i - \eta)^T)^{-1}. \end{array} \right. \tag{A.14}$$

A.3 Population precision

The full conditional distribution of the population precision is:

$$p(\tau|\dots) \propto_{\tau} \prod_{i=1}^N p(y_i|h_i(\theta_i), \tau^{-1}\Omega_i) p(\tau) = \prod_{i=1}^N f_{\mathcal{N}}(y_i, h_i(\theta_i), \tau^{-1}\Omega_i) p(\tau). \tag{A.15}$$

The appropriate conjugate prior for a univariate normal distribution with known mean is a Gamma distribution denoted $\mathcal{G}(\alpha, \beta)$, i.e.:

$$p(\tau) = f_{\mathcal{G}(\alpha, \beta)}(\tau) = \tau^{\alpha-1} \beta^{\alpha} \Gamma(\alpha)^{-1} \exp(-\beta\tau). \tag{A.16}$$

Then:

$$\begin{aligned}
 p(\tau|\dots) &\propto_{\tau} \prod_{i=1}^N p(y_i|h_i(\theta_i), \tau^{-1}\Omega_i) p(\tau) \\
 &\propto_{\tau} |\tau^{-1}\Omega_i|^{-N/2} \exp\left(-\frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \tau \Omega_i^{-1} (y_i - h_i(\theta_i))\right) \tau^{\alpha-1} \beta^{\alpha} \Gamma(\alpha)^{-1} \exp(-\beta\tau) \\
 &\propto_{\tau} \tau^{\alpha+\frac{1}{2} \sum_{i=1}^N n_i - 1} \exp\left(-\left(\beta + \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i))\right) \tau\right)
 \end{aligned}$$

which shows that:

$$\left\{ \begin{array}{l} \tau \sim \mathcal{G}(\cdot | \alpha^*, \beta^*), \\ \alpha^* = \alpha + n_{\text{tot}}/2, \\ \beta^* = \beta + \frac{1}{2} \sum_{i=1}^N (y_i - h_i(\theta_i))^T \Omega_i^{-1} (y_i - h_i(\theta_i)). \end{array} \right. \quad (\text{A.17})$$

Appendix B

LNAS model for sugar beet in ADJUSTIN'

Listing 14 The LNAS model for sugar beet in ADJUSTIN'

```
type Extern
  vec_t::Vector      ## vector of temperatures for all days
  vec_r::Vector      ## vector of radiations for all days
end

type Parameters
  q_0::Float64       ## initial biomass
  tau_init::Float64  ## initiation thermal time
  mu::Float64        ## radiation use efficiency
  k::Float64         ## Beer-Lambert coefficient
  e::Float64         ## leaf mass per area
  gamma_0::Float64   ## initial leaf allocation coefficient
  gamma_l::Float64   ## final leaf allocation coefficient
  mu_a::Float64      ## allocation median
  sigma_a::Float64   ## allocation standard deviation
  mu_s::Float64      ## senescence median
  sigma_s::Float64   ## senescence standard deviation
  tau_s::Float64     ## senescence thermal time delay
end

type State
  q_det::Float64     ## deterministic produced biomass
  q_sto::Float64     ## stochastic produced biomass
  gamma_det::Float64 ## deterministic leaf allocation coefficient
  gamma_sto::Float64 ## stochastic leaf allocation coefficient
  q_l::Float64       ## leaf biomass
  q_gl::Float64      ## green leaf biomass
  q_r::Float64       ## root biomass
end
```

```

type ProcessNoise
  q::Noise          ## process noise for the production on q
  gamma::Noise      ## process noise for the allocation on gamma
end

type ObservationNoise
  q_gl::Noise       ## observation noise for green leaf biomass
  q_r::Noise        ## observation noise for root biomass
end

function production!(n, xn, u, p, xnplus1)
  xnplus1.q_det = u.vec_r[n] * p.mu * (1 - exp(-p.k * xn.q_gl / p.e))
end

## here go the other modules

function f(n, xn, u, p, mn)
  xnplus1 = State(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
  thermal_time!(n, xn, u, p, xnplus1)
  production!(n, xn, u, p, xnplus1)
  noise!(xnplus1, mn.q)
  allocation!(n, xn, u, p, xnplus1)
  noise!(xnplus1, mn.gamma)
  senescence!(n, xn, u, p, xnplus1)
  update_total_leaf_biomass!(n, xn, u, p, xnplus1)
  update_green_leaf_biomass!(n, xn, u, p, xnplus1)
  update_root_biomass!(n, xn, u, p, xnplus1)
  return xnplus1
end

ofl = [obs_q_gl(x, on) = noise(x, on.q_gl),
       obs_q_r(x, on) = noise(x, on.q_r)]

@observation_function()

```

Appendix C

UKF algorithm in ADJUSTIN'

Listing 15 The UKF algorithm in ADJUSTIN'

```
function ukf(f, g, x0, u, p, so_exp, cfg; mn = Void, on = Void)

    ll_p = get_label_list(cfg.srl_p)
    ll_x = get_label_list(cfg.srl_x)
    ll_px = [ll_p; ll_x]

    so_p_pr_mean = SystemObservation(ll_p; title = "parameters_prediction_mean")
    so_p_pr_stdev = SystemObservation(ll_p; title = "parameters_prediction_stdev")
    so_p_up_mean = SystemObservation(ll_p; title = "parameters_update_mean")
    so_p_up_stdev = SystemObservation(ll_p; title = "parameters_update_stdev")
    so_x_pr_mean = SystemObservation(ll_x; title = "states_prediction_mean")
    so_x_pr_stdev = SystemObservation(ll_x; title = "states_prediction_stdev")
    so_x_up_mean = SystemObservation(ll_x; title = "states_update_mean")
    so_x_up_stdev = SystemObservation(ll_x; title = "states_update_stdev")

    nbr_p = length(ll_p)
    nbr_x = length(ll_x)
    nbr_mn = length(fieldnames(mn))
    nbr_px = nbr_p + nbr_x
    nbr_pxmn = nbr_p + nbr_x + nbr_mn
    nbr_particles = 2 * nbr_pxmn + 1

    pl = sample(p, cfg.srl_p, nbr_particles)
    xl = sample(x0, cfg.srl_x, nbr_particles)
    mat_px = to_mat(pl, ll_p, xl, ll_x)
    tml = get_merged_timeline(g)

    for k = 1:length(tml)

        ti, tf = k > 1 ? tml[k-1] : 0, tml[k]
        vec_px_mean, mat_px_cov = mean_and_covariance(mat_px)
        mat_p_cov = mat_px_cov[1:nbr_p,1:nbr_p]
```

```

vec_p_mean = vec_px_mean[1:nbr_p]
mat_x_cov = mat_px_cov[nbr_p+1:end,nbr_p+1:end]
vec_x_mean = vec_px_mean[nbr_p+1:end]

register!(so_p_pr_mean, ll_p, tf, vec_p_mean)
register!(so_p_pr_stdev, ll_p, tf, sqrt(diag(mat_p_cov)))
register!(so_x_pr_mean, ll_x, tf, vec_x_mean)
register!(so_x_pr_stdev, ll_x, tf, sqrt(diag(mat_x_cov)))

vec_sp_mean = vcat(vec_px_mean, zeros(nbr_mn))
mat_mn_cov = covariance(mn)
mat_sp_cov = cat([1,2], mat_px_cov, mat_mn_cov)
spl, wl_mean, wl_cov = sigma_points_and_weights(vec_sp_mean, mat_sp_cov)
fill_with_subset!(pl, xl, spl, ll_p, ll_x)
sol = next_prediction!(f, g, xl, u, pl, ti; mn = mn, on = on)
mat_px = to_mat(pl, ll_p, xl, ll_x)
mat_y = to_mat(sol)
vec_px_mean, mat_px_cov = weighted_mean_and_covariance(wl_mean, wl_cov, mat_px)
vec_y_mean, mat_y_cov = weighted_mean_and_covariance(wl_mean, wl_cov, mat_y)
mat_pxy_cor = weighted_correlation(wl_mean, wl_cov, mat_px, mat_y)
mat_gain = mat_pxy_cor * inv(mat_y_cov)
vec_y_exp = to_vec(so_exp, tf)
vec_px_mean += mat_gain * (vec_y_exp - vec_y_mean)
mat_px_cov -= mat_gain * mat_y_cov * mat_gain'
mat_p_cov = mat_px_cov[1:nbr_p,1:nbr_p]
vec_p_mean = vec_px_mean[1:nbr_p]
mat_x_cov = mat_px_cov[nbr_p+1:end,nbr_p+1:end]
vec_x_mean = vec_px_mean[nbr_p+1:end]

register!(so_p_up_mean, ll_p, tf, vec_p_mean)
register!(so_p_up_stdev, ll_p, tf, sqrt(diag(mat_p_cov)))
register!(so_x_up_mean, ll_x, tf, vec_x_mean)
register!(so_x_up_stdev, ll_x, tf, sqrt(diag(mat_x_cov)))

end

r = ResultUkf(so_p_pr_mean, so_p_up_mean, so_p_pr_stdev, so_p_up_stdev,
             so_x_pr_mean, so_x_up_mean, so_x_pr_stdev, so_x_up_stdev)

end

```

Appendix D

Database and results directories

As discussed in Section 5.3, the model `lnas` is loaded inside the application with the `load_model` function. A similar procedure instantiates the `State`, `Extern` and `Parameters` (the latter jointly with `ProcessNoise` and `ObservationNoise`) types using the respective flags `-x`, `-c`, `-p`. This requires that these objects be defined inside files stored in the `database/` directory, before being loaded with, for instance, `load_parameters("lnas", p0)` where it is implied that the definition `p = Parameters(...)` is done within the file `database/parameters/p0.jl`. The `.jl` extension is needed so that Julia automatically interprets this file and its content. It has to be noted, once again, that the instance of the `Parameters` needs to be called `p` and not something else for further reuse in the remaining of the application file. This could seem to be a lot of over-organization: defining objects in separate files rather than directly in a script where they are to be used for one run. It actually is much more convenient and saves a lot of time on the long run:

- first, for models with a lot of variables, it is more convenient to separate the different type definitions in different files;
- second, this approach will not be used only for the simulation of the model, but for all other applications, ranging from sensitivity analysis to different parameter and state estimation procedures; Potentially, a dozen of algorithms could be run for a given model, which is why it becomes interesting to use predefined applications: therefore, once one has coded the model and defined the necessary and appropriate states, external variables and parameters in the database, one can directly make simulations of the model and run several algorithms using it without a single line of code;
- third, for the study of subjects such as genotypic differentiation, the values of the parameters will differ from one genotype to another. Since there can be tens or hundreds of different genotypes for a given species, it makes sense to be able to (i) create the parameters files using a simple (Julia) script or (ii) reuse these parameters files for different simulations and estimations;
- fourth, each time that an application is run, its results are saved in the root directory `results/` under the name:

`$year-$month-$day-$hour-$minute-$second-$optional-name/`

(where `optional-name` is provided by the `-d` flag). The results of an application can be a single `SystemObservation` for a simulation or a more complex one when estimation algorithms are run (a

specific result type containing several `SystemObservations` for the mean and standard deviations of the estimated parameters and states, for instance). Not only the results are saved, but the state, external variables, parameters files used during the application are copied to the results directory. This ensures that, for a given results directory, it is known exactly what program was run with what files. It is a guarantee of reproducibility: even if the database files are modified, their versions used for a given run are saved as they are, and an experiment can be run in the exact same conditions again.

Appendix E

Parallel computation of mean and covariance

Let $X = (x_j^i) \in \mathcal{M}_{d,N}(\mathbb{R})$ be the matrix containing the values of parameters and states to be estimated for the different particles, with $j \in \llbracket 1, d \rrbracket$ and $i \in \llbracket 1, N \rrbracket$. In the MPI paradigm, this matrix is distributed among the n_p different processes. The range of indices relative to the process p is $r^{(p)} = \llbracket 1 + (p-1)N^p, pN^p \rrbracket$. Let $m_j^{(p)} = \sum_{i \in r^{(p)}} w_i x_j^i$ be the local mean of variable j for process p . The global mean for variable j can be expressed as:

$$m_j = \sum_{i=1}^N w^i x_j^i = \sum_{p=1}^{n_p} \sum_{i \in r^{(p)}} w^i x_j^i = \sum_{p=1}^{n_p} m_j^{(p)}. \quad (\text{E.1})$$

All the same, if

$$s = \sum_{i=1}^N (w^i)^2 = \sum_{p=1}^{n_p} \left(\sum_{i \in r^{(p)}} (w^i)^2 \right) \quad (\text{E.2})$$

the global covariance between variables j and k can be expressed as:

$$\begin{aligned} c_{jk} &= \frac{1}{1-s} \sum_{i=1}^N w^i (x_j^i - m_j)(x_k^i - m_k) \\ &= \frac{1}{1-s} \sum_{p=1}^{n_p} \sum_{i \in r^{(p)}} w^i (x_j^i - m_j^{(p)} + m_j^{(p)} - m_j)(x_k^i - m_k^{(p)} + m_k^{(p)} - m_k) \\ &= \frac{1}{1-s} \sum_{p=1}^{n_p} (\alpha_{jk}^{(p)} + \beta_{jk}^{(p)}) \end{aligned} \quad (\text{E.3})$$

with:

$$\begin{cases} \alpha_{jk}^{(p)} &= \sum_{i \in r^{(p)}} w^i (x_j^i - m_j^{(p)})(x_k^i - m_k^{(p)}) \\ \beta_{jk}^{(p)} &= \left(\sum_{i \in r^{(p)}} w^i \right) (m_j^{(p)} - m_j)(m_k^{(p)} - m_k) \end{cases} \quad (\text{E.4})$$

and where the computation of both $\alpha_{jk}^{(p)}$ and $\beta_{jk}^{(p)}$ can be parallelized.

Appendix F

Exchange of particles between processes

First the different lists of weights are gathered across the different processes with:

```
w1 = MPI.Allgather(w1_p, MPI.COMM_WORLD)
```

This is necessary to perform multinomial resampling for all the particles. A global list of indices $[\iota^i]_{1:n_s}$ is then generated on all processes using residual or systematic resampling [Douc and Cappe, 2005]. The creation of this list of indices is done in such a way that there will be no need to copy the list of Parameters or States thereafter. Once this list has been computed, the particles need to actually be resampled. In a sequential program, that would mean looping over the list of indices and do:

$$\begin{cases} \theta^i = \theta^{\iota^i} \\ x^i = x^{\iota^i} \end{cases} \text{ for } i \in \llbracket 1, n_s \rrbracket.$$

However, there is no guarantee, for a given process p and $i \in r_p$, that $\iota^i \in r_p$. Potentially, each process can require particles from each of the other $n_p - 1$ processes. If an element of $[\iota^{i,(p)}]_{1:n_s^{(p)}}$ belongs to r_q with $q \neq p$, then its value is added to a `Dict{Int, Array{Int}}` d_{pq} . The keys of d_{pq} are the indices of the particles that need to be sent from q to p , and the value associated to a key represent the indices of $[\iota^{i,(p)}]_{1:n_s^{(p)}}$ at which the key particle appears.

In practice, d_{pq} seldom contains elements as two distant processes have few chances to exchange particles. This approach allows to minimize the number of communications between processes. For all non-empty dictionaries d_{pq} , an `MPI.Irecv!` and an `MPI.Isend` requests are initiated and used to send from q to p the adequate particles in Matrix form. As far as Parameters are concerned, only the values of the estimated ones are required, but the whole State content needs to be sent: if only the values of the estimated hidden states were sent, this could make lose the consistency of a State as some states computed from two different Parameters would end up in the same State.

Once the particles have been exchanged between the different processes, the actual resampling operation can be done: for each element of $[\iota^{i,(p)}]_{1:n_s^{(p)}}$, either it belongs to r_p and $\theta^i = \theta^{\iota^i}$, $x^i = x^{\iota^i}$ can be done directly, or it belongs to r_q with $q \neq p$ and has to be retrieved from the exchanged particles. At the end of the resampling operation, the weights are reset uniformly.

Bibliography

- Aitken, A. C. (1936). 'On least squares and linear combination of observations'. In: *Proceedings of the Royal Society of Edinburgh* 55, 42–48.
- Aksoy, E. E., Abramov, A., Wörgötter, F., Scharr, H., Fischbach, A. and Dellen, B. (2015). 'Modeling leaf growth of rosette plants using infrared stereo image sequences'. In: *Computers and Electronics in Agriculture* 110, pp. 78–90.
- Allard, R. W. and Bradshaw, A. D. (1964). 'Implications of genotype-environmental interactions in applied plant breeding'. In: *Crop science* 4.5, pp. 503–508.
- Alvarez, I., Niemi, J. and Simpson, M. (2014). 'Bayesian inference for a covariance matrix'. In: Anderson, B. and Moore, J. (1979). *Optimal filtering*. Englewood Cliffs, USA: Prentice Hall.
- Andrews, D. F. and Mallows, C. L. (1974). 'Scale mixtures of normal distributions'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 36.1, pp. 99–102.
- Andrieu, C. and Roberts, G. O. (2009). 'The pseudo-marginal approach for efficient Monte Carlo computations'. In: *The Annals of Statistics* 37.2, pp. 697–725.
- Andrieu, C. and Thoms, J. (2008). 'A tutorial on adaptive MCMC'. In: *Statistics and Computing* 18.4, pp. 343–373.
- Andrieu, C., Doucet, A. and Holenstein, R. (2010). 'Particle Markov chain Monte Carlo methods'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 72.3, pp. 269–342.
- Apelt, F., Breuer, D., Nikoloski, Z., Stitt, M. and Kragler, F. (2015). 'Phytyping4d: a light-field imaging system for non-invasive and accurate monitoring of spatio-temporal plant growth'. In: *The Plant Journal* 82.4, pp. 693–706.
- Araus, J. L. and Cairns, J. E. (2014). 'Field high-throughput phenotyping: the new crop breeding frontier'. In: *Trends in Plant Science* 19.1, pp. 52–61.
- Baey, C., Cournède, P.-H. and Kuhn, E. (2017). 'Likelihood ratio test for variance components in nonlinear mixed effects models'. Submitted.
- Baey, C. (2014). 'Modélisation de la variabilité inter-individuelle dans les modèles de croissance de plantes et sélection de modèles pour la prévision'. PhD thesis. École Centrale Paris.
- Baey, C., Didier, A., Sébastien, L., Maupas, F. and Cournède, P.-H. (2013). 'Modelling the interindividual variability of organogenesis in sugar beet populations using a hierarchical segmented model'. In: *Ecological Modelling* 263, pp. 56–63.

- Baey, C., Trevezas, S. and Cournède, P.-H. (2016). 'A non linear mixed effects model of plant growth and estimation via stochastic variants of the EM algorithm'. In: *Communications in Statistics - Theory and Methods* 45.6, pp. 1643–1669.
- Baldazzi, V., Bertin, N., Génard, M., Gautier, H., Desnoues, E. and Quilot-Turion, B. (2016). 'Challenges in integrating genetic control in plant and crop models'. In: *Crop Systems Biology*. Springer, pp. 1–31.
- Barnard, J., McCulloch, R. and Meng, X. (2000). 'Modeling covariance matrices in terms of standard deviations and correlations, with application to shrinkage'. In: *Statistica Sinica* 10.4, pp. 1281–1311.
- Barthélémy, D., Edelin, C. and Hallé, F. (1989). 'Architectural concepts for tropical trees'. In: *Tropical forests. Botanical dynamics, speciation and diversity*. Ed. by Holm Nielsen, L., Nielsen, I. and Balslev, H. Academic Press, pp. 89–100.
- Bayol, B. (2016). 'Système informatique d'aide à la modélisation mathématique basé sur un langage de programmation dédié pour les systèmes dynamiques discrets stochastiques. Application aux modèles de croissance de plantes.' PhD thesis. Paris Saclay.
- Beal, S. L. and Sheiner, L. B. (1982). 'Estimating population kinetics'. In: *Critical reviews in biomedical engineering* 8, pp. 195–222.
- Beaumont, M. A. (2003). 'Estimation of population growth or decline in genetically monitored populations'. In: *Genetics* 164.3, pp. 1139–1160.
- Bennett, J. E., Racine-Poon, A. and Wakefield, J. C. (1996). 'Markov chain Monte Carlo for nonlinear hierarchical models'. In: *Markov chain Monte Carlo in practice*. Ed. by Gilks, W., Richardson, S. and Spiegelhalter, D. London, UK: Chapman and Hall, pp. 339–357.
- Berg, B. A. and Billoire, A. (2007). 'Markov chain Monte Carlo simulations'. In: *Wiley Encyclopedia of Computer Science and Engineering*. John Wiley and Sons, Inc.
- Betancourt, M. (2017). *A Conceptual Introduction to Hamiltonian Monte Carlo*.
- Beyer, R., Letort, V. and Cournède, P.-H. (2014). 'Modeling tree crown dynamics with 3D partial differential equations'. In: *Frontiers in plant science* 5, p. 329.
- Beyer, R., Etard, O., Cournède, P.-H. and Laurent-Gengoux, P. (2015). 'Modeling spatial competition for light in plant populations with the porous medium equation'. In: *Journal of Mathematical Biology* 70.3, pp. 533–547.
- Bezanson, J., Karpinski, S., Shah, V. B. and Edelman, A. (2012). 'Julia: a fast dynamic language for technical computing'.
- Bezanson, J., Edelman, A., Karpinski, S. and Shah, V. B. (2014). 'Julia: a fresh approach to numerical computing'.
- Blanco, G., Gerlagh, R., Suh, S., Barrett, J. and De Coninck, H. C. (2014). 'Drivers, trends and mitigation'. In: *Climate Change 2014: Mitigation of Climate Change. Contribution of Working Group III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press.
- Boone, E. L., Simmons, S. J., Bao, H. and Stapleton, A. E. (2008). 'Bayesian hierarchical regression models for detecting QTLs in plant experiments'. In: *Journal of Applied Statistics* 35.7, pp. 799–808.
- Bouriga, M. and Féron, O. (2013). 'Estimation of covariance matrices based on hierarchical inverse-wishart priors'. In: *Journal of Statistical Planning and Inference* 143.4, pp. 795–808.

- Brisson, N., Gary, C., Justes, E., Roche, R., Mary, B., Ripoche, D., Zimmer, D., Sierra, J., Bertuzzi, P., Burger, P., Bussi re, F., Cabidoche, Y., Cellier, P., Debaeke, P., Gaudill re, J., H nault, C., Maraux, F., Seguin, B. and Sinoquet, H. (2003). 'An overview of the crop model STICS'. In: *European Journal of Agronomy* 18.3. Modelling Cropping Systems: Science, Software and Applications, pp. 309–332.
- Brisson, N., Mary, B., Ripoche, D., Jeuffroy, M.-H., Ruget, F., Nicoulaud, B., Gate, P., Devienne-Barret, F., Antonioletti, R., Durr, C., Richard, G., Beau, N., Tayot, X., Plenet, D., Cellier, P., Machet, J.-M., Meynard, J.-M. and Del colle, R. (1998). 'STICS: a generic model for the simulation of crops and their water and nitrogen balances. I. Theory and parameterization applied to wheat and corn'. In: *Agronomie* 18.5–6, pp. 311–346.
- Brooks, S. P. and Gelman, A. (1998). 'General methods for monitoring convergence of iterative simulations'. In: *Journal of computational and graphical statistics* 7.4, pp. 434–455.
- Brouwer, J., Fussell, L. and Herrmann, L. (1993). 'Soil and crop growth micro-variability in the West African semi-arid tropics: a possible risk-reducing factor for subsistence farmers'. In: *Agriculture, Ecosystems and Environment* 45.3, pp. 229–238.
- Bustos-Korts, D., Malosetti, M., Chapman, S. and Van Eeuwijk, F. (2016). 'Modelling of genotype by environment interaction and prediction of complex traits across multiple environments as a synthesis of crop growth modelling, genetics and statistics'. In: *Crop Systems Biology*. Springer International Publishing, pp. 55–82.
- B dard, M. (2008). 'Optimal acceptance rates for Metropolis algorithms: moving beyond 0.234'. In: *Stochastic Processes and their Applications* 118.12, pp. 2198–2222.
- Campillo, F. and Rossi, V. (2006). 'Convolution particle filtering for parameter estimation in general state-space models'. In: *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 2159–2164.
- Canny, J. (1986). 'A computational approach to edge detection'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8.6, pp. 679–698.
- Capp , O., Moulines, E. and Ryden, T. (2005). *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer-Verlag New York.
- Cariboni, J., Gatelli, D., Liska, R. and Saltelli, A. (2007). 'The role of sensitivity analysis in ecological modelling'. In: *Ecological Modelling* 203, pp. 167–182.
- Carmier, P., Kyrgyzov, O. and Courn de, P.-H. (2017). 'A critical analysis of resampling strategies for the regularized particle filter'. submitted.
- Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P. and Riddell, A. (2017). 'Stan: a probabilistic programming language'. In: *Journal of Statistical Software, Articles* 76.1, pp. 1–32.
- Celeux, G., Hurn, M. and Robert, C. P. (2000). 'Computational and inferential difficulties with mixture posterior distributions'. In: *Journal of the American Statistical Association* 95.451, pp. 957–970.
- Chagneau, P., Mortier, F., Picard, N. and Bacro, J.-N. (2011). 'A hierarchical Bayesian model for spatial prediction of multivariate non-gaussian random fields'. In: *Biometrics* 67.1, pp. 97–105.
- Chapman, S. C. (2008). 'Use of crop models to understand genotype by environment interactions for drought in real-world and simulated plant breeding trials'. In: *Euphytica* 161.1, pp. 195–208.

- Chastaing, G., Gamboa, F. and Prieur, C. (2012). 'Generalized Hoeffding–Sobol decomposition for dependent variables – application to sensitivity analysis'. In: *Electronic Journal of Statistics* 6, pp. 2420–2448.
- Chen, Y. (2014). 'Inférence bayésienne dans les modèles de croissance de plantes pour la prévision et la caractérisation des incertitudes'. PhD thesis. École Centrale Paris.
- Chen, Y. and Cournède, P.-H. (2012). 'Assessment of parameter uncertainty in plant growth model identification'. In: *2012 IEEE 4th International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications*, pp. 85–92.
- Chen, Y. and Cournède, P.-H. (2014). 'Data assimilation to reduce uncertainty of crop model prediction with convolution particle filtering'. In: *Ecological Modelling* 290.Supplement C, pp. 165–177.
- Chen, Z. and Brown, E. N. (2013). 'State-space models for the analysis of neural spike train and behavioral data'. In: *Encyclopedia of Computational Neuroscience*. Ed. by Jaeger, D. and Jung, R. New York, NY: Springer New York, pp. 1–4.
- Christophe, A., Letort, V., Hummel, I., Cournède, P. H., De Reffye, P. and Lecœur, J. (2008). 'A model-based analysis of the dynamics of carbon balance at the whole-plant level in *Arabidopsis thaliana*'. In: *Functional Plant Biology* 35.11, pp. 1147–1162.
- Cournède, P.-H., Matthieu, A., Houllier, F., Barthélémy, D. and De Reffye, P. (2008). 'Computing competition for light in the Greenlab model of plant growth: a contribution to the study of the effects of density on resource acquisition and architectural development'. In: *Annals of Botany* 101.8, pp. 1207–1219.
- Cournède, P.-H., Letort, V., Mathieu, A., Kang, M. Z., Lemaire, S., Trevezas, S., Houllier, F. and De Reffye, P. (2011). 'Some parameter estimation issues in functional-structural plant modelling'. In: *Mathematical Modelling of Natural Phenomena* 6.2, pp. 133–159.
- Cournède, P.-H., Chen, Y., Wu, Q., Baey, C. and Bayol, B. (2013). 'Development and evaluation of plant growth models: methodology and implementation in the PYGMALION platform'. In: *Mathematical Modelling of Natural Phenomena* 8.4, pp. 112–130.
- Craufurd, P. Q., Vadez, V., Jagadish, S. K., Prasad, P. V. and Zaman-Allah, M. (2013). 'Crop science experiments designed to inform crop modeling'. In: *Agricultural and Forest Meteorology* 170, pp. 8–18.
- Davidian, M. and Giltinan, D. M. (1993). 'Some simple methods for estimating intraindividual variability in nonlinear mixed effects models'. In: *Biometrics* 49.1, pp. 59–73.
- Davidian, M. and Giltinan, D. M. (2003). 'Nonlinear models for repeated measurement data: an overview and update'. In: *Journal of Agricultural, Biological, and Environmental Statistics* 8.4, p. 387.
- De Reffye, P. and Houllier, F. (1997). 'Modelling plant growth and architecture: some recent advances and applications to agronomy and forestry'. In: *Current Science* 73, pp. 984–992.
- De Reffye, P., Fourcaud, T., Blaise, F., Barthélémy, D. and Houllier, F. (1997). 'A functional model of tree growth and tree architecture'. In: *Silva Fennica* 31(3), pp. 297–311.
- De Reffye, P., Elguero, E. and Costes, E. (1991). 'Growth units construction in trees: a stochastic approach'. In: *Acta Biotheoretica* 39.3, pp. 325–342.

- De Reffye, P. and Hu, B. (2003). 'Relevant qualitative and quantitative choices for building an efficient dynamic plant growth model: Greenlab case'. In: *International Symposium on Plant Growth Modeling, Simulation, Visualization and their Applications – PMA'03*. Ed. by Hu, B. and Jaeger, M. Plant Growth Modeling and Applications. Beijing, China: Springer and Tsinghua University Press, pp. 87–107.
- De Reffye, P., Jaeger, M. and Cournède, P.-H. (2009). 'Une histoire de la modélisation des plantes'. In: *Interstices*.
- De Reffye, P., Édelin, C., Françon, J., Jaeger, M. and Puech, C. (1988). 'Plant models faithful to botanical structure and development'. In: *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '88. New York, NY, USA: ACM, pp. 151–158.
- Del Moral, P. (2004). *Feynman–Kac Formulae: Genealogical and Interacting Particle Systems With Applications*. Springer-Verlag New York.
- Delyon, B., Lavielle, M. and Moulines, E. (1999). 'Convergence of a stochastic approximation version of the EM algorithm'. In: *The Annals of Statistics* 27.1, pp. 94–128.
- Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). 'Maximum likelihood from incomplete data via the EM algorithm'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 39.1, pp. 1–38.
- Des Marais, D. L., Razzaque, S., Hernandez, K. M., Garvin, D. F. and Juenger, T. E. (2016). 'Quantitative trait loci associated with natural diversity in water-use efficiency and response to soil drying in brachypodium distachyon'. In: *Plant Science* 251, pp. 2–11.
- Donnet, S., Foulley, J.-L. and Samson, A. (2010). 'Bayesian analysis of growth curves using mixed models defined by stochastic differential equations'. In: *Biometrics* 66.3, pp. 733–741.
- Douc, R. and Cappe, O. (2005). 'Comparison of resampling schemes for particle filtering'. In: *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005*. Pp. 64–69.
- Doucet, A., De Freitas, N. and Gordon, N. (2001). *Sequential Monte Carlo methods in practice*. New York: Springer-Verlag New York.
- Duane, S., Kennedy, A., Pendleton, B. J. and Roweth, D. (1987). 'Hybrid Monte Carlo'. In: *Physics Letters B* 195.2, pp. 216–222.
- Dupuy, L., Mackenzie, J., Rudge, T. and Haseloff, J. (2008). 'A system for modelling cell–cell interactions during plant morphogenesis'. In: *Annals of Botany* 101.8, pp. 1255–1265.
- Duval, M., Robert-Granié, C. and Fouley, J.-L. (2009). 'Estimation of heterogeneous variances in nonlinear mixed models via the SAEM-MCMC algorithm with applications to growth curves in poultry'. In: *Journal de la Société Française de Statistique* 150.2, pp. 65–83.
- Earl, D. and Deem, M. (2005). 'Parallel tempering: theory, applications, and new perspectives'. In: *Physical Chemistry Chemical Physics*.
- Édelin, C. (1977). 'Image de l'architecture des conifères'. PhD thesis. Université de Montpellier.
- El-Soda, M., Malosetti, M., Zwaan, B. J., Koornneef, M. and Aarts, M. G. (2014). 'Genotype x environment interaction QTL mapping in plants: lessons from Arabidopsis'. In: *Trends in Plant Science* 19.6, pp. 390–398.

- Elliott, R. J. and Hyndman, C. B. (2007). 'Parameter estimation in commodity markets: a filtering approach'. In: *Journal of Economic Dynamics and Control* 31.7, pp. 2350–2373.
- Epanechnikov, V. (1969). 'Nonparametric estimation of a multidimensional probability density'. In: *Teoriya Veroyatnostei i ee Primeneniya* 14.1, pp. 156–161.
- Evensen, G. (2009). *Data Assimilation: the Ensemble Kalman Filter*. Springer-Verlag Berlin Heidelberg.
- Evensen, G. (1994). 'Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics'. In: *Journal of Geophysical Research: Oceans* 99.C5, pp. 10143–10162.
- Fearnhead, P. (2011). 'MCMC for state-space models'. In: *Handbook of Markov chain Monte Carlo*. Ed. by Brooks, S. et al. London: Chapman and Hall, pp. 513–529.
- Fearnhead, P. and Meligkotsidou, L. (2016). 'Augmentation schemes for particle MCMC'. In: *Statistics and Computing* 26.6, pp. 1293–1306.
- Ford, E. D. and Kennedy, M. C. (2011). 'Assessment of uncertainty in functional-structural plant models'. In: *Annals of Botany* 108, pp. 1043–1053.
- Foulley, J.-L. and Quaas, R. L. (1995). 'Heterogeneous variances in Gaussian linear mixed models'. In: *Genetics Selection Evolution* 27.3, p. 211.
- Foulley, J.-L. (2002). 'Algorithme EM : théorie et application au modèle mixte'. In: *Journal de la société française de statistique* 143.3-4, pp. 57–109.
- Fournier, C. and Andrieu, B. (1999). 'ADEL-maize: an L-system based model for the integration of growth processes from the organ to the canopy. Application to regulation of morphogenesis by light availability'. In: *Agronomie* 19.3-4, pp. 313–327.
- Françon, J. (1991). 'Sur la modélisation informatique de l'architecture et du développement des végétaux'. In: *2e Colloque international sur l'Arbre*. Ed. by Édelin, C. Montpellier, France: Naturalia Monspelienis, pp. 231–247.
- Frühwirth-Schnatter, S. (2006). *Finite Mixture and Markov Switching Models*. Springer Series in Statistics. Springer-Verlag New York.
- Gabriel, E., Fagg, G. E., Bosilca, G., Angskun, T., Dongarra, J. J., Squyres, J. M., Sahay, V., Kambadur, P., Barrett, B., Lumsdaine, A., Castain, R. H., Daniel, D. J., Graham, R. L. and Woodall, T. S. (2004). 'Open MPI: goals, concept, and design of a next generation MPI implementation'. In: *Proceedings, 11th European PVM/MPI Users' Group Meeting*. Budapest, Hungary, pp. 97–104.
- Gelfand, A. E. and Smith, A. F. M. (1990). 'Sampling-based approaches to calculating marginal densities'. In: *Journal of the American Statistical Association* 85.410, pp. 398–409.
- Gelman, A., Roberts, G. O. and Gilks, W. R. (1996). 'Efficient Metropolis jumping rules'. In: *Bayesian statistics, 5 (Alicante, 1994)*. Oxford Science Publications. New York, USA: Oxford Univ. Press, pp. 599–607.
- Gelman, A. and Rubin, D. B. (1992). 'Inference from iterative simulation using multiple sequences'. In: *Statistical Science* 7.4, pp. 457–472.
- Geman, S. and Geman, D. (1984). 'Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images'. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 6.6, pp. 721–741.

- Gilks, W. R. (2005). 'Markov chain Monte Carlo'. In: *Encyclopedia of Biostatistics*. John Wiley and Sons, Ltd.
- Gilks, W. R., Best, N. G. and Tan, K. K. C. (1995). 'Adaptive rejection Metropolis sampling within Gibbs sampling'. In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 44.4, pp. 455–472.
- Gilks, W. R., Richardson, S. and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman and Hall.
- Girolami, M. and Betancourt, M. (2015). 'Hamiltonian Monte Carlo for hierarchical models'. In: *Current Trends in Bayesian Methodology with Applications*. Ed. by Upadhyay, S. K., Singh, U., Dey, D. K. and Loganathan, A. New York: Chapman and Hall/CRC.
- Godin, C., Costes, E. and Sinoquet, H. (1999). 'A method for describing plant architecture which integrates topology and geometry'. In: *Annals of Botany* 84.3, pp. 343–357.
- Golightly, A. and Wilkinson, D. (2008). 'Bayesian inference for nonlinear multivariate diffusion models observed with error'. In: *Computational Statistics and Data Analysis* 52.3, pp. 1674–1693.
- Golightly, A. and Wilkinson, D. J. (2011). 'Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo'. In: *Interface Focus* 1.6, pp. 807–820.
- Gordon, N., Salmond, D. and Smith, A. (1993). 'Novel approach to nonlinear/non-Gaussian Bayesian state estimation'. In: *IEEE Proceedings F (Radar and Signal Processing)* 140 (2), pp. 107–113.
- Granier, C. and Vile, D. (2014). 'Phenotyping and beyond: modelling the relationships between traits'. In: *Current Opinion in Plant Biology* 18, pp. 96–102.
- Grün, B. and Leisch, F. (2009). 'Dealing with label switching in mixture models under genuine multimodality'. In: *Journal of Multivariate Analysis* 100.5, pp. 851–861.
- Gustafsson, F. and Hendeby, G. (2012). 'Some relations between extended and unscented Kalman filters'. In: *IEEE Transactions on Signal Processing* 60.2, pp. 545–555.
- Haario, H., Saksman, E. and Tamminen, J. (1998). 'An adaptive Metropolis algorithm'. In: *Bernoulli* 7, pp. 223–242.
- Haario, H., Saksman, E. and Tamminen, J. (1999). 'Adaptive proposal distribution for random walk Metropolis algorithm'. In: *Computational Statistics* 14.3, pp. 375–395.
- Hallé, F. and Oldeman, R. (1970). *Essai sur l'architecture et la dynamique de croissance des arbres tropicaux*. Monographie de Botanique et de Biologie Végétale. Paris, France: Masson, p. 192.
- Hammer, G., Cooper, M., Tardieu, F., Welch, S., Walsh, B., Van Eeuwijk, F., Chapman, S. and Podlich, D. (2006). 'Models for navigating biological complexity in breeding improved crop plants'. In: *Trends in Plant Science* 11.12, pp. 587–593.
- Handschin, J. E. and Mayne, D. Q. (1969). 'Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering'. In: *International Journal of Control* 9.5, pp. 547–559.
- Hastings, W. K. (1970). 'Monte Carlo sampling methods using Markov chains and their applications'. In: *Biometrika* 57.1, pp. 97–109.
- Havre, Z. van, White, N., Rousseau, J. and Mengersen, K. (2015). 'Overfitting Bayesian mixture models with an unknown number of components'. In: *PLoS ONE* 10.7, pp. 1–27.

- Herdiyeni, Y., Lubis, D. I. and Douady, S. (2015). 'Leaf shape identification of medicinal leaves using curvilinear shape descriptor'. In: *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, pp. 218–223.
- Hill, J. (1975). 'Genotype-environment interaction – a challenge for plant breeding'. In: *The Journal of Agricultural Science* 85.3, pp. 477–493.
- Hirai, M. Y., Yano, M., Goodenowe, D. B., Kanaya, S., Kimura, T., Awazuhara, M., Arita, M., Fujiwara, T. and Saito, K. (2004). 'Integration of transcriptomics and metabolomics for understanding of global responses to nutritional stresses in *Arabidopsis thaliana*'. In: *Proceedings of the National Academy of Sciences* 101.27, pp. 10205–10210.
- Hodgman, C., French, A. and Westhead, D. (2009). *Bioinformatics*. Instant Notes. Taylor and Francis.
- Hoeffding, W. (1948). 'A class of statistics with asymptotically normal distribution'. In: *The Annals of Mathematical Statistics* 19.3, pp. 293–325.
- Hopkins, A., Vogel, K., Moore, K., Johnson, K. and Carlson, I. (1995). 'Genotype effects and genotype by environment interactions for traits of elite switchgrass populations'. In: *Crop Science* 35.1, pp. 125–132.
- Hosack, G. R., Peters, G. W. and Hayes, K. R. (2012). 'Estimating density dependence and latent population trajectories with unknown observation error'. In: *Methods in Ecology and Evolution* 3.6, pp. 1028–1038.
- Illian, J. B., Møller, J. and Waagepetersen, R. P. (2009). 'Hierarchical spatial point process analysis for a plant community with high biodiversity'. In: *Environmental and Ecological Statistics* 16.3, pp. 389–405.
- Isfan, D. (1993). 'Genotypic variability for physiological efficiency index of nitrogen in oats'. In: *Plant and Soil* 154.1, pp. 53–59.
- Jarquín, D., Pérez-Elizalde, S., Burgueño, J. and Crossa, J. (2016). 'A hierarchical Bayesian estimation model for multi-environment plant breeding trials in successive years'. In: *Crop Science*. Vol. 56, pp. 2260–2276.
- Jones, C. A., Kiniry, J. R. and Dyke, P. T. (1986). *CERES-Maize: a simulation model of maize growth and development*. Texas A&M University Press.
- Joseph, L. N., Elkin, C., Martin, T. G. and Possingham, H. P. (2009). 'Modeling abundance using n-mixture models: the importance of considering ecological mechanisms'. In: *Ecological Applications* 19.3, pp. 631–642.
- Julier, S. and Uhlmann, J. (1997). 'New extension of the Kalman filter to nonlinear systems'. In: *SPIE Proceedings* 3068, pp. 182–193.
- Jullien, A., Mathieu, A., Allirand, J.-M., Pinet, A., De Reffye, P., Cournède, P.-H. and Ney, B. (2010). 'Characterization of the interactions between architecture and source-sink relationships in winter oilseed rape (*Brassica napus*) using the GreenLab model'. In: *Annals of botany* 107.5, pp. 765–779.
- Kalman, R. E. (1960). 'A new approach to linear filtering and prediction problems'. In: *Transactions of the ASME – Journal of Basic Engineering* 82.Series D, pp. 35–45.
- Kitagawa, G. (1996). 'Monte Carlo filter and smoother for non-Gaussian nonlinear state space models'. In: *Journal of Computational and Graphical Statistics* 5.1, pp. 1–25.
- Kong, A., Liu, J. S. and Wong, W. H. (1994). 'Sequential imputations and Bayesian missing data problems'. In: *Journal of the American Statistical Association* 89.425, pp. 278–288.

- Koutroumpas, K., Ballarini, P., Votsi, I. and Cournède, P.-H. (2016). 'Bayesian parameter estimation for the Wnt pathway: an infinite mixture models approach'. In: *Bioinformatics* 32.17, pp. i781–i789.
- Kromdijk, J., Bertin, N., Heuvelink, E., Molenaar, J., Visser, P. H. B. de, Marcelis, L. F. M. and Struik, P. C. (2014). 'Crop management impacts the efficiency of quantitative trait loci (QTL) detection and use: case study of fruit load x QTL interactions'. In: *Journal of Experimental Botany* 65.1, pp. 11–22.
- Kuhn, E. and Lavielle, M. (2005). 'Maximum likelihood estimation in nonlinear mixed effects models'. In: *Computational Statistics and Data Analysis* 49.4, pp. 1020–1038.
- Kurth, W. (1994). 'Morphological models of plant growth: possibilities and ecological relevance'. In: *Ecological Modelling* 75. State-of-the-Art in Ecological Modelling proceedings of ISEM's 8th International Conference, pp. 299–308.
- Kusano, M., Fukushima, A., Arita, M., Jonsson, P., Moritz, T., Kobayashi, M., Hayashi, N., Tohge, T. and Saito, K. (2007). 'Unbiased characterization of genotype-dependent metabolic regulations by metabolomic approach in *Arabidopsis thaliana*'. In: *BMC Systems Biology* 1.1, p. 53.
- Launay, M. and Guerif, M. (2005). 'Assimilating remote sensing data into a crop model to improve predictive performance for spatial applications'. In: *Agriculture, Ecosystems & Environment* 111.1, pp. 321–339.
- Lavielle, M. (2014). *Mixed Effects Models for the Population Approach: Models, Tasks, Methods and Tools*. Chapman and Hall/CRC.
- Layer, T., Blaickner, M., Knausl, B., Georg, D., Neuwirth, J., Baum, R. P., Schuchardt, C., Wiessalla, S. and Matz, G. (2015). 'PET image segmentation using a Gaussian mixture model and Markov random fields'. In: *EJNMMI Physics* 2.1, p. 9.
- Le Roux, X., Lacoïnte, A., Escobar-Gutiérrez, A. and Le Dizès, S. (2001). 'Carbon-based models of individual tree growth: a critical appraisal'. In: *Ann. For. Sci.* 58.5, pp. 469–506.
- Leœur, J., Poiré-Lassus, R., Christophe, A., Pallas, B., Casadebaig, P., Debaeke, P., Vear, F. and Guillioni, L. (2011). 'Quantifying physiological determinants of genetic variation for yield potential in sunflower. SUNFLO: a model-based analysis'. In: *Functional Plant Biology* 38.3, pp. 246–259.
- LeGland, F., Musso, C. and Oudjane, N. (1998). 'An analysis of regularized interacting particle methods for nonlinear filtering'. In: *Proceedings of the 3rd IEEE European Workshop on Computer-Intensive Methods in Control and Signal Processing*, pp. 167–174.
- Lemaire, S., Maupas, F., Cournède, P.-H. and De Reffye, P. (2009). 'A morphogenetic crop model for sugarbeet (*Beta vulgaris* L.)'. In: *Crop Modeling and Decision Support*. Ed. by Cao, W., White, J. W. and Wang, E. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 116–129.
- Letort, V., Mahe, P., Cournède, P. H., De Reffye, P. and Courtois, B. (2008). 'Quantitative genetics and functional-structural plant growth models: simulation of quantitative trait loci detection for model parameters and application to potential yield optimization'. In: *Annals of Botany* 101.8, pp. 1243–1254.
- Lewin, A., Bochkina, N. and Richardson, S. (2007). 'Fully Bayesian mixture model for differential gene expression: simulations and model checks'. In: *Statistical Applications in Genetics and Molecular Biology* 6.1.

- Li, G., Rabitz, H., Yelvington, P. E., Oluwole, O. O., Bacon, F., Kolb, C. E. and Schoendorf, J. (2010). 'Global sensitivity analysis for systems with independent and/or correlated inputs'. In: *The Journal of Physical Chemistry A* 114.19, pp. 6022–6032.
- Lindenmayer, A. (1968). 'Mathematical models for cellular interactions in development I. Filaments with one-sided inputs'. In: *Journal of Theoretical Biology* 18.3, pp. 280–299.
- Lindley, D. and Smith, A. (1972). 'Bayes estimates for the linear model'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 34.1, pp. 1–41.
- Lindstrom, M. J. and Bates, D. M. (1990). 'Nonlinear mixed effects models for repeated measures data'. In: *Biometrics* 46.3, pp. 673–687.
- Liu, J., Wong, W. and Kong, A. (1994). 'Covariance structure and convergence rate of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes'. In: *Biometrika* 81, pp. 27–40.
- Liu, J. S. and Chen, R. (1995). 'Blind deconvolution via sequential imputations'. In: *Journal of the American Statistical Association* 90.430, pp. 567–576.
- Lunn, D., Jackson, C., Best, N., Thomas, A. and Spiegelhalter, D. (2012). *The BUGS Book: A Practical Introduction to Bayesian Analysis*. CRC Press / Chapman and Hall.
- Lunn, D., Spiegelhalter, D., Thomas, A. and Best, N. (2009). 'The BUGS project: evolution, critique and future directions'. In: *Statistics in Medicine* 28.25, pp. 3049–3067.
- Lunn, D. J., Thomas, A., Best, N. and Spiegelhalter, D. (2000). 'WinBUGS – a Bayesian modelling framework: concepts, structure, and extensibility'. In: *Statistics and Computing* 10.4, pp. 325–337.
- Mailhol, J., Revol, P. and Ruelle, P. (1996). 'Pilote : un modèle opérationnel pour déceler l'apparition de stress hydrique'. In: *ICID 16th international congress on irrigation and drainage: workshop on crop-water-environment models*. Cairo, Egypt.
- Maiti, R., Maiti, L. E., Maiti, S., Maiti, A. M., Maiti, M. and Maiti, H. (1996). 'Genotypic variability in maize cultivars (*Zea mays* L.) for resistance to drought and salinity at the seedling stage'. In: *Journal of Plant Physiology* 148.6, pp. 741–744.
- Makowski, D., Jeuffroy, M.-H. and Guérif, M. (2004). 'Bayesian methods for updating crop model predictions, applications for predicting biomass and grain protein content'. In: *Bayesian Statistics and Quality Modelling in the Agro-Food Production Chain*. Ed. by al., V. B. et. Kluwer Academic Publishers, pp. 57–68.
- Malefaki, S., Trevezas, S., Viaud, G. and Cournède, P.-H. (2014). 'Bayesian estimation for the Greenlab model adapted to the sugar beet plant'. In: *Proceedings of the 27th Panhellenic Statistics Conference*.
- Mamon, R. S. and Elliott, R. J., eds. (2014). *Hidden Markov Models in Finance: Further Developments and Applications, Volume II*. Vol. 209. International Series in Operations Research and Management Science. Springer US.
- Marcelis, L., Heuvelink, E. and Goudriaan, J. (1998). 'Modelling of biomass production and yield of horticultural crops: a review'. In: *Scientia Horticulturae* 74, pp. 83–111.
- Marin, J.-M., Mengersen, K. and Robert, C. P. (2005). 'Bayesian modelling and inference on mixtures of distributions'. In: *Bayesian Thinking*. Ed. by Dey, D. and Rao, C. Vol. 25. Handbook of Statistics Supplement C. Elsevier, pp. 459–507.

- Martre, P., Bertin, N., Salon, C. and Génard, M. (2011). 'Modelling the size and composition of fruit, grain and seed by process-based simulation models'. In: *New Phytologist* 191.3, pp. 601–618.
- McCulloch, C. E. (1994). 'Maximum likelihood variance components estimation for binary data'. In: *Journal of the American Statistical Association* 89.425, pp. 330–335.
- McDonald, P. G., Fonseca, C. R., Overton, J. M. and Westoby, M. (2003). 'Leaf-size divergence along rainfall and soil-nutrient gradients: is the method of size reduction common among clades?' In: *Functional Ecology* 17.1, pp. 50–57.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). 'Equation of state calculations by fast computing machines'. In: *The Journal of Chemical Physics* 21.6, pp. 1087–1092.
- Meza, C., Jaffrézic, F. and Foulley, J.-L. (2007). 'REML estimation of variance parameters in nonlinear mixed effects models using the SAEM algorithm'. In: *Biometrical Journal* 49.6, pp. 876–888.
- Migault, V., Pallas, B. and Costes, E. (2017). 'Combining genome-wide information with a functional structural plant model to simulate 1-year-old apple tree architecture'. In: *Frontiers in plant science* 7, p. 2065.
- Mingas, G., Bottolo, L. and Bouganis, C.-S. (2017). 'Particle MCMC algorithms and architectures for accelerating inference in state-space models'. In: *International Journal of Approximate Reasoning* 83. Supplement C, pp. 413–433.
- Monteith, J. L. (1965). 'Evaporation and environment'. In: *Symposia of the Society for Experimental Biology*. Vol. 19, pp. 205–224.
- Mortier, F., Flores, O. and Gourlet-Fleury, S. (2007). 'Spatial Bayesian models of tree density with zero inflation and autocorrelation'. In: *Journal de la Société Française de Statistique et Revue de Statistique Appliquée* 148.1, pp. 39–51.
- Murray, L. (2015). 'Bayesian state-space modelling on high-performance hardware using LibBi'. In: *Journal of Statistical Software* 67.1, pp. 1–36.
- Musso, C. and Oudjane, N. (1998). 'Regularisation schemes for branching particle systems as a numerical solving method of the nonlinear filtering problem'. In: *Proceedings of the Irish Signals Systems Conference*. Dublin, Ireland.
- Musso, C., Oudjane, N. and Le Gland, F. (2001). 'Improving regularized particle filters'. In: *Sequential Monte Carlo methods in practice*. Ed. by Doucet, A., De Freitas, N. and Gordon, N. Statistics for Engineering and Information Science. Springer, pp. 247–271.
- Newman, K. B., Fernández, C., Thomas, L. and Buckland, S. T. (2009). 'Monte Carlo inference for state-space models of wild animal populations'. In: *Biometrics* 65.2, pp. 572–583.
- Nobile, A. and Fearnside, A. T. (2007). 'Bayesian finite mixtures with an unknown number of components: the allocation sampler'. In: *Statistics and Computing* 17.2, pp. 147–162.
- Oudjane, N. and Musso, C. (1999). 'Multiple model particle filter'. In: *17ème Colloque sur le traitement du signal et des images*, pp. 681–683.
- Paninski, L., Ahmadian, Y., Ferreira, D. G., Koyama, S., Rahnama Rad, K., Vidne, M., Vogelstein, J. and Wu, W. (2010). 'A new look at state-space models for neural data'. In: *Journal of Computational Neuroscience* 29.1, pp. 107–126.

- Pape, J.-M. and Klukas, C. (2015). 'Utilizing machine learning approaches to improve the prediction of leaf counts and individual leaf segmentation of rosette plant images'. In: *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)*. Ed. by Tsafaris, S. A., Scharr, H. and Pridmore, T. BMVA Press, pp. 3.1–3.12.
- Parzen, E. (1962). 'On estimation of a probability density function and mode'. In: *The Annals of Mathematical Statistics* 33.3, pp. 1065–1076.
- Patrick, L. D., Ogle, K. and Tissue, D. T. (2009). 'A hierarchical Bayesian approach for estimation of photosynthetic parameters of C3 plants'. In: *Plant, Cell and Environment* 32.12, pp. 1695–1709.
- Perttunen, J., Siev Änen, R., Nikinmaa, E., Salminen, H., Saarenmaa, H. and Väkevä, J. (1996). 'LIGNUM: a tree model based on simple structural units'. In: *Annals of Botany* 77.1, pp. 87–98.
- Pieruschka, R. and Poorter, H. (2012). 'Phenotyping plants: genes, phenes and machines'. In: *Functional Plant Biology* 39.11, pp. 813–820.
- Pinheiro, J. C. and Bates, D. M. (1995). 'Approximations to the log-likelihood function in the nonlinear mixed-effects model'. In: *Journal of Computational and Graphical Statistics* 4.1, pp. 12–35.
- Plummer, M. (2003). 'JAGS: a program for analysis of Bayesian graphical models using Gibbs sampling'. In: *Proceedings of the 3rd international workshop on distributed statistical computing*. Vol. 124. Vienna, p. 125.
- Prusinkiewicz, P. and Lindenmayer, A. (1990). *The algorithmic beauty of plants*. New York, USA: Springer Verlag.
- Prusinkiewicz, P., Lindenmayer, A. and Hanan, J. (1988). 'Development models of herbaceous plants for computer imagery purposes'. In: *SIGGRAPH Computer Graphics* 22.4, pp. 141–150.
- Qi, R., Ma, Y., Hu, B., De Reffye, P. and Cournède, P. H. (2010). 'Optimization of source-sink dynamics in plant growth for ideotype breeding: a case study on maize'. In: *Computers and Electronics in Agriculture* 71.1, pp. 96–105.
- Quach, M., Brunel, N. and D'Alché-Buc, F. (2007). 'Estimating parameters and hidden variables in nonlinear state-space models based on ODEs for biological networks inference'. In: *Bioinformatics* 23.23, pp. 3209–3216.
- Quilot, B., Kervella, J., Génard, M. and Lescourret, F. (2005). 'Analysing the genetic control of peach fruit quality through an ecophysiological model combined with a QTL approach'. In: *Journal of Experimental Botany* 56.422, pp. 3083–3092.
- Quilot-Turion, B., Ould-Sidi, M.-M., Kadrani, A., Hilgert, N., Génard, M. and Lescourret, F. (2012). 'Optimization of parameters of the virtual fruit model to design peach genotype for sustainable production systems'. In: *European Journal of Agronomy* 42, pp. 34–48.
- Rabiner, L. R. (1989). 'A tutorial on hidden Markov models and selected applications in speech recognition'. In: *Proceedings of the IEEE* 77.2, pp. 257–286.
- Racine-Poon, A. (1992). 'Practical Markov chain Monte Carlo: comment'. In: *Statistical Science* 7.4, pp. 492–493.
- Raiffa, H. and Schlaifer, R. (1961). *Applied Statistical Decision Theory*. Division of Research, Graduate School of Business Administration, Harvard University.

- Rainforth, T., Naesseth, C. A., Lindsten, F., Paige, B., Van De Meent, J.-W., Doucet, A. and Wood, F. (2016). 'Interacting particle Markov chain Monte Carlo'. In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA, pp. 2616–2625.
- Rasch, A. and Bücker, H. M. (2010). 'EFCOSS: an interactive environment facilitating optimal experimental design'. In: *ACM Transactions on Mathematical Software* 37.2, 13:1–13:37.
- Reymond, M., Muller, B., Leonardi, A., Charcosset, A. and Tardieu, F. (2003). 'Combining quantitative trait loci analysis and an ecophysiological model to analyze the genetic variability of the responses of maize leaf growth to temperature and water deficit'. In: *Plant Physiology* 131.2, pp. 664–675.
- Richardson, S. and Green, P. J. (1997). 'On Bayesian analysis of mixtures with an unknown number of components'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 59.4, pp. 731–792.
- Ripley, B. D. (1987). *Stochastic Simulation*. New York, USA: John Wiley and Sons, Inc.
- Rivière, P., Dawson, J., Goldringer, I. and David, O. (2015). 'Hierarchical Bayesian modeling for flexible experiments in decentralized participatory plant breeding'. In: *Crop Science*. Vol. 55, p. 1053.
- Robbins, H. and Monro, S. (1951). 'A stochastic approximation method'. In: *The Annals of Mathematical Statistics* 22.3, pp. 400–407.
- Robert, C. (2007). *The Bayesian choice. From Decision-Theoretic Foundations to Computational Implementation*. Springer Texts in Statistics. Springer-Verlag New York.
- Robert, C. and Casella, G. (1999). *Monte Carlo statistical methods*. Springer Texts in Statistics. Springer Verlag New-York.
- Roberts, C. (1996). 'Markov chain concepts related to sampling algorithms'. In: *Markov chain Monte Carlo in practice*, pp. 45–57.
- Roberts, G. O. and Sahu, S. K. (1997). 'Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 59.2, pp. 291–317.
- Roberts, G. O. and Rosenthal, J. S. (1998). 'Optimal scaling of discrete approximations to Langevin diffusions'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60.1, pp. 255–268.
- Roberts, G. and Rosenthal, J. (2001). 'Optimal scaling for various Metropolis–Hastings algorithms'. In: *Statistical Science* 16.4, pp. 351–367.
- Rossi, V. and Vila, J.-P. (2006). 'Nonlinear filtering in discrete time: a particle convolution approach'. In: *Annales de l'Institut de Statistique de l'Université de Paris* 50.3, pp. 71–102.
- Sahu, S. K. and Cheng, R. C. H. (2003). 'A fast distance-based approach for determining the number of components in mixtures'. In: *The Canadian Journal of Statistics / La Revue Canadienne de Statistique* 31.1, pp. 3–22.
- Sainte-Marie, J., Viaud, G. and Cournède, P.-H. (2017). 'Indices de Sobol généralisés aux variables dépendantes : tests de performance de l'algorithme HOGS couplé à plusieurs estimateurs paramétriques'. In: *Journal de la Société Française de Statistique* 158.1.

- Saltelli, A., Andres, T. H. and Homma, T. (1993). 'Sensitivity analysis of model output. An investigation of new techniques'. In: *Computational Statistics and Data Analysis* 15, pp. 211–238.
- Saltelli, A., Chan, K. and Scott, E. M. (2000). *Sensitivity Analysis*. John Wiley and Sons, Ltd.
- Saltelli, A., Tarantola, S., Campolongo, F. and Ratto, M. (2004). *Sensitivity Analysis in Practice*. John Wiley and Sons, Ltd.
- Scharr, H., Minervini, M., French, A. P., Klukas, C., Kramer, D. M., Liu, X., Luengo, I., Pape, J.-M., Polder, G., Vukadinovic, D., Yin, X. and Tsaftaris, S. A. (2016). 'Leaf segmentation in plant phenotyping: a collation study'. In: *Machine Vision and Applications* 27.4, pp. 585–606.
- Schneider, M. K., Law, R. and Illian, J. B. (2006). 'Quantification of neighbourhood-dependent plant growth by Bayesian hierarchical modelling'. In: *Journal of Ecology* 94.2, pp. 310–321.
- Scoffoni, C., Rawls, M., McKown, A., Cochard, H. and Sack, L. (2011). 'Decline of leaf hydraulic conductance with dehydration: relationship to leaf size and venation architecture'. In: *Plant Physiology* 156, pp. 832–843.
- Searle, S. R., Casella, G. and McCulloch, C. E. (1992). *Variance Components*. John Wiley and Sons.
- Sherlock, C., Thiery, A. H. and Lee, A. (2017). 'Pseudo-marginal Metropolis–Hastings sampling using averages of unbiased estimators'. In: *Biometrika* 104.3, pp. 727–734.
- Simek, K. and Barnard, K. (2015). 'Gaussian process shape models for Bayesian segmentation of plant leaves'. In: *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)*. Ed. by Tsaftaris, S. A., Scharr, H. and Pridmore, T. BMVA Press, pp. 4.1–4.11.
- Smith, A. R. (1984). 'Plants, fractals, and formal languages'. In: *SIGGRAPH on Computer Graphics* 18.3, pp. 1–10.
- Smith, R. S., Guyomarç'h, S., Mandel, T., Reinhardt, D., Kuhlemeier, C. and Prusinkiewicz, P. (2006). 'A plausible model of phyllotaxis'. In: *Proceedings of the National Academy of Sciences* 103.5, pp. 1301–1306.
- Sorenson, H. (1985). *Kalman filtering: theory and application*. Ed. by Sorenson, H. IEEE Press.
- Stephens, M. (2000). 'Dealing with label switching in mixture models'. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 62.4, pp. 795–809.
- Tardieu, F. (2003). 'Virtual plants: modelling as a tool for the genomics of tolerance to water deficit'. In: *Trends in Plant Science* 8.1, pp. 9–14.
- Tatarinova, T. and Schumitzky, A. (2015). *Nonlinear Mixture Models. A Bayesian approach*. London: Imperial College Press.
- Taylor, W. E. (1977). 'Small sample properties of a class of two stage Aitken estimators'. In: *Econometrica* 45.2, pp. 497–508.
- Tierney, L. (1994). 'Markov chains for exploring posterior distributions'. In: *The Annals of Statistics* 22.4, pp. 1701–1728.
- Tisné, S., Serrand, Y., Bach, L., Gilbault, E., Ben Ameer, R., Balasse, H., Voisin, R., Bouchez, D., Durand-Tardif, M., Guerche, P., Chareyron, G., Da Rugna, J., Camilleri, C. and Loudet, O. (2013). 'Phenoscope: an automated large-scale phenotyping platform offering high spatial homogeneity'. In: *The Plant Journal* 74.3, pp. 534–544.

- Tisné, S., Schmalenbach, I., Reymond, M., Dauzat, M., Pervent, M., Vile, D. and Granier, C. (2010). 'Keep on growing under drought: genetic and developmental bases of the response of rosette area using a recombinant inbred line population'. In: *Plant, Cell and Environment* 33.11, pp. 1875–1887.
- Trevezas, S., Malefaki, S. and Cournède, P.-H. (2014). 'Parameter estimation via stochastic variants of the ECM algorithm with applications to plant growth modeling'. In: *Computational Statistics and Data Analysis* 78.Supplement C, pp. 82–99.
- Trevezas, S. and Cournède, P.-H. (2013). 'A sequential Monte Carlo approach for MLE in a plant growth model'. In: *Journal of Agricultural, Biological, and Environmental Statistics* 18.2, pp. 250–270.
- Tsukaya, H. (2005). 'Leaf shape: genetic controls and environmental factors'. In: *The International Journal of Developmental Biology* 49, pp. 547–555.
- Uhlmann, J. (1995). 'Dynamic map building and localization for autonomous vehicles'. PhD thesis. University of Oxford.
- Van Waveren, R., Groot, S., Scholten, H., Van Geer, F., Wosten, H., Koeze, R. and Noort, J. (1999). *Good Modelling Practice Handbook*. Tech. rep. 99-05. STOWA, Utrecht, The Netherlands.
- Viaud, G., Chen, Y., Bayol, B. and Cournède, P.-H. (2015). 'A comparison of a generic MCMC-based algorithm for Bayesian estimation in C++, R and Julia. Application to plant growth modeling'. In: *Annual Simulation Symposium*. Alexandria, United States.
- Vonesh, E. F. (1992). 'Non-linear models for the analysis of longitudinal data'. In: *Statistics in Medicine* 11.14-15, pp. 1929–1954.
- Vos, J., Evers, J. B., Buck-Sorlin, G., Andrieu, B., Chelle, M. and De Visser, P. H. (2009). 'Functional-structural plant modelling: a new versatile tool in crop science'. In: *Journal of Experimental Botany* 61.8, pp. 2101–2115.
- Wakefield, J., Gelfand, A. and Smith, A. (1991). 'Efficient generation of random variates via the ratio-of-uniforms method'. In: *Statistics and Computing* 1.2, pp. 129–133.
- Wakefield, J. C., Smith, A. F. M., Racine-Poon, A. and Gelfand, A. E. (1994). 'Bayesian analysis of linear and non-linear population models by using the Gibbs sampler'. In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 43.1, pp. 201–221.
- Wallach, D. (2006). 'Evaluating crop models'. In: *Working with Dynamic Crop Models: Evaluation, Analysis, Parameterization, and Applications*. Elsevier, Amsterdam, pp. 11–54.
- Wand, M. and Jones, M. (1994). *Kernel smoothing*. Monographs on Statistics and Applied Probability. Chapman and Hall/CRC.
- Wasson, A. P., Chiu, G. S., Zwart, A. B. and Binns, T. R. (2017). 'Differentiating wheat genotypes by Bayesian hierarchical nonlinear mixed modeling of wheat root density'. In: *Frontiers in Plant Science* 8, p. 282.
- White, N., Johnson, H., Silburn, P., Mellick, G., Dissanayaka, N. and Mengersen, K. (2012). 'Probabilistic subgroup identification using Bayesian finite mixture modelling: a case study in Parkinson's disease phenotype identification.' In: *Statistical Methods in Medical Research* 21 6, pp. 563–83.
- Wilhelm, W. and McMaster, G. S. (1995). 'Importance of the phyllochron in studying development and growth in grasses'. In: *Crop Science* 35.1, pp. 1–3.

- Wishart, J. (1928). 'Sampling errors in the theory of two factors'. In: *British Journal of Psychology. General Section* 19.2, pp. 180–187.
- Wu, L., Le Dimet, F.-X., De Reffye, P., Hu, B. and Cournède, P.-H. (2012a). 'An optimal control methodology for plant growth. Case study of a water supply problem of sunflower'. In: *Mathematics and Computers in Simulation* 82, pp. 909–923.
- Wu, Q.-L., Cournède, P.-H. and Mathieu, A. (2012b). 'An efficient computational method for global sensitivity analysis and its application to tree growth modelling'. In: *Reliability Engineering and System Safety* 107, pp. 35–43.
- Xu, L., Henke, M., Zhu, J., Kurth, W. and Buck-Sorlin, G. (2011). 'A functional–structural model of rice linking quantitative genetic information with morphological development and physiological processes'. In: *Annals of Botany* 107.5, pp. 817–828.
- Yan, H.-P., Kang, M. Z., De Reffye, P. and Dingkuhn, M. (2004). 'A dynamic, architectural plant model simulating resource-dependent growth'. In: *Annals of Botany* 93.5, pp. 591–602.
- Yao, W. (2012). 'Model based labeling for mixture models'. In: *Statistics and Computing* 22.2, pp. 337–347.
- Yin, X. and Struik, P. C. (2010). 'Modelling the crop: from system dynamics to systems biology'. In: *Journal of Experimental Botany* 61.8, pp. 2171–2183.
- Zeger, S. L. and Karim, M. R. (1991). 'Generalized linear models with random effects; a Gibbs sampling approach'. In: *Journal of the American Statistical Association* 86.413, pp. 79–86.

Titre : Méthodes statistiques pour la différenciation génotypique des plantes à l'aide des modèles de croissance

Mots-clés : modèles de croissance de plantes, inférence statistique, Julia, analyse d'images, différenciation génotypique, modèles hiérarchiques bayésiens

Résumé : Les modèles de croissance de plantes peuvent être utilisés afin de prédire des quantités d'intérêt ou évaluer la variabilité génotypique au sein d'une population de plantes ; ce double usage est mis en évidence au sein de ce travail.

Trois modèles de plantes sont ainsi considérés (LNAS pour la betterave et le blé, GreenLab pour *Arabidopsis thaliana*) au sein du cadre mathématique des modèles à espace d'états généraux.

Une nouvelle plate-forme de calcul générique pour la modélisation et l'inférence statistique (ADJUSTIN') a été développée en Julia, permettant la simulation des modèles de croissance de plantes considérés ainsi que l'utilisation de techniques d'estimation de pointe telles que les méthodes de Monte Carlo par chaînes de Markov ou de Monte Carlo séquentielles.

L'inférence statistique au sein des modèles de croissance de plantes étant de première importance pour des applications concrètes telles que la prédiction de rendement, les méthodes d'estimation de paramètres et d'états au sein de modèles à espaces d'états et dans un cadre bayésien furent

tout d'abord étudiées, et plusieurs cas d'étude pour les plantes considérées sont analysés pour le cas d'une plante individuelle.

La caractérisation de la variabilité au sein d'une population de plantes est envisagée à travers les distributions des paramètres de population au sein de modèles hiérarchiques bayésiens. Cette approche requérant l'acquisition de nombreuses données pour chaque individu, un algorithme de segmentation-suivi pour l'analyse d'images d'*Arabidopsis thaliana*, obtenues grâce au Phénoscope, une plate-forme de phénotypage à haut rendement de l'INRA Versailles, est proposé.

Finalement, l'intérêt de l'utilisation des modèles hiérarchiques bayésiens pour la mise en évidence de la variabilité au sein d'une population de plantes est discutée. D'abord par l'étude de différents scénarios sur des données simulées, et enfin en utilisant les données expérimentales obtenues à partir de l'analyse d'images pour une population d'*Arabidopsis thaliana* comprenant 48 individus.

Title: Statistical methods for the genotypic differentiation of plants using growth models

Keywords: plant growth models, statistical inference, Julia, image analysis, genotypic differentiation, Bayesian hierarchical models

Abstract: Plant growth models can be used in order to predict quantities of interest or assess the genotypic variability of a population of plants; this dual use is emphasized throughout this work.

Three plant growth models are therefore considered (LNAS for sugar beet and wheat, GreenLab for *Arabidopsis thaliana*) within the mathematical framework of general state space models.

A new generic computing platform for modelling and statistical inference (ADJUSTIN') has been developed in Julia, allowing to simulate the plant growth models considered as well as the use of state-of-the-art estimation techniques such as Markov chain Monte Carlo and sequential Monte Carlo methods.

Statistical inference within plant growth models is of primary importance for concrete applications such as yield prediction, parameter and state estimation methods within general state-space models in a Bayesian framework were

first studied and several case studies for the plants considered are then investigated in the case of an individual plant.

The characterization of the variability of a population of plants is envisioned through the distributions of parameters using Bayesian hierarchical models. This approach requiring the acquisition of numerous data for each individual, a segmentation-tracking algorithm for the analysis of images of *Arabidopsis thaliana*, obtained thanks to the Phénoscope, a high-throughput phenotyping platform of INRA Versailles, is proposed.

Finally, the interest of using Bayesian hierarchical models to evidence the variability of a population of plants is discussed. First through the study of different scenarios on simulated data, and then by using the experimental data acquired via image analysis for the population of *Arabidopsis thaliana* comprising 48 individuals.