On reliability and flexibility of scientific software in environmental science : towards a systematic approach to support decision-making

June Sallou

UNIVERSITÉ DE RENNES 1

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES 1

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par
June SALLOU

On Reliability and Flexibility of Scientific Software in Environmental Science : Towards a Systematic Approach to Support Decision Making

Thèse présentée et soutenue à Rennes, le 23/02/2022
Unité de recherche : IRISA

Rapporteurs avant soutenance :

Gordon BLAIR	Professeur	Lancaster University
Betty H.C. CHENG	Professeur	Michigan State University

Composition du Jury :

Examinateurs :	Gordon BLAIR	Professeur	Lancaster University
	Jean-Michel BRUEL	Professeur	Université de Toulouse
	Betty H.C. CHENG	Professeur	Michigan State University
	Anne-Cécile ORGERIE	Chargée de Recherche	CNRS
	Serge STINCKWICH	Head of Research	United Nations University Institute in Macau
Dir. de thèse :	Johann BOURCIER	Maître de conférences	Université Rennes 1
Co-dir. de thèse :	Jean-Raynald DE DREUZY	Directeur de Recherche	Université Rennes 1
Co-encadrant. de thèse :	Benoit COMBEMALE	Professeur	Université Rennes 1

Acknowledgement

The three years of my doctorate leading to this thesis have been an enlightening journey. It has brought me valuable experiences from a professional as well as a personal point of view.

For that, I would like to thank my brilliant and outstanding supervisors Johann Bourcier, Benoit Combemale and Jean-Raynald de Dreuzy. They have shown me what the academic world entails and they have passed on their passion to me. I am grateful for their patience, their guidance and their unconditional support. Of course, thank you for trusting me with this magnificent thesis' subject as the first one being in collaboration between the two teams and with such a high objective of initiating many more.

At the beginning of my PhD, I was told to find my role models in research, meaning the researchers who would make me star-struck at the thought of working with them. Well, I got these "stars in my eyes" by working on this thesis and by exchanging with the researchers in my research environment, i.e. the members of my teams and especially my supervisors. I was, and still am, amazed by their ability to grasp new concepts and to communicate, by their work ethics and above all by their passion. I can say that they have been great role models for me. They have deepened my drive to be as good a role model for other future researchers and to hopefully continue to pass on the passion for research, software engineering and environmental science. Thanks again for all that!

I want to thank the two research teams for welcoming me and especially the DiverSE team among which I spent most of my time. I took great pleasure in participating in the life of the research teams (DiverSE & DIMENV@risce), exchanging on research subjects complementary to mine as well as discovering others, and of course getting to know the personalities who make up these teams and working groups. I have been lucky enough to do a thesis that is at the interface of two fascinating disciplines that benefit from working together. I am glad to be able to be part of the movement of reflection and action for the use of technologies leading to a more sustainable society through this work. I am grateful for the working environment that was offered to me and allowed me to participate in several seminars, working groups and to meet researchers who shared their passion for their work with me. I also thank all the members for the various and numerous coffee

(or hot chocolate/tea in my case) breaks during which I have learned a lot about various subjects and laughed a lot. I enjoyed all those moments with you all. I am grateful for all the warm support you gave me, especially for the different (crazy) projects I participated in such as the Three Minute Thesis competition.

I also want to give a very special thank you to Noël Plouzeau and Olivier Barais who have trusted me and have accepted me to join the master's degree of software engineering despite (well, I would say thanks to) my background mainly focusing on environmental science, agronomy and bioinformatics. I am glad they gave me this opportunity to show my passion, my drive and my thirst to learn, to discover the world of software engineering and to merge it with the world of environmental science. I would definitely not be writing those lines if they had not acknowledged me and given me the chance to. I am hopeful that it will open up the path for students with a different background than software engineering and who want to explore it.

I would like to thank all the researchers that took time to discuss and shared their knowledge with me during seminars, follow-up committees, and various meetings over the period of my doctorate. I hope to continue the discussions alive and potential work.

On a personal view, I am most grateful to my family and particularly my mother. I hope they know how thankful I am to them for all the time and support they gave me as well as the patience they have shown. I admire you. You are the matches to my fire, I am thriving thanks to you. Mum, a very dear and special thank you to you for all the nonsense you had to deal with when I was writing this manuscript.

I do not forget my wonderful friends who have taken time to ask me questions and try to understand what my research work entailed, and who have made me sometimes feel as a superhero. You made me smiled so much thanks to that.

Finally, I would like to thank all the people who have been part of this PhD journey, and also, those who have supported and guided me along the way.

Résumé en français

Motivations

Le 9 août 2021, le Groupe d'experts intergouvernemental sur l'évolution du climat (GIEC) a publié son dernier rapport [1] qui affirme que les activités humaines ont une part substantielle dans le changement climatique, que le réchauffement de la planète est plus rapide qu'auparavant et qu'il y a une intensification des événements climatiques tels que les vagues de chaleur extrêmes et les inondations. Ces déclarations appuient l'appel des Nations Unies à prendre des mesures pour limiter le changement climatique [2]. Différentes parties prenantes sont les destinataires de ce message : les décideurs, tels que les représentants des gouvernements et des organisations, qui doivent prendre des mesures pour limiter le changement climatique et qui ont l'autorité et le pouvoir d'adopter des lois et de gérer leur territoire, ainsi que les scientifiques et le grand public qui sont impliqués et affectés par le processus décisionnel. Pour prendre des décisions et donc des mesures, ces parties prenantes doivent avoir accès à des données sur les événements liés au changement climatique et leurs impacts, ainsi que des informations sur les effets des éventuelles mesures à prendre vis-à-vis de ces événements. Les informations sur ces événements et leurs impacts peuvent aider à rendre la décision plus pertinente et donc plus durable. Par exemple, un maire a besoin de connaître les changements locaux du niveau marin et du littoral pour anticiper l'éventuelle submersion des zones côtières afin de décider de délivrer ou non un permis de construire dans une telle zone ou de faire évacuer la population.

Cependant, ces informations ne sont pas faciles à obtenir. Aussi catastrophiques que puissent être les impacts du changement climatique et du réchauffement planétaire, il n'existe actuellement aucun moyen de les caractériser et de les définir avec précision, car ces phénomènes font intervenir des processus environnementaux complexes. Il existe en effet une grande incertitude quant à la manière dont le changement climatique peut et va se manifester exactement et dans quelle mesure. Comme personne n'est en capacité d'affirmer avec certitude ce qui va se passer, les projections ne peuvent que donner quelques indications mais aussi apporter un éclairage.

La modélisation des phénomènes physiques est une solution pertinente pour apporter

des informations et faire des projections. Les modèles scientifiques, et plus encore, les logiciels scientifiques qui les intègrent, sont depuis longtemps utilisés par les scientifiques pour étudier et comprendre les phénomènes physiques. Plus particulièrement, les modèles de simulation permettent de représenter le comportement du système étudié en fonction de conditions spécifiques et dans un cadre dynamique dans le temps ou dans l'espace. Leur exécution permet une plus grande complexité et une meilleure représentation des phénomènes par les modèles grâce à des ressources avec une plus grande puissance de calcul [3]. Cela permet également de partager plus facilement les modèles et leurs résultats avec d'autres personnes concernées grâce à des services spécifiques [4], [5] (par exemple, laboratoires virtuels et cloud computing) [6], [7]. Les modèles permettent ainsi de réduire les incertitudes liées au manque de connaissance dans le domaine d'étude. Ils sont utilisés pour faire des projections dans une variété de domaines parmi lesquels on trouve les sciences de l'environnement et l'étude du changement climatique [8]. En tant que tels, ils peuvent être bénéfiques dans le contexte de la prise de décision.

Vers le soutien des modèles scientifiques pour la prise de décision

Les logiciels scientifiques (c'est-à-dire les modèles de simulation) peuvent être utilisés pour faire des projections. Mais pour qu'ils puissent être utilisés efficacement dans le contexte de la prise de décision en sciences environnementales, plusieurs autres conditions doivent être remplies.

Fiabilité des projections

Le processus décisionnel a pour but de faire évoluer l'organisation ou le système concerné. Dans le contexte du changement climatique, la décision peut être prise par le gouvernement et peut prendre la forme de lois. Cela peut, par exemple, avoir un impact sur l'aménagement du territoire avec le déplacement de la population, ou sur la vie quotidienne des citoyens avec des réglementations sur le type de transport autorisé et leur fréquence d'utilisation. Ces décisions peuvent donc avoir un impact considérable et des conséquences importantes. Il est ainsi de la responsabilité des décideurs de réfléchir aux bénéfices de leur décision par rapport à l'impact de cette dernière. Les décideurs ont besoin d'informations et de projections fiables pour les aider à prendre la décision la

plus appropriée. Les logiciels de simulation doivent donc également être considérés comme valides pour produire ces projections fiables.

Exploration des scénarios

L'incertitude entourant le changement climatique est double et peut être séparée en deux parties : l'incertitude épistémique et l'incertitude aléatoire. L'incertitude épistémique concerne le manque de connaissances précises sur les phénomènes en cause, car il n'existe pas beaucoup d'outils efficaces pour les observer et les mesurer. Dans le cas de l'étude des mouvements de la nappe phréatique entraînant des inondations, le système souterrain est encore mal connu en raison de la rareté des données pouvant être acquises, du manque d'outils pour les acquérir et du coût de leur acquisition. L'incertitude aléatoire concerne le fait que l'on ne peut pas affirmer avec certitude que les événements futurs se produiront et que la seule façon est de spéculer sur leur probabilité de se produire. Ces incertitudes ont conduit les experts en climatologie à élaborer une série de scénarios climatiques différents, des plus optimistes, dans lesquels les changements sont les plus faibles, aux plus pessimistes, dans lesquels les changements sont les plus conséquents. Les décideurs doivent donc disposer de projections pour les différents scénarios climatiques afin d'avoir un aperçu de l'éventail des possibilités concernant l'étendue des impacts des événements climatiques futurs. De plus, afin de décider si une action vaut la peine d'être entreprise, ils doivent disposer de la projection décrivant les impacts potentiels de cette action sur la limitation des effets du changement climatique. Par exemple, ils peuvent vouloir connaître les impacts de la construction d'un barrage sur la submersion de zones proches du littoral. Au total, les modèles de simulation doivent permettre d'explorer les scénarios climatiques ainsi que les scénarios concernant les actions potentielles à entreprendre.

Interactivité

Pour que les décideurs aient une vision globale des projections, l'exploration des scénarios doit être interactive. Ainsi, la génération des projections doit être rapide. Grâce à l'interactivité, les parties prenantes peuvent proposer de nouvelles potentielles actions à entreprendre et obtenir les projections du scénario correspondant. Ainsi, elles peuvent décider rapidement quelles actions méritent d'être explorées et mises en œuvre ou non. D'un point de vue pratique, cela permet également aux réunions des parties prenantes d'être efficaces car il n'y a pas besoin d'attendre longtemps pour obtenir les projections

de nouvelles idées d'actions. C'est particulièrement intéressant car il peut être difficile de rassembler toutes les parties prenantes lors d'une même réunion.

Défis à relever

Les exigences liées au processus décisionnel conduisent par extension à des exigences relatives aux modèles de simulation associés. La fiabilité des projections signifie que les simulations, c'est-à-dire les exécutions du modèle de simulation (alias le logiciel scientifique), doivent générer des résultats scientifiques fiables et crédibles par rapport au domaine d'étude. Le logiciel de simulation doit être validé pour répondre aux critères de qualité du domaine scientifique. De plus, l'exécution des modèles de simulation doit être rapide. Avec une exécution rapide des modèles, la simulation d'un seul scénario est plus rapide et cela permet d'exécuter beaucoup plus de simulations correspondant à de nombreux scénarios. Cela permet en même temps l'exploration des scénarios et l'interactivité nécessaire aux décideurs pour explorer les projections. Dans l'ensemble, les modèles de simulation doivent être validés pour garantir des projections fiables, avoir un temps d'exécution rapide et permettre l'exploration de scénarios.

Problématique

Cependant, les modèles de simulation élaborés au départ pour étudier les phénomènes peuvent ne pas répondre aux exigences susmentionnées, qui sont liées au contexte de l'aide à la prise de décision. Les modèles qui ont été élaborés par les experts sont devenus de plus en plus complexes en raison de la volonté de représenter plus fidèlement les phénomènes. Par exemple, davantage de variables ont été ajoutées pour représenter plus fidèlement les phénomènes (comme le nombre de couches pour représenter les sols dans un modèle géophysique). Parallèlement au gain de précision, cette complexité a fait augmenter le coût en calcul de l'exécution des modèles. L'exécution d'une seule simulation peut prendre beaucoup de temps (par exemple, de plusieurs heures à plusieurs jours ou semaines) [3] et nécessiter des ressources importantes. Ces contraintes de temps sont très problématiques dans le contexte de l'aide à la prise de décision [9]. En effet, un modèle de simulation qui prend trop de temps pour générer une seule projection empêche l'interactivité nécessaire aux parties prenantes, ainsi que l'exploration de scenarios. Avec une seule projection pouvant prendre des mois pour être générée, il est impossible de produire les projections

des nombreux scénarios climatiques et des multiples scénarios d'aménagement du territoire dans un délai convenable pour les décideurs.

De plus, les modèles de simulation en sciences environnementales sont généralement élaborés pour modéliser des zones spécifiques et locales. Leur élaboration est le résultat de plusieurs années de recherche et de travail sur un lieu spécifique. Les données sont collectées sur le terrain pour aider à construire le modèle et à le calibrer. En hydrogéologie, la collecte de données est une tâche difficile car le nombre d'outils permettant d'accéder aux données du sous-sol est limité et le coût du processus est élevé par rapport au nombre de données à collecter pour représenter de larges zones. Ainsi, il y a également un manque de données spatiales pour calibrer les modèles. Tout ceci limite la généralisation des projections à de plus grandes échelles et empêche de fournir une vision globale aux parties prenantes pour la prise de décision. En somme, le temps d'exécution important du logiciel de simulation qui en résulte en fait un obstacle à la satisfaction des exigences associées au contexte d'aide à la décision en sciences environnementales [10], [11]. Les modèles de simulation sont complexes et longs à exécuter et, par conséquent, généralement spécifiques à une situation précise (par exemple, un scénario climatique et un lieu) [12]. Ainsi, les modèles ne peuvent pas être utilisés tels quels pour fournir l'aide nécessaire à la prise de décision. Comme ces modèles de simulation sont le résultat d'efforts de recherche conséquents (temps, coût et connaissances), on souhaite les capitaliser. Une façon d'aborder le problème est d'adapter les modèles de simulation pour qu'ils répondent aux exigences de l'aide à la prise de décision. Il y a un changement de contexte d'utilisation des modèles de simulation et donc de nouvelles contraintes à satisfaire. L'objectif est d'adapter les propriétés des modèles de simulation qui étaient principalement utilisés dans le contexte de la recherche pour qu'ils puissent être utilisés dans le nouveau contexte de la prise de décision. Les modèles, qui sont longs à exécuter, précis avec une grande complexité et spécifiques à un scénario unique, doivent devenir rapides à exécuter, rester fiables et être flexibles en ce qui concerne l'exploration de scénarios. Ils ne doivent plus seulement décrire le monde physique, mais aussi agir sur lui à travers l'exploration des projections et la prise de décision.

Pour ce faire, le calcul des modèles de simulation doit être plus rapide. Avec une exécution plus rapide, les projections peuvent être produites plus rapidement, ce qui rend alors possible l'exploration de nombreux scénarios. Cela apporte également l'interactivité pour l'exploration des scénarios avec une diminution du temps nécessaire pour obtenir des projections. Comme les parties prenantes n'ont pas besoin de tous les détails possi-

bles dans chaque projection, nous pouvons agir sur la précision pour augmenter la vitesse d'exécution des modèles de simulation. Ce dont les parties prenantes ont besoin, c'est de disposer d'une évaluation globale des risques et des impacts grâce aux différentes projections sur les multiples scénarios climatiques. Ainsi, les contraintes sur la précision peuvent être allégées, tout en assurant la fiabilité, dans le contexte de la prise de décision. Par exemple, les parties prenantes ne voient pas l'utilité d'avoir les mouvements précis des nappes phréatiques donnés en millimètres pour chaque jour des 50 prochaines années. L'évaluation des zones où les nappes phréatiques sont susceptibles de remonter à la surface et de provoquer des inondations est le niveau d'information pertinent pour eux. Le fait d'échanger une certaine précision contre une meilleure vitesse d'exécution permet d'adapter les modèles de simulation pour aider la prise de décision.

Dans cette thèse, nous visons à adapter les modèles de simulation pour aider la prise de décision et à généraliser leur application à tout le contexte des sciences environnementales et de la recherche sur le changement climatique. Ainsi, le but n'est pas d'adapter ou de réaliser un compromis entre la précision et la vitesse d'exécution pour un modèle de simulation unique et spécifique, mais d'élaborer une approche systématique qui peut être appliquée à différents modèles de simulation du domaine. Par conséquent, une autre exigence que nous nous imposons est la généralisation de l'approche pour les modèles de simulation en sciences environnementales. Pour y parvenir, nous travaillons sur plusieurs défis. Tout d'abord, avant d'effectuer tout compromis ou adaptation sur les modèles de simulation, nous devons définir les caractéristiques des modèles scientifiques et ensuite assurer la validité des modèles. Nous devons déterminer que les modèles de simulation génèrent des projections fiables et à quel moment ils génèrent ces projections fiables (c'est-à-dire déterminer quelles sont les valeurs des données d'entrée du modèle de simulation conduisant à des données de sortie fiables). La définition de l'enveloppe de validité des modèles de simulation permet ensuite de procéder à l'adaptation grâce au compromis entre précision et flexibilité (c'est-à-dire la vitesse d'exécution). La question est de définir comment ce compromis peut être effectué pour répondre à l'exigence d'une exécution rapide des modèles de simulation tout en maintenant leur fiabilité. L'approche doit également être systématique pour permettre non seulement son application à un modèle unique et spécifique, mais aussi sa généralisation à différents modèles de simulation dans le domaine des sciences environnementales. Enfin, la dernière étape consiste à voir comment l'exploration de nouveaux scénarios est possible grâce aux modèles de simulation adaptés. Ainsi, nous décrivons comment l'approche globale composée de trois

étapes principales peut atteindre l'objectif principal d'adapter les modèles de simulation pour aider la prise de décision dans le contexte de la science environnementale et des études sur le changement climatique.

Les défis peuvent être résumés comme suit :

— **Défi 1** : Quelles sont les spécificités des modèles scientifiques et comment ces modèles sont-ils intégrés dans les logiciels scientifiques ?

— **Défi 2** : Comment réaliser de manière systématique le compromis entre précision et vitesse d'exécution tout en assurant la fiabilité des modèles de simulation ?

Contributions

Pour relever ces défis, nous procédons en plusieurs étapes qui sont organisées selon les contributions principales suivantes de cette thèse.

Démystifier les modèles scientifiques et leur complémentarité avec les modèles d'ingénierie

Tout d'abord, nous examinons les différents types de modèles qui peuvent être intégrés dans un logiciel : les modèles scientifiques, les modèles d'ingénierie et les modèles d'apprentissage automatique (c'est-à-dire les modèles empiriques). Nous examinons les spécificités et les points communs de ces modèles dans le contexte des systèmes cyber-physiques. Nous observons que malgré leurs particularités, les modèles scientifiques ont beaucoup en commun avec les modèles d'ingénierie. Les modèles scientifiques, qui sont historiquement utilisés de manière descriptive, sont maintenant également utilisés pour être prédictifs ou même prescriptifs [13] (c'est-à-dire pour produire des directives à appliquer dans le monde réel). Les modèles d'ingénierie qui ont une nature plus prescriptive intègrent de plus en plus de connaissances sur la description du monde réel. Ces observations nous incitent à considérer les modèles non pas selon leur types mais plutôt en fonction du rôle qu'ils jouent. Elles incitent également par extension à utiliser méthodes orientées vers les logiciels (et les modèles d'ingénierie) sur les modèles scientifiques. En somme, nous élaborons un cadriciel qui fournit une vision de la manière d'intégrer explicitement les trois rôles joués par les modèles - prescriptif, prédictif et descriptif - ainsi que leurs sources de données respectives, et nous mettons en évidence les actions associées pour les intégrer. Nous soulignons et illustrons la complémentarité et la dualité des différents

modèles et des données dans les systèmes socio-techniques. Ce travail collaboratif résulte d'un groupe de travail au séminaire Bellairs [14] sur le thème des modèles et des données et il a été publié dans le journal de IEEE Software (2020).

Garantir la fiabilité des logiciels scientifiques

Une fois que nous avons une vision globale du fonctionnement des modèles scientifiques, nous voulons assurer la fiabilité de ces modèles. Nous étudions la validation des logiciels scientifiques et le rôles des différentes personnes prenant part au développement de logiciels scientifiques. Nous mettons en évidence les responsabilités des différents rôles concernant les artefacts impliqués dans le développement du logiciel scientifique. Nous notons l'impact du langage de programmation utilisé pour élaborer le logiciel sur le processus de validation. Nous présentons ensuite une approche raisonnée pour le développement de logiciels scientifiques fiables qui permet de caractériser systématiquement l'enveloppe de validité de tels logiciels, de la rendre explicite et de conduire à une meilleure utilisation de ces logiciels. Nous donnons des directives aux concepteurs de langages afin de fournir des langages qui puissent aider les utilisateurs à effectuer la validation de leurs modèles scientifiques et de leurs logiciels logiciels associés, et aux utilisateurs des langages (i.e., les modélisateurs) pour assurer une validation complète des logiciels scientifiques. Le travail a été publié dans le journal de IEEE Computer dans le numéro de décembre (2021).

Le compromis entre précision et flexibilité des logiciels scientifiques

Après avoir compris les spécificités des logiciels scientifiques et le processus de validation associé, nous appliquons le calcul approximatif (une technique orientée logiciel principalement utilisée sur les modèles d'ingénierie) et l'adaptons pour réaliser un compromis entre précision et vitesse d'exécution sur les logiciels scientifiques. Nous présentons l'approche systématique qui permet de réduire le temps d'exécution des modèles de simulation tout en maintenant des résultats acceptables. Elle consiste à réduire automatiquement les itérations de la boucle principale d'un modèle de simulation en agrégeant les données spatiales ou temporelles correspondantes. Nous montrons que nous pouvons obtenir une accélération médiane de 95% en appliquant la technique sur un modèle géophysique d'une simulation hydraulique. Ce travail a été publié à la conférence internationale "International Conference On Computational Science" de 2020.

Table of Contents

Abstract

Scientific software, and more specifically simulation models, are crucial to support decision makers in deciding what is the better action to take to anticipate and tackle environmental issues such as climate change. However, their complexity make them hardly usable in such context as their execution is time-consuming or resource-demanding, which is the result of the primary goal of elaborating and using them for research. Therefore, there is a need to speed up their execution to make them match the requirements of the new context that is the support of decision making.

As the context of supporting decision making by exploring the multiple scenarios (*i.e.,* climate scenarios, geographical sites, potential solutions to enact) does not require very specific and detailed results, but rather a global risk assessment of the climate related events (*e.g.,* risk of flooding caused by the rise of the sea level and the underground water), a trade-off between accuracy and execution speed can be performed.

Several approaches to do so exist but are not satisfying as they are either time-consuming or resource-demanding. Only the approximate computing approach seems relevant to the context, but it is not adapted to scientific software. Thus, in this thesis, we aim to tailor scientific models to support decision making in environmental science by adapting approximate computing for them while ensuring their reliability.

Before performing the trade-off and tailoring, we investigate the complementarity of engineering and scientific models as well as the specificities of scientific models with regard to their validation. We propose the use of the MODA framework that helps to understand and visualize how models and data are integrated into a cyber-physical system and how they interact with each other according to the role they play. We also look at the validation of the scientific models and we propose a scientific V-Model that highlights the importance of appropriate tools to ensure the development of reliable scientific software.

The main contribution applies the guidelines introduced in the previous contributions to provide an approach, called loop aggregation, to systematically and automatically perform a trade-off between accuracy and speed for scientific models in the context of supporting decision making while ensuring their fidelity. We implement and validate the approach with a hydraulic model assessing the risk of flooding in the region of Normandy,

France. We show that a speed-up of more than 95% can be obtained.

Lastly, we investigate the optimisation of applying loop aggregation. We propose a data-driven based optimisation approach that aims to predict the conditions leading to the optimal trade-off for the exploration of new scenarios. We validate the approach with the same hydraulic model used in the main contribution and we quantify the potential error introduced by the prediction that the surrogate model can generate. We also examine the variability present in the approach and inspect several factors that can help improve the optimisation as a preliminary study.

Overall, this thesis focuses on the challenges of using scientific software to support decision making in environmental-related issues, and shows that, thanks to understanding what role the scientific model plays in the decision-making process, as well as the need for specific validation of the scientific model, the loop aggregation technique, and its associated optimisation, is a valid solution to tackle those challenges.

Chapter 1

Introduction

1.1 Context

> *"Climate change is the defining issue of our time and we are at a defining moment. From shifting weather patterns that threaten food production, to rising sea levels that increase the risk of catastrophic flooding, the impacts of climate change are global in scope and unprecedented in scale. Without drastic action today, adapting to these impacts in the future will be more difficult and costly."* – The United Nations [2]

On the ninth of August 2021, the Intergovernmental Panel on Climate Change (IPCC) published its latest report [1] that states that human activities do have a substantial part in climate change, that global warming is faster than before, and that there is an intensification of climate events such as extreme heat waves and flooding. Those statements bolster the United Nations' urges taking actions in order to limit climate change [2]. Different stakeholders are at the receiving end of that message : the decision makers, such as representatives of governments and organisations, who have the authority and power to enact laws and to manage their territory, as well as the scientists and the general public who are involved and affected by the decision-making process. To make decisions and therefore to take actions, those stakeholders need to be provided information about the climate change events and their impacts, along with information about the effects of the possible actions to be taken on those events. The information of those events and their impacts can help make the decision more pertinent and thus sustainable. For instance, a local mayor needs to know about sea level and coastline changes to anticipate the possible submersion of coastal areas to decide whether to issue a building permit in such an area or to relocate people.

However, that information is difficult to obtain. As catastrophic as the impacts of

climate change and global warming may be, there is currently no way to characterise and define them precisely, as these phenomena involve complex environmental processes. There is indeed a lot of uncertainty in how and to what extent climate change can and will manifest itself exactly. As it is not in anyone's capacity to state what will happen for sure, projections can only give some hints but also bring some insight.

Modelling the physical phenomena is a relevant solution to bring information and make projections. The scientific models, and more so, the associated scientific software that embed them, have long been used by scientists to study and understand the physical phenomena [15]. More particularly, simulation models help represent the behaviour of the system under study according to specific conditions and in a dynamic setting over time and space. Their execution enables more complexity and better representation of the phenomena in the models. Greater computational power has made this refinement possible [3]. It also allows to share the models and their results to other relevant people more easily through specific services [4], [5] (*e.g.,* virtual labs and cloud computing) [6], [7]. The models thus enable to reduce the uncertainties that are linked to the lack of knowledge in the domain of study. They are used to making projections in a variety of domains, among which are the environmental sciences and the study of climate change [8], [16]. As such, they bring a lot of potential in the context of decision making.

1.2 Towards the Support of Scientific Models for Decision Making

Scientific software (i.e., simulation models) can be used to make projections, and this addresses the core need to support the decision-making process. But for them to be effectively used in the context of decision making in environmental science, several other requirements need to be fulfilled.

1.2.1 Reliability of the Scientific Software

The decision-making process is intended to bring change within the organisation or the system concerned. In the context of climate change, the decision can be taken by the government and can take the form of laws and/or policies. This may, for example, have an impact on land use with the relocation of population, or on the daily life of citizens with regulations on the type of transport allowed and their frequency of use. Those decisions

can thus have significant impact and consequences. It is therefore the responsibility of decision makers to reflect on the benefits to be gained in relation to the impact of their decision. Decision makers therefore need dependable information and credible projections to help them make the most appropriate decision. So, the simulation software also have to be deemed valid to produce those projections with relevant fidelity to be informative and useful. The reliability of the simulation software describes the property of the software to maintain the excepted services, meaning the production of valid and credible projections. Having a high-enough reliability of the software leads to simulation projections that are deemed credible and reliable, that can then be used with trust by decision makers.

Hence, in this thesis, we use the word "reliability" (with regard to scientific software) to express the fact that the software provide the expected services, *i.e.,*producing scientifically valid projections that are usable by decision makers.

1.2.2 Exploration of Scenarios

The uncertainty around climate change is twofold and can be separated into the epistemic uncertainty and the random uncertainty. The epistemic uncertainty concerns the lack of precise knowledge about the phenomena involved, as there are not many efficient tools to observe and measure them. In the case of the study of the underground water table movements leading to flooding events, the underground system is still poorly known because of the scarcity of data that can be acquired, the lack of tools to acquire them, and the cost of acquiring them [11], [17]. The random uncertainty deals with the fact that future events cannot be stated to happen for sure, and the only way is to speculate on their probability to happen. Those uncertainties have led climate experts to elaborate a range of different climate scenarios, from the most optimistic ones in which the changes are the smallest to the most pessimistic ones in which the changes are the largest [18]. Decision makers thus need to have projections for the various climate scenarios to have an insight on the range of possibility concerning the extent of the impacts of future climate events. Moreover, to decide whether an action is worth taking, they need to have the projection describing the potential impacts of that action on limiting the effects of climate change. For instance, they may want to see the impacts of building a dam on the submersion of areas near the coastline. In all, the simulation models have to enable the exploration of the climate scenarios as well as the scenarios concerning the potential actions to be taken.

1.2.3 Interactivity

For decision makers to understand the global picture of the projections, the exploration of the scenarios needs to be interactive, and thus, the time to generate the projections has to support that requirement of interactivity. In this situation, the stakeholders can come up with new potential actions to be taken and have the projections of the related scenario at the same time. That way, they can decide in real time what actions are worth to explore and to implement or not. As a practical note, it also enables the meetings of stakeholders to be efficient as they do not need to wait for a long period of time to get the projections of new ideas of actions. It is particularly interesting as it can be challenging to gather all the different stakeholders in a same meeting.

1.2.4 Generalisation

The decision makers may need to make decisions at a scale that is different from the local scale (e.g., regional or national scales) and need projections for a wide area. The stakeholders can then have a global vision of an area that is not limited to a local and specific site. The simulation model should then enable the running of the different scenarios for different locations. Models are usually very specific to a local area (e.g., a specific catchment) as the model has been constructed thanks to field observations. That way, the stakeholders can have a global vision of an area that is not limited to a local and specific site.

1.2.5 Impacts on the Simulation Models

The requirements related to the decision-making process lead by extension to requirements that pertain to the associated simulation models. The reliability of the projections means that the simulations, *i.e.,* the executions of the simulation model (*a.k.a.* the scientific software), need to generate trustworthy results in terms of science credibility with regard to the domain of study. The simulation software must be validated to meet the quality criteria of the scientific field. Then the execution of the simulation models is required to be fast. With a fast execution of the models, the simulation of a single scenario is faster, and it makes it possible to run many more simulations corresponding to numerous scenarios. It enables at the same time the exploration of scenarios and the interactivity needed for the stakeholders to explore the projections. The exploration of new scenarios

and other locations is also of significance for decision making, thus simulation models provide that functionality in this context.

In all, simulation models need to be validated to ensure credible projections, to have a fast execution time and to enable the exploration of scenarios.

1.3 Problem Statement

However, the simulation models first elaborated in order to study the phenomena may not fulfil the aforementioned requirements, which are related to the context of supporting decision making.

The models that have been elaborated by experts have become more and more complex as a result of wanting to represent the phenomena more accurately [15]. For instance, more variables were added to represent the phenomena more faithfully (e.g., the number of layers to represent the soils in a geophysical model). Alongside the gain of higher fidelity, that complexity has made the computational cost of running the models higher. Running a single simulation can take a long time (e.g., from several hours to several days or weeks) [3] and requires significant resources. Such time frames are highly problematic in the context of supporting decision making [9]. Indeed, a simulation model taking too much time to generate a single projection prevents the interactivity needed by the stakeholders, as well as the exploration of scenarios. With a single projection taking months to be generated, it is impossible to produce the projections of the numerous climate scenarios and the multiple land use scenarios in a decent amount of time for decision making.

Furthermore, simulation models in environmental science are usually elaborated to model specific and local areas. Their elaboration is the results of several years of research and work in a specific location. Data are collected in the field to help build the model and to calibrate them. In hydrogeology, data collection is a difficult task because the number of tools to access underground data is limited and the cost of the process is high compared to the number of data needed to be collected to represent wide areas. Thus, there is also a lack of spatial data to calibrate models. All this limits the generalisation of the projections to larger scales and prevents providing a global vision to stakeholders for decision making.

In all, the resulting long execution time of the simulation software makes it an obstacle to fulfilling the requirements associated to the context of supporting decision making in environmental science [10], [11]. Simulation models are complex and long to execute, as

well as generally specific to a precise situation (*e.g.,* climate scenario and location) [12]. As such, the models cannot be used as they are to provide the support needed by decision making.

As those simulation models are the result of a lot of research effort (time, cost and knowledge), one wants to capitalise on them. One way to tackle the issue is to tailor the simulation models to make them fulfil the requirements of supporting decision making. There is a shift in the usage context for the simulation models and so, new constraints are to be met (Figure 1.1). The goal is to tailor the properties of the simulation models that were primarily used in the context of research for them to be used in the new context of decision making. The models, that are long to execute, accurate with a high complexity and specific to a unique scenario, are to become fast to execute, to remain reliable and to be flexible regarding the exploration of scenarios. They should no longer only describe the physical world, but also aim to act on it through the exploration of projections and decision making.

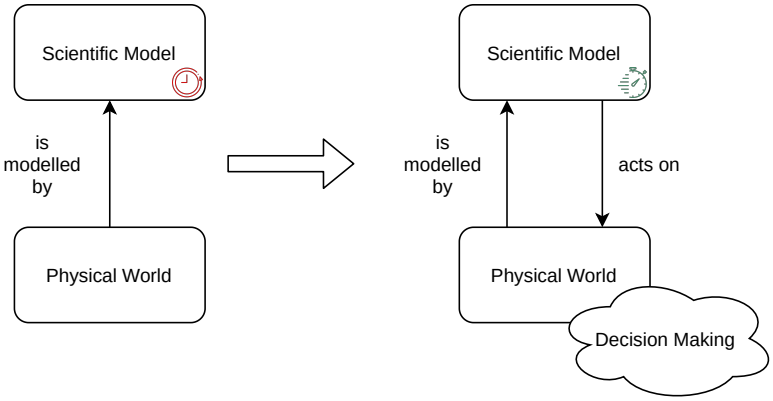


Figure 1.1 – Tailoring scientific models for the new context of supporting decision making.

To do so, the computation of the simulation models needs to be faster. With faster execution, the projections can be produced faster, which then makes the exploration of numerous scenarios possible. It also brings the property of interactivity for the exploration of scenarios with a decrease in the time to get projections. Because the stakeholders do not need every possible details in every projection, we can act on accuracy to increase the execution speed of the simulation models. What stakeholders need is to have the global assessment of risks and impacts thanks to the various projections over the multiple climate scenarios. So, the constraints on high accuracy can be alleviated, while ensuring reliability, in the context of decision making. For instance, stakeholders do not see the use in having

the precise movements of the underground water given in millimetres for every day in the next 50 years. The assessment of areas where groundwater is likely to surface and to cause flooding is the relevant level of information for them. Trading-off some accuracy for a better speed of execution enables to tailor simulation models to support decision making.

In this thesis, we aim to **tailor the simulation models to support decision making** and to generalise its application to the whole context of environmental science and climate change research. The goal is not to tailor or perform a trade-off of accuracy and execution speed for a unique and specific simulation model, but to elaborate a **systematic process** that can be applied on different simulation models of the domain. Hence, the objective of the thesis is the generalisation of the approach for simulation models in environmental science.

To achieve this, we are working on several challenges. First, before performing any trade-off or any tailoring on the simulation models, we need to define the general features across all scientific models to know how to use them properly and to ensure their validity. We need to determine if the simulation models generate reliable projections and when they generate those reliable projections (*i.e.,* determine what the ranges of inputs of the simulation model leading to reliable outputs are). The definition of the validity envelop of the simulation models then enables the tailoring to take place thanks to the trade-off between fidelity and flexibility (*i.e.,* speed of execution). The issue is to define how the trade-off can be performed to match the requirement of a fast execution of the simulation models while maintaining their reliability. The approach also needs to be systematic to enable not only its application on a unique and specific model, but its generalisation on different simulation models in environmental science. Finally, the last step is to see how the exploration of new scenarios is possible given the tailored simulation models. Thus, we describe how the global approach made of three main steps can fulfil the main goal of tailoring the simulation models to support decision making in the context of environmental science and climate change studies.

The challenges can be summarised as:

Challenge 1: What are the distinctive features of scientific models, and how are those models integrated into the scientific software?

Challenge 2: How to systematically perform the trade-off between accuracy and execution speed while ensuring the reliability of the simulation models?

Challenge 3: How to apply this systematic approach of trade-off to enable an opti-

mised exploration of new scenarios?

1.4 Thesis Contributions

To address these challenges, we proceed in several steps that are organised according to the following main contributions of this thesis. The first two contributions result from collaborative work and are the foundation for understanding and building the two core contributions of the thesis.

1.4.1 Demystifying the Scientific Models and their Complementarity to Engineering Models

First, we take a look at the different types of models that can be encompassed in software: scientific models, engineering models, and machine learning models (*i.e.,* empirical models). We inspect the distinctive and common features of those models in the context of cyber-physical systems. We observe that despite their particularities, scientific models have much in common with engineering models. Scientific models, historically used in a descriptive fashion, are now also used to be predictive or even prescriptive [13] (*i.e.,* producing guidelines to be enacted in the real world). Engineering models that have more of a prescriptive nature are integrating more knowledge to describe the real world. Those observations prompt us to consider the models not as types, but rather according to the role they play. They also encourage by extension the use of software- (and engineering model-) oriented methods on scientific models. In essence, we present a framework that provides a vision for how to explicitly integrate the three roles played by models – prescriptive, predictive, and descriptive – as well as their respective data sources and that highlights related actions to integrate them. We emphasise and illustrate the complementarity and duality of models and data in socio-technical systems. This collaborative work results from the Bellairs workshop [14] on the topic of Models and Data and has been published in the journal of IEEE Software (2020).

Benoit Combemale, Jörg Kienzle, Gunter Mussbacher, Hyacinth Ali, Daniel Amyot, Mojtaba Bagherzadeh, Edouard Batot, Nelly Bencomo, Benjamin Benni, Jean-Michel Bruel, Jordi Cabot, Betty H.C. Cheng, Philippe Collet, Gregor Engels, Robert Heinrich, Jean-Marc Jézéquel, Anne Koziolek, Sébastien Mosser, Ralf Reussner, Houari Sahraoui, Rijul Saini, **June Sallou**, Serge Stinckwich, Eugene Syriani, Manuel Wimmer. **A Hitchhiker's Guide to Model-Driven Engineering for Data-Centric Systems**. *IEEE Software*, vol. 38, 4, pp. 71–84, 2020.

1.4.2 Ensuring the Fidelity of Scientific Software

Once we have a holistic view of the functioning of scientific models, we want to ensure their fidelity with regard to the system they represent. We investigate the validation of scientific software and the roles of various stakeholders taking part in the development of scientific software. We highlight the responsibilities of the different roles regarding the artifacts involved in the development of the scientific software. We note the impact of the programming language used to elaborate the software on the validation process. We then present a reasoned approach for the development of scientific software that enables to systematically characterise the validity enveloped by such software, to make it explicit and to lead to a better use of those software. We give guidelines to language designers regarding helping the users perform the validation of their scientific models and associated software, and to language users (*i.e.,* modellers) to ensure the validation of the scientific software. The work has been published in the journal of IEEE Computer in the December issue (2021) [19].

Dorian Leroy, **June Sallou**, Johann Bourcier, Benoit Combemale. **When Scientific Software Meets Software Engineering**. *IEEE Computer*, vol. 54, 12, pp. 60–71, 2021.

1.4.3 Trading-off Accuracy for Flexibility with Scientific Software

Understanding the particularities of scientific software and the related validation process, we then apply approximate computing (a software-oriented technique mainly used

on engineering models) and adapt it to perform a trade-off between accuracy and execution speed on scientific software. We present the systematic approach that reduces the execution time of simulation models while maintaining acceptable results. It consists in automatically reducing the main loop of a simulation model by aggregating the corresponding spatial or temporal data. We show that we can obtain a median speed-up of 95% when applying the technique on a geophysical model of a hydraulic simulation. The work has been published in the Proceeding of the International Conference On Computational Science 2020 [20].

> **June Sallou**, Alexandre Gauvain, Johann Bourcier, Benoit Combemale, Jean-Raynald de Dreuzy. **Loop Aggregation for Approximate Scientific Computing**. *International Conference on Computational Science*, Jun 2020, Amsterdam, Netherlands. pp.141-155.

1.4.4 Optimising the Trade-off for Better Exploration of New Scenarios

The trade-off approach elaborated enables the exploration of new scenarios by systematically approximating the simulation models with a low factor. However, the best trade-off that is possible is not known beforehand (*i.e.,*the highest level of acceptable approximation) and the exploration is thus not optimal (with regard to the execution time). We then propose an optimisation approach to perform an optimal trade-off with regard to the simulation validation metric and execution time, in the case of new scenarios to explore. We also investigate what factors in the approach lead to better trade-offs. We experiment with the geophysical model previously used to validate the trade-off approach, and we show that optimal trade-offs can be obtained in certain cases thanks to the elaboration of a predictive model. We also show that the correctness and the validation of that predictive model is dependent on various factors related to its elaboration.

1.5 Context of this Thesis

The specificity of this thesis resides in the inter-disciplinary approach on the issue at hand. It deals with Software Engineering and Environmental Science (more specifically, Hydrogeology). Bringing those two fields together answers multiple interests and demands

related to each one of them, and, as a result, their collaboration brings a lot of potential breakthroughs.

1.5.1 Research context

Researchers in environmental science (ES) seeking the expertise of researchers in software engineering is relatively new. The first reaction was to reach out to numerical analysts to improve the mathematical aspect of their simulation models (*e.g.,* increasing the fidelity and complexity of their mathematical models). There is now an increasing interest in taking a look at the execution aspect of their scientific models (*e.g.,* HPC, virtual labs, decision-making support). We are faced with new challenges in the modelling process and the context of use of the models. The intention is to adopt a systemic approach of modelling at a larger scale. The goal is to integrate several models together as well as larger volumes of data to represent the complexity and interactions between the different physical phenomena or the different spatial compartments that exist in the real world [15], [21]. For instance, the objective is to make the surface water flow model interact with the groundwater flow model to better simulate the overall water flow behaviour in a catchment area. The integration of models from other fields such as economics enables to represent the human-environment interactions taking place in the real world with higher fidelity [16]. As the scientific models are more and more used to provide a support for decision making in a societal context (*e.g.,* environmental risk assessment, resource management, policy making), a goal is to take the role of the simulation models and their projections into account, and more specifically how they impact the real world through the decisions made and enacted by policymakers [13], [16]. In all, the goal is to adopt an approach involving multiple systems and the related challenges are to manage and to integrate [21]:

— several models representing several systems together

— large volume of data into the simulation models

— knowledge from other fields thanks to models

— the interactions between the models and the real world

This leads to requesting the expertise and skills already mastered in the software engineering field that can answer those challenges [22]. Researchers in software engineering (SE) are already accustomed to work with software that are integrated with their execution environment and the real world. This is the case when dealing with cyber-physical

systems (CPS) which integrate physical and software components together and for which the software receives information from the physical world through sensors and act on it through actuators. Also, we have gathered knowledge and elaborated tools to make such systems adapt according to the context of execution through a feedback loop [23]. We work on software that can be complex and encompass several engineering models and large datasets. Thus, a variety of methods and tools have been developed to address the challenges that are those now faced in ES regarding a multi-systemic execution. However, the domain of SE has its own challenges. Cyber-physical systems are only interacting with the physical world thanks to data and through the sensors and actuators. For a better adaptation of the CPS, a more complex representation of the physical world and the underlying phenomena captured in the data is needed. For instance, a smart grid system can be more efficient by integrating a social model of the behaviour of consumers regarding electricity consumption. The system can thus predict peak electricity demand and anticipate it by adapting the system to meet the coming demands. In software engineering, the aim is to make cyber-physical systems smarter with regard to their interactions with the real world. Engineering models developed in software engineering mainly involve specifying tasks to be performed and are meant to drive the development of system-to-be. There is a lack in representing the behaviour of phenomena from the real world system. This expertise of representing the world in models is part of the knowledge of ES researchers.

The collaboration of ES and SE researchers enables to mutually share the skills among ourselves and therefore, it truly benefits both fields in tackling the different challenges we face (*e.g.,*modelling real-world phenomena, dealing with multiple complex systems). It allows each field to focus on its own expertise area of knowledge by delegating the methods and techniques to come with to the other field.

In this thesis, we deal with a cyber-physical system involving the adaptation of the land use regarding climate change through decision making by stakeholders with the help of simulation models. The simulators have the role of describing the behaviour of climate phenomena, and predicting the state of the land use under such phenomena, thus providing the information to stakeholders to enact solutions in the physical world accordingly. The decision making is then the adaptation process led by the results generated by the software.

Being part of both research fields, we take the point of view of both research fields that thus enables us to fully comprehend the different issues and challenges at hand when exploring and answering the question of tailoring simulation models to support decision making related to climate change.

1.5.2 Social context

The Covid-19 crisis has put the use of simulation models at the forefront of the scene [24]. Indeed, the pandemic has required governments to act in order to curb it and to slow the spread of the virus. The simulation models were mainly used to make projections about the numbers of infected people and to decide on the establishment and the duration of national lockdown. It made the general public realise the existence, the impact and the crucial role of scientific models (*i.e.,* simulation models) in the decision-making process.

A wide range of stakeholders is involved in the use of the models with different degrees of influence:

— The scientists elaborate the scientific models and simulation software.

— The decision makers (*i.e.,* policymakers) use the projections provided by the simulation models to decide on actions to be taken regarding the land use and tackling future climate events.

— The general public is affected by the application of the decisions.

The scientific models and the projections have a central role in the process of tackling issues. They help understand the phenomena and variables involved in the issue thanks to the descriptive nature of the models. They also enable the policymakers to make a better informed and sustainable decision. The general public can access the projections on which the decision makers based their deliberations, and, they can better understand the policy decisions. It also make them more aware of the issues and the magnitude of their consequences, as well as the relative urgency or significance to tackle them. Understanding the issues makes it easier to respect and comply with the decisions.

Scientific models constitute a powerful tool to provide and share knowledge. They foster discussions about issues and solutions among stakeholders by enabling them to share a common ground and vocabulary, thus, making progress on the issue at hand. They are a means for mediation, as they allow different perceptions of the problem to be reflected and give another dimension of the interaction between human and environment. Their use also helps the general public to feel more involved in the decision-making process and can even make them want to voluntarily participate in tackling the issue. For instance, thanks to the use of smartphones, there are projects for which citizens can take pictures of species and help to identify the biodiversity of species in areas that are normally inaccessible to researchers.

To sum up, the use of such models can have a profound impact on society. Therefore, special care and attention should be put forth regarding their elaboration and validation in order to meet the peculiar demands of the usage context.

1.5.3 Implementation of the PhD

This thesis results from the Doctorate that took place in the University of Rennes 1, as one of the very first collaborations between the IRISA laboratory and the Geoscience department of the OSUR laboratory. I was part of the DiverSE team for the SE field and the DIMENV@risce team for the environmental modelling. I was assigned to the DiverSE team and the work in the thesis is dominantly adopting a software engineering approach to resolve the issue at hand. However, I took part in the weekly meetings and seminars in both teams. I had a lot of discussions with other members of the team working on environmental modelling to gather the challenges they face, and more particularly with the PhD student responsible for the development of the model for risk assessment of flood that is used as a motivating example for the implementation of our main contributions. The work really aims to greatly reflect the various and many interesting discussions between the members of both fields involved in this collaboration (supervisors and associated PhD students). A lot of time was dedicated to understand the practices and point of view of each field to effectively tackle the issue at hand.

What's more, the thesis uses the context of the RIVAGES 2100 project [25] as a motivation to understand the challenges faced by the different stakeholders involved by the use of scientific models to support decision making for environmental issues. The project aims to provide models and tools to help assess the environmental risk of flooding and salinisation of groundwater in the coastal region of Normandy in France, in the context of policy and decision making. I participated in several meetings with different stakeholders (*e.g.,* prefects, government representatives, representatives of the regional agency dedicated to environment, land-use and housing, representatives of the regional water supply agency) to exchange with them and collect their feedbacks and expectations.

1.6 Outline

The rest of this document has been organised as follows. Chapter 2 introduces the background of scientific modelling with the numerical analysis of scientific models and

their validation. Chapter 3 gives an overview of the state of the art of trading-off accuracy for better execution speed of software. Chapter 4 presents our contributions on ensuring the credibility of the results produced by the scientific software. We describe the complementarity of models and data, and we investigate the responsibility of the modellers with regard to the validation of the scientific software. Chapter 5 presents our main contribution which consists in a systematic approach to automatically reduce the time execution of complex scientific simulation models while maintaining acceptable results. Chapter 6 presents the automatic and a priori optimisation of the systematic approach to explore numerous scenarios. Chapter 7 concludes the thesis by describing the advances it brings in the fields of modelling in hydrology, science computing and software engineering. We discuss the perspectives it offers within the associated inter-disciplinary research scope and the global topic of sustainability.

Chapter 2

Background on Scientific Modelling

In this chapter, we introduce the concepts and vocabulary on which our contributions rely. We take a look at scientific software in Section 2.1 and their development in Section 2.2. In Section 2.3, we then describe the general validation process of scientific models.

2.1 Goal of Scientific Modelling

2.1.1 Scientific Models

Scientific modelling is an important part of the scientific approach and reasoning [3], [26]. Indeed, elaborating a model helps summarise the knowledge and the theories about the system under study, and helps formulate and test hypotheses about the system. This is part of the process of acquiring and creating more knowledge, to help humans better understand the world they live in.

More specifically, scientific modelling is the activity which consists in representing a physical phenomenon of the world. A scientific model is thus a representation of some aspects of that phenomenon [26]. It is used to study, understand and explain the phenomenon (*e.g.,* define, quantify, visualise, or simulate), based on established scientific knowledge defining a theory. It can also produce projections to anticipate the evolution of the physical phenomena in the future according to various scenarios [15]. A holistic view of a phenomenon or a system is assumed and different models can be used at different time- or space-scales. Scientific models encompass a wide range of representations, such as climate change models, electromagnetic models, protein synthesis models, or metabolic network models [27]–[30].

Different types of models are used for different aims: conceptual models to improve shared understanding, operational models to refine measurement, graphical models to

visualise the phenomenon, or mathematical models to quantify the phenomenon. More particularly, those mathematical models can be numerical (e.g., systems of differential equations), non-numerical (e.g., agent-based models) or based on analytics (e.g., machine learning models), and capture the behaviour of the modelled system. Numerical models can be further refined as continuous or discrete. Numerical models may be resolved analytically, meaning it can theoretically be done by hand with a piece of paper and a pen, or with the help of computers when it is not possible or more complicated to do so analytically. When computers are used for scientific modelling, simulations are carried out to make experimentation with the intent of gaining insight into the system's behaviour [3], [31] and usually involves a dynamic process. A simulation corresponds to the execution of the computer programs embedding the scientific models, the so-called *simulation codes*. Simulation models are thus referring to scientific models embedded in the scientific software, and by extension, they stand for the software.

In hydrogeology and climate change, the scientific models are usually numerical mathematical models and they are often based on partial differential equations [32]. The simulations, and so the projections, are thus based on those mathematical models embedded in the software. So, in this thesis, we restrain our focus on scientific models which are numerical mathematical models and on simulations models that correspond to scientific software that encompass those mathematical models.

In all, the scientific model can be represented as depicted in Figure 2.1. The mathematical model is composed of equations that can be summarised as a global function $Y = f(X, \theta)$. The function is a combination of variables X (*i.e.,* elements whose values change during the computation) and parameters noted θ (*i.e.,* elements whose values are fixed during the computation). The model thus enables the production of outputs, Y, from a set of inputs and parameters.

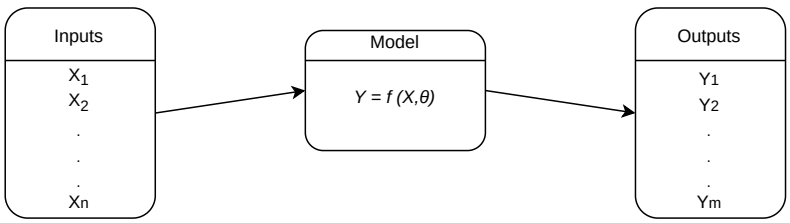


Figure 2.1 – Representation of a scientific model.

2.1.2 Numerical analysis and the quest for greater accuracy

Scientific models, *a.k.a.* numerical mathematical models, have first been used by scientists to help study and understand the world and the occurring physical phenomena. By representing those phenomena and analysing them, further knowledge has been created. As time went on, models integrated this new knowledge and became more complex up to a certain point. Formulating too complex models was considered a useless exercise as it was then impossible to investigate their behaviour analytically [3].

The emergence of computers then helped to overcome this limitation by performing the numerical analysis of the models. Scientific models were embedded into scientific software to be computed despite their apparent complexity. Too complex mathematical models to be analysed analytically could then be analysed computationally. Since then, further refinement of the simulation models has been a constant process. Models become more accurate regarding the system represented as they capture more details of the interactions. They are described at larger scale while the resolution remained fine-grained [9]. The number of processes representing the phenomenon is higher and thus the number of variables and parameters is also higher. Several aspects of the modelled phenomenon are captured and integrated into several sub-models meant to be computed together (*e.g.,* a model representing the physical aspect of the water dynamics of rainfall on the soil surface, another model of the physical behaviour capturing the water flows underground, and a model describing the economical impact of the flooding caused by the rise of the water table to the surface). As models with large scale or with complex equations require considerable computational power, those refinements are only possible by the growth of computational power of the resources enabling the software to be executed [33], [34].

To match this pursuit, the optimisation of all parts related to the computation of the simulation models was sought. Hardware resources evolve to provide higher performance and computational power. Since around 1970, the capacity of microprocessors, *a.k.a.* computer processors, which are the computer's central units that interpret and execute program instructions, has globally followed the Moore's law of a two-fold improvement every two years [35]. But since around 2005-2010, those observations are not satisfied anymore [36]. Parallelism was then introduced alongside to multi-core processors and multi-threads. The operations to be executed are divided to be performed at the same time. Parallel computing was later used with hardware architecture such as clusters and grids that pool the resources. Parallel computing is also an alternative to overcome the limitations of frequency scaling of microprocessors. More recently, the use of Graphical

Processing Units (GPU) has emerged to support parallel computing and to gain even more performance (up to tens and even hundred times according to [34]).

Software was also developed to benefit from those hardware capabilities. Specific languages and solvers were elaborated to embed the mathematical models into software and enable the computation of the scientific software. For instance, Mathematica [1] is a popular tool as it is seen as one of the first major systems to enable scientific computing [37]. It provides facilities for a wide range of applications thanks to numerous built-in libraries such as algorithm implementation, symbolic computation, statistics and machine learning and is written in the Wolfram language. We can also cite Matlab, [2] TK solver [3] and Modelica, [4] all of which are quite popular among the scientific modelling community [38]–[44]. Some standardisation was made for message passing (MPI) and for shared memory parallelism (OpenMP) and constituted a great step forward for parallel systems [36].

In all, the pursuit of greater accuracy of the models is tightly connected and driven by the improvement of the computational capabilities. The field of High Performance Computing (HPC) aims to make efficient use of the high performance of the computational power provided by supercomputers' architecture (*e.g.,* big servers, clusters and grids) and tools (*e.g.,* parallelism) to enable the execution of resource-demanding software. For scientific modelling, only HPC enables the computing and the elaboration of the highest accurate and complex scientific models [45] such as nuclear reactor power systems models [46]. It brings significant progress to mathematical modelling of real-world problems, new ideas for understanding and improving the mathematical methods and new insight into the nature of the studied systems [3], [47]. For example, Vital *et al.* [48] present a high-performance computing tool for climate change impact studies on grasslands ecosystems. The complex simulation model covers the metropolitan area with a resolution of 8x8km for weather scenarios and soil data split into three soil layers. The data processed are numerous (hourly data of water vapour pressure, air temperature, global solar radiation, rainfall, and wind speed) with weather series from 1970 to 2100. The simulations were executed on a cluster machine with 200 processors for about 25 hours, instead of around 5000 hours (*i.e.,* around 208 days) on a single processor. In the work of Mizielinski *et al.* [8], the HPC was also used to perform atmosphere-only global climate simulations over the period 1985–2011, at different resolutions (25 km, 60 km and 130 km), represent-

1. https://www.wolfram.com/mathematica
2. https://matlab.mathworks.com/
3. https://www.uts.com/Products/Tksolver
4. https://modelica.org/modelicalanguage.html

ing 144 million core hours of execution. As the observation is recounted in [49], scientists often view the improvement of computational performance not as saving computing time but as an opportunity to get a higher-fidelity approximation of the problem being solved, meaning a more accurate simulation model. As such, the pursuit of greater accuracy for scientific modelling is unlikely to cease and is strongly driven by the growth of HPC techniques.

2.2 Development and Refinement of Scientific Software

The goal of scientific models is to represent existing real world phenomena and simulate their behaviour with the help of software computation. The process of modelling continuous events through discrete computation involves the elaboration and the refinement of several artifacts leading to the development of the scientific software. In this section, we take a look at the development process of such software and the associated refinement that is applied to ensure that the specifications of the modelling objectives are respected.

2.2.1 Overview of the Development Process

The development of scientific software involves a specific process and concerns [50] and relies on the successive elaboration of various artifacts which are represented in Figure 2.2.

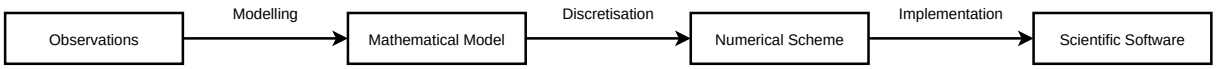


Figure 2.2 – The elaboration steps of scientific software

From a set of observations of a phenomenon and theories, scientists develop a mathematical model, *a.k.a.* continuous numerical model, often taking the form of systems of differential equations. Since these systems of equations cannot generally be solved analytically, they must be solved numerically, which requires a discretisation process. This step requires the application of a discretisation method (*e.g.,* finite element method, finite volume method) to obtain a numerical scheme. This numerical scheme is a discretised mathematical model, which specifies the sequence of computations to execute, given a

discretisation of the domain according to discretisation parameters (*e.g.,* space, time), an initial state (*e.g.,* initial temperature in every point of the domain) and inputs from the environment (*e.g.,* rainfall), to compute a numerical (as opposed to analytical) solution to the model. From that point on, software engineering concerns enter the development process with the implementation phase. It encompasses all concerns to obtain a reliable working software, including performance, concurrency, targeted architectures, memory management, data access, and so on. The software is then implemented with a programming language while fulfilling the aforementioned software engineering concerns.

The overall development is carried out in accordance to the modelling objective of the scientific domain (*e.g.,* simulation of some physical phenomena). The different artifacts are elaborated with regard to that objective and the context of use. The scientific model is thus tuned thanks to calibration, and refined through sensitivity analysis, to ensure that the execution of the scientific model matches the specific behaviour of the physical phenomena to simulate.

2.2.2 Calibration

When the model objectives are formulated and the structure of the model is set up, the model has to be calibrated to ensure that it matches the context of use. The mathematical model is composed of differential equations involving variables and parameters (cf. Section 2.1 & Figure 2.1) and represent the general behaviour of the physical processes. In order to make the model represent a specific situation, the values of the parameters of the model (θ) need to be fixed. Furthermore, without giving the values of the parameters, the simulations cannot be performed. For instance, a hydrogeological model of the underground water flow uses the global equation from Darcy's law (cf. Equation 2.1) as the foundation of the mathematical model. The equation involves some variable q (the flux of the fluid or flow rate) and some parameters such as the permeability of the soil (k), features of the soil layers (*e.g.,* L for the length and A for the area) and the viscosity of the fluid (μ). To match the model to a specific catchment in the real world, those parameters have to be specified.

$$q = \frac{kA\Delta\ P}{\mu L} \tag{2.1}$$

The process of fixing the values of the parameters of the model is called the calibration. It aims to find the best or more suitable parameters values [51] and to tune the model to

the context. The values of the parameters are fixed thanks to sample field data taken on the site to be modelled or in the literature that give a range of possible data thanks to surveying the various sample data available. Some parameters can be directly measured on the field but it is not often the case in environmental science and hydrogeology as the modelled space is generally large and data are difficult to retrieve (*e.g.,* underground data) [52]. Then the process of calibration is to find the value of the parameters that make the simulation results best match the observations from the field (*i.e.,* the data retrieved from the field). For instance, with the hydrogeological model, the calibration involves finding the best value for the permeability parameter so that the simulation gives a depth of the underground water that matches the one of the studied catchment thanks to historical data of the water flux. The goal is to compare the variable value of the simulation result to the value of the variable in the real world. The closer the values are, the better it is.

The common metrics used to compute the difference between the simulation results and the field data are the ones commonly used in statistics such as the mean squared error (MSE), the root mean squared error (RMSE) and the Nash–Sutcliffe efficiency (NSE) metrics [15], [53], [54].

As a lot of uncertainty is involved (*e.g.,* epistemic uncertainty and sampling uncertainty) [15], an acceptable margin between the simulation results and the field data is allowed. The value of the margin is defined by the modellers and the domain experts. The calibration is considered done when the discrepancy between simulation results and data is below within the acceptable margin.

The calibration can be performed on the mathematical model, when analytically resolvable, or the numerical scheme, when requiring few resources, as the goal is to determine the value of the parameters of the equations. However, as the equations tend to be complex and require some computation power, the calibration is often done through the execution of the simulation model.

2.2.3 Sensitivity Analysis

When elaborating the model, the goal is to reproduce the behaviour of the observed system. The mathematical model aims to encompass that behaviour inside the equations and combinations of variables and parameters. To decide on which variables, parameters or combinations to keep or modify, modellers want to estimate the importance of the different factors (*i.e.,* variables and parameters) on the model outputs. They want to know

to what extent the variation of the values of factors impacts the values of the outputs and to determine if such properties are in the scope of their model (*e.g.,* monotony, regularity) [55]. The influence of the key factors is investigated by the sensitivity analysis. Thanks to the analysis, the modellers have information about which factor they may remove from the mathematical model or which factor they would want to specify more and which should be subdivided into several. In essence, it is an exploration of the model to assess its properties. It varies the values of the factors to measure how much this changes the output result [55]. First, the ranges of values for the variables and parameters is defined. The values to explore in the defined ranges are generated according to a chosen technique (*e.g.,* random selection, with a fixed step between values). The outputs are then computed for the various values of variables and parameters. Finally, the sensitivity is calculated for each variable and parameter thanks to a sensitivity metrics (*e.g.,* Sobol method, variance analysis) [55]. According to the value of the sensitivity metrics, the influence of the variables and parameters is estimated and modellers can use it to guide the tuning of their model.

The sensitivity analysis along the calibration can be performed directly on the mathematical model, but is usually done thanks to the simulations of the scientific software. They both help to explore the model and to tune it to match the modelling objectives (*e.g.,* specificity of the modelled scenario and system).

2.3 Validation of Scientific Software

Scientists have long been elaborating scientific software to model world phenomena and to study them. Their main goal is to obtain scientific models that can be trusted to produce exploitable results according to the modelling context of their use. Therefore, they concern themselves with the credibility and the validity of those models in order to have useful and reliable results produced by the models and to prevent drawing wrong conclusions from them [56]. They aim to reduce the uncertainty of the models under an acceptable threshold they have established themselves or that is imposed on them by the modelling conditions (*e.g.,* the uncertainty of the observations and data used to elaborate the model, or the epistemic uncertainty related to the limited knowledge about the system under study). In all, they want the models to rightfully represent the system under study to make them trustworthy. It should be noted that the meaning of *rightfully representing the phenomena* depends on the modelling context. For instance, in environmental science,

and more specifically in hydrogeology, it is difficult and challenging to have very accurate models of the physical phenomena as a lot of uncertainty is involved due to the domain of study [57]. Indeed, scientists have to face the limited available knowledge in the domain. The system they study usually involves the underground water network. There is still great difficulty in having access to information about the underground system as there are limited means to collect them (*e.g.,* it is not possible to dig to unlimited depth). Linked to that, scientists face a lack or penury of data that they can use to build their models and come up with new theories and hypothesis. The collection of data costs a lot of time and money as it requires huge resources and complex processes (*e.g.,* permits from the city to build wells) [11].

The representation has to be more or less accurate according to the constraints that are imposed by the context in which the model is to be used. A unique and perfect scientific model to represent the physical phenomenon does not exist but there is a solution space that describes the range of possible models of the system. Therefore, the aim is to define when the model is valid or not, and more specifically, when it is valid or not according to the context of use.

2.3.1 Scientific Validation

The question is then how to know if the results produced by the model are valid or not. Scientists, who are here considered as the modellers, use an approach of validating their models to ensure the credibility of the phenomena represented in the model. The principle of the scientific validation is to check the credibility of the model with regard to the real world. One way is to ensure that the model generates results that match the observations made in the real world. However, the validation of scientific models has some limitations and some researchers even present the validation of scientific models as an utopian task or an impossible task [57], [58]. Indeed, models are by definition an approximation of the modelled system and in the case of scientific models, of the real world. As some aspects of the system are not taken into account and are ignored, there is a certain discrepancy between the results and the real world phenomena. The real world phenomena are not closed systems. Moreover, uncertainty is present at many different levels related to the modelling (*e.g.,* data sampling for calibration, knowledge about the system, discretisation of the mathematical model). For instance, the model is specified at a chosen scale, however the data collected to calibrate the parameters may not be measured at the same scale, or it may even be impossible to collect them [17]. The observations or the data collected in

the real world are also subject to uncertainty as their accuracy depends on the context of the collection (*e.g.,* tools, weather, time). That is why Oreskes*et al.* [58] recommend to analyse before using the results produced by the scientific models of natural systems, and more specifically, when they are used for predictions in a context of decision making.

As models are useful and essential tools, they still need to be validated in some way to be used. The idea is to take into account the modelling objectives and the modelling design in the validation process. The models are "validated" regarding a context of use (*e.g.,* decision making) and with an estimated level of accuracy that encompasses the uncertainty associated with the design of the model and corresponding data (*e.g.,* input values for the simulations). Thus, the validation is fundamentally based on the definition of an acceptability criterion. To ensure the fidelity and credibility of the model, this acceptability criterion should reflect the requirements of the context of use. It ought to take into account if the context does not require for very accurate outputs (*e.g.,* global vision of phenomena) or if it does (*e.g.,* specific and critical scenario). Validation amounts to ensuring that the uncertainty is within an acceptable range. Scientists have therefore found methods to try to estimate the uncertainty and limit it in the model. For instance, they usually use the processes of calibration and sensitivity analysis as support for the validation. Therefore, the acceptability criteria used for validation are commonly the same as the ones used for calibration [54]. The data collected are separated into two datasets. One dataset is used for the calibration of the model. The second one is used to check that the calibrated model can produce results comparable to the observations. Historical data are often employed to check if the model is able to reproduce the current state of the system and its past behaviour. Calibration and sensitivity analysis can help determine the level of uncertainty thanks to how they operate. The validation of the model is acknowledged when the level of uncertainty is below an acceptable threshold.

2.3.2 Software Validation

The validation of software is based on testing that the software behave as expected and described by the requirements that are expressed by stakeholders (*i.e.,* people requesting the creation of the software). The V-Model is a standard representation of the development of software, demonstrating the relationships between each phase of the development and the corresponding testing phase [59]. It is presented in Figure 2.3.

On the left top-down branch, the stakeholder requirements are refined and extended with more specific software concerns corresponding to the different levels of organisation

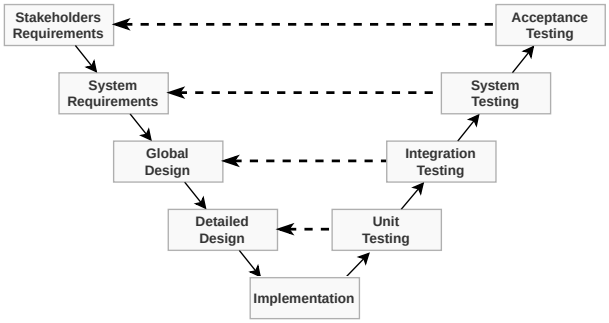


Figure 2.3 – The SE V-Model [59]

of the software. The elaboration of the different specifications follows a standard process. First, the *software requirements* are formulated from the global stakeholder requirements and they specify the expectations for the software as a whole. They contain information about the general software organization and the expected behaviours of the software in various scenarios of interest (*e.g.,* use case diagrams). Then the *general design* is drawn to more specifically describe the different components that shape the software (*e.g.,* component diagrams). This leads to the formulation of the *detailed design* of the software which aims to document all the structural details of the software (*e.g.,* class and sequence diagrams) in order to then allow the implementation of the software in a clear and efficient way. Using the various requirements contained in the previous specifications, the *implementation* of the software is generated.

The resulting software is then validated according to the stakeholder requirements, through V&V activities under the form of *unit*, *integration*, *system*, and *acceptance testing*. First, *unit testing* ensures that the implemented software respects the guidelines of the detailed design, meaning that each unit of the software code performs as expected (*e.g.,* mock-based testing). The fulfilment of the general design specifications are checked through *integration testing* as it makes sure that the different components interact correctly (*e.g.,* Big Bang testing, incremental testing). Then *system testing* controls that the implementation meets the software requirements by validating the complete and fully integrated software, considered here as the system of interest, using various testing techniques (*e.g.,* usability testing, load testing, regression testing, functional testing). Finally, compliance with the stakeholders' expectations is validated through *acceptance testing* (*e.g.,* user acceptance testing, business acceptance testing, operational testing).

2.3.3 Scientific Software Validation

To represent the continuous behaviour of some real world system, scientific software has the particularity to encompass a scientific model in the form of a numerical scheme resulting from the discretisation of a mathematical model. The modellers, *i.e.,* here the scientists and domain experts, are used to focus their effort on the validation of the mathematical model, meaning the validation regarding the scientific domain, whereas not as much effort is provided for the validation of the computational aspect of the simulation models. That can be explained by the fact that the domain experts that elaborate their scientific model, and associated simulation software, do not usually have a background in computer science or software engineering [60]. This results in an incomplete assessment of the uncertainty comprised in the simulation model, and thus, can lead to misleading projections. Work has been done to encourage the domain experts and modellers to adopt and to implement the practises regarding the verification and validation (V&V) of software [61]. Since scientific software testing is still an ongoing area of research, those practices are not always applied correctly and systematically [60], [62], [63].

Applying the techniques focusing on software does not ensure a global validation of scientific software. Validation must also take into account the specificity of scientific software that is the transition from a continuous model to a discrete implementation, all to represent a continuous phenomenon. Different studies propose specific approaches and techniques about the validation of scientific models that are encompassed in software [64]–[67]. For instance, convergence testing deals with checking the discretisation of the mathematical model. In this case, when the discretisation step is reduced, the discrepancy error between the simulation results and the observations or theoretical results tends to zero. Also, other techniques target that the physical principles underlying the real world system, such as conservation of energy and mass, are respected by the model. A general approach to scientific model testing is proposed in [64]. Broadly, this approach consists of (*i*) identifying and characterizing mathematically every source of uncertainty (aleatory and epistemic) [68] in both the scientific software and experimental measurements used as a reference, (*ii*) propagating those uncertainties to the outputs through techniques such as sensitivity analysis [69], (*iii*) defining a validation metric, *i.e.,*a mathematical operator comparing the numerical solution and the experimental measurements, and applying it to produce an assessment of the validity of the model over the validation domain, and finally (*iv*) employing statistical techniques such as regression fits of the validity metric over the validation domain, which enables to interpolate or extrapolate the validity metric over the

domain of intended use, thereby providing the uncertainty of the simulation results over the domain of application. In summary, ensuring the reliability of the scientific software is a complex task as it does not only involve the V&V techniques regarding the implementation and software part as well as the mathematical model, but also the numerical scheme. The scientific software has to be validated as a whole complex system that encompass and is related to several artifacts.

Take-away Messages of the Chapter

A model is a representation and is by definition only reflecting some parts or aspects of the modelled system instead of its whole. As such, some choices and approximations are made when designing it. There are several types of models: conceptual models, graphical models, mathematical models, etc. In environmental science, scientific models are mainly mathematical models, and so, we focus the thesis on that specific type. Scientific models are at the centre of the scientific research. Their primary goal is to understand the real world better and create knowledge about it. They can also be used to make predictions about the future behaviour of the system under study. To be used, however, the uncertainty of the model needs to be assessed or estimated. The validation of the model ensures that the amount of uncertainty is acceptable with regard to the modelling context and, thus, it ensures that the model is reliable and usable. As scientific software results from the refinement of several artifacts, the validation step involves specific testing focusing on those artifacts.

Chapter 3

State of the Art

In this chapter, we give an overview of the state of the art regarding the different approaches to speed up the execution of the scientific software. We focus on surrogate modelling and the related three types of approaches. We present the model reduction approach in Section 3.1, the data-driven approach in Section 3.2, and, the approximate computing approach in Section 3.3. We describe the specifics of each approach, and we assess their relevance regarding the challenges addressed in this thesis.

In the context of scientific modelling for research, the scientific models have become notably complex (cf. Section 2.1.2) and need to be specific to a precise scenario in order to reach a high accuracy. It is thus computationally expensive to execute those models. Each simulation requires significant computational resources. When we want to use those models in another context that demands some interactivity or running numerous simulations (*e.g.,* the exploration of what-if scenarios, decision making, and design space exploration), the computational cost of their execution is currently expensive and needs to be lowered. The need to tackle this high computational cost then becomes compelling. As a result, it is the subject of numerous research investigations and a variety of approaches to address the issue.

One type of approach uses to High Performance Computing (HPC) and relies on using more performant resources and complex infrastructures with large computational power such as supercomputers and grids. We presented HPC and its impacts on the evolution of the complexity of scientific models in Section 2.1.2. The current trend is to continue the improvement of the performance capabilities that such execution environments can provide. With better performances, more complex models can be executed and even faster, and so on. HPC can offer huge performance gain, however, it involves the necessary access to a large amount of high performance computational resources. Those conditions do not match with the requirements associated with the problem we tackle in this thesis: the use of the scientific models, which were first elaborated in the context of research, to support

decision making. As we are interested in a systematic and easily realisable approach to implement, we look for solutions without constraints on the execution environment and that can be performed on any classical computer (*e.g.,*personal computer, no access to data center).

The other type of approach involves trading off some model fidelity for more flexibility in the use and execution of the model. As described in Chapter 2, scientific models are by definition an approximate representation of the studied physical phenomenon as only some aspects are taken into account. Futhermore, the context of decision making about climate change and the exploration of what-if scenarios and projections deals with a lot of uncertainty. The trade-off between accuracy and flexibility is therefore suitable in this situation. The trade-off involves the use of an alternative version that is an approximated version of the complex and computationally expansive original model (*a.k.a.* reference model). The surrogate model, *i.e.,* the alternative version, is used to be executed in place of the reference model in a context or with an intent that is different than the one for which the reference model was elaborated. In our case, the surrogate models are the solution to trade off some fidelity for more flexibility of the simulation model, and more specifically for a faster execution speed [9] (cf. Figure 3.1). Indeed, surrogate models with sufficiently short runtimes can be used in interactive decision support environments [9], [70]. It also leads to using the simulation models in different contexts involving various stakeholders and purposes (*e.g.,*, decision making, investigation of the phenomena behaviour), providing more flexibility in the model usage and execution.

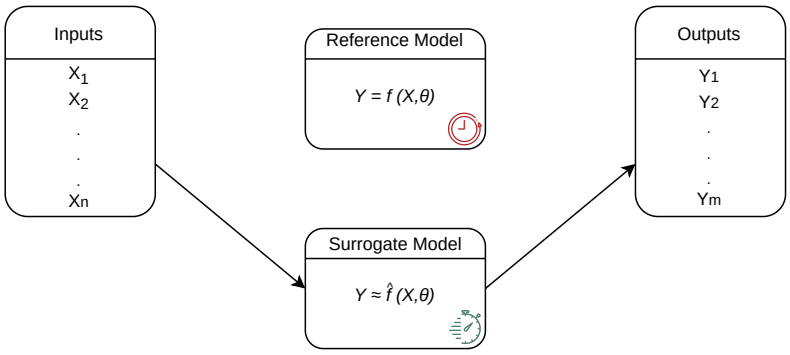


Figure 3.1 – Use of a surrogate model with faster execution in place of the reference model.

Several works propose classifications of surrogate modelling techniques [9], [71]–[74]. For example, Barquero et al. focus on data processing applications and on applying data sampling techniques to trade-off accuracy for performance [75]. Also, Alizadeh et al.

present a framework to classify and select appropriate surrogate modelling methods of engineering problems according to size, accuracy, and computational time characteristics, after reviewing more than 200 research papers [71]. As we do not intend to describe every specific technique that can be used for surrogate modelling but a global overview and we feel that some approaches are not represented, we use a simpler classification of surrogate modelling approaches applied in the literature. We organise the kinds of approach dealing with accuracy trade-off into three categories : (i) the model reduction approach, (ii) the data-driven approach and (iii) the approximate computing approach.

As a side note, surrogate models are also designated with different names such as metamodels [76], response surface models [77] or reduced models [78] which are used interchangeably without much regard to the approach employed. We choose to stick to the term of surrogate models in the remainder of the thesis as it is used in both the fields of computer science and environmental science.

3.1 Model Reduction Approach

The Model (Order) Reduction techniques are historically the first techniques that have been used by scientists and modellers to simplify their models. This approach considers the simulation model as a white box and takes the point of view of domain experts as it focuses on reducing the complexity of the mathematical model, and by implication, on making the execution of the scientific software faster. It thus involves domain knowledge and aims to play on the physical processes encompassed in the reference model. The techniques mainly apply a reduction on the dimension of the design space (*i.e.,* the number of possible configurations, meaning the range of values of the variables, is reduced) or on the number of variables and parameters.

3.1.1 Proper Orthogonal Decomposition

The proper orthogonal decomposition is a common projection-based surrogate method. It can be associated to statistical approaches such as the principal component analysis as the goal is to reduce the dimensionality of the simulation. The design space is reduced to a subspace thanks to the analysis of pattern in the variable behaviours. It aims to determine the variables that can be multiplied by a same value and still have the same general behaviour. Those variables are correlated to the physical properties of the system

and correspond to the subspace used to reduce the dimensionality of the simulation. To find the set of variables, several snapshots of the variables values according to simulation steps are used.

For instance, Hinze and Volkwein [79] apply proper orthogonal decomposition for a fluid dynamic mathematical model. They use 100 snapshots (sets of the values of the variables) for different dimensions for every step of the simulation. They note that the reference simulation takes approximately 17 times more CPU than their approach. Also, McPhee and Yeh [80] show that the method can be used in the context of decision making for groundwater management.

3.1.2 Simplified Physics

The simplified physics method consists of elaborating a new and simpler mathematical model (*i.e.,* lower fidelity model) for the studied system. As such, it does not require anything from the reference model. The simpler model is obtained thanks to the regular physics-based modelling process based on domain knowledge. The modellers elaborate the new and simpler equations encompassing the system behaviour thanks to their knowledge. They restart the modelling process from scratch.

Keating at al. apply this method to create a simpler model of a groundwater model [81] based on the domain knowledge that the underground water level is perturbed by the presence of wells and can be modelled thanks to characteristics regarding the location of those wells.

Some hybrid methods combine the use of a lower-fidelity model with the reference model to perform the trade-off between accuracy and performance. The lower-fidelity model is executed for parts of the simulation that do not require high accuracy whereas the reference model is executed for the parts that do. It enables us to keep details of the complex model, while benefiting from the speed of the simpler model [9]. The method is usually involved with multiple grid resolution simulation. The low-fidelity model deals with a coarse grid whereas the reference one deals with a finer-grained grid.

3.1.3 Relevance for Decision Making

The model reduction approach ensures the interpretability of the surrogate model with regard to the domain of study. Indeed, domain knowledge is conserved to better represent the physical processes of system under study. The reliability of the surrogate

model is more guaranteed. However, it requires a lot of time to elaborate the surrogate simulation model as it needs a new process of calibration and validation. It also takes a lot of expertise to simplify the physical processes with regard to the domain of study as well as to numerical analysis.

3.2 Data-driven Approach

The data-driven approach considers the simulation model as a black box since there is no modification made on the reference model and a whole new surrogate model is elaborated. This approach of surrogate modelling involves an empirical model that captures the input-output mapping of the reference model [9]. The empirical surrogate model has a statistical nature compared to the usual complex physics-based reference model in environmental science. The empirical model is trained, *i.e.,* calibrated, on a set of inputs and outputs of the complex model. Once the accuracy of the training is ensured, it can be used to predict the outputs of the complex model for new inputs with which the complex model has not been executed. In summary, the surrogate model emulates the process-based model simulation results as a function of inputs and parameters, but runs much faster [11].

That approach can yield undeniable benefits in terms of speed-up. Indeed, it is the core of the service offered by the company called Extrality. [1] They provide a service to generate a surrogate model of the simulation model using artificial intelligence (*i.e.,* a data-driven approach) to help users run more simulations faster and to explore them while keeping the accuracy of the physical process based model. The service is presented as a means to reduce the computation costs and to capitalise on all the past simulations run to improve the actual simulation model.

Several techniques enable the elaboration of the empirical model. They can be organised into two types : Machine Learning methods or Deep Learning methods. Both types of methods have made breakthroughs in computer science fields (*e.g.,* speech recognition, computer vision, natural language processing), which has led to their increasing application to scientific domains [11], [72].

1. https://www.extrality.ai/

3.2.1 Machine Learning

Machine learning algorithms can automatically improve their performance with respect to some tasks through experience [82]. There are various techniques that differ in their application according to the context of the surrogate modelling (*e.g.,* classification or regression, supervised or unsupervised methods) and to the nature of the reference model (*e.g.,* types of input and output data).

The use of machine learning for surrogate modelling has been an increasing interest in hydrological science [11]. Surrogate models enable to optimise tasks that are part of the modelling design process such as uncertainty quantification [73], sensitivity analysis [83] and calibration [84], [85]. Zhang at al. apply two techniques (*i.e.,* Gaussian Process Regression and Polynomial Chaos Expansion) to create a surrogate model for performing the calibration of a groundwater transport model [85]. Calibration can usually be a computationally expensive task, so, employing a surrogate model enables to determine suitable calibrated parameters for the reference model. Surrogate models can also operate as a decision support tool. For instance, Cai et al. elaborate a surrogate model of a watershed simulation model that is used in a decision-support framework to assess the measures regarding drought mitigation under several climate projections [86]. More particularly, the role of the hydro-agronomic model is to simulate the hydrological and agronomic response to climate change. The authors apply the support-vector machines technique to generate a fast surrogate model able to answer the need of interactivity in the decision-support process. The statistical surrogate model is created from a 2500 element dataset and produces results with a coefficient of determination superior to 0.9 for all the seven studied climate scenarios. It replaces the high-fidelity reference model to make the framework computationally compliant without loosing essential environmental responses to various climate inputs and land-use measures. Also, in [87], the authors use a machine learning approach and a Random Forest algorithm to create a surrogate model for urban flood prediction that is able to make real-time decision support more feasible. More than 16900 data are analysed to elaborate the surrogate model. A speed-up of a factor 3000 is observed with the surrogate model compared to the physics-based reference model. They conclude that machine learning surrogate models offer significant potential to support real-time decision making in the field of flood management.

3.2.2 Deep Learning

Deep Learning is a subset of Machine Learning **Ongsulee2017Nov**. It is based on Artificial Neural Networks of multiple layers. This multi-layering makes the algorithm automatically generate data labels, *i.e.,* features, from the raw input data given to it. The features better represent the information distribution in the overall dataset than the raw input data. They make the training and the elaboration of the surrogate model more efficient.

Zhang et al. [70] use a Deep Learning technique involving the creation of a surrogate model of a hydrogeological reference model for flood simulation in the context of decision making. They base the training of the model on a dataset with 2400 executions of the reference model as raw inputs. The surrogate model has a mean absolute percentage error of 20% in its predictions but is at least 30 times faster than its reference model. The model can also make flood predictions for different sizes and spatial characteristics while remaining effective. They note that the training of the neural network algorithm is time consuming.

Lu and Ricciuto [88] developped a technique that decreases the need for a reference model. The authors employ artificial neural networks for large scale Earth system models. The algorithm only requires 20 simulation runs, and it establishes the relationship between 8 model parameters and 42660 outputs. But the number of variables used as raw inputs is really high (more than 40000) at the beginning. This large number of inputs is exploited by the deep learning algorithm to find effective features to train the model.

3.2.3 Relevance for Decision Making

The set of data-driven approaches involving statistics can be very efficient in that the generated surrogate model can produce the results instantly [89]. The approaches also enable the generated surrogate model to be used for different scenarios, provided that the criteria are well defined and represent the information contained in the input data. However, the approach requires a relative high number of executions of the reference model for the calibration (*i.e.,* training phase) [72]. The number of executions has to be large to capture enough data for the statistical algorithm to extract the mapping relationship between inputs and outputs and to create a reliable surrogate model. It is due to the high complexity of the physics-based reference model. As there is a lot of variables and the equations are complex due to the several phenomena represented. The complexity of a

reference model makes the execution time long and makes the simulation model impossible to use as it is to support decision making. Ayzel and Heistermann [90] train deep learning based rainfall-runoff models and found that those models require more data to calibrate than a hydrologic physics-based reference model. What's more, there may be a lack of physical interpretability because of fact that the surrogate model only comes from the analysis of data coming from executions runs of the reference model [11]. Indeed, it is difficult to draw physical understanding from learned model as it is hard to interpret. The predictions of the surrogate model may not be easily understood, may be implausible or lack physical consistency [11]. Even though some approaches try to include domain knowledge in the learning process [11], this requires high expertise and takes time to do, which constitutes a barrier in the context of tailoring scientific models to support decision making with a systematic approach. In the case of decision making with high stakes, the lack of transparency also raises questions about the relevance and credibility of such models.

3.3 Approximate Computing Approach

The Approximate Computing (AC) approach adopts the point of view of a software engineer as the simulation model is considered as a white box and the related techniques focus on the computing aspect of the implementation of model (*i.e.,* the scientific software). The approach is widely used in computer science and related fields (*e.g.,* machine learning, machine vision, multimedia processing, signal processing, database search, robotics, gaming) to relax the computation accuracy so that the performance (*i.e.,*execution time or energy consumption) of non-critical software is improved [91]–[93]. It stems from the observation that performing exact computation requires a high amount of resource and that allowing selective or occasional approximation can provide disproportionate gains in efficiency [92] (*i.e.,* time execution and/or energy consumption). In some cases, the constraints put on accuracy can be loosened and it enables this efficiency gain by approximating the computation. It relies on the difference between the accuracy required by the developer or the user and the accuracy provided by the execution of the software. It introduces approximation into the program execution process while producing acceptable outputs with respect to the purpose of the application thanks to an approximation strategy. The approximate version of the software then constitutes the surrogate model.

Approximate Computing has a wide range of application and the techniques can be

classified according to their stack layer of application (*e.g.,* circuit, architecture, algorithm, application). Many classifications exist [91]–[96], however, to give as a simple and clear overview as possible, we decide to organise them according to two levels of application : hardware and software. A plethora of techniques exist and various works present surveys of them [91], [92], [94]. Thus, we do not intend to describe every AC technique again in this thesis as most of them are not applicable to the context we study. We aim to give a global overview of the different types of techniques and we focus on the ones most relevant to our study.

3.3.1 Approximate Computing for Hardware

At the hardware level, approximate computing deals with a less accurate circuit [97], [98] for computation or a reduction of voltage supply for some hardware components. The goal is to act on the execution environment of the software to have a gain of efficiency (*i.e.,* in terms of performance and/or energy). Approximation on the hardware layer mainly corresponds to approximating processing units or providing different hardware-supported data types. The circuits are targeted to perform the trade-off by lowering their supply voltage or modifying their original function (*e.g.,* voltage overscaling [99], [100], precision scaling [101]). It also aims to sacrifice the computing quality of processor, memory and storage components to improve the computational performance (*e.g.,* approximate storage [102], memoisation [103]).

The AC techniques regarding hardware and architecture do not match the context of the thesis. Applying those techniques requires very specific expertise in the field of computing or is dependent to specific hardware resources, and it does not meet our desire to provide a systematic and easy to implement approach. Hence, only the techniques related to software may be relevant for the issues of tailoring and trade-off that is tackled in the thesis.

3.3.2 Approximate Computing for Software

Approximate Computing is more interesting at the software level to answer the need for a systematic trade-off approach for scientific models. Again, numerous techniques exist and involve different aspects of software. Approximating software can be done by transforming some blocks of the code [104] or by switching between different versions of the algorithm inside the software [105], [106]. Compared to Algorithm Transformation

or Algorithm Selection techniques, Code Perforation techniques have the advantage that the modeller does not need to provide multiple implementations of the same algorithm or task. Those techniques are based on dropping some of the tasks by skipping some part of the code [107]. The tasks skipped can be unspecific parts of the code [108], [109] or they can correspond to loop iterations [110].

The techniques focusing on loop approximation are promising with regard to our context. Indeed, simulation models usually iterate over time or space. For the study of climate change, the simulation generates the evolution of values of the variables across time or space. The goal is to observe that evolution to assess the time (or the space) when the situation may be problematic or to get the state of those variables at a specific time in the future or for a specific location. The iterative process of the simulation manifests itself by the presence of loops inside the code of the software. Thus, the idea of executing fewer loop iterations to make the software run faster matches the context of scientific models.

In the next subsections, we describe those loop skipping approximation strategies to understand how the trade-off between accuracy and performance is carried out and to determine if they can be used in the case of scientific simulation models for environmental science.

Loop Perforation

Loop perforation is an approximation strategy which assumes that iterations of loops in a program take time to be computed when not all of them would be necessary to achieve a result similar to that obtained from all of the iterations. Within the software code, the appropriate loops are modified so that only a subset of the iterations is realised. In practice, for a loop incremented by 1 at each iteration, applying the loop perforation technique means changing the increment from 1 to p so that only every p-th iteration is performed [110]. It can be formulated as presented in Listing 3.1. p is called the approximation parameter. N is the original number of iterations in the loop.

```
for (i=0: i<N; i+=p){
        result(i) = process(input(i));
}
```

Listing 3.1 – Loop Perforation. p is the approximation parameter.

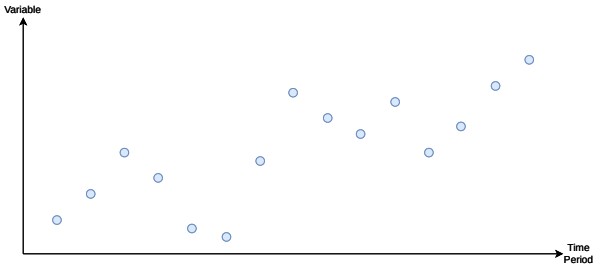
The number of iterations is reduced, fewer calculations are performed and a gain in performance is obtained. The choice of p is made according to the acceptability constraints

made on the software outputs. In all, loop perforation is a technique that achieves the trade-off between accuracy and performance without knowledge of the application domain.

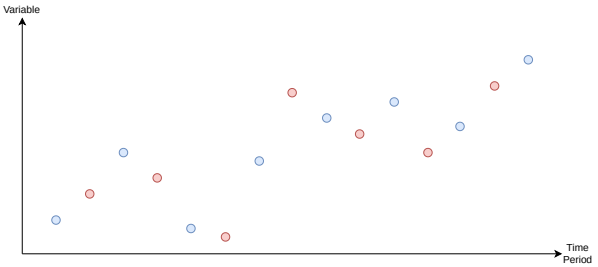
In many studies [95], [111]–[113], the application of loop perforation provides a significant gain of performance (*i.e.,* time execution and/or energy consumption). For instance, Sidiroglou et al. use loop perforation on seven software with various application domains (*e.g.,* media processing, computer vision, data mining, similarity search, financial analysis) [110]. For those software, the perforation does not modify the overall process and purpose of the loop computation. The output produced has the same nature and its meaning is not transformed. The output of the fully executed loop and the output of the perforated loop are comparable. The time execution of the software is faster by 64.20% thanks to the perforation of that loop. In this study, over all the loop perforations and software, the authors observe a significant performance improvement (around a factor of two in running time) for the studied applications while observing a small decrease of accuracy (about 5%).

In essence, that technique enables a significant speed-up by skipping some iterations in loops while ensuring the outputs remain reliable and acceptable with regard to the use of the software. Despite the focus on loop iterations, the technique cannot be applied to simulation models. A simulation is divided into steps that each represents either a new time step or a spatial step for which the behaviour of the system is observed. The output of the simulation is the evolution of that behaviour over all the steps. The Figure 3.2a is a representation of such output. The dots correspond to the values of the output variable along the steps which represents here the simulated time period. Removing some of the steps (whose output values are shown in red in the Figure 3.2b) means performing a totally different simulation. This is shown in Figure 3.2c : compared to the reference simulation presented in Figure 3.2a, the approximate simulation has a smaller number of values for the output variable, that corresponds to a smaller number of steps and represents a shorter simulated time period (or smaller spatial area when dealing with spatial simulation instead of time).

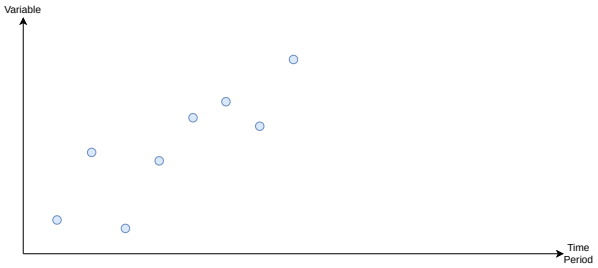
For simulation models with dependent loop iterations, as it is usually the case in environmental science, another factor is also at play. In this situation, during each iteration, the task is to compute the value of the output variable for the considered step with respect to input variables and the last value of the output variable that has been computed in the previous iteration. The new value of the output variable is dependent to its value from the previous iteration and the values of the input variables for the corresponding



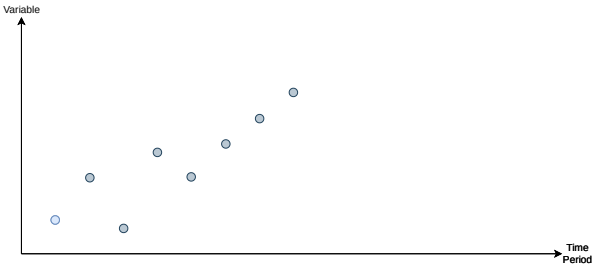
(a) Output of the reference simulation. The blue dots are the values of the studied variable according to the time steps (here representing the time period).



(b) Representation of the skipped iterations with $p = 2$. The red dots depict the reference values that are not computed by the skipped iterations.



(c) Output of the approximate simulation for independent loop iterations.



(d) Output of the approximate simulation for dependent loop iterations. The grey dots represent the new values of the variable for the computed iterations.

Figure 3.2 – Representation of the Loop Perforation technique ($p = 2$).

iteration. Skipping an iteration means not processing the values of the input variables for the given iteration. The value of the output variable for that iteration is not computed and it is thus not updated from the previous iteration. In the following iteration, the corresponding value of the output variable is computed with its non-updated current value and the corresponding values of the input variables. In this respect, the following values of the output variable are impacted by the skipped computation and the ignored values of the input variables. That behaviour is represented in Figure 3.2d. The first dot is blue as its value is the same as the one from the reference simulation (cf. Figure 3.2a). The following grey dots represent the variable values that are different from the corresponding dots of the reference simulation. Thus, they are also different from the associated blue dots present in the Figure 3.2c which presents the situation of simulation models with independent loop iterations.

Usually, simulations in environmental science, and more particularly, when dealing with the study of climate, involve some climate or weather variables such as the rainfall quantity. For a model that studies the evolution of the sea level according to the climate every day for a period of 50 years, removing some steps, *i.e.,* skipping some iterations in the main loop of the software, equates to removing some time steps and some events from the climate scenario (*i.e.,* values taken by the corresponding variables). The output is the evolution of the sea level for a shorter period than 50 years and for a different climate scenario. The time periods and the scenarios being different, the approximate simulation is not reliable and is not comparable to the reference one.

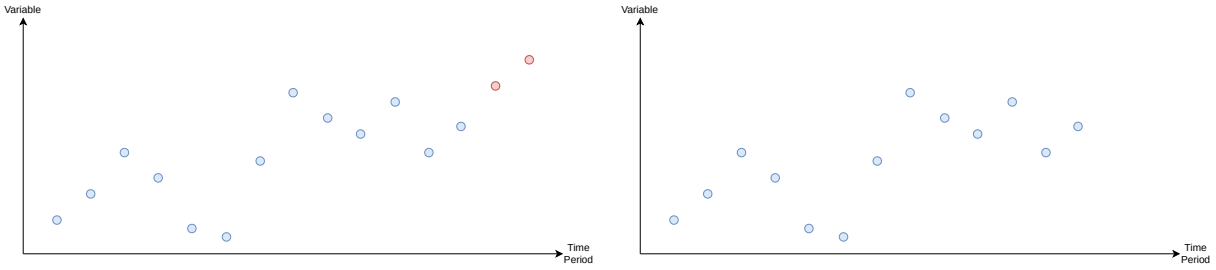
Loop Truncation

Loop truncation, also known as early loop termination, is a method that is based on the assumption that the computation of the last iterations of the loop does not bring much more accuracy to the final generated output and that those last iterations can be cut out. It skips a contiguous sequence of iterations at the end of the loop (cf. Figure 3.3a). The approximation parameter (p) corresponds to the number of iterations to skip (cf. Listing 3.2).

```
for (i=0: i<N-p; i+=){
        result(i) = process(input(i));
}
```

Listing 3.2 – Loop Truncation. p is the approximation parameter.

In [114], Meng et al. apply the technique on a K-means algorithm to show the possible speed-up for recognition and mining applications. The K-means algorithm is iterative and convergent, and, is used to cluster a set of points in a multi-dimensional space thanks to centroids. Each cluster is defined by a centroid which is located at its centre. First, the position of the centroids is randomly defined as the position of some of the points part of the set. Then their positions are refined through iterations. For every iteration, the distances between the points of the set and the centroids are computed. The points are assigned to the cluster corresponding to their closest centroid. The new position of the centroid is computed to correspond to the mean distance to all the points of the cluster. The iterations stop when the position of the centroids remain the same, meaning that all the points remain assigned to the same clusters. The author apply an early termination



(a) Representation of the skipped iterations with $p = 2$. The red dots depict the reference values that are not computed by the skipped iterations.

(b) Output of the approximate simulation.

Figure 3.3 – Representation of the Loop Truncation technique ($p = 2$).

strategy on the algorithm: when the number of points changing their cluster assignment is below a certain defined threshold, the iterations are stopped. A perfect convergence is not reached but a partial one is. A speed-up of a factor 3.5 is observed with an error rate of 1% (*i.e.,* 1% of the points are not assigned to their rightful cluster).

This type of technique does work greatly for models that are based on iterative convergence algorithms which are calculating solutions that are converging over the iterations and thus, doing an incremental refinement. However, this convergence is not necessary observed in simulation models. Indeed, for a simulation involving the time as a variable, each iteration corresponds to a time step of a time period over which we study the behaviour of different variables. Removing some of the last iterations means deleting the modelling of some of the last time steps and last values of the output (cf. Figures 3.3). As for the loop perforation technique, the approximate simulation is not comparable to the reference one. For instance, given a reference simulation modelling the movement of the sea level for a period of a month with each iteration corresponding to a one day time step, skipping the last 7 iterations results in a simulation of the sea level for only the first 3 weeks instead of the 4 weeks. The simulations are not comparable as they do not span over the same period of time. Thus, the loop truncation technique cannot be directly applied on scientific models (encompassing non convergent models) in a systemic approach.

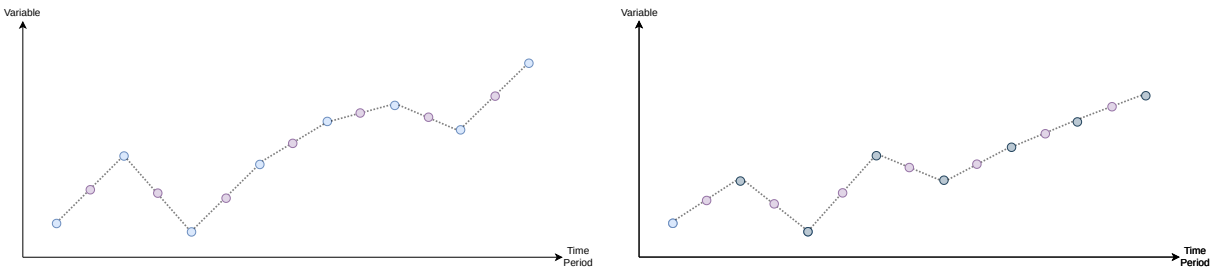
Loop Unrolling

Approximate loop unrolling [115], proposes, in addition to skipping iterations, to interpolate the results of non-computed iterations using an interpolation function (cf. List-

ing 3.3). It enables the accuracy to be generally better maintained than when only skipping the iterations. The method is represented in the Figure 3.4. The Figure 3.4a shows a representation of the application of the Loop Unrolling technique (for $p = 2$) with regard to the computation and the skipping of iterations and associated output values. The blue dots represent the values of the computed iterations whereas the red dots represent the skipped iterations and the resulting discarded output values. The output of the resulting approximate simulation is depicted in 3.4b. The output values of the skipped iterations are calculated thanks to interpolating the computed iterations. In the figure, the interpolation is a linear regression between the two values corresponding to the nearby (previous and following) computed iterations.

```
for (i=0: i<N; i+=p){
        result(i) = process(input(i));
        if i!=0{
                for (j=1; j<p; j++) {
                        result(i-j) = interpolate(result(i-p), result(i), j)
                }
        }
}
```

Listing 3.3 – Loop Unrolling. p is the approximation parameter.



(a) Output of the approximate simulation with a linear interpolation strategy for independent loop iterations. The purple dots are the interpolated values of the non-computed loop iterations.

(b) Output of the approximate simulation with a linear interpolation strategy for dependent loop iterations. The purple dots are the interpolated values of the non-computed loop iterations.

Figure 3.4 – Representation of the Loop Unrolling technique ($p = 2$).

The method is used by Rodriguez-Cancio et al. [115] to trade-off some accuracy for performance in different applications belonging to the domain of multimedia analysis,

musical synthesis, text search and machine learning. One interpolation function they apply is the linear interpolation. After the computation of the non-skipped iterations, the results of the skipped computations are interpolated. With an approximation parameter of 2 (*i.e.,* $p = 2$), the results of the odd skipped iterations are obtained by the linear interpolation of the even computed iterations. The result of the iteration number 1 is generated by interpolating the results from the iterations number 0 and 2. For the application of musical synthesis, the time execution speed-up is by average between 150 and 200% with a approximation parameter of 2, while generating acceptable results (*i.e.,* the audio produced is deemed still audible). The accuracy preservation is better with the approximate loop unrolling technique than with loop perforation.

In all, thanks to the interpolation stage, the output result of the approximate simulation has the same number of steps and represents the same time period (cf. Figure 3.4b) as the reference simulation. However, the approximate simulation and the reference simulation are not comparable in the case of dependent loop iterations. With loop unrolling, the interpolation stage comes after the computation stage. The values of input variables corresponding to the skipped iterations are not taken into account for the whole simulation. In the same way as stated for the loop perforation technique in Section 3.3.2, those values are part of the scenario considered for the simulation and skipping the iterations (and corresponding input values) changes the scenario of the considered simulation. The interpolation stage is carried out with different computed values of the output variable compared to the reference simulation (cf. Figure 3.2a & Figure 3.2c) whereas the computed values are the same as the reference one in the case of independent loop iterations (cf. Figure 3.4b). It means, that for models similar to the one forecasting the evolution of the sea level (cf. Section 3.3.2), the climate scenario taken into account in the approximate simulation is not the same as the one in the reference simulation. The approximate simulation is not reliable in this specific case.

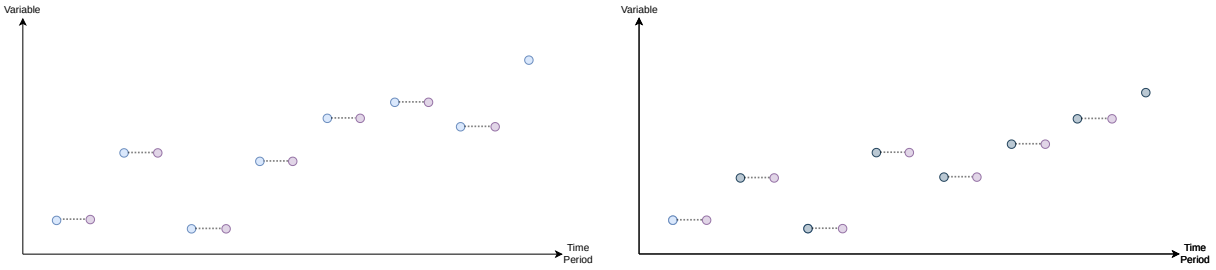
Loop Tiling

Loop tiling can be compared to loop unrolling. In the case of loop tiling, the interpolation function is the identity function (cf. Listing 3.4). The value of the output corresponding to the skipped iterations is the same value as the nearby output resulting from a computed iteration. Compared to loop unrolling, no computation is needed to assign the output value for the skipped iterations. The performance gain is then theoretically better. As the interpolated value is equal to the nearby value of a computed

iteration, the representation of the evolution of the output variable displays a tile pattern (cf. Figure 3.5b), giving its name to the technique.

```
for (i=0: i<N; i+=p){
        result(i) = process(input(i));
        if i!=0{
                for (j=1; j<p; j++) {
                        result(i-j) = result(i)
                }
        }
}
```

Listing 3.4 – Loop Tiling. p is the approximation parameter.



(a) Output of the approximate simulation for independent loop iterations. The purple dots are the interpolated values of the non-computed loop iterations.

(b) Output of the approximate simulation for dependent loop iterations. The purple dots are the interpolated values of the non-computed loop iterations.

Figure 3.5 – Representation of the Loop Tiling technique ($p = 2$).

Rodriguez-Cancio et al. [115] apply the technique (that they called *Nearest Neighbour* in their article) with several applications in the same way they do for the approximate loop unrolling technique. For the music synthesis application, they observe an improvement of the time execution that is comparable to the one with loop unrolling (around 150-200%), but, the overall improvement is slightly lower. Yet, for the application dealing with multimedia analysis, and more particularly a face recognition software, they note a better overall speed-up with the loop tiling. (around 110% in average).

In [116], the authors propose a software framework to identify patterns in data-parallel software and apply approximation techniques. Loop tiling is one technique they apply for a pattern that is based on the fact that the adjacent values of the input array are similar.

This is commonly the case for multimedia processing, where adjoining pixels are alike. The surrounding pixels are approximated by being attributed the same value as the value of the central pixel or a specific row / column of pixels. A speed-up of a factor 2 is observed for a quality loss inferior to 4%.

The tiling approximation is a technique that shows good speed-up results with minimal loss of accuracy for applications that present very similar adjacent input values (*e.g.,* signal processing such as sound, image and video processing). Nevertheless, this technique faces the same issues than the loop unrolling technique which are described in the associated section (cf. Section 3.3.2). For dependent loop iterations, the input values of the skipped iterations are disregarded and the simulation scenario of the approximate simulation is different from the scenario of the reference simulation. To illustrate this point, in Figure 3.5c, the output values of the computed iterations, represented by the grey dots, are not those of the reference simulation, represented by the blue dots (cf. Figure 3.2a).

3.3.3 Relevance for Decision Making

For scientific simulation models, the techniques related to loop approximation cannot be straightforwardly applied. Removing iterations from a simulation leads to different case studies and non-comparable simulations. There is the exception of convergent simulations for which it is possible to safely remove iterations whose results do not provide more information. However, without any information about the convergent criteria, it is not safe to assume that scientific simulations with different numbers of iteration can be comparable. In the case of our context of supporting decision making, fully removing some iterations corresponding to time steps would alter the duration of the simulation period. Even more, meteorologic data are highly variable and cannot be easily inferred between time steps. The recharge rate (ie. quantity of water to enter the aquifer per time unit) varies on a daily basis. Removing iterations would change the climate scenario. Thus, the model generated by the application of loop perforation or loop unrolling on a simulation model related to climate study would not be comparable to the initial model and would not constitute a surrogate in that aspect.

3.4 Summary

Surrogate models are used in place of a reference model when the modelling context is different from the one for which the reference model has been elaborated. One reason and context that calls for the use of surrogate models is the need for faster simulation. Several surrogate modelling approaches exist and each adopts a particular point of view to perform the elaboration of the alternate models. Yet, there is a lack of a specific approach to answer the trade-off needed for decision making support in the field of environmental science that fulfil the needs regarding the application of a systematic approach.

The described approaches present different advantages and drawbacks to their application when using scientific models to support decision making. They are summarised in the Table 3.1. The data-driven approach does not require domain expertise but is time-consuming and/or resource-demanding. The model reduction approach relies on the domain knowledge as well as expertise in mathematics to simplify the complex physics-based reference model. It is then time-consuming to elaborate the new models. Approximate computing (AC) involving loop approximation has a real potential to enable the trade-off of some accuracy for faster execution speed in a systematic manner as no domain knowledge is particularly required for its implementation and it is neither time-consuming nor resource-demanding. The knowledge about the physical processes that is encompassed in the reference model is not modified. However, AC is not yet fully directly adapted to general scientific simulation models that can be encountered in environmental science. Hence, it would be worth investigating whether it is possible to extend this approach to the scientific models under consideration in the context of supporting decision making.

Take-away Message of the Chapter

The approximate computing approach which focuses on approximating the computation of the software is especially interesting and relevant to fulfil requirements of the decision support. However, it is not yet applicable on scientific models in a systematic way. In this thesis, we investigate the possibility of adapting it for scientific models to answer the **challenge 2** (cf. Section 1.3).

Approach	Type	Technique	Criteria regarding a systematic application			Adapted to scientific models
			No Expertise required	Not Time-consuming	Not Resource-demanding	
Data-Driven	Machine Learning	Trees, K-means, etc	✓	✗	✓	✓
	Deep Learning	Neural Networks	✓	✗	✗	✓
Model reduction	Proper Orthogonal Decomposition		✓	✓	✗	✓
	Simplified Physics		✗	✗	✓	✓
Approximate Computing	Hardware	Overscaling, Approximate adders, etc	✗	✓	✗	✓
	Software	Loop Perforation	✓	✓	✓	✗
		Loop Truncation	✓	✓	✓	✗
		Loop Unrolling	✓	✓	✓	✗
		Loop Tiling	✓	✓	✓	✗

Table 3.1 – Overview of the techniques for trading-off accuracy for performance.

Chapter 4

Understanding the complementarity of scientific and engineering models

In this chapter, we describe two collaborative contributions that we have been part of as they help us understand how to define the related challenges and the context we are faced with in this thesis. They serve as foundation for the work we conduct in tackling the challenges in using scientific models to support decision making. They constitute the first steps to undertake before tailoring the scientific models to support decision making. The first contribution, presented in Section 4.1, is the result of a collaborative work gathering numerous researchers from the software engineering domain with various backgrounds and experiences with the diverse types of models[1]*. It proposes a framework dealing with the integration of diverse types of models and data and the role they play in a socio-technical system. The associated section is based on the paper published in the journal of IEEE Software [117]. The Section 4.2 explains the second contribution focusing on the specifics of the development and validation of scientific software, and is based on the paper published in the December 2021 issue of the journal of IEEE Computer [19].*

We first need to understand what the nature of scientific models entails to be able to tailor them by performing a trade-off thanks to approximate computing. We want to understand how their development cycle and their use can affect the trade-off approach. Approximate computing being usually carried out on engineering models, the so-called classical software, we ponder on its applicability on scientific software by comparing them to engineering models. To this end, we are interested in the relationship between scientific and engineering models as well as in the characteristics of scientific models that make them stand out in the software world.

1. 1st Workshop on Data and Models at Bellairs 2019 [14]

4.1 Demystifying the scientific and engineering models and their complementarity

The investigation and work described in this section is the result of a collaborative work gathering numerous researchers from the software engineering domain with various backgrounds and experiences with the diverse types of models[2]*. Thus, the effort and the resulting contribution have to be seen as a work based on a joint sharing of knowledge, expertise and thinking. We present the work we participated in as a researcher with experience and knowledge about scientific modelling, and more particularly for the brainstorming process, the writing of the specific parts related to scientific models and the reviewing stage. In this thesis, this work is particularly relevant as it enables us to demystify the scientific and engineering models and helps us understand their complementarity. Furthermore, this work presents the framework that we use as foundation to define our own exclusive contributions, which are presented in the following chapters. The core contributions of this thesis therefore constitute the implementation of the described framework in the context of using scientific simulation models to support decision making, and are as such a new contribution that demonstrates the application of such framework.*

The word *model* is used in many communities, for a good reason: they share a common definition. A model is an abstraction of an aspect of reality for a given purpose [118]. Models can be used to answer questions with responses that are sufficiently close to reality [119]. Beyond this common definition, different types of models have commonalities, but also notable differences, which are not yet fully understood. Existing work has discussed various types of models [120] and the roles they can play [121]. Here we study these notions of model types and roles with respect to their interplay with the available data. As a result, we provide a conceptual framework that demonstrates the relevant combinations of the different roles of models in engineering and scientific processes. We concentrate on three main types of models: scientific, engineering, and machine learning models.

4.1.1 Types of models

Scientific Model

As presented in Section 2.1.1 & Section 2.2.1, a scientific model is a representation of some aspects of a phenomenon of the world [3]. It is used to explain and analyse the

2. 1st Workshop on Data and Models at Bellairs 2019 [14]

phenomenon (*e.g.,* define, quantify, visualise, or simulate), based on established scientific knowledge defining a theory. A theory provides a framework with which models of specific phenomena and systems can be constructed. Models are validated or rejected by experiments or known theories. Upon validation, these models are typically used to predict future behaviour of the system through simulation or mathematical calculus. Different types of models are used for different aims: conceptual models to improve shared understanding, operational models to refine measurement, mathematical models to quantify a subject, or graphical models to visualise the subject. A holistic view of a phenomenon or a system is assumed and different models can be used at different time- or space-scales.

Scientific models encompass a wide range of representations, such as climate change models, electromagnetic models, protein synthesis models, or metabolic network models. Scientific models typically involve equation-based continuous formalisms such as differential equations, as well as discrete models (*e.g.,* state-based, event-driven, or agent-based models).

Associated data can be numerical or symbolic (*e.g.,* DNA nucleotides). Data are collected, produced, manipulated, and exploited in several ways at different points of the scientific method life-cycle. For example, observation data can be curated and then used in a calibration phase to set the parameters of a model; data can be directly processed by a model; or data can be produced as a result of model simulations.

Engineering Model

Models in engineering disciplines are devoted to support the definition and representation of a targeted system [120]. Engineering models represent concerns ranging from onboard control in autonomous vehicles for braking and obstacle avoidance, to traffic management models, information systems, business rules, etc. They are meant to drive, possibly with some degree of automation, the development of the system-to-be.

Engineering disciplines often use systematic processes and methods in addition to well-defined notations for their models. Those formalisms can be domain-specific (*e.g.,* BPMN or BPEL) or more generic (*e.g.,* UML or SysML). With these formal processes and languages, validation of the models includes the use of formal techniques, simulations, and tests.

Engineering models can represent a means to develop a physical system for a specific purpose that obeys physical laws, or a software-based system (including behaviour, structure, intentions, and/or configuration), or both (*e.g.,* cyber-physical systems). As

such, we also consider engineering models to be those that use decision logic to manage a system based on inputs from environment sensors to determine an appropriate response. In many cases, a feedback loop is used to govern how to adjust the response to account for different types of uncertainty (*e.g.,* errors, changing operational conditions).

During the design of a feedback loop, engineering models must keep track of key elements of the system (*e.g.,* physical and logical elements), which requires processing large volumes of data. For instance, when engineering models are used to control their environment, they have to be able to handle continuous data as opposed to discrete data.

Machine Learning Model

Machine Learning (ML) models are produced by automated learning algorithms out of sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to perform the task. They can be seen as an approximation of the conceptual relationship between a particular input and the expected or a priori unknown target output (cf. Section 3.2).

ML models are used for a wide range of applications, such as image classification, feature extraction, defect density prediction, language translation, or motion planning of robots. Common formalisms include neural networks, Bayesian classifiers, statistical models (*e.g.,* linear regression), and many others.

ML models are obtained according to the inductive reasoning principle, i.e., generalisation from specific cases. This approach implies a certain degree of uncertainty as to whether the specific cases sufficiently represent the rules and principles an ML model is intended to capture. The kind of data used in ML is mostly numeric for regression problems, but also symbolic for classification problems.

4.1.2 Complementarities and Synergies of Models

By definition, a model has a purpose, and thus plays one or more roles with respect to that purpose. A model plays:

— a *descriptive* role if it documents some current or past aspect of the system under study (which can be a software-intensive system or a natural system), facilitating understanding, and enabling analysis.

— a *prescriptive* role if it is a description of the system to be built, driving the constructive process, including runtime evolution in the case of self-adaptive systems

(*a.k.a.* models@runtime).

— a *predictive* role if it is used to predict information that one cannot or does not want to measure (which creates new knowledge and allows decision-making and trade-off analyses to be performed).

Each model type can play more than one role. A scientific model is first descriptive, but its main objective is to become predictive supporting what-if scenarios [13]. Embedded into a socio-technical system, it becomes prescriptive. For example, consider a prescriptive model of a decision-making tool for climate change using a predictive simulator based on a descriptive scientific model of the earth's water cycle. An engineering model typically starts by being descriptive (*e.g.,* a domain model describing key concepts and relationships), and then at design time is refined/transformed into a prescriptive model. But once the system is built as prescribed, the model becomes descriptive again as a form of documentation [122]. An engineering model can also be used as a predictive model: *e.g.,* an architecture model could be used to predict the performance of a specific configuration. An ML model is mostly used in a predictive role with the objective to infer new knowledge given some hypothetical input data. It might also be descriptive of a current or past relationship, or prescriptive if the results are used to make decisions. For instance, consider a prescriptive model of a smart farm where a predictive ML model is used to decide on irrigation plans based on descriptive historical data.

To create a model of any of the above types, knowledge *and* data are needed as input. The proportion of required knowledge or the importance of the availability of the required data to build the models are highly specific to each model type. For example, in ML models, we need problem-specific knowledge to choose the adequate ML technique(s), choose the ML meta-parameters (e.g., different kinds of layers and how they connect in a neural network), choose the input variables and the output variables, and then derive a specialised model from the data. In scientific models, knowledge formulates a hypothesis while data parametrise the model. In engineering models, we mainly use knowledge about the domain and possibly improve or tune the models with data.

Along with the respective importance of knowledge and data in the process of building the models, the order in which models and data are considered is specific to the type of models. Descriptive engineering models primarily start with data, including external data (*e.g.,* expert/domain knowledge expressed in requirements or constraints) or measured data (*e.g.,* exploitation data from previous systems). Engineering models are then used to prescribe the way the future system will be built. With knowledge about which algorithms

are best suited for a problem, ML typically starts with input/output system data or measured data for training, and iteratively (*e.g.,* with feedback loops) revises the model to address the problem at hand, where the resulting models are the main output of the process. In scientific models, the external data (*e.g.,* real-world observations) plays important roles while off-the-shelf models aim to describe existing phenomena and hence are regularly updated and improved.

4.1.3 The MODA Framework

Life-cycle support for current and future complex socio-technical (software-intensive) systems requires us to synergistically combine this range of models through well-founded techniques that leverage their overall benefits to satisfy many different purposes. In order to support this integration through engineering processes, we describe a conceptual *Models and Data* (MODA) framework that explicitly relates the different roles of the model types according to three kinds of data: input/output data, measured data, and external data. The MODA framework provides insight into the integration of the different roles that various model types play, including their data sources and related actions, resulting in a generalised view of common software development processes, technologies, and systems.

The systems we consider range from software-intensive systems (where software is the predominant component, *e.g.,* e-commerce applications) and cyber-physical systems (where software controls physical components, *e.g.,* smart grids) to more general socio-technical systems (where humans are in the loop with software-based systems, *e.g.,* crisis management systems). These systems are data-centric. Data are not only provided to and produced by the running system, but data about the software itself and its surrounding environment is collected (*e.g.,* performance data). All those data are processed by descriptive, predictive, and prescriptive models in order to adapt the system to handle the evolving data.

Figure 4.1 presents the MODA framework. The running software is depicted in red, different kinds of data are shown in yellow, different model roles are represented in white, and the arrows represent actions related to the models and data. Consider, *e.g.,* a *Crisis Management System* (CMS) intended to provide a responsive means to systematically detect crises and deploy resources to mitigate them (*e.g.,* traffic accidents). In this case, the running software typically comprises a distributed system with one or multiple back-ends and databases, sensors, and applications running on the smart devices of the different CMS participants (first responders, drivers, etc.), and vehicles.

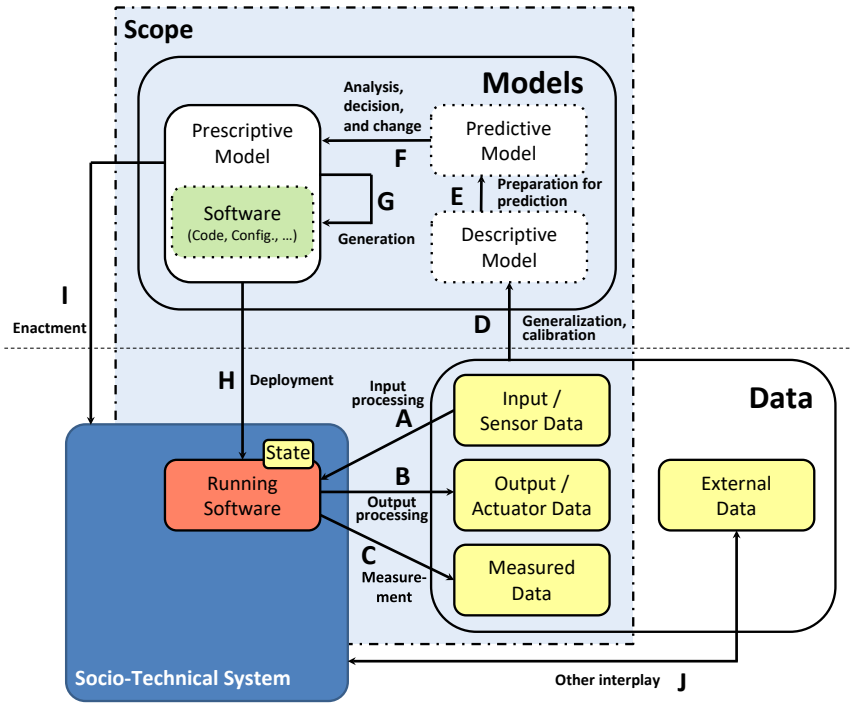


Figure 4.1 – The MODA Framework (dotted boxes are optional)

The running software processes input data and generates output, depicted by the arrows *A* and *B* labelled *Input processing* and *Output processing*, respectively. For CMS, input data include information gathered from phone calls (*e.g.,* number of vehicles involved, affected area, fire), and information gathered from sensors (*e.g.,* GPS data gathered from workers, vehicles, cameras, weather information). Output data include resource assignments and mission-related information communicated to activate emergency personnel, as well as information and requested actions sent to other systems (*e.g.,* the police).

The *C* arrow labelled *Measurement* represents the gathering of metadata or metrics about the running software. Gathering such data requires additional effort, ranging from code instrumentation and logging to human auditing. The gathered data might be filtered or aggregated in real-time, as well as stored for offline use. For CMS, possible metadata could include performance measurements, resource usage (*e.g.,* used network bandwidth), reliability data (*e.g.,* time-to-failure, noise in communication), and detected intrusions.

The last kind of data we distinguish, external data, is any kind of information that is not explicitly within the scope of the software in the current version of the system. For CMS, *e.g.,* historical data about past emergencies or social media data could be considered

to improve crisis management (J arrow).

The D arrow labelled *Generalisation, calibration* represents techniques that generalise from the different kinds of data to yield a descriptive model. These techniques include conceptual generalisation approaches such as abstraction, synthesis, and type induction, but also statistical approaches, regression, differential equation inference, complex event processing (*e.g.,* aggregating many small events into semantically meaningful ones), mining, as well as natural language processing and advanced machine learning techniques. The generalisation can happen in real-time (*e.g.,* for adaptive systems), or offline.

Building a CMS descriptive model includes matching the received data about an event with generic crisis templates to classify the unfolding crisis according to well-known crisis types. Subsequently, the templates are parametrised with specific event information, *e.g.,* the number of victims and vehicles on fire. Models of historical information about crises could be generated by mining historical data. Scientific models, *e.g.,* fire propagation modelling, modelling of physical roadway condition in response to different weather conditions, can be built to help assess the situation. ML techniques can be applied to analyse traffic patterns.

The E arrow (*Preparation for prediction*) refers to preprocessing techniques that combine data and descriptive models to build models that can be used to make predictions. Sometimes descriptive models can be used directly to make predictions (*e.g.,* fault tree analysis). More often, additional processing is required, *e.g.,* applying techniques for inter- and intra-polation, using statistical techniques (regression), preparing for simulation and training of ML models, *e.g.,* neural networks. For CMS, one might prepare a queuing network and then choose and parametrise a simulator to make traffic predictions. Neural networks could be trained to discover hidden behavioural patterns that can be used to prevent potential future accidents or shorten the emergency response time.

The F arrow (*Analysis, decision, and change*) represents decision support activities (*e.g.,* what-if analysis) and the application of consequent changes to the prescriptive model. For CMS, prescriptive models would include mission workflows, safety regulations that should be enforced, UML descriptions of the software solution, algorithms supporting communication, etc. What-if scenarios can be run manually or in an automated fashion (*e.g.,* using hill-climbing / optimising searches, genetic algorithms). Different approaches can be used for enacting a decision. For instance in self-adaptive systems, a decision might require a reconfiguration that can be achieved by making changes to the prescriptive architecture model (*e.g.,* by means of model transformations), or by updating configuration

files. In a software product line setting, foreseen adaptations can be achieved by selecting features that describe previously-designed alternatives, and then adapting the prescriptive model, *e.g.,* by model weaving or merging. For CMS, one could predict that the network noise will become significant in the near future and then trigger the decision to switch to a more robust communication protocol at runtime. Offline extrapolation of current crises data predicting future needs of the CMS might lead to the decision to develop new features for the next version of the CMS software.

The G arrow labelled *Generation* represents the typical software development activities that use high-level prescriptive models (*e.g.,* requirements models) to produce lower-level prescriptive models (*e.g.,* design models or executable code). The techniques used here include model transformations, model instantiation, and compilers. Recently, AI techniques have also been used in this step to optimise the generation process.

The H arrow (*Deployment*) involves deploying and executing or interpreting the low-level, executable models (*e.g.,* code). Here as well, AI techniques are beginning to emerge, *e.g.,* to optimise node configurations in cloud deployments. For CMS, a new architecture model could be distributed to all system nodes to switch to a new communication protocol.

Finally, the I arrow (*Enactment*) represents actions accomplished or enforced in a socio-technical system based on prescriptive models involving human/social dimensions (*e.g.,* policies, laws, standards). For CMS, new driving regulations could be devised that force truck drivers to adhere to stronger safety requirements. Such regulations would have to be enforced by legal means. Similarly, transparent information dissemination policies could be put in place to strengthen the population's feeling of security.

Besides socio-technical systems like CMS, MODA generalises state-of-practice processes, technologies, and other systems. Figure 4.2 highlights MODA's broad applicability with representative instantiations of MODA. We show three different software development processes (waterfall, iterative/agile, and test-driven development; Fig. 4.2a-c), as well as business process modelling and mining approaches (Fig. 4.2d). The generic nature of MODA even enables its use as an underlying structure for explaining business modelling approaches (Fig. 4.2e). In Fig. 4.2f-h, we consider workflows in scientific computing [123], commonly-used machine learning pipelines in software development, and the autonomic computing MAPE-K loop [124]. Note that recommender systems are similar to machine learning pipelines in that a *predictive model* is included in a *prescriptive model* to provide recommendations automatically. Finally, we consider the development of four systems (Fig. 4.2i-l), differing in terms of complexity, availability of data, and

requirements volatility: *(i)* a simple mobile app, *(ii)* a control and command system in an aircraft, *(iii)* a digital twin application, and *(iv)* a smart power grid application. Exemplified by the above processes, technologies, and systems, we have illustrated how the MODA framework is a common reference to guide the use of models, data sources, and their implied actions to improve the integration of different model roles and data sources.

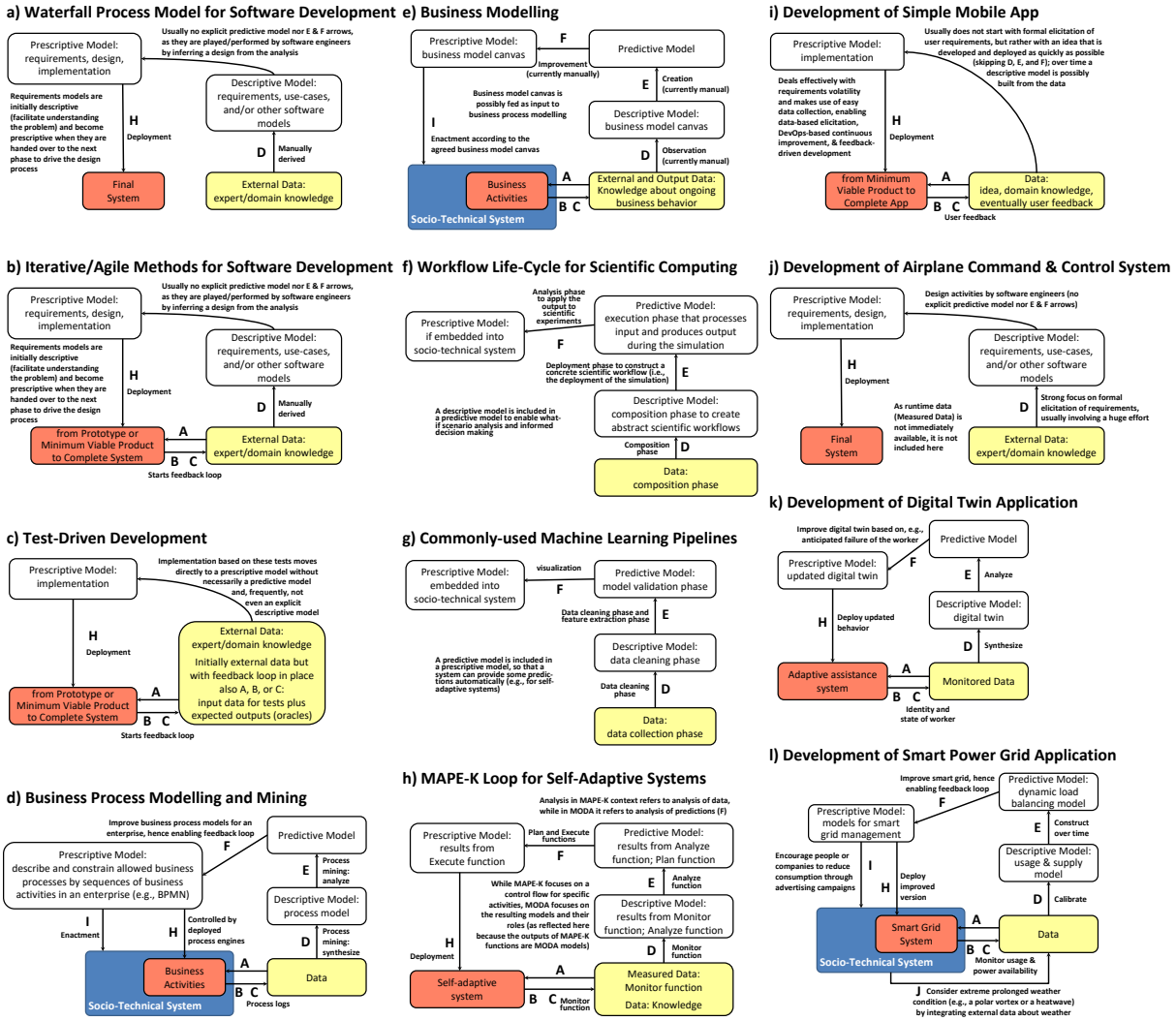


Figure 4.2 – Instantiations of the MODA Framework

4.1.4 Relevance of the MODA Framework

A key objective of software engineering (SE) research in the near future is to enable an engineering-based approach to support rigorous processes and techniques for model and data integration for the increasingly complex and dynamic socio-technical systems of tomorrow. To date, SE researchers have used AI as a tool to support SE tasks (*e.g.,* to improve testing techniques) or applied SE to AI (*e.g.,* to test AI software). Beyond this bidirectional use, we focus on relating the fundamental role of models obtained through AI to the fundamental roles of both scientific and engineering models.

We introduce the MODA framework as a conceptual reference framework that provides the foundations for identifying the various models and their respective roles within any model-based system development life-cycle. It is intended to be a guide to organise the various models in data-centric socio-technical systems.

Such a framework also facilitates the identification of open challenges that need to be addressed in the near future. We mention some of these challenges in the following and organise them according to the framework's arrows C to F in Fig. 4.1. This list is not an exhaustive treatment of all challenges (*e.g.,* quality attributes in general, and ethical considerations of ML in particular, are not discussed). In general, to make MODA effective, efforts are required in *foundational support* for data and model interplay (*e.g.,* protocols and interfaces).

One challenge deals with questions on *guidelines regarding when, what, and how (i) to observe* (*e.g.,* systematic methods to derive (arrow D) descriptive models out of observed data) or *(ii) to measure running software* (*e.g.,* need metrics that take into account the measured data (arrow C) such as data related to performance, load, and execution time). These issues are similar to the monitoring issues faced by the self-adaptive systems community [125] and the models@runtime approach [126]. However, MODA provides a more accurate explanation of the different types of models, and thus of the questions to be answered: what are the methods needed to systematically design the data processing pipeline from observations to decisions? How can we control data quality through the entire processing pipeline? How can established ML techniques be used to support design decisions? What are the optimal uses and constraints related to online-training and offline-training? How can ML techniques be used in data processing that purely runs online (*e.g.,* observation process), as measurement overhead needs to be kept low?

It is also important to *help determine when the different types of models must be made explicit within the process* (*e.g.,* when is it beneficial to have an explicit scientific

model – in addition to an engineering model – as a descriptive model?), and *elaborate semi-automated model transformations that assist the developers in accomplishing arrows D, E, and F*. This need requires a deep understanding of what kinds of factors affect the interplay between a descriptive model and a predictive model (*i.e.,* certain descriptive models make the prediction easier, but their creation requires considerable effort), and how to learn complex models with ML techniques (*i.e.,* models that process complex inputs and/or produce complex outputs). In this context, it is crucial to identify the limitations of the different types of models used in a system. Techniques need to be defined to mitigate these limitations and still be able to provide high-level guarantees. A plethora of model integration work exists mainly for engineering models [127]. Recent work has attempted to integrate ML and SE in Differentiable Programming [128], and ML and scientific models to define a Theory-Guided Data Science [129]. MODA goes further, though, as integration of the three different kinds of models requires a common understanding, and we envision that new kinds of model interfaces will need to be developed to address this heterogeneity challenge.

Finally, systematic methods are needed for the *operationalisation of decision making that apply predictive models to improve prescriptive models of the system* (arrow F), while ensuring important prescribed properties (*e.g.,* safety, security). MODA can again be used to pose relevant research questions. How can we combine useful knowledge extracted from observations of varying nature (traceability information, quality measures, structural/environment constraints) with previous/external knowledge in order to refine the predictive model and enhance or adapt the prescriptive model? How can we systematically deal with data uncertainty (either coming from uncertain data in the descriptive model or from the predictive model)?

We envision the MODA framework to be used as a hitchhiker's guide to explain, organise, and compare complex engineering processes, software development life-cycles, system life-cycles, and technologies, while creating a research momentum to address the open challenges.

4.1.5 What it Implies for Scientific Models and the Support of Decision Making

The MODA framework gives a clear definition and formalism of the approach we want to achieve. It highlights the need for a systematic approach to integrate the different types

of models depending on their roles and the associated data. In our case, it involves the systematic approach of tailoring descriptive models, here scientific models, to make them take a predictive role (cf. arrow E of the Figure 4.1) to improve the prescriptive model that is the decision making process (cf. arrow F). It also underlines the need of collaboration between the different communities of modelling (*e.g.,* SE and ES) to achieve the efficient and relevant integration and complementary of the models in a socio-technical system.

4.2 Understanding the Development of Scientific Software

*The investigation and work described in this section is the result of a collaborative work. We participated in this work as one of the two main co-authors of the associated paper, with equal effort put in the different stages of the contribution (*i.e., *brainstorming, writing, reviewing) as the other main author did. We present this work as it embodies the foundation needed to understand the challenge we face regarding the validation of the fidelity and credibility of the scientific models. We use the knowledge acquired in this work in the main contributions of the thesis which are described in the following chapters.*

In this section, we explore the overall scientific software development process, we provide an integrated view of the scientific computing and software engineering activities, artifacts and roles, and we discuss the trade-offs on the required V&V activities with regard to the computer languages at hand. This work offers a holistic view through which we introduce the existing V&V approaches from the literature to support such activities and the accountability of the respective roles.

4.2.1 The Engineering of Scientific Software

Software engineers generally follow an engineering process, *a.k.a.* Software Development Life Cycle (SDLC), that introduces the required activities to produce software. For instance, the V-model (cf. green part in Figure 4.3) has been introduced to highlight the relationships between each activity of the development life cycle and its associated activity of V&V (cf. Section 2.3.2). In the rest of the section, we use the V-model as a support for the identification of the main software engineering activities, regardless of the associated method (e.g., incremental, iterative or agile). The same activities can be retrieved in all methods, possibly with different time dimensions (e.g., a V-model in each sprint of an

agile method).

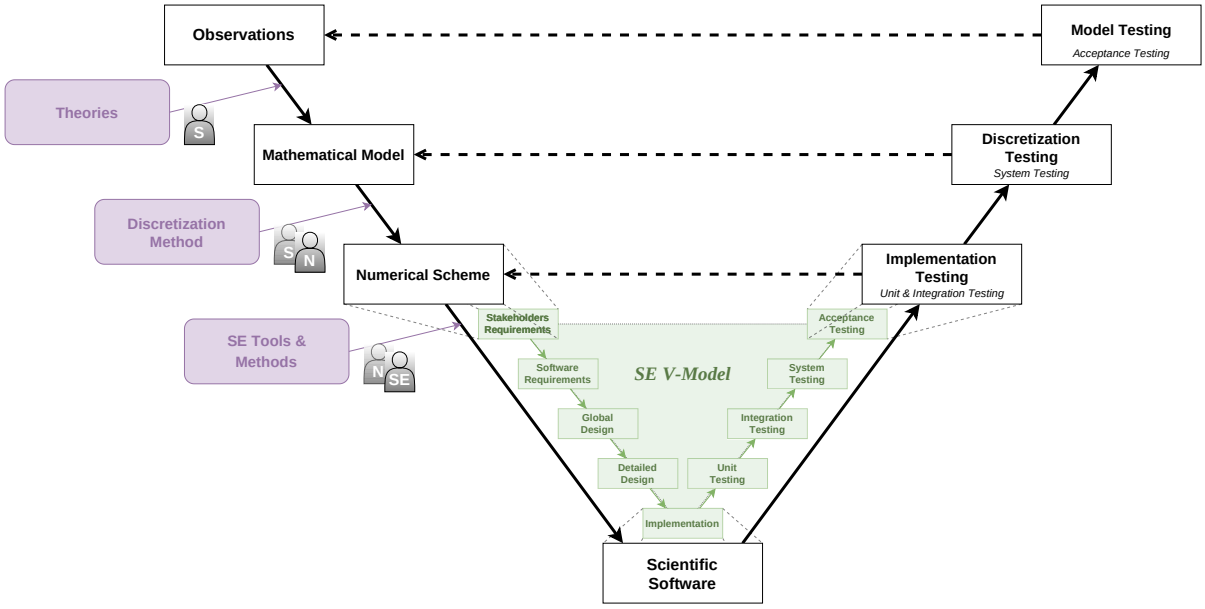


Figure 4.3 – Overall scientific software development process across the V-model.

In Figure 4.3, we propose an engineering process adapted to scientific software development in the form of a V-model, *a.k.a. scientific V-model*, subsuming the original *software engineering (SE) V-model*. We introduce the roles, activities and artifacts of this V-model, with an emphasis on V&V concerns. The V-model is composed of two branches: the first one on the left is top-down, corresponding to the successive elaboration of the various artifacts, and the second on the right is bottom-up, representing the validation process of the scientific software. At each level, a V&V activity on the ascending branch is located across from an artifact on the descending branch. The purpose of the V&V activity is to validate the scientific software with respect to the artifact in question. This is represented by the dashed arrows in Figure 4.3.

The development of scientific software involves several activities that manipulate different types of artifacts. Thus, the design of scientific software based on mathematical models requires the involvement and cooperation of various stakeholders ranging from scientists and engineers to experts in software engineering or numerical analysis. These stakeholders play one of three roles (according to the development context, one person might endorse more than one role): *scientists* as domain experts (S in Figure 4.3), *numerical analysts* as experts in the discretisation of a continuous phenomenon on a computer

(*N* in Figure 4.3), and *software engineers* as experts of software development to deliver the expected services (*SE* in Figure 4.3). It is worth noting the very different background and expertise of these three roles, and the tight collaboration required across the development process. The tasks linked to each role are associated with the activities and artifacts forming the V-model represented in Figure 4.3.

As presented in Section 2.2.1, the implementation of the scientific software is the result of a successive refinement of different artifacts, starting with *observations* and *theories* to elaborate the *mathematical model*, and going through *discretisation methods* to obtain a *numerical scheme*. The implementation of the discretisation process requires specific skills and it is undertaken by *numerical analysts* (*N* in dark grey in Figure 4.3), in collaboration with *scientists* who act as domain experts (*S* in light grey in Figure 4.3).

From that point on, software engineering concerns enter the development process, as well as the *software engineers*. While *numerical analysts* are required as domain expert (*N* in light grey in Figure 4.3), such software engineering concerns require specific skills brought by *software engineers* (*SE* in dark grey in Figure 4.3). This is represented on the middle lower part of Figure 4.3 with a second, nested V-model (in green) corresponding to the usual V-model used in software engineering (cf. Section 2.3.2). This V-model effectively bridges the gap between the *numerical scheme* – which can be considered as the *stakeholder requirements* for *scientific software* – and the software engineering process. It encompasses all the software engineering concerns to obtain a reliable working software, including performance, concurrency, targeted architectures, memory management, data access, and so on. The different artifacts and steps are the same as described in Section 2.3.2.

Once the software engineering V&V concerns have been addressed, the conformity of the scientific software to its original mathematical model is ensured throughout a step we called *discretisation testing* (cf. Figure 4.3). During this testing activity, *numerical analysts* aim to validate whether the scientific software produces results that are within an acceptable margin with regard to the results that would be obtained by analytically solving the mathematical model. The techniques used for discretisation testing vary in their level of rigour, depending mostly on the availability of an exact solution to the mathematical model. Without such a solution, techniques such as symmetry, conservation, or Galilean invariance testing can be employed [130].

These techniques leverage domain knowledge that the numerical solution should exhibit specific characteristics such as symmetry when provided with symmetric geometry

and conditions, conservation of some physical properties (*e.g.,* mass, energy), or be unaffected by operational changes such as inverting the axes of the coordinate system. Note that, as these tests rely on domain knowledge, some might not be used from one field to another (*e.g.,* when no symmetry is present). Another technique that can be used without an exact solution is code-to-code comparison [131], which consists in running a simulation of the same mathematical model embedded in another piece of scientific software, and comparing the results.

Yet, obtaining the highest degree of confidence in the discretised model requires techniques leveraging an exact solution to the mathematical model. When an analytical solution is not available, exact solutions can be obtained through the use of the method of manufactured solutions [132], which constrains the simulator to compute a solution chosen beforehand (the so-called manufactured solution). Whether analytical or manufactured, an exact solution allows to employ rigorous techniques such as discretisation error quantification, iterative convergence testing, and order-of-accuracy testing, respectively in order of increasing rigour [133]. In essence, these techniques aim to assess whether the discretisation error, *i.e.,* the difference between the numerical and the exact solutions, tends toward zero as the value of the discretisation parameters decreases, and thus that the numerical scheme is a correct discretisation of the mathematical model.

Finally, the last V&V step we designate as *model testing* is conducted by *scientists*. Model testing (*a.k.a.* model validation) aims to statistically quantify the disagreement between the numerical solutions and the available experimental data, and to assess whether the resulting uncertainty is acceptable within the application domain of the model [64]. Ideally, these data are collected through a set of validation experiments. Validation experiments are a special kind of experiment aiming to measure the conditions of the experiment as precisely and completely as possible, with less emphasis on controlling the environment than for other kinds of experiments [134]. This distinction allows to run simulations of the model whose inputs match very closely those of the experiments, and thus to assess the fidelity, in these precise conditions, of the outputs of the simulations with regard to collected measurements.

Overall, the different testing activities of the scientific V-model can be related to the corresponding steps of the software engineering V-model: the *implementation testing* step can be conceptually compared to the *unit and integration testing* step, the *discretisation testing* to the *system testing*, and the *model testing* to the final scientific software *acceptance testing*.

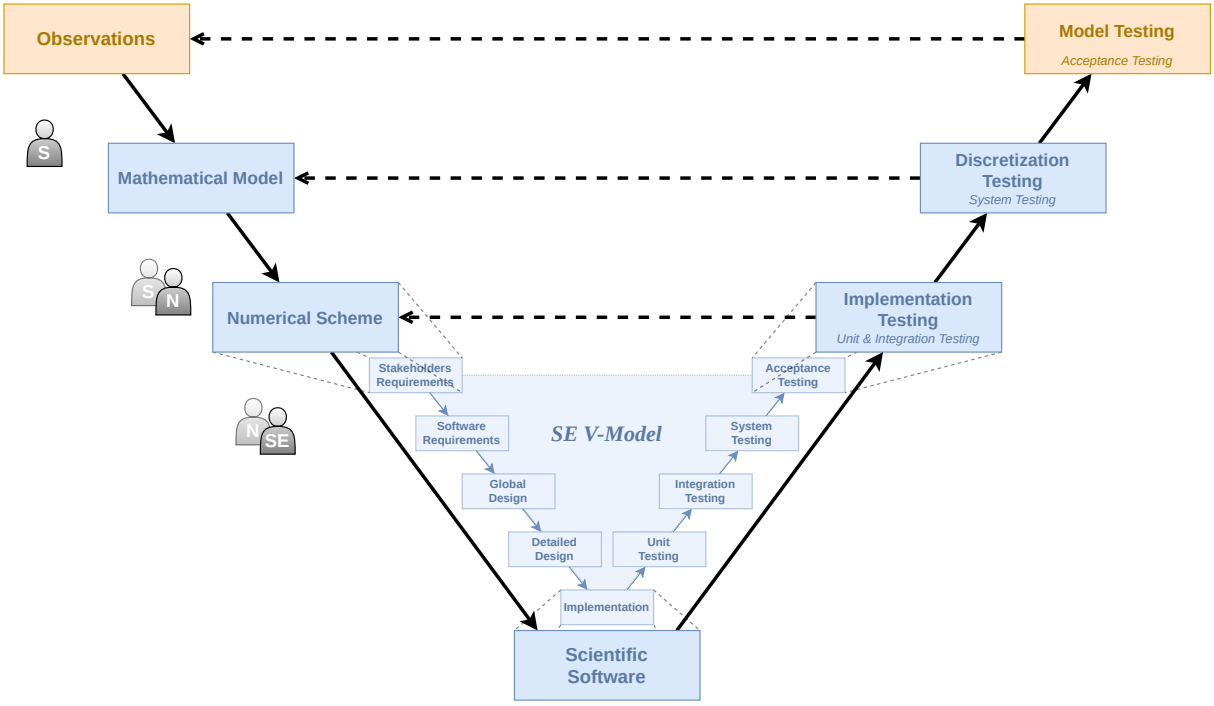
These activities serve the same purpose but use different techniques, according to the nature of the artifact under test. The observations fulfil the role of the stakeholders requirements in that they directly represent the essence of the phenomenon to be modelled. The mathematical model can be considered as the software requirements as it encompasses the global scientific behaviour of the physical system the software should encode. The numerical scheme reflects the purpose of the global and detailed designs as it specifies how the numerical solution to the mathematical model should be computed in the software, and how each step of computation is related to one another, in a similar way as the behavior and interaction of units and components are described in the design specifications.

However, the techniques used are distinctive as they depend on very different natures of artifacts – present on the left top-down branch of the V-model depicted in Figure 4.3 – on which they validate the software. For instance, discretisation testing, which we relate to system testing in the context of scientific software development, aims at validating the scientific software with regard to the mathematical model. Since the mathematical model consists of mathematical equations, associated V&V techniques manipulate mathematical concepts such as the convergence of numerical sequences in the case of iterative convergence testing. Alternatively, the system testing phase of the nested V-model uses the software requirements as a reference to assess the validity of the software. As those are software specifications, the testing techniques manipulate software states and behaviours. They ensure that the different states are consistent from a software perspective. For example, that multiple simultaneous runs of the software do not affect the output of each individual run.

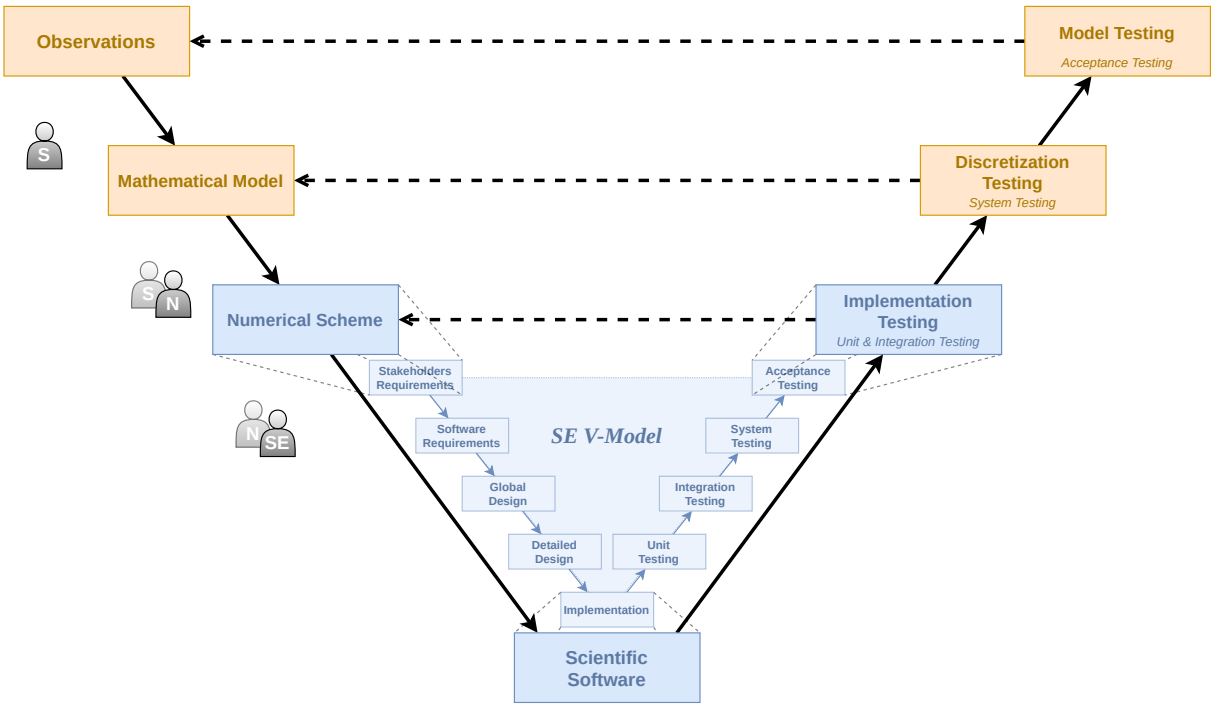
4.2.2 Modelling Comes with Responsibility

The overall development process is complex and challenging when it comes to ensure reliable scientific software. To mitigate this complexity and allow scientists and numerical analysts to directly implement their scientific software, existing computer languages come with dedicated constructs that balance the responsibility between the *language user* and the *language provider*. We explore various computer languages at hand to implement scientific software according to their provided constructs, and discuss the consequences on the expressive power and on the required V&V activities. We illustrate this in Figure 4.4, where we rely on the V-cycle for the development of scientific software proposed in Figure 4.3. In Figure 4.4, the orange and blue colours are used to identify who supplies what artifacts and who is responsible for what V&V activities. The orange colour refers

to language users, while the blue colour refers to language providers.



(a) Scientific software development using an existing simulator (eg., Modflow [135]).



(b) Scientific software development using languages for mathematical models (eg., Matlab, Mathematica).

Reusing Existing Simulators

Figure 4.4a shows the situation where scientists directly use existing scientific software such as MODFLOW [135] or MITgcm [136]. In this situation, language users are scientists configuring the models encoded into an existing piece of scientific software for their specific case. The language used in this case corresponds to a configuration language used to configure simulators and run simulations according to specific parameters (*e.g.,* FloPy for MODFLOW [137]). Language users specify fixed input quantities such as the geometry, or physical modelling parameters of the system and obtain a simulator specific to a certain context (*e.g.,*groundwater flows in a specific watershed area). Then, for each simulation, language users specify the input quantities that vary from one simulation to another, such as the initial and boundary conditions, or the system excitation over time.

Oftentimes, some of those input quantities are unknown (*e.g.,* when they cannot be measured in the field), or known only with a high uncertainty. In such cases, language users will need to conduct *model calibration* to reduce the parametric uncertainty of their model and obtain useful results. Over the years, a number of techniques have been developed to conduct model calibration [138]. Yet, model calibration cannot eliminate every source of uncertainty. Thus, some means of propagating uncertainties over the inputs through the model to obtain the corresponding output uncertainties are required (*e.g.,* sensitivity analysis [69]). While numerous black-box methods exist, white-box (*a.k.a.* open-box) methods that leverage domain knowledge receive more attention [139]. However, such methods require access to internal variables of the embedded model. The latter must thus be exposed as configurable simulation parameters to enable the use of certain state-of-the-art techniques.

In the situation depicted in Figure 4.4a, the simulator offers abstractions specific to the embedded mathematical model (*e.g.,* soil permeability, porosity), which users can parametrise to run context-specific simulations through the underlying scientific software. Thus, developers of a simulator must first endorse all the responsibilities associated with the development of a complete piece of scientific software, as covered in the previous section, including *implementation*, *discretisation*, and *model testing*.

An important language-related responsibility that befalls simulator developers is to clearly identify, document, and enforce as much as possible the validity envelope of the abstractions exposed by the configuration language of the simulator, to prevent its misuse. In the general sense, we define the validity envelope of a language as the set of valid programs one can write with that language. In the case of a configuration language dedicated

to a specific piece of scientific software (such as FloPy), this validity envelope must prevent unsound module and parameter combinations, and enforce the value of parameters to be within their bounds (*e.g.,* non-negative volume of rainfall).

Due to combinatorial explosion, an abstraction of the domain is generally required to identify and enforce the validity envelope of configuration languages, as far as unsound combinations of parameters are concerned particularly. This will, in turn, necessarily result in false positives or false negatives. Techniques such as sensitivity analysis can be used to identify the relationships between the different parameters, and thus detect such unsound combinations.

Languages for Mathematical Models

When existing scientific software do not embed the desired mathematical model, one will need to define their own. Defining a *mathematical model* and deriving the corresponding *scientific software* can be done directly, using languages such as Mathematica [3] or MATLAB. [4] Figure 4.4b depicts the development of such languages and the development of scientific software using them.

In this scenario, language users are scientists implementing their mathematical model using a language that offers constructs at the corresponding abstraction level (*i.e.,* continuous mathematics), and allows them to derive the corresponding piece of scientific software. Thus, they endorse the responsibility of addressing the V&V concerns associated with model testing – identified in the previous section – since those are out of scope of such a language. In addition, in order to validate the implementation of their mathematical model with regard to its specification, language users perform discretisation testing.

However, as explained in the previous section, when no analytical solution to the mathematical model is available, the most rigorous techniques for discretisation testing (*e.g.,* iterative convergence testing, order-of-accuracy testing) can only be used by employing the *method of manufactured solutions* [132], which requires to inject source terms into the discretised model. Therefore, to enable the use of these techniques, the language needs to either provide tools allowing it, or to directly give access to the discretised model. Otherwise, only less rigorous techniques such as those described in the previous section can be employed to conduct discretisation testing.

3. https://www.wolfram.com/mathematica
4. https://matlab.mathworks.com

Language providers are in charge of designing continuous mathematical constructs (*e.g.,* differential blocks in MATLAB's block diagrams) allowing language users to define their mathematical models, and also developing the infrastructure (*e.g.,* compiler) to derive reliable working scientific software from any mathematical model. This language infrastructure performs automated and possibly configurable discretisation and scientific software derivation (as indicated by the blue *numerical scheme* and *scientific software* boxes on Figure 4.4b). So, language providers must handle the corresponding V&V concerns (*i.e.,* the concerns pertaining to the discretisation of their language constructs), as well as the software engineering concerns – encapsulated into the *SE V-model* – which are tied to the implementation of their discretised language constructs.

For that reason, in addition to *software engineers*, *numerical analysts* take an active part in language design, and *scientists* come in assistance to provide the language requirements and criteria for the verification and validation of the language constructs and of their discretisation. While the techniques used to address the V&V concerns related to the discretisation of language constructs and of plain mathematical models are the same, they are here applied to individual language constructs and combinations thereof, instead of complete mathematical models. The objective is to cover as many valid uses of the language as possible to check that its validity envelope subsumes the application domain intended for the language.

Finally, to allow language users to validate the implementation of their mathematical model, tools for discretisation testing should be provided, in particular tools allowing an automated use of the method of manufactured solutions. This allows language users to perform the required validation at the level of abstraction of the language (the continuous mathematical level in this case), which is an essential functionality of software languages.

Languages for Numerical Schemes

When more control is required over what discretisation methods to apply and how to apply them, and when languages at the mathematical level of abstraction do not fit this need, one has to handle the discretisation step oneself. Figure 4.4c depicts a situation in which numerical analysts use languages dedicated to the definition of *numerical schemes* (or with the right abstractions to do so) such as NabLab [140], Julia [141], SciPy [142], GNU Octave (https://www.gnu.org/software/octave/), or Blitz++ (https://github.com/blitzpp/blitz). These languages allow to automatically derive the corresponding piece of scientific software, on which the scientific V&V activities listed in the previous

section can be conducted, without having to handle software engineering concerns.

In this scenario, language users are numerical analysts who apply discretisation methods on mathematical models provided by scientists to obtain numerical schemes, that they implement using a language that offers constructs at the corresponding level of abstraction, and that allows them to derive a piece of scientific software for the discretised model. Therefore, they endorse the responsibility of addressing the V&V concerns corresponding to discretisation testing while scientists endorse the one associated with model testing. Indeed, those responsibilities are out of the scope of a language at the numerical scheme level of abstraction, as indicated by the orange colour of the corresponding boxes in Figure 4.4c. Finally, language users must validate their implementation of their numerical scheme with regard to its specification, which is the aim of implementation testing.

In this situation, implementation testing is performed with the aid of tool support from the language that allows language users to stay at the same abstraction level (*i.e.,* discrete mathematics), without having to consider system concerns such as memory management or concurrency.

Language providers expose discrete mathematical constructs that allow language users to implement their numerical scheme. From the numerical schemes defined with those constructs, the infrastructure of the language (*e.g.,* model transformations, interpreters, compilers, code generators) derives the corresponding piece of scientific software. Therefore, language providers must tackle the system concerns (*e.g.,* memory management, concurrency, data management) and corresponding V&V concerns of the language infrastructure to make it possible to tackle the concerns of the scientific software thus generated. In essence, this means that the concerns must be addressed for any valid numerical scheme written by the language user, which is depicted in Figure 4.4c by the blue color of the *SE V-model* and *scientific software*.

While this task is mainly undertaken by software engineers, the assistance of numerical analysts is required to both define the required expressiveness for the language (which in turn allows to derive the stakeholder requirements from a given numerical scheme), and perform the necessary V&V activities to ensure that the effective semantics of the derived piece of software is equivalent to the semantics of the provided numerical schemes. In other words, software engineers require the assistance of numerical analysts to tackle concerns related to the abstractions at the discrete mathematics level, and clearly define the validity envelope of the language.

Then to foster the robustness of the V&V process spanning the entire V-model, lan-

guage providers should offer tool support allowing language users to perform implementation testing at the discrete mathematics level of abstraction. This tooling can take the form of a dedicated debugger, and optimally includes facilities dedicated to the analysis of numerical schemes such as plot drawing, monitoring variable values and watch expressions (*i.e.,* exogenous expressions of no interest for actual simulations, evaluated for debugging purposes only), and conditional breakpoints and step-by-step execution to detect and investigate faults in the numerical scheme.

Languages for Scientific Software

Finally, when performance is critical to a particular piece of scientific software, be it in terms of memory or execution time, specific hardware and architectures need to be targeted (*e.g.,* heterogeneous and performance-oriented infrastructures such as distributed CPU-GPU architectures). If no language for either mathematical models or numerical schemes supports the targeted type of architecture, system-level languages such as C, C++, and Fortran can be used, conjointly with APIs such as OpenMP, and standards such as MPI and SYCL. Continuous progress in the hardware domain and in the field of high performance computing (HPC) pushes the developers of performance-critical scientific software to use such system-level abstractions, as they are often the only way to leverage the latest HPC-specific technologies.

Figure 4.4d depicts developments where language users express the particularities of their simulator with regard to all the concerns involved in the development of scientific software, ranging from the mathematical model, to the encoded numerical scheme, to system-level concerns such as concurrency, memory, and data handling. In this case, language users must also endorse the responsibilities of addressing the V&V concerns corresponding to each step in the development process, from the V&V of the aforementioned system concerns, to discretisation testing, to model testing. In such a situation, another possibility for language users is to take the role of language provider for an existing language for mathematical models or numerical schemes. This places them as language providers in the situations depicted in Figure 4.4c or Figure 4.4b. As such, they can broaden the range of architectures that can be targeted by the existing language by including their architecture of choice, and thus foster the use of the language in the community as a language at a higher abstraction level than system-level languages that nonetheless allows to target state-of-the-art hardware.

4.2.3 Key Takeaways regarding Scientific Software and Associated Languages

When developing scientific software, dedicated languages for mathematical models and numerical schemes (*e.g.,* MATLAB, NabLab, SciPy) bring benefits to users from the scientific community with regard to languages at a low abstraction level (*e.g.,* C++, Fortran), as low-level V&V and performance concerns are addressed as part of the development of those high-level, dedicated languages. This means that users of such high-level languages, *i.e.,* scientists and numerical analysts, can leave aside software engineering concerns to focus on their area of expertise and the associated V&V concerns, resulting in an increased robustness of the end-to-end development process of scientific software.

In exchange for these benefits to language users, designers of high-level, dedicated languages face additional challenges: the higher the abstraction level offered by a language is, the more steps and complexity in the V&V process of the language itself there are. These languages have to guarantee the correctness and performance of the scientific software resulting from any valid piece of code, mathematical model, or numerical scheme written with them, and provide tools to language users to validate their use of the language. For instance, developing a dedicated language at the mathematical level requires the designers to ensure not only that their implementation and discretisation testing cover any mathematical model written with the language, but also that users are able to perform discretisation testing specific to their precise piece of scientific software.

We highlight in this section the importance of extensively documenting the V&V processes that are conducted as part of the development of a language. This allows language users to clearly identify what V&V concerns are or not already addressed as part of a language. Consequently, users can pick languages according to both their goals in terms of scientific software development, and to the V&V concerns they are able to continuously address themselves according to the state of the current knowledge. This way, scientists are less likely to have to handle software engineering concerns when they do not want to double as software engineers. In addition, this allows to capitalise on the aforementioned development and V&V overhead that comes with the development of those languages.

Another key point is the need for language providers to offer, as part of the language infrastructure, facilities to allow language users to conduct V&V activities tailored to their specific simulation model. This enables the continuity of the V&V process, from user-defined program to scientific software, and fosters both transparency of the complete

development process, and trust in the resulting scientific software.

This work leads to a clarification of the roles of the various stakeholders taking part in the development of scientific software. A reasoned approach for the development of reliable scientific software has been proposed that allows to clearly characterise the validity envelope of this type of software. We believe that the generalised use of our approach will allow to systematically characterise the validity envelope of scientific software, to make it explicit and thus lead to a better and safer use of these software.

4.3 Summary

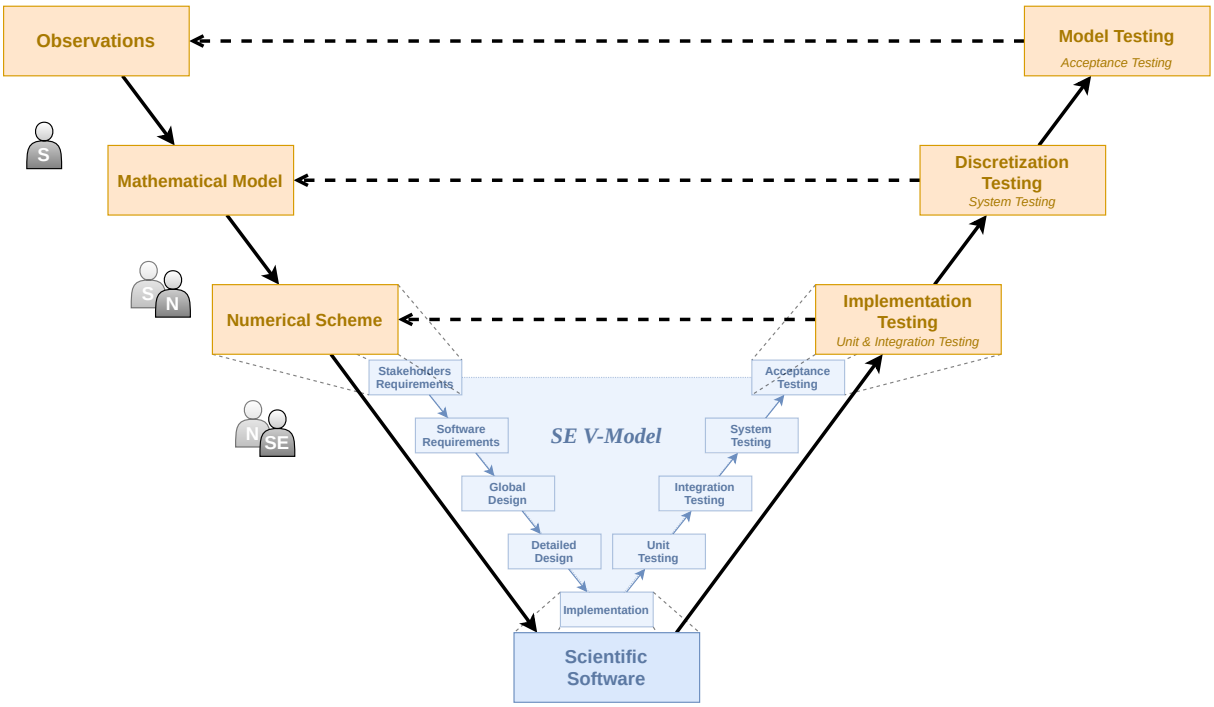
The presented contributions highlight the relevance and need for collaboration between the modelling communities of software engineering and environmental science. There is indeed a need to integrate the different types of models (*e.g.,* scientific and engineering models) according to the role they can take in a socio-technical system as well as a need to elaborate the systematic approaches that allows that integration. The case of supporting decision making is a relevant example. An approach to take into account the primary descriptive role of the scientific models and their newly predictive role for prescriptive purposes is essential to make scientific models a crucial support of decision making. To be applicable, this approach must be able to ensure the reliability of the results produced by the scientific models. As the development and validation of such models are based on several artifacts and the involvement of different roles or stakeholders, the approach has to take into account the modelling objectives and context as well as the domain of application. In all, the researchers in software engineering (and related domains) have thus a decisive part to play in tackling the different challenges faced by the researchers in environmental science. Their expertise is required to elaborate tools and methods (*e.g.,* languages, approximation technique) that can help the scientists ensure the reliability of the scientific models that are to be used in a socio-technical context such as the support of decision making.

Concerning the thesis' issue of tailoring of environmental models, the contributions enable us to answer **Challenge 1** of demystifying the scientific models and their complementarity to engineering models as well as their specificities regarding their development and, more specifically, their validation. The tailoring approach will answer the need for the integration of models with the descriptive, predictive, and prescriptive roles for supporting decision making. Moreover, the approach has to be validated to ensure its reliability with

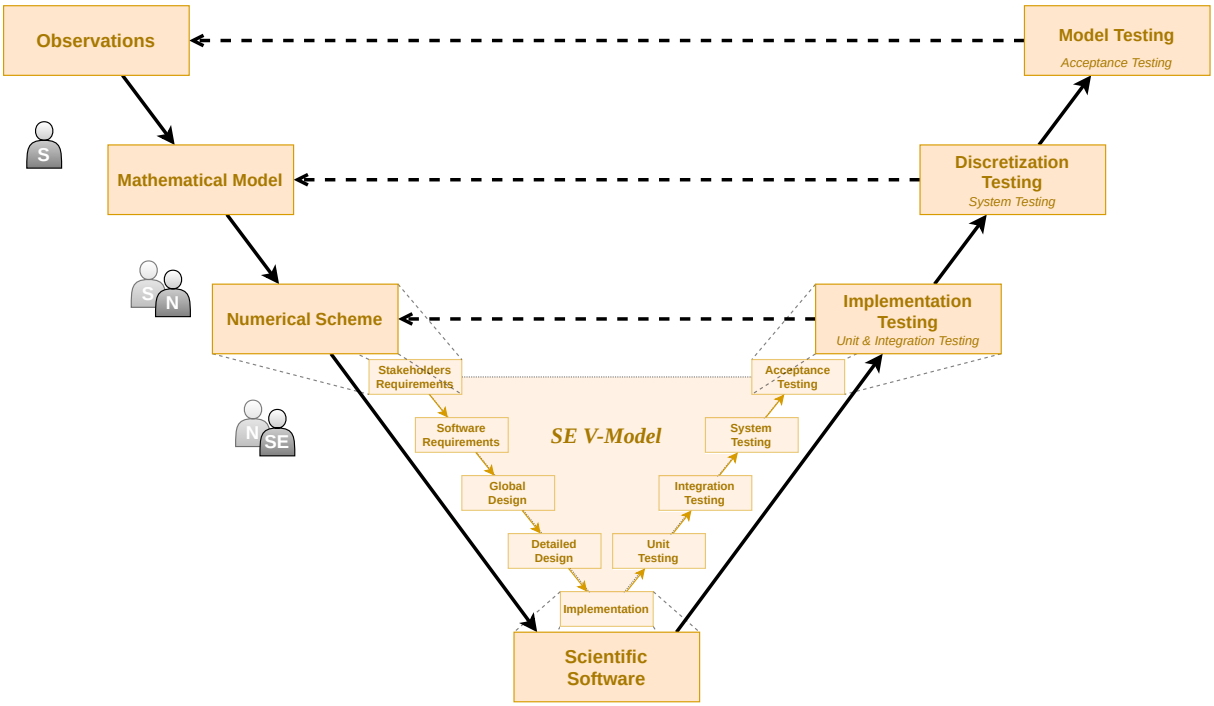
regard to the goal that is decision making and the environmental domain of its application. The validation cannot consist of V&V techniques that focus only on the implementation aspect but also must relate to its application domain. Validation has to involve the skills of the different stakeholder roles that are associated with the different artifacts which are embedded in a scientific software.

Take-away Messages of the Chapter

The systematic tailoring approach of scientific models, based on approximate computing, that we want to achieve, has to integrate the different stakeholder roles involved in the development process to support decision making and to ensure the reliability of the projections by taking care of the validation of the approach with regard to the modelling context (*i.e.,* application domain and objectives) and the specificities of scientific software.



(c) Scientific software development using languages for numerical schemes (eg., Blitz++, SciPy, NabLab).



(d) Scientific software development using languages with system abstractions (eg., C++, Python).

Figure 4.4 – Scientific software development: responsibilities among the language user (orange) and the language provider (blue), depending on the abstraction level of the provided language constructs.

Chapter 5

Approximate Scientific Computing

In this chapter, we propose to investigate AC for scientific simulation models to provide relevant trade-offs between accuracy and performance. This contribution is the main step to answer the challenges related to making scientific models support decision making. We present the loop aggregation technique that enables execution speed-up of the simulations while ensuring the credibility of the results, with a systematic approach with a minimal set-up.

We present a new approximate computing (AC) technique, called `loop aggregation`. According to the main variable of interest, we automatically reduce the main loop of the simulation model by aggregating the corresponding spatial or temporal data to a specific degree. This aggregation can be either applied as a pre-processing of the input data or by model transformation. For example, in the case of an a posteriori study of the causes of soil drying-up with all the necessary data available, it is worthwhile to use the pre-processing implementation. Or, in the case of a crisis situation and the continuous monitoring of a sudden and dangerous flooding episode, the model transformation will be able to manage the available on-going data. We apply `loop aggregation` on a geophysical model of a hydraulic simulation with various input data from different sites and climate series.

5.1 Approach Overview

To handle the issue of non-comparable simulations, that was exposed in the Section 3.3.3 and that are caused by AC when removing some iterations from the loop, we reduce the number of computations while ensuring comparable conditions (eg. same duration of the simulation period for Modflow). We introduce a new AC strategy adapted to scientific models, the `loop aggregation` technique (depicted in Fig. 5.1). This technique is similar to loop perforation (cf. Section 3.3.2) and loop unrolling (cf. Section 3.3.2) since it skips some number of loop iterations but adds specific stages to keep the results and simulation consistent with the baseline. It acts on the main loop of the simulator which is

the loop iterating at the highest level on all the input data and enclosing all the processing of those data. The `loop aggregation` technique has three stages:

— *aggregation* (highlighted in blue): the input values of the main variable of interest are aggregated through an aggregation function.

— *processing* (highlighted in violet): the operations within the loop are only performed on aggregated values.

— *interpolation* (highlighted in pink): the intermediate results are retrieved through an interpolation function.

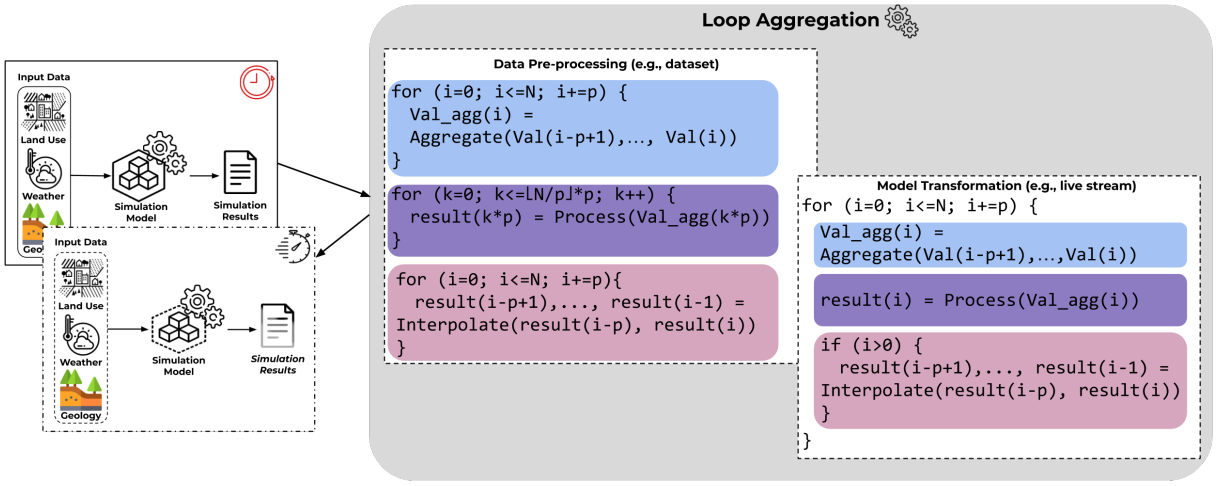


Figure 5.1 – The `Loop Aggregation` technique approach.

As shown in Fig. 5.1, the simulation context guides the type of the `loop aggregation` implementation : data pre-processing or model transformation. Both implementations are equivalent as they reflect the `loop aggregation` approach. The difference is that the three stages are not carried out at the same moment. When all the input data are available before the model is run, a black-box implementation relies on the separation of the three stages (Data pre-processing). The aggregation stage acts as a pre-processing before the model execution, hence the name of the strategy. The values of the input data are aggregated according to an aggregation parameter, p. The processing stage is carried out when the model runs with the model remaining as it is. The interpolation stage is then performed as post-processing after the model execution. With a simulation context of dynamic data flows (ie. stream data), the model transformation strategy is used. It is then necessary to perform the three stages of approximation dynamically to take data into account when they are retrieved. This implies accessing the model source code and adopting a white-

box approach. All the stages and the modifications are made within the main loop of the model and the value of the iteration step is replaced by p, the value of the aggregation parameter.

In essence, the `loop aggregation` technique adds an aggregation stage to the approximation process described in approximate loop unrolling (cf. Section 3.3.2) to enable AC with scientific models. It enables a black-box implementation with separate stages running at different times when all data are available or a white-box implementation with a model transformation when dealing with dynamic data. The number of computations is reduced by the use of the p-parameter and the approximate simulation is still comparable to the reference one. In theory, the technique can be applied to all scientific models with a main loop iterating over temporal or spatial data. There is no need for specific knowledge about the application domain of the model except for information about the use of the model.

5.2 Experimenting Loop Aggregation on a Hydrogeological Model

This section details the application of `loop aggregation` on a hydrogeological simulation model. The goal is to speed its execution up in order for the complex simulation model to be used in the context of decision making related to climate change.

5.2.1 Motivating Example

Hydrologists are working to determine the impact of the sea level rise on coastal aquifers, on increased saturation levels, and associated consequences on inland vulnerability. Between the current state of the aquifers and the predicted sea level rise and climate scenarios, hydrological models are expected to provide predictions.

Those models are based on the three-dimensional software Modflow [135] considered to be an international standard for simulating and predicting groundwater movements. It is based on Darcy's law and conservation principles to represent the groundwater flows. Groundwater flows are essentially modelled by a diffusion equation with Dirichlet boundary conditions when the groundwater level reaches the surface. The resulting parabolic partial differential equation is discretised with a finite difference method and integrated with classical implicit temporal schemes [143]. The quantity of interest to assess the

groundwater-issued vulnerability is derived from the depth to the groundwater level. When groundwater levels rise to some tens of centimetres to the surface, vulnerability becomes difficult to mitigate. Modflow requires both the geological and geographical settings of the studied site (inputs illustrated as *Geology* and *Land Use* in Fig. 5.2) and the meteorological forcing term (represented as the *Weather* input in Fig. 5.2) driving the infiltration and the recharge to the aquifer. This configuration does not change over the simulation period. The meteorological forcing term comes from climate scenarios available on the next century with the estimation of the different elements of the hydrological balance taken here as the input (recharge) to the aquifer. The groundwater flow model provides over the simulation period the location of the groundwater surface, more generally called water table.

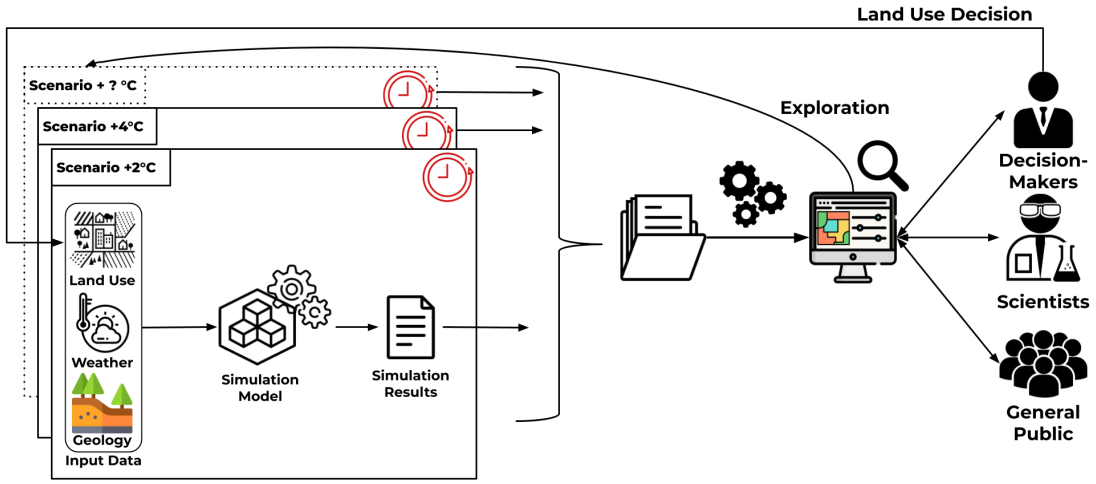


Figure 5.2 – Exploration of several climate scenarios simulations for various stakeholders.

As shown in Fig. 5.2, although the model was created by and for hydrologists, the simulation results can also be of interest to other users. Indeed, with growing awareness of climate change, general public including decision-makers increasingly ask to investigate by themselves the effect of climate change on property and land use planning. Overall, people want to explore the different future climate scenarios, and associated simulations, in an interactive way to make informed decisions or to understand their impact.

Since Modflow is a complex model, its execution can last more than a day. This simulation time is multiplied by the number of scenarios to explore which prevent effective and interactive exploration of the predictions. Making the model run faster would enable such exploration, but the predictions obtained must remain scientifically acceptable to respect the main trends and avoid any significant bias.

Thus, there is a need for finding a trade-off between accuracy and performance. A solution is to simplify the model. However, hydrologists and/or decision-makers may not have the expertise to make this trade-off through model reduction. This raises several scientific questions : (i) Can we make Modflow run faster while maintaining acceptable predictions? (ii) Can we do so without any expertise in hydrology or model simplification and any time-consuming/resource-demanding set-up? (iii) More generally, how to achieve it for scientific simulation models?

5.2.2 The Case Study of Modflow

The case study of our motivating example is based on the prediction of groundwater movements in a watershed near Lestre in Normandy in France to assess the risk of increased saturation at this site. The prediction period for the simulation is 42 years and is represented by 15340 stress periods (ie. time steps) whose duration is set to correspond to a simulated day. The parameters of the model have been set by hydrologists. Executing Modflow with those inputs and non-aggregated data constitutes the reference simulation.

5.2.3 Approximating the Model with the `Loop Aggregation` Approach

We derive the approximate simulations using our `loop aggregation` technique with different aggregation parameters (p) to assess the variation of the simulation execution time.

In the case of Modflow, the main loop iterating over the stress periods, the computation reduction is done by removing some of them. The values of the recharge rate are aggregated as it is the variable of interest. As all the input data are available upstream, we perform our `loop aggregation` approach on those data following the Data Pre-processing implementation. We experiment two strategies for the aggregation stage (detailed hereafter) and linear regression for results interpolation.

Strategy with the Mean as Aggregation Function of the Recharge

The recharge data are aggregated for p being equal to 2, 7, 30, 90, 182, 365, 730 and 3652, corresponding to stress period durations of 2 days, 7 days, 1 month, 3 months, 6 months, 1 year, 2 years and 10 years. Those values of p were chosen to represent meaningful periods for hydraulic and meteorological events.

The mean is chosen as the aggregation function. The choice has been made according to the use of the recharge rate and according to the advice of experts to maintain the overall flux balance. To ensure comparable approximate simulations, the aggregation has also to impact the values of the stress period duration. Indeed, the simulation period must represent a span of 42 years. The stress period duration of the aggregated stress periods is thus changed into the value of p. For instance, with $p = 2$, the inputs are modified as shown in Fig. 5.3a.

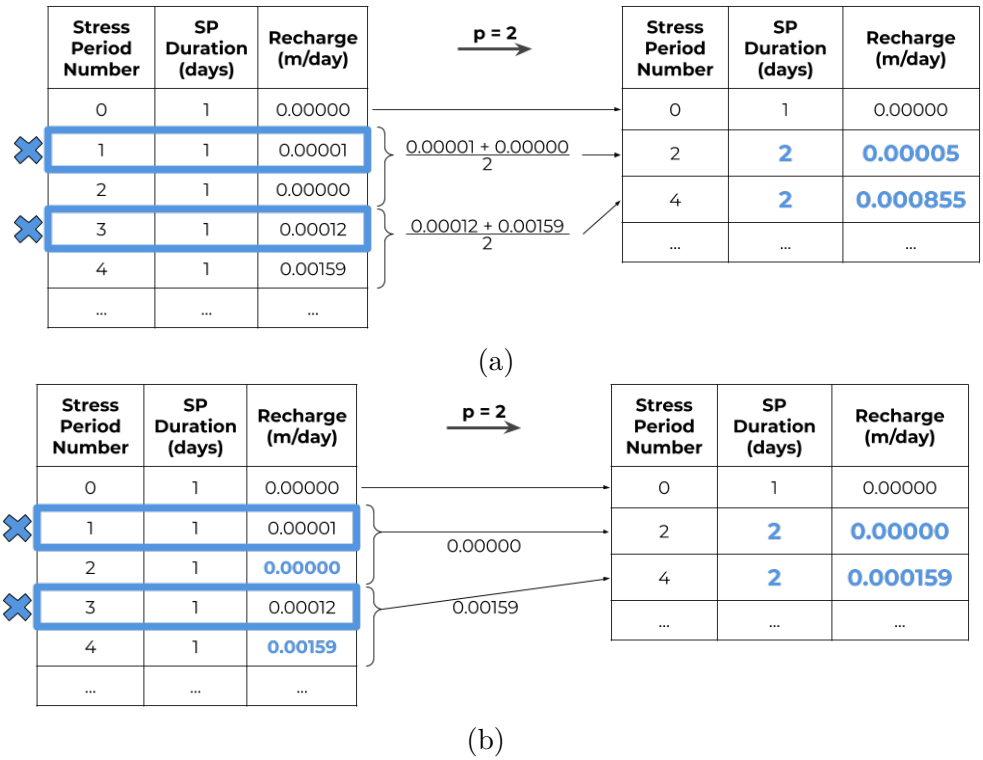


Figure 5.3 – Aggregation strategies with $p = 2$. a : mean as the aggregation function of the recharge. b : p-th value assigned as the aggregated value of recharge.

Strategy with Assigning the p-th Value as the Aggregated Value of Recharge

The aggregation is again carried out for p being equal to 2, 7, 30, 90, 182, 365, 730 and 3652. The stress period duration is changed into the value of p and the recharge value of the p-th stress period is assigned as the mean recharge value for the corresponding aggregated stress periods. The aggregation is carried out as presented in Fig. 5.3b for $p = 2$.

5.2.4 Conditions of the Experimentation

Modflow is run as a Fortran executable with compiled code for computing the groundwater flows in the aquifer. A wrapping software layer written in Python by hydrologists is used to configure Modflow and format the simulation inputs. The version of Modflow used is MODFLOW-NWT-SWR1, the U.S. Geological Survey modular finite-difference groundwater-flow model with Newton formulation and with the version number 1.1.4 released on 04/01/2018 associated with the SWR1 which version number is 1.04.0 released on 09/15/2016.

The experimentation is done on a single node with a Intel(R) Xeon(R) CPU E5-2650 v4 processor with 2.20GHz. Each simulation is run on a single core with 2 threads and 8GB RAM. The model is embedded inside a Docker image deployed on a virtual machine for each simulation. The virtual machine is a Alpine Linux 3.4.3 amd64. Through the Docker image, the memory available for each simulation is limited to 2Gb. These measures are taken to limit variations in the experimentation environment.

5.3 Evaluation

In this section, we validate our ability to apply the `loop aggregation` on our motivating scenario presented in 5.2.1. The goal is to answer the following research questions :

RQ1: Is `loop aggregation` able to perform substantial performance increase while maintaining meaningful results for experts?

RQ2: Is the `loop aggregation` technique able to produce relevant trade-offs for various input data such as climate scenarios and geographical sites?

5.3.1 Acceptation Criterion

The hydrological model is based on Modflow, which is a previously developped simulator (cf. Section 4.2.2), that has been calibrated and validated beforehand. Hence, the validation of the outputs resulting from the approximation simulations focuses on the validation with regard to the domain application (cf. model testing in Section 4.2.1). Domain experts have established an approximation indicator called acceptance criterion (here, the H indicator), which represents a threshold under which the indicator value should remain for the approximated results to be considered acceptable.

Simulation approximations are defined on the quantities of interest of the models. Considering the issues of coastal saturation, the relevant quantities derive from the proximity of the aquifer to the surface, what we can call the zone of saturation vulnerability. When the top of the aquifer (water table) approaches the soil surface at a distance smaller than d_c, water resources, soil humidity, flooding risks and other human activities are impacted. The characteristic distance d_c depends on the type of human activity (eg. agriculture or cities).

However, the transition from the water table being outside the vulnerability zone to being inside it is not sharp. Rather, there is a transition zone from a zone with no vulnerability, when the aquifer is deep enough, to a zone with vulnerability, when the aquifer is close to the surface (Fig. 5.4). The width of the transition zone will be noted Δd_c and be taken as a linear function of d_c with $\Delta d_c(\boldsymbol{x}) = \alpha d_c(\boldsymbol{x})$ where α is the proportionality factor and $\boldsymbol{x}$ is the position. With d_c typically of the order of 30 cm and α equal to 1/3, the transition width, Δd_c, is of the order of 10 cm.

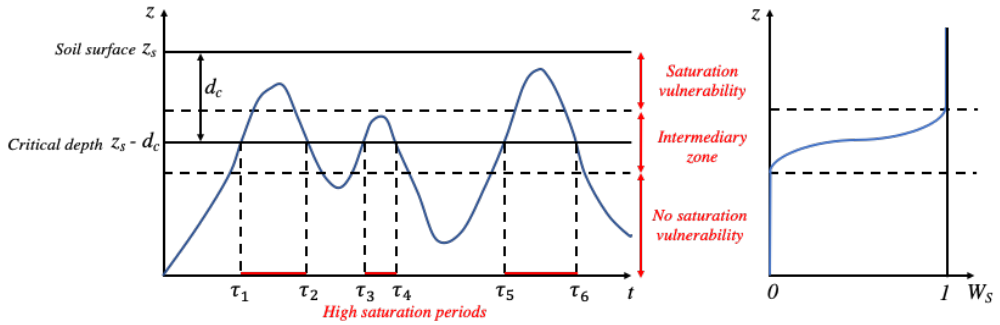


Figure 5.4 – Vulnerability zone and the associated representation of Ws.

Approximations on the quantities of interest will thus be weighted according to their proximity to the surface with the function $W_s(h)$, presented in 5.1, where h is the piezometric level and z_s is the altitude of the soil surface.

$$W_s(h) = \begin{cases} 0 & \text{if } h < z_s - \left(d_c + \frac{\Delta d_c}{2}\right) \\ sin\left(\frac{\pi}{2}\frac{h-\left(z_s-\left(d_c+\frac{\Delta d_c}{2}\right)\right)}{\Delta d_c}\right) & \text{if } z_s - \left(d_c - \frac{\Delta d_c}{2}\right) \leq h \leq z_s - \left(d_c + \frac{\Delta d_c}{2}\right) \\ 1 & \text{if } h > z_s - \left(d_c - \frac{\Delta d_c}{2}\right) \end{cases} \quad (5.1)$$

The H indicator $\|\Delta h\|_2$ on the saturation level is defined by Fig. 5.2. The threshold for the H indicator is set by hydrologists to 0.1 m, meaning that any approximation of

the water table depth within a margin of 10 cm is acceptable. Variables issued by the reference and approximate simulations are indexed by the letters R and A respectively.

$$\|\Delta h\|_2 = \sqrt{\frac{\sum_t \sum_x max\left[W_s\left(h_R\left(x,t\right)\right), W_s\left(h_A\left(x,t\right)\right)\right] * \left(h_R\left(x,t\right) - h_A\left(x,t\right)\right)^2}{\sum_t \sum_x max\left[W_s\left(h_R\left(x,t\right)\right), W_s\left(h_A\left(x,t\right)\right)\right]}} \tag{5.2}$$

5.3.2 Performance Increase with Loop Aggregation

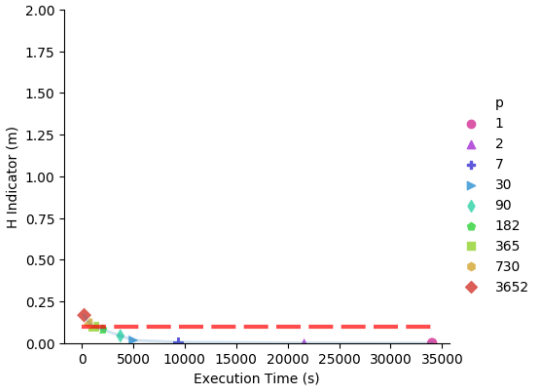
To answer **RQ1**, we assess the performance increase and the acceptation criterion when applying `loop aggregation` on the Modflow model (Section 5.2.1). We use two aggregation strategies with the same inputs (site, ie. Lestre, and recharge series). The reference simulation is run in 34032 seconds, ie. 9 hours, 27 minutes and 12 seconds.

Experiments with the Strategy of the Mean as the Aggregation Function

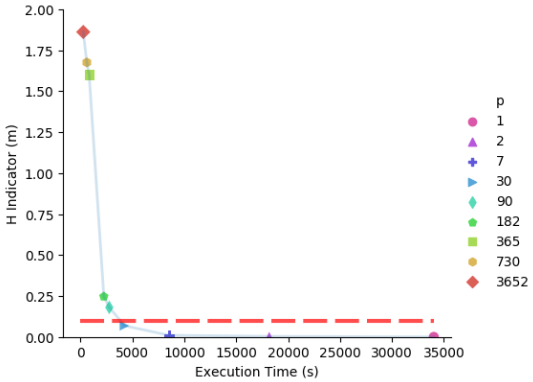
We observe in Fig. 5.5a-c that the more approximated the simulation is, the faster it is. It follows the rational idea that, for a dominantly linear model, the duration of the execution is directly linked to the number of iterations in the loop. With respect to the acceptance criterion, the approximated simulations performed in our experiment with a period of less than one year ($p = 365$) are considered to produce acceptable outputs. Within these acceptable outputs, the shortest execution time (1149 seconds or 19 minutes and 9.0 seconds) is obtained with the simulation of one year stress periods. The execution time is reduced by more than 29 times, a speed-up of more than 96.6%.

Test with the Strategy of the p-th Value Assigned to the Aggregated Value

`Loop aggregation` shows again a performance gain (Fig. 5.5b-d). We observe that the H indicator is higher for the same aggregation rate than for the previous strategy which is consistent with the fact that we introduce more approximation here (ie. the recharge values of the aggregated iterations are not taken into account). The best speed-up is 87.49% with $p = 30$ (Fig. 5.5d).



(a)



(b)

p	Time (s)	Speed-up (%)	H Ind. (m)
1	34032	0	0
2	21607	36.51	1.13E-03
7	9331	72.58	5.51E-03
30	5058	85.14	1.65E-02
90	3759	88.95	4.34E-02
182	2026	94.05	8.16E-02
365	1149	96.62	9.97E-02
730	647	98.1	1.27E-01
3652	241	99.29	1.67E-01

(c)

p	Time (s)	Speed-up (%)	H Ind. (m)
1	34032	0	0
2	18173	46.6	3.62E-03
7	8519	74.97	9.99E-03
30	4256	87.49	6.81E-02
90	2761	91.89	1.80E-01
182	2238	93.42	2.47E-01
365	820	97.59	1.6
730	597	98.25	1.68
3652	255	99.25	1.86

(d)

Figure 5.5 – Evolution of the H indicator and speed-up according to p for the mean aggregation function strategy (a and c) and for the strategy with the p-th value as the aggregated value (b and d). The red dashed line represents the value of the acceptation criterion. c,d : H Ind. = H Indicator.

Stability of the Execution Time Across Simulations

To assess the stability of the execution time obtained for the simulations, we run 30 replicates of the reference simulation and 30 replicates of the simulation with $p = 365$ and $p = 3652$. We use here the approximation with the mean aggregation function. The summary of the results is shown in 5.1. The execution times are stable enough to back the conclusion of substantial performance increase, ie. the standard deviations and relative standard errors are significantly lower than the speed-up.

p	Number of replicates	Mean (s)	Median (s)	Standard Deviation (s)	RSE (%)
1	30	3.57E+04	3.66E+04	3.01E+03	8.42
365	30	1.02E+03	9.68E+02	1.71E+02	16.77
3652	30	2.07E+02	2.00E+02	2.76E+01	13.33

Table 5.1 – Variability of time across replicate simulations. RSE = Relative Standard Error.

To answer **RQ1**, the `loop aggregation` provides substantial performance increase while preserving accepted results for the Modflow hydraulic simulator.

5.3.3 Approach Robustness

To answer **RQ2**, we experimentally explore using `loop aggregation` with other inputs such as climate scenarios (i.e. recharge series) or geographic sites. In these experiments, we use the mean aggregation strategy for the following case studies.

Another Climate Scenario

In this experiment, we use another climate scenario while the rest of the experiment inputs remain the same as in inprevious section. Again, `loop aggregation` leads to a substantial speed up 84.84% ($p = 90$), while remaining within the acceptation criterion.

Replication on other Geographical Sites

We conduct the same experimentation on 22 other geographical sites. The cumulative execution time amounts to 24 days, 15 hours, 44 minutes and 27 seconds. The speed-up between the reference simulation and the fastest acceptable approximation is illustrated in Fig. 5.6 across the sites.

Empirically, we find that `loop aggregation` enables an acceptable approximate simulation for all sites. The gains are not homogeneous but they are substantial. The mean and median speed-up are 91.93% and 95.13% with a minimum of 72.26% (Doville) and a maximum of 99.78% (Graye-sur-Mer).

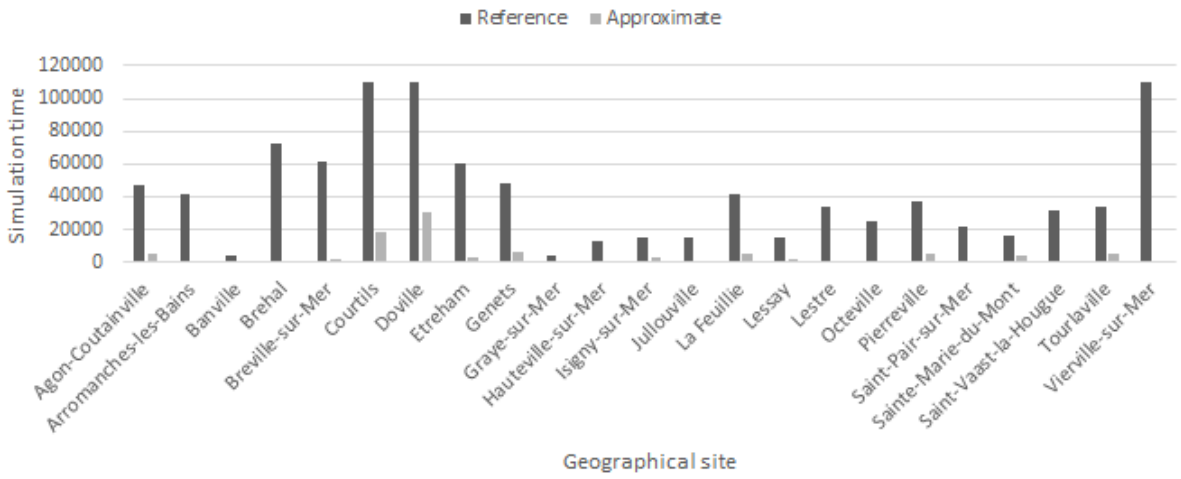


Figure 5.6 – Speed-up across different geographical sites.

To answer **RQ2**, the `loop aggregation` technique provides appealing speed-up while maintaining acceptable results with various inputs (ie. recharge series and geographical sites).

5.3.4 Threats to Validity

Although we empirically validate that the `loop aggregation` approach gives conclusive results for the Modflow scientific simulator in several scenarios, some internal and external limitations remain. Our approach is not analytical but empirical. Our technique may not be the only answer to find trade-offs between performance and acceptability with a minimal set-up. The experimentation was carried out on a specific environment. Care should be taken to ensure that the conclusion can be made with other environments. Moreover, while the indicator used to determine the acceptability of the approximated results were given by experts, it may not meet the expectations of other Modflow experts. To mitigate these limitations, the experimentations have been carried out with various inputs (several sites, another recharge series) and a second aggregation strategy. Regarding external limitations, our implementation of `loop aggregation` is limited to a single scientific simulation model based on a differential equation and to aggregating temporal

data. Following works are needed to apply the technique with an aggregation on spatial data as well as including other scientific simulation models (eg. other forms of equations).

5.4 Summary

We propose `loop aggregation`, an approximate scientific computing technique, that enables to automatically and systematically reduce the main loop of a simulation model by aggregating the corresponding spatial or temporal data. It can either be implemented as a black-box approximation with a data pre-processing or as a white-box model transformation. Our experimentation on a hydraulic simulator shows a median 95.13% speed-up of the simulation time while preserving acceptable results for all the 23 use cases. The approach is supported with a minimal set-up as opposed to time-consuming model reduction and resource-demanding statistical techniques. The flexibility provided ensures that users can explore the simulations according to their specific constraints. In particular, it addresses the **Challenge 2** regarding the systematic application of trade-off between accuracy and execution speed while ensuring the fidelity and credibility of the simulation models.

On a broader note, this approach fits perfectly into the MODA framework by integrating and combining the different roles that the scientific models can play when considered in the context of decision making. In addition, the reliability of the results is ensured given the definition of an acceptability criterion that incorporates knowledge about the application domain (hydrogeology, and assessment of flood risk).

Take-away Messages of the Chapter

We propose a new approximate computing technique by adapting the approach to systematically apply it on scientific models in the context of supporting decision making. The elaborated technique is called **loop aggregation** and is evaluated on a hydrogeological model that is based on the existing simulator named Modflow. The scientific software gains flexibility while ensuring its reliability thanks to the trade-off made by the approach. The model shifts from a descriptive role to a predictive role to answer prescriptive purposes in the context of decision making. It is thus tailored to that new context of use.

Chapter 6

Trade-off optimisation for Decision Making

In this Chapter, we propose an approach towards automatically determining the optimal value of the aggregation parameter to apply loop aggregation without running the reference simulation. The optimisation of the trade-off performed by loop aggregation leads to better speed-up of the simulation execution and by extension to more interactivity and easier exploration of scenarios by stakeholders for decision making.

In Chapter 5, we adapted approximate computing to scientific models to perform the trade-off between accuracy and execution speed by elaborating the loop aggregation technique. The approach can be systematically applied as it is easy to set up and it does not require us to modify the mathematical model embedded in the software with domain expertise (*i.e.,* environmental science and mathematics) or to use high performance computing resources. The reliability of the scientific models is preserved. The simulations are faster to execute while generating credible projections.

In our context of supporting decision making, there is a need to explore new scenarios through simulation by decision makers. The new scenarios to explore can be of different types: they can correspond to other geographical sites (*i.e.,* spatial areas), to other climate scenarios (*i.e.,* temporal evolutions) or to landscape modifications as solutions to assess before implementing them in the real world.

It is possible to apply loop aggregation, the approximate technique we presented in Chapter 5, to explore a new scenario (*e.g.,* assess the flood risk of another area) for which the simulation has not been run yet. The technique can be systematically performed with an aggregation parameter (*a.k.a.* aggregation rate, and noted as p) equals to 2 without much risk on the acceptability of the results. It reduces the execution time quite significantly while only having minimal impact on the accuracy. For instance, with the Modflow model, a speed-up of more than 36% was observed with the aggregation

parameter equals to 2 (cf. Figure 5.5c). Higher speed-up is possible with higher values of the aggregation parameter. For the Modflow model and the example described in Section 5.3.2, the best trade-off is reached with the aggregation parameter equals to 365 for a speed-up of more than 96% (cf. Figure 5.5c).

However, when we want a higher speed-up to provide better interactivity in the exploration of new scenarios, we do not a priori know the acceptable values of the aggregation parameter we can apply loop aggregation with. Indeed, we would need to compute the acceptability indicator to know if a certain value of the aggregation parameter would lead to a surrogate model producing acceptable results. That acceptability indicator is based on the discrepancy between the results of the reference simulation and the approximate simulation. Therefore, the simulation of the reference model has to be executed to determine which values of the approximate parameter are deemed adequate. But running the reference simulation removes the need for a surrogate model (*i.e.,* and approximate simulations).

To explore new scenarios in an optimal way due to loop aggregation, there is a need to determine the optimal value of the aggregation parameter (*i.e.,* its maximal value that leads to a maximal speed-up while still respecting the acceptability criterion). That raises several scientific questions: (i) Can we determine the optimal value of the aggregation parameter to apply loop aggregation for a new scenario? (ii) Can we determine it automatically and a priori without having to run the corresponding reference simulation?

In this chapter, we propose an approach to apply the loop aggregation technique for the exploration of new scenarios by determining the optimal aggregation parameter to use for those scenarios. More specifically, we investigate how to automatically determine that optimal value without running the reference simulation.

6.1 Approach Overview

The challenge is to provide an optimal trade-off between speed-up and accuracy for scientific models in the context of decision making. More specifically, we want to **automatically** and in **a priori** fashion determine the optimal value of the aggregation parameter to apply the loop aggregation technique on the simulation models (*i.e.,* reference models). To tackle it, we propose an approach, that we call *Loop Aggregation Prediction*, to optimise the application of the loop aggregation technique, *i.e.,* to obtain the optimal surrogate model leading to the best speed-up for the simulation model while ensuring

that the acceptability criterion is fulfilled. We leverage that optimisation approach on the commonly-used machine learning pipeline. It is represented by the Figure 6.1 that uses the MODA framework (cf. Section 4.1.3) and more specifically the instantiation corresponding to the Figure 4.2g. The goal is basically to automate the prediction of the optimal value of the aggregation parameter to apply loop aggregation with thanks to a predictive model, so to act on the F arrow represented on the figure.

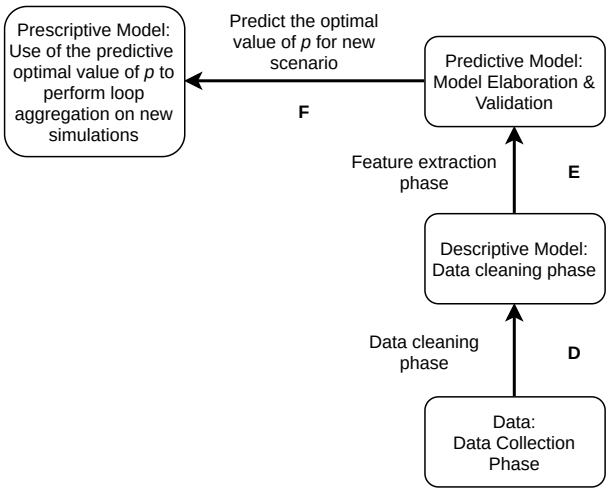


Figure 6.1 – Overview of the *Loop Aggregation Prediction* Approach (use of the MODA framework)

The predictive model is generated using a data-driven approach (cf. Section 3.2). This predictive model is to determine the values of the acceptability indicator (H) that would be obtained when applying loop aggregation on the simulation model for a new scenario with the corresponding examined values of the aggregation parameter (p). From those values of the acceptability indicator, the predicted optimal value of the aggregation parameter can be determined.

The elaboration of the predictive model requires several steps. The predictive model (*i.e.,* ML model, cf. Section 4.1.1) is built by a learning algorithm. The model corresponds to the relationship between the variable to predict and the input data, from which we want to predict the variable. It is inferred by the learning algorithm through the process that is called training. In our context, the variable to predict is the acceptability indicator and the input data encompass the examined values of the aggregation parameter as well as various information, called *features*, that characterise the scenario we want to explore when applying loop aggregation. The relationship between the variable to predict and the input data can be represented by the function g as $H = g(F, p)$ with H being the acceptability

indicator, F the features about the scenario and p the aggregation parameter. The steps of the model elaboration are the following: the data collection, the features definition, the model training and the model validation (cf. Figure 6.2). Each step is described in the following subsections.

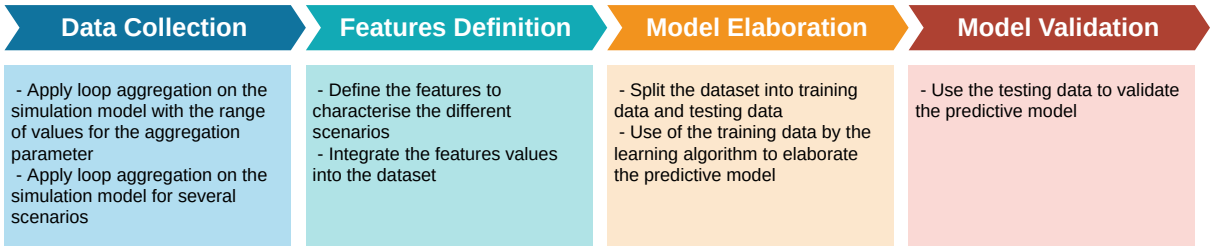


Figure 6.2 – Steps of the *Loop Aggregation Prediction* Approach.

6.1.1 Data Collection

As the predictive model is made through the processing of data by the learning algorithm, the first step is to collect the needed data that will serve as input data. The sample data correspond to the pairings of the values of the aggregation parameter to apply loop aggregation with and the values of the acceptability indicator for the corresponding approximate simulations resulting from the application of loop aggregation on the simulation model and the elaboration of the surrogate model. The variable to predict is the value of the acceptability indicator as it is that piece of information that determines if a value of the aggregation parameter can be used for loop aggregation to produce acceptable results.

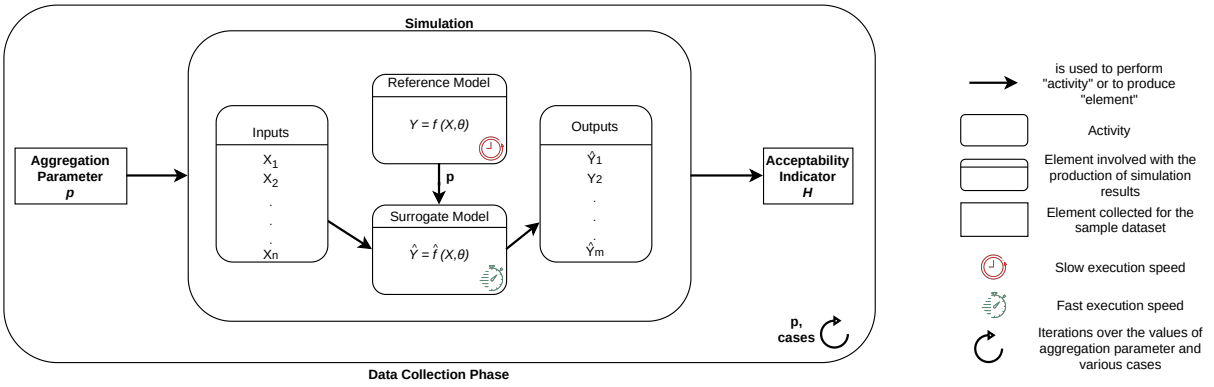


Figure 6.3 – Overview of the Data Collection Phase

The collection of the data is done by running simulations of cases which type is the same as the new scenario to explore. For instance, if the new scenario to explore is a

new geographical site, the simulations are run for a range of various geographical sites. In addition, for each case, the simulations are run for the different values of the aggregation parameter to assess. For example, if the range of values of the aggregation parameter is $p = \{1, 2, 7, 30, 90, 182, 365, 730, 3652\}$, nine simulations have to be run for each case corresponding to a geographical site. The process is represented in Figure 6.3.

From those simulations, the pairings are generated for the range of values of the aggregation parameters as well as for various cases. They can be summarised as shown in Table 6.1

Case	Aggregation Parameter	Acceptability Indicator
C_1	p_1	$H_{1,1}$
C_1	p_2	$H_{1,2}$
...	...	...
C_{Nc}	p_{Np-1}	$H_{Nc,Np-1}$
C_{Nc}	p_{Np}	$H_{Nc,Np}$

Table 6.1 – Sample Dataset after the Data Collection. N_c: Number of different cases. N_p: Number of different values for the aggregation parameter.

6.1.2 Features Definition

With only the pairings of the aggregation parameters and the acceptability indicator, it is not possible for the learning algorithm to establish the relationship between the two variables while taking into account the case. Indeed, to capture the relationship and the specificity of the case, the algorithm needs to have some information to discriminate each case. Moreover, without any information describing the new scenario to explore, the predictive model would predict generic values of the acceptability indicator for each aggregation parameter instead of values which are specific and relevant to that unique scenario.

As such, features that aim to describe and characterise each case are elaborated. Usually, those features are built to best represent the cases in such a way that the relationship between the paring can be extracted more easily by the learning algorithm. Thus, the goal of the features is to give information that highlight the specificities of each case. For instance, for a new scenario corresponding to a new geographical site to explore and the collected cases representing different sites, the features can be about the spatial specificities of the sites (*e.g.,* the surface area, the mean elevation, etc.). The phase of defining those features leads to the sample dataset presented in Table 6.2.

Case	Features			Aggregation Parameter	Acceptability Indicator
C_1	$F_{1,1}$	...	$F_{1,Nf}$	p_1	$H_{1,1}$
...	...	...	...	...	...
C_1	$F_{1,1}$	...	$F_{1,Nf}$	p_{Np}	$H_{1,Np}$
C_2	$F_{2,1}$	...	$F_{2,Nf}$	p_1	$H_{2,1}$
...	...	...	...	...	...
Ci	$F_{i,1}$	...	$F_{i,Nf}$	p_j	$H_{i,j}$
...	...	...	...	...	...
C_{Nc}	$F_{Nc,1}$	...	$F_{Nc,Nf}$	p_{Np}	$H_{Nc,Np}$

Table 6.2 – Dataset After the Features Definition. Nc: Number of different cases. Np: Number of different values for the aggregation parameter. Nf: Number of features.

6.1.3 Model Training

Once the sample dataset is completed (*i.e.,* samples collected and features defined), the next step is to elaborate the predictive model thanks to the learning algorithm. The learning algorithm aims to extract the relationship between the features of the cases, the aggregation parameter and the acceptability indicator. The predictive model is based on that relationship and uses it to predict the value of the acceptability indicator for a new scenario. In all, it can be represented by a function g as $H = g(F, p)$ with H being the acceptability indicator, F the features about the scenario and p the aggregation parameter.

Before elaborating the predictive model, the dataset is split into a training dataset and a testing dataset. As we want to apply loop aggregation to explore a new scenario in an optimal way, we want to assess if the predictive model can determine the optimal value of the aggregation parameter to be used to apply loop aggregation for that scenario. Thus, we split the sample dataset as follow. The *testing dataset* is composed of the data corresponding to the particular case C_k that plays the role of the new scenario to predict the optimal value of the aggregation parameter in the validation step. The *training dataset* is the dataset of the remaining cases that have been sampled. The Table 6.3 and Table 6.4 represent the situation for which the particular case C_k is the last case C_{Nc} of the sample dataset.

The learning algorithm uses the training dataset as inputs to generate the predictive model that match the data the best according to a process that is specific to the algorithm. For instance, the learning algorithm can use a linear regression, a polynomial regression

Case	Features			Approximation Parameter	Acceptability Indicator
C_1	$F_{1,1}$	...	$F_{1,Nf}$	p_1	$H_{1,1}$
...	...	...	...	...	...
Ci	$F_{i,1}$	...	$F_{i,Nf}$	p_j	$H_{i,j}$
...	...	...	...	...	...
C_{Nc-1}	$F_{Nc-1,1}$	...	$F_{Nc-1,Nf}$	p_{Np}	$H_{Nc-1,Np}$

Table 6.3 – Training Dataset. Nc: Number of different cases. Np: Number of different values for the aggregation parameter. Nf: Number of features.

Case	Features			Approximation Parameter	Acceptability Indicator
C_{Nc}	$F_{Nc,1}$	...	$F_{Nc,Nf}$	p_1	$H_{Nc,1}$
...	...	...	...	...	...
C_{Nc}	$F_{Nc,1}$	...	$F_{Nc,Nf}$	p_j	$H_{Nc,j}$
...	...	...	...	...	...
C_{Nc}	$F_{N,1}$	...	$F_{Nc,Nf}$	p_{Np}	$H_{Nc,Np}$

Table 6.4 – Testing Dataset. Nc: Number of different cases. Np: Number of different values for the aggregation parameter. Nf: Number of features.

or a decision tree regression. The goal is for the function to outputs the values of the acceptability indicator of the training dataset thanks to the values of the aggregation parameter and the features.

6.1.4 Model Validation

The last step before the predictive model can be used is to perform its validation. The goal is to check if the predictive model can determine the optimal value of the aggregation parameter for a new scenario. That means to check if the predicted values of the acceptability indicator lead to determining the correct optimal value of the aggregation parameter. In the same fashion as done in Section 5.3.2, the optimal value of the aggregation parameter (p_{max}) is the highest one for which the surrogate model, obtained through loop aggregation, still produces acceptable results (cf. Equation 6.1b). More specifically, it means that, for that aggregation parameter, the acceptability criterion ($H_j < Th$) is respected and the value of the associated indicator H_j is below the defined threshold Th (cf. Equation 6.1a).

$$H_{max} = H_j < Th | max(H_j) \tag{6.1a}$$

$$P_{max} = H_{max} | P_j \tag{6.1b}$$

The validation of the predictive model requires the definition of a validation metric to ensure the predictions made by the model are correct enough. In practice, to ensure that the prediction is correct for a new scenario, we use the testing dataset corresponding to the particular case C_k that plays the role of the new scenario. Only the information about the features of the testing case and the values of the aggregation parameter are given as inputs to the predictive model. The validation then consists in comparing the predicted values of the acceptability indicator and the real values and in checking if the predicted optimal aggregation parameter is correct thanks to the validation metric.

There are three types of predictions:

— correct estimation, when the predicted aggregation parameter is the correct one ($p_{max}^{pred} = p_{max}^{real}$).

— underestimation, when the predicted aggregation parameter is lower than the correct one ($p_{max}^{pred} < p_{max}^{real}$). It is not the optimal value so the speed-up is not the highest possible one, but the results remain acceptable according to the defined acceptability criterion.

— overestimation, when the predicted aggregation parameter is greater than the correct one ($p_{max}^{pred} > p_{max}^{real}$). With the overestimated aggregation parameter, the results of the surrogate simulation do not meet the acceptability criterion.

The correct estimations and underestimations are deemed as acceptable with regard to the application of the loop aggregation technique, whereas the overestimations are problematic as they lead to simulation results for which the acceptability criterion is not met.

The type of prediction made by the predictive model is by definition linked to the case C_k used for the testing dataset. Thus, the predictive model may lead to a correct estimation for one testing case C_{k1} but may also lead to an overestimation for another testing case C_{k2}. As we use a sample of the possible cases in the training dataset to elaborate the predictive model, we cannot ensure that the predictive model would lead to no overestimation for all the possible scenarios to explore that are not part of the collected cases or the testing case. The sample set may not encompass the diversity that is present

in the set of all the possible scenarios. Therefore, we take into account that fact in the validation of the prediction model. We deem the predictive model as a valid one if it leads to a majority of acceptable estimations (correct and underestimations) and only a few overestimations. Thus, the validation criterion for the predictive model is that the error that may be introduced by an overestimation and that is quantified by the validation metric has to be below an accepted value ($Error < \epsilon$). We define the specific validation metric we use in our experimentation in the Section 6.3.1.

6.1.5 Variability of the Predictive Model

When applying this overall approach, one needs to keep in mind that the predictive model results from a complex elaboration requiring several steps and involving many variable factors. Thus, the quality of the model can be impacted by those factors and, more specifically, at several levels: the learning algorithm and the training dataset [144]–[147].

The factors influencing the elaboration and the associated quality of the predictive model, and by extension its validation, are represented in the Figure 6.4 by the different circled numbers. They are:

1. the type of algorithm that is used (*e.g.,* an algorithm based on linear regression or on decision tree regression will produce different predictive model with different prediction quality)
2. the hyper-parameters of the algorithm (noted as λ in Figure 6.4) that are used to tuned the algorithm. Modifying their values impacts the generation of the predictive model. For instance, with a linear regression algorithm, an hyper-parameter dictates the application of the normalisation of the data before inferring the linear regression between the input data and the output data.
3. the values of the aggregation parameter
4. the cases sampled for the training dataset
5. the features used to characterise the cases

As those various factors can alter the validation of the predictive model, one needs to be aware of their impact on the predictive model. Because of the various factors involved in the elaboration of the predictive model, the design space of the predictive model is complex with a high dimensionality, as each factor can be broken down into several dimensions (cf. Figure 6.5). When applying the *Loop Aggregation Prediction* approach, one thus has

to take into account the complex design space of the predictive model when making their choices during the different steps of the elaboration of the predictive model.

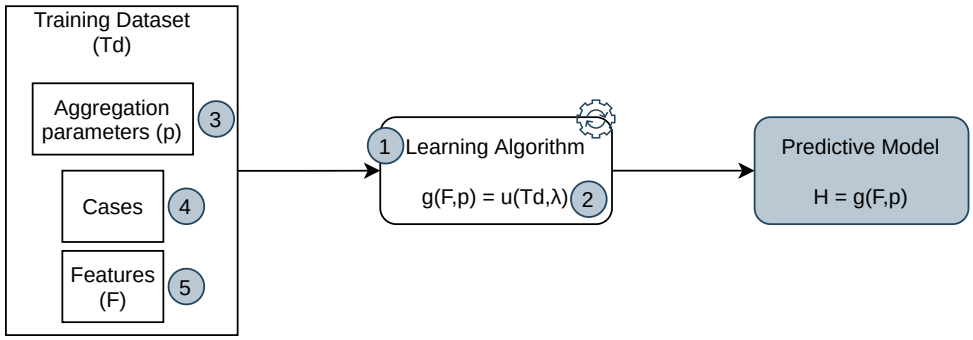


Figure 6.4 – Variables Impacting the Elaboration of the Model Validation. λ: hyper-parameters of the algorithm. $u()$: function representing the functioning of the learning algorithm. $g()$: function representing the functioning of the predictive model.

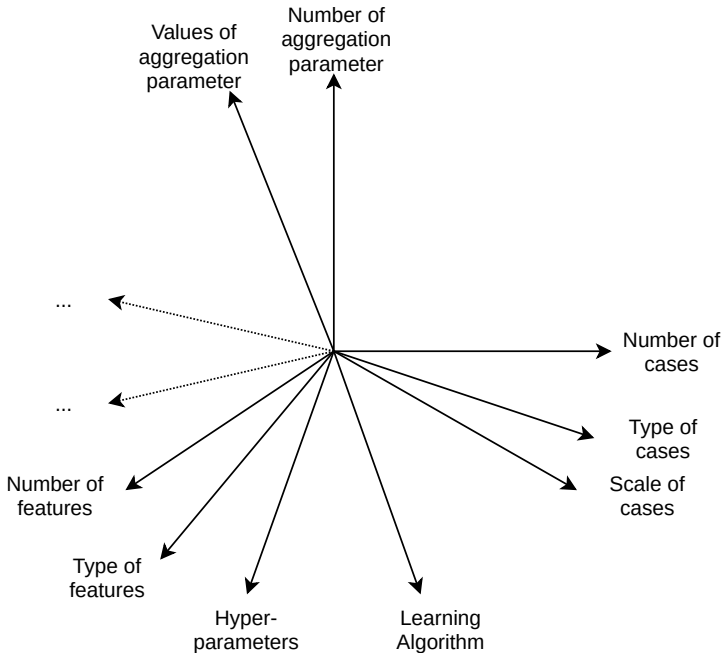


Figure 6.5 – High Dimensionality of the Design Space of the Predictive Model

The exploration of the design space enables to determine the impact of the factors on the accuracy of the predictive model and guides in the application of the *Loop Aggregation Prediction* approach. We explore a sub-space of that design space in following section to

highlight the impact of different factors on the accuracy and validation of the predictive model.

6.2 Experimentation

We apply the *Loop Aggregation Prediction* approach (*i.e.,* optimisation approach) in order to use the loop aggregation technique for new scenarios in the case of the Modflow model that is presented in Section 5.2. When applying the loop aggregation technique, the aggregation function we choose is the mean function, as the results, presented in Figure 5.5c and Section 5.3.2, show that this strategy enables a higher value of the aggregation parameter and a higher speed-up with the Modflow model.

As presented in Section 6.1.5, the accuracy and validation of predictive model is dependent on a number of factors in its elaboration process. Here, we apply the optimisation approach while acting on some of the factors and having the others fixed during our experimentation. We explore a sub-space of the design space of the predictive model to observe the impact of the different factors on the validation of the predictive model. We focus on the training data level by having different numbers and types of cases, different types of features, as well as different numbers of aggregation parameters. The learning algorithm we use is the random forest regressor from the scikit-learn [1] library (version 0.24.1). The random forest algorithm aims to build a predictor with a set of decision trees that grow in randomly selected subspaces of data [148]. It is considered as one of the most accurate general-purpose learning algorithm with a low risk of over-fitting (*i.e.,* inability of the predictive model to to make good prediction for data different from the training set) [148]. The associated hyper-parameters are the default ones of the scikit-learn library except for a number of trees of 1000 ($n\_estimators = 1000$). Various scenarios can be explored with the simulation model, such as climate scenarios, spatial scenarios (*i.e.,* geographical sites), and landscape scenarios corresponding to the potential solutions to enact to anticipate and reduce the risk of flooding. The experiments are performed for the exploration of a new spatial scenario corresponding to a geographical site. Therefore, the cases that constitute the sample dataset correspond to different geographical sites.

In all, we exhaustively explore a design sub-space of the predictive model to investigate the accuracy and validation of the predictive model. The Figure 6.6 represents the sub-space we explore in our experimentation. The dimensions corresponding to the number of

1. https://scikit-learn.org/stable/index.html

aggregation parameters, the scale of cases and the type of features are considered for two different values, whereas the dimension of the number of cases is explored for a range of values. We thus focus more on the impact of the number of cases in the sample dataset on the validation of the predictive model.

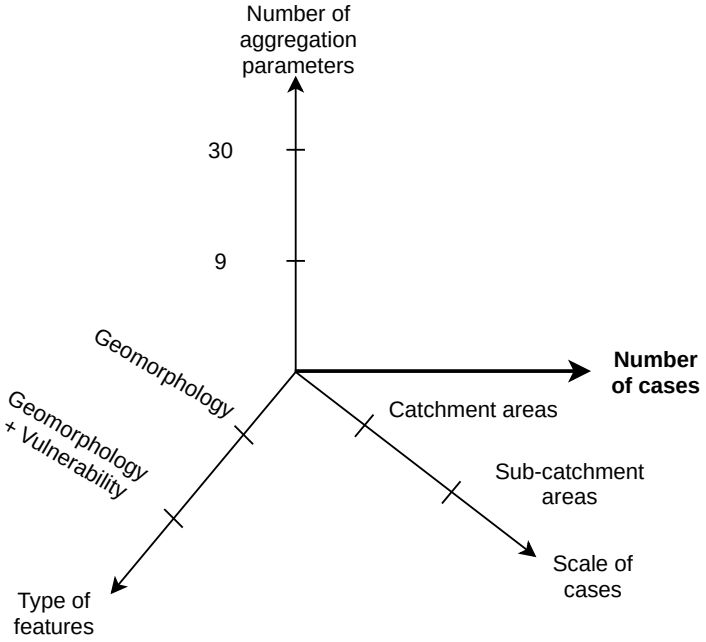


Figure 6.6 – Explored Sub-Space of the Design Space of the Predictive Model

On a practical note, the experimental conditions are the same as the ones presented in Section 5.2.4, except that the virtual machine is Alpine Linux 3.11.3 this time.

6.2.1 Aggregation Parameter

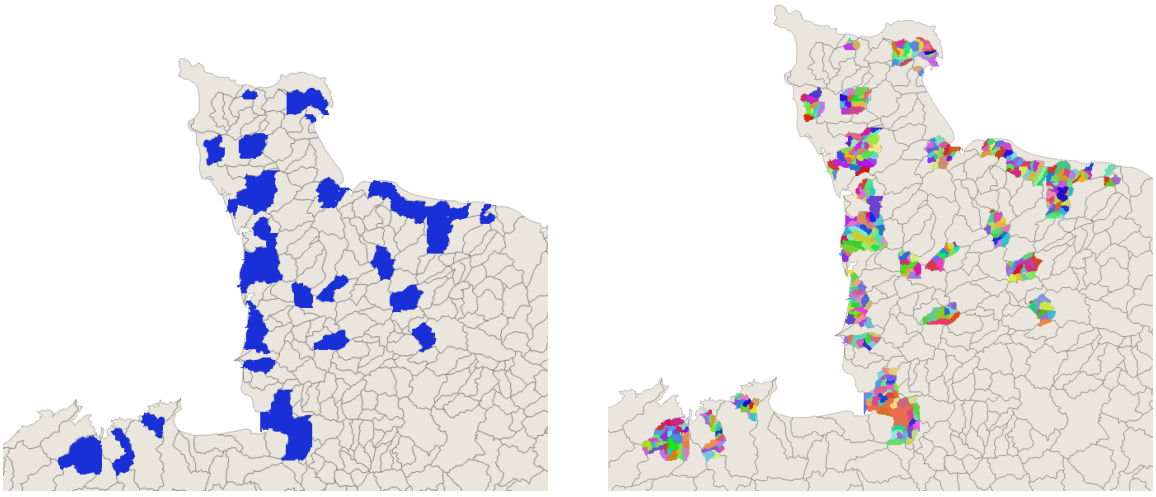
We apply loop aggregation for two different ranges of values of the aggregation parameter (Np) and we collect the data of the associated simulations. The value fo the aggregation parameter corresponds to the aggregation rate made during the loop aggregation technique (cf. Section 5.1). In our context of using the Modflow model, it represents the number of days we aggregate together to reduce the number of computations (cf. Section 5.2.3).

The first range is composed of nine values ($Np = 9$) and the values of the aggregation parameter are the same as those used in the experimentation done for applying the loop aggregation technique in Section 5.2.3 ($p = \{1, 2, 7, 30, 90, 182, 365, 730, 3652\}$).

The second range gathers thirty values ($Np = 30$) and the values are $p = \{1, 2, 7, 15, 21, 30, 45, 50, 60, 75, 90, 100, 125, 150, 182, 200, 250, 300, 330, 365, 550, 640, 730, 1000, 1500, 2000, 2250, 3000, 3182, 3652\}$.

6.2.2 Cases

The cases that are sampled in the data collection step correspond to different geographical sites. Data associated to the geographical sites are collected at two different scales. The first one is the scale of catchment areas (cf Figure 6.7a), whereas the second one is the scale of sub-catchment areas (cf. Figure 6.7b).



(a) Catchment Areas (b) Sub-catchment Areas

Figure 6.7 – The location of the spatial cases (Normandy, France).

We elaborate the predictive model for various numbers of cases ($Nc = \{5, 10, 15, 20, 25\}$ for the scale of the catchment areas and $Nc = \{5, 10, 15, 20, 25, 30, 35, 40, 45\}$ for the scale of the sub-catchments areas).

6.2.3 Features

We use 2 different types of domain features that we call Geomorphology and Vulnerability. Those features are elaborated by the domain experts. The *Geomorphology* features, deals with features about the landscape of the geographical site. It is composed of 5 different features:

— slope (%)
— elevation (m)
— area (m^2)
— distance to the stream (*i.e.,* rivers, in m)
— surface area ratio: the ratio between the surface area (taking into account the variation of the landscape) and the planar area.

The *Vulnerability* features are:

— the coastal vulnerability. The number of 75m*75m areas with an altitude inferior to 10m and a slope inferior to 2.5%.
— the hydraulic vulnerability. The number of 75m*75m areas with a distance to the stream inferior to 500m and a slope inferior to 2.5%.

All the values are the mean values for the spatial area considered as the type of case (*i.e.,* catchment or sub-catchment areas) except for the features dealing with ratio (*i.e.,* surface area ratio) and a count (*i.e.,* vulnerabilities).

In our experiments, we either use the *Geomorphology* features or the combination of the *Geomorphology* and the *Vulnerability* features to elaborate the predictive model.

6.3 Evaluation

In this section, we assess our ability to automatically and predict the optimal value of the aggregation parameter for new scenario with the Modflow model without running any simulation. The goal is to answer the following research questions:

RQ1: Can we apply the optimisation approach to automatically predict the optimal value of the aggregation parameter to perform loop aggregation on a new scenario without running the reference simulation?

RQ2: What are the factors of elaboration we can act on to ensure that we obtain a valid predictive model?

In all, we want to investigate the ability to learn from previously completed simulations and the impact of different factors on that ability.

6.3.1 Definition of the Validation Metric

As stated in Section 6.1.4, the predictive model elaborated by the loop aggregation prediction approach needs to be validated. The predictive model produces predicted val-

ues of the acceptability indicator and thus, the correctness of those values needs to be evaluated with regard to the objective of the predictive model (*i.e.,* predict the optimal aggregation parameter for applying loop aggregation on a new scenario). For the predictive model that is elaborated thanks to a training dataset, its validation is performed thanks to a testing dataset which regroups the data of a specific case. That specific case C_k plays the role of a new scenario that is to be explored thanks to loop aggregation and for which we need the optimal value of the aggregation parameter. Therefore, we evaluate the capacity of the predictive model to generate correct acceptability indicator values with regard to the declaration of the optimal aggregation parameter.

A validation metric is essential to assess the model and its correctness. Quality metrics, such as the mean square error, are commonly used in the elaboration and validation of predictive models through the data-driven approach. They evaluate the discrepancy of each predicted value to the corresponding real value. In our situation, we cannot adopt the same method. Each predicted value of the acceptability indicator is not to be taken on its own but in association with the other predicted values of the acceptability indicator for the whole range of values of the aggregation parameter that are considered for one case. Indeed, it is that group of values that leads to determining the predicted optimal value of the aggregation parameter. Therefore, it is necessary to have a validation metric that takes that fact into account. Moreover, in Section 6.1.4, we presented the different types of predictions and the need for them to be considered when defining the validation metric and performing the validation of the predictive model. Thus, we define some validation metrics in consequence by focusing on the error introduced by overestimation predictions and take into account the specificity of using a range of aggregation parameters to study a single case.

In all, we elaborate three validation metrics, that we call here the error metrics, to assess the correctness of the predictive model in order to assess:

- if the predictive model leads to overestimation predictions of the aggregation parameter.
- in the situation of overestimations, the impacts on applying loop aggregation with regard to:
  - how often there is on overestimation
  - to what extent error is introduced

In our experimentation, we do not define the validation criterion (cf. Section 6.1.4) and the associated value of the accepted error (ϵ) as we require the expertise of domain experts.

However, we use different validation metrics to quantify the correctness of the definition of the optimal aggregation parameter and assess its evolution according to different factors while exploring the design sub-space.

The $Error_{case}$ metric. To assess how much discrepancy the predictive model generates by its prediction of the optimal aggregation parameter for the case in the testing dataset, we use the $Error_{case}$ metric described in Equation 6.2a. It represents the discrepancy in terms of quantity of the acceptability indicator that is introduced by the prediction of the optimal aggregation parameter (P_{max}) when the prediction is an overestimation for the testing case.

The optimal aggregation parameter is determined thanks to Equation 6.1 for both the predicted values of the acceptability indicator (p_{max}^{pred}) and the real ones (p_{max}^{real}). The corresponding real values of the acceptability indicator corresponding to both real ($H_{max}^{real}(p_{max}^{real})$) and predicted aggregation parameters ($H_{max}^{real}(p_{max}^{pred})$) are then compared to check if the prediction introduces some additional error with a overestimation ($p_{max}^{pred}(case) > p_{max}^{real}(case)$).

$$Error_{case} = H_{max}^{real}(p_{max}^{pred})_{case} - H_{max}^{real}(p_{max}^{real})_{case} \qquad if\ p_{max}^{pred}(case) > p_{max}^{real}(case) \tag{6.2a}$$

With the $Error_{case}$ metric, the predictive model elaborated with a specific set of cases forming the training dataset is evaluated with only a specific testing case. In practise, the cases composing the training dataset and the testing case are chosen randomly, and the $Error_{case}$ is computed for this only sample dataset. That situation is represented by the Figure 6.8 where the sample dataset is composed of ten cases and only one specific case (depicted in purple) among them is used as the testing case during the model validation step.

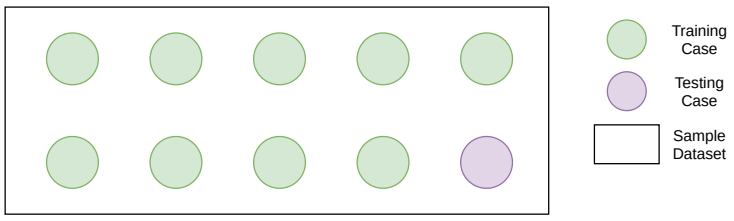


Figure 6.8 – Repartition of the cases involved in the computation of $Error_{case}$ with ten cases ($Nc = 10$)

However, the training dataset and the cases composing it can impact on the predictive

model and its predictive ability. Instead of evaluating the specific model for a specific case, we also want to evaluate its correctness independently of the cases that are part of the training dataset and the testing dataset.

The $Error_{global}$ metric. To assess the validation of the predictive model independently to the case used as the testing case, we define a global error metric ($Error_{global}$). It represents the mean error added by the predictive model, when dealing with overestimations, and corresponding to the different cases composing the datasets. In all, we iterate over for each case of the dataset for the testing case and also over the possible sample datasets (*i.e.,* combinations of cases).

To compute the $Error_{global}$ metric, we first compute the $Error_{configuration}$ metric. The latter serves to assess the error introduced by overestimations when iterating over the cases of the sample dataset to be chosen as the testing case. We call the configuration the set regrouping the instances of the sample dataset for which each case is iteratively used as the testing case. The Figure 6.9 illustrates that configuration for a sample dataset composed of ten cases. Only six of the ten cases are represented in order to save space. The principle is that the testing case, shown as a purple circle, is a different one in each instance.

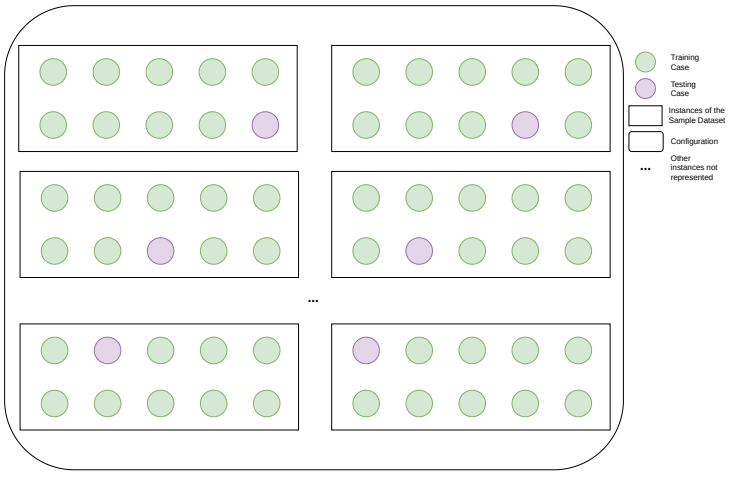


Figure 6.9 – Repartition of the cases involved in the computation of $Error_{configuration}$ with ten cases ($Nc = 10$)

The error metric of the configuration $Error_{configuration}$ is then calculated as the mean $Error_{case}$ metric over the different cases of the dataset (Nc) (cf. Equation 6.3a).

With that metric, we take into account the impact of the case of the sample dataset taken as the testing case. Yet, the sample dataset is the same in the configuration assessed with the $Error_{configuration}$ metric. Even though the cases that compose the sample dataset are randomly selected from the set of collected cases, the rest of the non-selected cases are not taken into account in the evaluation of the $Error_{configuration}$ metric.

Therefore, to have a robust evaluation and do a cross-validation, the process done for one configuration is replicated for 100 random configurations overall ($Nreplication = 100$). The random selection of the cases to elaborate the sample datasets, and by extension random configurations, is illustrated in the Figure 6.10 for an example with a size of sample dataset equals to ten cases. The global error metric ($Error_{global}$) is thus computed as the mean value of the $Error_{configuration}$ for all the configurations (cf. Equation 6.3b).

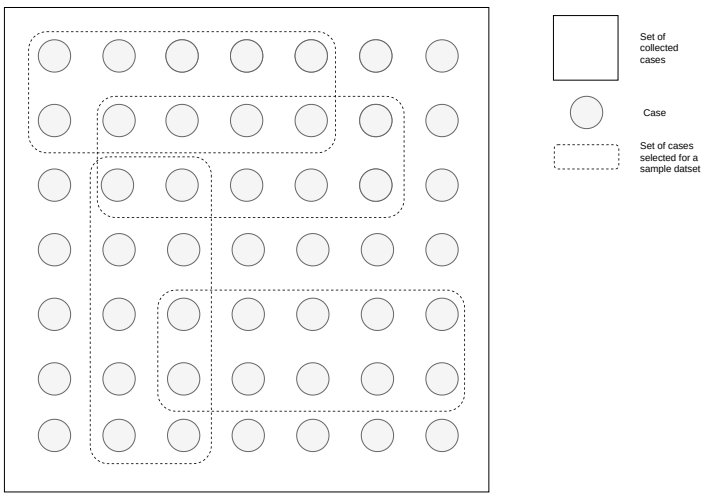


Figure 6.10 – Random selection of the cases composing the sample datasets. Example of sample datasets made of ten cases ($Nc = 10$).

$$Error_{configuration} = \frac{1}{Nc} \sum_{n=1}^{Nc} Error_{case} \tag{6.3a}$$

$$Error_{global} = \frac{1}{Nreplication} \sum_{n=1}^{Nreplication} Error_{configuration} \tag{6.3b}$$

The POG metric. To have some information about how many predictions made by the predictive model correspond to overestimations, we define the percentage of overestimations obtained globally over all the random configurations used by the $Error_{global}$ metric. We aim to quantify how often the predictive model makes overestimations. A predictive model leading to fewer overestimations is better. We call that metric *POG* for *Percentage Overestimation Global*. It is computed as the ratio of overestimation predictions over the total number of predictions and then converted into a percentage (cf. Equation 6.4).

$$Over_{case} = \begin{cases} 1 & \text{if } p_{max}^{pred} > p_{max}^{real} \\ 0 & \text{else.} \end{cases} \tag{6.4a}$$

$$POG = (\sum_{n=1}^{Nreplication*Nc} Over_{case}) * 100 \tag{6.4b}$$

Summary of the Validation Metrics

- $Error_{case}$: assesses how much the predictive model can introduce error when it lead to an overestimation of the aggregation parameter for a specific sample dataset and a specific testing case.
- $Error_{global}$: assesses how much the predictive model can introduce error when it lead to an overestimation of the aggregation parameter over 100 sample dataset and the different testing cases.
- *POG*: assesses how often the predictive model leads to overestimations.

6.3.2 Prediction of the Optimal Aggregation Parameter

In this section, we assess the quality of the ability to determine the optimal aggregation parameter by the predictive model thanks to the validation metrics we have previously presented in Section 6.3.1. In particular, we answer the research questions **RQ1** and **RQ2** defined at the beginning of Section 6.3.

RQ1: Ability to determine the optimal aggregation parameter

Depending on the training dataset used, the resulting predictive model can be validated or not with regard to the specific testing case. Indeed, there are situations where

$Error_{case} = 0$ meaning that the predictive model produces correct enough values of the acceptability indicator for the various aggregation parameter values and that the associated predicted optimal aggregation parameter (p_{max}^{pred}) is the acceptable for the considered case (*i.e.,* there is no overestimation).

The results concerning the *POG* metric attest that there are situations for which the predictive model leads to acceptable predictions (either a correct prediction of an underestimation). Actually, we can see on the figures 6.11a, 6.11b, 6.12a & 6.12b that the percentage of overestimations is globally ranging from 15% to 45% and it never equals 100%.

For instance, the predictive model is validated ($Error_{case} = 0$) for the following conditions:

— the training dataset encompasses the data concerning the catchment areas of Jullouville, La Pernelle, La Feuillie and Blainville-sur-Mer.

— the testing dataset corresponds to the data for the catchment area of Saint-Malo.

— the *Geomorphology* features

— $Np = 9$

— $Nc = 5$

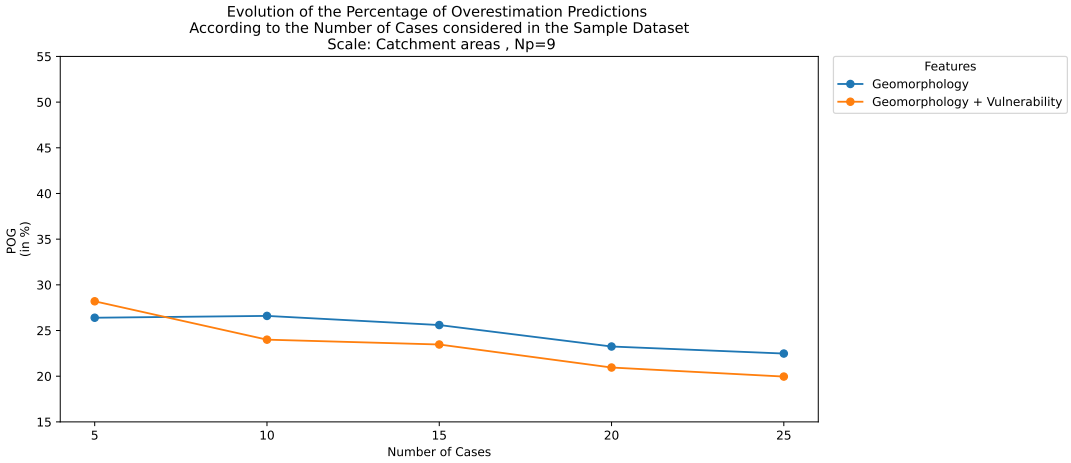
— the scale of catchment areas

However, the predictive model elaborated from the same sample dataset but with the training dataset containing the catchment areas of Jullouville, La Pernelle, La Feuillie and Saint-Malo was not validated with regard to the testing case of Blainville-sur-Mer as $Error_{case} = 0.069$. This observation highlights the impact of the composition of the training and testing datasets on the accuracy and validation of the predictive model to determine the optimal aggregation parameter of a new scenario.

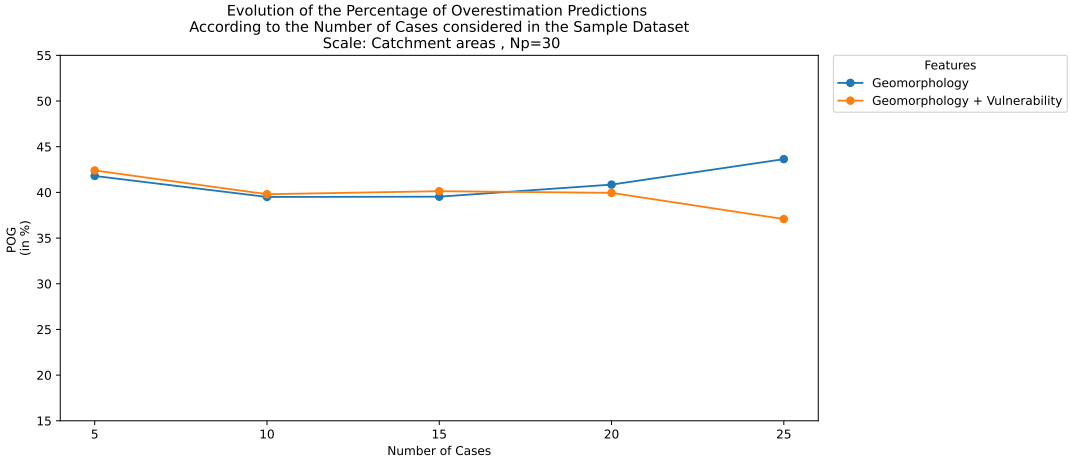
In all, to answer **RQ1**, the *Loop Aggregation Prediction* approach makes it possible to automatically and a priori determine the optimal aggregation parameter for a new scenario (*i.e.,* here a geographical site). In the sub-space of the design space that we explore, there are always acceptable predictions. The percentage of overestimations *POG* never equals 100% while investigating different values for the factors (*i.e.,* dimensions of the sub-space). But, not all predictions made by the predictive model are certain to be valid and to lead to the correct optimal aggregation parameter to be determined. The validation of the predictive model is thus also related to the various choices made along the elaboration of the predictive model and at each step of it. This result prompts us to

then look at the factors that impact on the correctness of the predictive model and to investigate the **RQ2**.

> There is an ability to learn from previously executed simulations. The *Loop Aggregation Prediction* approach makes it possible to automatically and a priori determine the optimal aggreation parameter. Yet, some predictions lead to errors while determining it. Overall, the global introduced error is low (0.05m maximum) and may not threaten the exploration of new scenarios through the application of loop aggregation on the simulaiton model. In the following section, we quantify those errors while exploring the design sub-space of the predictive model that is elaborated through the approach to investigate the impact on validation according to the different factors.

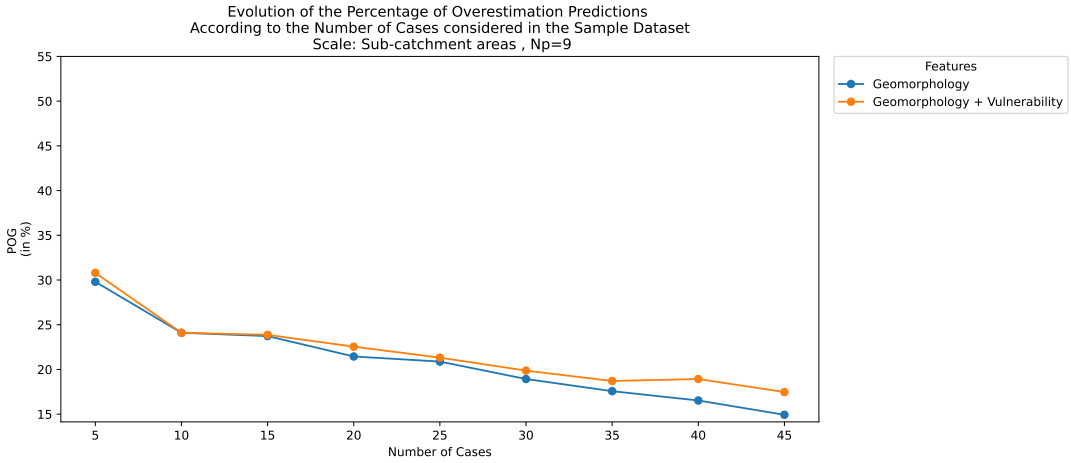


(a) Scale: catchment. $Np = 9$

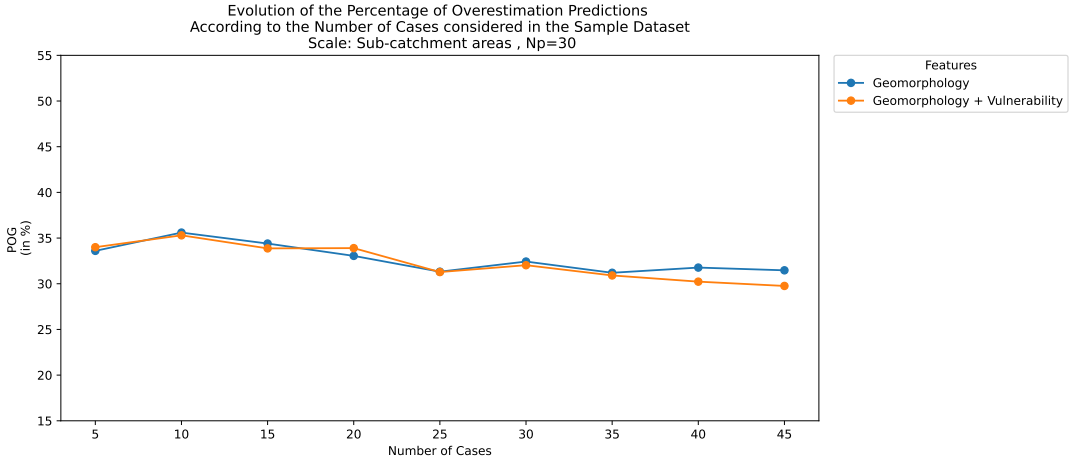


(b) Scale: catchment. $Np = 30$

Figure 6.11 – Evolution of the Percentage of overestimation predictions for Catchment areas. Np: number of aggregation parameters.



(a) Scale: sub-catchment. $Np = 9$

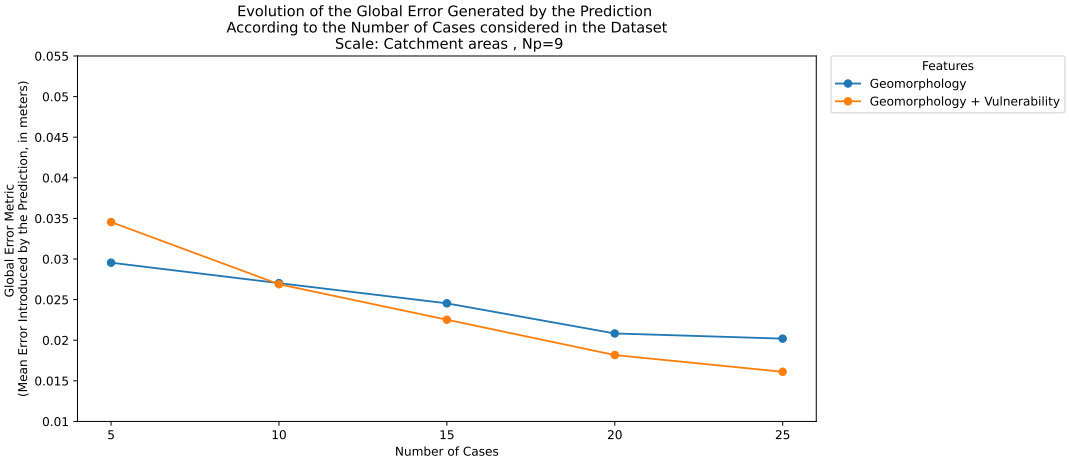


(b) Scale: sub-catchment. $Np = 30$

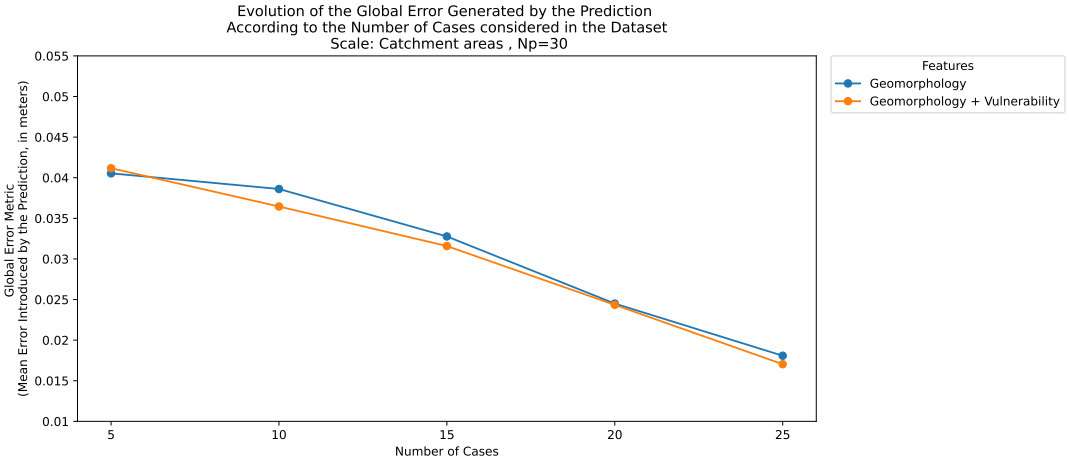
Figure 6.12 – Evolution of the Percentage of overestimation predictions prediction for Sub-catchment areas. Np: number of aggregation parameters.

RQ2: Factors Impacting the Correctness of the Predictive Model

To answer **RQ2**, we assess the correctness of the predictive model while exploring its design sub-space. We vary different factors involved in its elaboration (*i.e.,* dimensions of the sub-space): the number of aggregation parameters, the number of cases, the scale of the cases, and the features used to characterise the cases. We observe the evolution of the validation metrics according to those changes. The figures 6.13a, 6.13b, 6.14a & 6.14b illustrate the evolution of the $Error_{global}$ metric in the corresponding situations.

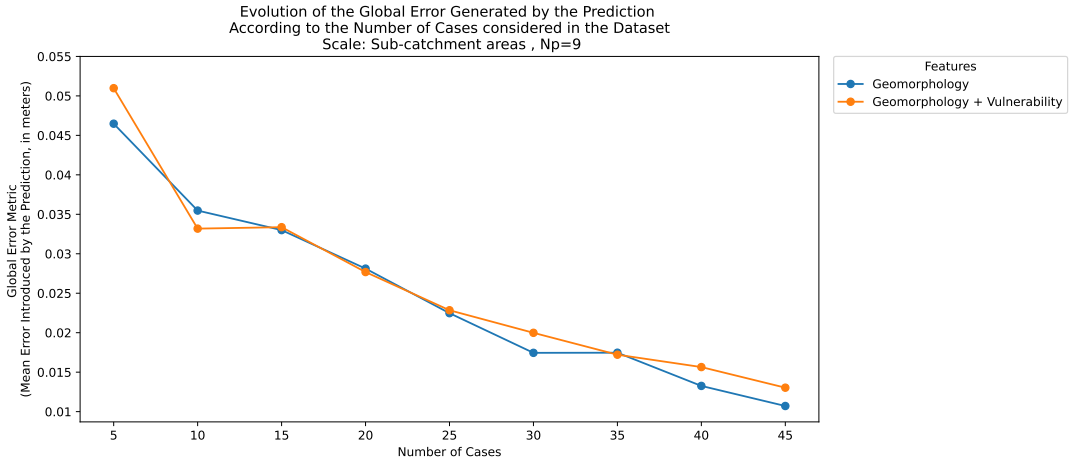


(a) Scale: catchment. $Np = 9$

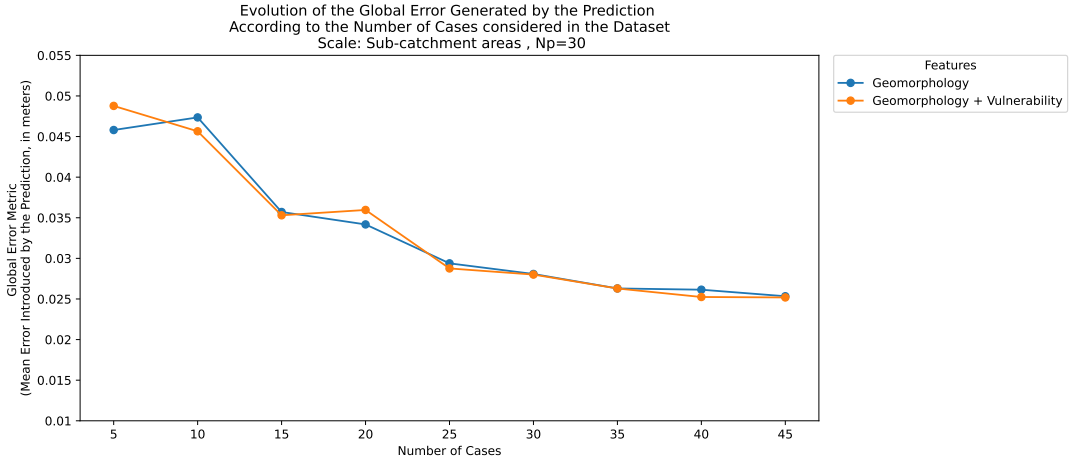


(b) Scale: catchment. $Np = 30$

Figure 6.13 – Evolution of the Global Error Metric for Catchment areas. Np: number of aggregation parameters.



(a) Scale: sub-catchment. $Np = 9$



(b) Scale: sub-catchment. $Np = 30$

Figure 6.14 – Evolution of the Global Error Metric for Sub-catchment areas. Np: number of aggregation parameters.

Number of Aggregation Parameters. We observe that the value of the quality metric is globally lower when dealing with a lower number of aggregation parameters. Using the nine possible values for the aggregation parameter (cf.Figure 6.13a & Figure 6.14a) has better prediction than $Np = 30$ (cf.Figure 6.13b & Figure 6.14b). This can be explained as there are more possible values and a higher chance to get the wrong one. The results concerning the *POG* metric (*i.e.,* percentage of overestimations) support that explanation. Indeed, the percentage of overestimations is globally lower when the number of aggregation parameters is lower.

However, the speed-up is potentially higher with more values of the aggregation parameter. For instance, for the Saint-Potan site, with nine possible values for the aggregation parameter, the real and predicted optimal aggregation parameter is 90 (cf. red values in Table 6.5), as it is the highest value for which the acceptability indicator is below the threshold of 0.1m (cf. acceptability criterion). But, with thirty possible values, the real optimal aggregation parameter is 150 and the predicted optimal aggregation parameter is 100 (cf. red values in Table 6.6). Even if the prediction is not correct in this configuration ($Error_{case} \neq 0$), the aggregation parameter is higher and by extension leads to higher speed-up when applying loop aggregation (cf. Section 5.3.2).

Site	Rate	H real	H pred
Saint-Potan	1	0.0	0.0
Saint-Potan	2	0.0026	0.0029
Saint-Potan	7	0.0104	0.0128
Saint-Potan	30	0.0275	0.0313
Saint-Potan	90	0.0670	0.0823
Saint-Potan	182	0.1215	0.1602
Saint-Potan	365	0.1426	0.1813
Saint-Potan	730	0.1854	0.2178
Saint-Potan	3652	0.2711	0.2735

Table 6.5 – H indicator values for Saint-Potan site with Geomorphology features and 9 aggregation parameters. $p_{max}^{real} = p_{max}^{pred} = 90$

With a lower number of aggregation parameters, the percentage of overestimations as well as the global error introduced by the predictive model is globally lower. Yet, the associated speed-up gained thanks to the loop aggregation application is also lower compared to a higher number of aggregation parameters.

Site	Rate	H real	H pred
Saint-Potan	1	0.0	0.0
Saint-Potan	2	0.0026	0.0030
Saint-Potan	7	0.0104	0.0136
Saint-Potan	15	0.0168	0.0212
Saint-Potan	21	0.0203	0.0239
Saint-Potan	30	0.0275	0.0322
Saint-Potan	45	0.0389	0.0455
Saint-Potan	50	0.0415	0.0480
Saint-Potan	60	0.0489	0.0588
Saint-Potan	75	0.0546	0.0663
Saint-Potan	90	0.0670	0.0823
Saint-Potan	100	0.0687	0.0856
Saint-Potan	125	0.0833	0.1062
Saint-Potan	150	0.0919	0.1176
Saint-Potan	182	0.1215	0.1602
Saint-Potan	200	0.1069	0.1404
Saint-Potan	250	0.1366	0.1599
Saint-Potan	...	...	...
Saint-Potan	3652	0.2711	0.2486

Table 6.6 – H indicator values for Saint-Potan site with Geomorphology features and 30 aggregation parameters. $p_{max}^{real} = 150$; $p_{max}^{pred} = 100$

Number of Cases. The number of cases part of the dataset has an impact on the correctness of the predictive model. Thanks to the Figure 6.13, we observe that the value of the global error metric globally decreases with a higher number of cases for both types of features when dealing with data at the scale of catchment (cf. Figure 6.13a & Figure 6.13b) and of sub-catchment areas (Figure 6.14a & Figure 6.14b). It seems logical, as having more cases in the training dataset means more information to generate more precisely the relationship function between inputs and outputs that the predictive model is built on. It is easier to infer the correct value for a new case when more cases have already been processed. There is a higher probability that the features of the new case to explore are similar to those of already treated cases.

To quantify the speed of the ability to learn according to the number of cases, we evaluate the order of magnitude of the necessary number of cases that would lead to a null global error, so leading to no overestimation. To do so, we interpolate the number of cases that would lead to a null value for the global error metric. We do this for each factor (*i.e.,* number of aggregation parameters, scale of the cases, type of features). We perform the interpolation by using a linear function. We use that type of function because of the linear aspect of the curve taken by the values of the percentage of overestimations (cf. Figure 6.11 & Figure 6.12) and by the values of the global error metric according to the number of cases (cf. Figure 6.13 & Figure 6.14). Moreover, when using a second order polynomial interpolation, there are no real solution(*i.e.,* $\nexists\ S \mid S \in \mathbb{R}$) for $Error_{global} = 0$. The estimated numbers of cases needed to reach a null value of the global error metric are presented in Table 6.7.

Number of aggregation parameters	**Scale of Cases**	**Features**	**Function**	**Estimated number of cases (order of magnitude)**
9	Catchment	Geomorphology	-0.00049799 x + 0.03190079	65
9	Catchment	Geomorphology + Vulnerability	-0.00091176 x + 0.03732907	40
9	Sub-catchment	Geomorphology	-0.00118049 x + 0.0486074	40
9	Sub-catchment	Geomorphology + Vulnerability	-0.00120761 x + 0.04823382	40
30	Catchment	Geomorphology	-0.00083812 x + 0.04588949	55
30	Catchment	Geomorphology + Vulnerability	-0.00081469 x + 0.04636059	55
30	Sub-catchment	Geomorphology	-0.00056813 x + 0.04734662	85
30	Sub-catchment	Geomorphology + Vulnerability	-0.00060514 x + 0.04837019	80

Table 6.7 – Estimated number of cases to reach a global error metric equals to zero according to different factors.

We observe that with a larger number of aggregation parameters, the number of cases would be globally larger for the predictive model to make no overestimation prediction. We also can see that the ability to learn from the past simulations is either the same

or faster for the features combining the *Geomorphology* and *Vulnerability* features. The latter may thus add some relevant information to better predict the optimal aggregation parameter.

> Using a sample dataset containing more cases leads to fewer over-estimations by the predictive model and to a lower global error introduced by it when applying loop aggregation on the simulation model. Also, the learning ability of the predictive model is enhanced when dealing with the Geomorphology and Vulnerability features.

Scale of Cases We observe that the global error metric is higher with data at the sub-catchment scale (cf.Figure 6.14a & Figure 6.14b) than at the catchment scale (cf.Figure 6.13a & Figure 6.13b) for the same number of cases. With a lower scale, the data encompass more specific information about the spatial features. We would think that it would lead to better predictions. However, it is not the case here. As the simulation of the Modflow model are run with data at the scale of the catchment area, using the same scale to predict the optimal aggregation parameter could be better.

Yet, the estimated number of cases to reach a null global error is lower for sub-catchment data (cf. Table 6.7) with nine possible values of the aggregation parameter ($Np = 9$). For instance, 42 cases would be needed for the dataset with Geomorphology features at the sub-catchment scale, whereas, 65 cases would be needed for data at the catchment scale. That means that the learning algorithm is extracting information faster from features related to sub-catchment scale. The higher slope for the curves for the sub-catchment areas supports that idea. When there are 30 possible values for the aggregation parameter, using data at the sub-catchment scale seems to require more cases (84 and 80 instead of 55 and 57). An explanation can be that the higher quantity of information present in the dataset (*i.e.,* using 30 different values of the aggregation parameter and more local information with sub-catchment areas) makes it more difficult to infer the relationship between the features and the optimal aggregation parameter.

If the goal is to have a predictive model always producing acceptable predictions with the lowest number of cases, the best situation would be to use features defined at the sub-catchment scale with the nine different values of aggregation parameters. However, if some additional constraint are to be put on reaching the best speed-up, using a dataset with features at the catchment scale with the thirty values of aggregation parameters would be best (cf. Section 6.3.2).

Features Selection Looking at the Figure 6.13 and Figure 6.14, the behaviour of the global error metric seems globally similar when dealing with *Geomorphology* or *Geomorphology + Vulnerability* features. We could have thought that using more features would result in better predictions. However, this observation suggests that adding features may not be relevant. The *Geomorphology* features describe the landscape of each site, which has a direct impact on the way groundwater behaves. The *Vulnerability* features are more focused on describing some specific areas inside the site and may be less relevant to characterise how the surrogate model will impact the results for the site.

We can say that the relevance of each feature is more important than the global number of features for the correctness of the predictive model. Moreover, using more relevant features can benefit with regard to the speed of learning from the number of cases.

Conclusion of the Experimentation

To answer **RQ1**, it is possible to elaborate a predictive model that outputs the right optimal value of the aggregation parameter for a new scenario (*i.e.,* a new geographical site). However, it is not always the case depending on the new scenario to predict and the training data used to elaborate the model. Indeed, the $Error_{global}$ metric, assessing the elaboration of the predictive models from different configurations of cases, is not equal to 0. That observation confirms that attention must be given on which dataset to use for the elaboration of the predictive model. So, the data collection step cannot be done at random and building the right training dataset has to be investigated. Overall, the error globally introduced by the predictive model is low (0.05m maximum). So, depending on the validation criterion and the value of the threshold of accepted error, the overestimations are not threatening to the exploration of new scenarios through the application of loop aggregation.
To answer **RQ2**, according to the results, to optimise the elaboration of the best predictive model, we need to define features which are related to the objective of the simulation and the functioning of the model. When the constraint is not to obtain any overestimation, it seems better to put effort into running sample simulations with a lower number of values of the aggregation parameter and more different scenarios. However, if the speed-up is more important than the accuracy, choosing to run simulations with more aggregation parameters values with less scenarios is a better method. In addition, using data at the sub-catchment scale would lead to better predictions when dealing with a larger dataset.
In all, we answer **Challenge 3** by proposing the *Loop Aggregation Prediction* approach. We show that it can automatically and a priori predict the optimal aggregation parameter for the Modflow model. However, optimising its process is required to lead to systematic correct predictions and acceptable results when applying loop aggregation on the simulation model.

6.3.3 Threats to Validity

Although we apply the optimisation approach by experimenting with different variables that impact on the correctness of the predictive model, some limitations still remain. As stated in Section 6.1.5, there are many variables in the optimisation approach that

can impact on the correctness of the predictive model. With the experimentation, we only focus on investigating some of them (*i.e.,* type of cases, number of cases, number of aggregation parameters, number and type of features) while other remain fixed (*i.e.,* learning algorithm and associated hyper-parameters). Besides, we have not explored all potential values for each factor (*e.g.,* choosing different values of the aggregation parameter but for the same number). However, this is a preliminary study that aims to give general information about the factors that impact the correctness of the predictive model (and the validation of the optimisation approach) and to what extent they impact it.

In our experiments, only the particular Modflow model presented in Section 5.2.1 is used and may not represent other scientific models. But, it can show that the different factors can significantly impact the quality of the prediction. for a model. The specific acceptability indicator defined by experts to match our context and presented in Section 5.3.1 is investigated. Other indicators may lead to other observations. Also, all the predictions are made with the same learning algorithm (*i.e.,* random forest regression algorithm). However, that means that other algorithms may lead to different predictions than the ones we observe. The associated hyper-parameters can also potentially change the observations.

Future work is then needed to be able to have a clear and global map of how the different factors impact the validation of the predictive model. It would equate to perform a sensitivity analysis of the elaboration of the predictive model.

6.3.4 Discussion

Given the experimentation findings, we conclude that the validation of the predictive model is dependent on the choices made during the different steps of the optimisation approach. There is a great variability in the quality of the predictions depending on the various factors (*e.g.,* number of cases, type of features, number of aggregation parameter to explore, etc). It seems difficult to know where to act to ensure systematic correct predictions currently, as the design space of the predictive model is large and complex. That's why, there is a need to rely on the domain knowledge to know where to look specifically and what may be relevant to improve the quality of the predictions. Indeed, the domain experts know better what physical parameters impact the results produced by the simulation model and how they may impact the value of the acceptability indicator. For instance, we have seen that the different types of features can impact the speed with which the predictive model is getting more accurate alongside the increase in the number

of cases considered in the model training. So, domain experts have more insight into what features could more relevant to optimise the trade-off made through the application of loop aggregation on the simulation model.

In summary, there are three ways to optimise the execution of a scientific model (at the software level): one based on domain knowledge with model reduction (cf. Section 3.1), another involving artificial intelligence and data (cf. Section 3.2), and the last involving computational approximation (cf. Section 3.3). The latter allows for a systematic application that is neither time nor resource intensive. The loop aggregation technique (cf. Chapter 5) can be applied as a black-box approach for an aggregation parameter equals to 2. But when a more optimal aggregation parameter is sought, an optimisation of the application of the technique is needed. The *Loop Aggregation Prediction* approach, *i.e.,* optimisation approach, however requires data and domain expertise. Therefore, the drawbacks of using the model reduction or the data-driven approaches reappear. Yet, even if the optimisation approach requires some sample simulations, it requires fewer samples than using a data-driven approach to generate a surrogate model. But optimising the optimisation approach would tend to need many more samples and be more comparable to the situation of using data-driven approach to generate surrogate models.

Moreover, when taking a step back, the loop aggregation technique is an optimisation of running the simulation model. The optimisation approach is a way to optimise the loop aggregation technique, and thus an optimisation of the optimisation of the simulation model. That optimisation approach works for some cases, but needs to be optimised to ensure correct predictions at all time. Then, that would mean doing an optimisation of the optimisation of the optimisation of the simulation model. Of course, using the optimal trade-off to explore scenarios to support decision making is the dream situation. But, we can ponder over the necessity to perform that level of optimisation when the goal is for the simulation model to run faster and to enable the exploration of scenarios with a flexibility regarding the accuracy of the results. Indeed, thanks to the experimentation we did, we know it is possible to quantify the error that is associated to the prediction of the optimal aggregation parameter. We can then let the users of the simulation model decide if they accept the potential additional approximation of using the predicted aggregation parameter for building the surrogate model that will be used to explore the different scenarios. In addition, the goal is for the stakeholders to explore a multitude of scenarios to have a global picture of the situation and problems to tackle or of the consequences of enacting the solutions scenarios. It gives them global information for the first reflective

step in the decision-making process. After getting information about the potential risks and relevant solutions to enact, the stakeholders will run the specific simulation (*i.e.,* the reference model without any approximation) of the solution they want to enact to have the precise details and risks assessment before the definite enactment of their decision. Having slightly more approximation introduced by the *Loop Aggregation prediction* approach does not seem that relevant finally.

6.4 Summary

We propose the *Loop Aggregation Prediction* approach to automatically and a priori predict the optimal aggregation parameter to use when applying loop aggregation on the simulation model for the exploration of new scenarios. We implement the optimisation approach for the Modflow model and observe that the optimal aggregation parameter can be predicted. However, its prediction is not always correct and may lead to some additional error when considering the results obtained after applying the loop aggregation technique on the simulation model. We give preliminary insight into what factors impact the correctness and to what extent. The domain knowledge would be required to understand where to look at specifically to have an efficient application of the optimisation approach (*e.g.,*. elaboration of relevant features to characterise the cases).

In all, the optimisation approach can be useful as it leads to optimal trade-off when applying loop aggregation. There is an ability to learn from previously executed simulation. The error introduced by overestimations by predictive model is relatively low and may not impact the global results of the exploration of the various scenarios by the stakeholders. With the additional error being quantified, it lets the users decide if they accept this potential additional error when exploring the various scenarios with the surrogate model through the definition of the threshold of the acceptable error in the validation criterion.

Take-away Messages of the Chapter

The *Loop Aggregation Prediction* approach enables us to automatically and in a priori fashion determine the optimal aggregation parameter to use when applying loop aggregation on the simulation model for the exploration of new scenarios. The resulting trade-off between speed-up and accuracy of the simulation model is optimal and leads to a better exploration of new scenarios by skaholders in their decision-making process. The variable efficiency of the approach according to different factors and choices made during its application highlights the significance of incorporating domain knowledge in the elaboration as well as the validation process to make it even more optimal.

Chapter 7

Conclusion and perspectives

7.1 Conclusion

Scientific models are at the heart of scientific research to understand the world around us. Scientists have long used them to represent and study real-world physical phenomena. However, a new context of use has emerged with the fact that they are now being used to predict the future behaviour of the system and to make projections. That way, scientific models are valuable in supporting decision making. The urgency surrounding global warming and environmental risks perfectly symbolises the advantage of using these models to support decision making. However, some requirements must be met to allow this shift in the context in which models are used and to make models accessible to stakeholders other than scientists and modellers (*e.g.,* policy makers, general public). The support of decision making requires the models to enable the exploration of scenarios and to be interactive, while ensuring credible projections. Yet, the models previously used by scientists have become very complex and time-consuming or resource-intensive to run. Therefore, these models need to be tailored to meet the new context. Their execution has to be sped up. The trade-off of accuracy for more flexibility of the models is relevant in this situation as decision support aims to provide an overview of the projections across scenarios or environmental risk assessment, rather than very specific data for each scenario.

We identified several challenges involved in achieving this trade-off. First, the specifics of the scientific models and associated software have to be determined (cf. **Challenge 1**, Section 1.3). Questions arise about the organisation and validation of scientific software and the comparison of scientific models with the engineering models that are usually embedded in standard software. Only once the nature, operation, and development of scientific software has been fully established, is it possible to consider how to trade off the accuracy and the execution speed of scientific models while ensuring their reliability (cf. **Challenge 2**, Section 1.3)? This involves identifying how to tailor them to the decision-making context. Finally, the concern is how to achieve this trade-off in a such

a way as to ensure an optimal exploration of scenarios which satisfies the fastest execution time of simulations while maintaining the fidelity of the models (cf. **Challenge 3**, Section 1.3).

Existing trade-off approaches entail the creation of a surrogate model which is an alternate version of the reference simulation model that answers to a different modelling objective than the one for the reference model. Those approaches adopt different strategies: model reduction, data-driven methods and approximate computing. However, none of them constitute a directly usable systematic approach to perform the trade-off between accuracy and speed for scientific models in environmental science. They are either time-consuming, resource-demanding or demand expertise, and, therefore, they do not fulfil the requirements associated with supporting stakeholders in decision making.

To achieve the trade-off in our context of interest, we addressed the aforementioned challenges through our contributions. First, we established that scientific models are complementary to engineering models. We presented the MODA framework focusing on the interplay of models and data as well as the integration of the different types and roles of models involved in a cyber-physical system. It helped highlight that, although scientific models tended to primarily have a descriptive nature, they are increasingly taking on the role of prescriptive models, providing guidelines of actions to enact in the real world (*e.g.,* which pesticide to use at which time and on which plants in the field). Engineering model embedded in standard software are commonly prescriptive models. As such, scientific models and engineering models can play the similar roles in a cyber-physical system. This endorses the idea that the methods traditionally applied to engineering models can be transferred to scientific models. Then we investigated the specificities of scientific models. We proposed an overview of the development of the associated scientific software and we highlighted the need for specific validation of the corresponding artifacts involved in the model development. Indeed, scientific software are the result of the refinement of the mathematical model and numerical scheme. They require the same V&V stage regarding the SE concerns of the implementation as the one ordinarily performed for engineering models and standard software, but, they also entail the specialised V&V stages regarding the numerical scheme and the mathematical model that are embedded into the scientific software. Finally, building on the previous contributions, we took into account the complementary nature of scientific and engineering models, and the specifics of scientific models to perform the trade-off between accuracy and execution speed on them. We investigated the application of a AC technique adapted to scientific models used in environmental

science. We thus developped a new technique, called **loop aggregation**, which can be applied in a systematic approach, that answers the requirements of supporting decision making for those specific models, and that ensures their reliability thanks to the use of an definite acceptability criterion. We validated the technique on a hydrogeological model of the underground water flow used to assess the flood risks in coastal areas.

Then we optimised the application of the loop aggregation technique by proposing an approach, called *Loop Aggregation Prediction*, that automatically and a priori predicts the optimal value of the loop aggregation parameter. The proposed approach can be used when the potential associated error is acknowledged and dealt with in the process of using the related surrogate model. The speed-up proposed by the surrogate model is then more optimal and provide a better interactivity for the decision makers to explore the various scenarios in order to decide on an action to enact.

Overall, we addressed the considered challenges by taking into account the context and the features of the scientific model to tailor it to support decision making in environmental science related issues such as climate change.

7.2 Perspectives

The contributions are innovative works in the fields of software engineering and environmental science and constitute a first step into capitalising the expertise from both fields to tackle the new challenges faced regarding the new context of use of scientific models for decision making but also in cyber-physical systems. As such, the contributions have opened up some new perspectives for future work. We identify several research questions related to the conclusion of our contributions that help give a roadmap of future work:

— Future RQ 1: How to generalise the loop aggregation technique to support decision making?

— Future RQ 2: How to use the loop aggregation technique for models at run-time?

— Future RQ 3: Can we apply loop aggregation in contexts other than supporting decision making?

— Future RQ 4: To what extent can loop aggregation be relevant in other contexts related to Environmental Science and ICT4S?

7.2.1 Future RQ1: Generalisation of the Loop Aggregation Technique

Optimisation for the Exploration of New Scenarios

As initiated by the contribution presented in Chapter 6, the application of the loop aggregation can be optimised. The optimisation approach that we propose answers that need, but is not optimal itself. Therefore, it would be beneficial to better understand all the various factors in the optimisation approach that lead to correct prediction of the optimal aggregation parameter to use for loop aggregation for generic scientific models and not only the Modflow model we use in our experimentation. Having such guidelines about the influence of the factors on the quality of prediction would help to directly implement the prediction of the optimal aggregation parameter in any context. Moreover, an interesting work would be to investigate how the incorporation of more domain knowledge could improve the approach and its application. For instance, the better fitted definition of the features used to characterise the cases involved in the model training could significantly reduce the number of overestimations we observed in our experimentation with the Modflow model. Again, the beneficial sharing of skills of the software and environmental domains can be pointed out.

Experimenting on Other Models

As stated in Section 5.3.4, the loop aggregation technique has only been validated on one environmental model. Applying the technique on other environmental models would bring more insight and information about its effect on the gain of execution speed. For example, some study could be made to determine if there is a maximal speed-up depending on the form of the mathematical models (*e.g.,* complexity of the systems of differential equations). What's more, the simulation model used in our evaluation is based on reusing the existing Modflow simulator. According to our scientific V-Model, we can think of applying loop aggregation on simulation models being implemented thanks to the description of another artifact and with another programming language. We could observe the possible adaptation it would need to implement the loop aggregation technique for a model implemented in a programming language associated with the description of the numerical scheme such as Nablab [140] (cf. Section 4.2.2). We can investigate the interaction between the discretisation testing and the validation of the surrogate model with respect to the acceptability criterion.

Integration into a Modelling Environment

The loop aggregation technique has been implemented for a hydrogeological model based on the Modflow simulator [20]. The code we wrote to do so was specific to the model. Producing a non-specific model code library would enable and facilitate the implementation of the technique for other scientific models by others. The application of the technique could thus be automated inside a modelling environment where modellers would provide the necessary information about the model such as the acceptability criterion and its threshold value, the range of approximation parameters, the aggregation function, and the interpolation function (cf. Section 5.1). We can for example think of virtual labs which provide a modelling environment for modellers to design their models, a environment to share the simulation results to other stakeholders such as policy makers and the general public [6].

7.2.2 Future RQ2: Automation of Surrogate Modelling According to the Runtime Context

As presented in Chapter 3, other surrogate modelling approaches and techniques exist. The loop aggregation technique does not replace the other statistical or model reduction approaches applied to scientific models in a standard context but rather complements them. Indeed, given its minimal set-up, it could be used during a first approximation phase to generate input/output pairs which could later be used for more efficient statistical approaches as well as allowing a first exploration of the model to better understand it for a possible model reduction approach later on.

An example of application is the approach of improving the training data of Machine Learning models with simulation results generated by process-based models that is applied in [11], [149]. Instead of building a classic predictive model thanks to the data collected about the system under study, the authors use a process-based model to generate more data that are then used in the elaboration of the predictive model. Instead of the potentially complex process-based model, the simulation results used for the training data would be generated by the surrogate model generated by applying the loop aggregation technique.

Another example of perspectives for combining the surrogate modelling approaches is the case of the service provided by the company named Extrality [1]. The service they

1. https://www.extrality.ai/

provide is to generate a surrogate model of the simulation model thanks to artificial intelligence (*i.e.,* a data-driven approach). It helps the users to run more simulations faster and to explore those simulations while keeping the accuracy of the physical process based model. The service is presented as a way to reduce the computation costs and to capitalise on all the past run simulations to improve the actual simulation model. The surrogate model that is build in this case needs executing simulations to be elaborated, as presented in the chapter on the state of the art and Section 3.2. Instead of running the reference simulation model to generate the samples used to elaborate the empirical surrogate model, the simulations would be run with the surrogate model made thanks to loop aggregation. It would save time and resource.

Therefore, there would also be a need for an approach guiding which surrogate modelling technique to use and when to use it according to the runtime context (*e.g.,* amount of available outputs that would have been already simulated, value of targeted accuracy, etc).

Another case where the loop aggregation technique can be used according to the runtime context of the model is when simulation results are critically needed in a short amount of time. For example, with an autonomous car, when an immediate response is to be made in a critical situation caused by a change in the environment (*e.g.,* an animal suddenly crossing the road), using the approximate computing approach to produce the results and infer the better action that needs to be made would be justified. The loop aggregation has thus the potential to be applied in a diversity of domains and application with regard to decision making.

7.2.3 Future RQ3: Extending the application of Loop Aggregation thanks to the generated speed-up

Loop aggregation has been elaborated to answer the needs associated to the support of decision making. However, as its functioning is based on using a faster surrogate model, we can think of investigating the other application situations where it could be beneficial. Surrogate models are already used in the exploration of the design space of scientific models, calibration and sensitivity analysis. Indeed, machine learning models as well as reduced models are used in the environmental science community to explore the design space of the models [150]. The loop aggregation technique could theoretically also be useful in those cases. That way, modellers can have more insight into the validation envelope of

their model and ensure to perform a better validation of the model.

Also, as the trade-off enables the exploration of multiple scenarios, it facilitates the modelling of systems at higher scales (from local to world systems) or to build complex systems that integrate various complex sub-systems. It can help achieve the idea of "models of everywhere" that is described by Beven [151] and that aims to use models to simulate and study the whole world thanks to a complex and global model (that is build upon previously developed models at a lower scale).

7.2.4 Future RQ4: Extending the Application of Loop Aggregation in the Field of ICT4S

Linked to the previous section, the loop aggregation technique can bring benefits in applying it in other context and in particular in ICT4S (Information and Communication Technology for Sustainability). The research field of ICT4S deals with the role that ICT plays in the digitalisation and transition to a sustainable society. Two ways that ICT can act are to (i) seek the optimisation of the energy consumption of technologies to avoid unnecessary use and (ii) to serve as a means to raise awareness about the sustainability related societal issues and help more informed-based and relevant decision making.

In this context, the loop aggregation technique we elaborated can be associated to the second way of action. Indeed, its goal is to support decision making dealing with environmental issues. However, as it is based on speeding up the execution of the simulation model by making fewer computations, there is a potential advantage in its application with regard to the energy consumption. When dealing with systems and context for which the accuracy of the simulation results is not critical, using a surrogate model generated by the loop aggregation technique could save some energy compared for instance to data-driven approaches. So, it could be interesting to assess if such technique is also relevant to be used as a means to reduce the energy consumption when dealing with scientific computing. This analysis would evaluate the footprints and the handprints generated. The reduction of energy consumption by the technique is not evident as there is the potential rebound effect of running many more faster simulations compared to fewer long simulations.

7.2.5 On the Collaboration of Environmental Science and Software Engineering Researchers

The different contributions have highlighted the benefits of collaboration of environmental science and software engineering researchers to tackle issues involving scientific models.

In particular, the MODA framework gives an overview of the different interactions of data and models according to their roles and, therefore, enables to represent the various use of models and associated contexts. The scientific V-Model presents the importance of appropriate tools to ensure the development of reliable scientific software, that are a crucial means to do scientific research. In all, to support the growing interest and need of building more complex and smarter cyber-physical systems that encompass several models, tools and methods have to be elaborated. That elaboration needs to involve all the different stakeholders with various expertise to ensure the relevance and the effectiveness of such tools and methods.

The contributions of the loop aggregation technique, as well as its associated optimisation approach, have shown the interest to use software engineering techniques usually applied for engineering models for scientific models to optimise their execution, and the need of the domain knowledge and expertise to optimise that specific optimisation. The domain knowledge is beneficial in helping the optimisation made by software engineering techniques on scientific models.

This notion of the importance of collaboration between the two domains has emerged and has gained momentum in recent years [22]. Therefore, I hope that this thesis and the associated contributions will prompt future collaborations to make cyber-physical systems more efficient, relevant and, especially, more sustainable. In particular, we can envision using CPS to address sustainability. Assessing sustainability requires modeling multiple complex systems and their associated interactions, dealing with large volumes of data, and integrate diverse domain knowledge. CPS are thus perfect means to address this task, as they allow system adaptation through feedback loops. There is also interesting investigations to lead on addressing the sustainability of the software-related processes while developing and maintaining such CPS.

In summary, we can envision smart-CPS for sustainability. CPS would integrate several scientific models which are used as simulators to describe and predict physical phenomena related to sustainability(*e.g.*,environmental, economical or social phenomena) so that pre-

scriptions about efficient adaptation of the system can be performed. The smart property of the CPS comes from the efficient adaptation that relates to the behaviour of physical phenomena, and also comes from the efficiency in adapting so that the system is sustainable with regard to its execution and maintenance (*e.g.*,using efficient (approximated) execution of models).

BIBLIOGRAPHY

[1] IPCC, *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*. M.-D. V., P. Zhai, A. Pirani, *et al.*, Eds. Cambridge University Press, 2021.

[2] United Nations, *Climate Change | United Nations*, [Online; accessed 21. June. 2021], 2021. [Online]. Available: https://www.un.org/en/global-issues/climate-change.

[3] P. Gerlee and T. Lundh, *Scientific Models*. Cham, Switzerland: Springer, 2016. DOI: 10.1007/978-3-319-27081-4.

[4] M. C. Cavalcanti, M. Mattoso, M. L. Campos, F. Llirbat, and E. Simon, « Sharing scientific models in environmental applications », *in Proceedings of the 2002 ACM Symposium on Applied Computing*, ser. SAC '02, Madrid, Spain: Association for Computing Machinery, 2002, pp. 453–457, ISBN: 1581134452. DOI: 10.1145/508791.508876. [Online]. Available: https://doi.org/10.1145/508791.508876.

[5] M. Feng, S. Zhang, and X. Gao, « Glacier runoff models sharing service and online simulation », *in 2010 Second International Conference on Advanced Geographic Information Systems, Applications, and Services*, Feb. 2010, pp. 123–126. DOI: 10.1109/GEOProcessing.2010.26.

[6] M. J. Hollaway, G. Dean, G. S. Blair, M. Brown, P. A. Henrys, and J. Watkins, « Tackling the challenges of 21st-century open science and beyond: a data science lab approach », *Patterns*, vol. 1, 7, p. 100 103, 2020, ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2020.100103. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666389920301379.

[7] W. A. Simm, F. Samreen, R. Bassett, *et al.*, « Se in es: opportunities for software engineering and cloud computing in environmental science », *in Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Society*, ser. ICSE-SEIS '18, Gothenburg, Sweden: Association for Computing Machinery, 2018, pp. 61–70, ISBN: 9781450356619. DOI: 10.1145/3183428.3183430. [Online]. Available: https://doi.org/10.1145/3183428.3183430.

[8] M. S. Mizielinski, M. J. Roberts, P. L. Vidale, *et al.*, « High-resolution global climate modelling: the upscale project, a large-simulation campaign », *Geoscientific Model Development*, vol. 7, *4*, pp. 1629–1640, 2014. DOI: 10.5194/gmd-7-1629-2014.

[9] M. J. Asher, B. F. W. Croke, A. J. Jakeman, and L. J. M. Peeters, « A review of surrogate models and their application to groundwater modeling », *Water Resources Research*, vol. 51, *8*, pp. 5957–5973, 2015. DOI: https://doi.org/10.1002/2015WR016967. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1002/2015WR016967. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2015WR016967.

[10] J. Roach and V. Tidwell, « A compartmental–spatial system dynamics approach to ground water modeling », *Groundwater*, vol. 47, *5*, pp. 686–698, 2009. DOI: https://doi.org/10.1111/j.1745-6584.2009.00580.x. eprint: https://ngwa.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1745-6584.2009.00580.x. [Online]. Available: https://ngwa.onlinelibrary.wiley.com/doi/abs/10.1111/j.1745-6584.2009.00580.x.

[11] T. Xu and F. Liang, « Machine learning for hydrologic sciences: an introductory overview », *WIREs Water*, vol. 8, *5*, e1533, 2021. DOI: https://doi.org/10.1002/wat2.1533. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wat2.1533. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wat2.1533.

[12] E. Weinan and B. Engquist, « Multiscale modeling and computation », *Notices of the AMS*, vol. 50, *9*, pp. 1062–1070, 2003.

[13] J.-M. Bruel, B. Combemale, I. Ober, and H. Raynal, « MDE in Practice for Computational Science », *in ICCS 2015*, Jun. 2015.

[14] *Data and Models at Bellairs 2019*, [Online; accessed 31. Oct. 2020], Nov. 2018. [Online]. Available: http://www.bellairs2019.ece.mcgill.ca.

[15] Z. Yu, « Hydrology, floods and droughts | modeling and prediction », *in Encyclopedia of Atmospheric Sciences (Second Edition)*, G. R. North, J. Pyle, and F. Zhang, Eds., Second Edition, Oxford: Academic Press, 2015, pp. 217–223, ISBN: 978-0-12-382225-3. DOI: https://doi.org/10.1016/B978-0-12-382225-3.00172-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123822253001729.

[16] P. Blair and W. Buytaert, « Socio-hydrological modelling: a review asking "why, what and how?" », *Hydrology and Earth System Sciences*, vol. 20, *1*, pp. 443–478, 2016. DOI: 10.5194/hess-20-443-2016. [Online]. Available: https://hess.copernicus.org/articles/20/443/2016/.

[17] T. Wagener, T. Gleeson, G. Coxon, *et al.*, « On doing hydrology with dragons: realizing the value of perceptual models and knowledge accumulation », *WIREs Water*, vol. 8, *6*, e1550, 2021. DOI: https://doi.org/10.1002/wat2.1550. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wat2.1550. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wat2.1550.

[18] IPCC, *Climate Change 2014: Synthesis Report. Contribution of Working Group I, II and III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. R. P. Core Writing Team and L. Meyer, Eds. 2014, p. 155, ISBN: 978-92-9169-143-2.

[19] D. Leroy, J. Sallou, J. Bourcier, and B. Combemale, « When scientific software meets software engineering », *Computer*, vol. 54, *12*, pp. 60–71, 2021. DOI: 10.1109/MC.2021.3102299.

[20] J. Sallou, A. Gauvain, J. Bourcier, B. Combemale, and J.-R. de Dreuzy, « Loop aggregation for approximate scientific computing », *in Computational Science – ICCS 2020*, V. V. Krzhizhanovskaya, G. Závodszky, M. H. Lees, *et al.*, Eds., Cham: Springer International Publishing, 2020, pp. 141–155, ISBN: 978-3-030-50417-5. DOI: 10.1007/978-3-030-50417-5_11.

[21] G. S. Blair, R. Bassett, L. Bastin, *et al.*, « The role of digital technologies in responding to the grand challenges of the natural environment: the windermere accord », *Patterns*, vol. 2, *1*, p. 100 156, 2021, ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2020.100156. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S266638992030204X.

[22] J. Kienzle, G. Mussbacher, B. Combemale, *et al.*, « Toward model-driven sustainability evaluation », *Commun. ACM*, vol. 63, *3*, pp. 80–91, Feb. 2020, ISSN: 0001-0782. DOI: 10.1145/3371906. [Online]. Available: https://doi.org/10.1145/3371906.

[23] B. H. Cheng, R. Lemos, H. Giese, *et al.*, « Software engineering for self-adaptive systems: a research roadmap », *in Software Engineering for Self-Adaptive Systems*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 1–26, ISBN: 9783642021602. [Online]. Available: https://doi.org/10.1007/978-3-642-02161-9_1.

[24] A. Saltelli, G. Bammer, I. Bruno, *et al.*, « Five ways to ensure that models serve society: a manifesto », *Nature*, vol. 582, pp. 482–484, Jun. 2020. DOI: 10.1038/d41586-020-01812-9.

[25] *RIVAGES 2100*, [Online; accessed 25. Aug. 2021], Jun. 2021. [Online]. Available: http://alex.gauvain.free.fr/RIVAGES.

[26] D. M. Bailer-Jones, *Scientific models in philosophy of science*. University of Pittsburgh Pre, 2009.

[27] R. Knutti and J. Sedláček, « Robustness and uncertainties in the new CMIP5 climate model projections - Nature Climate Change », *Nat. Clim. Change*, vol. 3, *4*, pp. 369–373, Apr. 2013, ISSN: 1758-6798. DOI: 10.1038/nclimate1716.

[28] L. Grcev and F. Dawalibi, « An electromagnetic model for transients in grounding systems », *IEEE Transactions on Power Delivery*, vol. 5, *4*, pp. 1773–1781, 1990. DOI: 10.1109/61.103673.

[29] J. Frank, J. Zhu, P. Penczek, *et al.*, « A model of protein synthesis based on cryo-electron microscopy of the E. coli ribosome - Nature », *Nature*, vol. 376, *6539*, pp. 441–444, Aug. 1995, ISSN: 1476-4687. DOI: 10.1038/376441a0.

[30] R. L. Chang, L. Xie, L. Xie, P. E. Bourne, and B. ø. Palsson, « Drug off-target effects predicted using structural analysis in the context of a metabolic network model », *PLOS Computational Biology*, vol. 6, *9*, pp. 1–18, Sep. 2010. DOI: 10.1371/journal.pcbi.1000938. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1000938.

[31] D. L. Olson, « Software Process Simulation », *in Encyclopedia of Information Systems*, Walthm, MA, USA: Elsevier, Jan. 2003, pp. 143–153, ISBN: 978-0-12-227240-0. DOI: 10.1016/B0-12-227240-4/00163-5.

[32] K. R. Rushton, *Groundwater hydrology: conceptual and computational models*. John Wiley & Sons, 2004.

[33] R. W. Keyes, « The Impact of Moore's Law », *IEEE Solid-State Circuits Society Newsletter*, vol. 11, *3*, pp. 25–27, Sep. 2006, ISSN: 1098-4232. DOI: 10.1109/N-SSC.2006.4785857.

[34] M. Lavrentiev, K. Lysakov, A. Romanenko, and M. Shadrin, « Modern parallel architectures to speed up numerical simulation », *in Advances in Mathematical Methods and High Performance Computing*, V. K. Singh, D. Gao, and A. Fischer, Eds. Cham: Springer International Publishing, 2019, pp. 259–269, ISBN: 978-3-030-02487-1. DOI: 10.1007/978-3-030-02487-1_15. [Online]. Available: https://doi.org/10.1007/978-3-030-02487-1_15.

[35] M. M. Waldrop, « The chips are down for Moore's law », *Nature News*, vol. 530, p. 144, Feb. 2016. DOI: 10.1038/530144a.

[36] M. M. Resch, T. Boenisch, M. Gienger, and B. Koller, « High performance computing: challenges and risks for the future », *in Advances in Mathematical Methods and High Performance Computing*, V. K. Singh, D. Gao, and A. Fischer, Eds. Cham: Springer International Publishing, 2019, pp. 249–257, ISBN: 978-3-030-02487-1. DOI: 10.1007/978-3-030-02487-1_14.

[37] R. F. Diran Basmadjian, *The Art of Modeling in Science and Engineering with Mathematica*. Andover, England, UK: Taylor & Francis, Apr. 2014, ISBN: 978-0-42917603-6. DOI: 10.1201/9781482286038.

[38] C. B. Moler, *Numerical Computing with Matlab*. Society for Industrial and Applied Mathematics, 2004. DOI: 10.1137/1.9780898717952. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898717952. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9780898717952.

[39] H. Schmidt and M. Jirstrand, « Systems Biology Toolbox for MATLAB: a computational platform for research in systems biology », *Bioinformatics*, vol. 22, *4*, pp. 514–515, Nov. 2005, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bti799. eprint: https://academic.oup.com/bioinformatics/article-pdf/22/4/514/446059/bti799.pdf. [Online]. Available: https://doi.org/10.1093/bioinformatics/bti799.

[40] J. H. Young and L. R. Stikeleather, « USE OF TK SOLVER IN AGRICULTURAL ENGINEERING INSTRUCTION », *Transactions of the ASAE*, vol. 34, *1*, pp. 301–0306, 1991. DOI: 10.13031/2013.31662.

[41] K. A. Mathews, « TK Solver 3.0 Organizes and Solves Physics Problems », *Comput. Phys.*, vol. 11, *2*, pp. 181–183, Mar. 1997, ISSN: 0894-1866. DOI: 10.1063/1.4822537.

[42] S. E. Mattsson, H. Elmqvist, and M. Otter, « Physical system modeling with Modelica », *Control Eng. Pract.*, vol. 6, *4*, pp. 501–510, Apr. 1998, ISSN: 0967-0661. DOI: 10.1016/S0967-0661(98)00047-1.

[43] P. Fritzson and V. Engelson, « Modelica — A unified object-oriented language for system modeling and simulation », *in ECOOP'98 — Object-Oriented Programming*, Berlin, Germany: Springer, May 2006, pp. 67–90, ISBN: 978-3-540-64737-9. DOI: 10.1007/BFb0054087.

[44] F. Casella and A. Leva, « Object-Oriented Modelling & Simulation of Power Plants with Modelica », *in Proceedings of the 44th IEEE Conference on Decision and Control*, IEEE, Dec. 2005, pp. 7597–7602, ISBN: 978-0-7803-9567. DOI: 10.1109/CDC.2005.1583388.

[45] C. Vecchiola, S. Pandey, and R. Buyya, « High-performance cloud computing: a view of scientific applications », *in 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, 2009, pp. 4–16. DOI: 10.1109/I-SPAN.2009.150.

[46] P. J. Turinsky, « ADVANCES IN MULTI-PHYSICS AND HIGH PERFORMANCE COMPUTING IN SUPPORT OF NUCLEAR REACTOR POWER SYSTEMS MODELING AND SIMULATION », *Nuclear Engineering and Technology*, vol. 44, *2*, pp. 103–122, 2012, ISSN: 1738-5733. DOI: 10.5516/NET.01.2012.500.

[47] V. K. Singh, D. Gao, and A. Fischer, *Advances in Mathematical Methods and High Performance Computing*. Cham, Switzerland: Springer, 2019. DOI: 10.1007/978-3-030-02487-1.

[48] J.-A. Vital, M. Gaurut, R. Lardy, *et al.*, « High-performance computing for climate change impact studies with the Pasture Simulation model », *Comput. Electron. Agric.*, vol. 98, pp. 131–135, Oct. 2013, ISSN: 0168-1699. DOI: 10.1016/j.compag.2013.08.004.

[49] V. R. Basili, J. C. Carver, D. Cruzes, *et al.*, « Understanding the high-performance-computing community: a software engineer's perspective », *IEEE Softw.*, vol. 25, *4*, pp. 29–36, Jul. 2008, ISSN: 0740-7459. DOI: 10.1109/MS.2008.103.

[50] J. C. Carver, N. P. C. Hong, and G. K. Thiruvathukal, *Software engineering for science*. CRC Press, 2016.

[51] D. T. Van der Molen and J. Pintér, « Environmental model calibration under different specifications: an application to the model sed », *Ecological Modelling*, vol. 68, *1*, pp. 1–19, 1993, Theoretical Modelling Aspects, ISSN: 0304-3800. DOI: https://doi.org/10.1016/0304-3800(93)90104-Z. [Online]. Available: https://www.sciencedirect.com/science/article/pii/030438009390104Z.

[52] H. V. Gupta, K. J. Beven, and T. Wagener, « Model calibration and uncertainty estimation », *in Encyclopedia of Hydrological Sciences*. 2006, ch. 131, ISBN: 9780470848944. DOI: https://doi.org/10.1002/0470848944.hsa138. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470848944.hsa138. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/0470848944.hsa138.

[53] N. Mizukami, O. Rakovec, A. J. Newman, *et al.*, « On the choice of calibration metrics for "high-flow" estimation using hydrologic models », *Hydrology and Earth System Sciences*, vol. 23, *6*, pp. 2601–2614, 2019. DOI: 10.5194/hess-23-2601-2019. [Online]. Available: https://hess.copernicus.org/articles/23/2601/2019/.

[54] P. Krause, D. P. Boyle, and F. Bäse, « Comparison of different efficiency criteria for hydrological model assessment », *Advances in Geosciences*, vol. 5, pp. 89–97, 2005. DOI: 10.5194/adgeo-5-89-2005. [Online]. Available: https://adgeo.copernicus.org/articles/5/89/2005/.

[55] R. Faivre, B. Iooss, S. Mahévas, D. Makowski, and H. Monod, *Analyse de sensibilité et exploration de modèles: application aux sciences de la nature et de l'environnement* (Collection Savoir-Faire). Editions Quae, 2013, 352 p.

[56] R. G. Sargent, « Verification and validation of simulation models », *in Proceedings of the 2010 Winter Simulation Conference*, 2010, pp. 166–183. DOI: 10.1109/WSC.2010.5679166.

[57] L. F. Konikow and J. D. Bredehoeft, « Ground-water models cannot be validated », *Advances in Water Resources*, vol. 15, *1*, pp. 75–83, 1992, Validation of Geo-hydrological Models Part 1, ISSN: 0309-1708. DOI: https://doi.org/10.1016/0309-1708(92)90033-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/030917089290033X.

[58] N. Oreskes, K. Shrader-Frechette, and K. Belitz, « Verification, validation, and confirmation of numerical models in the earth sciences », *Science*, vol. 263, *5147*, pp. 641–646, 1994. DOI: 10.1126/science.263.5147.641. eprint: https://www.science.org/doi/pdf/10.1126/science.263.5147.641. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.263.5147.641.

[59] K. Forsberg and H. Mooz, « The relationship of system engineering to the project cycle », *INCOSE International Symposium*, vol. 1, *1*, pp. 57–65, 1991. DOI: https://doi.org/10.1002/j.2334-5837.1991.tb01484.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.2334-5837.1991.tb01484.x. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2334-5837.1991.tb01484.x.

[60] I. Wiese, I. Polato, and G. Pinto, « Naming the pain in developing scientific software », *IEEE Software*, vol. 37, *4*, pp. 75–82, 2020. DOI: 10.1109/MS.2019.2899838.

[61] J. C. Carver, « Software engineering for computational science and engineering », *Computing in Science Engineering*, vol. 14, *2*, pp. 8–11, 2012. DOI: 10.1109/MCSE.2012.31.

[62] D. Heaton and J. C. Carver, « Claims about the use of software engineering practices in science: a systematic literature review », *Information and Software Technology*, vol. 67, pp. 207–219, 2015, ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2015.07.011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584915001342.

[63] A. Nanthaamornphong and J. C. Carver, « Test-Driven Development in scientific software: a survey », *Software Qual. J.*, vol. 25, *2*, pp. 343–372, Jun. 2017, ISSN: 1573-1367. DOI: 10.1007/s11219-015-9292-4.

[64] C. J. Roy and W. L. Oberkampf, « A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing », *Computer methods in applied mechanics and engineering*, vol. 200, *25-28*, pp. 2131–2144, 2011.

[65] B. Einarsson, *Accuracy and reliability in scientific computing*. SIAM, 2005.

[66] C. Beisbart and N. J. Saam, *Computer simulation validation: Fundamental concepts, methodological frameworks, and philosophical perspectives*. Springer, 2019.

[67] P. J. Roache, *Verification and validation in computational science and engineering*. Hermosa Albuquerque, NM, 1998, vol. 895.

[68] J. C. Helton, J. D. Johnson, W. L. Oberkampf, and C. J. Sallaberry, « Representation of analysis results involving aleatory and epistemic uncertainty », *International Journal of General Systems*, vol. 39, *6*, pp. 605–646, 2010.

[69] A. Saltelli, « Sensitivity analysis for importance assessment », *Risk analysis*, vol. 22, *3*, pp. 579–590, 2002.

[70] R. Zhang, R. Zen, J. Xing, D. M. S. Arsa, A. Saha, and S. Bressan, « Hydrological Process Surrogate Modelling and Simulation with Neural Networks », *Advances in Knowledge Discovery and Data Mining*, vol. 12085, p. 449, 2020. DOI: 10.1007/978-3-030-47436-2_34.

[71] R. Alizadeh, J. K. Allen, and F. Mistree, « Managing computational complexity using surrogate models: a critical review », *Res. Eng. Des.*, vol. 31, *3*, pp. 275–298, Jul. 2020, ISSN: 1435-6066. DOI: 10.1007/s00163-020-00336-7.

[72] S. Razavi, B. A. Tolson, and D. H. Burn, « Review of surrogate modeling in water resources », *Water Resources Research*, vol. 48, *7*, 2012. DOI: https://doi.org/10.1029/2011WR011527. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2011WR011527. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2011WR011527.

[73] B. Sudret, S. Marelli, and J. Wiart, « Surrogate models for uncertainty quantification: an overview », *in 2017 11th European Conference on Antennas and Propagation (EUCAP)*, 2017, pp. 793–797. DOI: 10.23919/EuCAP.2017.7928679.

[74] *Making the Most Out of Surrogate Models: Tricks of the Trade*, vol. Volume 1: 36th Design Automation Conference, Parts A and B, International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Aug. 2010, pp. 587–598. DOI: 10.1115/DETC2010-28813. eprint: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-pdf/IDETC-CIE2010/44090/587/2696939/587_1.pdf. [Online]. Available: https://doi.org/10.1115/DETC2010-28813.

[75] G. Barquero, J. Troya, and A. Vallecillo, « Trading accuracy for performance in data processing applications », *Journal of Object Technology*, vol. 18, *2*, B. Combemale and A. Shaukat, Eds., 9:1–24, Jul. 2019, The 15th European Conference on

Modelling Foundations and Applications, ISSN: 1660-1769. DOI: 10.5381/jot.2019.18.2.a9. [Online]. Available: http://www.jot.fm/contents/issue_2019_02/article9.html.

[76] M. N. Fienen, B. T. Nolan, L. J. Kauffman, and D. T. Feinstein, « Metamodeling for groundwater age forecasting in the lake michigan basin », *Water Resources Research*, vol. 54, *7*, pp. 4750–4766, 2018. DOI: https://doi.org/10.1029/2017WR022387. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2017WR022387. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2017WR022387.

[77] A. I. Khuri and S. Mukhopadhyay, « Response surface methodology », *WIREs Computational Statistics*, vol. 2, *2*, pp. 128–149, 2010. DOI: https://doi.org/10.1002/wics.73. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wics.73. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.73.

[78] P. Vermeulen, A. Heemink, and C. Te Stroet, « Reduced models for linear groundwater flow models using empirical orthogonal functions », *Advances in Water Resources*, vol. 27, *1*, pp. 57–69, 2004, ISSN: 0309-1708. DOI: https://doi.org/10.1016/j.advwatres.2003.09.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0309170803001490.

[79] M. Hinze and S. Volkwein, « Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: error estimates and suboptimal control », *in Dimension Reduction of Large-Scale Systems*, P. Benner, D. C. Sorensen, and V. Mehrmann, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 261–306, ISBN: 978-3-540-27909-9.

[80] J. McPhee and W. W.-G. Yeh, « Groundwater management using model reduction via empirical orthogonal functions », *Journal of Water Resources Planning and Management*, vol. 134, *2*, pp. 161–170, 2008. DOI: 10.1061/(ASCE)0733-9496(2008)134:2(161).

[81] E. H. Keating, J. Doherty, J. A. Vrugt, and Q. Kang, « Optimization and uncertainty assessment of strongly nonlinear groundwater models with high parameter dimensionality », *Water Resources Research*, vol. 46, *10*, 2010. DOI: https://doi.org/10.1029/2009WR008584. eprint: https://agupubs.onlinelibrary.

wiley.com/doi/pdf/10.1029/2009WR008584. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2009WR008584.

[82] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997, ISBN: 978-0-07-042807-2.

[83] D. Shahsavani and A. Grimvall, « Variance-based sensitivity analysis of model outputs using surrogate models », *Environmental Modelling & Software*, vol. 26, *6*, pp. 723–730, 2011, ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2011.01.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S136481521100003X.

[84] W. Burrows and J. Doherty, « Efficient calibration/uncertainty analysis using paired complex/surrogate models », *Groundwater*, vol. 53, *4*, pp. 531–541, 2015. DOI: https://doi.org/10.1111/gwat.12257. eprint: https://ngwa.onlinelibrary.wiley.com/doi/pdf/10.1111/gwat.12257. [Online]. Available: https://ngwa.onlinelibrary.wiley.com/doi/abs/10.1111/gwat.12257.

[85] J. Zhang, Q. Zheng, D. Chen, L. Wu, and L. Zeng, « Surrogate-based bayesian inverse modeling of the hydrological system: an adaptive approach considering surrogate approximation error », *Water Resources Research*, vol. 56, *1*, e2019WR025721, 2020, e2019WR025721 2019WR025721. DOI: https://doi.org/10.1029/2019WR025721. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2019WR025721. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR025721.

[86] X. Cai, R. Zeng, W. H. Kang, J. Song, and A. J. Valocchi, « Strategic planning for drought mitigation under climate change », *Journal of Water Resources Planning and Management*, vol. 141, *9*, p. 04 015 004, 2015. DOI: 10.1061/(ASCE)WR.1943-5452.0000510.

[87] F. T. Zahura, J. L. Goodall, J. M. Sadler, Y. Shen, M. M. Morsy, and M. Behl, « Training machine learning surrogate models from a high-fidelity physics-based model: application for real-time street-scale flood prediction in an urban coastal community », *Water Resources Research*, vol. 56, *10*, e2019WR027038, 2020. DOI: https://doi.org/10.1029/2019WR027038. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2019WR027038. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR027038.

[88] D. Lu and D. Ricciuto, « Efficient surrogate modeling methods for large-scale earth system models based on machine-learning techniques », *Geoscientific Model Development*, vol. 12, *5*, pp. 1791–1807, 2019. DOI: 10.5194/gmd-12-1791-2019. [Online]. Available: https://gmd.copernicus.org/articles/12/1791/2019/.

[89] M. N. Fienen, B. T. Nolan, D. T. Feinstein, and J. J. Starn, « Metamodels to bridge the gap between modeling and decision support », *Groundwater*, vol. 53, *4*, pp. 511–512, 2015. DOI: https://doi.org/10.1111/gwat.12339. eprint: https://ngwa.onlinelibrary.wiley.com/doi/pdf/10.1111/gwat.12339. [Online]. Available: https://ngwa.onlinelibrary.wiley.com/doi/abs/10.1111/gwat.12339.

[90] G. Ayzel and M. Heistermann, « The effect of calibration data length on the performance of a conceptual hydrological model versus lstm and gru: a case study for six basins from the camels dataset », *Computers & Geosciences*, vol. 149, p. 104 708, 2021, ISSN: 0098-3004. DOI: https://doi.org/10.1016/j.cageo.2021.104708. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0098300421000224.

[91] Q. Xu, T. Mytkowicz, and N. S. Kim, « Approximate computing: a survey », *IEEE Design Test*, vol. 33, *1*, pp. 8–22, 2016. DOI: 10.1109/MDAT.2015.2505723.

[92] S. Mittal, « A survey of techniques for approximate computing », *ACM Comput. Surv.*, vol. 48, *4*, Mar. 2016, ISSN: 0360-0300. DOI: 10.1145/2893356.

[93] J. Han and M. Orshansky, « Approximate computing: an emerging paradigm for energy-efficient design », *in 2013 18th IEEE European Test Symposium (ETS)*, 2013, pp. 1–6. DOI: 10.1109/ETS.2013.6569370.

[94] T. Moreau, J. San Miguel, M. Wyse, *et al.*, « A taxonomy of general purpose approximate computing techniques », *IEEE Embed. Syst. Lett.*, vol. 10, *1*, pp. 2–5, Mar. 2018, ISSN: 1943-0663. DOI: 10.1109/LES.2017.2758679. [Online]. Available: https://doi.org/10.1109/LES.2017.2758679.

[95] M. Bromberger, M. Hoffmann, and R. Rehrmann, « Do iterative solvers benefit from approximate computing? an evaluation study considering orthogonal approximation methods », *in Architecture of Computing Systems – ARCS 2018*, M. Berekovic, R. Buchty, H. Hamann, D. Koch, and T. Pionteck, Eds., Cham: Springer International Publishing, 2018, pp. 297–310, ISBN: 978-3-319-77610-1.

[96] M. Ammar Ben Khadra, « An introduction to approximate computing », *arXiv e-prints*, arXiv:1711.06115, arXiv:1711.06115, Nov. 2017. arXiv: 1711.06115 [cs.PL].

[97] H. Jiang, F. J. H. Santiago, H. Mo, L. Liu, and J. Han, « Approximate arithmetic circuits: a survey, characterization, and recent applications », *Proceedings of the IEEE*, vol. 108, *12*, pp. 2108–2135, 2020. DOI: 10.1109/JPROC.2020.3006451.

[98] S.-L. Lu, « Speeding up processing with approximation circuits », *Computer*, vol. 37, *3*, pp. 67–73, 2004. DOI: 10.1109/MC.2004.1274006.

[99] D. Mohapatra, V. K. Chippa, A. Raghunathan, and K. Roy, « Design of voltage-scalable meta-functions for approximate computing », *in 2011 Design, Automation Test in Europe*, 2011, pp. 1–6. DOI: 10.1109/DATE.2011.5763154.

[100] R. Hegde and N. Shanbhag, « Energy-efficient signal processing via algorithmic noise-tolerance », *in Proceedings. 1999 International Symposium on Low Power Electronics and Design (Cat. No.99TH8477)*, 1999, pp. 30–35. DOI: 10.1145/313817.313834.

[101] Y. Tian, Q. Zhang, T. Wang, F. Yuan, and Q. Xu, « Approxma: approximate memory access for dynamic precision scaling », *in Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*, ser. GLSVLSI '15, Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2015, pp. 337–342, ISBN: 9781450334747. DOI: 10.1145/2742060.2743759. [Online]. Available: https://doi.org/10.1145/2742060.2743759.

[102] A. Sampson, J. Nelson, K. Strauss, and L. Ceze, « Approximate storage in solid-state memories », *ACM Trans. Comput. Syst.*, vol. 32, *3*, Sep. 2014, ISSN: 0734-2071. DOI: 10.1145/2644808. [Online]. Available: https://doi.org/10.1145/2644808.

[103] C. Alvarez, J. Corbal, and M. Valero, « Fuzzy memoization for floating-point multimedia applications », *IEEE Transactions on Computers*, vol. 54, *7*, pp. 922–927, 2005. DOI: 10.1109/TC.2005.119.

[104] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, « Neural acceleration for general-purpose approximate programs », *in 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, 2012, pp. 449–460. DOI: 10.1109/MICRO.2012.48.

[105] J. Ansel, C. Chan, Y. L. Wong, *et al.*, « Petabricks: a language and compiler for algorithmic choice », *in Proceedings of the 30th ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI '09, Dublin, Ireland: Association for Computing Machinery, 2009, pp. 38–49, ISBN: 9781605583921. DOI: 10.1145/1542476.1542481. [Online]. Available: https://doi.org/10.1145/1542476.1542481.

[106] W. Baek and T. M. Chilimbi, « Green: a framework for supporting energy-conscious programming using controlled approximation », *in Proceedings of the 31st ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI '10, Toronto, Ontario, Canada: Association for Computing Machinery, 2010, pp. 198–209, ISBN: 9781450300193. DOI: 10.1145/1806596.1806620. [Online]. Available: https://doi.org/10.1145/1806596.1806620.

[107] H. Hoffmann, S. Misailovic, S. Sidiroglou, A. Agarwal, and M. Rinard, *Using Code Perforation to Improve Performance, Reduce Energy Consumption, and Respond to Failures*, Sep. 2009. [Online]. Available: https://dspace.mit.edu/handle/1721.1/46709.

[108] M. Rinard, « Probabilistic accuracy bounds for fault-tolerant computations that discard tasks », *in Proceedings of the 20th Annual International Conference on Supercomputing*, ser. ICS '06, Cairns, Queensland, Australia: Association for Computing Machinery, 2006, pp. 324–334, ISBN: 1595932828. DOI: 10.1145/1183401.1183447. [Online]. Available: https://doi.org/10.1145/1183401.1183447.

[109] I. Goiri, R. Bianchini, S. Nagarakatte, and T. D. Nguyen, « Approxhadoop: bringing approximations to mapreduce frameworks », *in Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '15, Istanbul, Turkey: Association for Computing Machinery, 2015, pp. 383–397, ISBN: 9781450328357. DOI: 10.1145/2694344.2694351. [Online]. Available: https://doi.org/10.1145/2694344.2694351.

[110] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard, « Managing performance vs. accuracy trade-offs with loop perforation », *in Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, ser. ESEC/FSE '11, Szeged, Hungary: Association for Computing Machinery, 2011, pp. 124–134, ISBN: 9781450304436. DOI: 10.1145/2025113.2025133.

[111] S. Misailovic, S. Sidiroglou, H. Hoffmann, and M. Rinard, « Quality of service profiling », *in Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ser. ICSE '10, Cape Town, South Africa: Association for Computing Machinery, 2010, pp. 25–34, ISBN: 9781605587196. DOI: 10.1145/1806799.1806808. [Online]. Available: https://doi.org/10.1145/1806799.1806808.

[112] A. Yazdanbakhsh, D. Mahajan, H. Esmaeilzadeh, and P. Lotfi-Kamran, « Axbench: a multiplatform benchmark suite for approximate computing », *IEEE Design Test*, vol. 34, *2*, pp. 60–68, 2017. DOI: 10.1109/MDAT.2016.2630270.

[113] S. Li, S. Park, and S. Mahlke, « Sculptor: flexible approximation with selective dynamic loop perforation », *in Proceedings of the 2018 International Conference on Supercomputing*, ser. ICS '18, Beijing, China: Association for Computing Machinery, 2018, pp. 341–351, ISBN: 9781450357838. DOI: 10.1145/3205289.3205317. [Online]. Available: https://doi.org/10.1145/3205289.3205317.

[114] J. Meng, S. Chakradhar, and A. Raghunathan, « Best-effort parallel execution framework for recognition and mining applications », *in 2009 IEEE International Symposium on Parallel Distributed Processing*, 2009, pp. 1–12. DOI: 10.1109/IPDPS.2009.5160991.

[115] M. Rodriguez-Cancio, B. Combemale, and B. Baudry, « Approximate loop unrolling », *in Proceedings of the 16th ACM International Conference on Computing Frontiers*, ser. CF '19, Alghero, Italy: Association for Computing Machinery, 2019, pp. 94–105, ISBN: 9781450366854. DOI: 10.1145/3310273.3323841. [Online]. Available: https://doi.org/10.1145/3310273.3323841.

[116] M. Samadi, D. A. Jamshidi, J. Lee, and S. Mahlke, « Paraprox: pattern-based approximation for data parallel applications », *in Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '14, Salt Lake City, Utah, USA: Association for Computing Machinery, 2014, pp. 35–50, ISBN: 9781450323055. DOI: 10.1145/2541940.2541948. [Online]. Available: https://doi.org/10.1145/2541940.2541948.

[117] B. Combemale, J. Kienzle, G. Mussbacher, *et al.*, « A hitchhiker's guide to model-driven engineering for data-centric systems », *IEEE Software*, vol. 38, *4*, pp. 71–84, 2020. DOI: 10.1109/MS.2020.2995125.

[118] H. Stachowiak, *Allgemeine Modelltheorie*. Springer, 1973.

[119] J. Ludewig, « Models in software engineering - an introduction », *Software & Systems Modeling*, vol. 2, pp. 5–14, 2003.

[120] E. A. Lee, « Modeling in engineering and science », *Commun. ACM*, vol. 62, *1*, pp. 35–36, Dec. 2018, ISSN: 0001-0782.

[121] T. Kühne, « Unifying explanatory and constructive modeling: towards removing the gulf between ontologies and conceptual models », *in MODELS 2016*, ACM, 2016, pp. 95–102.

[122] R. Heinrich, R. Jung, C. Zirkelbach, W. Hasselbring, and R. Reussner, « Software architecture for big data and the cloud », *in* Morgan Kaufmann, 2017, ch. An Architectural Model-Based Approach to Quality-aware DevOps in Cloud Applications, pp. 69–89.

[123] J. Liu, E. Pacitti, P. Valduriez, and M. Mattosa, « A survey of data-intensive scientific workflow management », *Grid Computing*, vol. 13, pp. 457–493, 2015.

[124] J. O. Kephart and D. M. Chess, « The vision of autonomic computing », *Computer*, vol. 36, *1*, pp. 41–50, Jan. 2003, ISSN: 1558-0814. DOI: 10.1109/MC.2003.1160055.

[125] R. De Lemos, H. Giese, H. A. Müller, *et al.*, « Software engineering for self-adaptive systems: a second research roadmap », *in Software Engineering for Self-Adaptive Systems II*, Springer, 2013, pp. 1–32.

[126] N. Bencomo, R. B. France, B. H. Cheng, and U. Aßmann, *Models@run.time: foundations, applications, and roadmaps*. Springer, 2014, vol. 8378.

[127] J. Kienzle, G. Mussbacher, B. Combemale, and J. DeAntoni, « A unifying framework for homogeneous model composition », *Software and Systems Modeling*, vol. 18, *5*, pp. 3005–3023, 2019. DOI: 10.1007/s10270-018-00707-8. [Online]. Available: https://doi.org/10.1007/s10270-018-00707-8.

[128] G. Baudart, M. Hirzel, and L. Mandel, « Deep probabilistic programming languages: A qualitative study », *CoRR*, vol. abs/1804.06458, 2018.

[129] A. Karpatne, G. Atluri, J. H. Faghmous, *et al.*, « Theory-guided data science: a new paradigm for scientific discovery from data », *IEEE Trans Knowl Data Eng*, vol. 29, *10*, pp. 2318–2331, Oct. 2017, ISSN: 1041-4347. DOI: 10.1109/TKDE.2017.2720168.

[130] B. E. Robertson, A. V. Kravtsov, N. Y. Gnedin, T. Abel, and D. H. Rudd, « Computational Eulerian hydrodynamics and Galilean invariance », *Mon. Not. R. Astron. Soc.*, vol. 401, *4*, pp. 2463–2476, Feb. 2010, ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2009.15823.x.

[131] W. L. Oberkampf, T. G. Trucano, and M. M. Pilch, « On the role of code comparisons in verification and validation. », Sandia National Laboratories, Tech. Rep., 2003.

[132] P. J. Roache, « Code verification by the method of manufactured solutions », *J. Fluids Eng.*, vol. 124, *1*, pp. 4–10, 2002.

[133] W. L. Oberkampf and C. J. Roy, *Verification and validation in scientific computing*. Cambridge University Press, 2010.

[134] W. L. Oberkampf and B. L. Smith, « Assessment criteria for computational fluid dynamics model validation experiments », *Journal of Verification, Validation and Uncertainty Quantification*, vol. 2, *3*, 2017.

[135] A. W. Harbaugh, *MODFLOW-2005, the US Geological Survey modular groundwater model: the ground-water flow process*. US Department of the Interior, US Geological Survey Reston, VA, 2005.

[136] A. Adcroft, J.-M. Campin, S. Dutkiewicz, *et al.*, « Mitgcm documentation », *Release checkpoint67a-12-gbf23121*, vol. 19, 2018.

[137] M. Bakker, V. Post, C. D. Langevin, *et al.*, « Scripting modflow model development using python and flopy », *Groundwater*, vol. 54, *5*, pp. 733–739, 2016.

[138] C. Jiang, Z. Hu, Y. Liu, Z. P. Mourelatos, D. Gorsich, and P. Jayakumar, « A sequential calibration and validation framework for model uncertainty quantification and reduction », *Computer Methods in Applied Mechanics and Engineering*, vol. 368, p. 113 172, 2020.

[139] H. Xiao, J.-L. Wu, J.-X. Wang, R. Sun, and C. Roy, « Quantifying and reducing model-form uncertainties in reynolds-averaged navier–stokes simulations: a data-driven, physics-informed bayesian approach », *Journal of Computational Physics*, vol. 324, pp. 115–136, 2016.

[140] B. Lelandais, M.-P. Oudot, and B. Combemale, « Fostering metamodels and grammars within a dedicated environment for hpc: the nablab environment (tool demo) », *in Proceedings of the 11th ACM SIGPLAN International Conference on Software Language Engineering*, ser. SLE 2018, Boston, MA, USA: ACM, 2018, pp. 200–204, ISBN: 9781450360296. DOI: 10.1145/3276604.3276620. [Online]. Available: https://doi.org/10.1145/3276604.3276620.

[141] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, « Julia: a fresh approach to numerical computing », *SIAM Review*, vol. 59, *1*, pp. 65–98, 2017. DOI: 10.1137/141000671. eprint: https://doi.org/10.1137/141000671. [Online]. Available: https://doi.org/10.1137/141000671.

[142] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, « SciPy 1.0: fundamental algorithms for scientific computing in Python », *Nat. Methods*, vol. 17, *3*, pp. 261–272, Mar. 2020, ISSN: 1548-7105. DOI: 10.1038/s41592-019-0686-2.

[143] R. Niswonger, S. Panday, and M. Ibaraki, *MODFLOW-NWT, A Newton Formulation for MODFLOW-2005: U.S. Geological Survey Techniques and Methods 6–A37*, [Online; accessed 4. Dec. 2019], 2011. [Online]. Available: https://pubs.usgs.gov/tm/tm6a37.

[144] T.-S. Lim, W.-Y. Loh, and Y.-S. Shih, « A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms », *Machine Learning*, vol. 40, *3*, pp. 203–228, Sep. 2000, ISSN: 1573-0565. DOI: 10.1023/A:1007608224229.

[145] D. R. Stockwell and A. Peterson, « Effects of sample size on accuracy of species distribution models », *Ecological Modelling*, vol. 148, *1*, pp. 1–13, 2002, ISSN: 0304-3800. DOI: https://doi.org/10.1016/S0304-3800(01)00388-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030438000100388X.

[146] L. Jollans, R. Boyle, E. Artiges, *et al.*, « Quantifying performance of machine learning methods for neuroimaging data », *NeuroImage*, vol. 199, pp. 351–365, 2019, ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2019.05.082. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1053811919304781.

[147] H. R. Pourghasemi and O. Rahmati, « Prediction of the landslide susceptibility: which algorithm, which precision? », *CATENA*, vol. 162, pp. 177–192, 2018, ISSN: 0341-8162. DOI: https://doi.org/10.1016/j.catena.2017.11.022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0341816217303909.

[148] G. Biau, « Analysis of a random forests model », *Journal of Machine Learning Research*, vol. 13, *38*, pp. 1063–1095, 2012. [Online]. Available: http://jmlr.org/papers/v13/biau12a.html.

[149] X. Jia, J. Willard, A. Karpatne, *et al.*, « Physics guided rnns for modeling dynamical systems: a case study in simulating lake temperature profiles », *in Proceedings of the 2019 SIAM International Conference on Data Mining (SDM)*, pp. 558–566. DOI: 10.1137/1.9781611975673.63. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611975673.63. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611975673.63.

[150] L. S. Matott and A. J. Rabideau, « Calibration of complex subsurface reaction models using a surrogate-model approach », *Advances in Water Resources*, vol. 31, *12*, pp. 1697–1707, 2008, ISSN: 0309-1708. DOI: https://doi.org/10.1016/j.advwatres.2008.08.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0309170808001504.

[151] K. J. BEVEN and R. E. ALCOCK, « Modelling everything everywhere: a new approach to decision-making for water management under uncertainty », *Freshwater Biology*, vol. 57, *s1*, pp. 124–132, 2012. DOI: https://doi.org/10.1111/j.1365-2427.2011.02592.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2427.2011.02592.x. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2427.2011.02592.x.

Publication List

— Benoit Combemale, Jörg Kienzle, Gunter Mussbacher, Hyacinth Ali, Daniel Amyot, Mojtaba Bagherzadeh, Edouard Batot, Nelly Bencomo, Benjamin Benni, Jean-Michel Bruel, Jordi Cabot, Betty H.C. Cheng, Philippe Collet, Gregor Engels, Robert Heinrich, Jean-Marc Jézéquel, Anne Koziolek, Sébastien Mosser, Ralf Reussner, Houari Sahraoui, Rijul Saini, **June Sallou**, Serge Stinckwich, Eugene Syriani, Manuel Wimmer. **A Hitchhiker's Guide to Model-Driven Engineering for Data-Centric Systems**. *IEEE Software*, Institute of Electrical and Electronics Engineers, 2020, pp.9. [117]

— Dorian Leroy, **June Sallou**, Johann Bourcier, Benoit Combemale. **When Scientific Software Meets Software Engineering**. *IEEE Computer*, Dec 2021. [19]

— **June Sallou**, Alexandre Gauvain, Johann Bourcier, Benoit Combemale, Jean-Raynald de Dreuzy. **Loop Aggregation for Approximate Scientific Computing**. *International Conference on Computational Science*, Jun 2020, Amsterdam, Netherlands. pp.141-155. [20]

List of Figures

UNIVERSITÉ DE RENNES 1

Titre : Agir sur la fiabilité et la flexibilité des logiciels scientifiques en sciences de l'environnement : Vers une approche systématique d'aide à la prise de décision

Mot clés : Génie Logiciel, Approximation, Prise de décision, Sciences de l'environnement

Résumé : Les logiciels scientifiques sont au cœur de l'aide à la prise de décision liée à la résolution des problèmes environnementaux grâce à la simulation. Cependant, leur complexité rend leur exécution coûteuse en temps ou en ressources, ce qui n'est pas compatible avec le contexte de la prise de décision interactive. L'objectif principal de la thèse est d'adapter les modèles scientifiques, et donc les logiciels scientifiques, afin de rendre leur utilisation pertinente et efficace dans de tels contextes. Nous étudions d'abord les modèles scientifiques et leur complémentarité avec les modèles d'ingénierie pour comprendre comment ils interagissent ensemble et comment cela a un impact sur l'adaptation souhaitée. Nous présentons le modèle MODA qui définit l'intégration des différents types et rôles que les modèles peuvent prendre dans un système sociotechnique. Nous étudions ensuite la spécificité des modèles scientifiques en termes de cycle de développement et de processus de validation. Nous décrivons une approche pour le développement de logiciels scientifiques fiables qui permet de caractériser clairement l'enveloppe de validité de ce type de logiciel. Enfin, nous proposons une approche systématique de l'adaptation des modèles scientifiques pour soutenir la prise de décision en troquant la précision contre la flexibilité. Nous adaptons une technique de calcul approximatif pour les modèles scientifiques. Nous l'évaluons sur un modèle hydro-géologique utilisé pour estimer le risque d'inondation dans les zones côtières. Nos résultats montrent une accélération significative avec une configuration requise minimale. Nous proposons également une approche d'optimisation pour généraliser l'adaptation des modèles scientifiques à la prise de décision. En somme, notre approche permet d'utiliser des modèles scientifiques pour aider à résoudre des défis environnementaux, mettant ainsi le génie logiciel au service de la société.

Title: On Reliability and Flexibility of Scientific Software in Environmental Science: Towards a Systematic Approach to Support Decision-Making

Keywords: Software Engineering, Approximate Computing, Decision Making, Environmental Science

Abstract: Scientific software are the centre stage to support decision-making related to tackling environmental issues thanks to simulation. However, their complexity makes their execution time-consuming or resource-demanding, which is not compatible to the context of interactive decision-making. The main goal of the thesis is to tailor the scientific models, and thus, scientific software, to make them relevant and efficient to be used in such contexts. We first study the scientific models and their complementarity to engineering models to understand how they interact together and how this can impact the desired tailoring. We present the MODA framework that defines the integration of the different types and roles that models can take in a sociotechnical system. We then investigate the specificity of the scientific models in terms of development cycle and validation process. We describe a reasoned approach for the development of reliable scientific software that allows to clearly characterize the validity envelope of this type of software. Finally, we propose a systematic approach of tailoring the scientific models to support decision-making by trading-off accuracy for flexibility. We adapt an approximate computing technique for scientific models. We evaluate it on a hydro-geological model used to assess the risk of flooding in coastal areas. Our results show a significant speed-up with a minimal set-up. We also propose a trade-off optimisation approach to generalise the tailoring of scientific models for decision-making. In all, our approach enables to use scientific models to help solve environmental challenges, putting software engineering at the service of society.