Méthodes Variationelles pour des Modèles Fonction-Structure de Plantes : Identification de Paramètre, Contrôle et Assimilation de Données

Lin Wu

Thèse

présentée par

Lin WU

Pour obtenir le titre de
Docteur de l'Université Joseph Fourier - Grenoble I
(arrêtés ministériels du 5 juillet 1984 et du 30 mars 1992)

Spécialité : **Mathématiques Appliquées**

Variational Methods Applied to Plant Functional-Structural Dynamics: Parameter Identification, Control and Data Assimilation.

Date de soutenance : 25 avril 2005

Composition du jury :

M. Philippe De Reffye	Président
M. Jean-Pierre Quadrat	Rapporteur
M. Frédéric Baret	Rapporteur
M. François-Xavier Le Dimet	Directeur
M. Bao-Gang Hu	Co-directeur
M. Hervé Rey	Examinateur

Thèse préparée au sein du Laboratoire de Modélisation et Calcul (IMAG projet IDOPT) et de l'INRIA rhônes-alpes.

Acknowledgments

This work cannot be done without the supports from my dear professors and colleagues. I'd like to express my sincere gratitude here.

To Prof. Fraçois-Xavier Le Dimet, my advisor, for his constant support during this long journey, for his careful guidance, for his decisive insights and encouragements. As he believed, we should start this thesis aiming at the opening of the door of applied mathematics for individual plant, if possible, agronomy.

To Prof. Bao.-Gang Hu, my advisor, for his endless support, for his transmission of passions and rigor in science, and for his generosity and kindness.

To Prof. Philippe De Reffye, for his acceptance of being president of the jury, for his tutorials on the art of plant growth, for his patience, for his vigorous promotions for this work, and for the good time which he has shared with his Chinese students.

To Prof. Jean-Pierre Quadrat and Prof. Frédéric Baret, for their acceptances of being my rapporteurs, and for their crucial suggestions on the refinement of the thesis.

To Mr. Hervé Rey, for his acceptance of being examiner in the jury, for his interests in this work, and especially for his comments on plant modelling.

I wish to thank Dr. Paul-Henry Cournède for his suggestions on the organization of the thesis, and to Mr. Jean-François Barczi for his helpful discussions.

My gratitude is extended to my colleagues for their helps and discussions. To Dr. Zhigang Zhan for his encouragements especially at difficult moments, for his discussions on nearly all aspects of this thesis. To Dr. Cyril Mazauric for his help on automatic differentiation and many others. To Dr. Amélie Mathieu for her leading opinions on the formulation of *GreenLab*. To Dr. Mengzhen Kang for her help on *GreenLab* programs. For the friendly atmosphere and the discussions in LMC and LIAMA, thanks my dear workmates, Hongping, Junqing, Claire, William, Sophie, Arthur, Veronika, Delia, Xing, Yujiu, Teng, Huaifeng, ... I cannot finish, but I have to.

Finally, a special thank to the financial support from Ambassade de France à Beijing and Institute of Automation, Chinese Academy of Sciences. Thanks for the experimental support from the work groups in Chinese Agriculture University and Chinese Academy of Agricultural Sciences.

This thesis is devoted to my parents. Xie Xie Ba Ma.

Contents

Introduction en français

Les *méthodes variationelles*, ou le calcul des variations, sont utilisées pour résoudre les problèmes mathématiques d'optimisation. C'est á dire, la minimisation de fonctionelles. De telles *fonctionnelles* sont définies dans les espace fonctionnels. Les variables indépendantes d'un fonctionnelle sont également des fonctions. Par exemple, une intégrale des termes qui contiennent une fonction inconnue et ses dérivées, est une fonctionnelle typique.

La théorie des méthodes variationnelles ainsi que leurs applications sont étudiées depuis l'époque de Newton. Initialisé par les frères Bernoulli, le problème de *brachystochrone* [5] lança la recherche dans ce domaine, suivi par une longue lignée des grands mathématiciens (Leibniz, Newton, Huygens et l'Hospital, même Gallilée). Euler et Lagrange ont étudié les conditions nécessaires pour minimiser une fonctionnelle en ajoutant des équations supplémentaires, connues sous le nom d'équations d'Euler-Lagrange. Pour plus de détails on se référera à [47]. Enrichi par les contributions de Weierstrass, de Hamilton, et de Morse, dans les années 60 les méthodes variationnelles sont devenues populaires, représenté par le principe de maximum de Pontryagin [89] et de programmation dynamique de Bellman [4].

Le but des théories variationnelles est de formuler les lois de la nature générales qui suggèrent le principe de l'économie des moyens. La nature semble aller au plus simple, comme Newton l'a écrit dans *Principia*:"Nature does nothing in vain, and more is in vain when less will serve;...", et comme Euler dit dans son *Methodus inveniendi*:"Every effect in nature follows a maximum or minimum rule." En raison de ce principe *variationel*, les méthodes variationnelles reflètent l'harmonie de notre nature, ce qui les rends facilement adaptables aux diverses applications de la physique, de la biologie, de l' économie et de l'ingénierie.

Vers la fin des années 60, J.-L. Lions a étudié les systémes régis par les équations aux dérivés partielles et leur contrôle [75], [76]. Il a présenté ces méthodes variationnelles dans le domaine des mathématiques appliquées et a constamment soutenu son développement. Dans les années 90, il a proposé quelques remarques sur l'environnement et la modélisation mathématiques [77], [78]. Ses remarques ont fondée le cadre mathématiques de la recherche en science environnementale. Il a proposé les étapes suivantes d'une méthodologie générale pour une meilleure compréhension de l'environnement [78]

Etape 1 : Modélisation mathématique pour la réalité.
Etape 2 : Analyse mathématique d'un modèle et de sa simulation numérique.
Etape 3 : Validation et correction avec des données d'observations disponibles.
Etape 4 : Contrôle du système.
Lions a souligné les composantes inconnues liées aux conditions initiales, aux conditions

limites, et aux modèles lui-même. Les composantes inconnues peuvent être estimées en résolvant un problème inverse par la minimisation d'une norme calculant l'écart entre l'observation et le rendement de modèle, ce qui s'appelle également le calibrage. Une question, qui est conceptuellement semblable au problème inverse, est l'assimilation de données qui vise à estimer le processus d'évolution avec toutes sortes d'informations disponibles. En bref, les étapes de Lions peuvent être récapitulées comme la modélisation, le calibrage, la validation et le contrôle.

Ces dernières années, les méthodes variationnelles ont été de plus en plus expérimentées en science environnementale, c'est à dire, météorologie, océanographie, et hydrologie. F.-X. Le Dimet et I.M. Navon fournissent une revue complète sur les méthodes variationnelles et d'optimisation pour la météorologie [70]. Nous montrons les ingrédients des méthodes variationnelles dans le système suivant:

$$\begin{cases} \dfrac{\partial \mathbf{X}}{\partial t} & = F(\mathbf{X}, \mathbf{K}, \mathbf{U}) + B(\mathbf{V}), & \mathbf{X} \in \mathcal{X} \\ \mathbf{X}(t_0) & = \mathbf{\Theta} & \\ C(\mathbf{X}) & = f, & \mathbf{X} \in \partial\mathcal{X} \\ \mathbf{Y} & = M(\mathbf{X}) & \mathbf{Y} \in \mathcal{M} \\ \text{opt } J & = J(\mathbf{Z}), & \mathbf{Z} \in \mathcal{Z} \end{cases} \tag{1}$$

où (i) $\mathbf{X}$ est variable d'état, dans la météorologie ce peut être le vecteur de la vitesse, de la pression, de la température ou de l'humidité liquides pour l'atmosphère ; (ii) F est opérateur non-linéaire différentiel pour l'équation ordinaire ou partielle ; (iii) $\mathbf{U}$ est la variable de commande, ou la variable d'entrée ; (iv) $\mathbf{K}$ est le vecteur des variables dans l'espace $\mathcal{K}$; (v) $\mathbf{V}$ est l'erreur de modèle, à laquelle on impose un opérateur B pour le raccordement de l'erreur avec la dynamique du modèle ; (vi) $\mathbf{\Theta}$ est la condition initiale ; (vii) C est l'opérateur pour inclure les conditions limites; (viii) $\mathcal{X}$ est l'espace d'état avec $\partial\mathcal{X}$ sa frontière, en météorologie ce peut être un domaine spatial de trois dimensions, $\mathcal{M}$ est l'espace des observations, M est la fonction qui projete l'espace $\mathcal{X}$ sur l'espace $\mathcal{M}$; (ix) J est la fonctionnelle objectif. $\mathcal{Z}$ est l'ensemble des variables où J est optimisé. $\mathcal{O}$ est l'ensemble $\{\mathbf{U}, \mathbf{V}, \mathbf{K}, \mathbf{\Theta}, f\}$.

Trois problèmes typiques P1-3 peuvent alors être définis:
(P1) Calibrage: Déterminons le parameteur $\mathbf{K}^*$ tel que $J(\mathbf{K}^*) = \min_{\mathbf{K}\in\mathcal{K}} J(\mathbf{K})$,
avec $J(\mathbf{K}) = \| \mathbf{Y} - \mathbf{Y}_{obs} \|^2_{\mathcal{M}}$

(P2) Assimilation des données: Déterminons la condition initiale $\mathbf{\Theta}^*$ tel que
$J(\mathbf{\Theta}^*) = \min_{\mathbf{\Theta}\in\mathcal{X}} J(\Theta)$ avec $J(\Theta) = \| \mathbf{Y} - \mathbf{Y}_{obs} \|^2_{\mathcal{M}}$

(P3) Contrôle: Déterminons la variable de contrôle $\mathbf{U}^*$ qui optimise une fonction coût J de la forme

$$\int_{t_0}^{t_1} G(\mathbf{X}, \mathbf{U}) dt$$

où $\| \cdot \|_{\mathcal{M}}$ est une norme dans $\mathcal{M}$, et G est une fonction appropriée au problème. Pour tous les problèmes P1-3, des composantes $\{\mathcal{O} \backslash \mathcal{Z}\}$ sont supposées être des facteurs connus.

Le but de l'assimilation de données est de reconstituer l'état d'un système avec l'aide de sources diverses, telles que les statistiques, les mesures directes et le rendement de l'équation principale [44], [14]. Les schémas qui fusionnent les données et le modèle dans l'assimilation de données suivent principalement deux approches : la théorie de contrôle et la théorie de l'estimation. Cette dernière est fondée sur le filtre de Kalman et ses variations [43] ; la première se fonde sur les méthodes variationnelles qui tirent profit de la puissance des techniques de contrôle optimal [69].

Le développement de l'assimilation de données en météorologie est lié à l'évolution des données et des modèles. Au début de la prévision numérique, les models avaient une discrétisation grossière et les méthodes d'assimilation de type interpretation optimale étaient suffisantes; les données et les modèles véritablement raffinés nécessitent des schémas efficaces d'assimilation de données, c'est à dire, filtre de Kalman étendu et méthodes variationnelles quadri-dimensionnelles.

Quelle est la raison pour laquelle un processus d'assimilation de données est habituellement nécessaire ? Elle est principalement due aux incertitudes du système (1.1). C'est-à-dire, toutes les variables dans l'ensemble $\mathcal{O}$ peuvent être considérées comme composées d'éléments connus et inconnus. Mais estimer tous ces termes en une seule fois est une idée trop ambitieuse, c'est pourquoi une stratégie étape par étape serait plus réaliste. Nous distinguons les incertitudes *en ligne* et *hors ligne*. La première est la conséquence des variations temporelles ou spatial-spécifiques du contexte pour le fonctionnement de modèle, par exemple, les incertitudes pour la condition initiale $\mathbf{\Theta}$ et l'entrée $\mathbf{U}$ du modèle. Les incertitudes en ligne peuvent seulement être enlevées en assimilant les observations d'exécution accumulées (c.-à-d. problème **P2**). Tandis que pour des incertitudes hors ligne, une fois estimée par des méthodes inverses, elles sont fixées pour l'utilisation postérieure. Par exemple la valeur du paramètre $\mathbf{K}$ pour le problème **P2** est choisie pour être le résultat du calibrage (problème **P1**).

Comparé avec les sciences environnementales, la modélisation des plantes ou des récoltes présente les mêmes difficultés et défis : les caractéristiques comme la complexité élevée et l'incertitude du système agissant avec l'environnement, aussi bien que des données insuffisantes et imprécises pour l'analyse et la validation des modèles. Cependant, la recherche sur ces plantes nécessite l'étude conjointe des problèmes d'évaluation et des problèmes de contrôle (une meilleure croissance avec des ressources limitées). Tandis qu'en météorologie ou océanographie, nous hésitons à parler de la contrôle. La possibilité de contrôler est le point clé des problèmes de développement durable.

Les modèles de croissance des plantes jouent un rôle essentiel en botanique, agronomie, physiologie, et informatique, ou dans autres secteurs de recherches agronomiques. Différents types de modèles, tels que les modèles basés sur des processus, les modèles géométriques, ou les modèles fonction-structure, ont été développés pour des applications précises.

Les modèles basés sur les processus liés à la physiologie des plantes répondent à des demandes agronomiques. Ces modéles évaluent les flux de carbone par le réseau du 'apparatus' des plantes vers une *récolte* selon un pilotage d'énergie du rayonnement et un écoulement de l'information (e.g. le stress ou des étapes du développement dans un ho-

raire phénologique). Commencé par de Wit en 1970 [35], cette approche a été développée et validée pour les plantes cultivées à une haute densité, c'est à dire, avec le canopy homogène. Les modèles simulent la croissance d'une *plante moyenne et virtuelle*. Les plantes sont considérées comme une grande feuille [2], et la biomasse est donc calculée selon la photosynthèse de cette grande feuille. En pratique, la superficie de cette grande feuille est évaluée par la notion d'indice de surface de feuille (LAI) [15]. La variabilité morphologique (e.g. entre racines et pousses) est négligée. Ce n'est pas le cas, si nous considérons des conditions hétérogènes, telles que des semis à une densité très faible dans des rangées espacées ou pour les plantes individuelles ornamentales. Le contrôle de qualité en horticulture, c'est à dire, pour la forme, la couleur, le goût et la composition des fruits, est également hors de la capacité du modèle basé sur les processus. Au contraire il dépend d'une description détaillée des entités morphologiques des plantes. En outre, la rétroaction dans l'accumulation de biomasse et dans la différentiation architecturale, est habituellement négligée dans cette approche.

Hallé *et al.* en 1978 a créé un modèle d'architecture des plantes visant une description unifiée d'informations topologiques [57]. Cette généralité architecturale a été bientôt considérée par de Reffye *et al.* pour la génération des arbres et des images de plantes fidèles à leur structure botanique et à leur développement en présentant les techniques d'axe de référence [27]. Les informaticiens ont développé les mathématiques basées sur la notion de grammaire, c'est à dire, les L-Systèmes étudiés par Prusinkiewicz *et al.* [91] et l'automate étudié par Blaise *et al.* [9], pour le formalisme de la morphogénèse pure des plantes (notez qu'Alfonseca et Ortega ont prouvé l'étape-équivalence des L-Systèmes et de l'automate [1]). Bien que la plante 3D, y compris l'information topologique et la géométrie d'organe, est entièrement mise en œuvre, les modèles architecturaux répondent principalement à la demande en aménagement, publicité ou jeux vidéo, mais ont peu d'applications dans la recherche agro-écologique.

L'apparition de modèle fonction-structure des plantes (FSPMs) répond au défi de traiter l'hétérogénéité de l'environnement pour les modèles basés sur les processus en tirant profit de la description spatiale de variabilité du modèle d'architecture. A contrario les modélisateurs architectuaux tentent de fournir leur moteurs phénologiques avec la biomasse incrémentiele calculée selon la connaissance physiologique des plantes. Nous nous référons à des FSPMs comme LIGNUM [88], AMAPhydro [30], GroGra [65] et COTONS [61]. Ces modèles traitent habituellement quelques composants structuraux de base, c'est à dire, l'unité élémentaire idéalisée [104] ou le métamer [3], les rapports de source-puit pour la distribution de biomasse, et un formalisme pour la photosynthèse. *GreenLab* [130] a été conçu pour réduire la charge de calcul due à la simulation fortement complexe des événements discrets des ces FSPMs. Cette complexité pénalise l'analyse et les applications des modèles, aussi bien que l'identification de paramètres. Le développement de *GreenLab* est un processus continue pour équilibrer la simplicité et la complexité en choisissant et en adaptant la connaissance biologique (c.-à-d. [57]) et mathématique (c.-à-d. [114]) pour former un modèle dynamique efficace utile pour une grande variété d'applications, par exemple en agronomie et en sylviculture [86].

Jusqu'ici il n'y a aucun formalisme spécifique de photosynthèse validé pour FSPMs au niveau des organes. Le workshop FSPM'04 [55] donne l'impression que le formalisme de la photosynthèse n'a pas été établi avec succès et qu'il n'a pas été examiné, mais ce

n'est qu'une question de temps, c'est à dire que le procédé de recherches est en cours. D'abord on doit calculer (ou mesurer) au niveau des organes la distribution des variables physiques (Chelle a introduit le terme *phytoclimate* [21]), telles que la lumière, la température, l'humidité, les états de l'eau de sol, etc. Ensuite la photosynthèse est probablement calculée par le modèle au niveau de feuille (e.g. le modèle de Farquhar [38] intégré dans RAPT [103]), ou le modèle au niveau de plante entière (concept de LAI de modèle de récolte, de Visser [34]), ou par leur équilibre (Drouet [36]).

La prévision et le contrôle ne sont pas des concepts nouveaux dans l'agriculture [87], [100], [37]. Toutefois ces travaux se concentrent plutôt sur des techniques de contrôle, i.e. le contrôle de point de réglage [109], le contrôle hiérarchique [20], [116], [117], et le contrôle intelligent [74], [24], [62], mais n'utilisent pas les méthodes variationnelles qui essayent de mettre en application les étapes de Lions. Les modèles choisis pour ces techniques de contrôle manquent habituellement d'un concept d'assimilation de données pour l'évaluation des incertitudes (notez que récemment des concepts d'assimilation de données sont acceptés dans l'évaluation de récolte [67], [41]), ce qui pourrait mener aux erreurs significatives de validation. Bertin et Heuvelink [8] montrent l'exemple de trois expériences où l'erreur est raisonnable alors que pour la quatrième elle est de 35%. En outre, la stratégie de réduction est populaire pour alléger la complexité de ces techniques de contrôle [101], [110]. Cependant, en raison de la complexité et de la jeunesse du bourgeonnant FSPMs, il y a très peu de tentatives d'application de la théorie du contrôle dans ce domaine. Le contrôle de l'effet d'éclaircie basée sur AMAPhydro [86] en est un exemple.

Après avoir adapté les méthodes variationnelles au modèle des plantes, on peut se poser la question : pouvons-nous transférer les expériences de conceptions et les méthodologies des méthodes variationnelles pour la modélisation et les applications de FSPMs ? Si oui, comment et dans quelle mesure?

Alors une série de questions vient après la question posée ci-dessus : (i) comment les méthodes variationnelles doivent-elles être utilisées dans la inter-évolution entre modèle et des données (semblable à cette de l'assimilation de données)? (ii) un FSPM comme *GreenLab* ainsi que des méthodes d'optimisation peuvent-ils fournir de nouveaux indices dans l'organisation et le traitement des énormes données d'expérience pour les agronomes ? (iii) comment agissent-ils les méthodes variationnelles, quand FSPMs devient de plus en plus complexe ? Par exemple le formalisme de photosynthèse pourrait être plus précis et plus compliqué. Les modèles des plantes doivent considérer différentes échelles de temps, i.e. en heures, en jours, etc., ou l'indice thermique. D'ailleurs ces FSPMs pourraient devenir plus en plus impliquées avec l'environnement hétérogene. (iv) quand l'approche analytique devient difficile, qu'en est il de l'exécution des méthodes numériques ? Nous espérons que les méthodes numériques conviennent aussi, quand le modèle devient complexe. (v) comment mettre en application les concepts dans les étapes de Lions pour FSPMs, et à l'envers comment ces concepts contribuent à la communauté de FSPM, particulièrement pour des concepts d'assimilation de données?

La thèse en fait est motivée par trois articles dans les proceedings d'une réunion du CNRS intitulée "Tendances nouvelles en modélisation pour l'environnement" : les remarques de J.L. Lions sur les sciences environnementales [78], l'introduction du modèle de morphogénèse des plantes par Ph. de Reffye [28], et l'explication de l'assimilation de

données par F.-X. Le Dimet [71]. Dans cette thèse, nous essayons de chercher les réponses à la question de base et aux suivantes formulées ci-dessus.

Chapter 1

Introduction

Variational methods, or calculus of variations, is the art to solve mathematical problems that deal with extremal *functional* questions. Such *functionals* are functions of functions, that is to say, the independent variables of a functional are also functions, for instance, an integral of the terms that contain an unknown function and its derivatives is a typical functional.

Variational methods have been retrieving the attentions of the mathematicians from the Newton age until the present time both in their theory developments and in their applications. Initialized by the Bernoulli brothers, the *brachystochrone problem* [5] started the research in this field, then followed a long line of great mathematicians, i.e., Leibniz, Newton, Huygens and l'Hospital, even Gallilée. Euler and Lagrange derived the necessary conditions for functional problems with differential equations as subsidiary conditions, known as Euler-Lagrange equations. For extensive details about variational methods in this period, please refer to [47]. Enriched by the contributions of Weierstrass, Hamilton, and Morse, in 1960s the variational methods prevailed, represented by Pontryagin's maximum principle [89] and Bellman's dynamic programming [4].

The aim of variational methods or calculus of variations is to formulate the general laws of nature that suggest the principle of the economy of means. Nature seems proceeding in the simplest, as Newton wrote in *Principia*:"Nature does nothing in vail, and more is in vain when less will serve; ...", and as Euler believed in his *Methodus inveniendi*:"Every effect in nature follows a maximum or minimum rule." Due to this *variational principle*, variational methods reflect the harmony of our nature, as makes it very flexible to be adapted for the applications in diverse branches of physics, biology, economy and engineering.

In the late 1960s, J.-L. Lions formulated control of systems governed by partial differential equations [75], [76]. He introduced variational methods in applied mathematics and constantly supported the development in this field. In the 1990s, he drew some remarks on mathematical modelling and environment [77], [78]. These remarks founded the framework of the research in environmental science. He proposed the following steps of a general methodology for better comprehension of environment [78]:

Step 1: Mathematical modelling for the reality.
Step 2: Mathematical analysis of the model and numerical simulation.

Step 3: Validation and correction with available observation data.
Step 4: Control of the system.

Lions highlighted the unknown components for initial conditions, boundary conditions and model itself in a general case. The unknown components can be estimated by solving an inverse problem that deals with the minimization of a norm for the discrepancy between the observation and model output, as is also called calibration. A question, which is conceptually similar to inverse problem, is the so-called data assimilation that aims at estimating the evolution process with all kinds of available information. In brief, the Lions steps can be summarized as modelling, calibration, validation and control.

In recent years, variational methods have been increasingly experienced in environmental science, i.e. meteorology, oceanography, and hydrology. F.-X. Le Dimet and I.M. Navon provide a comprehensive review on variational and optimization methods in meteorology emphasizing on data assimilation [70]. We show the ingredients of variational methods in the following system

$$\begin{cases} \dfrac{\partial \mathbf{X}}{\partial t} & = F(\mathbf{X}, \mathbf{K}, \mathbf{U}) + B(\mathbf{V}), & \mathbf{X} \in \mathcal{X} \\ \mathbf{X}(t_0) & = \mathbf{\Theta} & \\ C(\mathbf{X}) & = f, & \mathbf{X} \in \partial\mathcal{X} \\ \mathbf{Y} & = M(\mathbf{X}) & \mathbf{Y} \in \mathcal{M} \\ \text{opt } J & = J(\mathbf{Z}), & \mathbf{Z} \in \mathcal{Z} \end{cases} \tag{1.1}$$

where (i) $\mathbf{X}$ is state variable, in meteorology it can be vector of fluid velocity, pressure, temperature and humidity for the atmosphere; (ii) F is differential nonlinear operator for ordinary or partial differential equation; (iii) $\mathbf{U}$ is control variable, or input variable; (iv) $\mathbf{K}$ is parameter set; (v) $\mathbf{V}$ is model error, on which imposes an operator B for the connection of model error with model dynamics; (vi) $\mathbf{\Theta}$ is initial condition; (vii) operator C and function f are for boundary condition; (viii) M is observation map function that links model output $\mathbf{Y}$ with some observations $\mathbf{Y}_{obs}$; (ix) J is the objective functional. Here $\mathcal{X}$ is the state space with $\partial\mathcal{X}$ as its frontier, and in meteorology it can be a spatial domain of three dimensions, $\mathcal{M}$ is the space for the observation, $\mathcal{Z}$ is the variable set with respect to which J is optimized, and let $\mathcal{O}$ be the set of $\{\mathbf{U},\mathbf{V},\mathbf{K},\mathbf{\Theta},f\}$. Three typical problems P1-3 can then be defined:

(P1) Calibration: Determinate parameter $\mathbf{K}^*$ such that $J(\mathbf{K}^*) = \min_{\mathbf{K}\in\mathcal{K}} J(\mathbf{K})$,
with $J(\mathbf{K}) = \| \mathbf{Y} - \mathbf{Y}_{obs} \|^2_{\mathcal{M}}$.
(P2) Data assimilation: Determinate initial condition $\mathbf{\Theta}^*$ such that
$J(\mathbf{\Theta}^*) = \min_{\mathbf{\Theta}\in\mathcal{X}} J(\Theta)$ with $J(\Theta) = \| \mathbf{Y} - \mathbf{Y}_{obs} \|^2_{\mathcal{M}}$.
(P3) Control: Determinate control $\mathbf{U}^*$ to optimize some objective functional J like the integral form (*Lagrange problem*) $\int_{t_0}^{t_1} G(\mathbf{X}, \mathbf{U}) dt$.

where $\mathcal{K}$ is the parameter space, $\| \cdot \|_{\mathcal{M}}$ is a norm in $\mathcal{M}$, and G is a problem-relevant function. For all the problems P1-3, the components $\{\mathcal{O}\backslash\mathcal{Z}\}$ are assumed to be known factors.

The goal of data assimilation is to reconstruct the model state with the help of different

sources, such as statistical features, direct measurements and output of governing equation [44], [14]. The schemes that merge data and model in data assimilation follow mainly two approaches: control theory and estimation theory. The latter is based on Kalman filter and its variations [43]; the former relies on variational methods that take advantage of the full power of optimal control techniques [69].

The development of data assimilation in meteorology is a consistent inter-evolution of both data and model from coarseness to refinement. In the beginning, poor models and few data made it hard to insert data into model by "nudging" methods. Gradually plentiful data sets and truly fine models empower the efficient data assimilation schemes, i.e. extended Kalman filter and four-dimensional variational methods.

What is the reason that a data assimilation process is usually necessary? It is mainly due to the uncertainties of the system (1.1). That is, all the variables in the set $\mathcal{O}$ can be considered to consist of known and unknown components. The idea is that it is too ambitious to estimate all the unknown items in only one experiment by inverse methods. Instead a step-by-step strategy would be more realistic. Lions thus defined the *sentinel problem* that identifies certain unknown components according to local environment conditions without consideration of other unknown items [78]. We distinguish *online* and *offline* uncertainties. The former result from temporal or spatial-specific variations of the context for model running, for instance, the uncertainties for initial conditions $\mathbf{\Theta}$ and model input $\mathbf{U}$. The online uncertainties can only be removed by assimilating the accumulated runtime observations (i.e. problem **P2**). While for offline uncertainties, once estimated by inverse methods, they are fixed for later usage. For example the parameters value $\mathbf{K}$ for problem **P2** are chosen to be the calibration results of problem **P1**.

Compared with environmental sciences, plant or crop modelling has to face the same difficulties and challenges: features like high complexity and uncertainty of the system interacting with environment, as well as incomplete data for model analysis and validation. However, plant research distinguishes itself in that in addition to problems of estimation, control problems can also be defined for a better growth with limited resources. While in meteorology or oceanography, we hesitate to talk about control. The possibility of control is the key point in sustainable development issues.

Plant growth model plays an essential role in botany, agronomy, physiology, and computer sciences, or other plant-relevant research areas. Different kinds of models, such as process-based models, geometric models, or functional-structural models, have been developed for special applications.

Process-based models that is relative to plant physiology try to answer agronomic demands. They concern about the material flow of carbon from environment through the network of plant 'apparatus' into a pool called *yield* depending on an energy piloting of radiation and an information flow like stress or stages of development in a phenological timetable. Started by de Wit in 1970 [35], this approach has been developed and tested for field crops at a high density, that is, with homogenous canopies. The process-based models simulate the behavior of a *virtual mean plant* under the concept of one big-leaf [2], whose surface area is evaluated by introducing the notion of leaf area index (LAI) [15]. Therefore the morphological root or shoot variability vanishes in the larger compartments.

This is seldom to be true when considering heterogenous conditions, such as crops at a very low density in spaced rows or individual stands in ornamentals. The quality control in horticulture, i.e. for the shape, color, taste and composition of fruits, is also far beyond the abilities of process-based model, nevertheless depends on a detailed description of plant morphological entities. Furthermore, feedback between biomass accumulation and plant architectural differentiation in organogenesis and morphogenesis, is usually neglected.

Hallé *et al.* in 1978 gave birth to plant architecture model aiming at a unified description of topological information [57]. This architectural generality was soon accepted by de Reffye *et al.* for the generation of trees and plants images faithful to botanical structure and development by introducing reference axis techniques [27]. Computer scientists developed grammar-based mathematics, i.e. L-Systems by Prusinkiewicz *et al.* [91] and automaton by Blaise *et al.* [9], for the formalism of pure plant morphogenesis (Note that Alfonseca and Ortega have proved the step-equivalence of L-Systems and automaton [1]). Although 3D plant, including topological information and organ geometry, is fully implemented, architectural models answer mainly the demand in landscaping, advertising or video games, thus lead to few applications in agro-ecological research.

The emergence of functional-structural plant models (FSPMs) somehow answers the challenge of dealing with heterogeneity of environment for process-based model by taking advantage of the spatial variability description of architecture model. By contrast, architecture modelers attempt to furnish their phenological engines with calculated incremental biomass for the organogenesis according to plant physiological knowledge. We refer to such FSPMs as LIGNUM [88], AMAPhydro [30], GroGra [65] and COTONS [61]. These models usually deal with some basic structural components, i.e. idealized elementary unit [104] or metamer [3], source-sink relationships for the biomass partition, and a formulism for photosynthesis. FSPM *GreenLab* [130] was designed to reduce the computation load due to the highly complex discrete-event simulation of those FSPMs. This complexity hampers the model analysis and applications, as well as parameter identification. The development of *GreenLab* is a constant process of balancing the simplicity and complexity when choosing and adapting the biological (i.e. [57]) and mathematical knowledge (i.e. [114]) to form an efficient dynamical plant model useful for a wide variety of applications, say in agronomy and forestry [86].

So far there is no validated specific photosynthesis formulism for FSPMs at organ level. From the latest workshop FSPM'04 [55], we have the impression that although the photosynthesis formulism has not been successfully established and tested, it is just a matter of time, that is, the research process is clear. Firstly one needs to calculate (or measure) at organ level the distribution of physical variables, such as light, temperature, humidity, soil water conditions, etc., as is so-called phylloclimate (Chelle has introduced this term in his review of the subject [21]). Then photosynthesis is possibly calculated by leaf-level model (i.e. Farquhar model [38] integrated in RAPT model [103]), or whole-plant-level model (LAI concept of crop model, de Visser [34]), or by their balance (Drouet [36]).

Prediction and control are not brand-new concepts in agriculture [87], [100], [37]. However these researches concentrate rather on control techniques, i.e. set point control [109], hierocratical control [20], [116], [117], and intelligent control [74], [24], [62], but not a framework of variational methods that try to implement the Lions steps. The chosen models

for these control techniques usually lack of a *data assimilation* concept for the estimation of uncertainties (note that recently data assimilation concepts are being accepted in crop estimation [67], [41]), as might be the reason that they can lead to significant validation errors. For instance we can see reasonable errors for 3 experiments except for a 35% mis-estimation in the Montfavet experiment in [8]. Furthermore, reduction strategy is popular to alleviate the complexity of these control techniques [101], [110]. However, due to the complexity and juvenility of the burgeoning FSPMs, there are very few attempts on applying control theory in this domain. One noticeable example of this kind is about the control of pruning effect based on AMAPhydro [86].

After the travel from variational methods to plant modelling, we take a rest to ask the question: can we transfer the experiences in the conceptions and methodologies of variational methods into the modelling and applications of FSPMs? If yes, how and to what extent?

Then a series of questions come after the above basic one: (i) how variational method play its role in the inter-evolutional development of model and data in FSPMs similar to that in data assimilation? (ii) Can a FSPM like *GreenLab* together with optimization methods provide new clues in the organization and treatment of the enormous experiment data for the agronomists? (iii) How about the application of variational methods when FSPMs become increasingly complex? For instance, the photosynthesis formulism might be more precise and complicated. Plant models might take account of different time scales, i.e. calendar time index (in hours, days, etc.) and thermal time index. FSPMs might become more involved with environmental spatial varieties. (iv) When analytical approach becomes difficult, what is the performance of numerical methods? We hope the numerical methods be suitable too, as model becomes complex. (v) How to implement the concepts in Lions steps for FSPMs, and in reverse what do those concepts contribute to FSPM community, especially for data assimilation concepts?

The thesis is in fact motivated by three articles in the proceedings of a CNRS reunion titled "Tendances nouvelles en modélisation pour l'environnement": the remarks of J.L. Lions on environmental sciences [78], the introduction of plant morphogenesis model by Ph. de Reffye [28], and the explanation of data assimilation by F.-X. Le Dimet [71]. We try to seek the answers of the basic and ensuing questions above in this thesis.

In chapter 2, we introduce the functional-structural dynamics of *GreenLab*. Its strategy consists of firstly a simplification to grasp main factors, i.e. the source-sink relationships and the feedback of architectural information, then an evaluation of the simplification by inverse methods to fit the model parameters. We emphasize the paradigm that *GreenLab* should provide the interface for the complexisation. A photosynthesis formula is proposed to mimic plant growth with a highly simplified physiological basis. We also formulate the soil-plant interaction benefiting from the soil water balance equation introduced by De Reffye.

The chapter 3 is dedicated to the formularization of the variational methods based on *GreenLab* plant functional-structural dynamics for various applications, namely parameter identification, data assimilation, and optimal control. Adjoint equations are set up in the optimality systems for gradient calculation. We also show the flexibility of the variational

formulism for the same applications when smaller time scale within the thermal time index – growth cycle (GC) – is taken into account. The plant dynamical systems can be much more complicated, but with variational methods we succeed in formulating the adjoint equations for the new optimality systems in the refined time scale.

The chapter 4 concerns the numerical methods of the variational approach. We apply automatic differentiation (AD) techniques when the model implementation is inconvenient or too complicated for us to write the adjoint code. The automatic differentiation theory is briefly introduced. The relationship between AD and adjoint equation is also investigated. Then we summarize the steps of writing the AD code by hand line by line, and record the preliminary AD coding experience for project *GreenLab*.

Identification of the environmental factor parameters is the main topic of the chapter 5. We conduct twin experiments to generate artificial observations for identification. Variational methods are employed to track back the parameter values. Levenberg-Marquardt algorithm is adopted for iterative optimization benefiting from the special structure of least squares. Gradients are calculated by finite difference, since there are only four environmental parameters to be calibrated. The results of the numerical experiments show that parameter values can always be identified, no matter how environmental conditions fluctuate when leaf expansion period is considerably long. The influence of observation errors on the calibration results is also investigated.

In chapter 6, we firstly perform a model analysis on optimal sink strength of maize fruit, then an optimal control problem for sunflower water supply is presented and numerically solved. The resulting optimal water supply at each GC provides more refined irrigations, as reveals possible agronomic applications. Considering *online* uncertainties on initial condition $\mathbf{\Theta}$ and model input $\mathbf{U}$ for *GreenLab*, data assimilation problem is introduced for the control of these *online* uncertainties by assimilating the observations generated by twin experiments at GC level.

The last chapter contains conclusions that try to answer the questions in this introduction. Here we emphasize that our subject is an ideal type of *virtual plant* that is too simple to be the real one growing in field. Realistic applications of the thesis results urge further studies on FSPM physiological knowledge. Possible applications of FSPMs as a plant-environment system in such a high dimension are too complex for plant physiologists themselves, therefore, an interdisciplinary approach, especially with collaborations between plant physiologists, computer scientists, and mathematicians, is needed.

Chapter 2

Introduction to *GreenLab*: basics, dynamics, and model analysis

Résumé

Dans ce chapitre, nous présentons la dynamique de la structure fonctionelle de *GreenLab*. Sa stratégie de misé en œuvre se compose, d'abord d'une simplification pour décrire les facteurs principaux, c'est à dire, les rapports entre les sources-puits et la rétroaction d'information architecturale, ensuite d'une évaluation de la simplification par des méthodes inverses pour adapter les paramètres des modèles. Nous soulignons ici que *GreenLab* devra prendre en compte ce niveau supérieur de complexité. On propose une formule de photosynthèse pour imiter la croissance des plantes avec une base physiologique fortement simplifiée. Nous formulons également l'interaction entre sol et plantes grâce à l'équation de *soil water balance* introduite par Ph. De Reffye.

2.1 *GreenLab* functional-structural dynamics: basic and beyond

2.1.1 Introduction

Plant is an open complex system that undergoes exchange of energy flow with environment to support an *order* or *harmony* of plant form, namely architecture information (i.e. Hallé 's 23 models [57], figure 2.1, a-g). However, detailed architecture information of plant forms appears to be extremely variable, such that one can never observe two completely identical plant individuals. This remarkable variability of architectural growth possibly results from genetic reasons with respect to species and from interactions with environment.

Complex system is usually supposed to consist of microlevel *nonlinear* components. Although each might be simple, the nonlinear interactions among these micro-components result in complex dynamic behaviors, i.e. order or predictable patterns, chaos, self organizations at the edge of chaos [125], and self-replications [66].

We distinguish *genotype* and *phenotype* in that genotypes, generalized as local rules of the evolutions of micro-components, can generate complex behaviors of macro-phenomena, named as phenotypes. Wolfram suggests a two-scale automaton on a slow time scale of

controlling (master) components and on a fast time scale of controlled (slave) components to describe the fast processes of phenotypes of organisms determined by slow processes of genotypes [126]. Morphogenesis is then defined as the development of phenotypes.

Complex system concepts have been experienced in FSPM community, i.e. Sachs regards the remarkable variability of plant forms as dynamic self-organization resulting from interactions of microlevel branches [99]. It is yet unknown how *genotypes* organize such an architectural harmony during the evolution of complex system, although Rashevsky has done a inspiring research to model the morphogenesis of *phyllotaxis* by the dynamics of a ring of cells equipped with morphogen [97].

GreenLab was not designed to elucidate the mechanism of the plant complex system, i.e. how genotypes govern phenotypes, but rather a *description* of growth patterns for phenotypic plasticity of linear and branch patterns for vascular differentiation in the sense of functional-structural dynamics at organ level.

In essence, *GreenLab* follows the approach of reductionism. The word *dynamics* is rather a term of physics. One architectural *elementary entity* of plant functional-structural dynamics is chosen to be a *metamer* ([3], [9], another name *phytomer*) that is composed of a node, the internode from beneath, the apical bud, the associated organs, i.e. leaves or fruits, and the axillary buds that can develop into a branch in a new cycle (figure 2.1-(a), figure 2.2, see also idealized elementary unit [104]). Organs, such as internode, leaf, fruit, layer, are recognized in the metamer as physical *elementary entities* that experience growth process, for instance elongation, with the energy flow exchange under environment conditions of phylloclimate.

GreenLab functional-structural dynamics is characterized by its Newtonian physics. The laws of plant growth are drawn empirically or obtained by simplification under biological hypothesis on morphogenesis and biomass production. One can notice the similar process of hypothesis-laws-dynamics in Newtonian physics, i.e. the universal law of gravitation and its applications in astronomy. Once the laws are set up, theoretically speaking, plant growth is deterministic and reversible, and there will be no unpredictable complex phenomena such as chaos and self-organizations. However, based on simplified Newtonian dynamics, there will be less difficulties in applying modern mathematics on the analysis and applications of FSPMs. In our thesis, variational methods are investigated.

The philosophy of *GreenLab* is somehow simplism. That is, if we can find simpler representation, we will never employ the complex one. On the other hand, the simple representation of the components should be aggregated together for the phenomena that are complex enough. The philosophy is similar to that of Prusinkiewicz in his essay on art and science [93]:

> The ultimate goal of modelling nature is to construct simple yet faithful models of reality.

The development of *GreenLab* is therefore a constant process of balancing the simplicity and complexity when choosing and adapting the biological and mathematical knowledge to form an efficient dynamical plant model useful for a wide variety of applications in

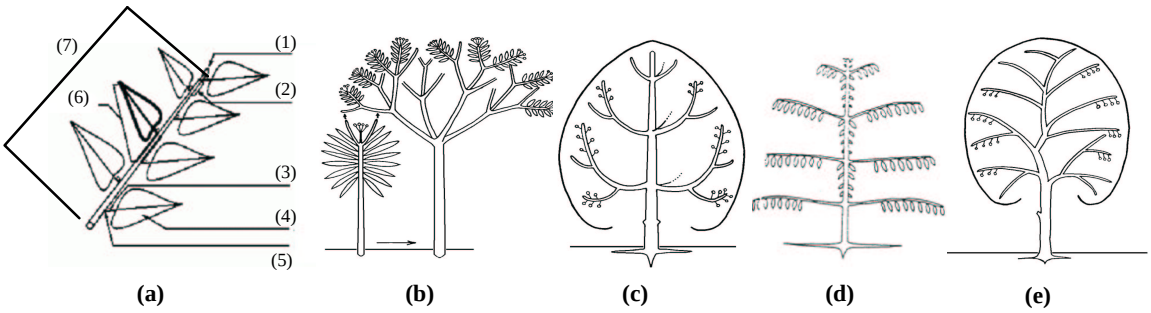


Figure 2.1: Botany notions and Hallé's architecture models [57]. **a.** Foliar axis. (1) Apical bud (2) Internode (3) Axillary bud (4) Leaf (5) Node (6) Metamer (7) Growth Unit. Plant growth originates from some specific cellular tissues, the so-called *meristems*. Elongation of axis is the functioning results of bud at its tip, the so-called *apical bud*. Leaves are inserted into the axis at the *nodes*. Between two consecutive nodes distinguishes the *internode*, and in turn between two internodes is the node. *Axillary buds* stem at leaf's axil. A *metamer* is a compound of associate organs with an internode, and *growth unit* is the part of the stem that corresponds to the growth during a lengthening period. **b-e.** Architectural models. **b.** Leeuwenberg model. **c.** Rauh model. **d.** Massart model. **e.** Troll model. The criteria for this classification relies mainly on growth mode (continous/rythmic growth, phyllotaxis, reiteration), branching patterns (sympodial/monopodial), differentiation of vegetative axis (plagiotropy/orthotropy), the sexuality (apical/lateral position of flowers).

Figure 2.2: Botanic items of a tree, from Figaro

agronomy and forestry.

This process results in a dynamic representation of the *harmony* or *order* of morphogenesis in forms of both architecture and geometry on the basis of a small set of mathematical equations and metamorphic rules. The pure research by simplification is complemented by an optimization process for the calibration of model parameters to fit the experiment data. Consequently the basis for simplification is narrowed or widened according to the analysis of fitting results. In general, the mathematical formalism is employed under botanic constraints to find the laws for this "virtual" plant, i.e., recursive equations, equilibrium of growth, and furthermore calibration and optimization.

2.1.2 Hypothesis

Basic concepts for *GreenLab* hypothesis are that (i) *GreenLab* virtual plant is an average plant; (ii) *GreenLab* modelling shall be faithful to botanical knowledge; (iii) *GreenLab* approach seeks to establish functional-structural dynamics with clear mathematical representations based on hypothesis and simplified rules.

Biological hypothesis

The theoretical plant is assumed to be made up of fresh matter that contains about 80% water (H_2O) and 15% of assimilates (H_2O + CO_2). The relative density is then set to 1.0 g/cm^3 for both plant individuals and assimilates, thus the moisture content (water divided by total weight) is approximately 0.8. *GreenLab* takes into account fresh matter in its calculation. Herein dry matter is assumed to be proportionally related to fresh matter. The simulator neglects the complex process of root system for water absorption, and operates only on ariel part, whose biomass is assumed to be proportional to that of root system.

Phenological hypothesis

Phenological hypothesises specify the temporal-spatio scale of the structural-functional dynamics, that is, the temporal and topological organization.

The topological structure is organized as series of a hierarchically ascending scales: metamer, Growth Unit (GU for short), Bearing Axis (BA), substructure (or branching structures, or architectural unit, see figure 2.3 for some botanical knowledge), and the whole plant individual. The metamorphic variations of the architectural atoms, metamers, are characterized by a notion of *Physiological Age* (PA) . PA of plant refers to the rule-controlled metamorphic phases from vegetative development to floral stage. The concept of PA dates back from the observations of Goethe about the irreversible progressive transform from leafy zone into inflorescence.

The temporal organization is based on the assumption that plants undergo growth cycles (GC) of a biological clock. During each GC the plant metabolism results in the emergence of a cohort of new organs. The GC is somehow like the annual cycle of four seasons, however, the duration of GC is variable and governed by the *Law of Sum of Temperatures* (LST) in agronomy (figure 2.4-a). The LST states that it is quite constant

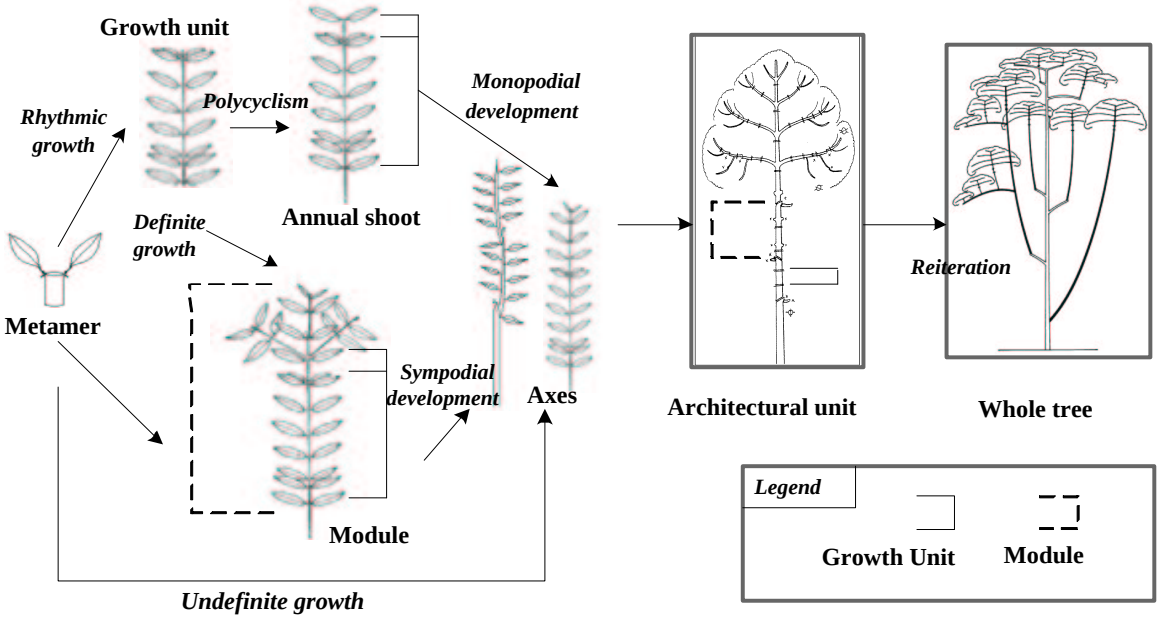
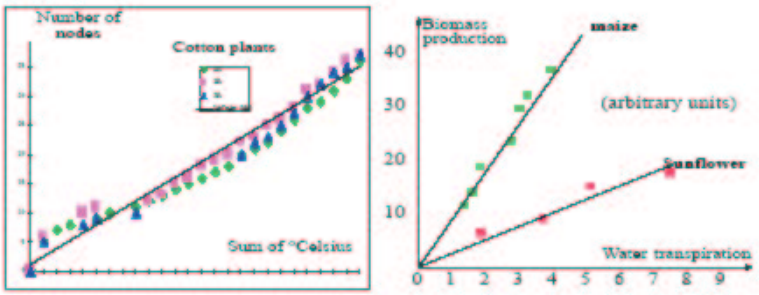


Figure 2.3: Architectural formation of a whole tree: some botanic notions. The juxtaposition of successive metamers in GU, and that of GUs in bearing axis forms a *rhythmic growth*, whereas in *continuous growth* there is only one metamer in each GU. Trees that bear annually one GU are called *monocyclism*, otherwise *polycyclism*, and the successive GUs of one year form the *annual shoot*. The axis comes into being when each of the GU sequence of the axis is born of the apical bud of the previous one (the so-called monopodial development). The axis is said to have an *order*, say i. The *order 1* axis grows out of seed. The axis that is born of the axillary bud of its parent $i-$order axis is said to have a higher order $i+1$, called bearing axis . By *Sympodial development* we mean that the axillary bud takes over the growth behavior of the apical bud, thus the branching structures has the same order of its parent bearing axis (this process is also called reiteration). The apical bud can die in this case. For more details about botanic knowledge, we refer to [27], [19], [3]. Some pictures in this figure are taken from [50].

.



(a) (b)

Figure 2.4: Laws used in *GreenLab* **a.** The law of sum of temperatures. **b.** Linear relationship between dry matter production and plant transpiration, figures from CIRAD

for the accumulation of the daily mean temperature above a genetic-based temperature received for a given plant during each GC. One GC duration can also vary for different species from a few days (herbacious plants) to one year (temperate trees). The count of GC is also called thermal time index that marks the plant *Chronological Age* (CA) . Here we precise the definition of *growth unit* in figure 2.1 in that the *lengthening period* for one growth unit is one *growth cycle*.

Remark 2.1 *PA and CA*
(i) Rivals notes the co-existence of PA and CA [96]. Temporally speaking, during each CA, plants can bear new organs of different PAs, that is, a bud can be born physiologically old, i.e. the newly-born short flowered axes. Spatially speaking, along the bearing axis, one can see the a rank of internodes of different PAs.
(ii) The concepts of both temporal organization of CA and spatial metamorphic organization of PA give birth to a mathematical description, the reference axis [60], for the jump of the status of different PAs along the bearing axis and branches. The refinement of reference axis leads to the so-called dual-scale automaton that features in its synchronization by the notion of CA, and it will be presented in the subsequent sections.

Functional hypothesis

GreenLab investigates the functioning of vegetative organs, namely leaves, for the biomass production to simplify the formularization of plant functional-structural dynamics. At present, this approach is rather conceptual and intuitive, nevertheless, the inverse methods for the parameter identification help to decrease the uncertainty of the intuition.

The *GreenLab* biomass production formulism operates at GC time index. The plant undergoes transpiration via its hydric apparatus network [2]. *Water Use Efficiency* (WUE) is defined as the ratio of photosynthetic carbon gain over water loss. Over sufficient long cultivar season, the long-period WUE_l is quite constant (see figure 2.4-b, [59]). The short-period WUE_s is consider to vary its value when taking into account the variety of the light and temperature conditions. Simplified formula for biomass production can then be obtained by combining the transpiration formula ([2], [29]) and WUE notion. Water stress is not considered.

It is also assumed that the biomass produced by source organs during a GC is gathered in a transitory reservation pool, and then entirely distributed into living organs according to the source-sink relations that settle their competitions for biomass acquisition. The organ expansion duration and expansion law are supposed to be invariable. Geometric shape of certain kind of organ does not change either, i.e. the internode always takes the form of a cylinder or a frustum.

2.1.3 Dynamics formulation

In this section we introduce *GreenLab* dynamics on organogenesis, functioning, and geometric morphogenesis of organs. The time of the functional-structural dynamics is chosen to be the absolute time - Chronological Age, and the space is constructed at organ level.

Morphogenesis: organogenesis dynamics

In this subsection we introduce plant organogenesis. Dual-Scale Automaton (DSA) [136] is reenforced with an emphasis on a Chronological-Age-based dynamics for linear and branch growth patterns. We aim at a complementary study for [32], and notations follow those in [32].

Notations

At each GC for one metamer of Physiological Age p, (i) an apical buds (initially set as seed) forms one GU of a set of new metamers that construct the axis, (ii) each axillary bud gives birth to one GU that construct the secondary branches. The two kinds of growth process consequently produce apical or lateral *substructures* that represent the self-similarity within the plant whole structure. The new metamers born of both apical and axillary buds may have the same PA p or a higher PA q. Thus the metamer is identified by 4 indices and denoted as $m_{pq}(k,n)$:

- The CA n of the plant.
- The CA k of the metamer, that is, the organs of this metamer have appeared for k GCs.
- The PA $p \in \mathcal{P} = \{1,\ldots,P\}$ of the bearing axis that the metamer belongs.
- The PA $q \in \mathcal{P}'_p = \{0\} \cup \{p,\ldots,P\}$ of the branches that result from the axillary buds of the metamer.

Here $q = 0$ indicates no axillary buds, P is the maximal PA, there are totally $f(P)$ types of metamers:

$$f(P) = \sum_{i=1}^{P}(i+1) = \frac{P(P+3)}{2}. \tag{2.1}$$

A metamer may bear several organs of $o-$type, whose number is denoted by $m^o_{pq}(k,n)$ ($o \in \mathcal{O} = \{e,a,f,c,r\}$, where e stands for internodes, a for leaves, f for fruits, c for layers or rings, r for root), as well as apical bud $m^A_{pq}(k,n)$ of number b^A_{pq} and several axillary buds $m^L_{pq}(k,n)$ of number $b^L_{pq}, q \in \mathcal{P}_p = \{p,\ldots,P\}$. Usually b^A_{pq} equals one or zero (death of

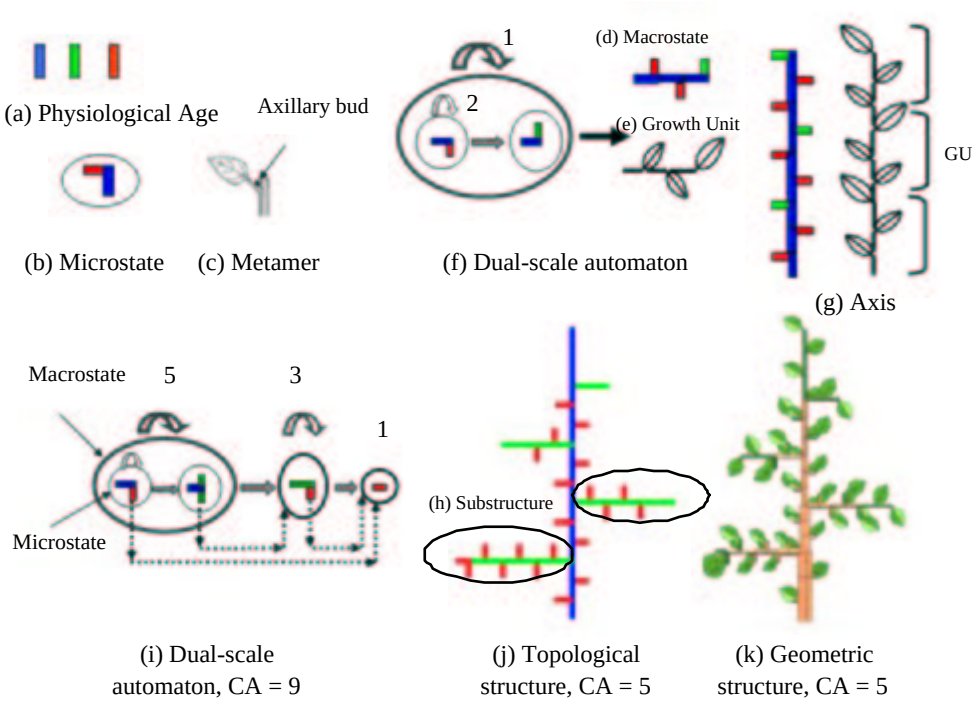


Figure 2.5: Dual-scale automaton

apical bud).

Metamers of same CA k at different plant CA, say n and $n+1$, have different sizes, due to the environment oscillations and the change of sink abilities of that type of organ to attract biomass. However when considering topological structures, the geometry of organs is not of our interest, therefore metamer $m_{pq}(k,n)$ is reduced to m_{pq} with two indices p and q, for buds similarly we have m^B_{pq} for $m^B_{pq}(k,n)$, $B \in \mathcal{B} = \{A, L\}$.

Dual-Scale Automaton (DSA) [136]

Now we consider the topological occupation of organogenesis. The Growth Unit of PA p, denoted as U_p, is a succession of metamers m_{pq} repeated r_{pq} times, here q for U_p is chosen from a PA index set $\mathcal{Q}_p \subseteq \mathcal{P}'_p$ according to biological rules or observations. The bearing axis is a concatenation of GU of different PAs together with the final apical bud. The apical bud of U_p can die or mute to older PA μ_p after τ_p-times repetitions of U_p. The axillary buds of metamer m_{pq} produce U_q that starts the growth of the secondary branches.

The process above can be described by *dual-scale automaton* thanks to the notions of macrostate and microstate. Microstate is defined to be the metamer that is characterized by the PA of its bearing axis and the PA of its axillary buds, and macrostate corresponds to the growth unit. Therefore a macrostate consists of succession of microstates. The concatenation of macrostates reflects the rhythmic growth, and forms the topological structure of the whole plant (figure 2.5).

example 1

The parameters for the plant is as

$$\begin{array}{c} P=3, N=9; \quad b_{pq}=1, \forall q \in \mathcal{Q}_p; \quad r_{13}=2, r_{12}=1, r_{23}=2; \\ \tau_1=5, \tau_2=3, \tau_3=1; \quad \mu_1=2, \mu_2=3, \mu_3=\bullet \end{array} \tag{2.2}$$

where $\bullet$ denote death. N is the maximal plant CA. The status transition graph of dual-scale automaton for this example is shown in figure 2.5. □

Formulation of organogenesis with an emphasis on CA-based dynamics

Diagrams in figure 2.5 are rather direct graphs resulting from the state transitions of automata. The transition functions for both macrostate and microstate automata are given descriptively in [136], [130]. We emphasize that DSA is already equipped with concept of Chronological Age, however, it lacks the description of a CA-based dynamics.

Definition 2.1 *We summarize the configuration matrices as follows*

$$\begin{cases} \overrightarrow{\mu} & \equiv [\mu_p]_{1\times P}, & \text{Mutation vector of PAs for apical buds} \\ \overrightarrow{\tau} & \equiv [\tau_p]_{1\times P}, & \text{Repetition vector for macrostates } U_p, p\in\mathcal{P} \\ \overrightarrow{\varphi^o} & \equiv [\varphi_i^o]_{1\times N}, & \text{Functioning vector for o-type organ} \\ \mathbf{R} & \equiv [r_{pq}]_{P\times(P+1)}, & \text{Repetition matrix for microstate } m_{pq} \text{ in } U_p \\ \mathbf{B} & \equiv [b_{pq}^B]_{P\times P}, & \text{Count matrix for axillary bud in metamer } m_{pq} \\ \mathbf{M_o} & \equiv [m_{pq}^o]_{P\times(P+1)}, & \text{Count matrix for organs in metamer } m_{pq}, o\in\{b,f\} \end{cases} \tag{2.3}$$

Functioning status φ_i^o indicate the appearance of $o-$type organ, precisely 0 for inexistence, 1 for appearance, herein N is the maximal Chronological Age. The $p-$row of microstate repetition matrix $\mathbf{R}$ signifies the repetition time r_{pq} of metamer m_{pq} in macriostate U_p. When $q\notin\mathcal{Q}_p$, we have $r_{pq}=0$. Usually the occurrences of different types of metamers comply with a ascending order of q. The first column of $\mathbf{R}$ corresponds the repetition time of metamers that have no axillary buds. The $p+1$ column of $p-$row indicates r_{pp} times of repetition of metamer m_{pp} and so on. In the case of all metamers have axillary buds, we denote $\bar{\mathbf{R}}$ for residue matrix after the erasion of the first column of $\mathbf{R}$ (0-valued), similarly $\bar{\mathbf{M}}_\mathbf{o}$ for $\mathbf{M_o}$.

The configuration Λ is defined as set of configuration matrices $\Lambda=\{\overrightarrow{\mu},\overrightarrow{\tau},\mathbf{R},\mathbf{B},\mathbf{M_o}\}$.

Definition 2.2 *We define the succession order of metamer occupation in macrostate U_p, that is, for $q_1,q_2\in\mathcal{Q}_p$, succession order,*

$$q_1 \prec q_2, \tag{2.4}$$

means that the apical bud $m_{pq_1}^A$ gives birth to metamer m_{pq_2}. The metamer m_{pq_1} is called the ascendant of metamer m_{pq_2}, and in reverse m_{pq_2} is the descendant of m_{pq_1}. The last metamer in U_p is called Terminal Metamer (TM) of U_p. The first index in the ordered sequence $\mathcal{Q}_p$ is denoted by $\underline{q}$, and the last $\overline{q}$, thus TM of PA p is denoted by $m_{p\overline{q}}$.

Definition 2.3 *Growth Unit formulation*
The Growth Unit U_p of PA p is a succession of metamers, each metamer except TM gives

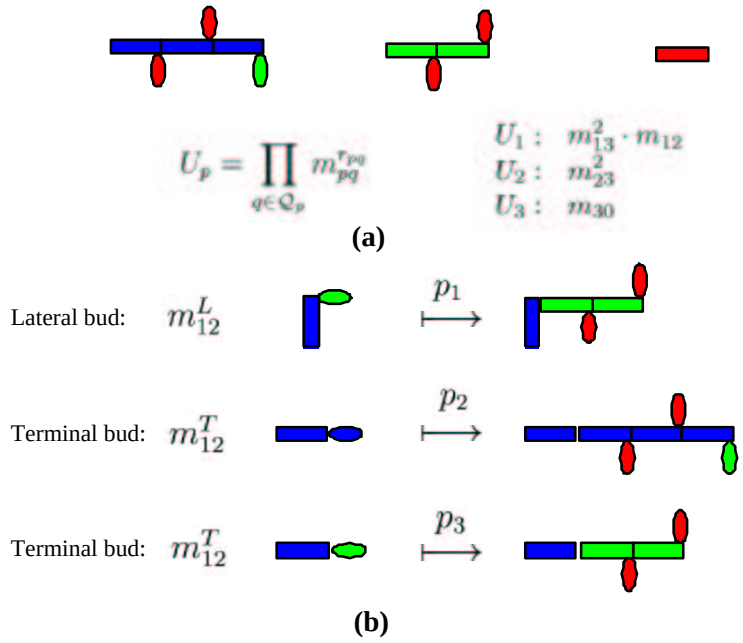


Figure 2.6: Growth Unit formulation (a) and growth rules (b) for example 1.

birth to its descendant during certain period, the so-called ***plastonchron****, we denote the formulation of Growth Unit U_p for one Growth Cycle as*

$$U_p = \prod_{q \in \mathcal{Q}_p} m_{pq}^{r_{pq}}. \tag{2.5}$$

The relation $a \cdot b$ means adjacent occupation of metamer a and b on the axis (note that the relation $\cdot$ bears no commutativity, that is, $a \cdot b \cdot c \neq a \cdot c \cdot b$). There are totally $\sum_{q \in \mathcal{Q}_p} r_{pq}$ plastonchrons in one GC. The sequence $\prod_{q \in \mathcal{Q}_q}$ follows an implicit ascending order of q.

The Growth Unit formulation of example 1 is shown as in figure 2.6-(a). For the next GC, the axillary buds m_{pq}^L and the apical bud of TM $m_{p\overline{q}}^T$ will give birth to new Growth Units according to the following definition of growth rules based on botanic knowledge.

Definition 2.4 *Growth rule $\mathbb{P}$:*
For growth unit U_p that repeats r times in its corresponding bearing axis, the growth rule for its associated buds $m_{pq}^B, B = \{L, T\}$ can be abstracted as

$$\begin{aligned} p_1 &: m_{pq}^L \longmapsto U_q, \quad && q \in \mathcal{Q}_p, \\ p_2 &: m_{p\overline{q}}^T, \longmapsto U_p \quad && r < \tau_p, \\ p_3 &: m_{p\overline{q}}^T, \longmapsto U_{\mu_p} \quad && r = \tau_p. \end{aligned} \tag{2.6}$$

The corresponding $U_p, r = \tau_p$ is called Terminal Growth Unit (TGU).

The growth rules of example 1 are shown as in figure 2.6-(b).

Definition 2.5 *Length function:*
The expected number of new buds in some GU U_l that result from for bud m_{pq}^B is named length function of metamer m_{pq}, denoted lg*, and equals to (rule 2.6)*

$$\lg(m_{pq}^B) = b_{l,\overline{m}}^A + \sum_{m \in \mathcal{Q}_l} r_{lm} b_{lm}. \tag{2.7}$$

The length function of growth unit U_l is defined as $\lg(U_l) = \lg(m_{pq}^B)$

example 2 : Holttum model
In this case, we have the maximal PA $P = 1$, the maximal CA $N = n + 1$. There is only one metamer ($M = \{m_{10}\}$) for macrostate U_1, that is $U_1 = m_{10}$. The configuration Λ is as

$$\mathbf{R} = (1 \quad 0), \mathbf{M_o} = (1 \quad 0), \mathbf{B} = \varnothing, \overrightarrow{\mu} = (\bullet), \overrightarrow{\tau} = (n+1), \overrightarrow{\varphi^d} = \mathrm{I}, \overrightarrow{\varphi^f} = (\underbrace{0 \ldots 0}_{n} \quad 1)$$

The growth rules are

$$\begin{array}{ll} p_2 : m_{10}^T \longmapsto m_{10} & r < n+1 \\ p_3 : m_{10}^T \longmapsto \bullet & r = n+1 \end{array}$$

The CA-based *organogenesis dynamics* is,

$$G(i) = m_{10}^i, \quad i = 1, \ldots, n+1. \tag{2.8}$$

□

Plant structure dynamics can be considered as a string rewriting process with metamer $\{m_{pq}(k,n)\}$ as its alphabet governed by growth rules (definition 2.4).

Definition 2.6 *Auxiliary alphabet. For convenience the parentheses ( ) is used to mark certain part of the strings, but does not mean any additional operation. When a metamer m_{pq} has dormant buds, it is marked as $\underline{m}_{pq}$, and after its buds grow into new metamers, it is marked as $\overline{m}_{pq}$. Similar to the notions in [92], one can add auxiliary letters to analysis strings. For instance, let V be extended by $V_E = V \cup \{\ [,\ ]\}$ and $V_\# = V \cup \{\#\}$. The words that are bracketed by [,] are lateral branches. The branches can be covered up by # to show the marked axis.*

example 3: Re-formulation of DSA example 1

From example 1, we have

$$N = 9, P = 3, M = \{m_{13}, m_{12}, m_{23}, m_{30}\}$$

The *configuration* Λ is as follows ($\bullet$ denotes the death)

$$\mathbf{R} = \begin{pmatrix} 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \mathbf{M_a} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \tag{2.9}$$

$$\overrightarrow{\tau} = \begin{pmatrix} 5 & 3 & 1 \end{pmatrix}, \overrightarrow{\mu} = \begin{pmatrix} 2 & 3 & \bullet \end{pmatrix}, \overrightarrow{\varphi^d} = \mathrm{I} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Growth rules are as (2.6) abstracts. Macrostates U_p are interpreted as in figure 2.6-(a) marked with the accolades { }. We list the organogenesis for each CA as (2.10).

$$
\begin{aligned}
CA\ 1: &\ \underline{m}_{13}^2 \cdot \underline{m}_{12} \\
CA\ 2: &\ (\overline{m}_{13}[\underline{m}_{30}])^2 \cdot \overline{m}_{12}[\{\underline{m}_{23}^2\}] \cdot \{\underline{m}_{13}^2 \cdot \underline{m}_{12}\} \\
CA\ 3: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\underline{m}_{30}])^2\{\underline{m}_{23}^2\}] \\
&\cdot(\overline{m}_{13}[\underline{m}_{30}])^2 \cdot \overline{m}_{12}[\{\underline{m}_{23}^2\}] \cdot \{\underline{m}_{13}^2 \cdot \underline{m}_{12}\} \\
CA\ 4: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{23}^2] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\underline{m}_{30}])^2\{\underline{m}_{23}^2\}] \\
&\cdot(\overline{m}_{13}[\underline{m}_{30}])^2 \cdot \overline{m}_{12}[\{\underline{m}_{23}^2\}] \cdot \{\underline{m}_{13}^2 \cdot \underline{m}_{12}\} \\
CA\ 5: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \{\underline{m}_{30}\}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{23}^2] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\underline{m}_{30}])^2\{\underline{m}_{23}^2\}] \\
&\cdot(\overline{m}_{13}[\underline{m}_{30}])^2 \cdot \overline{m}_{12}[\{\underline{m}_{23}^2\}] \cdot \{\underline{m}_{13}^2 \cdot \underline{m}_{12}\} \\
CA\ 6: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \{\overline{m}_{30}\}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2\{(\overline{m}_{23}[\underline{m}_{30}\overline{)}^2\} \cdot \{\underline{m}_{23}^2\}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \{\underline{m}_{23}^2\}] \\
&\cdot\{(\overline{m}_{13}[\underline{m}_{30}])^2 \cdot \overline{m}_{12}[\underline{m}_{23}^2]\} \cdot \underline{m}_{23}^2 \\
CA\ 7: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \{\overline{m}_{30}\}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \overline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2\{(\overline{m}_{23}[\overline{m}_{30}\overline{)}^2\} \cdot (\overline{m}_{23})[\underline{m}_{30}]^2 \cdot \underline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \{\underline{m}_{23}^2\}] \\
&\cdot\{(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{23}^2]\} \\
&\cdot(\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{23}^2 \\
CA\ 8: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \overline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \overline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2\{(\overline{m}_{23}[\overline{m}_{30}\overline{)}^2\} \cdot (\overline{m}_{23})[\underline{m}_{30}]^2 \cdot \underline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{23}^2] \\
&\cdot(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{23}^2 \\
CA\ 9: &\ (\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \overline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \overline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2\{(\overline{m}_{23}[\overline{m}_{30}\overline{)}^2\} \cdot (\overline{m}_{23})[\underline{m}_{30}]^2 \cdot \underline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot \overline{m}_{30}] \\
&\cdot(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2] \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{30}] \\
&\cdot(\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\overline{m}_{30}])^2 \cdot (\overline{m}_{23}[\underline{m}_{30}])^2 \cdot \underline{m}_{30}
\end{aligned}
\tag{2.10}
$$

□

Botanical analysis of example 3

Take CA 3 for instance

$$(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}[(\overline{m}_{23}[\underline{m}_{30}])^2\underline{m}_{23}^2] \cdot (\overline{m}_{13}[\underline{m}_{30}])^2 \cdot \overline{m}_{12}[\underline{m}_{23}^2] \cdot \{\underline{m}_{13}^2 \cdot \underline{m}_{12}\},$$

the *main axis* is

$$\overline{m}_{13}^2 \cdot \overline{m}_{12} \cdot \overline{m}_{13}^2 \cdot \overline{m}_{12} \cdot \underline{m}_{13}^2 \cdot \underline{m}_{12},$$

the *marked axis* is

$$(\overline{m}_{13}\#)^2 \cdot \overline{m}_{12}\# \cdot (\overline{m}_{13}\#)^2 \cdot \overline{m}_{12}\# \cdot \underline{m}_{13}^2 \cdot \underline{m}_{12},$$

and the *lateral branches* are

$$[\mathbf{s}_{13}] = [\mathbf{s}_{23}] = [\overline{m}_{30}]; \quad [\mathbf{s}_{12}] = [(\overline{m}_{23}[\underline{m}_{30}])^2 \underline{m}_{23}^2].$$

Some botanical notions can be illustrated by growth rules. For instance, sympodial development (figure 2.3) of a metamer m_{pq} can be simulated by defining a physiological mutation of death ($\mu_p = \bullet$) for its apical bud m_{pq}^A, and a reiteration of axillary bud, that is, $q = p$. Implementation of rhythmic growth (see figure 2.3, not considered yet in [94]) is straight-forward by the macrostate/microstate definition. Acrotonic growth of macrostate U_p can be described as

$$\lg(\mathbf{s}_{pq_1}) < \lg(\mathbf{s}_{pq_2}), \qquad q_1 \prec q_2, \quad q_1, q_2 \in \mathcal{Q}_p \tag{2.11}$$

where let length function lg of branch $\mathbf{s}_{pq}, q \in \mathcal{Q}_p$ signifies the number of letters other than [and] in word $\mathbf{s}_{pq}$.

Substructure concept

Substructure is an efficient simulation algorithm in tree theory based on self-similarities of plant [32]. In this section we formulate substructure concept and investigate its simulation efficiency.

Definition 2.7 *[32] Substructure $S_p(k,n)$ is defined as a word whose alphabet is the set of metamers $M = \{m_{pq}(k,n) | p \in \mathcal{P}, q \in \mathcal{P}'_p\}$.*

Hereafter we investigate plant topological information, thus the index n of $m_{pq}(k,n)$ is omitted.

Definition 2.8 *Let array $\mu = \{\mu_p, \ldots, \mu_P\}$ and l be its length. Let κ_p be the array of all elements of set $\mathcal{Q}_p$ and m_p be the array length. Bearing Axis of PA p is formulated as*

$$\mathbf{b}_p = \prod_{p=1}^{l} \{ \prod_{q=1}^{m_p} (m_{\mu(p),\kappa_p(q)})^{r_{\mu(p),\kappa_p(q)}} \}^{\tau_{\mu(p)}}. \tag{2.12}$$

Definition 2.9 *Branching substructure of PA p is defined as the bearing axis $\mathbf{b}_p$ together with all the lateral branching substructures (resulting from sprout of lateral buds) that stick to $\mathbf{b}_p$, denoted as S_p.*

From the definition of branching structure, we can observe a recursive mechanism, that is, the lateral branching structures are composed of branching structures with same or higher PAs.

Definition 2.10 *Macrostate substructure R_p of PA p is defined as the macrostate U_p together with all the lateral branching substructures that stick to U_p, denoted as R_p.*

Both S_p and R_p can bear new metamers during the GC, as consequently change their topological structures. We therefore denote $S_p(k)$ and $R_p(k)$ as the substructures that appeared k GC before, that is, the substructures has a CA k. When considering CA n of the whole plant individual, the geometric properties of substructures $S_p(k)$ and $R_p(k)$ evolve accordingly, hence substructures are denoted as $S_p(k,n)$ and $R_p(k,n)$ in this case.

Therefore the complete plant is $S_1(n,n)$. The bud that is destined to bear S_p is denoted as s_p.

Considering tag $S_1(3)$ in example 3

$$\underbrace{(\overline{m}_{13}[\overline{m}_{30}])^2 \cdot \overline{m}_{12}\overbrace{[(\overline{m}_{23}[\underline{m}_{30}])^2\{\underline{m}_{23}^2\}]}^{S_2(2)}}_{R_1(3)} \cdot \underbrace{(\overline{m}_{13}\overbrace{[\underline{m}_{30}]}^{S_3(1)})^2 \cdot \overline{m}_{12}\overbrace{[\{\underline{m}_{23}^2\}]}^{S_2(1)} \cdot \{\underline{m}_{13}^2 \cdot \underline{m}_{12}\}}_{S_1(2)}, \tag{2.13}$$

one can observe that string $S_1(3)$ contains recursive calls of chronologically young branching substructures, i.e. $S_1(2)$, and physiologically old branching substructures, i.e. $S_2(2)$, which should be performed before $S_1(3)$.

Theorem 2.1 *Branching substructure dynamics is governed by the following formulae [32]*

$$S_p(0) = s_p, \tag{2.14}$$
$$S_p(k) = R_p(k) \cdot S_p(k-1), \qquad 0 < k \leq \tau_p, \tag{2.15}$$
$$S_p(k) = T_p(k) \cdot S_{\mu_p}(k-\tau_p), \qquad k > \tau_p, \tag{2.16}$$

with

$$R_p(k) = \prod_{q \in \mathcal{Q}_p} \left\{ m_{pq}(k) \left[S_q(k-1) \right]^{b_{pq}} \right\}^{r_{pq}}, \tag{2.17}$$
$$T_p(k) = \prod_{l=k}^{k-\tau_p+1} R_p(l). \tag{2.18}$$

example 4: substructure instantiation of example 3 [32]

$$\begin{array}{ll}
S_p(0) = s_p, & 1 \leq p \leq 3, \\
S_3(k) = m_{30}, & \forall k \geq 1, \\
S_2(k) = (m_{23} S_3(k-1))^2 S_2(k-1), & 0 < k \leq 3, \\
S_2(k) = \prod_{l=k}^{k-2} (m_{23} S_3(l))^2 S_3(k-3), & \forall k > 3, \\
S_1(k) = (m_{13} S_3(k-1))^2 \cdot m_{12} S_2(k-1) \cdot S_1(k-1), & 0 < k \leq 5, \\
S_1(k) = \prod_{l=k}^{k-4} \left((m_{13} S_3(l))^2 \cdot m_{12} S_2(l) \right) \cdot S_2(k-5), & \forall k > 5
\end{array} \tag{2.19}$$

□

Definition 2.11 *Complexity definition*
The complexity of branching substructure is defined as times of branching substructure sticking.

Theorem 2.2 [1] *The complexity of branching substructure dynamics is at most linearly related to plant CA.*

[1] Observed by Yan et al. [129] and Kang et al. [63], presented in this thesis.

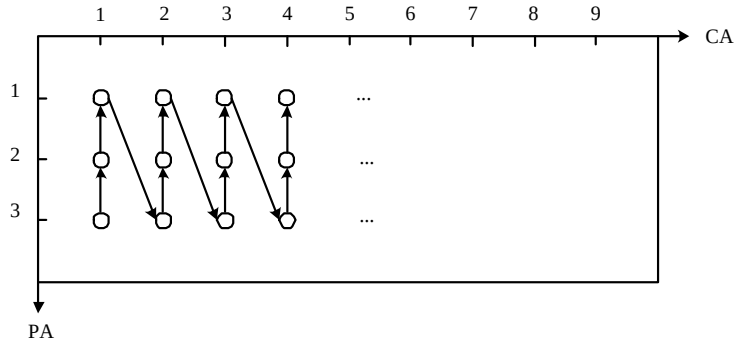


Figure 2.7: Computational sequence of substructure, where '$\circ$' signifies substructure at corresponding grid of PA and CA, and '$\rightarrow$' indicates the calculation sequence.

proof
For branching substructure dynamics, let $m_{pq}(k)$ be one application of a branching substructure. The term $[S_q(k-1)]^{b_{pq}}$ signifies b_{pq} attachments of $S_q(k-1)$. The increment of sticking times for branching substructure of PA p at its CA k is denoted $\omega_{p,k}$. Consider theorem 2.1, we have $\forall p \in \mathcal{P}$,

$$\omega_{p,k} = r_{pq}(1+b_{pq})+1, \qquad 0 < k \leq \tau_p, \tag{2.20}$$

$$\omega_{p,k} = \tau_p \cdot r_{pq}(1+b_{pq})+1, \qquad k > \tau_p. \tag{2.21}$$

(a) Suppose that all substructure $S_p(k)$ can be retrieved from substructure library $\mathcal{S} = \{S_p(k), 1 \leq p \leq P, 1 \leq k \leq N\}$. For whole individual plant, $p = 1$. Let $\beta_1 = \max_i \omega_{1,i}$, we have $\omega_k = \omega_{1,k} \leq \beta_1$.
(b) When there is no substructure library, one has to firstly set up the substructure library, the calculation sequence is thus from physiological-old and chronological-young substructure to physiological-young and chronological-old substructure (figure 2.7). Let $\beta^k = \max_p \omega_{p,k}$, then $\omega_k = \sum_{p=1}^{P} \omega_{p,k} \leq P \cdot \beta^k$. Let $\beta = \max_k \beta^k, 1 \leq k \leq N$, we have $\omega_k \leq P \cdot \beta$, that is to say, the substructure complexity is linearly proportional to number of physiological ages and number of chronological ages. Figure 2.8 shows the substructure computational graph for example 3.
Note that (1) substructure $S_P(k)$ of maximal PA P is the concatenation of metamer m_{P0}; (2) once $S_P(N)$ is calculated, the substructure library is constructed simultaneously. We can then perform more efficient simulation of case (a); and (3) substructure management is not considered (i.e. queries of substructures).

■

Remark 2.2 *(i) The plant computation usually concerns with counting processes or drawing procedures of organs. For the former, a number is associated with each substructure; for a organ drawing procedure, it corresponds to a copy of computer memory for macrostate or substructure. (ii) The efficiency of theorem 2.2 is obtained by applying substructure dynamics in a reverse manner with respect to Physiological Age, as in example 4.*

The substructure concept is sketched as a highly efficient computational algorithm that features mainly in that (i) *once and for all* calculations of substructure instances form a substructure library for both topological and geometric information; (ii) the strategy, to be *temporally economic at the cost of spatial storage*, is carried out *in a reverse manner* for

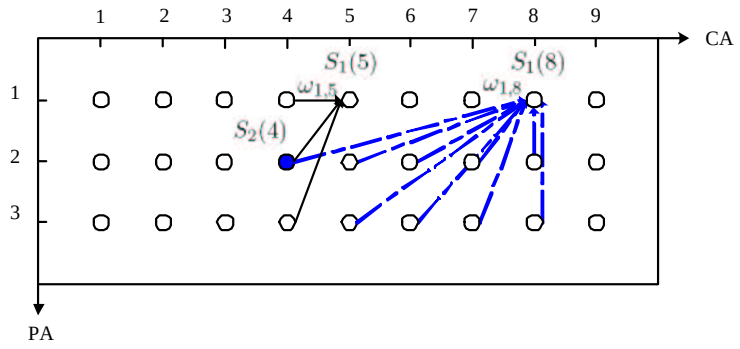


Figure 2.8: Computational graph of substructure $S_1(5)$ and $S_1(8)$ in example 3 with $\omega_{1,5}$ and $\omega_{1,8}$ as their incremental sticking times of substructures, where '$\circ$' signifies substructure at corresponding grid of PA and CA. Arrows '$\rightarrow$' and '$- \rightarrow$' indicate sticking of physiological-old and chronological-young substructures. Note that we just plot the computational graph for one GC, in fact the graph can be processed recursively, say physiological-old and chronological-young substructures can be linked to substructure $S_2(4)$ (marked as $\bullet$).

drawing and counting process, as is conceptually similar to that of dynamic programming and that of reverse mode of automatic differentiation. Numerical results for the substructure efficiency are shown in figure 2.9 employing AMAPsim software.

The disadvantage of substructure approach is the lost of flexibility. For example environment conditions are assumed to be optimal, and phylloclimate for metamers are ignored. When considering varying and heterogeneous environmental conditions within canopy, the two $S_2(4)$ (marked as '$\bullet$' in figure 2.8) called by $S_1(5)$ and $S_1(8)$ respectively are topologically identical but geometrically different, since two $S_2(4)$ undergo different environmental conditions. We thus have to consider environment differences for each substructure of PA p and of CA k for the construction of substructure library $\mathcal{S}$, consequently the size of the substructure library will be enormously increased. For efficient simulation, approximation methods for environment conditions have to be subtly designed, otherwise one can perform simulation metamer by metamer governed by growth rules.

Growth rules and substructure dynamics are essentially parallel. For example, (i) all the buds in one macrostate function in a parallel way for growth rules, (ii) the macrostate substructure at different CA $R_p(l)$ in theorem 2.1 can be performed parallelly. Thus further improvement of efficiency can be possibly achieved by ***parallel computing***.

Counting process of organ number [32]

Let $N_p^o(k)$ be the associated number for counting process of organs in branching substructure $S_p(k)$, according to theorem 2.1, we have

$$N_p^o(0) = 0, \tag{2.22}$$
$$N_p^o(k) = M_p^o(k) + N_p^o(k-1), \qquad 0 < k \leq \tau_p, \tag{2.23}$$
$$N_p^o(k) = L_p^o(k) + N_{\mu_p}^o(k-\tau_p), \qquad k > \tau_p, \tag{2.24}$$

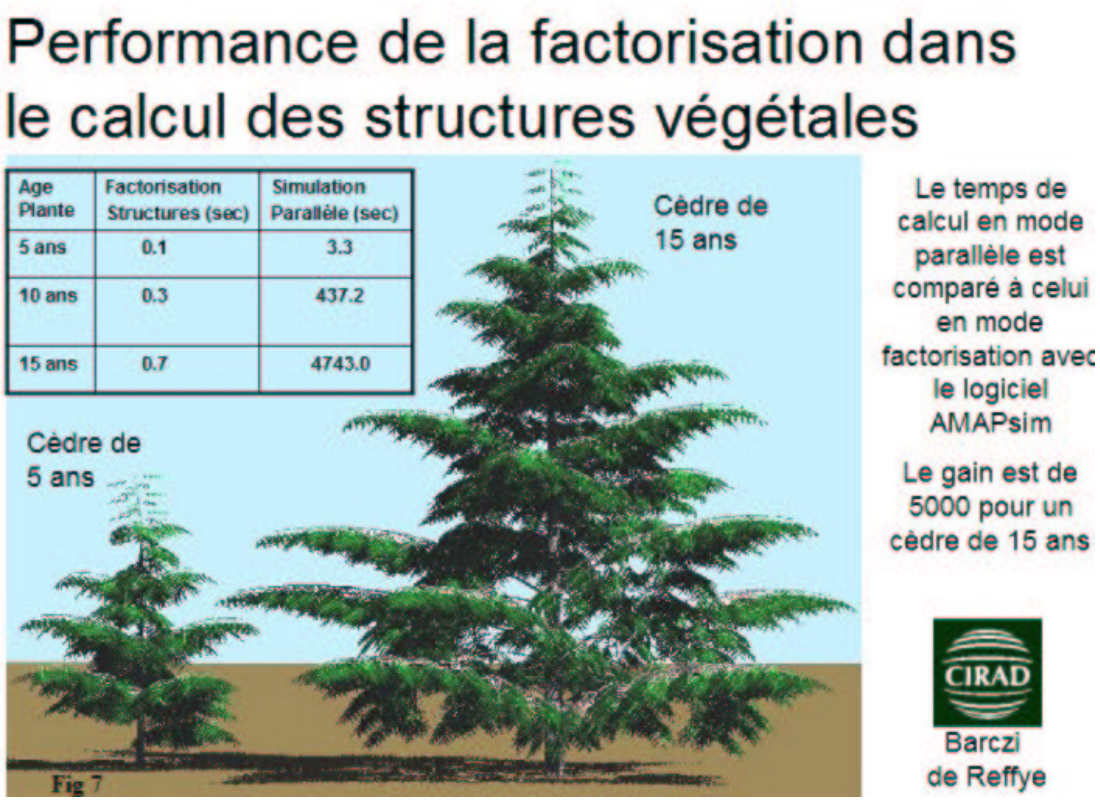


Figure 2.9: Complexity comparison between growth rules and substructure dynamics

with

$$M_p^o(k) = \sum_{q \in \mathcal{Q}_p} r_{pq} \cdot \{m_{pq}^o + b_{pq} \cdot N_q^o(k-1)\}, \tag{2.25}$$

$$L_p^o(k) = \sum_{l=k-\tau_p+1}^{k} M_p^o(l). \tag{2.26}$$

example 5: Counting process of leaf number for example 4 [32]
Since each metamer is attached by one leaf (see $\mathbf{M}_a$ in 2.9), we have

$$\begin{array}{lll} N_p^a(0) = 0, & & p = 1,2,3, \\ N_3^a(k) = 1, & & \forall k > 1, \\ N_2^a(k) = 2 + 2N_3^a(k-1) + N_2^a(k-1), & & 1 < k \leq 3 \\ N_2^a(k) = \sum_{l=k-2}^{k} \{2 + 2N_3^a(l)\} + N_3^a(k-3), & & \forall k > 3 \\ N_1^a(k) = 3 + 2N_3^a(k-1) + N_2^a(k-1) + N_1^a(k-1), & & 1 < k \leq 5 \\ N_1^a(k) = \sum_{l=k-4}^{k} \{3 + 2N_3^a(l) + N_2^a(l)\} + N_2^a(k-5), & & \forall k > 5 \end{array} \tag{2.27}$$

we have $N_2^a(4) = 3(2 + 2N_3^a(1)) + N_3^a(1) = 13, N_1^a(9) = 5(5 + N_2^a(4)) + N_2^a(4) = 109.$ □

Functioning: hydraulic approach

In section 2.1.2, we briefly introduce the basic hypothesis on *GreenLab* plant functioning. In this section we will present it in more details.

Problematic

Plant functioning can be roughly simplified as biomass production and allocation. Usually the physiological knowledge and concepts are originally achieved in process-based modelling or pure physiological research that deals with mechanical process at different levels from cells to individual leaves, then adapted (if necessary) and spread into the FSPM committee, since for architecture models there is only plant architecture in the beginning.

For process-based models, the objective is usually the canopy, and organs within the canopy are not distinguished. The canopy is considered as one big leaf with certain surface area (notion Leaf Area Index, LAI). Most of the process-based models are constructed on the base of LAI dynamics, i.e. STICS [16], EPIC [124], TOMGRO [7]. The biomass production can be calculated based on either light radiation or water transpiration. Both define the conversion efficiency to be *linear* for their simplest versions. For the former, the relation between accumulated biomass and intercepted radiation is defined as Radiation Use Efficiency (RUE [84]) that synthesizes the photosynthesis and respiration process. TOMGRO and EPIC mainly follow this approach. For the latter, Water Use Efficiency (WUE) is defined. STICS and CROPSYST [108] mainly follow this approach. As far as biomass allocation is concerned, Warren-Wilson [123] proposes to consider the plant as compartments of source and sink for the repartition of assimilation. Some models introduce the priorities among organs, and it is usually necessary to consider the organ expansion. TOMGRO follows this approach. An alternative proposed by Spaeth and Sinclair [107] is to extend the harvest index notion to the dynamic biomass accumulation in grains, as leads to less parameters. STICS follows this approach.

It is an open problem to model plant biomass production for FSPMs, whereas for biomass allocation, FSPM modelers tend to assign source-sink relationships for organs to pilot their competitions for produced biomass [104], [55].

Limited by our background of plant architecture, we urge three criteria for the assimilation of physiological knowledge. (i) The biomass production and allocation shall be performed at organ level; (ii) the biomass production formula shall be dynamic, rather than static curves of photosynthetic rate with respect to irradiance, temperature, and CO_2 concentration that omit time evolution [113]; (iii) the production formula shall provide accumulated biomass synchronized by the discrete events of sequent appearance of new organs, that is, by thermal time index of Growth Cycle.

Existing approaches of biomass production for FSPMs can also be summarized as two catalogues: light-related photosynthesis and water-related transpiration. For the former, the modelling follows either scale-up or scale-down strategy. By scale-up, firstly the photosynthesis of the elements, i.e. metamers, is formularized, then the effects of these basic elements are aggregated for an overall functioning of the canopy. We refer to RAPT [103] and LIGNUM [88]. By scale-down, the canopy photosynthesis formula for crop models is

adapted by assimilating 3D canopy structures [34].

Pure physiological research at leaf level, i.e. that in [113], shall be especially helpful to FSPM modelers. As Thornley states [113], photosynthesis depends on the past environment as well as current environment, thus immediate response of photosynthesis rate (environment conditions within minutes) and slower process of acclimation (environmental conditions over the past week or two) are distinguished. As a result, photosynthesis formulae have to balance their time step of simulation according to their proper objectives, i.e. focus on growth rate or growth accumulation. We have variety of time step from minutes (RAPT), or days, to years (LIGNUM).

The approach of water-related transpiration aims at the formulae for biomass production at somehow coarse time steps, nevertheless these formulae shall be relatively accurate for the *accumulated* biomass calculation. Although based on synthesis of biological processes, i.e. photosynthesis, acclimation, and biomass conversion, these formulae are essentially statistical, rather than mechanical. AMAPpara [29] follows this approach. It is notable that architectural modelers tend to either investigate physiologically simple process of transpiration based on plant hydraulic network at organ level (AMAPhydro [30]), or provide framework of bidirectional communication between architecture and environment without focus on physiological aspects [82].

The biomass allocation is termed as *irreducible* process of substrate transport from *sources* to *sinks* and chemical/biochemical conversions at the *sinks* [112]. The research mainly follows three approaches, featured by the controlling mechanisms, universal biological laws, and descriptive methods respectively.

By controlling mechanisms, we mean that the substrate transport is driven by transpiration and photosynthesis, and both transport and conversion are governed by a number of biological process, i.e. competition of meristems (sink strength), influence of local environment, and biological signals like hormone. The transport-resistance (TR) model is generally considered as mechanical [112]. In TR model formularization, carbon and nitrogen substrates enter plants through uptake process, and are transported into plant compartments via transport pathways driven by concentration difference of the substrates. The growth rates of the compartments consequently depends on the concentration of the substrates [111].

The universal biological laws aims at ubiquitous principles for living systems, i.e. plants, animals and microbes. Here we refer to two of them, allometry and teleonomy. The allometric relationship is defined as the dependence of a biological variable Y on body mass M by a scaling law [121], [122]

$$Y = Y_0 M^b, \tag{2.28}$$

where b is the scaling exponent and Y_0 a constant for specific kind of organism. The biological variables, for instance metabolic rates, heartbeat, or cross-section area of tree trunks, mostly satisfy the quarter-power scaling (arguable in [64]). In FSPM committee allometry relations characterize the biomass allocation of different plant parts (say parts i and j) in the form [104]

$$W_i = aW_j^b, \tag{2.29}$$

where W is the biomass of the compartment, and a, b empirical constants. The allometric relations of Growth Unit is given by [29]

$$\frac{h}{\sigma} = \alpha q^\beta, \tag{2.30}$$

where h, σ are the length and cross-section area of the GU respectively, and α, β empirical parameters. Whereas teleonomy refers to the assumption that plant growth undergoes some *goal* [112], [79], e.g. the functional shoot/root equilibrium for an optimal relative growth rate, the linear relationship between foliage quantity and cross-sectional area of woody structures (pipe model [102]), and the tree branch angle maximizing effective leaf area [58]. It is interesting to note that the allometric scaling law of West et al. [121] is derived from *quantitive analysis* under a *geometric assumption* of space-filling fractal-like branching network with size-invariant final branch and a *teleonomy restriction* that requires the energy for resource distribution to be minimized.

By description methods, we mean that the mechanistic knowledge is encapsulated into the description of observations. One excellent example is the experimentally observed source-sink relationships accounted by TR model [83]. Substrate flow depends on the source-sink relationships (synthesis of [104], [112])

$$\frac{dM_i}{dt} = \frac{dM_j}{dt} = \frac{\kappa}{d_{ij}(t)} \cdot S_i(t) \cdot P_j(t), \tag{2.31}$$

where i indicates the source compartment index, j the sink compartment index, M_i, M_j are the substrates of the two compartments, κ is the proportional factor, S_i is the source strength of $i-$compartment, P_j is the sink strength of $j-$compartment, and d_{ij} is the distance between the two compartments. The *sink strength* is defined as the potential growth rate of sink compartments [7], that is to say, the potential capacity to accumulate substrates. Similarly the *source strength* can be defined as potential capacity of source compartments to provide substrates. The biomass allocation is therefore regulated by source-sink relationships. The calibration of parameters of the source-sink models is obligatory due to their description feature.

Concerning the state-of-art of biomass allocation for FSPMs, Sievänen et al. state that [104]

> The distribution of growth that results from source-sink relationships, material transport and tree-level control mechanisms can be technically incorporated into FSMs in a fairly straightforward manner. However, it is clear that so far our biological understanding of the processes involved has not supported very well this level of modelling.

One can thus naturally have the impression that FSPM modelling in the functioning aspect is an active process, in which firstly a coarse framework is set up, then mechanisms can be added after we clearly understand them, such as hormonal control, sink strength mechanism, and nonlinear substrate fluid.

Approach of GreenLab

Before launching *GreenLab* formularization of biomass production and allocation, we try to briefly summarize some features of several FSPMs, and an abstract interface is proposed for the functioning aspect of FSPM.

- LIGNUM narrows itself in specific tree modelling (Scots pine, *Pinus sylvestris L.* as the case), and the element unit for the analysis of both metabolism and spatial structure is chosen to be *terminal buds*, and *tree segment* between two *branching points*. A tree segment consists of sapwood, heartwood, bark and foliage. There is no hierarchical organization of architecture units, but LIGNUM claims that it is possible to adopt AMAP model [60] for the enforcement of the architectural description. Annual photosynthesis for *unit mass of foliage* is calculated for each segment, and the total number of segments is derived by an *empirical function* of new buds number with respect to the weight of the parent segment. Pipe model is adapted to its element units for biomass allocation. Note that the LIGNUM scientists are physiologists, and their physiological background is relatively solid in FSPM committee. There is a natural *annual synchronization* of photosynthesis and organogenesis, however when LIGNUM intends to be generic, the synchronization might be a problem.

- RAPT features largely in its spatial cellular discretization of the tree canopy, and in the distribution of environment condition, i.e. light, within the discrete canopy. The photosynthesis is calculated by Farquhar model [38] considering the density of leaf and environment conditions in each 3D cell. There is no clear architecture dynamics, and the validation is at branch level.

- The concept of the Wageningen approach [34] is to couple the crop model with architectural model. The 3D architectural model provides geometry information of organs and environment condition (i.e. light level per leaf). The physiological model furnishes number of appearing flowering shoot, and biomass growth rate. The synchronization time step is set artificially and empirically to one week. Their functioning formulae originate from *crop modelling* considering canopy 3D features (communication with 3D architectural model), but not biomass production and allocation at *organ* level for individual plant.

- L-systems approach mainly concentrates on providing bidirectional communication between architecture and environment (see GroGra [65] on growth grammar of L-systems, and open L-systems for bilateral conceptual model [82]). Concrete and complete FSPMs based on L-systems thus call for further collaboration between L-systems modelers and physiologists, i.e. the attempt exemplified in [34].

- AMAPhydro follows the hydraulic approach [29]. Biomass production is calculated according to linear empirical relationships between leaf evaporation and substrate assimilation. It adopts pipe model for biomass allocation. Both biomass production and allocation is implemented at organ level. However, AMAPhydro considers little about fluctuations of environmental conditions.

GreenLab derives from AMAPhydro, and its strategies in the functioning aspect of FSPM modelling are mainly:

- The aim of *GreenLab* is *not to be mechanical*, but to provide a theoretical idealtype of individual plant characterized by its functional-structural dynamics.
- The long-term prediction of accumulated biomass (during each GC) is firstly established via hydraulic approach to provide a framework, then the short-term complications, i.e. daily photosynthesis and respiration, phylloclimate, and the substrate transport and resistance process, could be possibly incorporated thanks to the detailed analysis of these specific subsystems. Alternatively speaking, the formularization of biomass production and allocation for *GreenLab* is not fixed. An *interface*, which lists the functioning formulae (i.e. photosynthesis) and the corresponding input (i.e. phylloclimate conditions of light, temperature, and CO_2 concentration), is provided instead as the bridge to the future complexisation.
- Although the physiological aspect of *GreenLab* is simple, intuitive, even naive, it shall be *illustrative* for the analysis of the phenomenon based on simplified functional-structural dynamics. *GreenLab* shall not violate the common sense.

An abstract interface for functioning We emphasize that *GreenLab* does not intend to fix its biomass production formulae on the hydraulic approach, but to provide an implementation of a *functioning interface* that is defined *for the FSPM element unit* ω as some nonlinear function,

$$\frac{dq(\omega(t))}{dt} = \Phi_1\left(\Upsilon(\omega(t)), \mathbf{U}(t)\right), \qquad \omega \in \Omega_n \tag{2.32}$$

$$\triangle q = \Phi_2\left(\overline{\Upsilon}(\omega), \overline{\mathbf{U}}\right), \qquad \omega \in \Omega_n \tag{2.33}$$

in either a photosynthesis-like form Φ_1 of growth rate (2.32) or an accumulated one Φ_2 (2.33), where dq/dt is the biomass production rate, $\triangle q$ is the accumulated biomass production for a given period e.g. one GC in *GreenLab* , Υ is the property set associated with the element unit that can be a voxel cell or a single leaf, $\mathbf{U}$ is the environmental input for the element unit, such as phylloclimate conditions of light, temperature, and soil water content, $\overline{\Upsilon}$ and $\overline{\mathbf{U}}$ are average values during the given period for Υ and $\mathbf{U}$ respectively. The property set Υ contains typically the volume or surface area of the single leaf, certain density index for the 3D cell, and the geometric information of the element unit within canopy. The space Ω_n can be either the results of structure dynamics governed by growth rules at discrete time index n, or an occupation of the plant growth in the 3D cellular canopy. The plant growth system can be considered as a dynamic system with a dynamic structure (abbreviation DS^2 in [48]) of functional-structural characteristics. The total biomass production Q is thus the aggregation of these element units,

$$\frac{dQ}{dt} = \int_{\Omega_n} \Phi_1\left(\Upsilon(\omega(t)), \mathbf{U}(t)\right) \cdot T(\omega) \cdot d\omega, \tag{2.34}$$

$$\triangle Q = \int_{\Omega_n} \Phi_2\left(\overline{\Upsilon}(\omega), \overline{\mathbf{U}}\right) \cdot T(\omega) \cdot d\omega, \tag{2.35}$$

where $T(\omega)$ is some transform function, when $T(\omega) \equiv 1$, the total biomass is then the sum of biomass produced by each element unit. The produced biomass is then allocated into these element units as

$$\frac{dg(\omega(t))}{dt} = \Gamma_1(\Omega_n, \frac{dQ}{dt}), \tag{2.36}$$

$$\Delta g(\omega) = \Gamma_2(\Omega_n, \Delta Q), \tag{2.37}$$

where Γ_1, Γ_2 are nonlinear functions for biomass allocation, $\frac{dg(\omega(t))}{dt}$ the growth rate of element unit ω, and $\Delta g(\omega)$ the biomass distributed into element unit ω at the end of the given period indexed by n. Let Ω_A the parameter set for function pair $\{\Phi_1, \Gamma_1\}$ or the pair $\{\Phi_2, \Gamma_2\}$.

Proposition 2.1 *The function Φ_1 of growth rate in formula (2.32) and Φ_2 of accumulation in (2.33) can take the same analytical forms at the time interval $[t_0, t]$, if $\Phi_i, i = \{1, 2\}$ is the solution of the following integral equation*

$$\int_{t_0}^{t} \Phi_i \left(\Upsilon(t), \mathbf{U}(t)\right) dt = \Phi_i \left(\frac{\int_{t_0}^{t} \Upsilon(t)dt}{t - t_0}, \frac{\int_{t_0}^{t} \mathbf{U}(t)dt}{t - t_0} \right), \qquad i = 1, 2. \tag{2.38}$$

proof
For certain element unit ω, the term $\Upsilon(\omega(t))$ can be abbreviated as $\Upsilon(t)$, and similarly $q(\omega(t))$ as $q(t)$. Consider the time interval $[t_0, t]$, we have the averages

$$\overline{\Upsilon} = \frac{\int_{t_0}^{t} \Upsilon(t)dt}{t - t_0}, \qquad \overline{\mathbf{U}} = \frac{\int_{t_0}^{t} \mathbf{U}(t)dt}{t - t_0}. \tag{2.39}$$

Let $q(t) = q_2, q(t_0) = q_1$. Considering (2.32) we have

$$\triangle q = q_2 - q_1 = \int_{q_1}^{q_2} dq = \int_{t_0}^{t} \Phi_1 \left(\Upsilon(t), \mathbf{U}(t)\right) dt. \tag{2.40}$$

Considering (2.33) and (2.39) we have

$$\triangle q = \Phi_2 \left(\frac{\int_{t_0}^{t} \Upsilon(t)dt}{t - t_0}, \frac{\int_{t_0}^{t} \mathbf{U}(t)dt}{t - t_0} \right). \tag{2.41}$$

Associating (2.40) and (2.41), we can see that, function Φ_1 and Φ_2 take the same analytic forms, if $\Phi_i, i = \{1, 2\}$ is the solution of the following integral equation

$$\int_{t_0}^{t} \Phi_i \left(\Upsilon(t), \mathbf{U}(t)\right) dt = \Phi_i \left(\frac{\int_{t_0}^{t} \Upsilon(t)dt}{t - t_0}, \frac{\int_{t_0}^{t} \mathbf{U}(t)dt}{t - t_0} \right). \qquad i = 1, 2.$$

■

Remark 2.3 *The proposition 2.1 indicates that the induction of function form Φ_2 for accumulated biomass production from growth rate form Φ_1 is not straightforward, e.g. for parabola or exponential extinction curves Φ_1 and Φ_2 take different form. It is necessary to conduct physiological experiments for the expression form of Φ_2. However the knowledge of growth rate form Φ_1, i.e. the extinction effect of light in photosynthesis, is helpful for the design of Φ_2.*

Overview of *GreenLab* functioning *GreenLab* biomass production follows the approach of the water-related transpiration. The transpiration is driven by the difference of hydric potential, which marks water pressure, through the plant *organ-level* hydraulic network. Then the notion of Water Use Efficiency outlines the relation between fresh biomass production and plant transpiration *during one GC*. The produced biomass is conceptually assumed to be held in a central transient pool, and *flows* without delay into organs according to their *relative* sink strength. The organogenesis dynamics, i.e. number of organs at each GC, is formulated by growth rules or substructure concept. The time step synchronization between metabolism and structural dynamics is the Growth Cycle.

The functioning during one GC is assumed to be composed of three stages [29] shown in figure 2.10. Firstly at the beginning of the GC, apical and axillary buds sprout into new metamers, that is, growth rules are performed. Organs, such as internodes and leaves, undergo primary growth, and their expansion volumes are calculated according to reserved fresh biomass, organ relative sink strength, and organ expansion rates. Secondly during the GC, architecture remains and the fresh biomass is fabricated through transpiration. Finally at the end of the GC, organs, namely ring (layer) and fruit, develop similarly to that in primary growth, and the process is called secondary growth.

Biomass production The *GreenLab* biomass production formulae for the moment are accumulated ones. In this thesis, we try to propose such formulae for the illustration of functional-structural dynamics. Our strategy is firstly to be mathematically workable and clear, then to enrich the FSPM formularization with physiological knowledge. Physiological experiments are being conducted at Chinese Agriculture University (Beijing, China), and experiment data from Wageningen university (Netherland) are being analyzed for the calibration and formulae design.

Transpiration Water is taken from soil by roots and flows through the plant hydraulic network up to leaves where it is partially transpired via stomata to provide necessary energy fluxes for mechanical processes say photosynthesis. For plant hydraulic structures, we refer to [115], [131] for an Ohm-like analog about the water evaporated by the plant, the plant resistance, and the hydraulic potential. The Ohm-like law for *average* plant transpiration during GC n, denoted $I(n)$, is given by [30]

$$I(n) = \frac{\Psi(n)}{R(n)}, \tag{2.42}$$

where $R(n), \Psi(n)$ is the plant average resistance and hydraulic potential of the plant transpiration pathway from root via vessels and leaves to atmosphere at GC n.

Conversion Thanks to the notion of Water Use Efficiency [59] [15], whose value is denoted as η, the accumulated biomass of GC n is given by

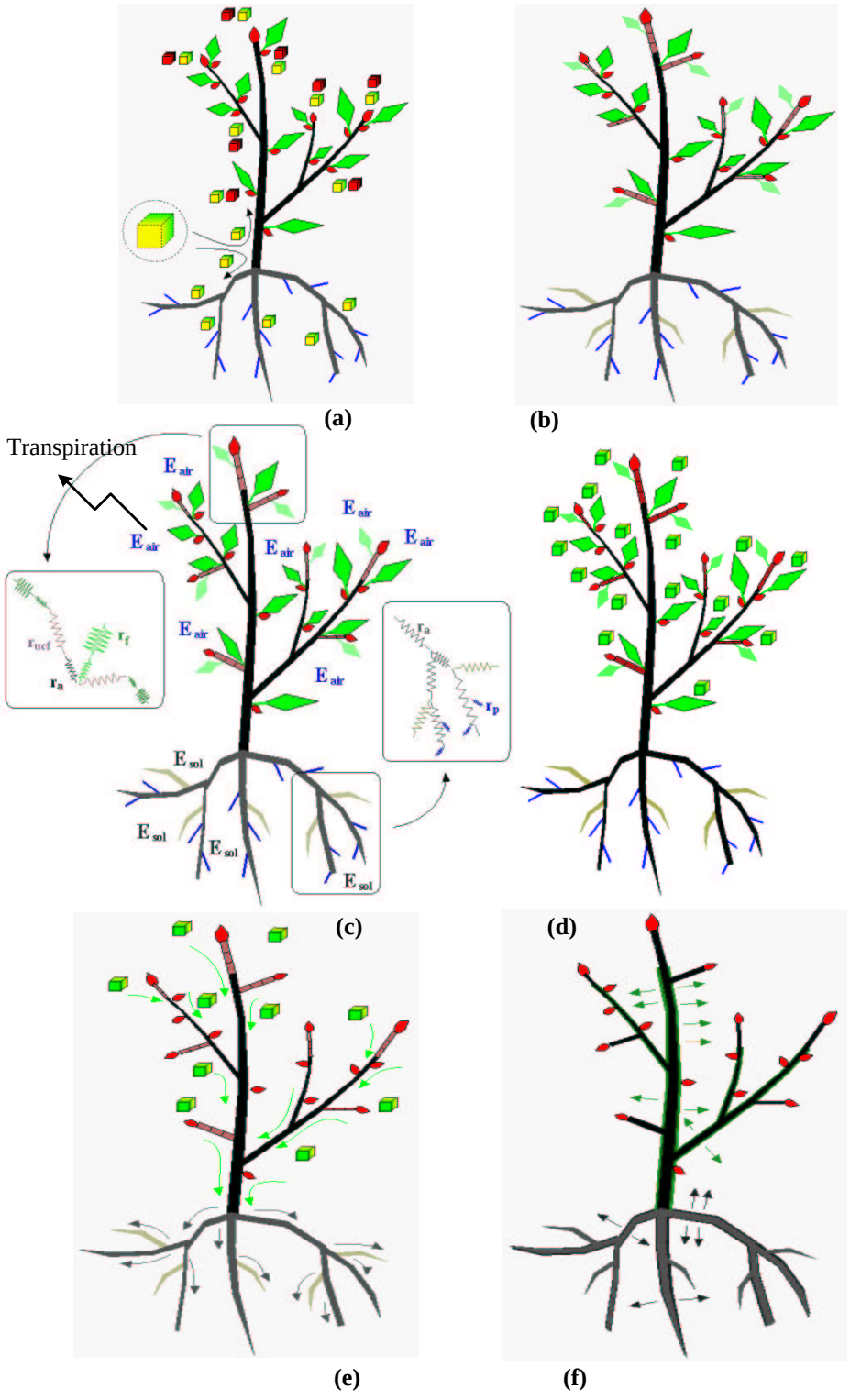


Figure 2.10: Functioning of one Growth Cycle: (a) allocation of reserved biomass into organs, i.e. leaves and internodes; (b) expansion of functioning organs in the primary growth; (c) plant transpiration via Ohm-like hydraulic network; (d) fresh biomass fabricated thanks to Water Use Efficiency notion; (e) allocation of reserved biomass into organs, i.e. fruits and rings; (f) secondary growth. Pictures from AMAPhydro [29]

$$Q(n) = \eta(n) \cdot I(n) = \eta(n) \cdot \frac{\Psi(n)}{R(n)}. \tag{2.43}$$

Let $E = \eta \cdot \Psi$ be the biomass production potential [30], we have

$$Q(n) = \frac{E(n)}{R(n)}. \tag{2.44}$$

Resistance Fourcaud et al. propose an assembly method to compute the hydraulic equilibrium, in which the woody hydraulic network is assumed to be formed by a set of hydraulic elements (such as leaves, root hairs, and sapwood) whose resistance is calculated mainly according to their geometric characteristics [40]. We list typical values for hydraulic resistance (in arbitrary units) [105],

Ground-root interface	5
Architecture (axes, leaves)	25
Leaf-atmosphere interface	800

Table 2.1: Typical resistance values of plant hydraulic pathways from root to atmosphere

We thus assume that the major hydraulic resistance lies in leaves, and the plant hydraulic network consists of *parallel vessels* from root hairs to leaves. The resistance of each leaf (indexed i) complies with Darcy's law [30],

$$R_i = \frac{\rho_i e_i}{S_i} + \frac{\rho_2 l_i}{s_i}, \tag{2.45}$$

where S_i is the surface area of $i-$th leaf, e_i the leaf thickness, l and s the length and cross-sectional area of the petiole, ρ_1, ρ_2 the hydraulic resistivity of the blade and the petiole respectively. We suppose a constant e for all leaves, and the ratio $\frac{l_i}{s_i}$ is fixed under allometric rules. The hydraulic resistance of a leaf can thus simplified as in [31], [105]

$$R_i = \frac{r_1}{S_i} + r_2, \tag{2.46}$$

where r_1 is the leaf blade resistance per unit area, and r_2 is the average resistance of the network that consists of the nerves and petiole of leaf (typically resistance values are about $r_1 = 800,000$ and $r_2 = 0.96$). The conductance C of N_a parallel leaves as a whole is

$$C = \sum_{i=1}^{N_a} \frac{1}{R_i} = \sum_{i=1}^{N_a} \frac{1}{\frac{r_1}{S_i} + r_2}. \tag{2.47}$$

Production It is well known that plant undergoes growth extinction when light conditions are intense enough, and a parabola-like response of growth to the variety of temperature conditions. We thus mimic accumulated plant growth during GC n under temperature and light environmental conditions as the following empirical formulae (for soil water conditions, we will formularize in the following sections),

$$E_i(n) = P_2 \left(\frac{T_i(n) - T_{\min}}{T_{\max} - T_{\min}} \right)^{\alpha - 1} \left(1 - \frac{T_i(n) - T_{\min}}{T_{\max} - T_{\min}} \right)^{\beta - 1} \cdot P_1 \left(1 - \exp(-kL_i(n)) \right), \tag{2.48}$$

$$q_i(n) = \frac{E_i(n)}{R_i(n)}, \tag{2.49}$$

where E_i, T_i, L_i are the biomass production potential, temperature, and light conditions for the $i-$th individual leaf, $T_{\max}, T_{\min}$ are the maximal and minimal temperatures that plant can bear, let $\Omega'_E = \{\alpha, \beta, P_1, P_2, k\}$ the set of environmental parameters, $q_i(n)$ is the biomass produced at GC n by $i-$th leaf.

Supposing that plant grows in homogeneous canopy for simplification, that is $\forall i, E_i = E, T_i = T, L_i = L$, plant bears total N_a functioning leaves, and that the transform function $T(\omega) \equiv 1$, we have the total biomass produced at GC n

$$E(n) = E_M \cdot \left(\frac{T(n) - T_{\min}}{T_{\max} - T_{\min}}\right)^{\alpha-1} \left(1 - \frac{T(n) - T_{\min}}{T_{\max} - T_{\min}}\right)^{\beta-1} \cdot (1 - \exp(-kL(n))), \tag{2.50}$$

$$Q(n) = \sum_{i=1}^{N_a} \frac{E(n)}{\frac{r_1}{S_i(n)} + r_2}, \tag{2.51}$$

where $E_M = P_2 \cdot P_2$, and let $\Omega_E = \{\alpha, \beta, E_M, k\}$ the parameter set for environment factors of light and temperature, $\Omega_H = \{r_1, r_2\}$ the parameter set for hydraulic structures. Note that in this version light and temperature factors are processed separately. The temperature response curve takes the form of beta law $p(x)$,

$$p(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}, \tag{2.52}$$

where $x = \frac{T - T_{\min}}{T_{\max} - T_{\min}}, 0 \le x \le 1$, and $B(\alpha, \beta) = \int_0^1 x^{\alpha-1}(1-x)^{\beta-1}dx$, thus $\int_0^1 p(x)dx = 1$, and when $x^* = \frac{\alpha-1}{\alpha+\beta-2}$, $p(x^*) = \max p(x)$. In (2.50), $B(\alpha, \beta)$ is aggregated into E_M. The parameter set Ω_E should be calibrated according to experiment data.

Remark 2.4 *For the environmental formulae (2.50) of biomass production potential, when temperature takes optimal value $x = x^*$ and L is big enough for plant to function in response of light extinction, E_M will be a constant value. In this case we thus call plant grows in optimal environmental conditions. The previous versions of GreenLab work under optimal environmental conditions, and E_M is empirically set to 1000.*

Biomass allocation in *GreenLab* is determined by the relative sink strength of organs. The biomass attraction (sink) of an $o-$type organ of CA k is defined as

$$d_p^o(k) = P_p^o \phi^o(k), \tag{2.53}$$

where P_p^o is the sink strength of $o-$type organ that is defined as the organ capacity to attract biomass, ϕ_p^o is the normalized distribution function (or expansion function) characterizing the evolution of the sink strength from CA 1 to CA t_p^o, t_p^o being the organ lifespan in GCs. We chose beta distribution in the discrete form (2.54)-(2.55) for ϕ_p^o which is shown in figure 2.11,

$$\phi_p^o(k) = \frac{1}{S}\left(\frac{k - 0.5}{t_p^o}\right)^{a_p^o - 1}\left(1 - \frac{k - 0.5}{t_p^o}\right)^{b_p^o - 1}, \tag{2.54}$$

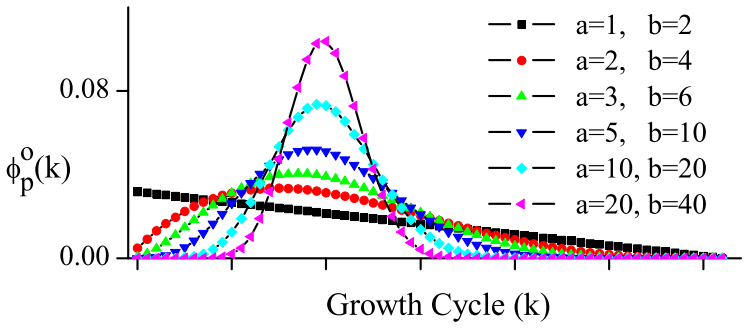


Figure 2.11: The curve cluster with different values of parameters a, b, t_p^o is set to 63 (total GCs of CAU sunflower).

$$S = \sum_{i=1}^{t_p^o} \left(\frac{k-0.5}{t_p^o}\right)^{a_p^o-1} \left(1-\frac{k-0.5}{t_p^o}\right)^{b_p^o-1}. \tag{2.55}$$

The total biomass attraction (or demand) $D(n)$ of plant at GC n is

$$D(n) = \sum_{o,p} \sum_{k=1}^{t_p^o} d_p^o(k) \cdot \Delta N_p^o(n-k), \tag{2.56}$$

where $\Delta N_p^o(n-k)$ is the number of $o-$type organs of PA p and CA k at GC n. Let $\Omega_D = \{P_p^o, a_p^o, b_p^o\}$ the parameter set for biomass distribution.

Recall the substructure counting formulae (2.22)-(2.26), $\Delta N_p^o(n-k)$ is then calculated as the difference of organ numbers between plant GC $n-k+1$ and $n-k$, that is,

$$\Delta N_p^o(n-k) = N_p^o(n-k+1) - N_p^o(n-k). \tag{2.57}$$

Note that in this presentation, primary growth and secondary growth are efficiently processed together. The calculation is thus physiologically false, but mathematically equal to that of separating two kinds of growths. In practice, the secondary organs in demand function (2.56) are those of older CA $k+1$ with respect to the primary organs of CA k.

The biomass increment $\Delta q_p^o(k,n)$ and total cumulated biomass $q_p^o(k,n)$ of $o-$type organ with CA k at current GC n is calculated according to relative sink strength of organs,

$$\Delta q_p^o(k,n) = \frac{d_p^o(k)}{D(n)} \cdot Q(n-1), \tag{2.58}$$

$$\begin{aligned} q_p^o(k,n) &= \sum_{j=1}^{k} \Delta q_p^o(j, n-(k-j)) \\ &= P_p^o \sum_{j=1}^{k} \frac{\phi_p^o(j) Q(n-(k-j)-1)}{D(n-(k-j))}. \end{aligned} \tag{2.59}$$

Recursive equations of biomass production In this section, we specify the biomass production formula (2.51). The plant maximal PA of leaves is usually 1, hence the PA

index p for leaves is omitted. The leaves are indexed by CA i and an integer j that identifies $j-$th leaf of all those with CA i. The biomass production potential of leaves is therefore denoted as $E_{ij}(n)$ when considering heterogeneous environment conditions. It is supposed that all the leaves appearing concurrently at GC $n-i$ with the same CA i have the same volume $q^a(i,n)$ at GC n, each leaf of PA p has a constant thickness e, and its leaf functioning timespan for photosynthesis is τ_p^a GCs. Let us denote leaf surface as $S(i,n)$. The biomass production formula is thus

$$\begin{aligned} Q(n) &= \sum_{i=1}^{n} \sum_{j=1}^{\Delta N^a(n-i)} \frac{E_{ij}(n)}{\frac{r_1}{S(i,n)}+r_2}, \qquad n \le \tau^a, \\ Q(n) &= \sum_{i=1}^{\tau^a} \sum_{j=1}^{\Delta N^a(n-i)} \frac{E_{ij}(n)}{\frac{r_1}{S(i,n)}+r_2}, \qquad n > \tau^a, \end{aligned} \tag{2.60}$$

$$S(i,n) = \frac{P^a}{e} \sum_{j=1}^{i} \frac{\phi^a(j) Q(n-(i-j)-1)}{D(n-(i-j))}. \tag{2.61}$$

Substitute (2.61) into (2.60), we have

$$\begin{aligned} Q(n) &= \sum_{i=1}^{\tau} \sum_{j=1}^{\Delta N^a(n-i)} \frac{E_{ij}(n)}{\frac{r_1}{S(i,n)}+r_2} \qquad (2.62) \\ &= \sum_{i=1}^{\tau} \sum_{j=1}^{\Delta N^a(n-i)} \frac{E_{ij}(n) S(i,n)}{r_1 + r_2 S(i,n)} \\ &= \sum_{i=1}^{\tau} \sum_{j=1}^{\Delta N^a(n-i)} \frac{E_{ij}(n) \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}{\frac{er_1}{P^a} + r_2 \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}, \qquad (2.63) \end{aligned}$$

where $\tau = n$ if $n \le \tau^a$, and $\tau = \tau^a$ if $n > \tau^a$. When the canopy is considered to be homogenous, $E_{ij}(n)$ is reduced to $E(n)$, and the recursive formula (2.63) becomes

$$Q(n) = \sum_{i=1}^{\tau} \frac{E(n) \Delta N^a(n-i) \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}{\frac{er_1}{P^a} + r_2 \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}. \tag{2.64}$$

Furthermore, if plant undergoes and optimal light and temperature conditions, $E(n)$ can be reduced to E_M, we thus have the recursive formula in previous version

$$Q(n) = E_M \sum_{i=1}^{\tau} \frac{\Delta N^a(n-i) \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}{\frac{er_1}{P^a} + r_2 \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}. \tag{2.65}$$

***GreenLab* implementation of the functioning interface (2.32)-(2.37)** The biomass production function Φ_2 is as (2.51) shows, and the biomass allocation function Γ_2 is as (2.58) shows. We have the model parameter set

$$\Omega_A = \Omega_E \cup \Omega_H \cup \Omega_D = \{\alpha, \beta, E_M, k, r_1, r_2, P_p^o, a_p^o, b_p^o\}. \tag{2.66}$$

Morphogenesis: geometrical dynamics

This section is reproduced from [32] to keep its mathematical beauty and to complete the formularization of *GreenLab* functional-structural dynamics. Primary work dates back to [26], [27].

The geometric information of the substructures is considered to be stored in a $4 \times N$ matrics of reals describing a set of N points of $\mathbb{R}^4$ (the projective space associated to $\mathbb{R}^3$). These points record the vertices of a triangulation of the plant shape. The set of these matrices is denoted $\mathcal{M}_4^*$. Some operations on $\mathcal{M}_4^*$ are introduced, such as the geometrical transformations represented by 4×4 matrices and the internal action of columns concatenation represented by the operator $\oplus$. The neutral element of $\oplus$ is denoted $\mathbb{O}$ and it means the empty 4×0 matrix (the matrix of four rows with zero columns).

Four transformations are defined according to botanic observations of plant geometry.

- The translation $\rho_{pq}(k, n)$ along $z-$axis in the local metamer coordinate of the length of the metamer $m_{pq}(k, n)$
- The rotation θ_p of the branching angle around the $x-$axis in the local metamer coordinates.
- The rotation ψ_p of the phyllotaxy angle around the $z-$axis in the local metamer coordinates.
- The rotation ω_b of $2\pi/b$ around the $z-$axis in the local metamer coordinates corresponding to the regular repartition of the buds when appear simultaneously b buds on an internode.

Omitting the index n in $S_p(k, n)$ which is common to all substructures, $S_p(k) \in \mathbb{M}_4^*$ follows the recurrent equations:

$$\begin{aligned} S_p(0) &= \mathbb{O}, && (2.67)\\ S_p(k) &= R_p(k, S_p(k-1)), \qquad 0 < k \le \tau_p, && (2.68)\\ S_p(k) &= T_p(k, S_{\mu_p}(k-\tau_p)), \qquad k > \tau_p, && (2.69) \end{aligned}$$

with

$$\Omega_{pq}(k, .) \;:\; X \in \mathcal{M}_4^* \mapsto m_{pq}(k)\psi_p \left\{ \bigoplus_{i=0}^{b_{pq}-1} \{\omega_{b_{pq}}^i \theta_p S_p(k-1)\} \oplus X \right\} \in \mathcal{M}_4^* \tag{2.70}$$

$$R_p(k, .) = \bigcirc_{q \in \mathcal{Q}_p} \Omega_{pq}^{\circ r_{pq}}, \tag{2.71}$$

$$T_p(k, .) = \bigcirc_{l=k}^{k-\tau_p+1} R_p(l, .) \tag{2.72}$$

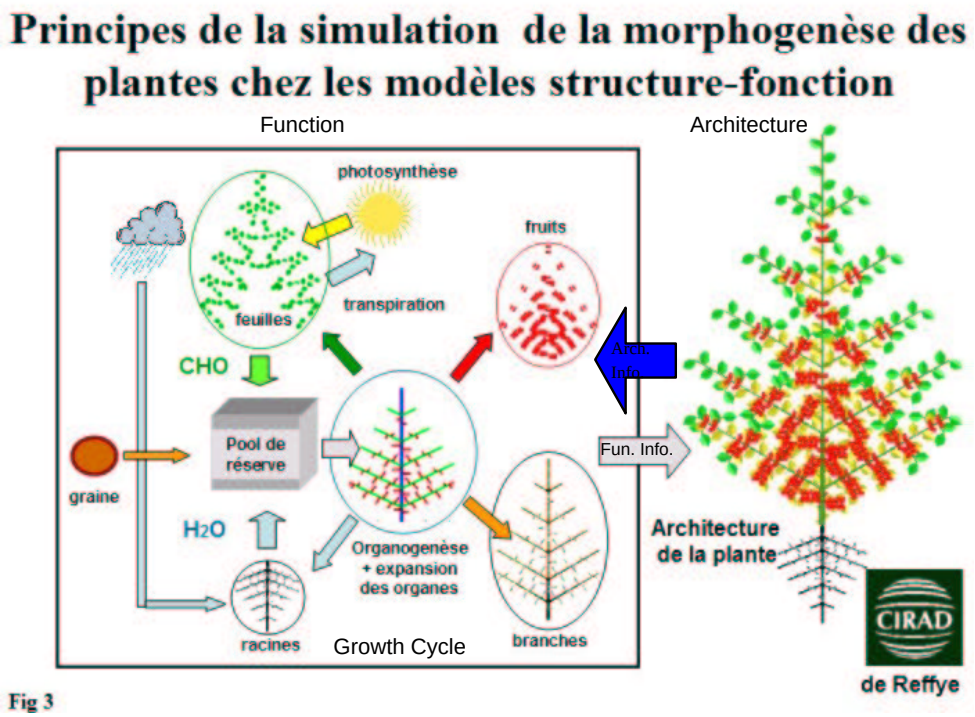


Figure 2.12: Plant functional-structural dynamics, interaction between functional information (fun. info.) and architectural information (arch. info.).

GreenLab functional-structural dynamics

In brief, *GreenLab* functional-structural dynamics can be illustrated as a quintuplet $H =< M, \mathbb{P}, \Lambda, \mathbb{F}, \Omega_A >$, where M is alphabet of plant string rewriting system, $\mathbb{P}$ is plant growth rule, Λ is set of configuration matrices in definition 2.1, $\mathbb{F} = \{\Phi, \Gamma\}$ is the function pair for biomass production and allocation, whose parameter set is denoted Ω_A. Environmental conditions $\mathbf{U}$ defined in the functioning interface can be considered as the input (control) of the dynamic system, whereas the property set Υ can be either model input or measurements.

The functional-structural dynamics provides an overall simulation of the plant phenotype development that results from the retroaction between functioning (biomass production and allocation) and structure (organogenesis). The influence of architecture information upon functioning, i.e. number of functioning leaves in the biomass production formula, is evident. In reverse, the feedback of functioning upon architecture is exemplified in that the number of organs depends on an empirical function with respect to a quantity of biomass production divided by demand $f\,(Q/D)$ [80] (figure 2.12, see also LIGNUM [88]).

Recalling the formularization (1.1) for variational methods in chapter 1, we rewrite the quintuplet H as a dynamic system in the discrete form,

$$\left\{ \begin{array}{lcl} \mathbf{X}(n+1) & = & F^d(\mathbf{X}(n), \Omega_A, \mathbf{U}(n)) \\ \mathbf{Y}(n) & = & M^d(\mathbf{X}(n)) \\ \mathbf{X}(1) & = & \Theta \end{array} \right. \tag{2.73}$$

associated with some objective index,

$$I = J^d(\Omega_A, \Theta, \mathbf{U}(1), \ldots, \mathbf{U}(n)). \tag{2.74}$$

Herein, $\mathbf{X}(n)$ is the state variable that signifies the fresh biomass accumulation of $\sum_{o,p} N_p^o(n)$ organs. The number of the incremental components of the state variable during GC n is $\sum_{o,p} \Delta N_p^o(n)$. The organ numbers are calculated by growth rule $\mathbb{P}$ or the formulae of structure dynamics (2.22)-(2.26) and that of (2.57). The observation vector $\mathbf{Y}$ associated with state variable $\mathbf{X}$ is chosen to be the *accumulated* organ volume. The objective index function J^d are problem-relevant. The rest notations of the dynamic system (2.73)-(2.74) are defined in previous sections.

When considering the feedback of functioning upon structure [80], *GreenLab* dynamic system (2.73) is *close-loop*, as possibly empowers the system less sensitive to initial conditions and model parameters.

Once we set up the dynamic system (2.73), variational methods can then be applied to solve the problem P1–P3 in Lions steps. We will formularize them in chapter 3.

2.2 Calibration of endogenous parameters

In previous versions of *GreenLab*, model calibration is partially implemented thanks to the down-to-earth research of Zhan *et al.* [134], [135]. The idea is that the perturbation of environment can induce a corresponding oscillation of the calibration results, however, it is expected that the values of the elements in endogenous parameter set $\Omega_G = \Omega_H \cup \Omega_D$ will not change, or change slightly under optimal environmental conditions, since endogenous factors are independent of environment. The calibration of genetic parameters is performed to minimize the discrepancy between model output and measurements at organ level with a constant $E = 1$ at each GC by the techniques of Generalized Least Square methods (GLS) and finite difference scheme. Note that we will discuss the variational formulism of calibration problem in chapter 3 benefiting from the functional-structural dynamics, and will conduct numerical experiments for the calibration of environmental parameters Ω_E in chapter 5.

In Zhan *et al.*'s work, the parameters are classified into two catalogues: the *direct* ones that can be estimated directly from observations and the *hidden* ones that can only be accessed from measurements by inverse methods. Typical parameters for the former are topological organizations (e.g. the configuration Λ) and allometric properties (i.e. blade thickness, scaling coefficients for petiole and that for internode). The latter for the moment includes hydraulic resistance Ω_H and source-sink relationships Ω_D.

Plant grows in terms of spatio-temporal occupation. Consequently experimental measurements or observations are spatio-temporal samples of the plant growth. Usually it is the case that we are lack of observations, because growth information always leaks under specific sampling strategies at the discrete time point and over certain parts within canopy. For *GreenLab* the sampling is carried out upon axis (spatial organization of PA) one time every one or several GCs (temporal organization of CA). The resulting measurements in-

cluding length, diameter, area, and fresh mass of each sampled organ, are aggregated into the observation set (see [135])

$$\mathcal{M} = \{y^{obs}_{i,j,k}\}, \tag{2.75}$$

where i is $i-$th axis of total l axes, j the $j-$th growth unit (rank) along the axis counted from the plant stump upwards, k the index of measurement types (e.g. 0 for internode diameter, 1 for internode length, 2 for internode mass, 3 for petiole diameter etc.). Supposing the total ranks and measurement types of $i-$th axis be u_i and v_i respectively, the number of measurement items are therefore

$$|\mathcal{M}| = \sum_{i=1}^{l} u_i \cdot v_i. \tag{2.76}$$

The calibration problem can then be written as

$$\min_{\Omega_G} \sum_{m=1}^{|\mathcal{M}|} \left\| {}^{m}y_{i,j,k} - {}^{m}y^{obs}_{i,j,k} \right\|^2. \tag{2.77}$$

In the case of multi-fitting, the problem is

$$\min_{\Omega_G} \sum_{n=1}^{N} \sum_{m=1}^{|\mathcal{M}_n|} \left\| {}^{m}_{n}y_{i,j,k} - {}^{m}_{n}y^{obs}_{i,j,k} \right\|^2, \tag{2.78}$$

where $({}^{m}_{n}y_{i,j,k})$ is the *GreenLab* simulation result corresponding to $m-$th measurement item $({}^{m}_{n}y^{obs}_{i,j,k})$ for $n-$th GC, N the total number of GCs that we aim to fit. The calibration results for cotton are listed in table 2.2, multi-fitting results of cotton are shown in figure 2.13.

Ω_G	P^a	P^e	P^c	P^f	P^m	B_a
Cotton	1.0	0.10	0.30	–	–	0.37
Ω_G	B_e	B_f	B_m	r_1	r_2	
Cotton	0.25	–	–	31	0.30	

Table 2.2: Fitting results of endogenous parameters for cotton, [135], which are pruned to be "single-stem plant" if necessary in the experiments. There is only one physiological age, and the index p is omitted. Here $C_o = a^o + b^o$, $B_o = \frac{a^o}{C_o}$. The parameter C_o is fixed for calibration purpose and B_o is to be estimated. The notation '–' signifies that the corresponding parameter is not defined for the species.

2.3 Plant growth under soil water conditions

2.3.1 Soil water balance

[2] Plants participate in soil water circulation by transpiration. Water is taken from soil by roots and flows through the plant hydraulic network up to the leaves, where water is

[2]The soil moisture model in this section is due to Ph. de Reffye.

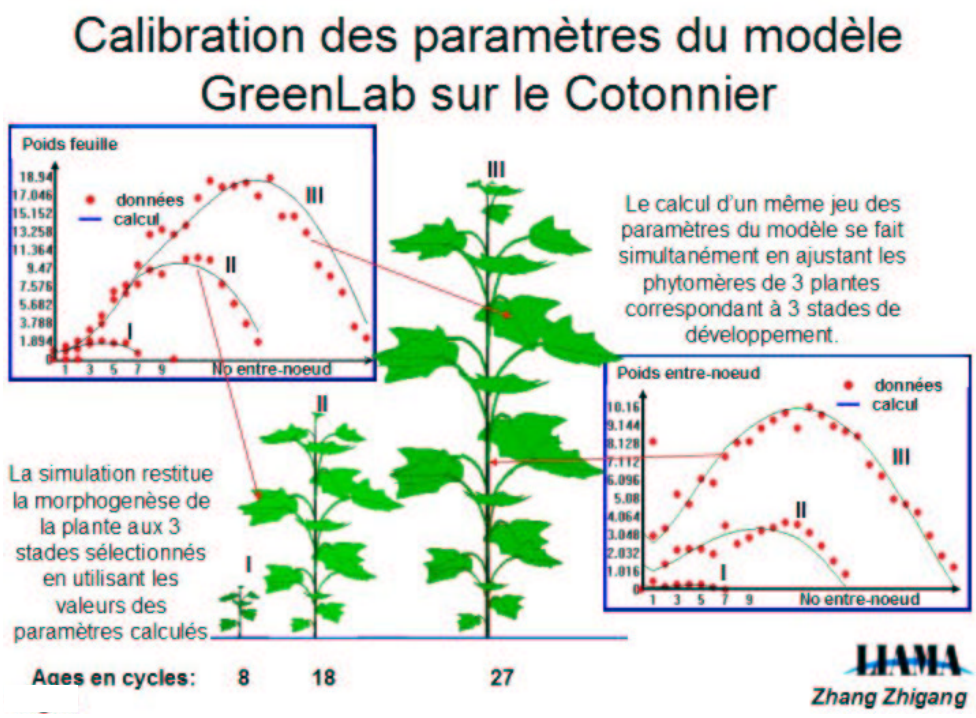


Figure 2.13: Multi-fitting results for cotton, N=3, GC=8,18,27 respectively

transpired to provide necessary energy fluxes for photosynthesis. The water content in the superior soil layers, named soil moisture, is important for the study of bio-geophysical processes in agricultural or forestry ecosystems. Soil water balance is achieved when we simplify this complex soil-plant system by concentrating on plant transpiration, soil evapotranspiration, and water supply from both irrigation and precipitations.

Suppose $Q_w(t)$ is the water content in soil per surface unit. It can be considered as a potential. The loss of water by evapotranspiration is

$$\frac{dQ_w(t)}{dt} = -c_1 \cdot (Q_w(t) - Q_{wmn}), \tag{2.79}$$

where Q_{wmn} corresponds to the wilt point of soil water content beneath which the plant cannot extract water from soil. c_1 is an evapotranspiration coefficient. Likewise, if $U(t)$ is the water supply at t, the water gained by the soil is

$$\frac{dQ_w(t)}{dt} = c_2(Q_{wmx} - Q_w(t))U(t), \tag{2.80}$$

where Q_{wmx} is the water field capacity above which the water flows away and c_2 is an absorption coefficient.

The differential equation for the evolution of the soil water content is thus

$$\frac{dQ_w(t)}{dt} = \{-c_1(Q_w(t) - Q_{wmn}) + c_2(Q_{wmx} - Q_w(t))U(t)\}. \tag{2.81}$$

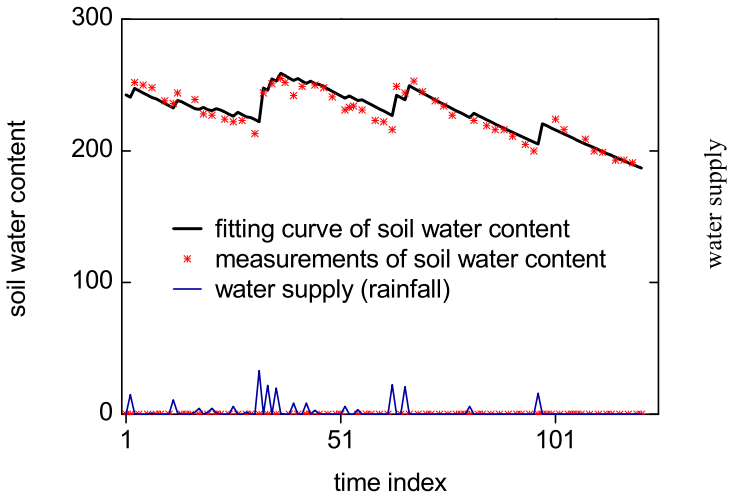


Figure 2.14: Fitting results of the calibrated soil moisture model, from Ph. de Reffye, published in [128]

Figure 2.14 shows the fitting results of the calibrated soil moisture model (2.81) using measurements of the soil water content and rainfalls done in Ivory Coast. Considering plant transpiration, Equation (2.81) becomes,

$$\frac{dQ_w(t)}{dt} = \underbrace{-c_1(Q_w(t) - Q_{wmn})}_{\text{soil evapotranspiration}} + \underbrace{c_2(Q_{wmx} - Q_w(t)) \cdot U(t)}_{\text{water absorption}} - \underbrace{PT(t)}_{\text{plant transpiration}}, \tag{2.82}$$

where $PT(t)$ is the plant transpiration and is linearly proportional to plant biomass production calculated by equation (2.51). The discrete form of equation (2.82) at GC scale is ($\Delta t = 1$ GC):

$$\begin{aligned} Q_w(n+1) = \; & (1 - c_1 - c_2 \cdot U(n))Q_w(n) + Q_{wmn}c_1 \\ & + Q_{wmx}c_2 \cdot U(n) - \rho \cdot Q(n), \end{aligned} \tag{2.83}$$

where $Q_w(n)$ is the soil water content at GC n, $U(n)$ is the water supply during GC n, $Q(n)$ is the plant biomass production during GC n and ρ is the ratio between plant transpiration and plant biomass production.

2.3.2 Soil-plant interaction

Recall the plant hydraulic formula (2.43), supposing that the hydraulic potential from root to atmosphere is linearly proportional to the soil water content near the root, the biomass production potential E is thus linearly proportional to the current soil water content:

$$E(n) = \kappa \cdot \frac{Q_w(n) - Q_{wmn}}{Q_{wmx} - Q_{wmn}}.$$

Consider the light, temperature and soil water conditions together, we have

$$E'(n) = E_M \cdot \frac{Q_w(n) - Q_{wmn}}{Q_{wmx} - Q_{wmn}} \cdot \left(\frac{T(n) - T_{\min}}{T_{\max} - T_{\min}}\right)^{\alpha-1} \left(1 - \frac{T(n) - T_{\min}}{T_{\max} - T_{\min}}\right)^{\beta-1} \cdot (1 - \exp(-kL(n)))\,. \tag{2.84}$$

Parameter E_M is the amplitude that combines all three factors (note that κ is assembled into E_M). The association of equations (2.51), (2.84), (2.59) and (2.83) provides the mathematical formalism of the soil-plant system that enables us to study interactions between plant and water resources in soil.

Rewrite the recursive equation (2.64):

$$Q(n) = E'(n) \sum_{i=1}^{\tau} \frac{\Delta N^a(n-i) \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}{\frac{er_1}{P^a} + r_2 \sum_{k=1}^{i} \frac{\phi^a(k) Q(n-(i-k)-1)}{D(n-(i-k))}}, \tag{2.85}$$

We implement equation (2.83), (2.84), (2.85), (2.58) and (2.59) in software *MiniGreenLab* that is compatible with *GreenLab* in that they share the same configuration files for model parameters, and provide the same simulation results on biomass production and allocation under optimal environmental conditions ($E(n) = E_M$).

2.4 Conclusion

In this chapter *GreenLab* structure dynamics is introduced, and the empirical environmental formulae is designed empirically for fresh biomass production potential, as provides a complete formulation of plant FSPM dynamics. The plant dynamic system can serve as the base for the applications of applied mathematics, i.e. the subject of this thesis – variational methods. We exemplify how submodels of environmental factors can be integrated into *GreenLab*, taking soil water balance as an example. Further developments are needed in the design of the biomass production formulae with the corporation of physiologists, and in the simulation coupling with phylloclimate models. The formulation is ready to be extended to the case of stochastic plant [73], [63].

Chapter 3

Variational methods for FSPM dynamics: theoretical formulation

Résumé

Ce chapitre est consacré à la formulation des méthodes variationnelles basées sur la dynamique de la structure fonctionnelle des plantes de *GreenLab* pour différentes applications, c'est à dire, identification de paramètre, assimilation de données, et contrôle optimal. Les équations adjointes sont introduites dans les systèmes d'optimalité pour le calcul du gradient. Nous montrons également la flexibilité du formulisme variationnel pour les mêmes applications quand une plus petite échelle de temps dans l'index thermique de temps, cycle de croissance (GC), est prise en considération. Les systèmes dynamiques de plantes peuvent être beaucoup plus compliqués, mais avec des méthodes variationnelles nous réussissons à formuler les équations adjointes pour les nouveaux systèmes d'optimalité dans les échelles raffinées avec de plus petits intervalles de temps.

3.1 Introduction

In this chapter, we formulate plant functional-structural dynamics as a continuous dynamic system that corresponds to *GreenLab* discrete system (2.73) in chapter 2.

$$\begin{cases} \frac{d\mathbf{X}}{dt} &= F(\mathbf{X}, \Omega_A, \mathbf{U}), \\ \mathbf{Y} &= M(\mathbf{X}), \\ \mathbf{X}(t_0) &= \Theta, \\ I &= J(\mathbf{X}, \mathbf{U}). \end{cases} \tag{3.1}$$

Remark 3.1 *GreenLab is originally a discrete model at GC time index. When state variable is chosen to be organ fresh biomass, one can hardly obtain a continuous form F along plant development from seeding to harvest, since the number of organs keeps changing. In chapter 3 the formulation based on continuous system 3.1 is for the purpose of illustration of variational methods and their applications. One might argue that the realistic plant dynamic system is rather complex, however, it does not change the formulation for concept illustration.*

We do succeed in defining a dynamic system in which the state variable dimension is fixed (chapter 6). In this case system 3.1 is assumed to be the corresponding continuous form

of GreenLab original discrete system, in the sense that given initial condition Θ and model input $\mathbf{U}$, if we have a unique GreenLab simulation according to the difference equation of the recursive formula (2.63), the system (3.1) can be supposed to have unique solution when additional conditions, e.g. the values of $\Theta, \mathbf{U}$, are provided.

If we consider plant growth within GCs governed by certain dynamical equation in either continuous or discrete form with organogenetic differentiation at GC index, the description for FSPM dynamics can be even more complex. The formulation for this case can be abstracted as dynamic system with dynamic structure (DS^2) [48].

Recalling system (1.1), we list the ingredients [70] defined for a variational method applied to FSPMs

- State variable $\mathbf{X}$ that describes plant growth status.
- Model $\frac{d\mathbf{X}}{dt} = F(\mathbf{X}, \Omega_A, \mathbf{U})$ that describes the plant functional-structural dynamics.
- Control $\mathbf{U}$ that can be initial condition Θ, model input (or control) of environmental light, temperature and soil water conditions. Here $\mathbf{U} \in \mathbf{U}_{ad}$, where $\mathbf{U}_{ad}$ denotes the admissible set of control.
- Objective function J that can be either a quantity that measures discrepancy between model output and observations or some index, e.g. fruit yield.
- Data or observations $\mathbf{Y}_{obs}$. When observation system M is an identity operator, we have directly the observation $\mathbf{X}_{obs}$ of the state variable $\mathbf{X}$.

The variational methods are the algorithms to solve the optimization problem

$$\min_{\mathbf{U} \in \mathbf{U}_{ad}} J(\mathbf{U}) \tag{3.2}$$

in the context of variational ingredients.

The cardinal problem is how to link together the model, data and control. The frequently asked questions about data and model are how to fix model parameters from the observations, that is, the calibration problem **P1** in Lions steps, and how to estimate the model uncertainties when assimilating the diverse information that are distributed locally in both time and space, that is, the problem of data assimilation **P2**. The investigation between model and control leads straightforward to the optimal control problem **P3**. More complicate problem can be proposed in the presence of model, data and control together, that is, the online process of data assimilation and optimal control.

The approach of Lions steps in meteorology has been pioneered by F.-X. Le Dimet [68], who introduced optimal control theory and started the constant progress of variational methods applications in environment sciences, i.e. meteorology, oceanography, and hydrology. It seems that the FSPM community will experience the same history as that in meteorology, with the model under development, the data under collection, and the request of estimation and control under implementation. The difficulties impeding the applications of variational method in FSPM are also similar to those in the early age of the attempts in meteorology, that is, the model is imperfect and keeps promoted in the direction of

extreme complex structures (e.g. simulation of the functioning of thousands of leaves), the collection of data is an onerous work and the observations at organ level are often sparse. Likewise, one might witness the similar milestones from theoretical prototypes [69] to operational applications [14] in FSPM community.

Variational problem (3.2) is a typical optimization problem. Up to now, significant advances have been achieved in optimization theory [45], [133], e.g. large-scale optimization. To solve problem (3.2) more efficiently, one can benefit from model dynamics for a reduction of the problem size thanks to the optimal control technique [89], [75]. Furthermore, the optimal control theory provides a unified formulism framework for the problem **P1-P3**.

The aim of the second part of the thesis is thus to introduce variational methods and concepts to FSPMs. In the first attempt, we follow the variational formulism of the theoretical prototype with standard procedures [69], then numerical experiments are conducted to illustrate the functioning of variational methods. Operational applications in FSPMs are expected in the near future.

The second part is organized as follows. Chapter 3 deals with mainly the detailed formulism of problem **P1-P3** using optimal control theory. The formulism is extended to the case within GCs to show its *adaptability*. FSPM dynamics within GCs is an implementation example of interface (2.32), (2.34), (2.36) proposed in chapter 2 in the form of growth rate. In practice, computational reduction can be achieved by automatic differentiation techniques that will be introduced in chapter 4. Numerical experiments of variational methods on model calibration, data assimilation and control are presented in chapter 5 and chapter 6.

3.2 Optimal control techniques for problem P1-P3

We follow the methodology proposed in [69], [72]. Before launching the formulation, we firstly recall some background knowledge.

Definition 3.1 ***Directional derivative or Gâteaux derivative***
Suppose X and Y are Banach spaces, $U \subseteq X$ is open, and $T : U \mapsto Y$, then T has a directional derivative (or Gâteaux derivative) $\hat{T}(u, \alpha)$ in direction α, if and only if

$$\lim_{t \to 0^+} \frac{T(u + t \cdot \alpha) - T(u)}{t}$$

exists.

Definition 3.2 *If $\hat{T}(u, \alpha)$ can be written as*

$$\hat{T}(u, \alpha) = \langle Z, \alpha \rangle_X, \forall \alpha \in X,$$

where $\langle \cdot, \cdot \rangle_X$ indicates norm of X, and Z depends on u, then Z is the gradient of T with respect to u, and is denoted by $\nabla_u T$.

Theorem 3.1 *If Gâteaux derivative $\hat{T}(u, \alpha)$ is linear and continuous with respect to α at u, X is a Hilbert space equipped with norm $\langle \ , \ \rangle_X$, then there exists a gradient of T at u.*

Definition 3.3 *([14])* ***Adjoint operator***
Given a linear operator $A : E \mapsto F$, and the norm $\langle , \rangle_E$ and $\langle , \rangle_F$ are defined in space E and F. The adjoint of A is the linear operator A^* such that for any vector (x, y) in the suitable spaces

$$\langle Ax, y\rangle_E = \langle x, A^* y\rangle_F .$$

When E and F are Euclidean spaces, the $\langle , \rangle_E$ and $\langle , \rangle_F$ signify the scalar products. In this case the adjoint operator takes the form of transposition of matrix, that is $A^* = A^T$. By Reize theorem, the adjoint operator always exists and it is unique, assuming spaces with finite dimension.

3.2.1 Parameter identification

Supposing that the model input $\mathbf{U}$ and initial condition Θ are known, let $\mathbf{K}$ the parameter vector for calibration, the model dynamics of (3.1) is thus

$$\begin{cases} \frac{d\mathbf{X}(t)}{dt} = F(\mathbf{X}(t), \mathbf{K}(t)), \\ \mathbf{X}(T_0) = \Theta, \end{cases} \tag{3.3}$$

where time-varying parameter $\mathbf{K}(t) \in \mathcal{K}$, $\mathcal{K}$ is the space of functions at time interval $[T_0, T_1]$ equipped with some norm $\langle\ ,\ \rangle_{\mathcal{K}}$ defined as

$$\langle u, v\rangle_{\mathcal{K}} = \int_{T_0}^{T_1} \langle u, v\rangle\, dt. \tag{3.4}$$

Here $\langle\ ,\ \rangle$ denotes scalar product. Let the observation system M be the identity operator, and the observation of state variable denoted $\mathbf{X}_{obs}$ that is distributed on interval $[T_0, T_1]$, the parameter Ω_A is calibrated by minimizing the discrepancy between observations and model calculations:

$$J(\mathbf{K}) = \frac{1}{2} \int_{T_0}^{T_1} \|\mathbf{X}(t) - \mathbf{X}_{obs}(t)\|^2\, dt. \tag{3.5}$$

Gradient calculation

Perturbating $\mathbf{K}$ in the direction of h, the model (3.3) and objective function (3.5) become the Tangent Linear Model (TLM)

$$\begin{cases} \frac{d\hat{\mathbf{X}}}{dt} = \frac{\partial F}{\partial \mathbf{X}}\hat{\mathbf{X}} + \frac{\partial F}{\partial \mathbf{K}}h, \\ \hat{\mathbf{X}}(T_0) = 0, \end{cases} \tag{3.6}$$

and the Gateaux derivative of J

$$\hat{J}(\mathbf{K}, h) = \int_{T_0}^{T_1} \left\langle \mathbf{X}(t) - \mathbf{X}_{obs}(t), \hat{\mathbf{X}}(t) \right\rangle dt, \tag{3.7}$$

where $\hat{\mathbf{X}}(t)$ is perturbation of $\mathbf{X}(t)$. After a scalar product by the adjoint variable P on (3.6) and an integral along $[T_0, T_1]$, it comes

$$\int_{T_0}^{T_1} \left\langle \frac{d\hat{\mathbf{X}}}{dt}, P \right\rangle dt = \int_{T_0}^{T_1} \left\langle \frac{\partial F}{\partial \mathbf{X}} \cdot \hat{\mathbf{X}}, P \right\rangle dt + \int_{T_0}^{T_1} \left\langle \frac{\partial F}{\partial \mathbf{K}} \cdot h, P \right\rangle dt. \tag{3.8}$$

Integrate (3.8) by parts and apply the definition 3.3 on adjoint operator, we have

$$\begin{aligned} \hat{\mathbf{X}}(T_1) \cdot P(T_1) - \hat{\mathbf{X}}(T_0) \cdot P(T_0) \quad &= \int_{T_0}^{T_1} \left\langle \hat{\mathbf{X}}, \tfrac{dP}{dt} + \left[\tfrac{\partial F}{\partial \mathbf{X}}\right]^T P \right\rangle dt \\ &\quad + \int_{T_0}^{T_1} \left\langle \left[\tfrac{\partial F}{\partial \mathbf{K}}\right]^T P, h \right\rangle dt. \end{aligned} \tag{3.9}$$

Note that $\hat{\mathbf{X}}(T_0) = 0$, by imposing

$$\begin{cases} \frac{dP}{dt} + \left[\frac{\partial F}{\partial \mathbf{X}}\right]^T P = \mathbf{X}(t) - \mathbf{X}_{obs}(t), \\ P(T_1) = 0, \end{cases} \tag{3.10}$$

the formula (3.9) becomes

$$\hat{J}(\mathbf{K}, h) = \int_{T_0}^{T_1} \left\langle -\left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P, h \right\rangle dt = \left\langle -\left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P, h \right\rangle_{\mathcal{K}}. \tag{3.11}$$

According to definition 3.2, Gâteaux derivative $\hat{J}(\mathbf{K}, h)$ in perturbation direction h equals to $\langle \nabla_{\mathbf{K}} J, h \rangle_{\mathcal{K}}$, thus gradient of J with respect to $\mathbf{K}$ is

$$\nabla_{\mathbf{K}} J = -\left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P. \tag{3.12}$$

Note that when $\mathbf{K}$ is time-invariant, Gâteaux derivative $\hat{J}(\mathbf{K}, h)$ in perturbation direction h becomes

$$\hat{J}(\mathbf{K}, h) = \int_{T_0}^{T_1} \left\langle -\left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P, h \right\rangle dt = \left\langle -\int_{T_0}^{T_1} \left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P dt, h \right\rangle. \tag{3.13}$$

In this case, gradient $\hat{J}(\mathbf{K}, h)$ equals to

$$\nabla_{\mathbf{K}} J = -\int_{T_0}^{T_1} \left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P dt. \tag{3.14}$$

3.2.2 Data assimilation

The data assimilation problem is to retrieve the state of the dynamics given diverse observations, or to predict the state of the dynamics at a given date in the future. Usually *time-dependant* or *site-dependant* parameters are assimilated for the better estimation or prediction. In the case of plant growth, these parameters can possibly be the unknown

initial condition Θ or mis-measured environmental conditions, such as light $L(n)$ and temperature $T(n)$ at GC n in (2.50).

The formulation of optimality system for data assimilation with respect to environmental conditions is similar to that of the parameters identification problem in the previous section. And we list the optimality system for data assimilation with respect to initial conditions as follows.

Supposing that the model parameter Ω_A is calibrated, we have,

Model:

$$\begin{cases} \frac{d\mathbf{X}(t)}{dt} = F(\mathbf{X}(t)), \\ \mathbf{X}(T_0) = \Theta. \end{cases} \tag{3.15}$$

Observation discrepancy:

$$J(\Theta) = \frac{1}{2} \int_{T_0}^{T_1} \|\mathbf{X}(t) - \mathbf{X}_{obs}(t)\|^2 \, dt. \tag{3.16}$$

TLM:

$$\begin{cases} \frac{d\hat{\mathbf{X}}(t)}{dt} = \frac{\partial F}{\partial \mathbf{X}} (\mathbf{X}(t)) \, \hat{\mathbf{X}}(t), \\ \hat{\mathbf{X}}(T_0) = v, \end{cases} \tag{3.17}$$

where v is the perturbation on initial condition Θ.

Adjoint model:

$$\begin{cases} \frac{dQ(t)}{dt} + \left[\frac{\partial F}{\partial \mathbf{X}}\right]^T Q = \mathbf{X}(t) - \mathbf{X}_{obs}(t), \\ Q(T_1) = 0. \end{cases} \tag{3.18}$$

The gradient of J with respect to Θ is

$$\nabla_\Theta J = -Q(T_0). \tag{3.19}$$

3.2.3 Optimal control

The optimal control problem for FSPM dynamics is to determine model input $\mathbf{U}$ by optimizing some objective index. In this case the parameter set Ω_A is known. Supposing that the uncertainties have been assimilated as in the previous section, the system is written on $[T_1, T_2]$ as

$$\begin{cases} \frac{d\mathbf{X}(t)}{dt} = F\left(\mathbf{X}(t), \mathbf{U}(t)\right), \\ \mathbf{X}(T_1) = \Theta, \end{cases} \tag{3.20}$$

where $\mathbf{U}(t) \in \mathcal{U}$, and $\mathcal{U}$ is the space of functions at time interval $[T_1, T_2]$ equipped with norm defined by (3.4) (denoted by $\langle \ , \ \rangle_{\mathcal{U}}$). The yield index can be chosen as

$$I = \int_{T_1}^{T_2} G\left(\mathbf{X}\left(t\right), \mathbf{U}\left(t\right)\right) dt \tag{3.21}$$

that calculates the yield of vegetable organs (2.59) or the total weight of the plant production.

Gradient calculation

We compute the Gâteaux derivative of I and model (3.20) in the direction of a perturbation u on U, it comes

$$\begin{cases} \frac{d\hat{\mathbf{X}}}{dt} = \frac{\partial F}{\partial \mathbf{X}} \cdot \hat{\mathbf{X}} + \frac{\partial F}{\partial \mathbf{U}} \cdot u, \\ \hat{\mathbf{X}}\left(T_1\right) = 0, \end{cases} \tag{3.22}$$

and

$$\begin{aligned} \hat{I}\left(\mathbf{U}, u\right) &= \int_{T_1}^{T_2} \left(\frac{\partial G}{\partial \mathbf{X}} \left(\mathbf{X}(t), U(t)\right) \cdot \hat{\mathbf{X}}(t) + \frac{\partial G}{\partial \mathbf{U}} \left(\mathbf{X}(t), \mathbf{U}(t)\right) \cdot u \right) \cdot dt \\ &= \int_{T_1}^{T_2} \left\{ \left\langle \left[\frac{\partial G}{\partial \mathbf{X}}\right]^T, \hat{\mathbf{X}} \right\rangle + \left\langle \left[\frac{\partial G}{\partial \mathbf{U}}\right]^T, u \right\rangle \right\} \cdot dt. \end{aligned} \tag{3.23}$$

After a scalar product with adjoint variable W on (3.22) and an integral on $[T_1, T_2]$, it comes

$$\int_{T_1}^{T_2} \left\langle \frac{d\hat{\mathbf{X}}}{dt}, W \right\rangle \cdot dt = \int_{T_1}^{T_2} \left\langle \frac{\partial F}{\partial \mathbf{X}} \cdot \hat{\mathbf{X}}, W \right\rangle dt + \int_{T_1}^{T_2} \left\langle \frac{\partial F}{\partial \mathbf{U}} \cdot u, W \right\rangle dt. \tag{3.24}$$

Integrating by parts, we have

$$\begin{aligned} \hat{\mathbf{X}}\left(T_2\right) \cdot W\left(T_2\right) - \hat{\mathbf{X}}\left(T_1\right) \cdot W\left(T_1\right) &= \int_{T_1}^{T_2} \left\langle \hat{\mathbf{X}}, \frac{dW}{dt} + \left[\frac{\partial F}{\partial \mathbf{X}}\right]^T W \right\rangle dt \\ &\quad + \int_{T_1}^{T_2} \left\langle \left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W, u \right\rangle dt. \end{aligned} \tag{3.25}$$

Note that $\hat{X}\left(T_1\right) = 0$, and by imposing

$$\begin{cases} \frac{dW}{dt} + \left[\frac{\partial F}{\partial \mathbf{X}}\right]^T W = \left[\frac{\partial G}{\partial \mathbf{X}}\right]^T, \\ W\left(T_2\right) = 0, \end{cases} \tag{3.26}$$

the formula (3.25) becomes

$$\int_{T_1}^{T_2} \left\langle \hat{\mathbf{X}}, \left[\frac{\partial G}{\partial \mathbf{X}}\right]^T \right\rangle dt = - \int_{T_1}^{T_2} \left\langle \left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W, u \right\rangle dt. \tag{3.27}$$

Substitute (3.27) into 3.23, we have

$$\begin{aligned}\hat{I}(\mathbf{U},u) &= \int_{T_1}^{T_2}\left\{\left\langle\left[\frac{\partial G}{\partial \mathbf{X}}\right]^T,\hat{\mathbf{X}}\right\rangle+\left\langle\left[\frac{\partial G}{\partial \mathbf{U}}\right]^T,u\right\rangle\right\}\cdot dt \\ &= \int_{T_1}^{T_2}\left\{-\left\langle\left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W,u\right\rangle+\left\langle\left[\frac{\partial G}{\partial \mathbf{U}}\right]^T,u\right\rangle\right\}dt \\ &= \left\langle\left(-\left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W+\left[\frac{\partial G}{\partial \mathbf{U}}\right]^T\right),u\right\rangle_{\mathcal{U}}.\end{aligned} \tag{3.28}$$

According to definition 3.2, gradient of I with respect to $\mathbf{U}$ is then

$$\nabla_{\mathbf{U}} I=-\left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W+\left[\frac{\partial G}{\partial \mathbf{U}}\right]^T. \tag{3.29}$$

3.2.4 Comparisons

We list the adjoint models for parameters identification, data assimilation and optimal control problems as follows

$$\begin{cases}\frac{dP}{dt}+\left[\frac{\partial F}{\partial \mathbf{X}}\right]^T P=\mathbf{X}(t)-\mathbf{X}_{obs}(t)\\ P(T_1)=0\end{cases}$$

$$\begin{cases}\frac{dQ(t)}{dt}+\left[\frac{\partial F}{\partial \mathbf{X}}\right]^T Q=\mathbf{X}(t)-\mathbf{X}_{obs}(t)\\ Q(T_1)=0\end{cases} \tag{3.30}$$

$$\begin{cases}\frac{dW}{dt}+\left[\frac{\partial F}{\partial \mathbf{X}}\right]^T W=\left[\frac{\partial G}{\partial \mathbf{X}}\right]^T\\ W(T_2)=0\end{cases}$$

It can be seen that these adjoint models differ only in the forcing terms that depend on the forms of cost functions, i.e. $\mathbf{X}(t)-\mathbf{X}_{obs}(t)$ and $\left[\frac{\partial G}{\partial \mathbf{X}}\right]^T$. The term $\left[\frac{\partial F}{\partial \mathbf{X}}\right]^T$ can be deduced either by hand or by automatic differentiation tools, as possibly leads to reusable work for the three problem **P1-P3**. We list their gradients formulae as follows:

$$\nabla_{\mathbf{K}} J=-\left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P$$

$$\nabla_{\Theta} J=-Q(T_0)$$

$$\nabla_{\mathbf{U}} I=-\left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W+\left[\frac{\partial G}{\partial U}\right]^T$$

The calculation of these formulae provides efficiently the gradient information for unconstrained optimization algorithms, i.e. the general Newton methods introduced in chapter 2.

3.3 Formulation within GCs

We have formulated several optimal control systems for FSPM dynamics at GCs. In this section, we will illustrate the adaptability of the variational formulism when physiological knowledge is taken into account within GCs. The concept of abstract interface introduced

in chapter 2 is employed, that is, we do not denote the functioning formulae explicitly, but suppose the functioning equation to be differentiable and close given initial or boundary conditions.

Suppose we have dates $T_o, T_1, \ldots, T_n$ at which the plant architecture evolves, namely the interval $[T_0, T_1]$ is the duration of growth cycle 1, and likewise $[T_{n-1}, T_n]$ the GC n. The plant FSPM dynamics is given by [1],

Within GC 1, $[T_0, T_1]$

$$\left\{ \begin{array}{lcl} \frac{d\mathbf{X}_1}{dt} & = & F_1(\mathbf{X}_1, \mathbf{K}_1),\ \mathbf{X}_1 \in R^{N_1}, N_1 = n_1, \\ \mathbf{X}_1(T_0) & = & \mathbf{X}_0, \end{array} \right. \tag{3.31}$$

where $\mathbf{X}_1$ is the state variable for the newly-born n_1 organs.

Within GC 2, $[T_1, T_2]$

$$\left\{ \begin{array}{l} \frac{d\mathbf{X}_2}{dt} = F_2(\mathbf{X}_2, \mathbf{K}_2),\ \mathbf{X}_2 \in R^{N_2}, N_2 = n_1 + n_2, \\ \mathbf{X}_2(T_1) = \left[\begin{array}{l} \mathbf{X}_1(T_1) \\ 0 \end{array} \right], \end{array} \right. \tag{3.32}$$

where $\mathbf{X}_2$ is the state variable for N_2 organs at GC 2, and there are n_2 newly-born organs.

Likewise within GC k, $[T_{k-1}, T_k]$

$$\left\{ \begin{array}{l} \frac{d\mathbf{X}_k}{dt} = F_k(\mathbf{X}_k, \mathbf{K}_k),\ \mathbf{X}_k \in R^{N_k}, N_k = \sum_{i=1}^{k} n_i, \\ \mathbf{X}_k(T_{k-1}) = \left[\begin{array}{l} \mathbf{X}_{k-1}(T_{k-1}) \\ 0 \end{array} \right], \end{array} \right. \tag{3.33}$$

where $\mathbf{X}_k$ is the state variable for N_k organs at GC k (there are n_k newly-born organs), and parameter $\mathbf{K}_k \in \mathcal{K}_k$, $\mathcal{K}_k$ is the space of functions at time interval $[T_{k-1}, T_k]$ equipped with norm defined by (3.4) denoted by $\langle\ ,\ \rangle_{\mathcal{K}_k}$. Note that the organ number N_k and n_k are calculated according to substructure dynamics (2.22)-(2.26).

3.3.1 Parameters identification

Let $\mathbf{K} = \{\mathbf{K}_1, \ldots, \mathbf{K}_n\}$ be the unknown parameters. On each interval $[T_{k-1}, T_k]$ we have observations $\tilde{\mathbf{X}}_k$. The problem is to identify the parameters $\mathbf{K}$ by minimizing the discrepancy between observations and model calculations:

$$J(\mathbf{K}) = \sum_{k=1}^{n} \frac{1}{2} \int_{T_{k-1}}^{T_k} \left\| \mathbf{X}_k(t) - \tilde{\mathbf{X}}_k(t) \right\|^2 dt. \tag{3.34}$$

Gradient calculation

Pertubating $\mathbf{K}$ in the direction of $h = [h_1, \ldots, h_n]$ on $\mathbf{K}_1, \ldots, \mathbf{K}_n$, the model (3.31)-(3.33) becomes

[1] F.-X. Le Dimet contributes greatly to the formulation in this section

$$\begin{cases} \frac{d\hat{\mathbf{X}}_k}{dt} = \frac{\partial F_k}{\partial \mathbf{X}_k}(\mathbf{X}_k, \mathbf{K}_k)\,\hat{\mathbf{X}}_k + \frac{\partial F_k}{\partial \mathbf{K}_k}(\mathbf{X}_k, \mathbf{K}_k)\cdot h_k, \\ \hat{\mathbf{X}}_k(T_{k-1}) = \begin{bmatrix} \hat{\mathbf{X}}_{k-1}(T_{k-1}) \\ 0 \end{bmatrix}, \end{cases} \tag{3.35}$$

and the Gâteaux derivative of J is

$$\hat{J}(\mathbf{K}, h) = \sum_{i=1}^{n} \int_{T_{k-1}}^{T_k} \left\langle \mathbf{X}_k(t) - \tilde{\mathbf{X}}_k(t), \hat{\mathbf{X}}(t) \right\rangle dt, \tag{3.36}$$

where $\hat{\mathbf{X}}_k$ is the perturbation of $\mathbf{X}_k$.

Denote $P_1, \ldots, P_n$ the adjoint variable. After the scalar product with adjoint variable P_k on (3.33) and the integral on $[T_{k-1}, T_k]$, it comes

$$\begin{aligned} \sum_{i=1}^{n} \int_{T_{k-1}}^{T_k} \left\langle \frac{d\hat{\mathbf{X}}_k}{dt}, P_k \right\rangle dt &= \sum_{i=1}^{n} \int_{T_{k-1}}^{T_k} \left\langle \frac{\partial F_k}{\partial \mathbf{X}_k} \cdot \hat{\mathbf{X}}_k, P_k \right\rangle dt \\ &\quad + \sum_{i=1}^{n} \int_{T_{k-1}}^{T_k} \left\langle \frac{\partial F_k}{\partial \mathbf{K}_k} \cdot h_k, P_k \right\rangle dt, \end{aligned} \tag{3.37}$$

where $<\ ,\ >$ is the scalar product in R^{N_k}. Integrate (3.37) by parts,

$$\begin{aligned} &\underbrace{\sum_{i=1}^{n} \left[\left\langle \hat{\mathbf{X}}_k(T_k), P_k(T_k) \right\rangle - \left\langle \hat{\mathbf{X}}_k(T_{k-1}), P_k(T_{k-1}) \right\rangle \right]}_{I_1} \\ &\underbrace{- \sum_{i=1}^{n} \int_{T_{k-1}}^{T_k} \left\langle \hat{\mathbf{X}}_k, \frac{dP_k}{dt} + \left[\frac{\partial F_k}{\partial \mathbf{X}_k} \right]^T \cdot P_k \right\rangle dt}_{I_2} \\ &\underbrace{= \sum_{i=1}^{n} \int_{T_{k-1}}^{T_k} \left\langle \left[\frac{\partial F_k}{\partial \mathbf{K}_k} \right]^T \cdot P_k, h_k \right\rangle dt}_{I_3}. \end{aligned} \tag{3.38}$$

For term I_1, considering the sum of k and $k-1$ item,

$$\begin{aligned} &\left\langle \hat{\mathbf{X}}_k(T_k), P_k(T_k) \right\rangle - \left\langle \hat{\mathbf{X}}_k(T_{k-1}), P_k(T_{k-1}) \right\rangle + \\ &\left\langle \hat{\mathbf{X}}_{k-1}(T_{k-1}), P_{k-1}(T_{k-1}) \right\rangle - \left\langle \hat{\mathbf{X}}_{k-1}(T_{k-2}), P_{k-1}(T_{k-2}) \right\rangle. \end{aligned} \tag{3.39}$$

Note that in (3.39),

$$\hat{\mathbf{X}}_k(T_{k-1}) = \begin{bmatrix} \hat{\mathbf{X}}_{k-1}(T_{k-1}) \\ 0 \end{bmatrix},$$

and considering Euclidean product

$$\left\langle \hat{\mathbf{X}}_{k-1}(T_{k-1}), P_{k-1}(T_{k-1}) \right\rangle = \sum_{i=1}^{N_{k-1}} \mathbf{X}_{k-1}^i(T_{k-1}) \cdot P_{k-1}(T_{k-1}), \tag{3.40}$$

$$\begin{aligned}\left\langle \hat{\mathbf{X}}_k\left(T_{k-1}\right), P_k\left(T_{k-1}\right)\right\rangle &= \sum_{i=1}^{N_k} \hat{\mathbf{X}}_k^i\left(T_{k-1}\right)\cdot P_k^i(T_{k-1}) \\ &= \sum_{i=1}^{N_{k-1}} \hat{\mathbf{X}}_k^i\left(T_{k-1}\right)\cdot P_k^i(T_{k-1}),\end{aligned} \tag{3.41}$$

where the superscript i of $\hat{\mathbf{X}}_k^i(T_{k-1})$ denotes the i-th component of vector $\hat{\mathbf{X}}_k(T_{k-1})$, imposing

$$P_k^i(T_{k-1}) = P_{k-1}^i(T_{k-1}), \qquad i = 1, 2, \ldots, N_{k-1},\ k = 1, \ldots, n-1, \tag{3.42}$$

the term I_1 becomes

$$\left\langle \hat{\mathbf{X}}_n\left(T_n\right), P_n\left(T_n\right)\right\rangle - \left\langle \hat{\mathbf{X}}_0\left(T_0\right), P_0\left(T_0\right)\right\rangle.$$

Note that $\hat{\mathbf{X}}_0(T_0) = 0$ and imposing

$$P_n(T_n) = 0, \tag{3.43}$$

the term I_1 becomes 0.

For term I_2 in (3.38), imposing

$$\frac{dP_k}{dt} + \left[\frac{\partial F_k}{\partial \mathbf{X}_k}\right]^T \cdot P_k = \mathbf{X}_k - \tilde{\mathbf{X}}_k, \qquad k = 1, \ldots, n, \tag{3.44}$$

and comparing with (3.36), we have $I_2 = \hat{J}(\mathbf{K}, h)$.

We rewrite term I_3 in vector form

$$I_3 = \sum_{k=1}^{n} \left\langle \left[\frac{\partial F_k}{\partial \mathbf{K}_k}\right]^T P_k, h_k \right\rangle_{\mathcal{K}_k}. \tag{3.45}$$

Equation (3.38) then becomes

$$\hat{J}(\mathbf{K}, h) = -I_3 = < \nabla_{\mathbf{K}} J, h >, \tag{3.46}$$

where

$$h = \begin{pmatrix} h_1 \\ \vdots \\ h_k \\ \vdots \\ h_n \end{pmatrix}.$$

The gradient of J with respect to $\mathbf{K}$ is therefore

$$\nabla_{\mathbf{K}} J = \begin{pmatrix} \left[\frac{\partial F_1}{\partial \mathbf{K}_1}\right]^T P_1 \\ \vdots \\ \left[\frac{\partial F_k}{\partial \mathbf{K}_k}\right]^T P_k \\ \vdots \\ \left[\frac{\partial F_n}{\partial \mathbf{K}_n}\right]^T P_n \end{pmatrix}. \tag{3.47}$$

The adjoint model of the optimal control system (3.31)-(3.34), are the association of equation (3.42)-(3.44).

3.3.2 Optimal control

The formulation of optimal control within GCs is similar to that of parameters identification in the previous section. We summarize the results as follows:

Model:

Within GC 1, i.e. the internal $[T_0, T_1]$

$$\begin{cases} \frac{d\mathbf{X}_1}{dt} = F_1(\mathbf{X}_1, \mathbf{U}_1), \qquad \mathbf{X}_1 \in R^{N_1}, N_1 = n_1, \\ \mathbf{X}_1(T_0) = \Theta. \end{cases} \tag{3.48}$$

Within GC 2 $[T_1, T_2]$

$$\begin{cases} \frac{d\mathbf{X}_2}{dt} = F_2(\mathbf{X}_2, \mathbf{U}_2), \qquad \mathbf{X}_2 \in R^{N_2}, N_2 = n_1 + n_2, \\ \mathbf{X}_2(T_1) = \begin{bmatrix} \mathbf{X}_1(T_1) \\ 0 \end{bmatrix}. \end{cases} \tag{3.49}$$

Within GC k $[T_{k-1}, T_k]$

$$\begin{cases} \frac{d\mathbf{X}_k}{dt} = F_k(\mathbf{X}_k, \mathbf{U}_k), \qquad \mathbf{X}_k \in R^{N_k}, N_k = \sum_{i=1}^{k} n_i, \\ \mathbf{X}_k(T_{k-1}) = \begin{bmatrix} \mathbf{X}_{k-1}(T_{k-1}) \\ 0 \end{bmatrix}. \end{cases} \tag{3.50}$$

Yield index:

$$I(\mathbf{U}) = \sum_{k=1}^{n} \int_{T_{k-1}}^{T_k} G_k(\mathbf{X}_k(t), \mathbf{U}_k(t))\, dt, \tag{3.51}$$

where $\mathbf{U} = [\mathbf{U}_1\ \mathbf{U}_2\ \ldots\ \mathbf{U}_n]^T$, and $\mathbf{U}_k \in \mathcal{U}_k$. Here $\mathcal{U}_k$ is the space of functions at time interval $[T_{k-1}, T_k]$ equipped with norm $\langle\ ,\ \rangle_{\mathcal{U}_k}$ defined by (3.4).

TLM of model (3.50)

$$\begin{cases} \frac{d\hat{\mathbf{X}}_k}{dt} = \frac{\partial F_k}{\partial \mathbf{X}_k}(\mathbf{X}_k, \mathbf{U}_k)\hat{\mathbf{X}}_k + \frac{\partial F_k}{\partial \mathbf{U}_k}(\mathbf{X}_k, \mathbf{U}_k) u_k, \\ \hat{\mathbf{X}}_k(T_{k-1}) = \begin{bmatrix} \hat{\mathbf{X}}_{k-1}(T_{k-1}) \\ 0 \end{bmatrix}, \end{cases} \tag{3.52}$$

where u_k is the perturbation of $\mathbf{U}_k$, $u = [u_1\ u_2\ \ldots\ u_n]^T$.

Adjoint model:

$$\begin{cases} \frac{dQ_k}{dt} + \left[\frac{\partial F_k}{\partial \mathbf{X}_k}\right]^T \cdot Q_k = \left[\frac{\partial G_k}{\partial \mathbf{X}_k}\right]^T, & k = 1, \ldots, n, \\ Q_k^i(T_{k-1}) = Q_{k-1}^i(T_{k-1}), & i = 1, 2, \ldots, N_{k-1},\ k = 1, \ldots, n-1, \\ Q_n(T_n) = 0. \end{cases} \tag{3.53}$$

The gradient of I with respect $\mathbf{U}$ is

$$\nabla I_{\mathbf{U}} = \begin{pmatrix} -\left[\frac{\partial F_1}{\partial \mathbf{U}_1}\right]^T Q_1 + \left[\frac{\partial G_1}{\partial \mathbf{U}_1}\right]^T \\ \vdots \\ -\left[\frac{\partial F_k}{\partial \mathbf{U}_k}\right]^T Q_k + \left[\frac{\partial G_k}{\partial \mathbf{U}_k}\right]^T \\ \vdots \\ -\left[\frac{\partial F_n}{\partial \mathbf{U}_n}\right]^T Q_n + \left[\frac{\partial G_n}{\partial \mathbf{U}_n}\right]^T \end{pmatrix}. \tag{3.54}$$

3.3.3 Identification of GC intervals

The dates $T_1, T_2, \ldots, T_n$ can either be set empirically, or obtained by solving the following optimization problem:

$$\begin{aligned} &\min_{T_1,\ldots,T_n} J = \sum_{k=1}^{n} \int_{T_{k-1}}^{T_k} \left\| \mathbf{X}(t) - \tilde{\mathbf{X}}(t) \right\|^2 dt \\ &s.t.\quad 0 \le T_k - T_{k-1} \le D_m, \\ &\qquad \int_{T_{k-1}}^{T_k} \theta(t)\, dt = T_{\text{sum}}, \end{aligned} \tag{3.55}$$

where $\mathbf{X}(t)$ satisfies the model dynamics, D_m is the maximal duration of GC, $\theta(t)$ the environmental temperature, and T_{sum} the sum of temperatures needed for the emergence of new organs during the GC.

3.4 Conclusion

In this chapter, we formulate several optimal control systems at and within GCs to illustrate the adaptability of variational formulism, however, no numerical experiments are presented for the case within GCs in this thesis due to the absence of the physiological knowledge that are suitable to be integrated into *GreenLab* within GCs.

Chapter 4

Variational methods for numerical purpose: automatic differentiation techniques

Résumé

Ce chapitre concerne les méthodes numériques de l'approche variationnelle. Nous appliquons les techniques de la différentiation automatiques (AD) pour simplifier la création du code adjoint. La théorie de la différentiation automatique est brièvement présentée. Le rapport entre AD et l'équation adjointe est également étudié. Ensuite, nous récapitulons les étapes d'écriture du code adjoint à la main, ligne par ligne, et présentons les expériences préliminaire pour l'écriture du code adjoint dans le cadre du projet *GreenLab*.

4.1 Introduction

From the variational problems formulation in chapter 3, one can observe that the model dynamics yields efficient gradient calculations. With model simulations and gradient information, these variational problems are solved by numerical optimization algorithms (see Newton methods in section A.3.1) in following chapters. However, *GreenLab* is a simulation program and the *state variable* – definition of X in system (3.1) – is not well recognized, as is different from the case in meteorology that state variables are clearly defined. The derivation of adjoint model (or adjoint code) is therefore difficult. In this thesis, we evaluate two solutions to this difficulty: one is to rewrite plant growth model in a systematic way (section 6.4) and then derive its adjoint model (adjoint computation approach in section 6.6.4); the other is to resort to Automatic Differentiation (AD) techniques, which will be introduced in great details in this chapter. Note that in section 4.2.4, we illustrate the close relationships between AD and adjoint models.

This chapter is arranged as follows. In section 4.1, we present a general introduction of AD. Section 4.2 is dedicated to its theoretical aspects. The differentiation rule (4.36) is derived in details. In section 4.3, we apply AD techniques line by line to *GreenLab* source code (implemented in MATLAB, and Scilab version also exists) based on differentiation rule. We mainly record the process and the experiences for AD coding by hand. Since

detailed formulation of *GreenLab* problem **P2-P3** will be presented afterwards in chapter 6, we will postpone the validation results of adjoint code for these problems in subsequent corresponding sections.

> AD is a set of techniques based on the mechanical application of the chain rule to obtain derivatives of a function given as a computer program. AD exploits the fact that every computer program, no matter how complicated, executes a sequence of elementary arithmetic operations such as additions or elementary functions. By applying the chain rule of derivative calculus repeatedly to these operations, derivatives of arbitrary order can be computed automatically, and accurate to working precision.
>
> Introductionary remarks about AD at autodiff.org

AD is different from *symbolic differentiation* and approximations by *finite differences*. For algebraically simple functions, it is natural to exploit symbolic differentiation to obtain explicit derivative expressions according to differentiation rules. However, for complex functions with large number of variables, the analytic expression for gradient or Hessian can be tedious and error-prone, or even impossible.

The gradient can also be approximated by finite differences of function values for suitable chosen intervals [46]. Suppose a program computing function $y = F(\mathbf{X}), F : R^n \to R$, the approximation of the directional derivative $F'(\mathbf{X}, d\mathbf{X})$ can be computed by forward difference

$$F'(\mathbf{X}, d\mathbf{X}) = \frac{F(\mathbf{X} + \varepsilon \cdot d\mathbf{X}) - F(\mathbf{X})}{\varepsilon}, \tag{4.1}$$

where $d\mathbf{X}$ is some normalized direction, ε is some very small positive number. If X is n-dimensional vector, the forward difference formula has to be evaluated for n times to obtain the gradient. Although carefully choose of difference formulae leads to less function evaluation times, the cost ratio between the function and its derivatives is proportional to n. Furthermore, it is a dilemma to find the right ε. Truncation error of Taylor series occurs when ε is not small enough. When ε tends to zero, consequently the round-off error of ε cannot be neglected because of the machine's floating-point format. For more accurate gradients, trade-off between rounding error and truncation error is needed in finding the best ε, and usually it takes numerous additional executions of F [90]. Nevertheless the calculated derivatives are just approximations.

AD is superior to symbolic differentiation and finite difference approximation thanks to its accuracy and efficiency [52]. Firstly, AD incurs no truncation errors at all. AD exploits systematic application of chain rule and elementary differential calculus, as is somehow like symbolic differentiation. However AD differs in that the chain rule is applied to actual numbers rather than algebraic expressions. There are no auxiliary symbolic variables, but a differentiation trace, which is usually derived from the original programs. Secondly, a key point is that the chain rule can be applied in various ways. One of those variants is the so-called reverse, bottom-up, backward or cotangent linear mode, which yields gradients at no more than five times the number of operations needed for the evaluation of the underlying scalar function. The reverse mode computes the sensitivity of the function output

with respect to intermediate variables and propagates from one statement to the previous statement according to the chain rule. In contrast, the so-called direct, top-down, forward or tangent linear mode computes sensitivity of intermediate variables with respect to input variables and propagates from one statement to the next according to the chain rule.

AD is used in the areas, such as sensitivity analysis, optimal design, optimal control, and inverse problem (refer to 823 entries of publication collections on AD theory and applicatioins at autodiff.org), where derivatives evaluation forms the bottleneck. For instance, in data assimilation [69] of geophysical flows such as atmosphere, ocean, rivers, aquifers, AD has been widely exploited to obtain the gradients of residual forms with respect to initial conditions and other unknown quantities. The residues usually take the form of sum squares between the observations and predictions in the geophysical flows. The calculation may involve millions of variables. It is almost impossible to employ gradients without AD for minimization algorithms.

Currently there are several software implementations of automatic differentiation. These software differ in both common and distinct features regarding supported language (Fortran, C, C++, MATLAB), derivative order, application mode of chain rule (forward or reverse), implementation techniques (source-to-source transformation or overloading). We refer to some of them, TAMC [49], ADIFOR [10], Odyssée [39] for Fortran, ADMAT [119] for MATLAB, ADIC [11] for C, and ADOLC [53] for C and C++.

4.2 Theoretical aspects

4.2.1 Origin

Automatic differentiation has a long history, arguably dating back to Newton and Leibniz in their applications of derivative calculations to numbers [54]. In 1970s and early 1980s, algorithmic transformation of programs to compute derivatives in reverse mode was realized by a number of researchers. In the pioneer work of Speelpenning (1980, surveyed in [53]), in order to avoid the explosion of algebraic expressions when deriving the symbolic differentiation of some functions F expressed as m dependent variables $\mathbf{Y}$ with respect to n independent variables $\mathbf{X}$, he investigated into an optimizing compiler to the source code that can be generated from the symbolic representation of the derivatives.

In his thesis he advocated the differentiation of evaluation algorithms rather than formulae, and made the striking observation that the gradient of a scalar function ($m = 1$) can always be obtained for no more than five times the evaluation of the function itself, and for m-vector function the row-wise computation of Jacobian matrix costs no more than $5 \times m$ times the effort of evaluating the vector function. The upbound is independent of the number of variables. When the dimension of the vector function m is larger than n, the Jacobian matrix can be obtained more cheaply column by column through propagating gradients forward.

Speelpenning example [52]

The function which Speelpenning evaluates is

```
x(n+1) = x(1)
for i = n+2, ..., 2n
  x(i) = x(i-n) * x(i-1)
end for
y = x(2n)
```

Figure 4.1: Function evaluation (also called direct code) of Speelpenning example

$$y = f(x) \equiv \prod_{i=1}^{n} x_i. \tag{4.2}$$

The $i-$th component of gradient of (4.2) is

$$\frac{\partial f}{\partial x_i} = \prod_{j \neq i} x_j. \tag{4.3}$$

For each component obtained by symbolic differentiation, $n-1$ multiplications are needed. It is almost the same calculation load as the function itself. Let us denote the work ratio ω between the gradient and the function as follows [52],

$$\omega = \frac{work\{f, \nabla f\}}{work\{f\}}, \tag{4.4}$$

therefore for symbolic differentiation $\omega \propto n$. For finite difference the calculation load can be even more time-consuming. At first sight it seems that the proportional complexity with respect to variable dimension n is inevitable.

Speelpenning starts from sequential programs rather than explicit expressions for the function evaluation, and then he extends the programs to evaluate the function and gradient simultaneously.

The sequential program (pseudocode) of f is as in figure 4.1
All the quantities are allocated in a single array x, with the first n elements `x(i)`, `i=1,..,n` as input or independent variables $\langle x_i \rangle_{i=1,\ldots,n}$, and with the subsequent elements `x(i)`, `i=n+1,..,2n` as intermediate variables

$$\langle x_{n+i} \rangle_{i=1,\ldots,n} = \prod_{j=1}^{i} x_j. \tag{4.5}$$

The last element `x(2n)` is then assigned to the output or dependent variable y.

It is natural to evaluate the gradients of intermediate quantities $\langle x_{n+i} \rangle_{i=1,\ldots,n}$ with respect to independent variables according to the chain rule in a forward manner as in figure 4.2. Note that these intermediate quantities $\langle x_{n+i} \rangle$ are calculated by smooth functions that possess gradients ∇x_{n+i} with respect to n independent variables. Each intermediate quantity x_i is associated with its corresponding gradient ∇x_i. And at the end we have the gradient of the function ∇x_{2n}. The scheme involves $n^2/2$ multiplications, and the work

```
x(n+1) = x(1)
∇x(n+1) = e(1)
for i = n+2, ..., 2n
  ∇x(i) = e(i-n) * x(i-1) + x(i-n) * ∇x(i-1)
  x(i) = x(i-n) * x(i-1)
end for
y = x(2n)
∇y = ∇x(2n)
```

Figure 4.2: Forward differentiation (also called tangent code) of Speelpenning example. Here e(i) is the cardinal unit vector of n dimension

ratio is therefore $\omega = n/2$.

It is Speelpenning that is the first to discover that calculation efficiency can be achieved by accompanying each variable $x_i, i = 1, \ldots, 2n$ with some scalar quantity, say $\bar{x}_i$, and propagating these scalar quantities in a reverse manner. He defined the scalar quantity $\bar{x}_i$ as follows

$$\bar{x}_i = \frac{\partial f}{\partial x_i}, \tag{4.6}$$

thus

$$\begin{aligned} \bar{x}_i = \frac{\partial f}{\partial x_i} = \frac{\partial\left(\prod_{j=1}^{n} x_j\right)}{\partial x_i} = \frac{\partial\left(\prod_{j=1}^{i-n} x_j \cdot \prod_{j=i-n+1}^{n} x_j\right)}{\partial x_i} \\ = \prod_{j=i-n+1}^{n} x_j, \qquad i = n+1, \ldots, 2n. \end{aligned} \tag{4.7}$$

From (4.7), we have the recursive equation for $\bar{x}_i, i = n+1, \ldots, 2n$,

$$\begin{aligned} \bar{x}_{i-1} = \prod_{j=i-n}^{n} x_j = x_{i-n} \cdot \prod_{j=i-n+1}^{n} x_j \\ = x_{i-n} \cdot \bar{x}_i, \qquad i = n+1, \ldots, 2n. \end{aligned} \tag{4.8}$$

Especially when $i = 2n$, $\bar{x}_{2n} = 1$.

For $i = 1, \ldots n$, considering (4.3), (4.6), we have

$$\bar{x}_i = \frac{\partial f}{\partial x_i} = \prod_{j=1}^{i-1} x_j \cdot \prod_{j=i+1}^{n} x_j = x_{n+i-1} \cdot \bar{x}_{n+i}. \tag{4.9}$$

The formulae (4.7) and (4.9) pilot the calculation of scalar quantities $\bar{x}_i, i = 1, \ldots, 2n$ with the initial condition $\bar{x}_{2n} = 1$ in the reverse manner, as is shown in figure 4.3. The calculation of function and gradient consists of $3n-3$ multiplications in the reverse scheme in figure 4.3, therefore the work ratio $\omega = 3$. In the following section, this reverse approach is generalized as automatic differentiation technique in the reverse mode.

```
x(n+1) = x(1)
for i = n+2, ..., 2n
  x(i) = x(i-n) * x(i-1)
end for
y = x(2n)

x̄(2n) = 1;
for i = 2n, ..., n+2
  x̄(i-1) = x̄(i) * x(i-n);
  x̄(i-n) = x̄(i) * x(i-1);
end for
x̄(1) = x̄(n+1);
∇y = ⟨x̄(i)⟩, i = 1, .., n
```

Figure 4.3: Reverse differentiation (also called adjoint code) of Speelpenning example

4.2.2 Differentiation algorithms

In this section, we formulate the differentiation rule in AD algorithms. For extremely details, please see [52], [54], [49], and [119]. We follows mainly the notations and the presentation in the monograph of Griewank [54]. Firstly the evaluation procedure [54] for the calculation of function in sequential programs is introduced, then the chain rules for differentiation is investigated following the results of Giering and Kaminski [49], finally we propose and prove the equivalence between the differentiation rule and the definition of adjoint variable in automatic differentiation theory.

Evaluation procedures

Considering function

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) : D \subset R^n \mapsto R^m \tag{4.10}$$

which is evaluated by sequential program procedure, we assume that all quantities v_i calculated during the function evaluation are recorded as a sequence as:

$$\left\{ \underbrace{v_{1-n}, \ldots, v_0}_{\mathbf{x}}, v_1, v_2, \ldots, v_{l-m}, \underbrace{v_{l-m+1}, \ldots, v_l}_{\mathbf{y}} \right\}. \tag{4.11}$$

We catalogue $\{v_i\}$ as independent variables, intermediate variables and dependent variable [119]:

$$\begin{array}{ll} \mathbf{x} \equiv \{v_{1-n}, ..., v_0\} & \text{independent variables,} \\ \mathbf{v}' \equiv \{v_1, v_2, ..., v_{l-m}\} & \text{intermediate variables,} \\ \mathbf{y} \equiv \{v_{l-m+1}, \ldots, v_l\} & \text{dependent variables.} \end{array} \tag{4.12}$$

Each quantity v_i of intermediate and dependent variables is calculated by program code of one line or an elemental function f_i with respect to its arguments $\{v_j\}$, in which $j \prec i$.

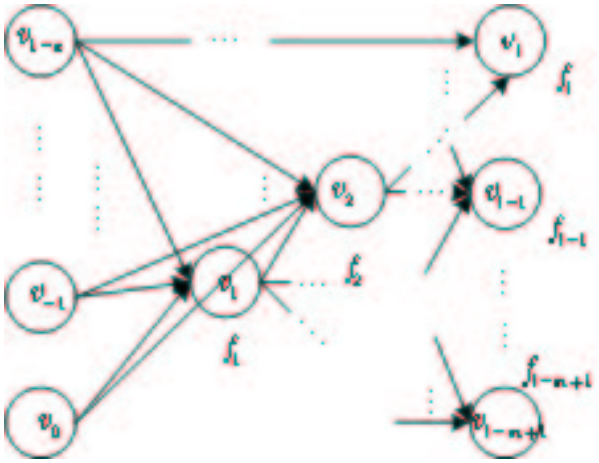


Figure 4.4: Computational Graph for evaluation procedure, similar to that of [54]

$\prec$ means the relation that v_i depends directly on v_j. When v_i is in the argument set, we have $i \prec i$. That is

$$v_i = f_i(\{v_j\}_{j \prec i}). \tag{4.13}$$

The evaluation procedure can also be illustrated as a computational graph in figure 4.4.

Chain rules

Let $\mathbf{v}^i = \mathbf{f}^i(\mathbf{v}^{i-1}), \mathbf{f}^i : R^{l+n} \mapsto R^{l+n}$, with $\mathbf{v}$ the vector of all the quantities $[v_{1-n}, \ldots, v_l]^T$. At each step i, v_i of $\mathbf{v}^i$ is updated according to (4.13) from all the $\{v_j\}_{j \prec i}$ of $\mathbf{v}^{i-1}$. The rest of the quantities of $\mathbf{v}^i$ remain the same as those of $\mathbf{v}^{i-1}$, that is:

$$\begin{aligned}
\mathbf{f}^1 : \quad & R^n \mapsto R^{n+l} \\
& \mathbf{x} \mapsto \mathbf{v}^1 \\
\mathbf{f}^i : \quad & R^{n+l} \mapsto R^{n+l}, \quad i = 2, \ldots, l-m \\
& \mathbf{v}^{i-1} \mapsto \mathbf{v}^i \\
\mathbf{f}^{l-m+1} : \quad & R^{n+l} \mapsto R^m \\
& \mathbf{v}^{l-m} \mapsto \mathbf{y}
\end{aligned} \tag{4.14}$$

The function (4.10) is then evaluated by the composition,

$$\mathbf{f} \equiv \mathbf{f}^{l-m+1} \circ \ldots \circ \mathbf{f}^1 \equiv \bigcirc_{i=1}^{l-m+1} \mathbf{f}^i. \tag{4.15}$$

Here the notation $\circ$ denotes function on the intermediate results at each step, and $\bigcirc_{i=1}^{l}$ is the composition of l steps. The Jacobian matrix of $\mathbf{f}$ is defined as

$$J_{ij}(\mathbf{x}_0) \equiv \left. \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_j} \right|_{\mathbf{x}=\mathbf{x}_0}, 1 \leq i \leq n, 1 \leq j \leq m \tag{4.16}$$

for i-th component $\mathbf{f}_i$ with respect to $j-$th variable $\mathbf{x}_j$, where $\mathbf{x}_0$ is some initial quantity. By applying chain rule, we have

$$J(\mathbf{x}_0) \equiv \left.\frac{\partial \mathbf{f}^{l-m+1}}{\partial \mathbf{v}^{l-m}}\right|_{\mathbf{v}^{l-m}=\bigcirc_{i=1}^{l-m}\mathbf{f}^i(\mathbf{v}_0)} \cdot \ldots \cdot \left.\frac{\partial \mathbf{f}^1}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_0}. \tag{4.17}$$

Precisely Jacobian of $\mathbf{f}^i, i = 2, \ldots, l-m$ with respect to $\mathbf{v}^{i-1}$ is

$$A_i \equiv \frac{\partial \mathbf{f}^i\left(\mathbf{v}^{i-1}\right)}{\partial \mathbf{v}^{i-1}} = \begin{pmatrix} 1 & 0 & \cdot & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 1 & \cdot & 0 & \cdot & \cdot & \cdot & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdot & 1 & \cdot & \cdot & \cdot & 0 \\ c_{i,1-n} & c_{i,2-n} & \cdot & c_{i,i-1} & c_{i,i} & \cdot & \cdot & 0 \\ 0 & 0 & \cdot & \cdot & \cdot & 1 & \cdot & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 \end{pmatrix}, \tag{4.18}$$

where

$$c_{i,j} = \frac{\partial f_i}{\partial v_j}, \qquad 1-n < j < l, 1 < i < l. \tag{4.19}$$

Denote $\mathbf{v}^0, \mathbf{v}^{l-m+1} \in R^{n+l}$ as vectors of all quantities whose corresponding components are $\mathbf{x}$, $\mathbf{y}$ respectively, and rewrite $\mathbf{f}^1, \mathbf{f}^{l-m+1}$ as $\mathbf{f}^{1,\prime} : \mathbf{v}^0 \mapsto \mathbf{v}^1, \mathbf{f}^{l-m+1,\prime} : \mathbf{v}^{l-m} \mapsto \mathbf{v}^{l-m+1}$, Jacobian of $\mathbf{f}^{1,\prime}$ and $\mathbf{f}^{l-m+1,\prime}$ then take the form of (4.18), that is,

$$\begin{aligned} A_1 &= \frac{\partial \mathbf{f}^{1,\prime}}{\partial \mathbf{v}^0}, \\ A_{l-m+1} &= \frac{\partial \mathbf{f}^{l-m+1,\prime}}{\partial \mathbf{v}^{l-m}}. \end{aligned} \tag{4.20}$$

Hence Jacobian of $\mathbf{f}^1, \mathbf{f}^{l-m+1}$ can be denoted by

$$\begin{aligned} \frac{\partial \mathbf{f}^1}{\partial \mathbf{x}} &= A_1 \cdot P_n^T, \\ \frac{\partial \mathbf{f}^{l-m+1}}{\partial \mathbf{v}^{l-m}} &= Q_m \cdot A_{l-m+1}, \end{aligned} \tag{4.21}$$

where

$$P_n \equiv [I, 0, \ldots, 0] \in R^{n\times(n+l)}, Q_m \equiv [0, \ldots, 0, I] \in R^{m\times(n+l)}. \tag{4.22}$$

There are at least two strategies for the evaluation of Jacobian $J(\mathbf{x}_0)$. By performing in the forward mode we start the matrix association from the right, i.e. firstly $\frac{\partial \mathbf{f}^2}{\partial \mathbf{v}^1} \cdot \frac{\partial \mathbf{f}^1}{\partial \mathbf{x}}$, then multiple $\frac{\partial \mathbf{f}^3}{\partial \mathbf{v}^2}$ at the left side of the results. All the intermediate results have n columns, because $\frac{\partial \mathbf{f}^1}{\partial \mathbf{x}}$ has n columns. In contrast, in the reverse mode we start from the association from the left, namely $\frac{\partial \mathbf{f}^{l-m+1}}{\partial \mathbf{v}^{l-m}} \cdot \frac{\partial \mathbf{f}^{l-m}}{\partial \mathbf{v}^{l-m-1}}$, and so on. The intermediate results in this case have m rows. When $n > m$, the reverse mode calculation costs less time, as we have addressed above in the introduction to the work of Speelpenning.

Differentiation in forward and reverse modes

Denote

$$\mathbf{v}_0^i = \mathbf{v}^i\big|_{\mathbf{x}_0} = \mathbf{f}^1(\mathbf{x}) \circ \bigcirc_{j=2}^{i} \mathbf{f}^j(\mathbf{v}^{j-1})\Big|_{\mathbf{x}=\mathbf{x}_0}. \tag{4.23}$$

Define the **adjoint variable** of an intermediate variable as sensitivity of $\mathbf{f}_k, 1 \le k \le m$ (k-th component of $\mathbf{f}$) with respect to this intermediate variable itself similar to (4.6) as

$$\bar{\mathbf{v}}^i = \nabla_{\mathbf{v}^i}\left(\bigcirc_{j=i+1}^{l-m} \mathbf{f}^j(\mathbf{v}^{j-1}) \circ \mathbf{f}_k^{l-m+1}(\mathbf{v}^{l-m})\right)\Big|_{\mathbf{v}_0^i} = \nabla_{\mathbf{v}^i}\mathbf{f}_k|_{\mathbf{v}_0^i}, \tag{4.24}$$

when $i = 0$, we have the value $\bar{\mathbf{x}}_0$ for adjoint variable $\bar{\mathbf{x}}$ with respect to $\mathbf{x}_0$,

$$\bar{\mathbf{x}}_0 = \nabla_{\mathbf{x}}\mathbf{f}_k|_{\mathbf{x}=\mathbf{x}_0}, \tag{4.25}$$

when $i = l - m + 1$, we have the value $\mathbf{y}_{k,0}^{-}$ for adjoint variable $\bar{\mathbf{y}_k}$ with respect to $\mathbf{x}_0$,

$$\bar{\mathbf{y}}_{k,0} = \nabla_{\mathbf{y}_k}\mathbf{f}_k|_{\mathbf{x}=\mathbf{x}_0} = 1. \tag{4.26}$$

Let $\dot{\mathbf{v}}^i$ be the perturbation on intermediate variable $\mathbf{v}^i$, by definition 3.2, we have

$$\hat{\mathbf{f}}_k(\mathbf{v}^i, \dot{\mathbf{v}}^i) = \left\langle \bar{\mathbf{v}}^i, \dot{\mathbf{v}}^i \right\rangle. \tag{4.27}$$

By chain rule, perturbation $\dot{\mathbf{v}}^i$ depends on previous perturbation $\dot{\mathbf{v}}^{i-1}$,

$$\begin{aligned} \dot{\mathbf{v}}^i &= \frac{\partial \mathbf{f}^i}{\partial \mathbf{v}^{i-1}}\bigg|_{\mathbf{v}^{i-1}=\mathbf{v}_0^{i-1}} \dot{\mathbf{v}}^{i-1} \\ &= A_i|_{\mathbf{v}_0^{i-1}} \cdot \dot{\mathbf{v}}^{i-1}. \end{aligned} \tag{4.28}$$

With A_i equal to (4.18), we have the differentiation formulae in forward mode

$$\begin{aligned} \dot{\mathbf{v}}_i^i &= \sum_{j=1}^{|I_i|} \dot{\mathbf{v}}_j^{i-1} \cdot c_{i,j}, \quad j \in I_i, \\ \dot{\mathbf{v}}_k^i &= \dot{\mathbf{v}}_j^{i-1}, \qquad\qquad k \ne i, 1 \le k \le n+l, \end{aligned} \tag{4.29}$$

where I_i is the set of argument indexes of f_i, that is $j \in I_i$, given $v_i = f_i\left(\{v_j\}_{j \prec i}\right)$, $|I_i|$ signifies the total number of arguments of f_i. Since for $k \ne i$, there is no change for perturbation $\dot{\mathbf{v}}_k^i$, the superscript i of $\dot{\mathbf{v}}_k^i$ is therefore neglected, and we rewrite (4.29) in scalar form as follows

$$\begin{aligned} \dot{v}_i &= \sum_{j=1}^{|I_i|} \dot{v}_j \cdot \tfrac{\partial f_i}{\partial v_j}, \quad j \in I_i, \\ \dot{v}_k &= \dot{v}_k, \qquad\qquad k \ne i. \end{aligned} \tag{4.30}$$

Formula (4.27) holds for each intermediate variable $\mathbf{v}^i$, we have

$$\hat{\mathbf{f}}_k\left(\mathbf{v}^i, \dot{\mathbf{v}}^i\right) = \hat{\mathbf{f}}_k\left(\mathbf{v}^{i-1}, \dot{\mathbf{v}}^{i-1}\right), \tag{4.31}$$

hence

$$\left\langle \bar{\mathbf{v}}^i, \dot{\mathbf{v}}^i \right\rangle = \left\langle \bar{\mathbf{v}}^{i-1}, \dot{\mathbf{v}}^{i-1} \right\rangle . \tag{4.32}$$

Substitute (4.28) into (4.32), and then apply the definition of adjoint operator 3.3, we have

$$\left\langle \bar{\mathbf{v}}^{i-1}, \dot{\mathbf{v}}^{i-1} \right\rangle = \left\langle \bar{\mathbf{v}}^i, A_i|_{\mathbf{v}_0^{i-1}} \cdot \dot{\mathbf{v}}^{i-1} \right\rangle = \left\langle A_i^T|_{\mathbf{v}_0^{i-1}} \cdot \bar{\mathbf{v}}^i, \dot{\mathbf{v}}^{i-1} \right\rangle , \tag{4.33}$$

that is

$$\bar{\mathbf{v}}^{i-1} = A_i^T|_{\mathbf{v}_0^{i-1}} \bar{\mathbf{v}}^i. \tag{4.34}$$

With A_i equal to (4.18), we have the differentiation formulae in reverse mode:

$$\begin{aligned} \bar{\mathbf{v}}_k^{i-1} &= \bar{\mathbf{v}}_k^i + \bar{\mathbf{v}}_i^i \cdot c_{i,k}, \quad k \in I_i, k \neq i \\ \bar{\mathbf{v}}_i^{i-1} &= \bar{\mathbf{v}}_i^i \cdot c_{i,i}, \qquad k \in I_i, k = i \\ \bar{\mathbf{v}}_k^{i-1} &= \bar{\mathbf{v}}_k^i, \qquad k \notin I_i \end{aligned} \tag{4.35}$$

Since for $k \in I_i, k \neq i$, there are no changes of adjoint variables, and for $k \in I_i$, there are accumulations of adjoint variables. The superscript i of $\bar{\mathbf{v}}_k^i$ can therefore be neglected. We rewrite (4.35) in scalar form as follows:

$$\begin{aligned} \bar{v}_k &= \bar{v}_k + \bar{v}_i \cdot \frac{\partial f_i}{\partial v_k}, \quad k \in I_i, k \neq i \\ \bar{v}_i &= \bar{v}_i \cdot \frac{\partial f_i}{\partial v_i}, \qquad i \prec i \\ \bar{v}_i &= 0, \qquad i \not\prec i \end{aligned} \tag{4.36}$$

Proposition 4.1 *By performing the differentiation rule (4.36) with adjoint variables $\bar{\mathbf{y}}$ initialized as I, and with intermediate adjoint variables $\bar{\mathbf{v}}'$ initialized as 0, the results of backward propagation of the differentiation rule provide the gradient.*

proof [1] Suppose that a procedure (4.11) evaluate function (4.15). For each intermediate function $\mathbf{f}^i$, its Jacobian is as (4.18).

Perturbating $\mathbf{x}$ with $\dot{\mathbf{x}}$, according to chain rule, we have

$$\dot{\mathbf{y}} = Q_m A_{l-m+1} \ldots A_2 A_1 P_n^T \dot{\mathbf{x}}. \tag{4.37}$$

The propagation of intermediate variable satisfies (4.28). We rewrite (4.37) as

$$\begin{bmatrix} \dot{\mathbf{y}} \\ \dot{\mathbf{v}}' \end{bmatrix} = T\left(\dot{\mathbf{x}}\right), \tag{4.38}$$

with $\dot{\mathbf{v}}'$ denoting all intermediate perturbation $[\dot{v}_1, \ldots \dot{v}_{l-m}]^T$, and with operator T denoting tangent linear model (TLM) of the composite function (4.15).

According to the differentiation rule in matrix form (4.34), with some initial quantities $[\bar{\mathbf{y}}, \bar{\mathbf{v}}']^T$, the backward propagation is

[1] The presentation and proof of the equivalence are inspired by the discussion with Marc Honnorat

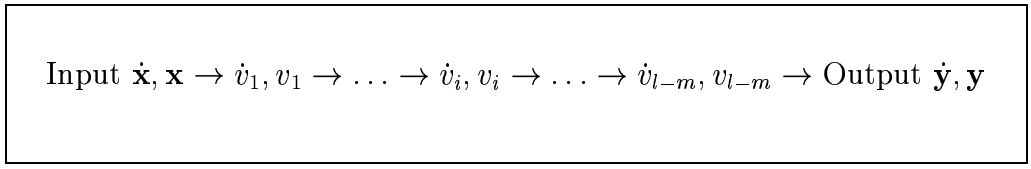


Figure 4.5: Differentiation diagram in forward mode, where $\dot{v}_i$ is differentiated according to rule (4.30).

$$\bar{\mathbf{x}}^T = P_n A_1^T A_2^T \ldots A_{l-m+1}^T Q_m^T \bar{\mathbf{y}}^T. \tag{4.39}$$

We rewrite (4.39) as follows,

$$\bar{\mathbf{x}} = A\left(\begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix}\right), \tag{4.40}$$

where A denotes the operator of adjoint model of the composite function. Now we check the inner product $\left\langle \begin{bmatrix} \dot{\mathbf{y}} \\ \dot{\mathbf{v}}' \end{bmatrix}, \begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix} \right\rangle$:

$$\begin{aligned} \left\langle \begin{bmatrix} \dot{\mathbf{y}} \\ \dot{\mathbf{v}}' \end{bmatrix}, \begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix} \right\rangle &= \left\langle T(\dot{\mathbf{x}}), \begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix} \right\rangle \\ = \left\langle \dot{\mathbf{x}}, T^T\left(\begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix}\right) \right\rangle &= \left\langle \dot{\mathbf{x}}, A\left(\begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix}\right) \right\rangle = \langle \dot{\mathbf{x}}, \bar{\mathbf{x}} \rangle . \end{aligned} \tag{4.41}$$

That is,

$$\left\langle \begin{bmatrix} \dot{\mathbf{y}} \\ \dot{\mathbf{v}}' \end{bmatrix}, \begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{v}}' \end{bmatrix} \right\rangle = \langle \dot{\mathbf{x}}, \bar{\mathbf{x}} \rangle . \tag{4.42}$$

Setting

$$\bar{\mathbf{y}} = I, \bar{\mathbf{v}}' = 0, \tag{4.43}$$

we have

$$\dot{\mathbf{y}} = \langle \dot{\mathbf{x}}, \bar{\mathbf{x}} \rangle . \tag{4.44}$$

By the definition 3.2, we have

$$\bar{\mathbf{x}} = \nabla_{\mathbf{x}} \mathbf{f}_k, \qquad 1 \le k \le m. \tag{4.45}$$

■

The differentiation diagrams in forward and reverse modes are shown in figure 4.5 and figure 4.6 respectively.

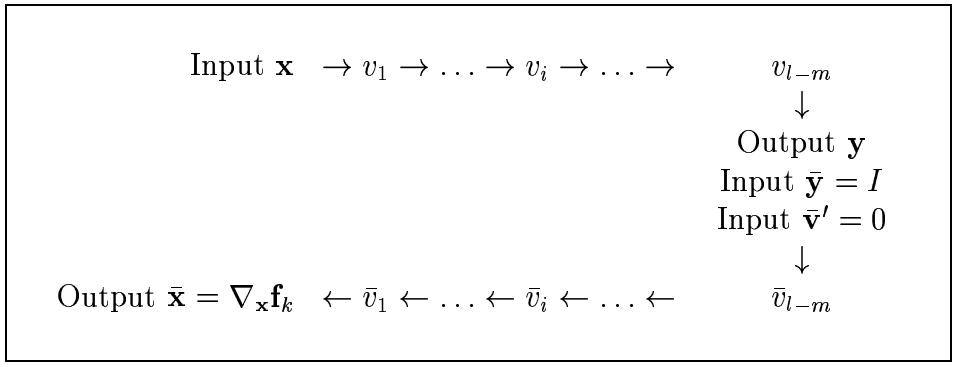


Figure 4.6: Differentiation diagram in reverse mode, where $\bar{v}_i$ is differentiated according to rule (4.36).

```
y = exp(x1)*(4*x1^2 + 2*x2^2 + 4*x1*x2 + 1);
```

Figure 4.7: Direct code of function (4.46)

Example

Considering the following function

$$y = f(x_1, x_2) = \exp(x_1) \cdot \left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1\right) \tag{4.46}$$

used in the demo of MATLAB optimization toolbox, its gradient is

$$\begin{array}{l} \frac{\partial f}{\partial x_1} = y + \exp(x_1) \cdot (8x_1 + 4x_2), \\ \frac{\partial f}{\partial x_2} = \exp(x_1) \cdot (4x_1 + 4x_2 + 2). \end{array} \tag{4.47}$$

The MATLAB code of function (4.46) may look like as in figure 4.7,

Let `x1ad, x2ad, yad` be the adjoint variables of `x1, x2`, and `y`, the adjoint code derived by differentiation rule (4.36) is as figure 4.8

The input of this piece of adjoint code is `x1, x2, yad`, and its output is `y, x1ad, x2ad`, thus we denote a subroutine named `fun_ad`. One call of this subroutine is exemplified in figure 4.9. Note that the adjoint variable is initialized according to formula (4.43), and the running results `[x1ad, x2ad]` provide exactly the gradient (4.47) at `[x1, x2]`.

```
y = exp(x1)*(4*x1^2 + 2*x2^2 + 4*x1*x2 + 1)
x1ad = x1ad + yad * (y + exp(x1) * (8*x1 + 4*x2));
x2ad = x2ad + yad * exp(x1) * (4*x1 + 4*x2 + 2);
yad = 0;
```

Figure 4.8: Adjoint code of function (4.46)

```
% initialization
x1ad = 0;
x2ad = 0;
yad = 1;

% call the adjoint subroutine
[x1ad, x2ad, y] = fun_ad (yad, x1, x2);
```

Figure 4.9: Gradient calculation using adjoint code

4.2.3 Differentiability and complexity [54]

In practice, program code usually consists of thousands or even millions of lines with complex structures like loops, switches, and subroutines. The concepts is that whatever the structure is, the program can be considered as a sequence of elementary arithmetic operations, i.e. +, -, *, /, nonlinear functions of a single argument, i.e. exp(), and if-then thresholds. The sequence of evaluation procedure represents arbitrary f as a composition function of its elemental components f_i (4.13), which are selected from certain set Γ:

$$\Gamma \equiv \{c, \pm, \times, /, \exp, \sin, \ldots\}. \tag{4.48}$$

The computational graph of the evaluation procedure is as figure 4.4 shows. Supposing $f_i \in \Gamma$ satisfies elemental differentiability (ED) as follows [54]

Assumption ED: *All elemental function f_i are d times continuously differentiable on open domains D_i, that is $f_i \in \mathrm{C}^d(D_i), 0 \leq d \leq \infty$.*

with the following proposition (chain rule)

Proposition 4.2 *[54] Under assumption ED, for the function $y = f(x) : x \in D \subseteq \Re^n \mapsto \Re$, which is defined by evaluation procedure, we have $f \in \mathrm{C}^d(D), 0 \leq d \leq \infty$.*

When all the elemental functions are smooth ones like + - * /, sin(), exp(), the composite function f is at least once continuously differentiable. If the evaluation procedure contains nondifferentiable functions like sign() and max(), the composite funtion is directionally differentiable.

AD techniques ensure *a priori* bounds upon the cost of evaluating derivative with that of evaluating function itself as a reference. When considering accesses to memory hierarchy, vectorization and parallelism in multiprocessor systems, the realistic complexity modelling is more difficult than the counting process of additions, multiplications and other arithmetic operations. In order to achieve complete analysis of complexity, Griewank developed a flexible complexity model that regards extended programs as additive parts of evaluation procedures (For extreme details, see [54]).

Adjoint and tangent code for elemental sequences

In this section we derive the adjoint and tangent code for function (4.46) in form of elemental function sequence that satisfies the ED assumption. The evaluation procedures

are therefore concatenations of elemental functions that are taken from set (4.48). Realistic derivation of adjoint and tangent is similar when substituting elemental functions with composite functions (one line of evaluation code), and processing adjoint derivation according to differentiation formula (4.36). Note that there is no variable overwriting of v_i in the example, thus no additional safeguard of these variable values is needed, whereas in practical applications, the safeguard demands special caution, as is usually a dilemma of space and efficiency that is the main subject of check-point algorithm of Griewank. The derivation is shown as in figure 4.10.

By setting (x1, x2) = $(-1, 1)$ and (x1tg, x2tg) = $(0, 1)$, the output of tangent code (b) gives $\frac{\partial f}{\partial x_2}$. With intermediate adjoint variables viad, i=-1,0,..,9 and input variables x1ad, x2ad initialized as as zero, and with the adjoint variable yad equal to one, the output of adjoint code (c) provides the numerical values of the gradient:

$$\begin{aligned} \text{x1ad} &= \left.\frac{\partial f}{\partial x_1}\right|_{(\text{x1,x2})=(-1,1)} \\ \text{x2ad} &= \left.\frac{\partial f}{\partial x_2}\right|_{(\text{x1,x2})=(-1,1)} \end{aligned} \tag{4.49}$$

4.2.4 AD in the viewpoint of discrete adjoint model

In this section we try to elucidate the link between variational formulism in chapter 3 and AD techniques via the formulation of discrete adjoint model, for details we refere to the work of [137].

For numerical simulation purpose the variational formulation of the model in system (3.1) and adjoint equations (3.30) has to be discretized. We usually build the adjoint model directly from a discrete forward model, rather than by discretizing these continuous adjoint equations (3.30), such that the computational inconsistency can be avoided [137]. AD techniques, especially differentiation rules in matrix form (4.34), are interpreted as the transposition of the matrices that correspond to tangent code [137], [49] to calculate the item of transposed Jacobian $\left[\frac{\partial F}{\partial \mathbf{X}}\right]^T P$ in (3.30), where P is the adjoint variable [81].

We illustrate the relation between AD and adjoint model in the case of data assimilation problem, in which the model (3.15) and the cost function (3.16) are discretized and denoted

$$\begin{aligned} \mathbf{X}(t_r) &= D_{0\to r}\left(\mathbf{X}\right)\mathbf{X}\left(t_1\right) \\ &= D_{0\to 1}D_{1\to 2}\ldots D_{r-1\to r}\left(\mathbf{X}\right)\mathbf{X}\left(t_1\right) \end{aligned} \tag{4.50}$$

and

$$J\left(\mathbf{X}_0\right) = \frac{1}{2}\sum_{r=1}^{n}\left\langle\left(\mathbf{X}_r - \tilde{\mathbf{X}}_r\right), \left(\mathbf{X}_r - \tilde{\mathbf{X}}_r\right)\right\rangle, \tag{4.51}$$

where n is the total number of time intervals on which observations $\tilde{\mathbf{X}}_r$ are available, and the dynamic model is as

$$\mathbf{X}\left(t_r\right) = D_{r-1\to r}(\mathbf{X})\mathbf{X}\left(t_{r-1}\right), \tag{4.52}$$

```
x1 = -1;
x2 = 1;
v_1 = x1;
v0 = x2;
v1 = exp(v_1);
v2 = 4 * v_1 ^ 2;
v3 = 2 * v0 ^ 2;
v4 = 4 * v0 * v_1;
v5 = 2 * v0;
v6 = v2 + v3;
v7 = v4 + v5;
v8 = v6 + v7;
v9 = v1 * v8;
y = v9;
```

(a)

```
x1 = -1; x1tg = 0.0;
x2 = 1; x2tg = 1.0;
v_1 = x1;
v_1tg = x1tg;
v0 = x2;
v0tg = x2tg;
v1 = exp (v_1);
v1tg = v1 * v_1g;
v2 = 4 * v_1 ^ 2;
v2tg = 8 * v_1 * v_1tg;
v3 = 2 * v0 ^ 2;
v3tg = 4 * v0 * v0tg;
v4 = 4 * v0 * v_1;
v4tg = 4*v0tg*v_1 + 4*v0*v_1tg;
v5 = 2 * v0;
v5tg = 2 * v0tg;
v6 = v2 + v3;
v6tg = v2tg + v3tg;
v7 = v4 + v5;
v7tg = v4tg + v5tg;
v8 = v6 + v7 + 1;
v8tg = v6tg + v7tg;
v9 = v1 * v8;
v9tg = v1tg * v8 + v1 * v8tg;
y = v9;
ytg = v9tg;
```

(b)

```
% model direct ...

x1 = -1;
x2 = 1;
v_1 = x1;
...
y = v9

% Initialize adjoint of
% intermediate variables
% viad and input variables
% xjad equal to zero;

yad = 1.0;

v9ad = v9ad + yad;
yad = 0.0;
v1ad = v1ad + v9ad*v8;
v8ad = v8ad + v9ad*v1;
v9ad = 0.0;
v6ad = v6ad + v8ad;
v7ad = v7ad + v8ad;
v8ad = 0.0;
v4ad = v4ad + v7ad;
v5ad = v5ad + v7ad;
v7ad = 0.0;
v2ad = v2ad + v6ad;
v3ad = v3ad + v6ad;
v6ad = 0.0;
v0ad = v0ad + 2*v5ad;
v5ad = 0.0;
v0ad = v0ad + v4ad*4*v_1;
v_1ad = v_1ad + v4ad*4*v0;
v4ad = 0.0;
v0ad = v0ad + v3ad * 4*v0;
v3ad = 0.0;
v_1ad = v_1ad + v2ad*8*v_1;
v2ad = 0.0;
v_1ad = v_1ad + v1ad * exp(v_1);
v1ad = 0.0;
x2ad = v0ad;
x1ad = v_1ad;
```

(c)

Figure 4.10: Direct code (a), tangent (b) and adjoint (c) of function (2.34) in form of elemental sequence

where $D_{r-1\to r}(\mathbf{X})$ is the nonlinear operator corresponding to the model. Herein let $\mathbf{X}_n = \mathbf{X}(t_n = T_1), \mathbf{X}_0 = \mathbf{X}(t_1 = T_0)$. Perturbating (4.50) and (4.51), we have tangent linear model (TLM)

$$\begin{aligned} \hat{\mathbf{X}}(t_r) &= A_{0\to r}(\mathbf{X})\,\hat{\mathbf{X}}(t_1) \\ &= A_{0\to 1}\ldots A_{r-1\to r}(\mathbf{X})\,\hat{\mathbf{X}}(t_1) \end{aligned} \tag{4.53}$$

and

$$\hat{J}(\mathbf{X}_0) = \sum_{r=1}^{n} \left\langle \left(\mathbf{X}_r - \tilde{\mathbf{X}}_r\right), \hat{\mathbf{X}}_r \right\rangle. \tag{4.54}$$

According to definition 3.2,

$$\hat{J}(\mathbf{X}_0) = \left\langle \nabla_{\mathbf{X}_0} J, \hat{\mathbf{X}}_0 \right\rangle. \tag{4.55}$$

Substituting (4.53) into (4.54) (note that $\hat{\mathbf{X}}_r = \hat{\mathbf{X}}(t_r)$), then associate the results with (4.55) and apply the definition 3.3 on adjoint operator, we have

$$\begin{aligned} \left\langle \nabla_{\mathbf{X}_0} J, \hat{\mathbf{X}}_0 \right\rangle &= \left\langle \sum_{r=1}^{n} \left(\mathbf{X}_r - \tilde{\mathbf{X}}_r\right), A_{0\to r}(\mathbf{X})\,\hat{\mathbf{X}}_0 \right\rangle \\ &= \left\langle \sum_{r=1}^{n} A^T_{r\to 0}(\mathbf{X}) \left(\mathbf{X}_r - \tilde{\mathbf{X}}_r\right), \hat{\mathbf{X}}_0 \right\rangle. \end{aligned} \tag{4.56}$$

That is

$$\nabla_{\mathbf{X}_0} J = \sum_{r=1}^{n} A^T_{r\to 0}(\mathbf{X}) \left(\mathbf{X}_r - \tilde{\mathbf{X}}_r\right), \tag{4.57}$$

where

$$A^T_{r\to 0}(\mathbf{X}) = A^T_{1\to 0} A^T_{2\to 1} \ldots A^T_{r\to r-1}(\mathbf{X}). \tag{4.58}$$

Let

$$d_r = \mathbf{X}_r - \tilde{\mathbf{X}}_r, \qquad r = 1, \ldots n. \tag{4.59}$$

Considering the evaluation structure (4.58), (4.57) can be rewritten in recursive form [14],

$$\begin{aligned} \nabla_{\mathbf{X}_0} J &= \sum_{r=1}^{n} A^T_{r\to 0}(\mathbf{X})\, d_r \\ &= A^T_{1\to 0} A^T_{2\to 1} \ldots\ldots\ldots A^T_{n-2\to n-1} A^T_{n\to n-1}(\mathbf{X})\, d_n + \\ &\quad A^T_{1\to 0} A^T_{2\to 1} \ldots A^T_{n-3\to n-2} A^T_{n-1\to n-2}(\mathbf{X})\, d_{n-1} + \\ &\quad A^T_{1\to 0} A^T_{2\to 1}(\mathbf{X})\, d_2 + \\ &\quad A^T_{1\to 0}(\mathbf{X})\, d_1 \\ &= A^T_{1\to 0}\left[d_1 + A^T_{2\to 1}\left[d_2 + \ldots + A^T_{n\to n-1}(\mathbf{X})\, d_n\right]\ldots\right]. \end{aligned} \tag{4.60}$$

By introducing adjoint model

$$\begin{array}{rcll} \bar{\mathbf{X}}_{r-1} & = & A^T_{r\to r-1}\left(\mathbf{X}\right)\left(\bar{\mathbf{X}}_r + d_r\right), & r = n, \ldots, 1, \\ \bar{\mathbf{X}}_n & = & 0, & \end{array} \tag{4.61}$$

and associating (4.60) and (4.61), the gradient is thus

$$\nabla_{\mathbf{X}_0} J = \bar{\mathbf{X}}_0. \tag{4.62}$$

In fact, the adjoint model can be partly implemented based on AD techniques. Suppose that program code, which implements direct model of $D_{r-1\to r}\left(\mathbf{X}\right)$, is as

$$D_{r-1\to r}\left(\mathbf{X}\right) = F_N\left(\mathbf{V}_N\right)\ldots F_2\left(\mathbf{V}_2\right)F_1\left(\mathbf{V}_1\right), \tag{4.63}$$

where matrix $F_i, i = 1, \ldots, N$ represents either a piece of code or a subroutine, $\mathbf{V}_i$ represents intermediate variables. Then the tangent linear model (4.53) and adjoint model (4.61) can be obtained by forward and reverse mode of AD respectively according to differentiation rule in matrix form (4.28) and (4.34). That is

$$\begin{array}{rcll} A_{r-1\to r}\left(\mathbf{X}\right) & = & A_N \ldots A_2 A_1\left(\mathbf{X}\right) & \\ A_i & = & \frac{\partial F_i}{\partial \mathbf{V}_i}, & i = 1, \ldots, N \end{array} \tag{4.64}$$

and

$$A^T_{r\to r-1}\left(\mathbf{X}\right) = A^T_1 A^T_2 \ldots A^T_N\left(\mathbf{X}\right). \tag{4.65}$$

Implementing model (4.61) with resulting AD adjoint code (4.65), the gradient is given by (4.62). When cost function takes the form other than (4.51), by properly introducing adjoint model (4.61), we can obtain similar results of AD coding (4.65) and gradient calculation (4.62) [137].

4.3 Practical AD coding by hand for *GreenLab* problem P2-P3

In this section we deal with the implementation of AD or adjoint techniques applied to *GreenLab* source code. There do exist AD tools like ADMAT [119] or ADiMAT [118], however, the former employs the technique of operator overloading and cannot process matrices with more than 3 dimensions; and the latter follows the source-to-source approach but concentrates only on the forward mode of differentiation. With these deficiencies in mind we apply AD in both forward and reverse modes directly line by line based on the source code (thus called AD by hand).

In practice, we follow three steps: preprocessing, differentiation and postprocessing. *GreenLab* AD coding experiences, which are usually problem-independent, are recorded in this section. Note that validation results for *GreenLab* optimal control and data assimilation problems in the postprocessing step are presented in the sections subsequent to the problem formulation sections in chapter 6.

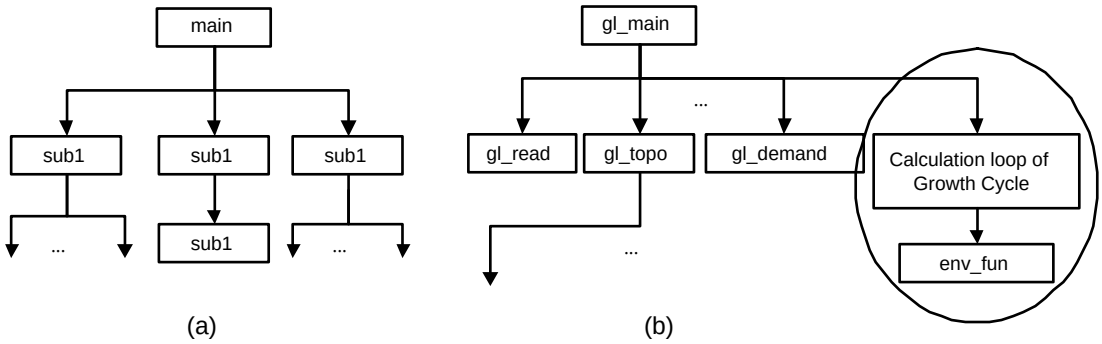


Figure 4.11: A general code tree (a), and the code tree (b) for the the *GreenLab* implementation of the recursive equation (2.64), the parts in the circle need to be differentiated.

x → QSW, QTW;	TP → Q;
QSW → E;	sumiaj → TPM;
QTW → Q;	QO → QTp;
Q → sumiaj, QO;	QTp → QT;
E → TPM;	QT → CostJ;
TPM → TP;	

Figure 4.12: Variable trace of the code in figure 4.11-(b). Here $v_i \rightarrow v_j$ indicates that v_j depends directly on v_i.

4.3.1 Preprocessing – Code analysis

The treatment of preprocessing is the code analysis that prepares for the differentiation. It is mainly composed of the determination of *code tree* that helps to locate the code needed to be differentiated, and the *variables trace* that is used to pilot the differentiation sequence.

Code tree

The code tree describes the calling relationships of all the subroutines in the source code of a numerical model, as is shown in figure 4.11. We need to distinguish the simulation parts, which calculate the dynamics and cost function and need to be differentiated, from auxiliary subroutines like data reading and saving.

Variable trace

The variables trace records the sequence (4.11) and the computational graph (figure 4.4). The variables in the variable trace are those that depend directly or indirectly on the independent variables and thus have influences on the cost function. These variables are called *active*. The constants or other auxiliary variables that are not listed in variable trace are called *passive*. Each active variable is associated with an adjoint variable, and the code for its calculation is differentiated according to differentiation rule (4.36) to construct an adjoint statement. There is no need of adjoint statements for passive variables. Part of variables trace for the code in figure 4.11-(b) is shown in figure 4.12.

```
subroutine Z = sub(T)
...
expression 1                    do I=1,N
expression 2                      Expression 7
if (condition) then               Expression 8
  expression 3                  end do
  expression 4                  expression 9
else                            expression 10
  expression 5                  ...
  Q = sub1(P)                   return
  expression 6                  end subroutine
endif
```

Figure 4.13: Source code of a general subroutine, taken from [81].

4.3.2 Differentiation – Coding techniques

The code of numerical model is usually written in standard programming languages that consist mainly of only a few elements, such as assignments, conditional statements, loops, subroutine calls and I/O statements, as makes it possible to develop automatic differentiation tools [49]. In this section we firstly conduct AD coding for a general subroutine that covers most of the programming structures, then some specific coding techniques are introduced.

AD coding for a general subroutine

Suppose that a subroutine is shown as in figure 4.13. Its adjoint code (figure 4.14 can be obtained by firstly recalling the direct code and safeguarding the variables in trace, i.e. in figure 4.12, into some tape (can be either a big array or a file), and then reversing the code and applying differentiation rule (4.36) to each line of the general subroutine, if necessary, reading the variables values from the tape.

Specific techniques

Treatment of matrix intrinsic functions and operators For the scientific programming language based on matrix operation, i.e. MATLAB and Scilab, the differentiation for their intrinsic functions and operators demands one step further of the differentiation rule (4.36) taking into account of matrix features. Take the intrinsic function 'sum' as example, the direct code is

$$\mathrm{QT}\,(:,1) = \mathrm{sum}\,(\mathrm{QTp}, 2)\,,$$

where $\mathrm{QTp} \in [M]_{m\times n}$ that has n columns. That is

$$\mathrm{QT}\,(i,1) = \sum_{j=1}^{n} \mathrm{QTp}\,(i,j) \qquad i = 1,\ldots,m.$$

```
subroutine [Z, adT] = ad_sub (T, adZ)
...
% direct code
save 1
expression 1
save 2
expression 2
test1 = condition
if (condition) then
  save 3
  expression 3
  save 4
  expression 4
else
  save 5
  expression 5
  save sub1
  Q = sub(P)
  save 6
  expression 6
end if
do I = 1, N
  save 7
  expression 7
  save 8
  expression 8
end do
save 9
expression 9
save 10
expression 10
```

```
% differentiation code
load 10
ad_expression 10
load 9
ad_expression 9
do I = N:-1:1
  load 8
  ad_expression 8
  load 7
  ad_expression 7
end do
if (test1) then
  load 4
  ad_expression 4
  load 3
  ad_expression 3
else
  load 6
  ad_expression 6
  load sub1
  [Q,adP] = ad_sub1(P, adQ)
  load 5
  ad_expression 5
end if
load 2
ad_expression 2
load 1
ad_expression 1
...
return
end subroutine ad_sub
```

Figure 4.14: Adjoint code of the general subroutine in figure 4.13, taken from [81].

```
saveQT = QT;
QT(:,1) =sum(QTp,2);
```

(a) direct code

```
QT = saveQT;
sztemp = size(QTp);
QTpad = QTpad+QTad*ones(1,sztemp(2));
QTad = 0;
```

(b) adjoint code

Figure 4.15: Adjoint coding example about intrinsic matrix function sum.

```
for i = 1:N
  ...
  for p = 1:maxp
    if i ¡= Tu_O(id_B,1,m)
      tt = i;
    end
    ...
    refer to tt
    ...
  end   % for p
end     % for i
```

(a) direct code

```
for i = N:-1:1
  ...
  for p = maxp:-1:1
    if i ¡= Tu_O(id_B,1,m)
      tt = i;
    end
    ...
    refer to tt
    ...
  end   % for p
end     % for i
```

(b) adjoint code

Figure 4.16: Adjoint coding example about the treatment of passive variables.

Let QTad,QTpad be the adjoint variables of QT,QTp, applying differentiation rule (4.36), we have

$$\mathrm{QTpad}\,(i,j) = \mathrm{QTpad}\,(i,j) + \mathrm{QTad}\,(i) \qquad i = 1 \ldots m, j = 1 \ldots n.$$

The MATLAB implementation of formula above is shown as figure 4.15.

Treatment of passive variables Passive variables influence the value of cost function too, although there are no adjoint statements associated with them. The positions of passive variables are flexible in the adjoint code regardless of the reverse manner of adjoint coding. The principle is that whenever a passive variable is referred to in the adjoint code, its value should be exactly the same as that in the corresponding direct code. For example, in the direct code of the *GreenLab* recursive formulae 2.64 the simulation of plant growth depends on the leaf functioning τ (in direct code denoted tt, figure 4.16), which is a passive variable and is less than leaf functioning timespan τ_p^a that is stored in Tu_O(id_B,1,m). The corresponding adjoint code concerning τ^a is shown as follows.

Treatment of denominators Some numerator terms in direct code may become denominators after differentiation, such as intrinsic function SQRT. When these terms become zero, the output of TLM or adjoint code will be NaN. Add a small number ε to the

```
...
ew_val = y.^(K1-1) .* (1-y).^(K2-1) /Normwt;
```

(a) direct code

```
...
flagy0 = 0;
if y==0
  flagy0 = 1;
  y = y+eps;
end

flagy1 = 0;
if y==1
  flagy1 = 1;
  y = y-eps;
end
```

```
yad = yad + ew_valad / Normwt *
    ((K1-1)*y.^(K1-2)*(1-y).^(K2-1)-y.^(K1-1)
    * (K2-1) * (1-y).^(K2-2));
ew_valad= 0;

if flagy0 == 1
  y = y-eps;
end
if flagy1 == 1;
  y = y+eps;
end
```

(b) adjoint code

Figure 4.17: Adjoint coding example about the treatment of denominators, where eps is a small real number chosen as the floating point relative accuracy.

denominator in this case, as the following example indicates in figure 4.17.

4.3.3 Postprocessing – Validation

Let us denote $y = D(x)$ any block of direct code, with $\delta y = A(\delta x)$ representing its TLM and $\bar{x} = A^T(\bar{y})$ representing its adjoint. The validation process follows the approach in [137]. We present the validation formulae in this section, and the numerical validation results are shown in chapter 6.

Validation of TLM code

Perturbating direct code, according to Taylor series we have

$$D(x + \alpha h) - D(x) = \alpha A(h) + O(\alpha \|h\|). \tag{4.66}$$

With $\alpha \to 0$, we check the function $f(\alpha)$

$$\begin{aligned} \lim_{\alpha \to 0} f(\alpha) &= \lim_{\alpha \to 0} \frac{D(x+\alpha h) - D(x)}{\alpha A(h)} \\ &= 1 \end{aligned} \tag{4.67}$$

for the correctness of TLM code.

Validation of adjoint code

Both TLM and adjoint are linear model. Imposing a perturbation h to the TLM, we have the following algebraic identity

$$(Ah)^T Ah = h^T A^T (Ah) = h^T \left(A^T (Ah)\right). \tag{4.68}$$

The check procedure is then

- Generate h.
- Given h, run TLM and record the result $y = Ah$.
- Evaluate scalar product $y^T y$.
- Given y, run the adjoint code and record the result $\bar{x}$.
- Evaluate scalar product $h^T \bar{x}$.
- Evaluate the identity $y^T y \equiv h^T \bar{x}$. If this identity is valid within machine accuracy, than the adjoint code is correct.

Validation of gradient

The check of gradient is similar to the validation of TLM. Suppose that cost function with respect to control x can be denoted as $J(x)$, its Taylor series,

$$J(x + \alpha h) - J(x) = \alpha \langle \nabla_x J, h \rangle + O(\alpha \|h\|). \tag{4.69}$$

With $\alpha \to 0$, we check the function $f(\alpha)$:

$$\begin{aligned} \lim_{\alpha \to 0} f(\alpha) &= \lim_{\alpha \to 0} \frac{J(x+\alpha h) - J(x)}{\alpha \langle \nabla_x J, h \rangle} \\ &= 1. \end{aligned} \tag{4.70}$$

4.4 Conclusion

In essence AD is the technique of gradient calculation for an optimization algorithm, especially for local ones that needs gradients. AD in reverse mode has a close relationship with adjoint model. In this chapter several issues on how to perform automatic differentiation by hand are introduced. We record some experiences on which we develop adjoint code for numerical solutions of the *GreenLab* optimal control problem and the data assimilation problem in chapter 6. The adjoint code can improve the performance of gradient-based optimization algorithms profoundly, in particular for those of which the cost function is time-consuming and there are enormous variables to optimize (exemplified in section 6.6, table 6.7).

Chapter 5

Variational methods applications I: model calibration

Résumé

L'identification des paramètres environnementaux est la matière principale. Nous réalisons les expériences jumelles pour produire des données artificielles d'observation pour l'identification. Les méthodes variationnelles sont utilisées pour retrouver les valeurs des paramètres. Les alogrithmes de Levenberg-Marquardt est adopté pour l'optimisation itérative. Les gradients sont calculés par différence finie, puisque il y'a que quatre paramètres à identifier. Les résultats de calibrage dans ce chapitre montrent l'efficacité et la praticabilité de l'approche variationnelle. Les valeurs du paramètre peuvent toujours être identifiées, de toute façon les conditions environnementales fluctuent quand la période d'expansion de la feuille est considérablement longue. L'influence des erreurs d'observation sur les résultats de calibrage est également étudiée.

5.1 Introduction

The calibration problem (**P1**) answers partly the question that which values of model parameters should be taken once the model has been set up. It is also called an inverse problem, and the algorithms that resolve the problem are thus named as inverse methods. In most cases it involves minimization of a sum of least squares that measure the difference between model simulation results and corresponding observations.

Calibration should be the first step for *GreenLab* to be applicable. Let us denote a vector function g that encapsulates *GreenLab* FSPM dynamics

$$\{^m y_{i,j,k}\} = g(\Omega_A, \Theta, \mathbf{U}), \tag{5.1}$$

where Θ is the initial condition, i.e. seed volume, $\mathbf{U}$ is the model input, such as light and temperature environmental conditions, $\{^m y_{i,j,k}\}$ is *GreenLab* simulation results, whose superscripts and subscripts are defined as the observation set (2.75), Ω_A are model parameters defined as (2.66):

$$\Omega_A = \Omega_G \cup \Omega_E = \Omega_H \cup \Omega_D \cup \Omega_E \tag{5.2}$$
$$\Omega_G = \{r_1, r_2\} \cup \{P_p^o, a_p^o, b_p^o\} \tag{5.3}$$
$$\Omega_E = \{E_M, k, \alpha, \beta\} \tag{5.4}$$

Here we suppose that parameter $\mathbf{K}$ is time-invariant. The calibration problem can be formulated either by variational formulism as optimality system (see chapter 3),

$$\begin{cases} \frac{d\mathbf{X}(t)}{dt} = F(\mathbf{X}(t), \mathbf{K}), \\ \mathbf{X}(T_0) = \Theta, \\ \frac{dP}{dt} = -\left[\frac{\partial F_1}{\partial X}\right]^T P + \mathbf{X}(t) - \tilde{\mathbf{X}}(t), \\ P(T_1) = 0, \\ I(\mathbf{K}) = \frac{1}{2}\int_{T_0}^{T_1} \left\|\mathbf{X}(t) - \tilde{\mathbf{X}}(t)\right\|^2 dt, \end{cases} \tag{5.5}$$

with its gradient as

$$\nabla_{\mathbf{K}} I = -\int_{T_0}^{T_1} \left[\frac{\partial F}{\partial \mathbf{K}}\right]^T P \cdot dt, \tag{5.6}$$

or by introducing vector function (5.1) as minimization of a sum of square terms

$$\min_{\mathbf{K}} \sum_{m=1}^{|\mathcal{M}|} \left\| g(\Omega_A, \Theta, \mathbf{U}) - {}^m y_{i,j,k}^{obs} \right\|^2, \tag{5.7}$$

where $|\mathcal{M}|$ is as (2.76) shows, ${}^m y_{i,j,k}^{obs}$ is as (2.75) shows, and $\mathbf{K}$ is the parameter set to be calibrated.

Note that in practice, objective function $I(\mathbf{K})$ usually contains regularization term for the optimization algorithms to escape from local optima, however in this chapter no regularization is considered in order to investigate the effect of observation errors for calibration. We will introduce regularization term in section 6.7.3.

In previous studies, the calibration is performed mainly for endogenous parameter set Ω_G ($\mathbf{K} = \Omega_G$) [30], [134] based on a constant biomass production potential E under optimal environmental conditions ($\mathbf{K} = \Omega_G$). In this chapter, the calibration is extended to varying environmental conditions. We conduct numerical experiments for the calibration of environmental parameter set Ω_E ($\mathbf{K} = \Omega_E$) under the assumption that the endogenous parameters are independent of fluctuating environment [33].

In the following sections, we will discuss several issues on calibration problem (5.7) and (5.5), then the numerical experiments and calibration results will be presented.

5.2 Several issues for the calibration problem

5.2.1 Optimization algorithm for least squares problem

We employ LM methods in the form of Moré [85] for the parameter identification (see section A.5 in appendix A for algorithm details). An interface is developed under MATLAB

environment (Fortran MEX) for the academic software MINPACK-1 that implements LM methods in Fortran.

5.2.2 Gradient calculation

In general, the gradient of the calibration problem for numerical algorithms can be obtained by either finite difference scheme such as the forward formula (4.1) or by its alternative, AD techniques introduced in chapter 4. The finite difference scheme is employed in this chapter, and the applications of AD techniques are postponed to chapter 6, which is mainly due to

- The LM algorithms take advantage of the special structure of the gradient (A.15) and Hessian matrix (A.16) of the least squares problems, as leads to efficient iterative process.
- The implementation of AD by hand is a tedious and error-prone process, and there is only 4 elements in Ω_E needed to be calibrated.
- For large-scale problems in which enormous parameters need to be optimized, the optimality system say (5.5) shall be crucial (see data assimilation problem in chapter 6).

5.2.3 Validation

The model validation is to evaluate the prediction error of model output $\{{}^{m}y_{i,j,k}\}$ using calibrated parameter $\mathbf{K}^*$. The input ${}^{V}\mathbf{U}$ and the observation $\left\{{}^{V,m}y^{obs}_{i,j,k}\right\}$ for the model validation should be independent of those of ${}^{C}\mathbf{U}$ and $\left\{{}^{C,m}y^{obs}_{i,j,k}\right\}$ for calibration. The initial condition ${}^{V}\Theta$ and model input ${}^{V}\mathbf{U}$ for validation should be provided by related studies (e.g. similar to that of data assimilation problems in meteorology). The relative prediction error δ can be examined, for instance, by

$$\delta = \frac{\sum_{n=1}^{N}\sum_{i,j,k}\left\|g\left(\mathbf{K}^*,{}^{V}\Theta,{}^{V}\mathbf{U}\right) - {}^{V,m}_{n}y^{obs}_{i,j,k}\right\|^2}{\sum_{n=1}^{N}\sum_{i,j,k}\left\|{}^{V,m}_{n}y^{obs}_{i,j,k}\right\|^2}, \tag{5.8}$$

where the left subscript n signifies the GC index, N is the total number of GCs for which observations are measured.

5.2.4 Observation error, ill-posed calibration problem

Introducing dynamics, function (5.1) can be rewritten as

$$\{{}^{m}_{n}y_{i,j,k}\} = f\left({}^{m}_{n-1}y_{i,j,k}, \Omega_A, \Theta, \mathbf{U}\right). \tag{5.9}$$

Considering observation error and model error, we add error item ε , which is chosen to be a random variable with uniform distribution, to the function (5.9), that is,

$$\left\{{}^{S,m}_{n}y_{i,j,k}\right\} = f\left({}^{S,m}_{n-1}y_{i,j,k}, \Omega_A, \Theta, \mathbf{U}\right) + \varepsilon. \tag{5.10}$$

The variable ${}^{m}_{n}y_{i,j,k}$ then becomes a stochastic one ${}^{S,m}_{n}y_{i,j,k}$. The calibration has to be associated with the uncertainty of ε.

A problem is *well-posed* if its solution exists, is unique and depends continuously on the data. It becomes an *ill-posed* one when it fails to satisfy at least one of these criteria [56]. By *ill-posed* calibration problem, we mean that the calibration results $\mathbf{K}^*$ may be non-unique, or change dramatically with respect to slight perturbation to observation data. If (i) there is no observation error, (ii) the map from parameter set Ω_A and model output $\{{}^{m}_{n}y_{i,j,k}\}$ (equation (5.1)) is bijective, the calibration is always well-posed. If not, the regularization of the sum of the least squares (5.7) by some strictly positive definite term is a general way to handle the ill-posed problem.

5.3 Numeric experiments

In this section, we conduct numerical experiments for the calibration of environmental parameter Ω_E, in which mainly three factors are investigated, they are, the environmental fluctuations, the effect of leaf expansion lifespan, and the observation errors.

All the numerical experiments herein and hereafter employ artificial observations that are generated by *twin experiments*. The twin experiments are usually conducted when the *in situ* observations are not ready or impossible to be accessed [132], [135]. Its aim is thus to show the effectiveness and robustness of the working schedule or algorithms. In our case, it can be formulated as

$$\{{}^{\mathrm{ref},m}_{n}y_{i,j,k}\} = g({}^{\mathrm{ref}}\Omega_E, \Theta, \mathbf{U}), \tag{5.11}$$

where ${}^{\mathrm{ref}}\Omega_E$ is set of the values of reference parameters, and ${}^{\mathrm{ref},m}_{n}y_{i,j,k}$ is the artificial observation generated by the simulation of *GreenLab* with ${}^{\mathrm{ref}}\Omega_E$. For all the experiments in this section, we set

$${}^{\mathrm{ref}}\Omega_E = \begin{pmatrix} 1000 & 0.009 & 3.5 & 3.0 \end{pmatrix}. \tag{5.12}$$

The proper virtual plant for these numerical experiments shall be illustrative with respect to the three concerned factors, however, it should not be very complex. We thus chose planche26 (figure 5.1) in [31], the configurations for planche26 are as the example 3 in chapter 2, that is,

$$N = 15, P = 1, M = \{m_{10}\},$$

and

$$\mathbf{R} = \begin{pmatrix} 1 & 0 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 & 0 \end{pmatrix}, \quad \mathbf{M_a} = \mathbf{M_f} = \begin{pmatrix} 1 & 0 \end{pmatrix},$$

$$\overrightarrow{\tau} = (40), \quad \overrightarrow{\mu} = (\bullet), \quad \overrightarrow{\varphi^d} = \overrightarrow{\varphi^f} = \mathrm{I}.$$

Direct parameters are

$$e = 0.05, \quad \tau^a = 5, \quad t^a = t^e = 5, \quad t^f = 1.$$

Some of the hidden parameters are

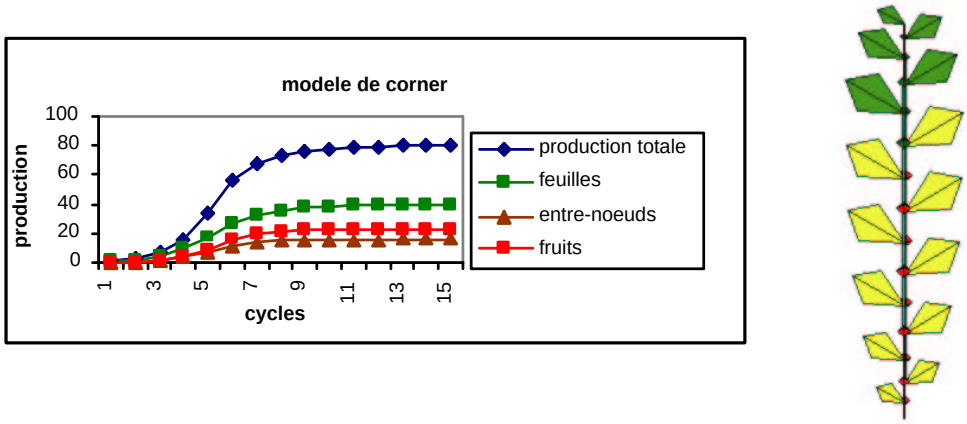


Figure 5.1: The virtual plant *Planche26* for numerical calibration experiments ($E = 1000$) (a) biomass allocation curves for different types of organs, (b) the picture of 3D geometric plant. The leaves turn yellow when they stop functioning. Taken from [31]

$$P^a = 0.5, \quad P^e = 0.2, \quad P^f = 0.3, \quad r_1 = 5000, \quad r_2 = 50.$$

5.3.1 Numerical experiment 1: on environmental fluctuation

Design

The aim is to perform calibration of Ω_E under fluctuant environmental conditions without observation errors.

We keep the leaf expansion lifespan to be 5 GCs. The model inputs (environmental conditions a-d in figure 5.2) vary from smoothly (a) to violently (c), even randomly (d). Twin experiment is conducted for ${}^{\text{ref}}\Omega_E$ to generate target plant data (plant total weight at each GC). For each environmental conditions, we start the calibration with the initial parameters $\Omega_E^0 = 0.8 \times {}^{\text{ref}}\Omega_E$.

Results

With generated target plant data under environmental conditions (a)-(d), we employ LM method to track back the environmental factor parameter value. For all the four cases, we succeed in finding the reference parameters ($\mathbf{K}^* = {}^{\text{ref}}\Omega_E$). Figure 5.2 lists the calibration results and the environmental conditions, 3D geometric plant is shown in figure 5.3.

Discussion

In our experiences for the calibration under variety of environmental conditions, the inverse method can always find the target parameter ${}^{\text{ref}}\Omega_E$, no matter how fluctuant the environmental conditions are. The model input $\mathbf{U}$ does not influence the features of the implicit bijective map in (5.1) between parameter Ω_E and model output $\{{}^m y_{i,j,k}\}$. The calibration for numerical experiment 1 is a well-posed problem.

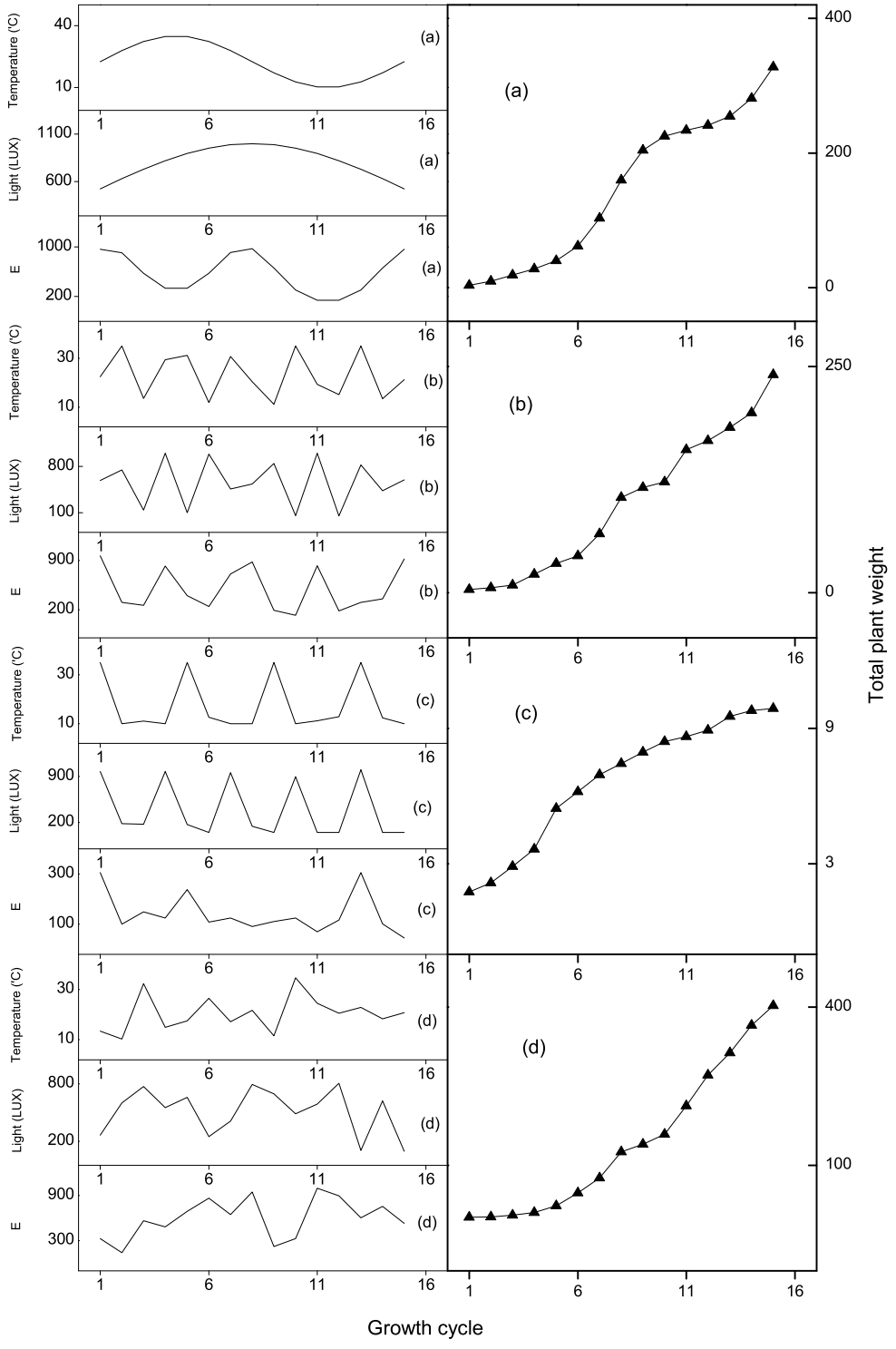


Figure 5.2: Calibration results of environmental factor parameters under fluctuate environmental conditions (a)-(d) without observation errors. For the calibration results (right), the notation '▲' marks the artificial observations, and '—' signifies the fitting curves.

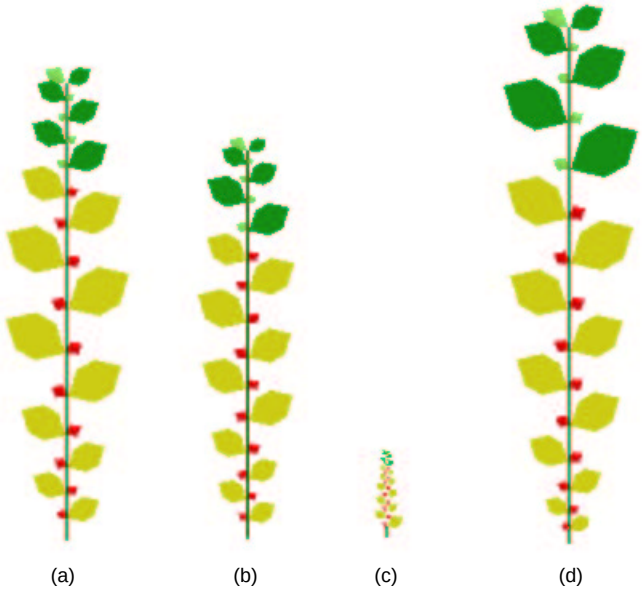


Figure 5.3: 3D geometric plant calculated for the environmental conditions (a)-(d)

5.3.2 Numerical experiment 2: on observation errors

Design

The aim is to perform calibration of Ω_E when the model is not perfect. Observation errors are considered to generate the target plant data. The error item ε in (5.10) is chosen to be a random variable with uniform distribution, such that the plant target data ${}^{S,m}_{n}y_{i,j,k}$ satisfies

$$\frac{\left\| {}^{Sm}_{n}y_{i,j,k} - {}^{\mathrm{ref},m}_{n}y_{i,j,k} \right\|}{\left\| {}^{\mathrm{ref},m}_{n}y_{i,j,k} \right\|} \leq \sigma, \tag{5.13}$$

where $\sigma \in [0, 1]$ is a relative bound of the observation error, target plant data ${}^{\mathrm{ref},m}_{n}y_{i,j,k}$ is calculated according to (5.11).

We set the leaf lifespan to 5 GCs, and environmental conditions are chosen to be the fluctuate case (b) in figure 5.2. Three levels of observation error are considered by setting σ equals to 0.2, 0.1 and 0.05 respectively. For each level, random target plant data are generated 23×3 times for calibration.

Results

We list 3×3 calibration results in detail as in table 5.1 and in figure 5.4. For each level of observation errors, the calibration is performed for three times. Another 20×3 times of calibration is conducted for the statistic purpose, and the results are collected in figure 5.5.

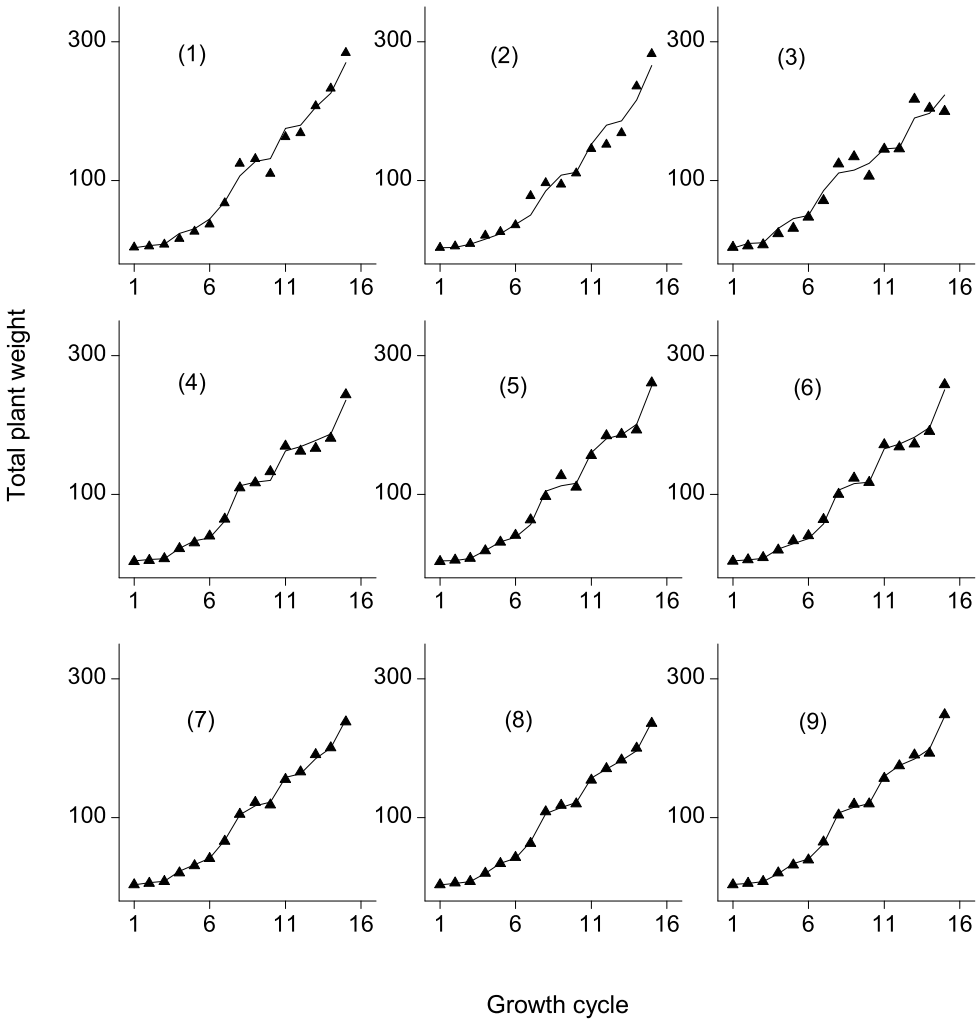


Figure 5.4: Calibration results with different levels of observation errors. Picture (1)-(9) show the target plant data marked as '▲' and fitting curves marked as '—', corresponding to those (1)-(9) listed in table 5.1.

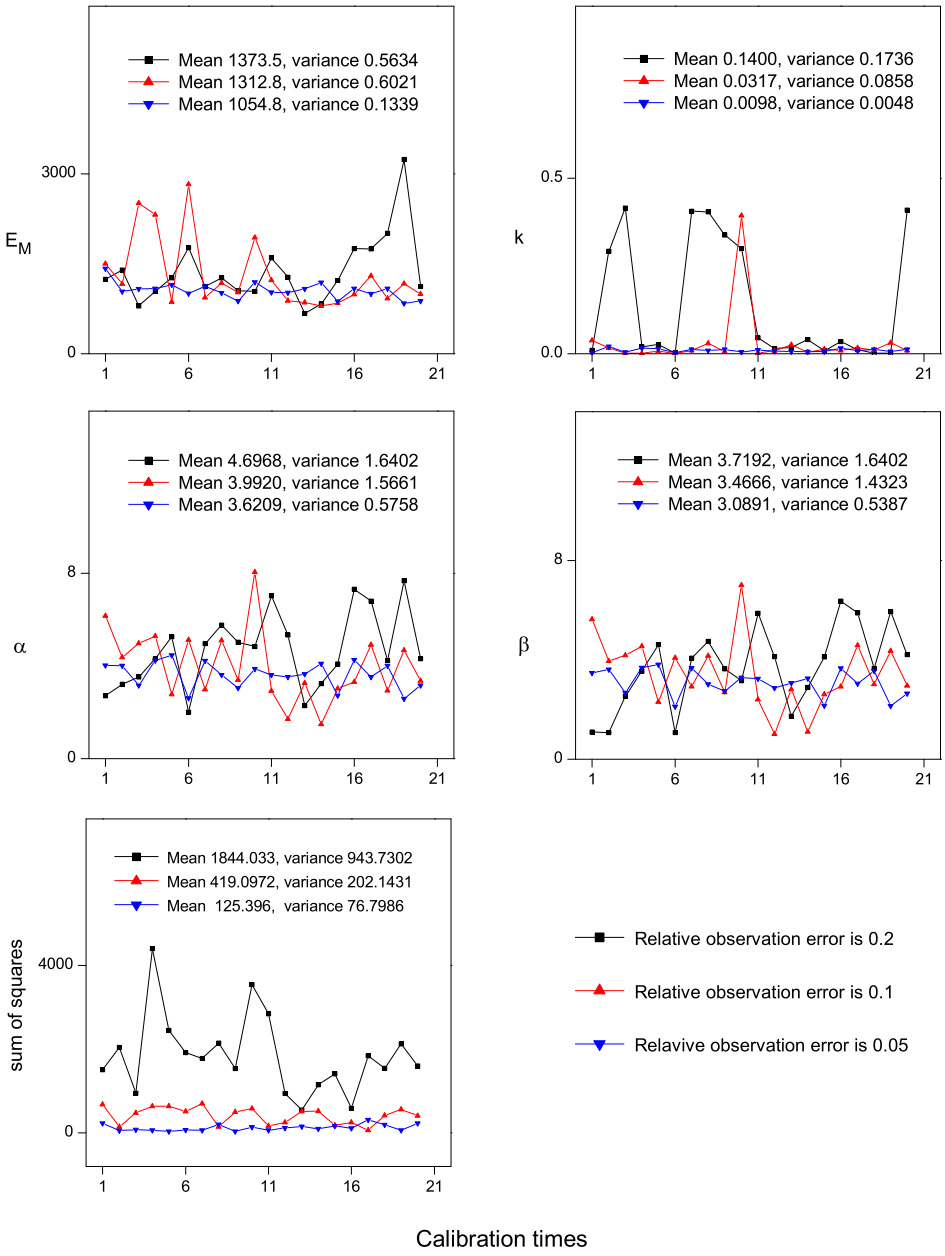


Figure 5.5: Statistics of calibration results when observation errors decrease from $\sigma = 0.2$ to $\sigma = 0.05$.

	E_M	k	α	β
(1) $\sigma = 0.2$	1223.9	0.0023	2.5379	2.1517
(2) $\sigma = 0.2$	918.5	0.0200	2.8707	3.4661
(3) $\sigma = 0.2$	1397.5	0.0031	3.3213	1.5405
(4) $\sigma = 0.1$	1554.6	0.0052	5.4313	4.5029
(5) $\sigma = 0.1$	1287.9	0.0209	4.9448	4.6905
(6) $\sigma = 0.1$	1474.0	0.0041	4.5370	4.0761
(7) $\sigma = 0.05$	1071.8	0.0039	3.0737	2.4200
(8) $\sigma = 0.05$	1017.5	0.0149	3.9046	3.3218
(9) $\sigma = 0.05$	1134.3	0.0172	4.3741	3.9201

Table 5.1: Calibration results of 9 times indexed as (1)-(9) with different levels of observation errors.

Discussion

The calibrated Ω_E^* shown in table 5.1 and figure 5.4 can be quite different from reference parameter ${}^{ref}\Omega_E$, even in the case of small observation errors ($\sigma = 0.05$), however, the statistic results in figure 5.5 show that we have better fitting results when observation error is becoming smaller, e.g. we have smaller least square residuals, and the calibrated Ω_E^* is getting closer to reference parameter ${}^{ref}\Omega_E$.

The experiment results indicate that the calibration problem for *in situ* measurements might be ill-posed, because when there is small observation error $\sigma = 0.05$, the numerical results differ dramatically for each calibration (table 5.1).

5.3.3 Numerical experiment 3: on leaf expansion effect

Design

The aim is to perform calibration when the leaf expansion lifespan t^a is long enough that environmental conditions are fluctuate during the leaf expansion period.

The leaf expansion lifespan is set to 1, 5, 10, 15 GCs respectively. The model input is chosen to be the environmental condition (a) in figure 5.2. The observation errors are considered for the calibration in case of long leaf expansion.

Results

Simulation of leaf expansion effect

Firstly we investigate the effect of leaf expansion. Under optimal environmental condition ($E(n) = 1000$), the 3D geometric plants are recorded in figure 5.6 for different leaf expansion lifespans. Then under near-periodical environmental condition (figure 5.2-a), the 3D geometric plants are recorded in figure 5.7 for different leaf expansion lifespans.

Calibration results

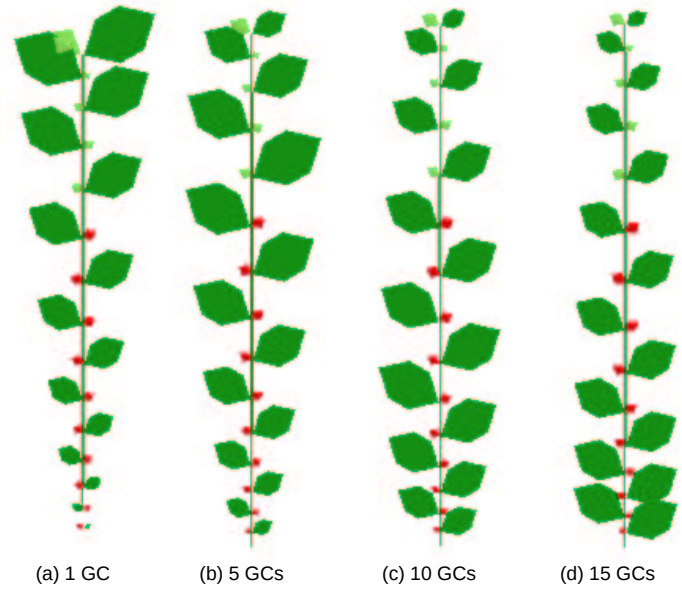


Figure 5.6: Simulations of planche26 with different leaf expansion lifespans under optimal environmental conditions

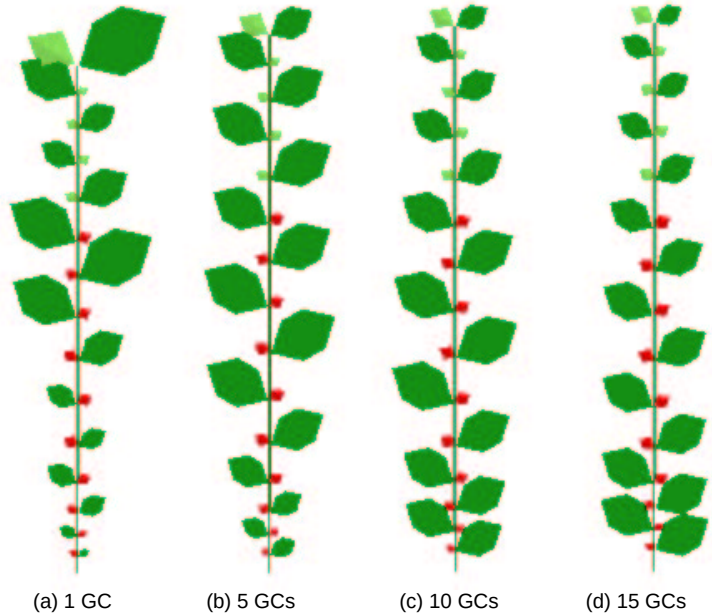


Figure 5.7: Simulations of planche26 with different leaf expansion lifespans under environmental condition (a) in figure 5.2.

In case of near-periodical environmental conditions, we conduct two times of calibration for each leaf expansion lifespan. One is the case that the observation errors are not considered, and the other is the case $\sigma = 0.2$. We list the calibration results in table 5.2 and in figure 5.8.

	E_M	k	α	β
(a) $\sigma = 0$	1000.0	0.0090	3.5000	3.0000
(a) $\sigma = 0.2$	1257.8	0.0020	3.5388	3.0140
(b) $\sigma = 0$	1000.0	0.0090	3.5000	3.0000
(b) $\sigma = 0.2$	1502.5	0.1102	5.7992	5.1158
(c) $\sigma = 0$	1000.0	0.0090	3.5000	3.0000
(c) $\sigma = 0.2$	1702.0	0.3764	6.9489	6.1964
(d) $\sigma = 0$	1000.0	0.0090	3.5000	3.0000
(d) $\sigma = 0.2$	1238.2	0.1398	4.5323	4.0436

Table 5.2: Calibration results with different leaf expansion lifespans as (a)-(d) in figure 5.7 without ($\sigma = 0$) and with observation errors ($\sigma = 0.2$) under environmental fluctuation as (a) in figure 5.2.

Discussion

When leaf expansion time is one GC, the effect of periodical environmental conditions on plant growth is clearly observed (e.g. the periodical plant geometries in figure 5.7). When leaf expansion time is long enough, the effect of environmental conditions is equalized by the expansion of leaves, and becomes vanished in (b)-(d) in figure 5.7. The calibration tracks back exactly the reference parameter values when there are no observation errors. When there are considerable observation errors, the calibration provides acceptable results (see table 5.2 and figure 5.8).

5.4 Conclusion

In this chapter, we present primary results on the calibration of environmental parameters Ω_E. The calibration process tracks back exactly reference parameter ${}^{ref}\Omega_E$ no matter the fluctuation of environment and the effect of leaf expansion. The influence of observation errors are also investigated for *GreenLab* calibration. The results of numerical experiments show that we obtain more accurate calibrated parameter values when observation errors become smaller, however, the calibration results is sensitive to the perturbation of observations. The calibration with *in situ* measurements is thus probably an ill-posed problem.

There are several issues that need further developments.

- The calibration results are achieved from numerical experiments, thus the remarks drawn in the discussion sections are rather numerical observations. Further theoretical studies are needed to verify these primary conclusions (especially for those remarks on well-posed and ill-posed problem).

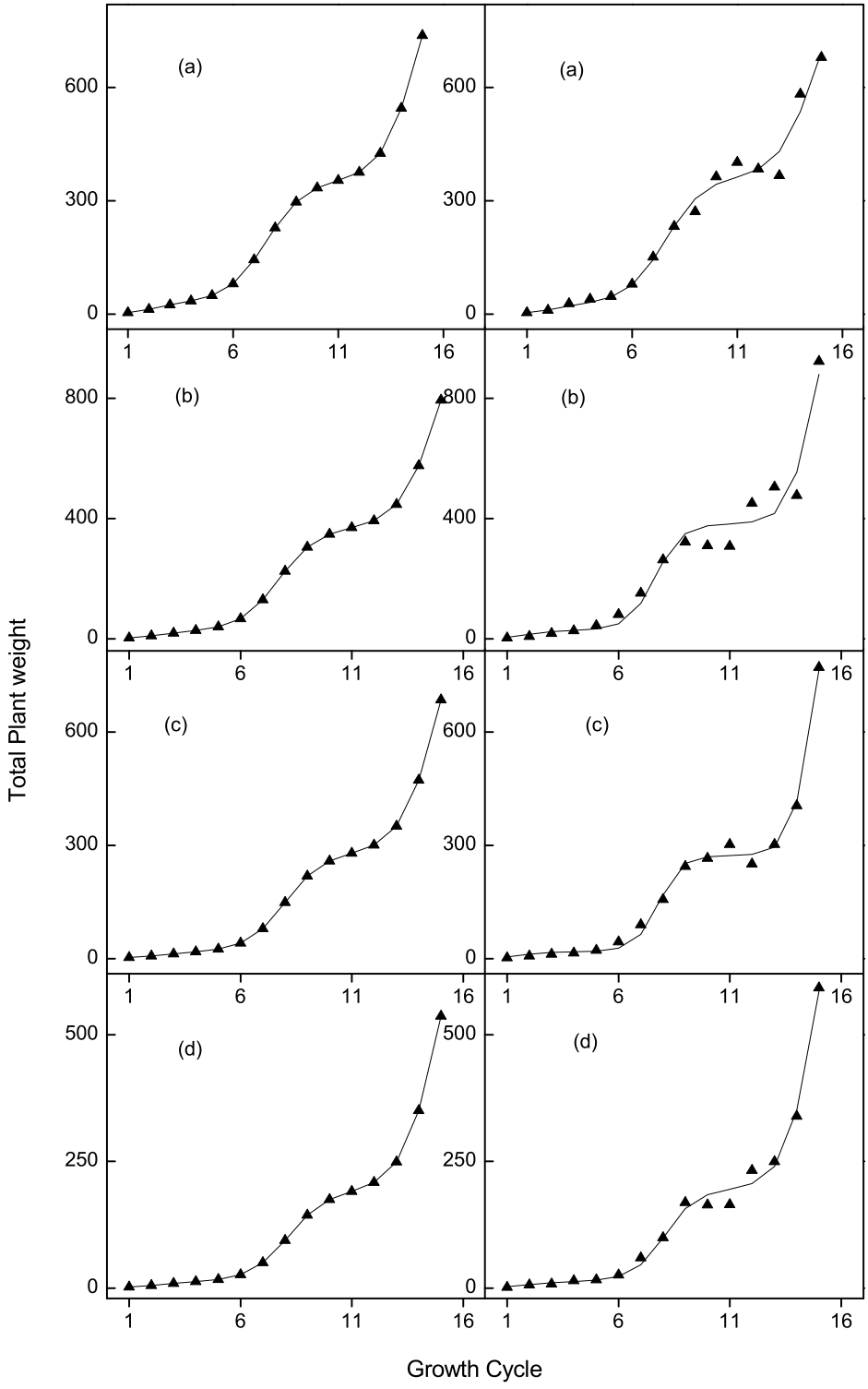


Figure 5.8: Calibration results with different leaf expansion lifespans as (a)-(d) in figure 5.7 under environmental fluctuation as (a) in figure 5.2. The left column shows the results for the calibration without observation errors, and the right column shows that with observation errors ($\sigma = 0.2$). The target plant data are marked as '▲', and '—' indicates fitting curves.

- All the numerical experiments are conducted with artificial plant target data generated by twin experiments. The treatment on *in situ* measurements is under process with the collaboration of university Wagegingen for the calibration of the environmental parameters.

- The presentation is less illustrative for variational formulism framework, since we choose finite difference scheme for the gradient calculation. The application of optimality system (5.5) is ineluctable when the parameters for calibration become increasingly enormous, as will be exemplified in the data assimilation problem in chapter 6.

Chapter 6

Variational methods applications II: optimal control and data assimilation

Résumé

Dans ce chapitre, nous analysons tout d'abord le puit optimal pour maïs, ensuite un problème de contrôle optimal pour l'approvisionnement en eau appliqué au tournesol est présenté et résolu. L'approvisionnement optimal obtenu à chaque GC fournit les irrigations plus raffinées, ce qui raméne des applications potentielles en agronomie. En considérant les incertitudes *en ligne* sur l'état initial $\mathbf{\Theta}$ et sur le modèle d'entrée $\mathbf{U}$ pour *GreenLab*, le problème d'assimilation de données est introduit pour le contrôle de ces incertitudes par assimiler les observations produites par des expériences jumelles au niveau du GC.

6.1 Introduction

Data assimilation problem **P2** and optimal control problem **P3** play important roles in the application of calibrated models.

The optimal control problem seeks for desired plant growth favorably responding to certain objective. In section 6.3 we present model analysis of optimal fruit sink strength of maize with respect to fruit weight objective. This problem is rather a univariate static optimization problem, however, the general cases of model analysis, in which parameters can be multivariate and time-variant, are in essence optimal control problems.

In section 6.4, we explicitly formulate the soil-plant dynamic system in details by defining its state variables. Section 6.5 is devoted to common issues on problem **P2-P3**, namely the adopted optimization algorithm and calibration of sunflower endogenous parameters.

Once the soil-plant dynamic system is set up, the optimal control problem for water supply can be defined. The problem is solved by adjoint model and by automatic differentiation techniques. The numerical results reveal possible agronomic applications.

The aim of data assimilation is to alleviate model uncertainties that are either site-relevant or time-relevant based on diverse sources, i.e. observations, statistics and model

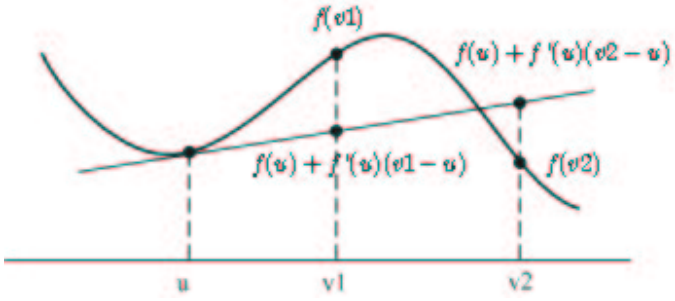


Figure 6.1: Detection of nonconvexity.

simulations. For cultivar crops such as sunflower, we can seldom control the temperature and light conditions in open fields. In this case, these two environmental factors are thus chosen to be the factors for data assimilation. It is supposed that the initial conditions contain error and need to be assimilated. In the last part of this chapter, numerical experiments are performed to show how data assimilation concepts and methods can contribute in the simulation and application of *GreenLab* model.

6.2 Non-convexity detection

Before optimization, the features of the objective function $f(x)$, say convexity, are useful. In this thesis, we are interested mainly convexity (please refer to [23] for its definition).

Theorem 6.1 *[23] Let $f : \Omega \subset \mathcal{V} \mapsto R$ a differentiable function defined in a normed vector space $\mathcal{V}$, and $\mathcal{U}$ is a convex subset of Ω, the function f is convex on $\mathcal{U}$ if and only if*

$$f(v) \geq f(u) + f'(u)(v - u), \forall u, v \in \mathcal{U} \tag{6.1}$$

When objective function f is nonconvex, there always exist points in $\mathcal{U}$ that violate equation (6.1), for example point $v2$ in figure 6.1. The following algorithm detects the function nonconvexity by evaluating N points u in $\mathcal{U}$ with M points v according to theorem 6.1.

algorithm Nonconvexity detection

Given M, N, $P = 0, i = 0$,
1. If $i > N$, calculate $\gamma = \frac{P}{M*N}$ then stop,
otherwise generate randomly point $u \in \mathcal{U}$, $i := i + 1, j = 0$.
2. If $j > M$, goto step 1, otherwise generate randomly point $v \in \mathcal{U}$, $j := j + 1$.
3. If $f(v) < f(u) + f'(u)(v - u)$, set nonconvexity flag ω to be true, that is, $\omega(i, j) = 1$, and let $P := P + 1$, otherwise $\omega(i, j) = 0$ ■

The parameter γ evaluates function nonconvexity. The function with a higher γ likely has more nonconvex points, however, convexity is not assured when $\gamma = 0$.

6.3 Optimization of source-sink relationships – a case study on maize

By model analysis, we mean mainly the mathematical analysis, e.g. the growth equilibrium characterized by, for instance, the biomass allocation trend among organs and the biomass production limit along GCs [31], or growth patterns governed by model parameters, i.e. an optimal fruit sink strength for an virtual plant that is composed of internodes, leaves, and fruits [86]. For the former, the results in [31] are obtained under optimal conditions (E is constant), and shall be renewed in the general case when E possibly varies at each GC. For the latter, it is based on the biological teleonomic assumption that the fruits undergo overwhelming growth for optimal fruit weight.

In this section, the source-sink relationship under optimal environmental conditions is investigated in order to analyze the impact of fruit sink strength on the plant biomass production and partition for an optimal yield in the case of maize [127]. The optimization problem of fruit sink strength originates from the fact that there is competition between the fruit and the leaves in demand of biomass. A theoretically optimal fruit sink strength value is determined by solving a bound-constrained optimization problem, and is named as optimal value fruit sink reference. According to the fruit sink reference we can tell the maize in the field is the optimal one or not in the sense of *GreenLab* plant functional-structural characteristics, and possible instructions for hybridization can thus be drawn.

6.3.1 Calibration results of maize

Calibration experiments are conducted in China Agriculture University (CAU) with sufficient fertilizer and water. The maize is well protected from pests and diseases; row and column spacing are distant enough to prevent competitions between maize individuals. For maize, there is no layer, no petiole but sheath for the leaves, the female flower grows to be a cob. Calibrated parameter values are stored in testfile maize.m that configures the *GreenLab* software. The calibration results of direct and hidden parameters are listed in the table

Similar to example 3, we list the information of the calibrated maize as follows,

$$N = 30, P = 1, M = \{m_{10}\}.$$

The configuration Λ is:

$$\mathbf{R} = \begin{pmatrix} 1 & 0 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 0 & 0 \end{pmatrix}, \mathbf{M_a} = \mathbf{M_f} = \mathbf{M_m} = \begin{pmatrix} 1 & 0 \end{pmatrix},$$

$$\overrightarrow{\tau} = (22), \overrightarrow{\mu} = (\bullet), \overrightarrow{\varphi^d} = \mathrm{I}, \quad \begin{cases} \varphi^f(i) = 1 & i = 16 \\ \varphi^f(i) = 0 & i \neq 16 \end{cases} \quad \begin{cases} \varphi^m(i) = 1 & i = 22 \\ \varphi^m(i) = 0 & i \neq 16 \end{cases}$$

Here the superscript f denote female flower (fruit), and m male flower. The direct parameters are

$$e = 0.0283, \tau^a = 12, t^a = t^e = 8, t^f = 17, t^m = 3.$$

The hidden parameters is shown in the following table.

Ω_G	P^a	P^e	P^c	P^f	P^m	B_a
Maize	1.0	1.89	–	201	1.49	0.47
Ω_G	B_e	B_f	B_m	r_1	r_2	
Maize	0.94	0.61	0.50	38	–	

Table 6.1: Fitting results of endogenous parameters for maize [135], The parameter C_o is fixed (to 6) for calibration purpose.

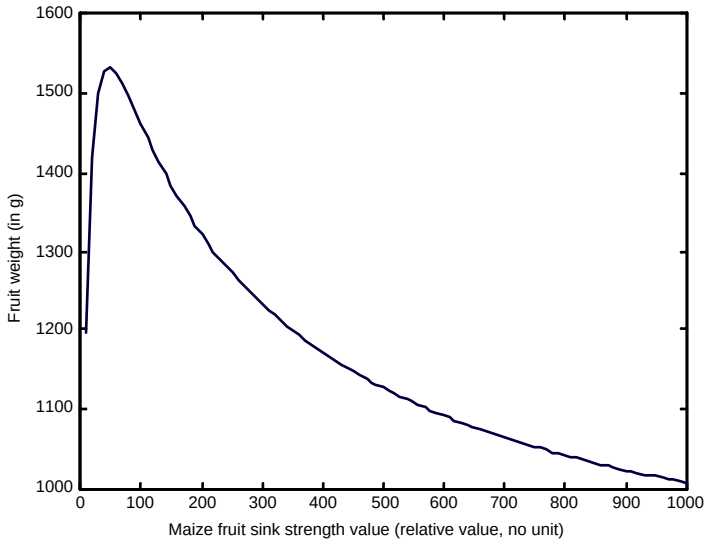


Figure 6.2: Simulations with respect to different maize fruit sink strength values

6.3.2 Optimization of fruit sink strength for maize

From the calibration results, the sink strength of female flower (fruit) is extraordinarily bigger than those of other organs. The change of fruit sink strength thus affects the maize growth profoundly. Now we mathematically set the fruit sink strength ranging from 10 to 3000 with successive increments of 10, other parameters remain the same as those calibrated ones, and the calculated fruit weight with respect to different fruit sink strength is shown in figure 6.2.

One may wonder if there exists an optimal fruit sink strength for a maximal fruit yield, and if it is possible to modify the fruit sink strength to increase the production. Supposing that fruit sink strength increases from zero, the augmental sink of fruit attracts more fresh biomass. Consequently fruit weight increases from null accordingly. On the other hand, as the biomass allocated to leaves diminishes, the ability of leaf photosynthesis lessens and thus the fresh biomass produced at the GC decreases. When fruit sink strength increases until positively infinity, there will be no fresh biomass for leaves, and maize stops growing. The fruit weight is zero in this case. Therefore mathematically speaking, there might be an optimal fruit sink strength that maximizes the fruit production. We formularize it as a static optimization problem

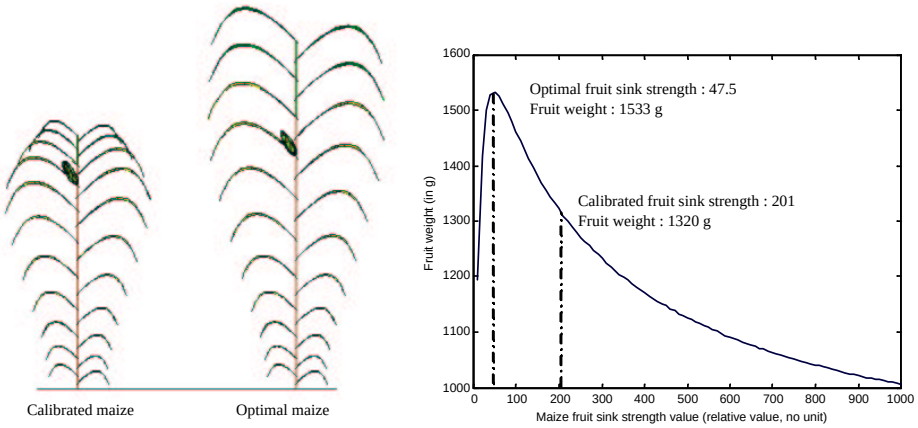


Figure 6.3: Comparison of 3D simulation results and fruit weight between theoptimal maize with the fruit apprearing at the 16^{th} internode and the calibrated one

$$\begin{aligned} &\max_{x \in R} f(x) \\ &s.t. \qquad 0 < x < 3000, \end{aligned} \tag{6.2}$$

where x is the fruit sink strength that is chosen as optimization variable and subjects to a bound constraint, objective function $f(x)$ is the *GreenLab* calculation of fruit biomass $q^o(16, 30)$ according to formula (2.59). The problem (6.2) is a typical univariate *bound-constrained optimization* one (see for instance [18] for information). We select subroutine 'fminbnd' as the optimization solver, in which golden section search is used to solve the problem (6.2).

The optimal fruit sink strength is 47.5, and the optimal fruit weight is 1533 g. The calibrated fruit sink strength is 201, and the corresponding fruit weight is 1320 g (comparison figures 6.3, 6.4). When fruit appears at GC 16 as it does, the optimal fruit weight can be 16% heavier than that of the calibrated maize. Supposing that the fruit can appear at each GC, we process the optimization process similarly. The results is listed in the right of figure 6.5. The optimal fruit appearance position is at 20^{th} GC (suppose that there are at least two internodes between male and female flowers), and the corresponding optimal fruit sink strength is 311. Its optimal fruit weight marks 1931 g and is 46% heavier than that of the calibrated maize (left of figure 6.5).

According to figure 6.2, for certain maize species we can tell it is an optimal one or not. The species of maize with high fruit sink strength and less fruit production might be valuable, when hybridizing it with a relatively vegetative maize to diminish its fruit sink strength, so that the new hybridized one can approach to the calculated optimal reference. If the computed sink strength of the observed plant is on the right of the optimal one, the optimal plant will have the drawback to produce more biomass to fulfill it. So the plant will need more water and fertilizers. Economically speaking it may be not interesting and another economical constraint has to be introduced. On the other hand in a left position the gain is obvious.

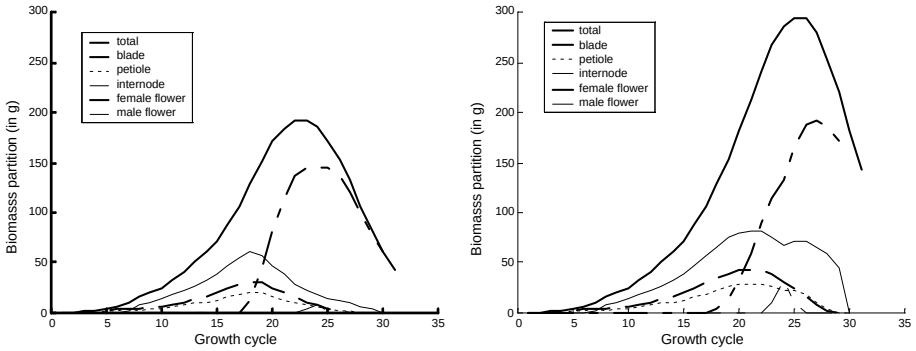


Figure 6.4: Comparison of Biomass production and partition between the optimal maize with the fuit appearing at 16^{th} internode and (right) and calibrated one (left)

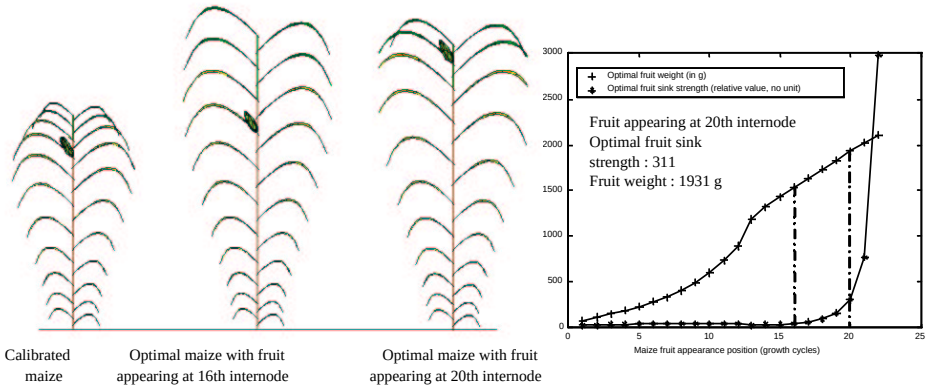


Figure 6.5: Comparison of different maize growth between the calibrated maize and theoptimal ones with the fruit appearing at 16^{th} and 20^{th} internode respectively

Note that the optimal reference depends on the form of objective function, which can also be the fruit shape or else. The optimization in this section is in fact a one-factor analysis, and the results may also depend on the other internal or external factors, which for the moment are assume to be nearly the same for all species of maize. The physiological details, e.g. the effect of leaf extension and senescence, are capsulated in the description of source-sink relationships. The concept of optimal reference is ready to be extended to multi-factor analysis by choosing the plant endogenous parameters that are evaluated to have important impacts on plant growth.

6.4 Soil-plant dynamic system

Let $\tau_w = \max_{p=1,\ldots,P} \tau_p^a$ the maximal functioning time for leaves of all physiological ages (in most cases, $P = 1, \tau_w = \tau^a$), and let state variable $\mathbf{X}(n-1), \mathbf{X}(n)$ at GC $n-1$ and n be denoted respectively as

$$\mathbf{X}(n-1) = \begin{bmatrix} Q(n-\tau_w) \\ \vdots \\ Q(n-2) \\ Q(n-1) \\ Q_w(n-1) \end{bmatrix}, \mathbf{X}(n) = \begin{bmatrix} Q(n-\tau_w+1) \\ \vdots \\ Q(n-1) \\ Q(n) \\ Q_w(n) \end{bmatrix}.$$

We rewrite the soil-plant system (2.83)-(2.85) in one difference equation

$$\begin{cases} \mathbf{X}_j(n) = \mathbf{X}_{j+1}(n-1), \qquad j = 1, \ldots, \tau_w - 1, \\ \mathbf{X}_{\tau_w}(n) = E(n-1) \cdot \dfrac{\mathbf{X}_{\tau_w+1}(n-1) - Q_{wmn}}{Q_{wmx} - Q_{wmn}} \cdot \\ \qquad \sum_{i=1}^{\tau} \dfrac{\Delta N^a(n-i) \cdot \sum_{k=1}^{i} \dfrac{\phi^a(k) \cdot \mathbf{X}_j(n-1)}{D(n-(i-k))}}{\dfrac{er_1}{P^a} + r_2 \cdot \sum_{k=1}^{i} \dfrac{\phi^a(k) \cdot \mathbf{X}_j(n-1)}{D(n-(i-k))}}, \\ \mathbf{X}_{\tau_w+1}(n) = (1 - c_1 - c_2 \cdot U(n-1))\mathbf{X}_{\tau_w+1}(n-1) + Q_{wmn}c_1 + \\ \qquad Q_{wmx}c_2 \cdot U(n-1) - \mathbf{X}_{\tau_w}(n-1), \end{cases} \tag{6.3}$$

where $E(n)$ is calculated by (2.50). The initial condition is

$$\mathbf{X}(0) = \Theta = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ Q0 \\ QW0 \end{bmatrix}, \tag{6.4}$$

where $Q0$ is seed volume, and $QW0$ is the initial soil water content.

The organ accumulated biomass $q_p^o(i,n)$ at GC n is

$$q_p^o(i,n) = P_p^o \sum_{k=1}^{i} \frac{\phi_p^o(k)\mathbf{X}_{\tau_w}(n-(i-k)-1)}{D(n-(i-k))}. \tag{6.5}$$

6.5 Related issues on problems P2 and P3

In this chapter we employ open-field sunflower as the objective plant for data assimilation problem **P2** and optimal control problem **P3**. Endogenous parameters Ω_G are calibrated, and the parameters of soil water balance and environmental parameters Ω_E are set to empirical values from previous studies. The Sequential Quadratic Programming (SQP) algorithms are adopted for the minimization of these problems.

6.5.1 Optimization algorithm for problems P2-P3

Sequential Quadratic Programming is arguably the most successful method for nonlinearly constrained problems, especially for medium-scale or small-scale ones. Also, SQP of large-scale version has been designed recently with encouraging results. The problems **P2-P3** under homogeneous environment are usually the cases of medium-scale, hence we mainly adopt SQP algorithm in this chapter. For algorithm details of SQP, please refer to section A.6 in appendix A.

6.5.2 Calibrated sunflower

The calibration experiments are similar to that of maize introduced in chapter 2. We list the information of the calibrated sunflower as follows:

$$N = 63, P = 1, M = \{m_{10}\},$$

the configuration Λ:

$$\mathbf{R} = \begin{pmatrix} 1 & 0 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 0 & 0 \end{pmatrix}, \mathbf{M_a} = \mathbf{M_f} = \begin{pmatrix} 1 & 0 \end{pmatrix},$$

$$\overrightarrow{\tau} = (38), \overrightarrow{\mu} = (\bullet), \overrightarrow{\varphi^d} = \mathrm{I}, \quad \begin{cases} \varphi^f(i) = 1 & i = 38 \\ \varphi^f(i) = 0 & i \neq 38 \end{cases}$$

Note that superscript f denote female flower (fruit). The direct parameters are

$$e = 0.043, \tau^a = t^a = t^e = t^f = 25.$$

The hidden parameters are calibrated under optimal environmental conditions ($E(n) = E_M = 1000$), and are shown in the following table

6.6 Optimal control of soil-plant dynamic system

Denote a $(\tau_w + 1)$-dimensional vector function F for the equation (6.3), we have the soil-plant dynamic system

$$\begin{cases} \mathbf{X}(n) = F(\mathbf{X}(n-1), U(n-1)), \\ \mathbf{X}(0) = \mathbf{\Theta}. \end{cases} \tag{6.6}$$

Ω_G	P^a	P^e	P^c	P^f	P^m	B_a
Sunflower	1.0	0.044	0.042	1000.0	–	0.465
Ω_G	B_e	B_f	B_m	r_1	r_2	
Sunflower	0.56	0.76	–	39246	55.6	

Table 6.2: Fitting results of genetic parameters for sunflower under optimal environmental conditions (LIAMA software, *CornerFit* [134]). The notations are as those in table 2.2. The parameter C_o is fixed (to 7) for calibration purpose.

Considering the organ biomass formulae of increment (2.58) and accumulation (2.59), one can naturally devise the objective index I as the sum of some scalar-valued function l:

$$\max I = q^f(n,n) = \sum_{i=0}^{n-1} \Delta q^f(i,n) = \sum_{i=0}^{n-1} l(\mathbf{X}(i), U(i)). \tag{6.7}$$

The dynamic system (6.6) and the objective function of fruit yield (6.7) form the optimal control system for water supply.

6.6.1 Consideration of water resource constraint

In numerous cases, water resources are limited because of drought or economic reasons. For a given total quantity of water supply Δ, the water supply amount $U(i)$ at GC i is under the following linear constraint

$$\sum_{i=0}^{n-1} U(i) = \Delta, \tag{6.8}$$

and boundary constraint

$$0 \le U(i) \le \Delta, \qquad i = 0, \ldots, n-1. \tag{6.9}$$

The fruit yield index will depend on the irrigation strategy during the plant growth. In this chapter the optimal control of the plant water supply is always under the resource constraint (6.8)-(6.9), however, the constraint can be treated by standard constrained optimization algorithms, i.e. sequential quadratic programming introduced in the previous section, therefore we do not formulate it explicitly in the following sections. The rest sections deal mainly the calculation of gradient $\nabla_{\mathbf{U}} I$.

6.6.2 Function convexity

Given initial condition $\mathbf{\Theta}$, applying dynamic equation (6.6) recursively, we have

$$\begin{aligned} \mathbf{X}(1) = \Phi_1(\mathbf{U}) &= F(\mathbf{X}(0), U(0)) \\ \mathbf{X}(2) = \Phi_2(\mathbf{U}) &= F(F(\mathbf{X}(0), U(0)), U(1)) \\ \vdots &= \vdots \\ \mathbf{X}(n) = \Phi_n(\mathbf{U}) &= F(\ldots F(\mathbf{X}(0), U(0)), \ldots, U(n-2)), U(n-1)) \end{aligned} \tag{6.10}$$

where $\mathbf{U} = [U(0) \ldots U(n-1)]^T$. Substitute (6.10) into (6.7), we have the objective index

$$\begin{pmatrix}
0&1&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&1\\
0&0&0&1&0&1&1&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&0&0&1&0&0&0&1&0&1&0&0&1&1&1&0&1&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&1&1&0&1&0&1&1&1&1&1&1&1&0&1&0\\
0&1&1&0&1&1&1&1&0&1&1&1&0&1&1&0&1&1&1&0\\
1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0\\
1&1&1&0&0&1&0&1&1&1&1&1&1&0&1&1&0&0&1&1\\
0&0&0&0&1&1&1&0&0&0&1&0&0&0&0&1&1&1&0&0\\
1&0&0&1&1&1&1&1&0&1&1&1&1&0&1&0&0&1&1&0\\
0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&1&0&1&0&1&0&1&0&1&1&0&0&0\\
0&0&1&0&1&1&0&0&0&1&0&0&0&1&1&1&1&0&0&1\\
1&0&0&1&0&1&1&1&1&0&1&1&0&1&0&1&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{pmatrix}_{N\times M}$$

Figure 6.6: Matrix of convexity flag $[\omega_{ij}]_{1\leq i\leq N,1\leq j\leq M}$. The function is nonconvex, and $\gamma = 0.3175$.

$$I = \mathcal{J}(\mathbf{U}). \tag{6.11}$$

The optimal control problem (6.6)-(6.7) then equals to static optimization problem (6.11).

The convexity of function (6.11) can be evaluated by algorithm presented in section 6.2. We set $N = 20, M = 20$, and the test results are shown in figure 6.6. The function (6.11) is nonconvex. Note that subset $\mathcal{U}$ is determined by constraints (6.8) and (6.9).

6.6.3 Optimality system

Recall the results of the variational formulation of optimality system for optimal control

$$\begin{cases}
\frac{d\mathbf{X}(t)}{dt} &= F\left(\mathbf{X}(t), \mathbf{U}(t)\right), \\
\mathbf{X}(T_1) &= \Theta, \\
\frac{dW}{dt} &= -\left[\frac{\partial F}{\partial \mathbf{X}}\right]^T W + \left[\frac{\partial G}{\partial \mathbf{X}}\right]^T, \\
W\left(T_2\right) &= 0, \\
I &= \int_{T_1}^{T_2} G\left(\mathbf{X}\left(t\right), \mathbf{U}\left(t\right)\right) dt,
\end{cases} \tag{6.12}$$

with its gradient

$$\nabla_{\mathbf{U}} I = -\left[\frac{\partial F}{\partial \mathbf{U}}\right]^T W + \left[\frac{\partial G}{\partial \mathbf{U}}\right]^T, \tag{6.13}$$

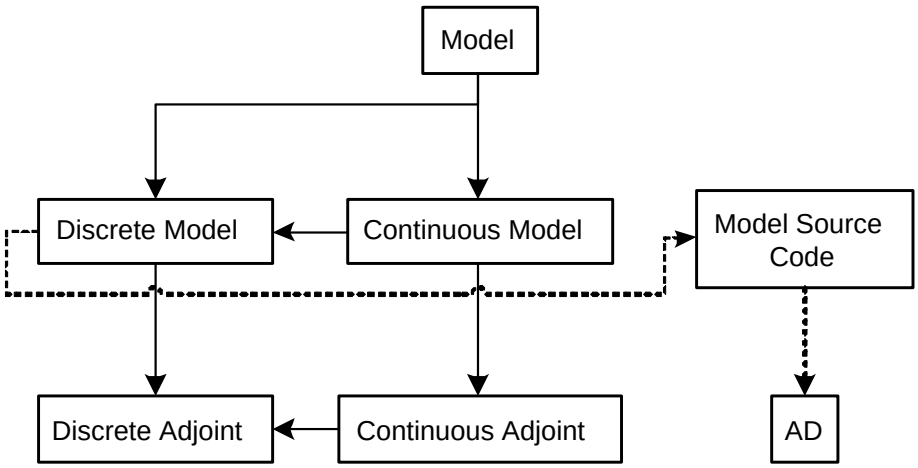


Figure 6.7: Approaches for gradient calculation of the optimal control system using adjoint computation and AD techniques.

where G is the corresponding continuous form of l in discrete objective function (6.7). Note that we use the same notation F as that of discrete form (6.6), as incurs no confusion since they work parellelly in their own contexts.

6.6.4 Numerical solutions

Direct methods that solve the optimality system (6.12) attempt to convert the infinite optimal control system into a finite one by discretization schemes. The discretization results construct nonlinear programming problems for which numerical optimization algorithms, such as SQP, are employed to locate the optimal values of control variable $\mathbf{U}$.

A key issue for these optimization algorithms is the calculation of gradient $\nabla_{\mathbf{U}} I$ for objective function I. Several approaches have been experienced, which are clearly presented in a recent article [51]. We summarize these diverse approaches in figure 6.7.

- At first sight, one may expect that a discretization of the continuous adjoint equation in (6.12) together with an evaluation of gradient formula (6.13) provides the gradient information. Unfortunately this approach does not always produce consistent calculation of gradients. Special cautions, e.g. a suitable integration scheme for the continuous adjoint equation, are needed to avoid the inconsistency, which usually appear to be nontrivial tasks (for more information, see section 3.3 in [51]).
- An alternative approach is directly based on discrete adjoint model derived from the discrete model. The process is essentially to apply the chain rule recursively, therefore, the gradients calculated in this method are exact.
- Finally, AD techniques undergo systematically the chain rule, and provide consistent gradients given program code of the discrete model.

The last two approaches are supposed to provide equivalent gradients. The theoretical analysis of their equivalent relation is presented in section 4.2.4 with observation discrepancy as the objective function.

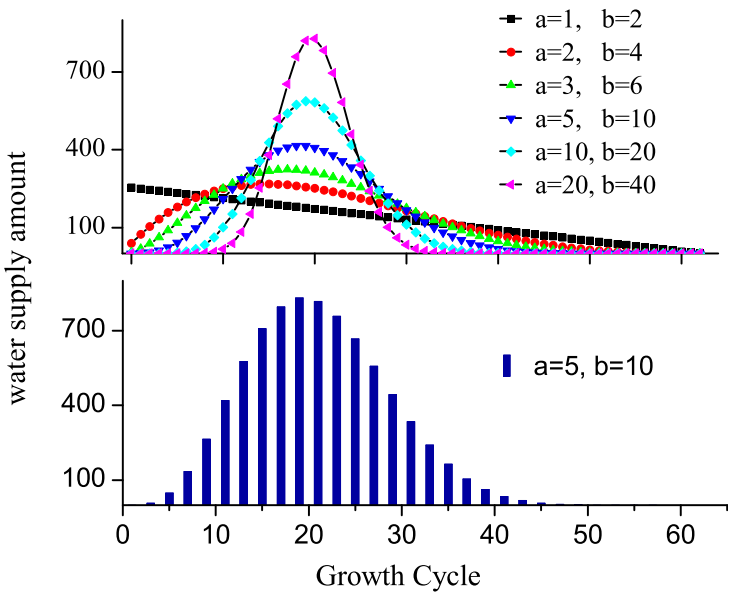


Figure 6.8: Water supply curves governed by beta function parameters a, b, supply frequency ω and irrigation initial date γ. For the upper curves, $\omega = 1, \gamma = 1$; in the lower histogram, $\omega = 2, \gamma = 1$.

Since *GreenLab* is already a discrete model (6.6), we will evaluate the discrete adjoint computation and AD techniques to solve the optimal control problem (6.12). We start the numerical solution with a simplified case in which the optimal control system is reduced into a mixed integer nonlinear programming problem (MINLP).

MINLP problem for simplified water supply

In order to alleviate calculation load and investigate rhythmic irrigation, instead of optimizing the water supply at each GC, the control variables are reduced into four parameters $[a, b, \omega, \gamma]$ by approximating water supply curves as a cluster of beta functions with a, b the function parameters and with ω, γ the water supply frequency and the irrigation initial date respectively. The water supply $U(i)$ at GC i is thus

$$U(i) = \frac{\Delta}{S}\left(\frac{i-0.5}{N}\right)^{a-1}\left(1-\frac{i-0.5}{N}\right)^{b-1},$$

where the total water supply Δ is set to 8000 units, and S is a normalization factor of beta function:

$$S = \sum_{i=1}^{N}\left(\frac{i-0.5}{N}\right)^{a-1}\left(1-\frac{i-0.5}{N}\right)^{b-1}.$$

Considering frequency ω and initial irrigation date γ, the practical water supply is redistributed at GC i, with $i = j \times \omega + \gamma$, $j \in \mathrm{N}$ and $1 \leq i \leq N$. For the other GCs, the amount of water supply is set to zero (see figure 6.8). Then a mixed integer nonlinear programming problem **P4** (MINLP) is formulated as

$$\begin{aligned} &\max_{x\in \mathrm{R}^4} f(x) \\ &\text{subject to,} \quad 1 \le x_1, x_2 \le 100, \\ &\qquad\qquad\quad 1 \le x_3, x_4 \le N, \end{aligned}$$

where $x = [x_1, x_2, x_3, x_4]^T = [a, b, \omega, \gamma]^T$, the bounds for x_1, x_2 ensures a sufficient amount of distribution curves, and N is the sunflower total number of GC.

Genetic algorithms (GA) is employed to solve the static optimization problem **P4** [128]. We list the optimization results of 3 tries as table 6.3.

Try Number	Appearing Generation	Optimal parameter values				Optimal fruit weight (g)
		a	b	γ	ω	
1	110	1.3686	1.1934	1	2	1196.3
2	92	1.3263	1.1632	1	2	1196.6
3	108	1.3263	1.1571	1	2	1196.6

Table 6.3: Optimization results of MINLP GA solver

And we plot the water supply strategy of the second try, which is found at the 92th generation of GA evolution in figure 6.10-(a).

Optimal control approaches: adjoint computation of Lagrange problem

Optimal control system whose objective function is an integral term as shown in (6.12) is cast into Lagrange problem [13]. System (6.6)-(6.7) is a Lagrange problem in discrete form. In this section, we will investigate the second approach addressed in section 6.6.4. The adjoint model is derived directly from the discrete model of the optimal control problem. Note that the derivation of adjoint model in continuous form is addressed in chapter 3.

Perturbating dynamic model (6.6) and cost function (6.7) with respect to $\mathbf{U}$, we have

$$\begin{cases} \hat{\mathbf{X}}_{n+1} = \dfrac{\partial F}{\partial \mathbf{X}}(X_n, \mathbf{U}_n)\hat{\mathbf{X}}_n + \dfrac{\partial F}{\partial \mathbf{U}}(\mathbf{X}_n, \mathbf{U}_n)\hat{\mathbf{U}}_n, \\ \hat{\mathbf{X}}_0 = 0, \end{cases} \tag{6.14}$$

and

$$\begin{aligned} \hat{I} &= I(\mathbf{U}+\hat{\mathbf{U}}) - I(\mathbf{U}) \\ &= \sum_{n=0}^{N-1} \left(\frac{\partial l}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\,\hat{\mathbf{X}}_n + \frac{\partial l}{\partial \mathbf{U}}(\mathbf{X}_n, \mathbf{U}_n)\,\hat{\mathbf{U}}_n \right) \\ &= \underbrace{\sum_{n=0}^{N-1} \left(\frac{\partial l}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\,\hat{\mathbf{X}}_n \right)}_{\hat{I}_1} + \underbrace{\sum_{n=0}^{N-1} \left(\frac{\partial l}{\partial \mathbf{U}}(\mathbf{X}_n, \mathbf{U}_n)\,\hat{\mathbf{U}}_n \right)}_{\hat{I}_2}. \end{aligned} \tag{6.15}$$

Let $M^{m\times p}$ denotes matrix with m rows and p columns, we have

$$\frac{\partial F}{\partial \mathbf{X}} \in M^{m\times m}, \hat{\mathbf{X}}_n \in M^{m\times 1}; \quad \frac{\partial F}{\partial \mathbf{U}} \in M^{m\times p}, \hat{\mathbf{U}}_n \in M^{p\times 1}; \quad \frac{\partial l}{\partial \mathbf{X}} \in M^{1\times m}, \frac{\partial l}{\partial \mathbf{U}} \in M^{1\times p}.$$

Herein $\mathbf{X}(n)$ is abbreviated as $\mathbf{X}_n$, $\mathbf{U}_n$ for $\mathbf{U}(n)$ and so on, control variable $\mathbf{U}_n$ is denoted as a $p-$dimensional vector in the general case, $\mathbf{U}$ is thus a vector of Np dimensions, the Jacobian matrix $\frac{\partial F}{\partial \mathbf{X}} = [a_{ij}]_{m\times m}$, a_{ij} is the partial derivative of i-th function of F with respect to j-th variable of $\mathbf{X}$.

Introducing adjoint variable $\bar{\mathbf{X}}_n$, we evaluate the scalar product in space $\underbrace{R^m \times \ldots \times R^m}_{N+1}$ as

$$\begin{aligned}
&\sum_{n=0}^{N} [\bar{\mathbf{X}}_n]^T \hat{\mathbf{X}}_n \\
= \;& \left[\bar{\mathbf{X}}_0\right]^T \hat{\mathbf{X}}_0 + \\
& \sum_{n=0}^{N-1} [\bar{\mathbf{X}}_{n+1}]^T \left[\frac{\partial F}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\, \hat{\mathbf{X}}_n + \frac{\partial F}{\partial \mathbf{U}}(\mathbf{X}_n, \mathbf{U}_n)\, \hat{\mathbf{U}}_n\right] \\
= \;& \left[\bar{\mathbf{X}}_0\right]^T \hat{\mathbf{X}}_0 + \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{X}}_n \right\} + \\
& \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{U}}(\mathbf{X}_n, \mathbf{U}_n)\right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{U}}_n \right\}.
\end{aligned} \tag{6.16}$$

Imposing that the adjoint variable $\bar{\mathbf{X}}_n$ satisfies

$$\begin{cases} \bar{\mathbf{X}}_n = \left[\frac{\partial F}{\partial \mathbf{X}}(\mathbf{X}_{n+1}, \mathbf{U}_{n+1})\right]^T \bar{\mathbf{X}}_{n+1} + \left[\frac{\partial l}{\partial \mathbf{X}}(\mathbf{X}_{n+1}, \mathbf{U}_{n+1})\right]^T, n = 0, \ldots, N-1, \\ \bar{\mathbf{X}}_N = 0, \end{cases} \tag{6.17}$$

the scalar product (6.16) becomes,

$$\begin{aligned}
& [\bar{\mathbf{X}}_N]^T \hat{\mathbf{X}}_N + \sum_{n=0}^{N-1} [\bar{\mathbf{X}}_n]^T \hat{\mathbf{X}}_n \\
= \;& \left[\bar{\mathbf{X}}_0\right]^T \hat{\mathbf{X}}_0 + \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{X}}_n \right\} + \\
& \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{U}}(\mathbf{X}_n, \mathbf{U}_n)\right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{U}}_n \right\} + \\
& \sum_{n=0}^{N-1} \left\{ \left[\frac{\partial l}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\right] \hat{\mathbf{X}}_n \right\} - \\
& \sum_{n=0}^{N-1} \left\{ \left[\frac{\partial l}{\partial \mathbf{X}}(\mathbf{X}_n, \mathbf{U}_n)\right] \hat{\mathbf{X}}_n \right\}.
\end{aligned} \tag{6.18}$$

Note that $\bar{\mathbf{X}}_N = 0$, and $\hat{\mathbf{X}}_0 = 0$,

$$
\begin{aligned}
& \sum_{n=0}^{N-1} \left\{ \left[\frac{\partial l}{\partial \mathbf{X}} (\mathbf{X}_n, \mathbf{U}_n) \right] \hat{\mathbf{X}}_n \right\} \\
= & \sum_{n=0}^{N-1} \left[-\bar{\mathbf{X}}_n + \left[\frac{\partial F}{\partial \mathbf{X}} (\mathbf{X}_n, \mathbf{U}_n) \right]^T \bar{\mathbf{X}}_{n+1} + \left[\frac{\partial l}{\partial \mathbf{X}} (\mathbf{X}_n, \mathbf{U}_n) \right]^T \right]^T \hat{\mathbf{X}}_n + \\
& \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_n, \mathbf{U}_n) \right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{U}}_n \right\}.
\end{aligned} \tag{6.19}
$$

Substitute adjoint model (6.17) into (6.19), we have

$$
\sum_{n=0}^{N-1} \left\{ \left[\frac{\partial l}{\partial \mathbf{X}} (\mathbf{X}_n, \mathbf{U}_n) \right] \hat{\mathbf{X}}_n \right\} = \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_n, \mathbf{U}_n) \right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{U}}_n \right\}. \tag{6.20}
$$

Substitute (6.20) into (6.15), and recall definition 3.2,

$$
\begin{aligned}
& \left\langle \nabla_{\mathbf{U}} I, \hat{\mathbf{U}} \right\rangle \\
= & \sum_{n=0}^{N-1} \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_n, \mathbf{U}_n) \right]^T \bar{\mathbf{X}}_{n+1} \right]^T \hat{\mathbf{U}}_n \right\} + \sum_{n=0}^{N-1} \left(\frac{\partial l}{\partial \mathbf{U}} (\mathbf{X}_n, \mathbf{U}_n) \hat{\mathbf{U}}_n \right) \\
= & \sum_{n=0}^{N-1} \left\{ \left\{ \left[\left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_n, \mathbf{U}_n) \right]^T \bar{\mathbf{X}}_{n+1} \right]^T + \frac{\partial l}{\partial \mathbf{U}} (\mathbf{X}_n, \mathbf{U}_n) \right\} \hat{\mathbf{U}}_n \right\} \\
= & \left\langle \begin{bmatrix} \left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_0, \mathbf{U}_0) \right]^T \bar{\mathbf{X}}_1 + \left[\frac{\partial l}{\partial \mathbf{U}} (\mathbf{X}_0, \mathbf{U}_0) \right]^T \\ \vdots \\ \left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_{N-1}, \mathbf{U}_{N-1}) \right]^T \bar{\mathbf{X}}_N + \left[\frac{\partial l}{\partial \mathbf{U}} (\mathbf{X}_{N-1}, \mathbf{U}_{N-1}) \right]^T \end{bmatrix}, \begin{bmatrix} \hat{\mathbf{U}}_0 \\ \vdots \\ \hat{\mathbf{U}}_{N-1} \end{bmatrix} \right\rangle.
\end{aligned} \tag{6.21}
$$

Hence the gradient $\nabla_{\mathbf{U}} I$ is

$$
\nabla_{\mathbf{U}} I = \begin{bmatrix} \left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_0, \mathbf{U}_0) \right]^T \bar{\mathbf{X}}_1 + \left[\frac{\partial l}{\partial \mathbf{U}} (\mathbf{X}_0, \mathbf{U}_0) \right]^T \\ \vdots \\ \left[\frac{\partial F}{\partial \mathbf{U}} (\mathbf{X}_{N-1}, \mathbf{U}_{N-1}) \right]^T \bar{\mathbf{X}}_N + \left[\frac{\partial l}{\partial \mathbf{U}} (\mathbf{X}_{N-1}, \mathbf{U}_{N-1}) \right]^T \end{bmatrix}. \tag{6.22}
$$

Note that gradient $\nabla_{\mathbf{U}} I$ is a vector with $N \cdot p$ dimensions.

Numerical results In order to apply the theoretical results, we rewrite *MiniGreenLab* in the Lagrange problem: state variable $\mathbf{X}$ is defined; function F of model dynamics is implemented; and objective function l is calculated according to formula (2.59). Partial derivatives, such as $\frac{\partial F}{\partial \mathbf{X}}$, $\frac{\partial F}{\partial \mathbf{U}}$, $\frac{\partial l}{\partial \mathbf{X}}$, and $\frac{\partial l}{\partial \mathbf{U}}$, are either achieved by symbolic differentiation, or approximated by finite difference scheme. The gradients are obtained by solving the

adjoint equation (6.17) and by calculating the formula (6.22). With these gradient calculations, the optimization methods, i.e. SQP algorithm, provide the numerical solution of optimal water supply $\mathbf{U}_i, i = 0, \ldots, N-1$.

The Lagrange problem is a classic issue in optimal control theory, which is proved to be equivalent to Meyer and Bolza problems [25]. For previous versions implemented in Scilab, we adapt *MiniGreenLab* for the purpose of applying Scilab toolbox dynoptim [1], in which theoretical results are obtained by algebraic derivation. These results are exactly the same as what we have formulated in this section based on perturbation analysis within variational formulism framework. The latest version of *GreenLab* is written in MATLAB, therefore we re-implement toolbox dynoptim in MATLAB. SQP algorithms are adopted to numerically solve the optimization problems in the context of optimal control. We plot the water supply strategy found by the optimization solver in figure 6.10-(b) with initial conditions set to the optimization results of problem **P4** that are shown in figure 6.10-(a).

Optimal control approaches: AD techniques

In this approach, AD techniques introduced in chapter 4 are employed for an efficient and accurate gradient calculation, which we provide to standard optimization algorithms, i.e. SQP method implemented in subroutine 'fmincon'. Finally the optimization results illustrate how different water supply strategies effect the sunflower growth.

Following the instructions in chapter 4 we derive the adjoint code for the sunflower water supply problem by hand. The size of the optimal control problem is not so large, because homogenous environmental conditions within canopy are assumed and recursive formula (2.64) is implemented for the purpose of simulation efficiency. In our AD coding, the safeguards of variables values in direct code are conducted whenever these variables change their values. When problem size becomes extremely enormous, some of the variables have to be recalculated in case of value changing, and safeguard strategies must be carefully designed to save the cost of computer memory.

In section 4.2.4, we illustrate the equivalence between discrete adjoint model (4.61) and AD differentiation rule in matrix form (4.34). Supposing that program code implements direct model similar to formula (4.63), the equivalence between some discrete adjoint model and AD differentiation rule is expected in the case of optimal control problem (6.12). Note that the adjoint variable in the context of AD is defined in (4.6) or (4.24) as the sensitivity of model output with respect to intermediate variables in series (4.11). The gradient of the optimal control problem is obtained by AD techniques as the output of adjoint code according to (4.25), instead of the calculation of complex formula (6.22) in the context of discrete adjoint computation.

Verification of adjoint code and gradient calculation The verification of adjoint code follows the process in section 4.3.3. We write the direct code of *MiniGreenLab* as

$$[\mathsf{y}] = \mathsf{sim\_U}(\mathsf{x}),$$

[1] Discrete time optimal control problems toolbox, S. Berthaud, J.P Chancelier, M. Cohen de Lara http://www.enpc.fr/cereve/HomePages/mcdl/scilab/.

where x $\in R^{63}$ signifies the water supply **U** during sunflower growth, y $\in R$ is the calculated fruit yield of calibrated sunflower. The tangent linear code is encapsulated into subroutine

$$[\mathsf{y},\ \mathsf{ytg}] = \mathsf{sim\_U\_tg}(\mathsf{x},\ \mathsf{xtg}),$$

and let adjoint code denoted

$$[\mathsf{y},\ \mathsf{xad}] = \mathsf{sim\_U\_ad}(\mathsf{x},\ \mathsf{yad}).$$

Exact gradient can be provided by either forward or reverse mode of AD. For the former, $\frac{\partial y}{\partial x_i}\Big|_x$, the $i-$th component of gradient at x, is given by the output ytg of subroutine sim_U_tg with its input xtg set to cardinal unit vector e_i. For the latter, the gradient is obtained by one time evaluation of subroutine sim_U_ad with its input yad set to 1, that is, $\nabla_x y =$xad.

We chose x randomly for three times (denoted as $x^{(i)}, i = 1, 2, 3$ respectively) to verify the adjoint code. For each x, we process as follows for three times with random h.

- Generate perturbation h randomly.
- Run [y, u] = sim_U_tg(x, h), and record u.
- Evaluate scalar product $u^T u$.
- Given u, run [y, v] = sim_U_ad(x, u), and record the result v.
- Evaluate scalar product $h^T v$.
- Evaluate the identity $u^T u \equiv h^T v$, and record the difference $u^T u - h^T v$.

We list the results in table 6.4, and the number of significant figures for each calculation item is chosen to be 16. We observe that the difference item are much more likely negative, there are, however, positive ones.

For $x^{(1)}$	$\|h\|$	$u^T u$	$h^T v$	$y^T y - h^T \bar{x}$
1	100	5.06554034720038	5.06554034720039	-6.217248937900877e-015
2	1000	1700.689597719033	1700.689597719032	9.094947017729282e-013
3	10000	266950.7205866337	266950.7205866338	-1.164153218269348e-010
For $x^{(2)}$	$\|h\|$	$u^T u$	$h^T v$	$y^T y - h^T \bar{x}$
1	100	1.57531769520953	1.57531769520953	-2.886579864025407e-015
2	1000	155.8982673378681	155.8982673378684	4.547473508864641e-013
3	10000	16720.48640131183	16720.48640131185	-1.455191522836685e-011
For $x^{(3)}$	$\|h\|$	$u^T u$	$h^T v$	$y^T y - h^T \bar{x}$
1	100	1.24608097853649	1.24608097853649	2.220446049250313e-016
2	1000	74.31515217887814	74.31515217887824	-9.947598300641403e-014
3	10000	8516.538458226190	8516.538458226187	3.637978807091713e-012

Table 6.4: Validation of adjoint code for the optimal control problem

Validation results on gradient is examined according to formula (4.70). We list the value of $f(\alpha)$ varying α from 10^0 to 10^{-17} in table 6.5. In order to highlight the oscillation when perturbations are small enough to be comparable with machine relative accuracy of floating point (in our case about 10^{-16}), we refine the scale of α from 2^0 to 2^{-50} and plot the curve of $f(\alpha)$ in figure 6.9.

α	$f(\alpha)$	α	$f(\alpha)$
10^0	0.45854188773616	10^{-9}	1.00000002198355
10^{-1}	0.91891387122884	10^{-10}	1.00002152871544
10^{-2}	0.99137248272000	10^{-11}	1.00032715069494
10^{-3}	0.99913171463553	10^{-12}	1.00100631064940
10^{-4}	0.99991311577400	10^{-13}	0.99232815567581
10^{-5}	0.99999131115344	10^{-14}	1.01873993168239
10^{-6}	0.99999913266132	10^{-15}	0.75462217161658
10^{-7}	0.99999991633645	10^{-16}	3.77311085808292
10^{-8}	1.00000002198355	10^{-17}	0.00000000000000

Table 6.5: Validation of gradient calculation, where $\|h\| = 1000$. Gradients obtained by finite difference are acceptable only within certain range of perturbation αh, i.e. $\alpha \in [10^{-4}, 10^{-10}]$ for accuracy of 4 decimal places.

Optimization results The optimization results using AD techniques are compared in figure 6.10 (c)-(e). There are many local optima depended on the initial conditions. It seems that the irrigation rhythm is important for fruit biomass accumulation. In the early stage from breeding to approximately 15-20 GC, the strategies (c)-(e) illustrate an irrigation frequency about 4 or 5 GC, and then follow an irrigation rhythm of every two GCs. All the four strategies indicate that from 58 GC to the end of the growth, there is no need of irrigation. Furthermore it seems that the fruit biomass is less sensitive to the irrigation amount distributed at each GC. There is only maximal 9 g difference of the fruit biomass among these optima, but the irrigation amount at each GC of (c)-(e) can differ dramatically. We can see roughly in (c) and (e) the amount peaks every 9 and 6 GC respectively from 31 GC to 52 GC. The iterations of optimization process (c)-(e) are shown in figure 6.11 and in figure 6.12.

Different water supply strategies and their resulting fruits are compared in figure 6.13. 3D plant geometries are also calculated and compared. The fruit yield (I) and sunflower height (H) are quite different for water distribution strategies (a), (b), (c). With the optimal strategy (c), the fruit is 18% heavier than with strategy (b) and the plant is 15% higher. It is interesting to note that for strategy (b), the fruit is 51% heavier than that of strategy (a), but the plant is shorter. It is due to the abundance of water supply of strategy (a) at early GCs that favors internode growth, and deficient water supply later on when the fruit appearances.

For strategy (c)-(e) in figure 6.10, we replace water supply at each GC by the mean of irrigation amounts within certain period and plot smoothed water supply strategies in figure 6.14. We aim at a statistic analysis over certain stage (mean every 8, 12 and 16 GCs

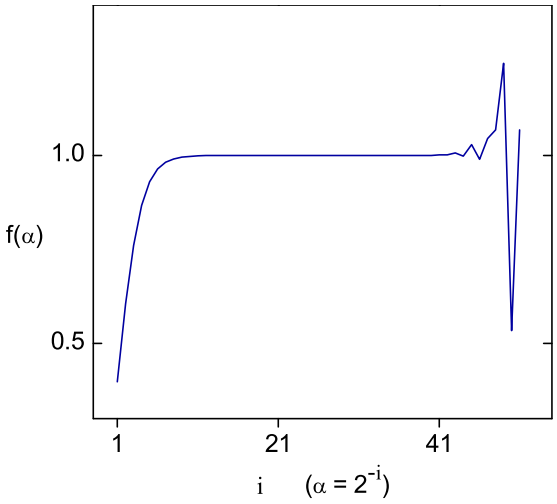


Figure 6.9: Validation of gradient calculation, where $\|h\| = 1000$. For finite difference scheme, coefficient α of acceptable perturbations αh with accuracy of 4 decimal places ranges from 2^{-19} to 2^{-42} ($[2^{-19}, 2^{-42}] \approx [10^{-6}, 10^{-13}]$, note that h randomly differs from that in table 6.5). At the left side of the curve, we have truncation errors in gradient approximation of finite difference since the perturbations αh are big enough compared with x; the oscillation arises at the right side of the curve, because the perturbations are small enough compared with the machine relative accuracy of floating point, and we fall across round-off errors.

respectively) of sunflower development for all these strategies. The comparison results show that these strategies share common statistical properties.

6.6.5 Discussion

On AD techniques and adjoint computation

The derivation of adjoint equation depends on the form of objective function. Once changed, i.e. from fruit yield to observation discrepancy, the derivation has to be repeated completely, as well as the software implementation. Usually the discrete adjoint is as complex as its direct model. In addition, the derivation of discrete adjoint equation *analytically* by hand is error-prone. Whereas in AD approach, this tedious derivation of adjoint equation is left to AD software for automatical process.

Some Jacobian matrices in adjoint equation are obtained by Finite Difference schemes (FD), i.e. $\frac{\partial F}{\partial X}$. Hence the two approaches provide slight different performance (see table 6.6). Finite difference schemes without adjoint computation are also evaluated for comparison purpose. In this case, gradients are calculated according to formula (4.1).

On efficiency comparison between AD techniques and FD schemes

Since the difference of gradient calculation between AD techniques and finite difference schemes is slight, if the perturbations are proper for finite difference (see figure 6.9), we *roughly* assess the efficiency of the two approaches through counting their evaluation time

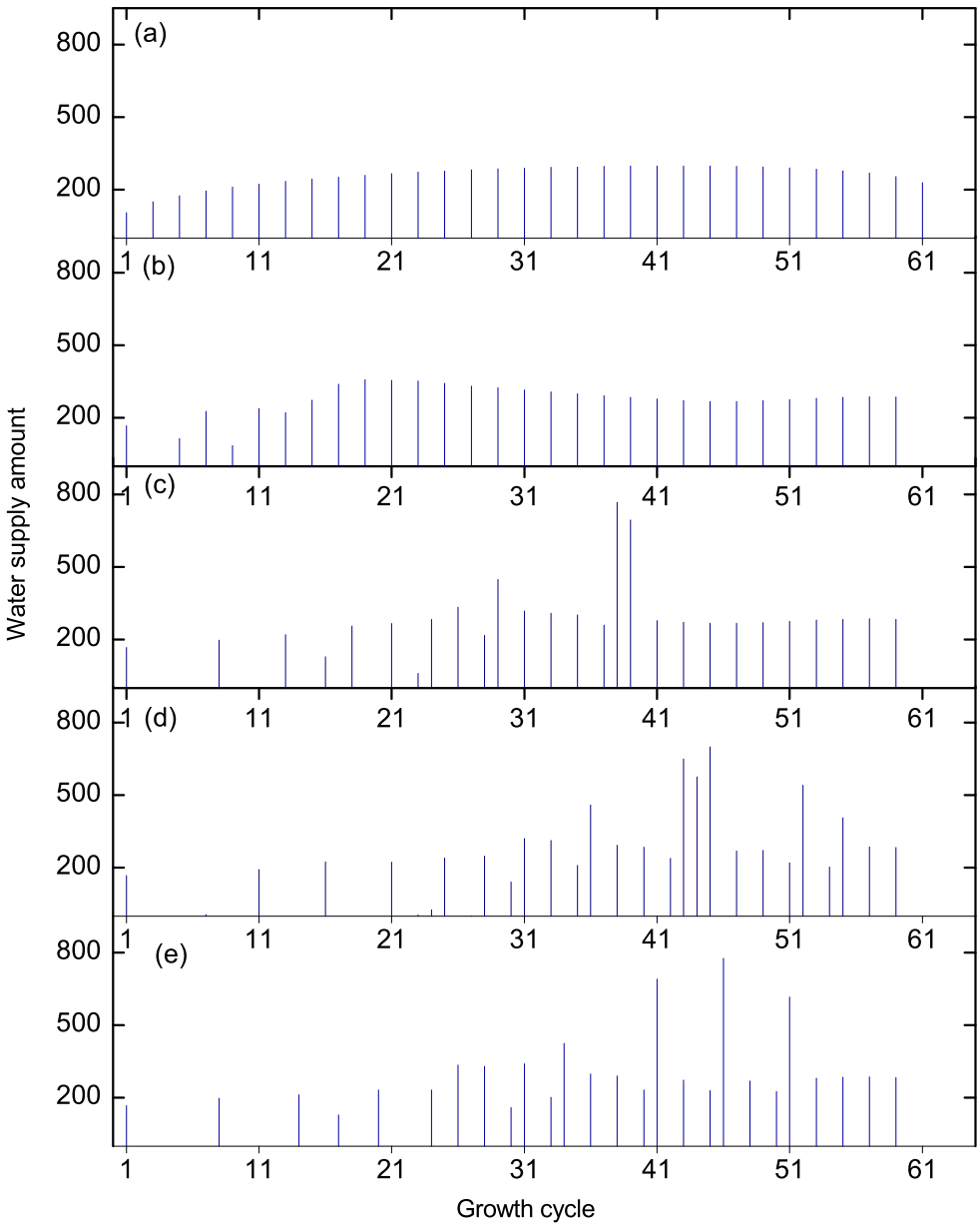


Figure 6.10: Comparison of the water supply strategies: (a) water supply strategy found by solving problem **P4** [128]; (b) water supply strategy found by solving adjoint equation (6.17) with solution (a) as initial condition; (c)-(e) water supply strategies found by solving optimal control system with random initial conditions employing AD techniques.

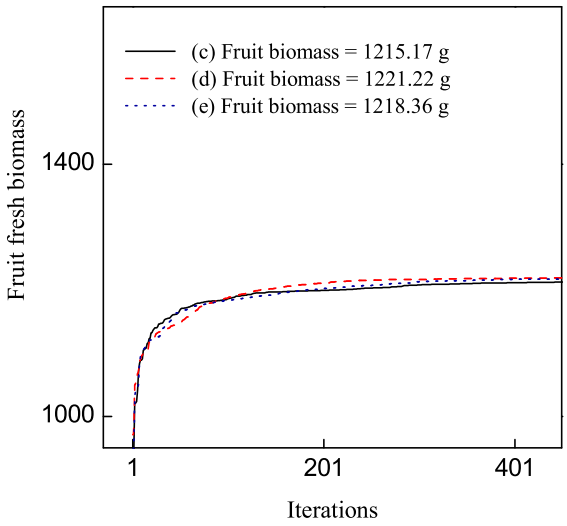


Figure 6.11: Objective function value during optimization iterations for AD approach of optimization. There are very slight increases of objective function values after 400 iterations.

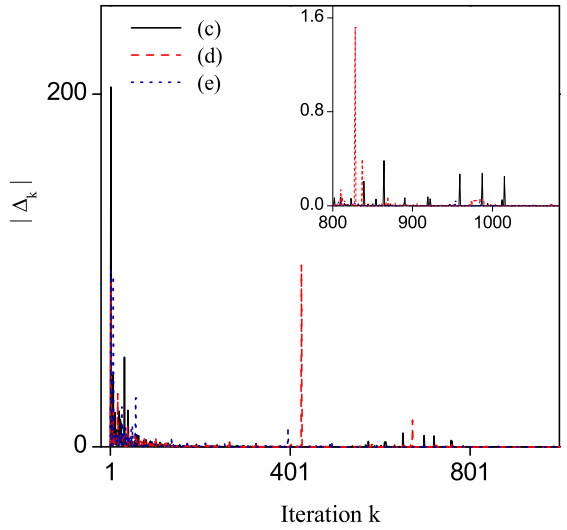


Figure 6.12: Directional derivatives Δ_k at each iteration k for AD approach of optimization. $|\Delta_k| = |\hat{\mathcal{J}}(\mathbf{U}_k, d_k)| = | < \nabla\mathcal{J}(\mathbf{U}_k), d_k > | = |\nabla\mathcal{J}(\mathbf{U}_k)^T d_k|$, where $\mathcal{J}(\mathbf{U})$ is the static objective function of fruit biomass weight (6.11), $\mathbf{U}_k$ is the water supply distribution at iteration k, and d_k is the search direction at iteration k.

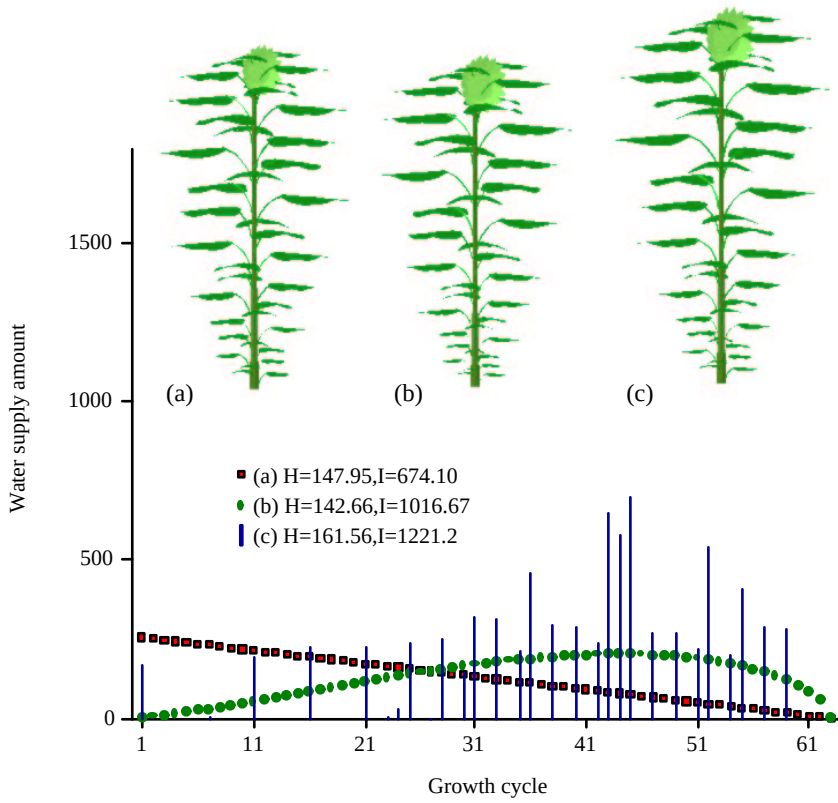


Figure 6.13: Comparison of different water supply strategies and the resulting fruit weight. Water supply strategy (a) is linear; (b) is the water supply strategy that complies with beta function $B(a, b), a = 2.3761, b = 1.5894$; (c) is the solution of optimal control system (figure 6.10-(d)).

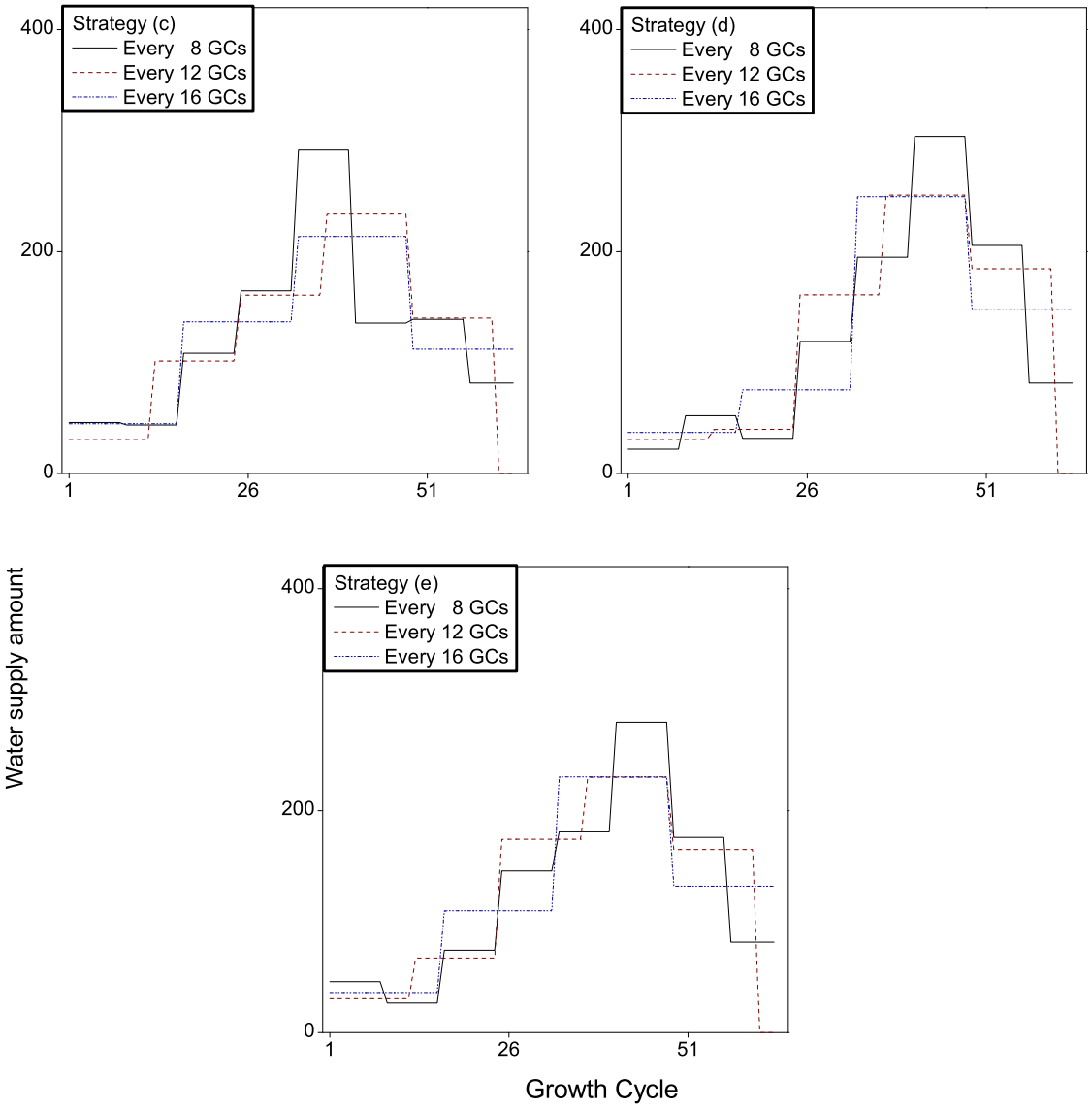


Figure 6.14: Investigation of means of water supply every 8, 12 and 16 GCs respectively for strategy (c)-(e) in figure 6.10. Most of these smoothed curves show one-peak histograms, as can be explained by the prominent evapotranspiration of mid-season growth [17]. Strategy (d) and (e) share extremely similar histogram every 16 GCs.

.

Iter. Num.	4	8	32	64	128	256	512
AC	1039.22	1082.88	1142.08	1173.70	1187.27	1200.36	1207.02
FD	1039.19	1083.84	1139.63	1172.78	1192.19	1206.70	1212.51
AD	1039.22	1082.88	1139.07	1174.00	1189.55	1207.98	1218.16

Table 6.6: Comparison of function values in iterative process for adjoint computation (AC), Finite Difference (FD) and AD techniques with same initial condition chosen randomly.

of objective function called by SQP optimization solver.

Less attentions are paid to analysis of accuracy. Iterative methods using finite difference might fail to converge with unsuitable perturbations [51]. However, according to our numerical experiments, the difference of the optimization iterative process is slight between AD techniques ad FD schemes with suitable perturbations (see table 6.6)

Simulation of *GreenLab* for 63-GC sunflower takes averagely 10.4381 seconds (average of 20 simulations). By contrast, simulation of *MiniGreenLab* takes averagely 0.2094 second. It takes averagely 0.7581 second for *MiniGreenLab* adjoint code. The ratio of evaluation cost between adjoint code and direct code is thus 3.749, which is within the well-known upbound 5.

For optimal strategies (c)-(e) in figure 6.10, we record the evaluation time N_{ad} of adjoint code for gradient calculation in AD optimization approach, and the evaluation time N_{fd} of direct code in FD optimization approach in table 6.7). The saving time T_1 in AD approach compared with FD approach is calculated as

$$T_1 = (N_{fd} - 5N_{ad}) \cdot \tau, \tag{6.23}$$

where τ is the evaluation time in seconds of the direct code.

	N	N_{ad}	N_{fd}	J_{ad}	J_{fd}	T_1	T_2
(c)	1000	3414	66309	1215.17	1213.63	2.86	2.17
(d)	1000	3158	65954	1221.20	1220.03	2.92	2.24
(e)	1000	2877	65820	1218.35	1213.30	2.99	2.23

Table 6.7: Comparison of evaluation times of adjoint code in AD approach and direct code in FD approach with same initial conditions. We employ *MiniGreenLab* for optimization, thus $\tau = 0.2094$s. Herein N is the total number of iterations, J_{ad} is the optimal value of cost function found by AD techniques, J_{fd} is optimal result found by FD schemes. T_1 is the saving time in hours calculated according to (6.23), and T_2 is the practical saving time in hours in numerical experiments.

SQP algorithm can be roughly divided into three parts: gradient calculation, direction update and line search. In our numerical experiments, function evaluations in line search for AD techniques are calculated by adjoint code too, hence the T_1 estimated by formula

(6.23) is great than practical saving time T_2. Further efficiency can be obtained by employing adjoint code for gradient calculation and by employing direct code for line search process separately.

The practical saving time T_2 depends on the machine status, i.e. available memory space, CPU occupation, etc. The comparison in table 6.7 is nothing but a rough estimation of the evaluation time, however, the efficiency of AD techniques is evident.

AD techniques are especially suitable for increasingly complex FSPMs. When objective function is time-consuming, the theoretical saving time T_1 can be enormous. In the case of complete version *GreenLab* ($\tau = 10.4381$s), supposing that iterative processes of optimization are the same as those in table 6.7, the saving times for strategy (c)-(e) calculated by formula (6.23) are 5.94, 6.06, 6.21 days respectively.

6.7 Data assimilation for individual plant: preliminary results

6.7.1 Motivation and formulation

We have introduced the general concepts and basic formulation of data assimilation in chapter 1 and chapter 3 respectively. In this section, we present the preliminary results of data assimilation applications for individual plant.

Firstly we simulate plant growth for the 63-GC sunflower under light, temperature, and soil water content environmental conditions. Model parameters, i.e. Ω_G, are set identically to those in optimal control problem in the previous section. We plot the environmental conditions in figure 6.15. Initial conditions are determined empirically, namely $\Theta = \{\text{Q0, QSW0}\} = \{1.0, 1000\}$, where Q0 is seed biomass and QSW0 is the initial soil water content. The total biomass accumulation and allocation among organs is shown in figure 6.16. Note that these environmental conditions are artificial data for numerical purpose, and too regular to be true. The virtual sunflower is very small compared with the calibrated one, since the biomass production potential E at some GCs are very low under these artificial environmental data according to formula (2.84).

Why data assimilation? It is essentially because of *online* site-relevant or time-relevant uncertainties, which can only be estimated by in situ observations. By term *online* we mean that these uncertainties can not be assimilated before plant begins growth. By contrast, *offline* uncertainties, i.e. genetic parameters Ω_G, are fixed for certain species, no matter the plant grows in field or not. For *GreenLab*, the subject is an *average virtual* plant individual. One can measure initial conditions, i.e. seed biomass Q0 as a mean for several seed samples, and environmental input $\mathbf{U}$, i.e. temperature displayed in thermometer fixed in certain position within or above plant canopy, but these measurements are rather estimations, not real values for the average plant individual. We show in figure 6.17 how these uncertainties affect plant growth considerably. These uncertainties should be estimated by data assimilation schemes for virtual plant growth close to reality.

In summary, uncertainties for *GreenLab* are initial conditions Θ and environmental

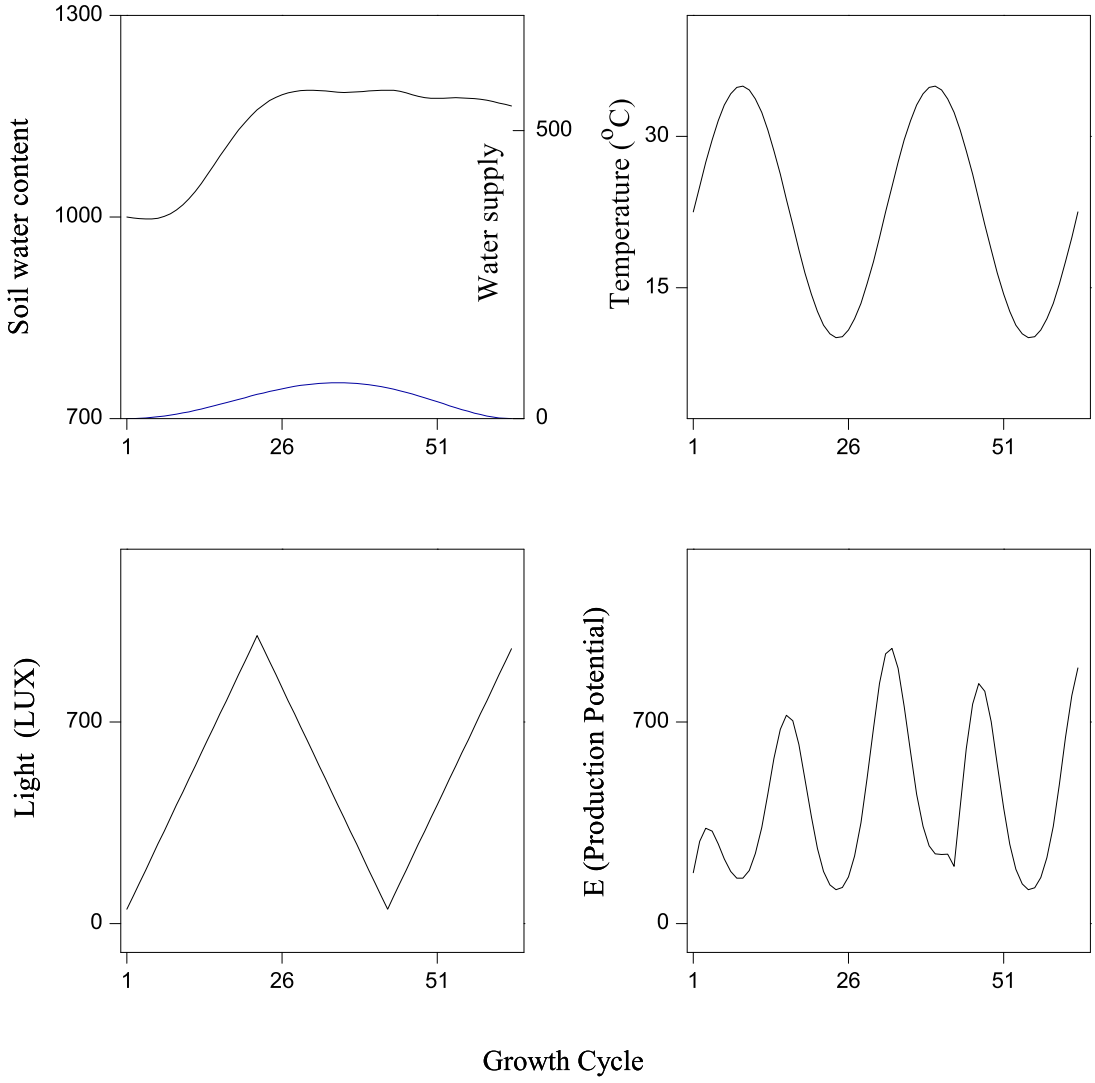


Figure 6.15: Artificial environmental data of 63-GC sunflower for the numerical experiments of data assimilation.

.

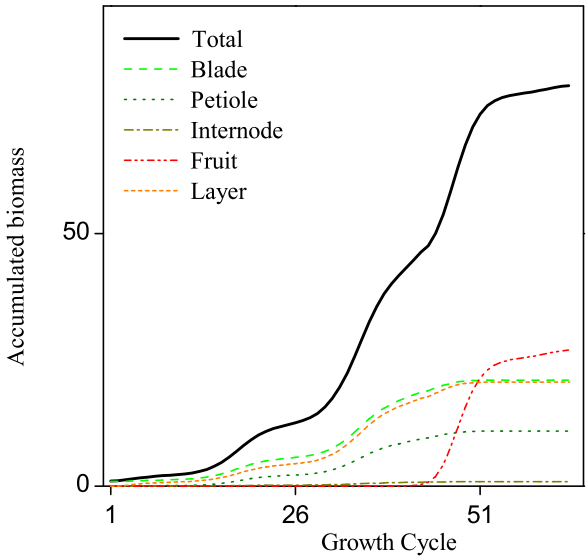


Figure 6.16: Biomass accumulation and allocation of the 63-GC sunflower given empirical initial conditions $\Theta = \{1.0, 1000\}$ under environmental conditions in figure 6.15.

.

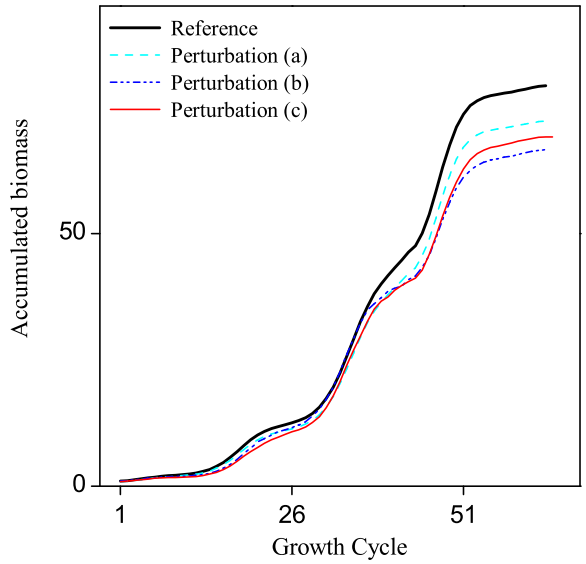


Figure 6.17: Influence of online uncertainties. The reference total biomass accumulation is chosen as that in figure 6.16. We perform three perturbations, namely (a) for a 10% decrease of seed biomass, (b) for a randomly shift of temperature within 3 ^{o}C, and (c) the combination of (a) and (b). One can observe that the *GreenLab* simulation is very sensitive to these uncertainties.

.

input **U**. The formulation of data assimilation problem for *GreenLab* is therefore slightly different from **P2** that only deals with initial conditions. Considering *GreenLab* dynamics in discrete form, the problem is

$$\begin{cases} \mathbf{X}(n) & = & F\left(\mathbf{X}(n-1), \mathbf{U}(n-1)\right), \\ \mathbf{X}(0) & = & \Theta, \\ \min J(\mathbf{K}) & = & \sum_{n=1}^{N} \|\mathbf{X}(n) - \mathbf{X}_{obs}(n)\|^2, \end{cases} \tag{6.24}$$

where N is total GC number, $\mathbf{K} = \{\Theta, \mathbf{U}\}$ is the set of all the parameters to be assimilated. For 63-GC sunflower, let $\mathbf{K}$ be the set of environmental input of light and temperature conditions at each GC together with the initial conditions of seed biomass and soil water content. Hence $\mathbf{K} \in R^M, M = 128$. We denote $\mathbf{K}_{ref}$ the reference parameter set, in which environmental input values are set as in figure 6.15, and initial conditions are identical to those in figure 6.16. The numerical experiments of data assimilation for plant individual is thus to track back $\mathbf{K}_{ref}$ from artificial observations $\mathbf{X}_{obs}$ at each GC generated by twin experiment,

$$\{\mathbf{X}_{obs}(n)\}_{n=1,\ldots,N} = g(\mathbf{K}_{ref}), \tag{6.25}$$

where function g signifies *GreenLab* simulation.

6.7.2 Function convexity

Substituting (6.25) into the formula of $J(\mathbf{K})$ in system (6.24), and applying dynamic equation recursively, similar to the process in section 6.6.2, we have the static optimization problem,

$$J = \mathcal{J}_1(\mathbf{K}), \tag{6.26}$$

which equals to optimal control system (6.24). We apply the algorithm in section 6.2 to evaluate convexity of (6.26) with $N = 20, M = 20$, and the matrix of convexity flag $[\omega_{ij}]$ is shown in figure 6.18. The objective function (6.26) is nonconvex.

6.7.3 Towards better assimilation

The issues in this section involve the techniques and experience in numerical optimization for plant data assimilation problem. We mainly introduce variable scaling for better convergence of the optimization iteration, and the regularization item imposed upon objective function in system (6.24) to escape the iteration from local optima.

Scaling variables

The techniques of scaling variables in optimization problems are useful when these variables vary dramatically in different levels of amplitude [45]. The objective function close to minimum can be approximated as a quadratic elliptical function. The huge discrepancies of variables frequently make the elliptical function extremely narrow, as results in large condition number of its Hessian matrix, therefore descent direction methods converge

$$
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}_{N \times M}
$$

Figure 6.18: Matrix of convexity flag $[\omega_{ij}]_{1\leq i\leq N, 1\leq j\leq M}$. The function is nonconvex, and $\gamma = 0.075$.

slowly, and sometimes get trapped before convergence. By scaling these variables in approximately same level of amplitude, better optimization performance can be expected.

Let x_k the optimization variable in iteration k, the scaling

$$x_k = T(y_k) \tag{6.27}$$

thus transform the optimization process into a sequence of scaled variables y_k.

In the case of data assimilation for sunflower, the ranges of the variables are approximately

$$\begin{array}{lcl} \text{Q0} & \in & [0, 2] \\ \text{QSW0} & \in & [800, 1200] \\ T(n) & \in & [0, 40] \\ L(n) & \in & [0, 1500] \end{array} \tag{6.28}$$

where $T(n), L(n)$ are temperature and light conditions at GC n respectively. In our numerical experiments, these variables are mapped linearly into interval [0, 1] for optimization. The scaling formula is thus

$$s_k = \frac{x_k - x_l}{x_u - x_l} \cdot (s_u - s_l) + s_l, \tag{6.29}$$

$$x_k = \frac{s_k - s_l}{s_u - s_l} \cdot (x_u - x_l) + x_l, \tag{6.30}$$

where s_k are scaled value of x_k, $s_u = 1, s_l = 0$ are upbound and lowbound of scaling range, x_u, x_l are the upbound and lowbound of variables defined in (6.28).

Adding regularization item in objective function

The main difficulty and challenge of numerical optimization is about the interrogation: how to escape from local optima (see figure A.1). The optimization results depend on the form of optimization problems. For instance, the local optimum of convex functions is itself a global optimum, and by contrast, for highly nonlinear objective functions, the location of their global optima is often a masty task.

One can count on stochastic searching algorithms like genetic algorithms, however, there is always a compromise of balancing global performance and searching efficiency. The global algorithms usually do not make full use of problem-relevant information, i.e. the gradients. The hunt for global optimum is at the cost of searching efficiency. In addition, there is no guarantee that the solution found by global algorithms is a global optimum. In practice, alternatively local algorithms can be employed, supplemented by *a priori* information about the solution. It is especially true for large-scale problems, in which the exploitation of global algorithms is generally unsupportable due to the vast computation load.

In our case since we can roughly measure the uncertainties $\mathbf{K}$. These measurements are usually not far from the optimum $\mathbf{K}^*$. Let us denote *a priori* information for $\mathbf{K}^*$ by

$\mathbf{K}_{pri}$, then the searching process is expected to approach $\mathbf{K}^*$ by imposing a regularization item on objective function as

$$J(\mathbf{K}) = \sum_{i=1}^{N} \|\mathbf{X}(n) - \mathbf{X}_{obs}(n)\|^2 + \alpha \cdot \underbrace{\|\mathbf{K} - \mathbf{K}_{pri}\|^2}_{\text{Regularization item}}, \tag{6.31}$$

where α is coefficient for regularization. In our numerical experiments, we simply let $\mathbf{K}_{pri} = \mathbf{K}_{ref}$ and $\alpha = 1$.

Considering regularized optimal control system, let us denote the equivalent static objective function as

$$J = \mathcal{J}_2(\mathbf{K}), \tag{6.32}$$

and let $N = 60, M = 60$, we apply the algorithm in section 6.2 to evaluate the convexity of function (6.32). The results are $[\omega_{ij}] = [0]_{1\le i\le 60, 1\le j\le 60}, \gamma = 0$. The function (6.32) is likely convex, that is, the regularization item reduces the function nonconvexity.

6.7.4 Numerical solutions

Similar to optimal control problem **P3** on water supply, the scheme of gradient calculation for data assimilation system (6.24) can possibly be finite difference schemes, adjoint computation (similar to the process in section 6.6.4) or automatic difference techniques. AD techniques are employed for the numerical solution of system (6.24), due to its efficiency and accuracy shown in the water supply problem. We will firstly verify the adjoint code, then numerical results will be presented.

Validation of adjoint code and gradient calculation

The validation process of adjoint code follows that in section 4.3.3. The direct code is denoted by subroutine

$$[\mathsf{y}] = \mathsf{sim\_DA}(\mathsf{x}),$$

where $\mathrm{x} \in R^M, M = 128$ signifies initial conditions of seed biomass and soil water content, the light and temperature conditions at each GC, $\mathrm{y} \in R$ is the norm of the discrepancy between $\mathbf{X}$ and $\mathbf{X}_{obs}$. The tangent linear code is encapsulated into subroutine

$$[\mathsf{y}, \mathsf{ytg}] = \mathsf{sim\_DA\_tg}(\mathsf{x}, \mathsf{xtg}),$$

and let adjoint code denoted

$$[\mathsf{y}, \mathsf{xad}] = \mathsf{sim\_DA\_ad}(\mathsf{x}, \mathsf{yad}).$$

Exact gradient can be provided by either forward or reverse mode of AD as in section 4.3.3, that is, $\nabla_x y =$xad.

We chose x for three times (denoted as $x^{(i)}, i = 1, 2, 3$ respectively) to verify the adjoint code. For each x, we process as follows for three times with random h.

- Generate perturbation h randomly.
- Run [y, u] = sim_DA_tg(x, h), and record u.
- Evaluate scalar product $u^T u$.
- Given u, run [y, v] = sim_DA_ad(x, u), and record the result v.
- Evaluate scalar product $h^T v$.
- Evaluate the identity $u^T u \equiv h^T v$, and record the difference $u^T u - h^T v$.

We list the results in table 6.8, and the number of significant figures for each calculation item is chosen to be 16.

For $x^{(1)}$	$\|h\|$	$u^T u$	$h^T v$	$y^T y - h^T \bar{x}$
1	10	1256207645.152190	1256207645.152190	4.768371582031250e-007
2	100	43913441912.06044	43913441912.06046	-1.525878906250000e-005
3	1000	15861275631286.73	15861275631286.71	1.757812500000e-002
For $x^{(2)}$	$\|h\|$	$u^T u$	$h^T v$	$y^T y - h^T \bar{x}$
1	10	1960088068.413013	1960088068.413014	-1.192092895507813e-006
2	100	812331612778.6439	812331612778.6438	1.220703125000000e-004
3	1000	17180039125400.29	17180039125400.30	-1.562500000000e-002
For $x^{(3)}$	$\|h\|$	$u^T u$	$h^T v$	$y^T y - h^T \bar{x}$
1	10	325294481.0626455	325294481.0626458	-2.980232238769531e-007
2	100	29302165006.29209	29302165006.29211	-1.525878906250000e-005
3	1000	218804956739.3478	218804956739.3480	-2.136230468750000e-004

Table 6.8: Validation of adjoint code for data assimilation problem, where $x^{(1)} = 0.7 \cdot x^{\text{ref}}$, $x^{(1)} = 0.9 \cdot x^{\text{ref}}$, $x^{(1)} = 1.1 \cdot x^{\text{ref}}$, x^{ref} is the reference parameter value.

Validation results on gradient is examined according to formula (4.70). We list $f(\alpha)$ in table 6.9 with α varying from 10^0 to 10^{-15} and plot function $f(\alpha)$ in figure 6.9, in which α ranges from 2^0 to 2^{-50}.

Optimization results

We show the optimization results of SQP algorithms in figure 6.20 and figure 6.21. Initial searching points are both set to $0.9\mathbf{K}_{ref}$. For the former, there is no penalty item in the objective function, and for the latter, the penalty item is imposed. We track back reference parameter $\mathbf{K}_{ref}$ in the case with penalty item, whereas for the case without penalty item, the searching process converge slowly (directional derivatives tend to zero), and the optimization results seem to diverge from reference parameter $\mathbf{K}_{ref}$, although the observation target $\{\mathbf{X}_{obs}(n)\}$ is well fitted.

The iterations of optimization process are shown in figure 6.22 and in figure 6.23. The optimization results differ considerably with respect to different initializations when there

α	$f(\alpha)$	α	$f(\alpha)$
10^{0}	0.6682924196320	10^{-8}	1.0000004205149
10^{-1}	0.9740965665650	10^{-9}	1.0000049408444
10^{-2}	0.9974839177700	10^{-10}	1.0000209748790
10^{-3}	0.9997491337940	10^{-11}	1.0003244438884
10^{-4}	0.9999749208470	10^{-12}	1.0050712427274
10^{-5}	0.9999974927910	10^{-13}	1.0444769201124
10^{-6}	0.9999997533360	10^{-14}	1.0236417343686
10^{-7}	1.0000000237405	10^{-15}	2.0835185743790

Table 6.9: Validation of gradient calculation, where $\|h\| = 10$. Gradients obtained by finite difference are acceptable only within certain range of perturbation $\alpha h, \alpha \in [10^{-5}, 10^{-11}]$.

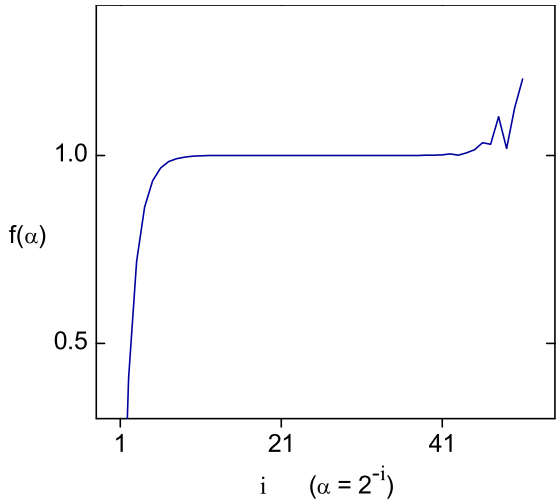


Figure 6.19: Validation of gradient calculation, where $\|h\| = 10$. Coefficient α of acceptable perturbations αh with accuracy of 4 decimal places ranges from 2^{-15} to 2^{-34} (approximately $[10^{-5}, 10^{-11}]$).

is no regularization.

Considering the case without penalty item, for the iteration 100, optimization result Q0 is close to its reference value, whereas environmental conditions and initial soil water content QSW0 are still considerable different from their reference values. This fact hints that Q0 is possibly more important (or sensitive) than other variables. Note that there are oscillations for light condition L at around 20-30 GCs, where L is close to 1000, that is to say, the plant individual is under light extinction effects. Plant growth is less sensitive to light conditions during these period.

6.7.5 Remarks

The significance of the results is to elucidate the necessity of data assimilation process in FSPM applications. In addition, we formulate the variational framework and develop essential techniques, namely automatic differentiation, for the solution of the data assimilation problem.

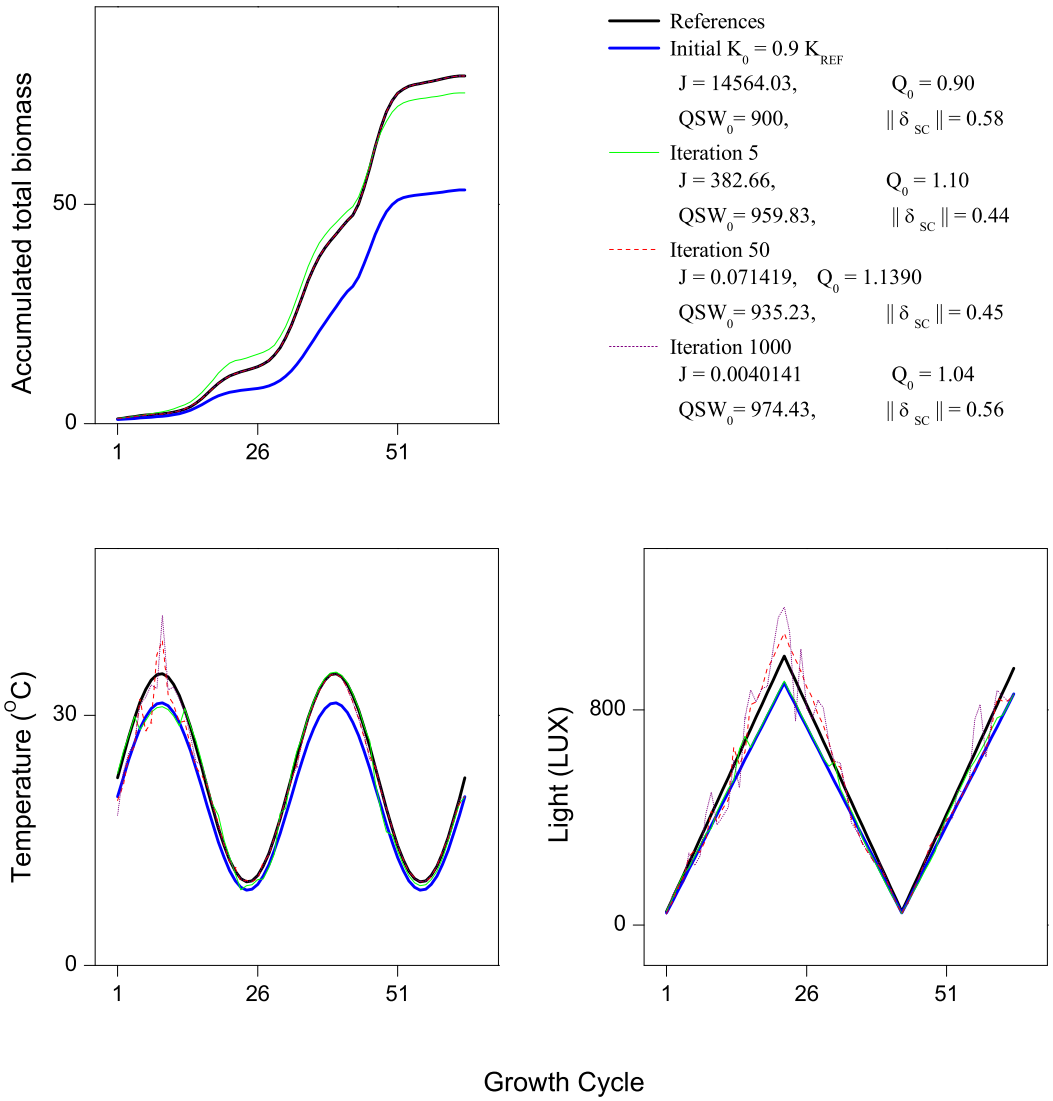


Figure 6.20: Iterative optimization results for the case without penalty item. For each iteration in the figure, we list the iteration results, in which J is the objective function value, δ_{SC} is the quality that measures the discrepancy between variable value $\mathbf{K}_k$ of the iteration and reference parameter $\mathbf{K}_{ref}$ ($\delta_{\mathrm{SC}} = \|(\mathbf{K}_{ref} - \mathbf{K}_k)/\mathrm{SC}\|$, and SC is the scaling coefficient vector).

.

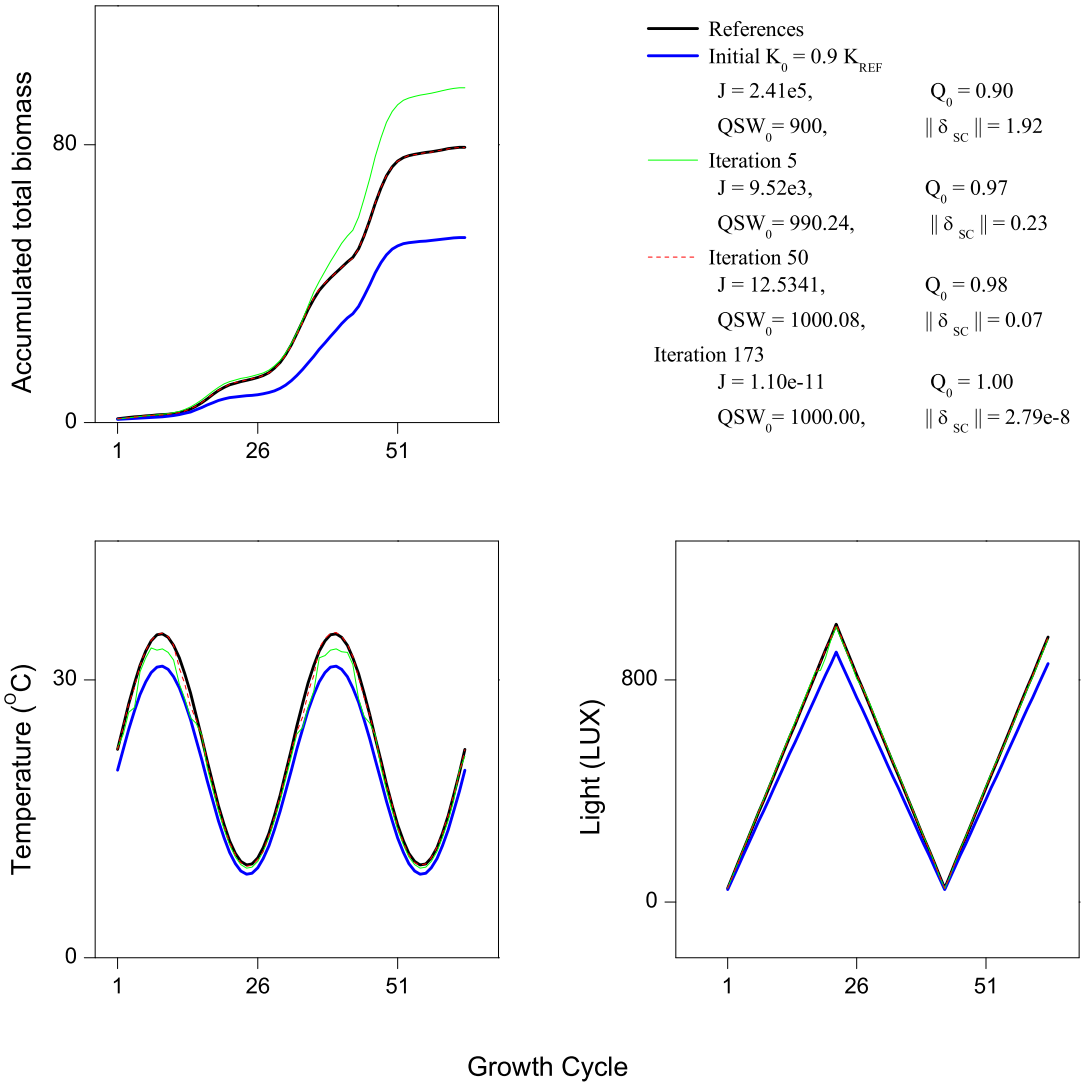


Figure 6.21: Iterative optimization results for the case with penalty item. For each iteration in the figure, the notations are the same as those in figure 6.20.

.

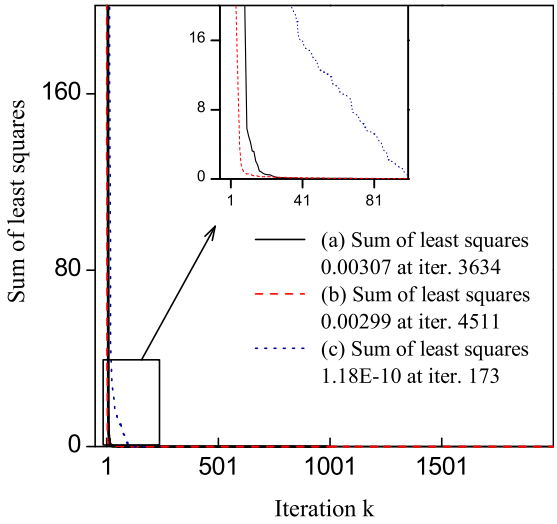


Figure 6.22: Sum of least squares during optimization iterations for AD approach of optimization. The case (a) is obtained with initial $\mathbf{K}_0 = 0.9\mathbf{K}_{ref}$ and without penalty item as in system (6.24). In case (b) the initialization is as $\mathbf{K}_0 = 0.95\mathbf{K}_{ref}$, and there is no penalty item. For case (c) penalty item is added and $\mathbf{K}_0$ is initialized as $0.9\mathbf{K}_{ref}$ for optimization. Denoting optimization results of case (a) by $\mathbf{K}_1$, and similarly $\mathbf{K}_2$ for case (b), we evaluate their difference as $\delta = \frac{\|\mathbf{K}_1 - \mathbf{K}_2\|}{\|\mathbf{K}_1\|} = 0.157$

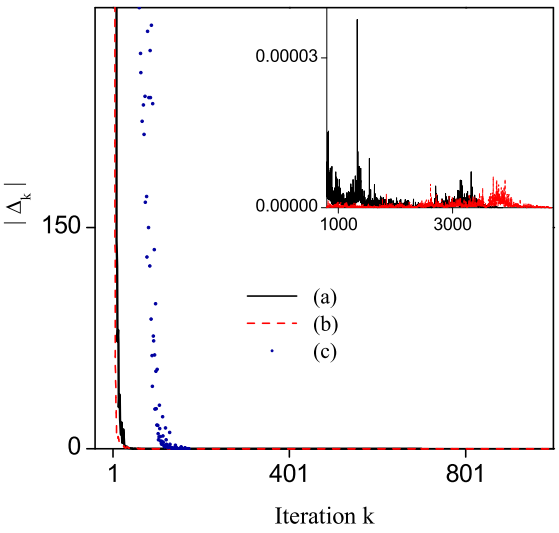


Figure 6.23: Directional derivatives Δ_k at each iteration k of optimization for case (a)-(c) in figure 6.22. $|\Delta_k| = |\hat{\mathcal{J}}_i(\mathbf{K}_k, d_k)| = | < \nabla \mathcal{J}_i(\mathbf{K}_k), d_k > | = |\nabla \mathcal{J}_i(\mathbf{K}_k)^T d_k|$, where $\mathcal{J}_i(\mathbf{K}), i = 1, 2$ is defined as in (6.26) and (6.32), $\mathbf{K}_k$ is the parameter value at iteration k, and d_k is the search direction at iteration k.

The data assimilation results presented in this chapter are rather initial and preliminary. Aiming at better assimilation results, one has to employ refined data assimilation schemes. Here we refer to mainly two improvements: control of model error and sensitivity analysis of observation errors. For the former, since practical model is always imperfect, the optimal control system with model error as control variable, and with data assimilation performance as cost function, can be defined for better assimilation of observations. Usually it involves a reduction of error space. For the latter, the optimality system for data assimilation problem is considered to impose an accordance between model and observations for better assimilation performance, that is, the observations should be provided to the assimilation system with respect to their importance or priorities, as is the research content of sensitivity analysis of observation errors.

6.8 Conclusion

In this chapter, we firstly illustrate an optimization problem of maize source-sink relationships for the mathematical model analysis. The attempts on the last two Lions steps are then presented, namely on data assimilation problem **P2** and on control problem **P3**. Optimal water supply strategies for better fruit yield (at organ level) are obtained by solving the optimal control system of soil-plant dynamics. The online uncertainties of initial conditions and environmental model input are distinguished and estimated by data assimilation concepts and schemes. These two problems are formulated in the framework of variational methods, and AD techniques prove to be very efficient for solving these variational systems.

The definition and solution of these problems demonstrate how apply mathematics (variational methods in our case) is introduced to FSPM research to obtain the results that can not be achieved by FSPM modelers themselves. The optimal strategies in figure 6.10 are the consequence of numerical simulation and optimization based on the collection of biological knowledge and mathematical techniques. More precisely, variational formulation of FSPM dynamics that contains abstract interface to biologists, is employed to bridge the gap between physiological inductions and mathematical deductions.

We point out that the work is initial, and somehow lacks of practical support and theoretical insight. For instance, in data assimilation problem, we use artificial observations generated by twin experiments. *In situ* measurements should be gathered for the down-to-earth applications of data assimilation concepts and methodologies. In optimal control problem, we basically emphasize numerical experiments, and less attentions are paid in theoretical analysis, i.e. the observability and controllability conditions of the water supply control system.

Chapter 7

Conclusion and perspective

As addressed in the thesis title, our efforts are mainly to develop variational methods for plant functional-structural characteristics. The research of this thesis is rather a start and part of the constant attempt on introduction of applied mathematics to FSPM community. We summarize the thesis results, and try to answer questions (i)-(v) proposed in chapter 1.

The research is based on the mathematical formulation of *GreenLab* plant functional-structural dynamics. Plant organogenesis is presented with an emphasis of CA-based dynamics. The abstract interface for plant physiology is proposed for future developments of *GreenLab*.

We introduce variational method and control theory into FSPM comity, as provides a formularization framework when FSPMs become increasingly complex (question (iii)). For instance we formulate optimal control and parameter calibration problems within growth cycles. Coupling with plant physiological knowledge in this case is more convenient, since it is possible to consider calendar time index say in hours or days.

We introduce data assimilation conceptions and methods into FSPM research to implement the idea that intends to merge dynamic FSPM and observations for a better estimation for plant growth (question (i), (ii), (v)).

We introduce adjoint model (code) and automatic differentiation techniques for the numerical solution of plant optimal control system and data assimilation system. The calculation of gradient in this way for optimization algorithms is independent of the model size in the sense that the calculation cost of adjoint code is no more than 5 times of that of model direct code. The numerical approach for the variational method is therefore expandable when FSPM evolves, i.e. considering the heterogeneous phylloclimate conditions in the canopy for plant organs, the dimension of state variable can be enormously increased (question (iv)).

We introduce the functioning of environmental factors, such as light, temperature and soil water content, to mimic plant growth in the calculation of biomass production for GreenLab. We then rewrite the GreenLab as a dynamical soil-plant system, as facilitates the formulation of control problems, i.e. water supply optimal control system. The numerical results lead to possible sustainable agricultural applications.

We now suggest some areas that require more investigations relevant to this thesis.

- The functional aspects of *GreenLab* can possibly be reinforced by implementing the abstract physiological interface with the collaboration of physiologists, especially for the introduction of the functioning within GCs and the consideration of heterogeneous phylloclimate. In both cases, variational methods provide the framework for model analysis and applications.
- So far our research is limited to deterministic plants. The concepts and methods for the estimation and control can be expended to the case of stochastic plants.
- Considering the online uncertainties, the following statement is natural: we cannot obtain satisfactory results for the processes of validation and control in Lions steps without the estimation of online uncertainties. A proper approach might be firstly an estimation with the help of runtime observations, then the process of validation or control follows. That is what we call the *validation and control in the context of data assimilation*. F.-X. Le Dimet demonstrates a similar process of sensitivity analysis in the presence of observation, as involves a formulism framework of second order adjoint model based on the optimality system [72].

Conclusion et perspective en français

Comme indiqué dans le titre de la thèse, nos efforts se sont portés sur les méthodes variationnelles pour identifier les caractéristiques de la structure fonctionnelle des plantes. Ce travail de thèse est plutôt un début, mais il constitue une première étape pour l'utilisation des mathématiques appliquées dans la communauté FSPM. Nous récapitulons ici les résultats de la thèse, et essayons de répondre aux questions (i)-(v) proposées dans le chapitre 1.

Le travail est fondé sur la formulation mathématique de la dynamique de la structure fonctionnelle des plantes de *GreenLab*. L'organogenèse des plantes est présentée en mettant l'accent sur la dynamique basée sur la notion de CA. On propose l'interface pour la physiologie des plantes pour des développements futurs de *GreenLab*.

Nous avons introduit les méthodes variationnelles et la théorie du contrôle optimal dans FSPM, ce qui fournit un cadre quand les FSPMs deviennent de plus en plus complexes (question (iii)). Par exemple nous formulons des problèmes de contrôle optimal et de calibrage de paramètres dans des cycles de croissance. Le couplage avec la connaissance physiologique des plantes devient plus facile, puisque dans ce cas les indices calendaires (par jour ou heure) sont probablement considérés.

Nous avons présenté des conceptions et des méthodes d'assimilation de données dans la recherche de FSPM pour fusionner la dynamique de FSPM et les observations pour une meilleure évaluation de la croissance des plantes (question (i), (ii), (v)).

Nous présentons le modèle adjoint (code) et les techniques de différentiation automatique pour la solution numérique du système de contrôle optimal et du système d'assimilation de données pour les plantes. Le calcul du gradient pour les algorithmes d'optimisation est indépendant de la taille du modèle, car le coût de calcul du code adjoint n'est pas plus de 5 fois celui du code direct du modèle. L'approche numérique pour la méthode variationnelle est donc extensible quand FSPM évolue, c'est à dire, considérons que le phytoclimat hétérogène conditionne dans le canopy pour des organes des plantes, la dimension de la variable d'état peut être énormément augmentée (question (iv)).

Nous présentons une formule empirique pour calculer la production de biomass en considérant l'influence des facteurs environnementaux (l'éclairement, la température, et la teneur en eau du sol). Nous réécrivons alors *GreenLab* comme un système dynamique sol-plantes, ce qui facilite la formulation des problèmes de contrôle, i.e. le contrôle optimal d'approvisionnement en eau. Les résultats numériques mènent probablement à des applications agricoles durables.

Nous suggérons maintenant quelques voies de recherche ouvertes par ces travaux.

- les aspects fonctionnels de *GreenLab* peuvent probablement être renforcés en tenant compte de l'interface physiologique avec la collaboration des physiologistes, particulièrement pour l'introduction du fonctionnement dans GCs en considérant les phytoclimats hétérogènes. Dans les deux cas, les méthodes variationnelles fournissent le cadre pour l'analyse et le contrôle.

- jusqu'ici notre recherche est limité aux plantes déterministes. Les concepts et les méthodes de cette thèse pourront être adoptés pour l'évaluation et le contrôle dans le cas des plantes stochastiques [73], [63].

- considérant les incertitudes en ligne, la remarque suivante est normale : nous ne pouvons pas obtenir des résultats satisfaisants pour les processus de la validation et du contrôle dans les étapes de Lions sans évaluer des incertitudes en ligne. Une approche appropriée pourrait être comme ci-dessous : d'abord nous réalisons une estimation avec l'aide des observations, ensuite une validation ou un contrôle est exécuté. C'est ce que nous appelons *la validation et le contrôle dans le contexte de l'assimilation de données.* F.-X. Le Dimet démontre un processus d'analyse de sensibilité en présence de l'observation, ce qui implique un cadre de formulisme du modèle adjoint du second ordre basé sur le système d'optimalité [72].

Appendix A

Brief Introduction to Optimization

A.1 Introduction

Mathematically an optimization problem can be formulated as minimization or maximization of an *objective function* $f(x)$ subject to *constraints* on its *variables* x as

$$\begin{array}{ll} \min\limits_{x \in R^n} f(x) & \\ c_i(x) = 0, & i = 1, ..., m_e, \\ c_j(x) \leq 0, & j = m_e + 1, ..., m, \end{array} \tag{A.1}$$

where $c_i(x)$ are equality and inequality constraints. Suppose Ω be the feasible set such that for $\forall x \in \Omega$, x satisfies the constraints $c_i(x), i = 1..m$.

Nowadays optimization algorithms tend to be extremely complex, such that there is little chance for the user who aims to solve an optimization problem to write his own implementation from zero to a usable released software. Instead, it would be better that the user selects routines from available high-quality mathematical software libraries. For suitable applications of these libraries, one needs to remain well-informed by the nature of the algorithms and the important features of the optimization software. For this purpose Gill, Murray, and Wright overview in extreme details the existing local algorithms in their monograph [45]. In this thesis, we mainly select routines in MATLAB optimization toolbox, or academic software like MINPACK-1, rather than implement these traditional algorithms ourselves.

A.2 Classifications of optimization methods

There are variety of classifications for optimization problems and algorithms according to the characteristics of their three ingredients: objective function (e.g. differentiable or not), variables (e.g. discrete or continuous) and constraints (e.g. linear or nonlinear). Other classifications are based on the algorithm behaviors, e.g. global and local algorithms, stochastic and deterministic methods, and static and dynamic optimization.

A.2.1 Static and dynamic optimization

The optimization problem (A.1) is called to be static. It becomes a dynamic one, when the variable x is associated with evolutionary laws or dynamics of some state variable y, i.e. the

differential equation $\frac{dy}{dt} = g(y, x, t)$. In this section, we deal with only static optimization problem. The dynamic problems are presented in chapter 3 and solved in the framework of variational formularization in the following chapters.

A.2.2 Global and local algorithms

Definition A.1 *A point x^* is a global minimum if $f(x^*) \leq f(x)$ for all x in feasible set Ω; a point x^* is a local minimum if there is a neighborhood* N *of x^* such that $f(x^*) \leq f(x)$ for $x \in$ N.*

Global algorithms aim to find the global optima, whereas local algorithms are satisfied with the local optima. Local algorithms usually employ 'downhill' searching strategy, and the search stops whenever an optimum is found (figure A.1-a). By contrast, the searching process of global algorithms escapes from the local optima according to either *deterministic* (figure A.1-b) or *stochastic* searching methods (figure A.1-c). The backtracking mechanism, i.e. branch and bound, is being employed extensively in deterministic methods. For stochastic methods, the search jumps away local minima based on probabilistic decisions, for instance genetic algorithms and simulated annealing.

Many successful global optimization algorithms can escape from local optima at the cost of additional computation load, in which a sequence of local algorithms is sometimes processed. Applications of global algorithms for *GreenLab* can be seen in [22] (simulated annealing) and in [128] (genetic algorithms). It is usually time-consuming for global algorithms, and furthermore the problem-specific properties, i.e. model dynamics and gradient information, are usually neglected in the black-box-like simulation of the objective function in these global algorithms. The performance can deteriorate when the model or the optimization problem becomes increasingly complex. In this thesis, we mainly employ local algorithms integrated with the plant functional-structural dynamics for the sake of optimization efficiency.

A.3 Introduction to local algorithms

Local optimization theory concerns mainly [23],

- The existence and uniqueness of the solution of problem (A.1).
- The characteristics of the solutions, that is, sufficient or necessary optimality conditions.
- The design of algorithms, in which a sequence $(x_k)_{k>0}, s.t., \forall x_0, \lim_{k\to\infty} x_k = x$, is constructed benefiting from the characteristics of the solutions.

The sequence has similar iterative structure

$$x_{k+1} = x_k + \alpha_k d_k, \tag{A.2}$$

where x_k is the $k-$th iterative intermediate, α_k is searching step, and d_k is searching direction. One basic algorithm of this type is the Newton method and its variants.

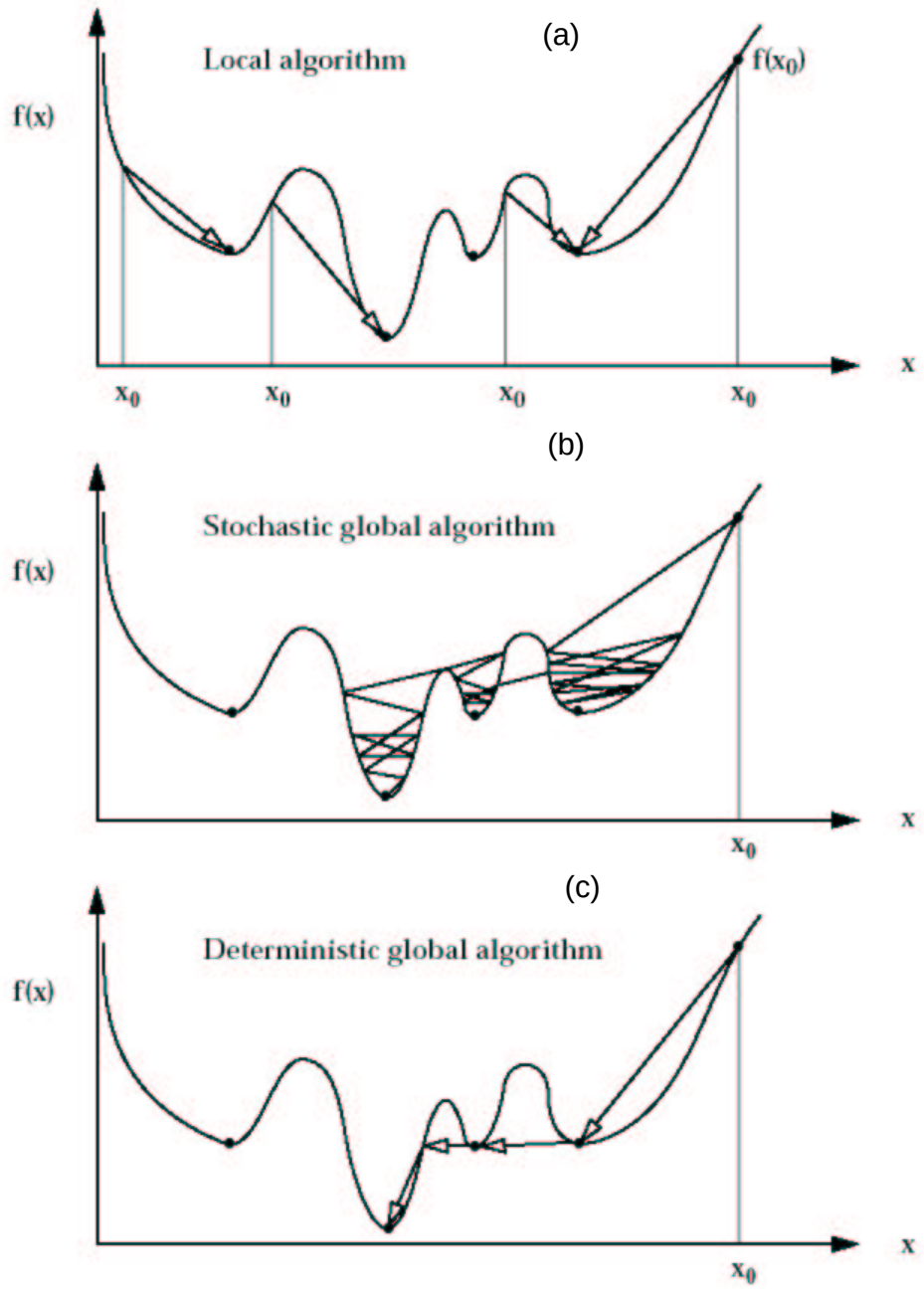


Figure A.1: The structure of local and global minimization algorithms (from 'Mathematical optimization', Oak Ridge National Laboratory)

A.3.1 Newton methods

The Taylor series expansion of f (A.1) of second order approximates f near to x_k

$$f(x_k + p) \simeq f(x_k) + \nabla f(x_k)^T p + \frac{1}{2} p^T \nabla^2 f(x_k) p, \tag{A.3}$$

where $\nabla f(x_k), \nabla^2 f(x_k)$ is the gradient and Hessian matrix of f at x_k respectively,

$$\nabla f(x) = \left[\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right]^T \tag{A.4}$$

$$\left[\nabla^2 f(x) \right]_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}, \quad 1 \leq i, j \leq n. \tag{A.5}$$

Minimizing the right-hand side of (A.3), we have the Newton equation

$$\nabla^2 f(x_k) p = -\nabla f(x_k). \tag{A.6}$$

Let p_k be the solution of (A.6) (defined as Newton direction), the iteration

$$x_{k+1} = x_k + p_k = x_k - \left\{ \nabla^2 f(x_k) \right\}^{-1} \nabla f(x_k), \tag{A.7}$$

approximately approaches to minimum of problem (A.1).

In general, Hessian matrix $\nabla^2 f(x_k)$ is not guaranteed to be positive definite. For indefinite Hessian matrices, matrix factorization technique is adopted in *modified Newton methods* to adjust these Hessian matrices to produce positive definite ones. Another approach is to approximate curvature information (provided by Hessian matrices) based on accumulated information of function f and its gradient ∇f at each iteration. This approach is called *quasi-Newton method*, in which approximate Hessian matrices A_k are updated according to the following two different formulae,

- Davidon-Fletcher-Powell formula (DFP)

$$A_{k+1} = A_k - \frac{d_k g_k^T A_k + A_k g_k d_k^T}{d_k^T g_k} + (1 + \frac{g_k^T A_k g_k}{d_k^T g_k}) \frac{d_k d_k^T}{d_k^T g_k}, \tag{A.8}$$

- and Broyden-Fletcher-Goldfarb-Shanno formula (BFGS)

$$A_{k+1} = A_k - \frac{A_k g_k g_k^T A_k}{g_k^T A_k g_k} + \frac{d_k d_k^T}{d_k^T g_k}, \tag{A.9}$$

where $g_k = \nabla f(x_{k+1}) - \nabla f(x_k), d_k = x_{k+1} - x_k$.

With initial searching point x_0 far from optimal solution x^*, its Hessian matrix is not guaranteed to be positive definite at x_0. In this case, Newton directions are not always descent directions, therefore, a step-length procedure must be included, the quasi-Newton algorithm is as follows,

algorithm

Step 1: Let $x_0 \in R^n, A_0 = I \in R^{n \times n}, 0 \leq \varepsilon < 1, k := 0$.

Step 2: If $\|g_k\| \leq \varepsilon$, stop; otherwise calculate d_k by solving $A_k d = -g_k$.

Step 3: Let

$$x_{k+1} = x_k + \alpha_k d_k, \tag{A.10}$$

where α_k is the update step that can be determined by a linear search process

$$\alpha_k = \arg\min_{\alpha > 0} \quad \phi(\alpha) = f(x_k + \alpha d_k). \tag{A.11}$$

Step 4: Update A_k according to formula (A.8) or formula (A.9).

Step 5: Let $k := k + 1$, go to step 2. ■

The line search process (A.11) is a univariate minimization problem. There are standard algorithms that deal with this problem, i.e. golden section search method [45]. The idea of golden section search is to perform an interval reduction procedure by comparing only function values at boundaries of evaluating intervals. When the interval length is reduced to sufficient small positive value, all the points within the resulting interval are approximate minima.

We abstract a more general flow chart of iterative optimization algorithm as in figure A.2.

Practical optimization problems are usually imposed with certain constraints, and by introducing Lagrange multiple (or penalty functions) these problems can be converted into unconstrained ones.

A.4 Remarks on optimization methods in this thesis

We employ mainly three local optimization algorithms.

- The source-sink relationships analysis in chapter 6 is a univariate bound-constrained optimization problem, and we employ optimization toolbox routine *fminbnd* that implements golden section search method.
- In chapter 5, Levenberg-Marquardt (LM) algorithm for the calibration of environmental parameters is in essence a Newton method with its objective function as sum of least squares, as yield efficient calculation of gradients and Hessian matrices (see algorithm details in section A.5). We write FORTRAN MEX for academic software MINPACK-1 that implements LM algorithm.
- Sequential Quadratic Programming (SQP) algorithm is employed for the water supply optimization in chapter 6. The idea of SQP is to solve the original problem by a sequence of QP (Quadratic Programming) subproblem. BFGS formula is employed for the estimate of Hessian matrix of the Lagrangian in the implementation of optimization toolbox routine *fmincon*. Algorithm introduction is presented in section A.6

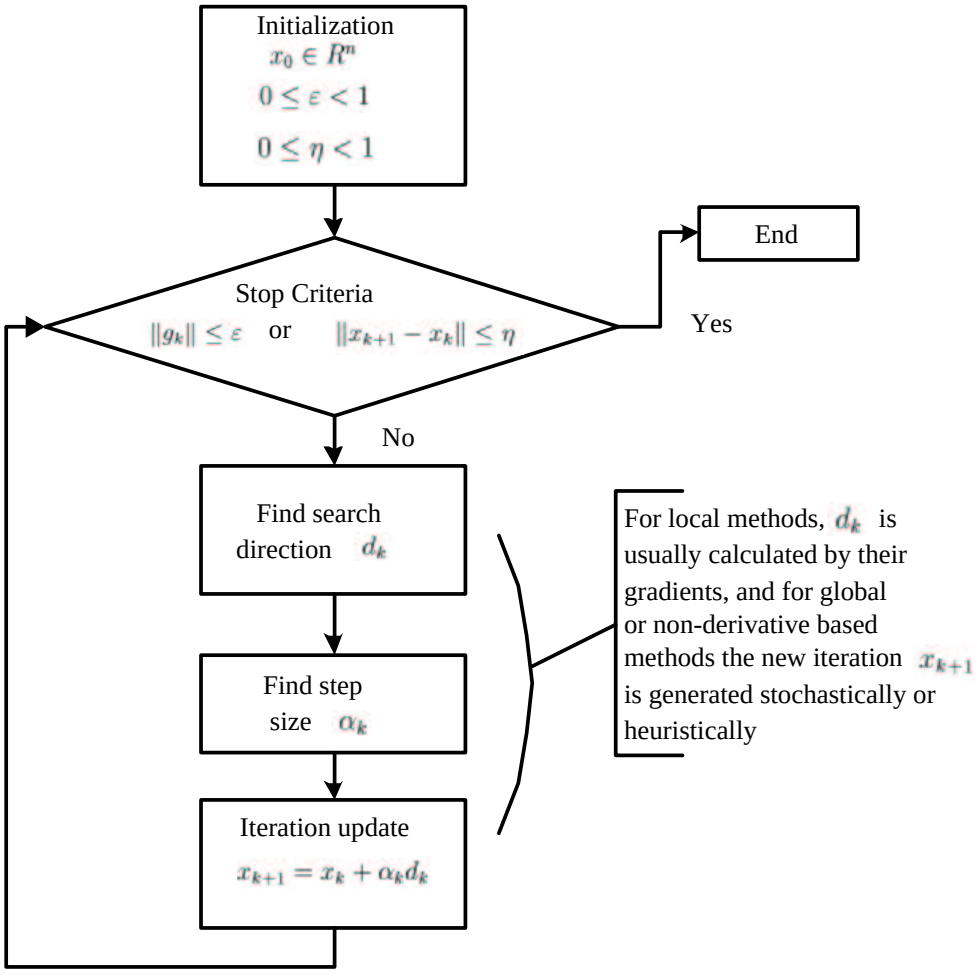


Figure A.2: Flow chart of iterative optimization algorithm.

The formulation in chapter 3 illustrates that model dynamics yield an efficient calculation of gradient by introducing adjoint model. In our numerical experiments, the gradients can be obtained by either Finite Difference or Automatic Differentiation techniques (AD, chapter 4). The former is suitable when the dimension n of variable x is small (for calibration problem in chapter 5 $n = 4$), and for the latter, it is highly efficient when optimizing scalar objective function with respect to variables x with enormous dimension ($n = 128$ for data assimilation problem and $n = 63$ for optimal control problem in chapter 6).

A.5 On least squares algorithms

[1] When the objective function of optimization problem (A.1) takes the form of a sum of squares of nonlinear $m-$vector function $r : R^n \mapsto R^m$,

$$\begin{aligned} f(x) &= \tfrac{1}{2}\sum_{i=1}^{m}[r_i(x)]^2 \\ &= \tfrac{1}{2}r(x)^T r(x) = \tfrac{1}{2}\|r(x)\|^2, \qquad m \gg n, \end{aligned} \tag{A.12}$$

it is classified into Least SQuares (LSQ) problems. Here $\|r(x)\|$ is termed as residual at x. Identification of parameters for some system $\phi(x,t)$ can be represented as this form by denoting $r_i(x) = \phi(x,t_i) - y_i$, where vector $x = \{x_i\}$ signifies system parameters, and y_i is the data obtained from a target function $\mathcal{F}(t_i)$, that is,

$$\min_{x\in R^n} \int_{t_0}^{t_1} \left(\phi(x,t) - \mathcal{F}(t)\right)^2 dt, \tag{A.13}$$

or in the discrete form,

$$\min_{x\in R^n} \sum_{i=1}^{m} \left(\phi(x,t_i) - \mathcal{F}(t_i)\right)^2. \tag{A.14}$$

Generalized Newton methods can be employed to minimize least squares (A.12), however, the LSQ problem (A.14) has special structures, e.g. the gradient $\nabla f(x)$ and Hessian matrix $\nabla^2 f(x)$ of function (A.12) equal to

$$\nabla f(x) = J(x)^T r(x), \tag{A.15}$$

$$\nabla^2 f(x) = J(x)^T J(x) + Q(x), \tag{A.16}$$

where $J(x)$ is the $m \times n$ Jacobian matrix of $r(x)$, $Q(x) = \sum_{i=1}^{m} r_i(x)G_i(x)$, and $G_i(x) = \nabla^2 r_i(x)$ the Hessian matrix of $r_i(x)$, thus usually special methods are designed for LSQ problems, of which we refer to Gauss-Newton methods, Levenberg-Marquardt method, and quasi-Newton approximations.

Let a quantity subscripted by k be the quantity value at x_k, the Newton equation (A.6) is then

[1] The formularization in this section is a mixed presentation of [45], [133]

$$(J_k^T J_k + Q_k)p_k = -J_k^T f_k. \tag{A.17}$$

Several approaches approximate the Newton direction p_N of equation (A.17), and we list some as follows,

- **Gauss-Newton Method:** Neglect the second order update Q_k when x_k tends to the solution, we have

  $$J_k^T J_k p_k = -J_k^T f_k. \tag{A.18}$$

  This equals to solve the following optimization problem

  $$\min_{p \in R^n} \frac{1}{2}\|J_k p + f_k\|_2^2. \tag{A.19}$$

- **Levenberg-Marquardt (LM) Method:** Approximate (A.17) as

  $$(J_k^T J_k + \lambda_k I_k)p_k = -J_k^T f_k, \tag{A.20}$$

  where λ_k is a non-negative scalar, I_k the $n \times n$ identity matrix. This equals to

  $$\begin{array}{ll} \min\limits_{p \in R^n} & \frac{1}{2}\|J_k p + f_k\|_2^2 \\ s.t. & \|p\|_2 \leq \Delta. \end{array} \tag{A.21}$$

  The constraint of (A.21) defines a trust region that ensures descent for the optimization iteration. λ_k, Δ are method parameters.

- **Quasi-Newton approximation:** The second order term Q_k is approximated by quasi-Newton update formulae (A.8) or (A.9), e.g.

  $$A_{k+1} = A_k - \frac{W_k g_k g_k^T W_k}{g_k^T W_k g_k} + \frac{d_k d_k^T}{d_k^T g_k}, \tag{A.22}$$

  where $W_k = J_{k+1}^T J_{k+1} + A_k$, $g_k = J_{k+1}^T f_{k+1} - J_k^T f_k$ and $d_k = x_{k+1} - x_k$.

Both Gauss-Newton method and LM method are based on the assumption that $J_k^T J_k$ provides a good approximation of Q_k. In Gauss-Newton method, if $J(x)$ is rank-deficient, the algorithm is not defined, and the searching process is trapped into bad estimation of the optimum. LM method overcomes this drawback by introducing trust region strategy. For the large-residual problems (large residual $\|f(x^*)\|$ at optimum x^*), both the methods can fail to converge, and one has to consider the second order information and tries quasi-Newton approximation method.

A.6 On Sequential Quadratic Programming (SQP)

[2] The basic idea of SQP is to approximate (A.1) at a given iteration x_k by a quadratic programming subproblem, and then to determine the next iteration x_{k+1} using the solution

[2]The presentation of SQP follows the formulation in [133], [12]

of the subproblem around x_k.

For equality constrained problem

$$\begin{array}{ll} \min\limits_{x \in R^n} & f(x) \\ s.t. & c(x) = 0, \end{array} \tag{A.23}$$

define its Lagrangian function as

$$\mathcal{L}(x, \lambda) = f(x) - \lambda^T c(x). \tag{A.24}$$

The stationary point of $\mathcal{L}(x, \lambda)$ satisfies the first order necessary condition of optimization problem (A.23),

$$\begin{array}{rcl} \nabla f(x) - \nabla c(x)^T \lambda & = & 0, \\ -c(x) & = & 0, \end{array} \tag{A.25}$$

where $\nabla c(x) = \left(\nabla c_1(x), \quad \ldots, \quad \nabla c_m(x)\right)^T$ is the Jacobian matrix of vector-valued function $c(x)$. The Newton-Raphson step $\begin{pmatrix} \hat{x}_k & \hat{\lambda}_k \end{pmatrix}$ for solving (A.25) satisfies

$$\begin{pmatrix} W(x_k, \lambda_k) & -\nabla c(x_k)^T \\ -\nabla c(x_k) & 0 \end{pmatrix} \begin{pmatrix} \hat{x}_k \\ \hat{\lambda}_k \end{pmatrix} = - \begin{pmatrix} \nabla f(x_k) - \nabla c(x_k)^T \lambda_k \\ -c(x_k) \end{pmatrix}, \tag{A.26}$$

where $W(x, \lambda) = \nabla^2 f(x) - \sum\limits_{i=1}^{m} (\lambda)_i \nabla^2 c_i(x)$. Rewrite (A.26) as

$$\begin{array}{rcl} W(x_k, \lambda_k)\hat{x}_k + \nabla f(x_k) & = & \nabla c(x_k)^T [\lambda_k + \hat{\lambda}_k], \\ c(x_k) + \nabla c(x_k)\hat{x}_k & = & 0, \end{array} \tag{A.27}$$

thus $\hat{x}_k$ is the Kuhn-Tucker point of Quadratic Program **QP1**,

$$\begin{array}{ll} \min & d^T \nabla f(x_k) + \frac{1}{2} d^T W(x_k, \lambda_k) d \\ s.t. & c(x_k) + \nabla c(x_k) d = 0, \end{array} \tag{A.28}$$

Thus Newton methods (A.26) for problem (A.23) is equivalent to solve a sequence of problem **QP1**. Similarly, considering inequality constrained problem,

$$\begin{array}{lll} \min\limits_{x \in R^n} & f(x) & \\ s.t. & c_i(x) = 0 & i \in \mathcal{E} = \{1, \ldots, m_e\}, \\ & c_i(x) \geq 0 & i \in \mathcal{I} = \{m_e + 1, \ldots, m\}, \end{array} \tag{A.29}$$

the corresponding quadratic problem is as the following Quadratic Program **QP2**,

$$\begin{array}{lll} \min & d^T \nabla f(x_k) + \frac{1}{2} d^T W(x_k, \lambda_k) d & \\ s.t. & c_i(x_k) + \nabla c_i(x_k) d = 0 & i \in \mathcal{E}, \\ s.t. & c_i(x_k) + \nabla c_i(x_k) d \geq 0 & i \in \mathcal{I}. \end{array} \tag{A.30}$$

The new iterate x_{k+1} is generated by taking a step from x_k in the direction d_k that is the solution of **QP2**. The update of the multipliers λ_k is estimated by

$$\lambda_{k+1} = \lambda_k + \alpha(\lambda_{\text{qp}} - \lambda_k), \tag{A.31}$$

where α is the update steplength, and λ_{qp} is the lagrangian multiplier of **QP2**.

The nonlinear constrained problem (A.29) is thus solved by a sequence of subproblem **QP2**. The basic algorithm of SQP can be stated as

algorithm [12]

Given x_0, λ_0, A_0 and a merit function Ψ, $k = 0$,
1. Solve **QP2** for d_k, λ_k.
2. Choose steplength α so that
$$\Psi(x_k + \alpha d_k) < \Psi(x_k).$$
3. Update as
$$x_{k+1} = x_k + \alpha d_k,$$
$$\lambda_{k+1} = \lambda_k + (\lambda_{\text{qp}} - \lambda_k).$$
4. Stop if converged.
5. Compute A_{k+1}.
6. Set $k := k + 1$, go to step 1. ■

Here Hessian matrix $W(x_k, \lambda_k)$ of **QP2** is approximated by A_k. In this thesis, we adopt subroutine 'fmincon' in MATLAB optimization toolbox 2.0 for the numerical optimization. The implementation of SQP basic algorithm involves the choice of merit function Ψ, the determination of steplength α, and the update of A_k. In 'fmincon', BFGS update formula (A.9) is used.

Bibliography

[1] Alfonseca M., Ortega A., *A study of the representation of fractal curves by L systems and their equivalences*, IBM Journal of Research and Development, Vol. 41:6, pp. 727-736, 1997

[2] Allen R.G., Pereira L.S., Raes D., Smith M., *Crop evapotranspiration guidelines for computing crop water requirements*, FAO Irrigation and Drainage Paper No. 56, Rome, 299 pp, 1998

[3] Barthélémy D., Caraglio Y., Costes E., *La notion d'age physiologique des méristèmes chez les végétaux*, in : J. Bouchon, P. de Reffye et D. Barthélémy (Eds), Modélisation et simulation de l'architecture des végétaux. Science Update, INRA Editions, pp. 89-136, 1997

[4] Bellman R., *Dynamic Programming*, Princeton Univ. Press, 1965

[5] Bernoulli J., *Problema novum ad cujus solutionem mathematici invitantur*, Acta Eruditantur, Leipzig, juin 1696

[6] Berthaud S., *Résolution numérique de problèmes d'optimisation dynamique en économie de l'environnement à l'aide du logiciel scientifique SCILAB*, Rapport de stage, CIRED and CEREVE, 2000

[7] Bertin N., Gary C., *Evaluation d'un modéle dynamique de croissance et de développement de la tomate (Lycopersicon esculentum Mill), TOMGRO, pour différents niveaux d'offre et de demande en assimilates*, agronomie 13, pp. 395-405, 1993

[8] Bertin N., Heuvelink E., *Dry-matter production in a tomato crop: comparison of two simulation models*, Joural of Horticultural Science 68 (6), pp. 995-1011, 1993

[9] Blaise F., Barczi J.-F., Jaeger M., Dinouard P., de Reffye, P., *Simulation of the growth of plants. Modeling of metamorphosis and spatial interactions in the architecture and development of plants*, in: Cyberworlds, T.L. Kunii and A. Luciani (Eds). John Wiley & Sons, Ltd, Tokyo, Japon, pp. 81-109, 1998

[10] Bischof C., Carle A., Corliss G., Griewank A., Hovland P., *ADIFOR - Generating Derivative Codes from Fortran Programs*, Scientific Programming, no. 1, pp. 1-29, 1992

[11] Bischof C., Roh L., Mauer A., *ADIC – An Extensible Automatic Differentiation Tool for ANSI-C*, Software – Practice and Experience, Vol. 27, No. 12, pp. 1427-1456, 1997

[12] Boggs P.T., Tolle J.W., *Sequential Quadratic Programming*, Acta Numerica, pp.1-52, 1995

[13] Boltyanski V., *Sufficient conditions for Lagrange, Mayer, and Bolza optimization problems*, Mathematical Problems in Engineering, vol. 7, pp. 177-203, 2001

[14] Bouttier F., Courtier P., *Data assimilation concepts and methods*, Lecture note of ECMWF, March 1999

[15] Brisson N., Wery J., *Introduction aux modèles de culture*, Ecole Chercheur Modélisation Le Croisic, 14-19, October 2002

[16] Brisson N. et al., *STICS : a generic model for the simulation of crops and their water and nitrogen balances, I. Theory and parameterization applied to wheat and corn*, agronomie 18, pp. 311-346, 1998

[17] Brouwer C., Heibloem H. *Irrigation Water Needs*, FAO Irrigation and water management training manuls, Training manual no. 3, 1986

[18] Byrd R.H., Lu P., Nocedal J., Zhu C., *A limited memory algorithm for bound constrained optimization*, Technical Report NAM-08, Northwestern University, 1994

[19] Caraglio Y., Barthélémy D., *Revue critique des termes relatifs à la croissance et à la ramification des tiges des végétaux vasculaires*, J. Bouchon, P. de Reffye et D. Barthélémy (Eds), Modélisation et simulation de l'architecture des végétaux. Science Update, INRA Editions, pp. 11-87, 1997

[20] Challa H., *Modeling for crop growth control*, In ISHS Acta Horticulturae 248: International Symposium on Models for Plant Growth, Environmental Control and Farm Management in Protected Cultivation, 1989

[21] Chelle M., *Phylloclimate or the climate perceived by individual plant organs: What is it? How to model it? What for?*, New Phytologist 166/3, June 2005

[22] Chemouny S., *Estimation des paramètres du modèle de croissance et d'architecture végétale AMAPpara. Application au cotonnier taillé*, DEA, Académie de Montpellier, Université des Sciences et Techniques du Languedoc, 1997

[23] Ciarlet P.G., *Introduction à l'analyse numérique, matricielle et à l'optimization*, Masson, 1985

[24] Clark J.Y., Warwick K., *Artificial Keys for Botanical Identification using a Multilayer Perceptron Neural Network (MLP)*, Artificial Intelligence Review, Vol. 12, No. 1-3, pp. 95-115, 1998

[25] Culioli J.-C., *Introduction à l'optimisation*, Ecole des Mines de Paris, Ellipses, 1994

[26] de Reffye Ph., *Modélisation de l'architecture des arbres par des processus stochastiques. Simulation spatiale des modèles tropicaux sous l'effet de la pesanteur. Application au Coffea robusta*, Thèse Doct. Etat, Univ. Paris-Sud Centre d'Orsay, 195p, 1979

[27] de Reffye Ph., Edelin C., Françon , Jaeger M., Puech C., *Plant models faithful to botanical structure and development*, ACM SIGGRAPH Computer Graphics, v.22 n.4, pp.151-158, 1988

[28] de Reffye Ph., Houllier F., Blaise F., Barthélémy D., Dauzat J., Auclair D., *Modélisation et simulation de la croissance d'une architecture végétale : une approche morphogénétique expérimentale*, Journées du Programme Environnement, Vie et Sociétés du CNRS, Coordinateur: Blasco F., pp. 91-112, 1997

[29] de Reffye Ph., Fourcaud T., Blaise F., Barthélémy D., Houllier F., *A functional model of tree growth and tree architecture*, Silva Fennica 31 (3), pp. 297-311, 1997

[30] de Reffye Ph., Blaise F., Chemouny S., Jaffuel S., Fourcaud T., Houllier F., *Calibration of hydraulic growth model on the architecture of cotton plants*, Agronomie, 19, pp. 265-280, 1999

[31] de Reffye Ph., Le Roux J., Yan H.-P., Kang M.-Z., *Etude du comportement en croissance du modèle de plante GreenLab*, LIAMA report

[32] de Reffye Ph., Goursat M., Quadrat J.P., Hu B.-G., *The dynamic equations of the tree morphogenesis GreenLab model*, INRIA report, No. 4877, 2003

[33] de Reffye P., Hu B.-G., *Relevant qualitative and quantitative choices for building an efficient dynamic plant growth model: GreenLab Case*, In: Plant Growth Modeling and Applications: Proceedings - PMA03, Hu B.-G. and Jaeger M., (eds.), Tsinghua University Press and Springer, Beijing, China, 2003, pp. 87-107

[34] de Visser P.H.B., Marcelis L.F.M., Heuvelink E., *Functional-structural modelling of Chrysanthemum*, proceedings FSPM'04, pp. 307-310, 2004

[35] de Wit C.T., *Dynamic conceps in biology*, in: Setlik I. (Ed.), Prediction and Mesasurement of Photosynthetic Activity, Pudoc, wageningen, pp. 17-23, 1970

[36] Drouet J.-L., Pagès L., *GRAAL: a model of GRowth, Architecture and carbon ALlocation during the vegetative phase of the whole maize plant, model description and parameterisation*, Ecological Modelling, 165, pp.147-173, 2003

[37] Farkas I. (Chair), Cluster 4: Life Support Systems; Technical Area (4a): Modelling and Control in Agricultural Processes, in: *Proceedings of the 15th IFAC World Congress on Automatic Control*, Barcelone, Espagne, 2002

[38] Farquhar G.D., von Caemmerer S., Berry J.A., *A biochemical model of photosynthetic CO_2 assimilation in leaves of C_3 species*, Planta, 149, pp.78-90, 1980

[39] Faure C., Papegay Y., *Odyssée user's guide*, Version 1.7, 1998

[40] Fourcaud Th., Blaise F., Barthélémy D., Houllier F., de Reffye Ph., *A physiological approach for tree growth modelling in the software AMAPpara*, Proc. Second Workshop IUFRO WP S5.01-04, 1997

[41] François C., Cayrol P., Kergoat L. et Moulin S., *Assimilation techniques of remote sensing measurements into vegetation models : overview, limits and promises*, Proceedings du $8^{éme}$ Symposium International Mesures Physiques et Signatures en Télédétection, Aussois, France, ISPRS-CNES. pp. 649-658, 8-12 Janvier 2001

[42] Gary C., Jones J.W., Tchamitchian M., *Crop modelling in horticulture: state of the art*, Scientia Horticulturae 74, pp. 3-20, 1998

[43] Ghil M., Cohn S., Tavantzis J., Bube K., Isaacson E., *Applications of estimation theory to numerical weather prediction*, In Dynamic Meteorology: Data Assimilation Methods, Bengtsson L., Ghil M. and E. Källén (Eds.), Springer Verlag, pp. 139-224, 1981

[44] Ghil M., *The essence of data assimilation, or why combine data with models*, Inaugural lecture, Proceedings of the third international symposium on assimilation of observations in meteorology and oceanography, Québec City, Cannada, 7-11 June 1999

[45] Gill P.E., Murray W., Wright M.H., *Practical optimization*, Academy Press, 1981

[46] Gill P.E., Murray W., Saunders M.A., Wright M.H., *Computing forward-difference intervals for numerical optimization*, SIAM J. Sci. Stat. Comput. 4, pp. 310-321, 1983

[47] Giaquinta M, Hildebrandt S, *Calculus of Variations*, Springer, 1996

[48] Giavitto J.-L., Godin C., Michel O., Prusinkiewicz P., *Computational models for integrative and developmental biology*, LaMI research report no. 72-2002, Universit d'Evry Val d'Essonne, 2002

[49] Giering R., Kaminski T., *Recipes for Adjoint code construction*, ACM trans. on math. software, vol. 24, no. 4, pp. 437-474, 1998

[50] Godin C., Caraglio Y., *A multiscale model of plant topological structures*, Journal of Theoretical Biology 191, pp. 1-46, 1998

[51] Griesse R., Walther A., *Evaluating gradients in optimal control: continuous adjoints versus automatic differentiation*, Journal of Optimization Theory and Applications, Vol. 122, No. 1, pp. 63-86, July 2004

[52] Griewank A., *On Automatic Differentiation*, in Mathematical Programming: Recent Developments and Applications, edited by M. Iri and K. Tanabe, pp. 83-108, Kluwer Academic Publishers, Amsterdam, 1989

[53] Griewank A., Juedes D., Utke J., *ADOL–C, A Package for the Automatic Differentiation of Algorithms Written in C/C++*, ACM Trans. Math. Software, Vol. 22, No. 2, pp. 131-167, 1996

[54] Griewank A. *Evaluating Derivatives : Principles and Techniques of Algorithmic Differentiation*, SIAM book on Frontiers in Applied Mathematics, 2000

[55] Godin G., Hanan J., Kurth W., Lacointe A., Takenaka A., Prusinkiewicz P., de Jong T., Beveridge C., Andrieu B., *Proceedings - 4th International Workshop on Functional-Structural Plant Models*, Campus ENSAM/INRA2, place Viala Montpellier, France, 07-11 June 2004

[56] Hadamard J., *Lectures on the Cauchy problem in linear partial differential equations*, Yale University Press, 1923

[57] Hallé F., Oldeman R.A.A., Tomlinson P.B., *Tropical trees and forests*, Springer Verlag, Berlin, 441p, 1978

[58] Honda H., Fisher J.B., *Tree branch angle: maximizing effective leaf area*, Science 199, pp. 888-890, 1978

[59] Howell T.A., Music J.T., *Relationship of dry matter production of filed crops to water consumption*, Les besoins en eau des cultures, Conférence internationale, Paris, 11-14 Sept. 1984

[60] Jaeger M., de Reffye Ph., *Basic concepts of computer simulation of plant growth*, J. Biosci., vol. 17 no. 3, pp. 275-291, 1992

[61] Jallas E., Martin P., Sequeira R., Turner S., Cretenet M., Gérardeaux E., *Virtual COTONS ©, the firstborn of the next generation of simulation models*, Virtual Worlds 2000, pp. 235-244

[62] Jallas E., Martin-Clouaire R., Sequerira R.A., Martin P., Nowak R., Crétenet M., Intelligence artificielle et aide à la decision en agriculture, In E. Malézieux, G. Trébuil et M. Jaeger (Eds), Modélisation des agroécosystémes et aide à la decision, pp. 217-240, 2001

[63] Kang M.-Z., de Reffye Ph., Barci J.-F., Hu B.-G., *Fast algorithm for stochastic tree computation*, International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision 11, WSCG 2003

[64] Kozlowski J., Konarzewski M., *Is West, Brown and Enquist's model of allometric scaling mathematically correct and biologically relevant?*, Functional Ecology 18, pp. 283-289, 2004

[65] Kurth W., Soboda B., *Growth grammars simulating trees - an extension of L-systems incorporating local variables and sensitivity*, Silva Fennica 31, pp. 285-295, 1997

[66] Langton C.G., *Studying Artificial Life with cellular automata*, Physica D 22, pp. 120-149, 1986

[67] Lauvernet C., *Assimilating high temporal frequency SPOT data to describe canopy functioning*, Thesis LMC/IMAG, IDOPT/ADAM project, 2004

[68] Le Dimet F.-X., *A general formalism of variational analysis*, CIMMS Report No. 22, 1982

[69] Le Dimet F.-X., Talagrand O., *Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects*, Tellus, 38A, pp. 97-110, 1986

[70] Le Dimet F.-X., Navon I.M., *Variational and Optimization Methods in Meteorology: A Review*, SCRI report No. 144 , 83pp, 1988

[71] Le Dimet F.-X., *Assimilation de données : une approache globale de la modélisation*, in *Tendances nouvelles en modélisation pour l'environnement*, Journées du Programme Environnement, Vie et Sociétés du CNRS, Coordinateur: Blasco F., pp. 351-359, 1997

[72] Le Dimet F.-X., Navon I.M., Daescu D.N., *Second Order Information in Data Assimilation*, Monthly Weather Review, 130, No. 3, pp. 629-648, 2002

[73] Le Roux J., *Mean and Variance of stochastic trees*, LIAMA report, November 2001

[74] Lemmon H. E., *COMAX: an expert system for cotton crop management*, Science 233, pp. 29-33, 1986

[75] Lions J.-L., *Contrôle optimal de sysémes gouvernés par des équations aux dérivées partielles*, 1968

[76] Lions J.-L., *Quelques méthodes de résolution des problèmes aux limites non linéaires*, 1969

[77] Lions J.-L., *Distributed systems with incomplete data and problems of environment - Some remarks*, In *Mathematics Climate and Environment*, Diaz J.-I., Lions J.-L. (Eds), pp. 58-101, 1993

[78] Lions J.-L., *Modélisation mathématique et environnement - Quelques remarques*, In *Tendances nouvelles en modélisation pour l'environnement*, Journées du Programme Environnement, Vie et Sociétés du CNRS, Coordinateur: Blasco F., pp. 79-90, 1997

[79] Mäkelä A., Givnish T.J., Berninger F., Buckley T.N., Farquhar G.D., Hari P., *Chanllenges and opportunities of the optimality approach in plant ecology*, Silva Fennica 36 (3), pp. 605-614, 2002

[80] Mathieu A., Cournéde P.-H., de Reffye Ph., *The influence of photosynthesis on the number of metamers per growth unit in GreenLab model*, proceedings FSPM'04, pp. 248-252, 2004

[81] Mazauric C., *Assimilation de données pour les modèles d'hydraulique fluviale. Estimation de paramétres, analyse de sensibilité et décomposition de domaine*, Thése au sein du IDOPT projet, LMC/IMAG, Université Joseph Fourier, 2003

[82] Mech R., Prusinkiewicz P., *Visual Models of Plants Interacting with Their Environment*. Proceedings of SIGGRAPH 96, pp. 397-410, 1996

[83] Minchin P.E.H, Thorpe M.R., Farrar J.F., *A simple mechanistic model of phloem transport which explains sink prioriy*, Journal of Experimental Botany 44, pp. 947-955, 1993

[84] Monteith J.L., *Solar radiation and productivity in tropical ecosystems*, J. Appl. Ecol. 9, pp. 747-766, 1972

[85] Moré J.J., *The Levenberg-Marquardt algorithm: implementation and theory*, in G.A. Watson (Ed), Lecture Notes in Mathematics 630: Numerical Analysis, Springer-Verlag, Berlin, pp. 105-116, 1978

[86] Nosenzo R., de Reffye P., Blaise F., Le Dimet F.-X., *Principes de l'optimisation des modes de conduites culturales avec les modèles mathématiques de plantes*, in Modélisation des agro-écosystémes et aide à la décision, CIRAD, Collection Repères, Montpellier, France, pp. 145-172, 2001

[87] Penning de Vries F.W.T., Teng P.S., Metselaar K. (Eds), *Systems Approaches for Agricultural Development*, Proceedings of the International Symposium on Systems Approaches for Agricultural Development, 2-6 December 1991, Bangkok, Thailand

[88] Perttunen J., Sievänen R., Nikinmaa E., Salminen H., Saarenmaa H., Väkevä J., *LIGNUM: a tree model based on simple structural units*, Annals of Botany, 77, pp. 87-98, 1996

[89] Pontryagin L.S., Boltyanskii V.G., Gamkrelidze R.V., Mishchenko E.F., *The Mathematical Theory of Optimal Processes*, Interscience Publishers, 1962

[90] Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P., *Numerical Recipes in C: The Art of Scientific Computing*, Second Edition, 1992

[91] Prusinkiewicz P., Lindenmeyer A., Hanan, J., *Developmental Models Of Herbaceous Plants For Computer Imagery Purposes*, ACM SIGGRAPH Computer Graphics, v.22 n.4, pp.141-150, 1988

[92] Prusinkiewicz P., *Visual models of morphogenesis*, in Artifial Life: an Overview, C.G. Langton (Ed), MIT press, pp. 61-74, 1995

[93] Prusinkiewicz P., *In search of the right abstraction: The synergy between art, science, and information technology in the modeling of natural phenomena*, in: C. Sommerer and L. Mignonneau (Eds.): Art Science. Springer, Wien, pp. 60-68, 1998

[94] Prusinkiewicz P., Remphrey W.R., *Characterization of architectural tree models using L-systems and Petri nets*, in M. Labrecque (Ed): L'arbre - The Tree 2000: Papers presented at 4th International Symposium on the Tree pp. 177-186

[95] Rey H.R., *Utilisation de la modélisation 3D pour l'analyse et la simulation du développement et de la croissance végétative d'une plante de tournol en conditions environnementales fluctuantes (température et royonnement)*, Ph.D Thèse à l'École Nationale Supérieure Agronomique de Montpellier, 2003

[96] Rivals P., *Essai su la croissance des arbres et sur leurs systémes de floraison*, Journée d'Agriculture Tropicale et de Botanique appliquée, vol. XII (12), vol. XII (1-2-3), vol. XIV, 1965

[97] Rosen R., *Dynamical system theory in biology*, Wiley-Interscience, 1970

[98] Sasaki Y.K., *Some basic formalisms in numerical variational analysis*, Monthly Weather Review 98, pp. 857-883, 1970

[99] Sachs T., *Self-organization of tree form: a model for complex social systems*, Journal of Theoretical Biology 230, 197-202, 2004

[100] Seginer I., *Optimal control of the greenhouse environment : an overview*, Acta Horticulturae 406, pp. 191-198, 1996

[101] Seginer I., Ioslovich I., *Seasonal Optimization of Greenhouse environment for a Simple two-stage Crop Growth Model*, J. Agric. Engng. Res. 70, pp. 145-155, 1998

[102] Shinozaki K., Yoda K., Hozumi K., Kira T., *A quantitative analysis of plant form - the pipe model theory*, Japanese Journal of Ecology 14, pp. 97-105, 1964

[103] Sinoquet, H., Le Roux X., Adam B., Ameglio T., Daudet F.-A., *RATP: a model for simulating the spatial distribution of radiation absorption, transpiration and photosynthesis within canopies: application to an isolated tree crown*, Plant, Cell and Environment 24, pp. 395-406, 2001

[104] Sievanen R., Nikinmaa E., Nygren P., Ozier-Lafontaine H., Perttunen L., Hakula H., *Components of functional-structural tree models*, Annals of Forestry Science 57, pp. 399-412, 2000

[105] Soler C., Sillion F.X., Blaise F., de Reffye Ph., *A physiological Plant Growth Simulation Engine Based on Accurate Radiant Energy Transfer*, INRIA Report No. 4116, 31p, 2001

[106] Soler C., Sillion F.X., Blaise F., de Reffye Ph., *An efficient instantiation algorithm for simulating radiant energy transfer in plant models* ACM Trans. Graph. 22 (2) pp. 204-233, 2003

[107] Spaeth S.C., Sinclair T.R., *Linear increase in soybean harvest index during seedfilling*, Agronomy Journal 77, pp. 207-211, 1985

[108] Stöckle O.S., Martin S., Campbell G.S., *CropSyst, a cropping systems model: water/nitrogen budgets and crop yield*, Agric. Syst. 46, pp. 335-359, 1994

[109] Takakura T., Ohara G., Nakamura Y., *Direct digital control of plant growth iii analysis of the growth and development of tomato plants*, in ISHS Acta Horticulturae 87: Symposium on Potential Productivity in Protected Cultivation, 1978

[110] Tap R.F., *Economics-based Optimal Control of Greenhouse Tomato Crop Production*, Thesis Wageningen University, Netherlands, 2000

[111] Thornley J.H.M., *Modelling allocation with transport/conversion processes*, Silva Fennica 31 (3), pp. 341-355, 1997

[112] Thornley J.H.M., *Modelling shoot:root relations: the only way forward?*, Annals of Botany 81, pp. 165-171, 1998

[113] Thornley J.H.M., *Dynamic model of leaf photosynthesis with acclimation to light and nitrogen*, Annals of Botany 81, pp. 421-430, 1998

[114] Thornley J.H.M., Johnson I.R., *Plant and Crop Modelling: An Mathematical Approach to Plant and Crop Physiology* , Blackburn Press 2000, first print 1990

[115] Tyree M.T., Ewers F.W. *The hydraulic architecture of trees and other woody plants*, New Phytologist, Tansley Review 119, No. 34, pp. 345-360, 1991

[116] van Henten E.J., *Greenhouse climate management: an optimal control approach*, Thesis Wageningen University, Netherlands, 1994

[117] van Straten G., Challa H., van de Braak N.J., *Towards integration*, Chapter 6 in: J.C. Bakker et al. (eds.) Greenhouse climate control - an integrated approach. Wageningen Pers, pp. 249-261, 1995

[118] Vehreschild A., *Semantic Augmentation of MATLAB Programs to Compute Derivatives*, Thesis of Institute for Scientific Computing, RWTH Aachen University, 2001

[119] Verma A., *Structured automatic differentiation*, Ph.D thesis, Cornell University, 1998

[120] Wang E., Engel T., *SPASS: a generic process-oriented crop model with versatile windows interfaces*, Environmental Modeling & Software 15, pp. 179-188, 2000

[121] West G.B., Brown J.H., Enquist B.J., *A General Model for the Origin of Allometric Scaling Laws in Biology*, Science 276, pp. 122-126, 1997

[122] West G.B., Brown J.H., Enquist B.J., *A general model for the structure and allometry of plant vascular systems*, Nature 400 (12), pp. 664-667, 1999

[123] Warren-Wilson J., *Control of crop processes*, in Crop processes in controlled environment, A.R. Rees, K.E. Cockshul, D.W. Hand, R.G. Hurd (Eds), Academic Press, London, pp. 7-30, 1972

[124] Williams J.R., Jones C.A., Kiniry J.R., Spanel D.A. *The EPIC crop growth model*, Trans. Am. Soc. Agric. Eng. 32, pp. 497-511, 1989

[125] Wolfram S., *Cellular Automata as Models of Complexity*, Nature 311, pp. 419-424, 1984

[126] Wolfram S., *Approaches to complexity engineering*, Physica D, 22, pp. 385-399, 1986

[127] Wu L., de Reffye Ph., Le Dimet F.-X., Hu B.-G., *Optimization of source-sink relationships based on a plant functional-structural model - a case study on maize*, Plant Growth Modeling and Applications: Proceedings - PMA03, Hu B.-G. and Jaeger M., (eds.), Tsinghua University Press and Springer, Beijing, China, 2003, pp. 285-295

[128] Wu L., Le Dimet F.-X., Hu B.-G., Cournède P.-H., de Reffye Ph., *A water supply optimization problem for plant growth based on GreenLab model*, Proceedings of the 7th African Conference on Research in Computer Science, 2004 November, pp. 101-108

[129] Yan H.-P., Barci J.-F., de Reffye Ph., Hu B.-G., Jaeger M., Le Roux J., *Fast algorithms of plant computation based on substructure instances*, International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision 10, WSCG 2002, pp. 145-153

[130] Yan H.-P., Kang M.-Z., de Reffye P., Dingkuhn M., *A dynamic, architectural plant model simulating resource-dependent growth*, Annals of Botany 93, pp. 591-602, 2004

[131] Yang S., Tyree M.T., *Hydraulic resistance in Acer sacccharum shoots and its influence on leaf water potential and transpiration*, Tree Physiolo. 12, pp. 231-242, 1993

[132] Yang J.-Q., *Assimilation de données variationalle pour les problèmes de transport des sédiments en rivière*, Thése au sein du IDOPT projet, LMC/IMAG, Université Joseph Fourier, 1999

[133] Yuan Y.-X., Sun W.-Y., *Optimization theory and methods*, in Chinese, Academy Press China, 2001

[134] Zhan Z., Wang Y., de Reffye Ph., Wang W., Xiong Y., *Architectural modeling of wheat growth and validation study*, 2000 ASAE Annual International Meeting, Milwaukee, Wisconsin, July 9-12, 2000

[135] Zhan Z., de Reffye Ph., Houllier F., Hu B.-G., *Fitting a functional-structural growth model to plant architecture data*, Plant Growth Modeling and Applications: Proceedings - PMA03, Hu B.-G. and Jaeger M., (eds.), Tsinghua University Press and Springer, Beijing, China, 2003

[136] Zhao X., de Reffye Ph., Barthélémy D., Hu B.-G., *Interactive simulation of plant architecture based on a dual-scale automaton model*, Plant Growth Modeling and Applications: Proceedings - PMA03, Hu B.-G. and Jaeger M., (eds.), Tsinghua University Press and Springer, Beijing, China, 2003

[137] Zou X., Vandenberghe F., Pondeca M., Kuo Y.-H., *Introduction to adjoint techniques and the MM5 adjoint modeling system*, NCAR technical note, 1997

Variational Methods Applied to Plant Functional-Structural Dynamics: Parameter Identification, Control and Data Assimilation

Abstract

The thesis is devoted to a unified variational approach for diverse applications, such as parameter calibration, optimal control and data assimilation, based on plant architecture and functioning. The mathematical formulation of the functional-structural plant model GreenLab is completed by the introduction of an empirical formula on environmental factors to mimic photosynthesis. A soil water balance submodel has been integrated into GreenLab to descript the dynamic soil-plant system. The dynamics formulation enables efficient numerical solutions for the variational systems by introducing the corresponding adjoint model. Differentiation algorithms are employed to derive adjoint code by hand in a systematic way directly from GreenLab source code. This variational approach is followed to solve an optimal control problem of sunflower water supply for better fruit production. Data assimilation concept is introduced to decrease the model uncertainties both in initial conditions and external model parameters. The defined problems and optimal control techniques proposed in this thesis reveal possible agronomic applications.

Key words: Functional-Structural Plant Model (FSPM), adjoint model, differentiation algorithms, optimal control, data assimilation

Méthodes Variationelles pour des Modèles Fonction-Structure de Plantes : Identification de Paramètre, Contrôle et Assimilation de Données

Resumé

La thèse est consacrée à une approche variationnelle unifiée pour des applications diverses, telles que l'identification de paramètres, la contrôle optimal et l'assimilation de données, pour la modèlisation de l'architecture et du fonctionnement des plantes. La formulation mathématique du modèle fonction-structure de plantes GreenLab est réalisé par l'introduction d'une formule empirique sur des facteurs environnementaux pour modèliser la photosynthèse. Un sous-modèle d'équilibre de l'eau dans le sol a été ajouté dans GreenLab pour le système dynamique de sol-plantes. La formulation dynamique permet d'obtenir des solutions numériques efficaces pour les systèmes variationnels en utilisant le modèle d'adjoint correspondant. Les algorithmes de différentiation sont utilisées pour différentier le code GreenLab d'une manière systématique afin d'obtenir le code d'adjoint. L'approche variationnelle est utilisée pour résoudre un problème de d'approvisionnement optimal d'eau pour le tournesol et pour une meilleure production de fruits. Le concept de l'assimilation de données est utilisé pour diminuer les incertitudes sur la condition initiale et les paramètres externes de modèles. Les resultats sur les problèmes étudiés montrent que les concepts d'assimilation de données et de contrôle optimal sont utilisables en agronomie.

Mots clefs: modèle fonction-structure de plantes, modèle adjoint, algorithmes de différentiation, contrôle optimal, assimilation de données