Centre Européen de Recherche et de Formation Avancée du Calcul scientifique
Équipe : Computational Fluid Dynamics

7 septembre 2020
Mémoire

Étude de modèles de substitution, application à la prévision immédiate de lames d'eau

Hadrien Godé

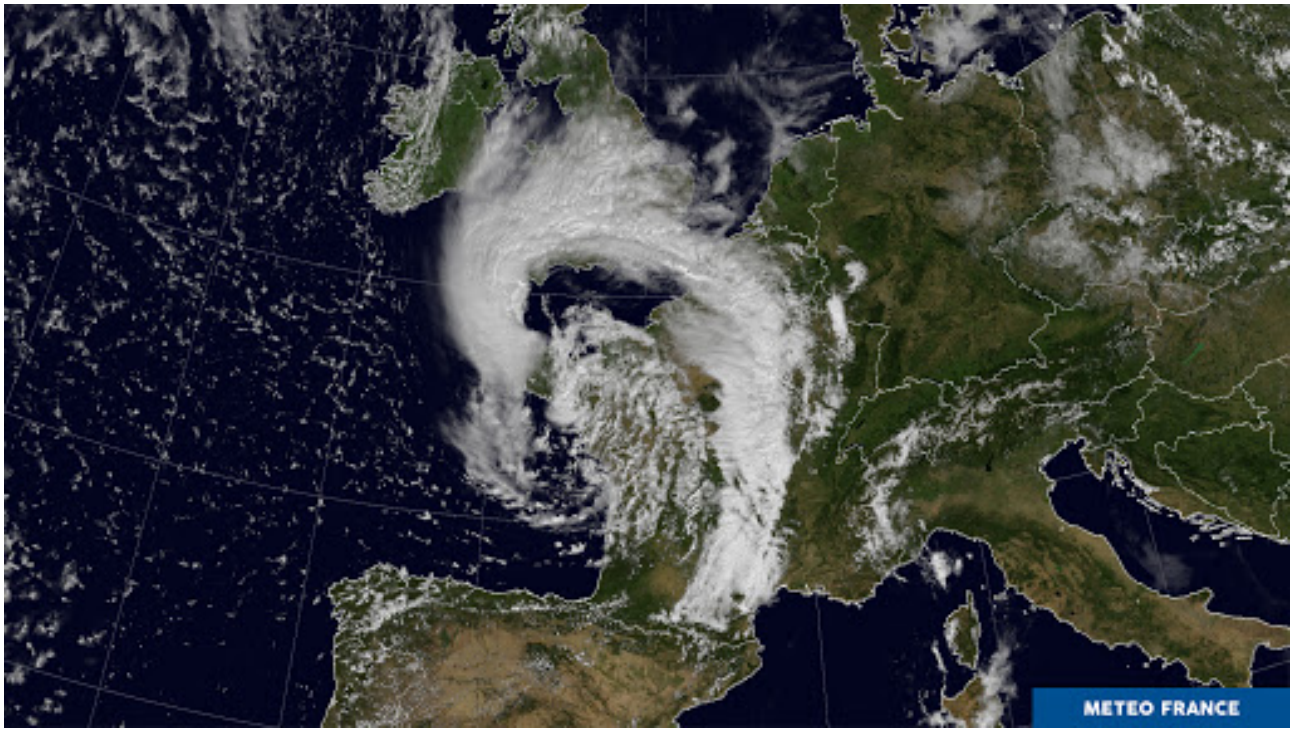


Image satellite de la tempête Miguel 07/06/2019 à 17h00

Abstract

Short term rainfall prevision is an important part of meteorological research, the goal is to accurately predict the rainfall to avoid any physical or material damage. Today, Météo-France works with solver of fluid dynamic fundamental equations to predict the rainfall : Arome-NWC. The issue is that this method has a high computational cost, and so the computation is set to save predictions every fifteen minutes for the next six hours.

So, we suggest building surrogate models to give rainfall forecasts with a higher time resolution : one minute, and a reduced calculation time compared to the time needed to run the Arome-NWC model : we want the prediction to be made before the first predicted point so the computation should take less than one minute. Our surrogate models are built by combining two different techniques : Data reduction by Proper Orthogonal Decomposition (POD), which carries out dimensionality reduction while keeping existing coherent structures, and a machine learning technique, in our case Kriging or Random Forest.

We will also investigate the possibility of using unsupervised machine learning, the goal is to create local surrogate adapted to every rainfall physical comportment and by consequences enhance the surrogate model quality. This method has been recently developed at CERFACS and is still not much published in scientific literature.

A tool specifically developed for surrogate model has been made at CERFACS : BATMAN. This tool use kriging for his ability to fit most problem and to give uncertainty quantification of the results. It will be part of the work to integrate new Machine Learning methods used for the Meteo-France research.

It will be demonstrated that a surrogate model built with POD and a variant of Random Forest : Extra Trees, is the best method tested and satisfy both the computation time criteria of one minute and the prediction quality criterion $Q^2 = 0.8$

Résumé

La prévision des lames d'eau [1] à court terme est une partie importante de la recherche météorologique, le but est de prédire avec précision les lames d'eau pour éviter tout dommage physique ou matériel. Aujourd'hui, Météo-France travaille avec des solveur d'équations fondamentales de la dynamique des fluides pour prédire les lames d'eau : Arome-NWC. Le problème est que cette méthode a un coût de calcul élevé et que le calcul est donc configuré pour enregistrer les prédictions toutes les quinze minutes pendant les six heures suivantes.

Ainsi, nous suggérons de construire des modèles de substitution pour donner des prévisions de lames d'eau avec une résolution temporelle plus élevée : une minute, et un temps de calcul réduit par rapport au temps nécessaire pour exécuter le modèle Arome-NWC : nous voulons que la prédiction soit faite avant le premier point prédit donc le calcul devrait prendre moins d'une minute. Nos modèles de substitution sont construits en combinant deux techniques différentes : la réduction des données par décomposition orthogonale appropriée (POD), qui effectue une réduction de dimensionnalité tout en conservant les structures cohérentes existantes, et une technique d'apprentissage automatique, dans notre cas le Kriging ou Random Forest.

Nous étudierons également la possibilité d'utiliser l'apprentissage automatique non supervisé, le but étant de créer un substitut local adapté à chaque comportement physique des lames d'eau et par conséquent d'améliorer la qualité du modèle de substitution. Cette méthode a été récemment développée au CERFACS et est encore peu publiée dans la littérature scientifique.

Un outil spécifiquement développé pour le modèle de substitution a été réalisé au CERFACS : BATMAN. Cet outil utilise le krigeage pour sa capacité à s'adapter à la plupart des problèmes et à donner une quantification de l'incertitude des résultats. Il s'inscrira dans le cadre des travaux d'intégration des nouvelles méthodes de Machine Learning utilisées pour la recherche Météo-France.

Il sera démontré qu'un modèle de substitution construit avec POD et une variante de Random Forest : Extra Trees, est la meilleure méthode testée et satisfait à la fois les critères de temps de calcul d'une minute et le critère de qualité de prédiction $Q^2 = 0.8$

1. Lame d'eau est le terme qui rassemble les phénomènes météorologiques suivants : précipitations et orages

Remerciements

Je tiens tout d'abord à remercier mon tuteur de stage M. Jouhaud, pour le temps qu'il m'a accordé pendant tous le stage et ses conseils toujours pertinents. De m'avoir fait confiance, pour réaliser ce stage et pour la maintenance de l'outil BATMAN.

Je remercie chaleureusement l'équipe Deep4Cast, dont les réunions mensuelles étaient toujours aussi intéressantes, pour les conseils reçus et la bonne volonté démontrée par chaque membre de l'équipe. Et particulièrement Mme Cabrera, auteure principale de l'article que nous avons co-écrit.

Je remercie l'administration qui fait un travail efficace et qui a veillé au bon déroulement du stage.

Mais aussi l'équipe CSG et particulièrement Mme. Dast dont le dévouement a toujours prouvé son efficacité lors des pires problèmes logiciels possibles.

Merci à tous les stagiaires et doctorants du CERFACS pour les pauses café et déjeuners partagés. Pour toute l'aide reçue et leur bonne humeur.

Enfin et plus généralement, à l'ensemble des personnes présentes au CERFACS qui égaient tous les jours un peu plus ce lieu.

Table des matières

Introduction

Un modèle de substitution est un modèle des résultats du processus qu'il substitue, c'est une approche statistique visant à apprendre des résultats disponibles la physique du modèle, pour être capable de donner une prédiction à un point non calculé. Notre but est donc de conserver la physique du modèle pour permettre une bonne interpolation de l'espace, tout en conservant un coût de calcul maîtrisé. Les modèles de substitution ont été utilisés dans de nombreux domaines : l'ingénierie des mines pour retrouver la position d'un gisement à partir des forages [1], l'archéologie pour reconstruire des ossements abîmés, la météorologie pour reconstruire les champs de données à partir d'observations locales.

La prévision du temps est essentiellement traitée comme un problème de mécanique des fluides. L'atmosphère est régie par l'équation de Navier-Stokes ainsi que par les équations de la thermodynamique. En schématisant, à partir d'une mesure de l'état initial, l'évolution du fluide atmosphérique est calculé pour diverses échéances, par des modèles complexes de Prévision Numérique du Temps (PNT). Ces modèles peuvent être globaux (résolution 10km) comme les modèles ARPEGE (Météo-France) ou IFS (ECMWF, Centre Européen) ou à aire limitée : modèle AROME [2] sur l'Europe de l'Ouest (résolution kilométrique).

Ces modèles ont un coût de calcul important, Météo-France doit faire des choix quant aux quantités de données calculées et stockées. Aujourd'hui, le modèle AROME stocke les prévisions de lames d'eau a un intervalle de quinze minutes pour les six prochaines heures. Augmenter la résolution temporelle serait théoriquement possible, mais le coût de calcul et l'espace de stockage nécessaire seraient bien trop importants. C'est face à cette problématique que nous cherchons à mettre en place des modèles de substitutions, le but étant de créer des prédictions avec une résolution temporelle d'une minute tout en conservant un temps de calcul supplémentaire inférieur à la minute (pour que la première prédiction arrive avant le premier point calculé).

Pour maintenir un coût de calcul faible, tout en conservant la physique du modèle nous utiliserons dans notre modèle de substitution une technique de décomposition de l'espace appelée POD [3]. Le but est de compresser les données tout en gardant les structures cohérentes de l'écoulement, nous obtiendrons un espace réduit appelé aussi espace POD. Dans l'espace POD, nous testerons divers techniques d'interpolation, pour en déduire la plus adaptée à notre problème. Ce type de modèle de substitution a prouvé son efficacité pour les études CFD.

Un outil de développement de modèle de substitution appelé BATMAN [4] a été développé au CERFACS. Il a été utilisé au CERFACS pour l'étude de système de substitution pour des problèmes, de combustion et de compréhension de phénomène naturel de mécanique des fluides. Son fonctionnement est basé sur la POD et l'interpolation par processus Gaussien (aussi nommé Kriging). Nous testerons dans ce travail de nouvelles méthodes d'interpolations, provenant de la bibliothèque d'apprentissage automatique Scikit-Learn [5] et les intégrerons à l'outil BATMAN.

Nous savons que les lames d'eau ont des comportements physiques différents en fonction de leurs conditions de formation. Par conséquent le calcul peut être biaisé par la présence de plusieurs comportements physiques de lames d'eau sur la France, chacune ayant un impact sur la prédiction de l'autre. Nous essaierons de construire des modèles de substitutions locaux qui représenteront chacun des comportements physiques. Les modèles de substitution locaux ont démontré être capables d'améliorer la qualité par rapport aux modèles de substitution globaux dans les problèmes contenant plusieurs physiques.

Nous verrons dans ce mémoire, un premier chapitre sur le contexte de travail. Puis nous allons présenter les différentes techniques qui pourraient potentiellement être utilisées :

— Dans le chapitre 2, les modèles réduits non intrusifs basés sur la POD
— Dans le chapitre 3, les modèles réduits non intrusifs locaux

Nous présenterons l'outil BATMAN et les modifications apportées dans le chapitre 4. Nous finirons par étudier la performance de ces techniques pour le cas de la prévision immédiate des lames d'eau, et donnerons une proposition de méthode à utiliser pour Météo-France dans le chapitre 5.

Chapitre 1

Le cadre d'étude

1.1 Présentation du CERFACS

Le CERFACS est un centre de recherche fondamentale et appliquée, spécialisé dans la modélisation et la simulation numérique. Grâce à ses ressources informatiques et aux chercheurs en calcul haute performance, le CERFACS traite des grands problèmes de recherches scientifique et technique d'intérêt public et industriel. Il accueille des chercheurs interdisciplinaires tels que mathématiciens appliqués, analystes numériques, ingénieurs logiciels qui conçoivent et développent des méthodes et des solutions logicielles innovantes pour répondre aux besoins des domaines aéronautique, spatial, climatique, énergétique et environnemental.

Le CERFACS est impliqué dans de grands projets nationaux et internationaux et interagit fortement avec ses sept actionnaires : Airbus Group, CNES, EDF, Météo France, ONERA, Safran et Total. Il est également associé à des partenaires comme le CNRS (Unité de Recherche Associée), l'Irit (laboratoire commun), le CEA et Inria (accords de coopération).

Le CERFACS est composé de cinq équipes :

— Aviation and environment
— Climate modeling and Global change (GLOBC)
— Computational Fluids Dynamics (CFD [1])
— Service informatique
— Parallel algorihms

Dans le cadre du stage je suis membre de l'équipe CFD. Cette équipe est la plus grande du CERFACS. Elle se concentre sur la simulation de mécanique des fluides en développant des méthodes numériques avancées appliquées aux avions, fusées, hélicoptères, moteurs de voiture, turbines, etc. L'équipe CFD est fortement liée à d'autres équipes comme GLOBC ou PAE qui utilisent également les calculs CFD pour prévoir les changements climatiques ou l'impact environnemental de l'aviation : en effet, derrière ces thèmes, se retrouvent les mêmes équations qui régissent les écoulements de fluides.

L'équipe CFD développe des outils essentiels dans de nombreux domaines applicatifs avec un leitmotiv aujourd'hui bien connu dans le monde industriel : calculons les systèmes (avions, moteurs) avant de les construire. Cela permet de réduire le coût des tests considérablement ainsi que d'éviter les erreurs de conception.

Le CFD est également un domaine où le calcul haute performance se trouve : les simulations CFD reposent sur des grilles de calcul fines avec des milliards de points nécessitant l'utilisation des supercalculateurs à millions de cœurs. La réalisation de calculs fiables et efficaces est un autre objectif principal pour lequel Cerfacs est reconnu comme un partenaire clé.

1. La mécanique des fluides numérique (MFN), plus souvent désignée par le terme anglais computational fluid dynamics (CFD), consiste à étudier les mouvements d'un fluide, ou leurs effets, par la résolution numérique des équations régissant le fluide.

1.2 La simulation numérique haute fidélité

Les écoulements rencontrés dans les problèmes de conception sont pour la plupart dominés par des mouvements chaotiques : la turbulence. L'impact de la turbulence peut être positif ou négatif, l'ingénieur doit donc être capable de prédire ces effets lors de la conception de systèmes. Malheureusement, le mouvement turbulent est très complexe et présente la plupart du temps des caractéristiques tridimensionnelles et instables. Ce mouvement consiste en la superposition de tourbillons dont le spectre de taille est très large, celui-ci balaye des grosses structures dont la taille dépend de la géométrie et correspondant à des fluctuations de basses fréquences jusqu'aux petites structures associées à des fluctuations de hautes fréquences.

La résolution de la turbulence est au cœur de l'activité de l'équipe CFD du CERFACS. Les modèles de simulation numérique, peuvent, quand utiliser correctement, représenter fidèlement la réalité, généralement ce sont des modèles à fort coût de calcul. Au CERFACS, de nombreux logiciels sont développés en interne comme AVBP [6], ou NTMIX [7], ces logiciels effectuent une résolution directe de schémas numériques basés sur les équations de Navier-Stokes [2] (DNS), ou une forme filtrée (LES [3]) ou simplifiée de ces équations (RANS [4]/ URANS).

1.3 Les moyens du CERFACS

1.3.1 Les formations continues

Le CERFACS qui travaille avec de nombreux étudiants (Master2, Doctorants, Post-Doctorants) propose des formations dans les domaines du calcul scientifique que ce soit en présentiel à Toulouse ou en ligne. Depuis 2011, le Cerfacs a mis en place un cycle de formations avancées en calcul scientifique destiné aux étudiants, doctorants, ingénieurs et chercheurs, des secteurs académiques et appliqués. Mon maître de stage en est le responsable.

Ces formations, d'une durée de 1 à 5 jours, se déroulent sur le site du Cerfacs, à Toulouse. Elles sont données par les chercheurs et ingénieurs du Cerfacs. Des formations pratiques aux logiciels développés ou co-développés par le Cerfacs (AVBP, OpenPALM, DSCLIM, CESC, OASIS, BATMAN) sont incluses dans ce programme.

Afin d'ouvrir ses formations à un public plus large et d'offrir plus de flexibilité dans la gestion de l'apprentissage, le Cerfacs propose des formations en ligne sur le modèle des SPOC. Ces formations, 100 % à distance, se déroulent sur une durée typique de 3 semaines et nécessitent environ 3 heures de travail par semaine. Elles bénéficient des dernières avancées pédagogiques en termes de formation à distance.

1.3.2 Les supercalculateurs

Trois calculateurs fournissent au Cerfacs une capacité d'environ 880 Tflop/s [5] permettant de traiter la majeure partie des besoins de simulation essentiels. A ces moyens internes s'ajoutent ceux des partenaires (Météo-France et le CCRT).

Le cluster [6] Kraken comprend 7 020 coeurs Intel SkyLake pour une puissance de 577 Tflop/s répartis dans deux partitions. Partition calcul (498 Tflop/s) : 185 noeuds de calcul dotés de 96 GO de mémoire. 2 noeuds de calcul disposant chacun de 64 coeurs and Rome à 2 Ghz et 256 GO de mémoire. Partition Pre

2. En mécanique des fluides, les équations de Navier-Stokes sont des équations aux dérivées partielles non linéaires qui décrivent le mouvement des fluides Newtoniens

3. La simulation des grandes structures de la turbulence (SGS ou en anglais LES pour Large Eddy Simulation) est une méthode utilisée en modélisation de la turbulence. Elle consiste à filtrer les petites échelles qui sont modélisées et en calculant directement les grandes échelles de la cascade turbulente.

4. Dans le cadre du traitement en mécanique des fluides de la turbulence, l'utilisation de la décomposition de Reynolds appliquée aux solutions de l'équation de Navier-Stokes permet de simplifier le problème en faisant disparaître les fluctuations de périodes et d'amplitudes courtes. La méthode est connue sous le nom de moyenne de Reynolds ou sous le terme anglais de RANS pour Reynolds-averaged Navier–Stokes.

5. Le nombre d'opérations en virgule flottante par seconde (en anglais : floating-point operations per second ou FLOPS) est une unité de mesure de la performance d'un système informatique.

6. En informatique. Un cluster est une grappe de serveurs sur un réseau, appelé ferme ou grille de calcul

et Post-Processing (79 Tflop/s) : Support aux activités de Deep Learning et IA. Visualisation et Post-traitement : 5 noeuds dotés de 288 GO de mémoire et une carte Nvidia Tesla M60. L'environnement logiciel NICE permet de prendre efficacement en charge l'affichage déporté, largement utilisé lors du confinement. Noeuds à grande mémoire : 1 noeud doté de 768 GO de mémoire destiné au traitement des maillages les plus importants + un noeud doté de 1.5 PO de mémoire. La solution a été intégrée par Lenovo et NeoTekno, elle est entrée en production au mois de mai 2018.

Le cluster Némo comprend 7 480 coeurs répartis dans trois partitions. Partition calcul (276 Tflop/s) : 288 noeuds de calcul. Partition de Pré/Post-Traitements (13 Tflop/s) : 12 noeuds de post-traitement dotés de 256 GO de mémoire DDR4 et un noeud doté de 512 GO de mémoire DDR4 destiné au traitement des maillages les plus importants. L'ensemble de ces noeud est équipé des mêmes processeurs que ceux de la partition calcul. Partition Knight Landing (11 Tflop/s crête) : constituée de 4 Noeuds Intel Knights Landing offrant chacun 64 coeurs à 1.3 Ghz avec 96 GO de mémoire et 16 GO de MCDram permet d'assurer les portages et optimisation dans cet environnement.

Le cluster Scylla (traitement de données volumineuses). Ce cluster est dédié à la gestion des données volumineuses et leur post-traitement, mais aussi à la conception d'intelligence artificielle. En particulier il a été mis en service pour post-traiter et diffuser les résultats des simulations réalisées par les chercheurs du Cerfacs dans le cadre des exercices CMIP5 et CMIP6 (Coupled Model Intercomparison Project Phase 5 et 6) dans le cadre des travaux effectués par le Cerfacs lors des deux derniers exercices du GIEC. Ce cluster est également utilisé par les autres équipes du Cerfacs disposant de problématiques similaires.

En support aux activités de recherche (support aux thèses et ANR), les ressources attribuées dans le cadre des appels à projets Genci sur les trois centres nationaux (Cines, Idris et TGCC) étendent significativement les ressources académiques. Ces dernières sont complétées par les réponses aux appels internationaux (ex. programmes Prace et Incite).

Chapitre 2

Modèles réduits non-intrusifs basés sur la Décomposition Orthogonale aux Valeurs Propres

Lors de l'étude ou la conception de nouveaux systèmes, nous avons affaire à des calculs de dimensions élevées. Les temps de calcul pour simuler ces systèmes avec un set de paramètres peuvent être très longs, il est donc souvent difficile de pouvoir parcourir la plage de données des paramètres possibles pour optimiser ou comprendre les phénomènes physiques s'appliquant à notre système. C'est pourquoi le CERFACS développe des méthodes de simulation numérique basse fidélité, ces systèmes aussi appelés modèles de substitution ou modèles réduit possèdent une forme analytique paramétrée à partir de l'interpolation ou de la régression d'échantillons « entrées-sorties » de haute fidélité.

Un outil développé par le CERFACS pour les modèles basse fidélité s'appelle BATMAN et sera présenté section 4. Pour accélérer les calculs de régression, BATMAN projette les données de calcul haute fidélité dans une base de dimension plus petite en utilisant la POD (voir section 2.1). Dans l'espace réduit, nous devons calculer les coefficients de la projection. Pour cela plusieurs méthodes existent :

- Méthode basée sur la projection, requiert l'accès aux équations gouvernantes. Mais elle donne les meilleures estimations, cependant nous ne pouvons pas l'utiliser dans le cas où nous n'avons pas accès aux codes sources.
- Méthode de minimisation des résidus, utilise une équation plus simple par exemple pour RANS, les simulations se font en minimisant les résidus des équations d'Euler.
- Méthode non intrusive, nécessite seulement l'accès aux données de sortie des simulations pour approximer les coefficients de l'espace réduit. Généralement l'approximation est faite par apprentissage automatique.

Dans le cadre de mon travail, nous nous intéresserons à la dernière méthode : non intrusive. Le fonctionnement de cette méthode sera décrit dans ce chapitre.

2.1 Décomposition orthogonale aux valeurs propres

La décomposition orthogonale aux valeurs propres [8] est une technique qui permet d'approximer un système de dimension élevée par un autre de dimension nettement plus faible, tout en gardant les structures physiques cohérentes. La première utilisation des POD est apparue en 1967 lorsque des chercheurs ont essayé d'extraire les structures cohérentes d'un écoulement turbulent par POD. La POD s'est révélée être un outil utile dans le traitement de problèmes multidimensionnels, que ce soit l'étude des structures d'un écoulement [3] [9], ou bien déduire les données manquantes des séries de données et extraire les fonctions orthogonales empiriques pertinentes [10] [11].

Si nous avons une fonction u dépendante des variables d'espace x et de temps t, qui est le résultat d'une simulation par éléments finis. Nous pouvons décomposer les résultats de la simulation dans la

base éléments finis :$\left\{\varphi^{(j)}(\boldsymbol{x})\right\}_{j=1}^{n}$:

$$u\left(\boldsymbol{x},t_i\right)=u^n\left(\boldsymbol{x},t_i\right)=\sum_{j=1}^{n}u^{(j)}\left(t_i\right)\varphi^{(j)}(\boldsymbol{x})$$

Où u^n correspond à une solution discrétisée. Nous essayons de construire la meilleure approximation de u, grâce au travaux de Holmes [3] nous savons qu'une solution est d'imposer l'orthogonalité aux fonctions de base Φ_k. Par conséquent, le problème est de trouver une famille de fonction Φ_k pour satisfaire la meilleure approximation au sens des moindres carrées :

$$\min\sum_{i=1}^{N_t}\left\|u^n\left(\boldsymbol{x},t_i\right)-\sum_{k=1}^{K}\left(u^n\left(\boldsymbol{x},t_i\right),\Phi_k(\boldsymbol{x})\right)_{\mathcal{M}}\Phi_k(\boldsymbol{x})\right\|_{\mathcal{M}}^2$$

Où $\mathcal{M}\in\mathbb{R}^{n\times n}$ est la matrice de masse. ET la base POD POD $\left\{\Phi_k(\boldsymbol{x})\right\}_{k=1}^{K}$ est supposé contenue dans l'espace vectoriel créé par les fonctions de base éléments finis $\left\{\varphi^{(j)}(\boldsymbol{x})\right\}_{j=1}^{n}$:

$$\Phi_k(x)=\sum_{i=1}^{n}\Phi_k^j\varphi^{(j)}(x)$$

Une propriété intéressante des POD est que dans un point de vue énergétique, les POD sont une décomposition optimale, c'est-à-dire qu'avec un petit nombre de modes N, nous pouvons obtenir une représentation réaliste du modèle décomposé. Si la fonction u représente une vitesse, alors la somme des amplitudes Φ_k : $\sum_{k=1}^{+\infty}\lambda_k=E$, où E est l'énergie cinétique turbulente intégrée sur le domaine considéré. Lorsque nous décomposons notre modèle en utilisant un POD, nous choisissons un critère : P_ϵ, pour que la décomposition s'arrête lorsque la dimension N_{gal} de la décomposition vérifie :

$$\frac{\sum_{n=1}^{N_{gal}}\lambda_n}{\sum_{n=1}^{N_{POD}}\lambda_n}=\frac{E\left(N_{gal}\right)}{E\left(N_{POD}\right)}\geq P_\epsilon$$

Où N_{POD} est le nombre maximum de modes possibles, cette valeur est connue en résolvant les équations de Fredholm. P_ϵ représente l'énergie conservée dans le modèle de substitution de notre décomposition. Nous travaillerons avec la valeur usuelle $P_\epsilon=0.99$, ce qui veut dire que nous conservons 99% de l'énergie du système lorsque nous utilisons une POD.

2.2 Méthode d'interpolation

BATMAN intègre de base une méthode d'interpolation par processus gaussien (aussi nommé Kriging [1]). Les processus Gaussiens ont l'atout de pouvoir estimer l'incertitude des résultats et d'être très adaptable, mais le désavantage d'avoir des coûts de calcul importants. L'utilisation de Kriging dans une base POD a déjà prouvé son efficacité [12] [13] [14].

Cependant vu que dans le cadre de nos recherches pour Météo-France, nous avons un critère de temps de calcul à respecter, nous essaierons une autre méthode prouvée plus efficace en termes de temps de calcul : Random Forest [15], et toutes ces variantes : ExtraTrees [16] et AdaBoost [17].

2.2.1 Une méthode géostatistique : Kriging

La méthode de Kriging [1] est une interpolation géostatistique, qui utilise non seulement la distance aux points voisins, mais aussi, les relations entre ces points. Pour prédire la valeur d'un point de l'espace non mesuré $\widehat{Y}$, la méthode d'interpolation utilise les mesures l'entourant Y_i et les compare : $\widehat{Y}=\sum_{i=1}^{N}\lambda_iY_i$.

Les pondérations λ_i ne s'appuient pas seulement sur la distance entre les points relevés et l'emplacement de prévision, mais aussi sur l'organisation spatiale générale des points relevés. Pour utiliser

la disposition spatiale dans la pondération, il faut quantifier l'autocorrélation spatiale. Ainsi, dans le krigeage ordinaire, la pondération λ_i dépend d'un modèle ajusté selon les points relevés, de la distance par rapport à l'emplacement de prévision et des relations spatiales entre les valeurs relevées autour de celui-ci.

Pour ce faire le krigeage procède en deux étapes :

— Il crée les variogrammes et les fonctions de covariance pour évaluer les valeurs de dépendance statistique (appelée autocorrélation spatiale), dépendant du modèle d'autocorrélation (ajustage du modèle)

— Il prédit les valeurs inconnues (formulation d'une prévision)

Un variogramme empirique [1] nous permet de voir les valeurs qui devraient être identiques du fait de leur proximité. Le semi-variogramme empirique est : $\gamma(h) = \frac{1}{2n}\sum_{i=1}^{N}(Y_i - Y_{i+h})^2$. Un modèle d'ajustement (aussi appelé noyau) est alors appliqué à ce semi-variogramme. Par conséquent, la variabilité de notre modèle est inférieure à celle de nos données. En d'autres termes Kriging lisse le gradient. Plusieurs modèles d'ajustement sont disponibles et permettent de s'adapter aux caractéristiques de nos données : stationnaire/ instationnaire et isotropique/ anistropique.

Nous pouvons décrire notre modèle comme SRN :

— Sill : correspond au maximum de γ

— Range : La zone de corrélation, si la distance est supérieure à la zone, il n'y a pas de corrélation, alors que si la distance est inférieure à la zone, les données sont autos corrélées.

— Nugget Si la distance entre les points est nulle, γ doit être nul. Mais, les erreurs de mesure créent un effet de nugget. C'est l'interception du modèle en Y.

Une fois que le modèle est créé , les coefficients λ sont calculés pour satisfaire la condition MSE. Ce qui nous donne : $\lambda_i = K^{-1}k$

K étant la matrice de covariance, $K_{ij} = C(Y_i - Y_j)$ et k étant le vecteur de covariance $k_i = C(Y_i - Y)$ avec la covariance $C(h) = C(0) - \gamma(h) = Sill - \gamma(h)$

$$\begin{pmatrix} \gamma_{11} & \dots & \gamma 1j \\ \vdots & \ddots & \vdots \\ \gamma_{i1} & \dots & \gamma_{mn} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix} = \begin{pmatrix} \gamma_{1X} \\ \vdots \\ \gamma_{nX} \end{pmatrix}$$

Nous pouvons donc exprimer Y comme $\widehat{Y} = R(S) + m(s)$. En fonction de la tendance, il existe plusieurs techniques de Kriging (ordinary kriging étant la plus utilisée) :

— Simple : la variable est stationnaire et la moyenne est connue

— Ordinary : la variable est stationnaire et la moyenne est inconnue

— Universal La variable est non stationnaire et nous avons une tendance

Ordinary kriging est la méthode la plus utilisée. Dans ce cas, la matrice de covariance est augmentée :

$$\begin{pmatrix} \gamma_{11} & \dots & \gamma 1j & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma_{i1} & \dots & \gamma_{mn} & 1 \\ 1 & \dots & 1 & 0 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ -\mu \end{pmatrix} = \begin{pmatrix} \gamma_{1X} \\ \vdots \\ \gamma_{nX} \\ 1 \end{pmatrix}$$

Une fois que les coefficients λ sont calculés, son produit avec le résidu $R_i = Y_i - m$ au point connu nous donne le résidu R(S). Nous pouvons donc estimer $\widehat{Y}$. L'erreur est estimée par le moment d'ordre 2 : $\sigma^2 = \sum_{i=1}^{N} \lambda_i \gamma_i x - \mu$

L'avantage de cette méthode est que l'interpolation est exacte sur les points de l'échantillonnage d'entraînements, et qu'elle donne une estimation de l'erreur d'interpolation. Cette méthode a été utilisée avec succès pour des problèmes d'interpolations spatio-temporelle [18] [19].

1. Le variogramme est défini pour toute fonction aléatoire intrinsèque et dépendant uniquement de l'interdistance h, alors que la fonction de covariance ne l'est que pour le cas d'une fonction aléatoire stationnaire d'ordre 2. De plus, l'estimation du variogramme n'est pas biaisée par la moyenne, au contraire de la covariance.

Les différents noyaux disponibles pour Kriging

Comme vu précédemment, plusieurs modèles d'ajustement ou noyaux peuvent être appliqués au noyau du semi-variogramme. Nous allons voir certains des noyaux disponibles dans l'outil BATMAN et leur propriété. Ainsi nous allons chercher à comprendre comment sélectionner le bon noyau en fonction des caractéristiques de nos données. Des études complètes [20] ont été réalisées pour comprendre l'influence des noyaux sur la qualité des processus gaussiens (Kriging), le but étant de créer une méthode de sélection du noyau.

Il existe deux types principaux de noyaux : les noyaux stationnaires et instationnaires, les noyaux stationnaires ne dépendent que de la distance de deux points de données et non de leurs valeurs absolues et sont donc invariants aux translations dans l'espace d'entrée, tandis que les noyaux non stationnaires dépendent également des valeurs spécifiques des points de données. Les noyaux stationnaires peuvent encore être subdivisés en noyaux isotropes et anisotropes, où les noyaux isotropes sont également invariants aux rotations dans l'espace d'entrée et les anisotropes ne le sont pas.

L'outil BATMAN permet de rentrer des noyaux composés de sommes ou de produits de noyaux de bases. Nous pouvons définir les noyaux dans un tableau 2.1, puis les classer 2.2, et enfin présenter les opérations disponibles 2.3.

Nom du noyau	Formule mathématique % propriété
Constant kernel	$k(x_1, x_2) = constant\_value \; \forall \; x_1, x_2$
Dot product kernel	$k(x_i, x_j) = \sigma_0^2 + x_i \cdot x_j$
ExpSineSquared	$k(x_i, x_j) = \exp\left(-\frac{2\sin^2(\pi d(x_i, x_j)/p)}{l^2}\right)$
Matern	$k(x_i, x_j) = \frac{1}{\Gamma(\nu)2^{\nu-1}}\left(\frac{\sqrt{2\nu}}{l} d(x_i, x_j)\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}}{l} d(x_i, x_j)\right)$
RadialBasisFunction (RBF)	$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right)$
RationalQuadratic	$k(x_i, x_j) = \left(1 + \frac{d(x_i, x_j)^2}{2\alpha l^2}\right)^{-\alpha}$
White	$k(x_1, x_2) = noise\_level$ if $x_i == x_j$ else 0

Table 2.1 – Les différents noyaux disponibles

Stationnaire		Instationnaire
isotrope	anisotrope	anisotrope
Constant White	Matern RBF	RationalQuadratic ExpSineSquared Dot product

Table 2.2 – Classification des noyaux

Nom de l'opération	Formule mathématique
CompoundKernel	Kernel which is composed of a set of other kernels.
Exponentiation	$k_{exp}(X, Y) = k(X, Y)^p$
Product	$k_{prod}(X, Y) = k_1(X, Y) * k_2(X, Y)$
Sum	$k_{sum}(X, Y) = k_1(X, Y) + k_2(X, Y)$

Table 2.3 – Les opérations disponibles sur les noyaux

Nous pouvons donc créer une infinité de noyaux différents, il est important de sélectionner un noyau qui soit adapté à notre problème, le choix du noyau détermine presque toutes les propriétés de généralisation d'un modèle.

2.2.2 Arbre et forêt en Machine Learning

Principe de l'arbre de décision

Les modèles d'arbre de décision [15] en Machine Learning, sont des arbres de décision classiques. C'est une série successive de tests conditionnels, nous pouvons imaginer un arbre de décision pour une consultation chez le médecin, voir figure 2.1

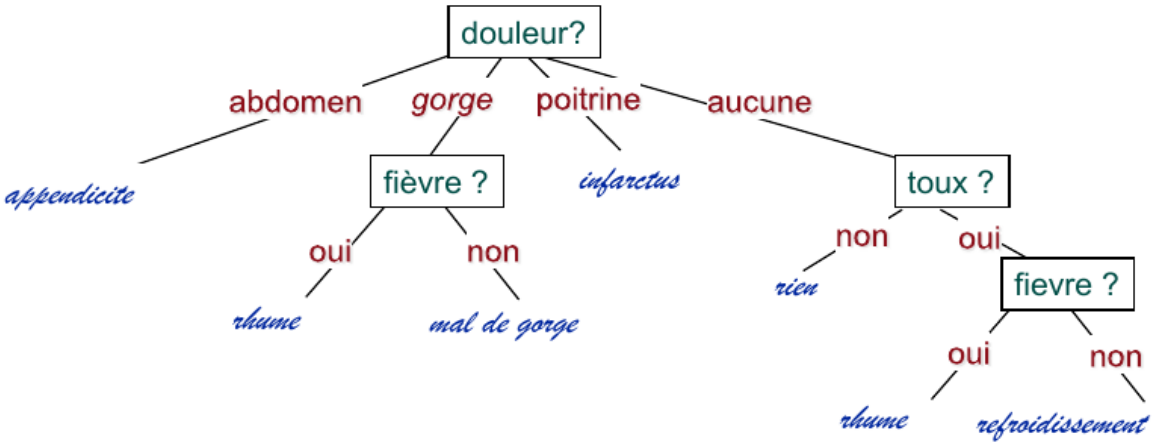


FIGURE 2.1 – Arbre décisionnel factice d'une consultation médicale

Un arbre de décision, partitionne l'espace X des observations en autant de régions qu'il a de feuilles. Au sein d'une même région, toutes les observations reçoivent alors la même étiquette. Pour prédire l'étiquette d'une observation, on suit les réponses au test depuis la racine de l'arbre, jusqu'à arriver à la feuille qui représente notre résultat. Pour un problème de régression, cette étiquette est l'étiquette moyenne des observations dans cette région :

En supposant n observations : $\overrightarrow{x}^1, \overrightarrow{x}^2, ..., \overrightarrow{x}^n$ de X étiquetées par : $y^1, y^2, ..., y^n$, et R régions : $R_1, R_2, ..., R_n$, on peut écrire :

$$f(\overrightarrow{x}) = \sum_{r=1}^{R} \delta_{\overrightarrow{x} \in R_r} \frac{1}{R_r} \sum_{i:\overrightarrow{x}^i \in R_r} y_i$$

Pour un problème de classification, cette étiquette est l'étiquette moyenne des observations dans cette région :

$$f(\vec{x}) = \sum_{r=1}^{R} \delta_{\vec{I} \in R_r} \underset{c=1,...,C}{\arg\max} \sum_{y:\vec{x}^i \in R_r} \delta\left(y^i, c\right)$$

CART : faire pousser un arbre de décision

Maintenant que nous avons vu le principe des arbres de décisions, nous allons explorer comment nous les créons à partir d'un set de données. Pour ce faire nous utilisons un algorithme appelé CART [15] : *Classification And Regression Tree*. CART partitionne les données *une variable à la fois* ce qui crée des frontières de décision orthogonales aux axes.

Les frontières créées par CART suivent des variables séparatrices $j \in 1, ..., p$, cette variable définit deux régions correspondant aux enfants du nœud considéré. Dans le cas où la variable de séparation est une valeur réelle, elle s'accompagne alors d'un point de séparation qui est la valeur de l'attribut par rapport à laquelle va se faire la décision. Par exemple si nous considérons un cas simple à deux variables de décision sur chaque axe, voir figure 2.2

A chaque itération de CART, l'algorithme cherche parmi toutes les valeurs possibles de j, ou le cas échéant parmi toutes les valeurs possibles de s pour déterminer le couple (j, s) qui minimise un critère prédéfini.

Dans le cas d'un problème de régression, ce critère est l'erreur quadratique moyenne.

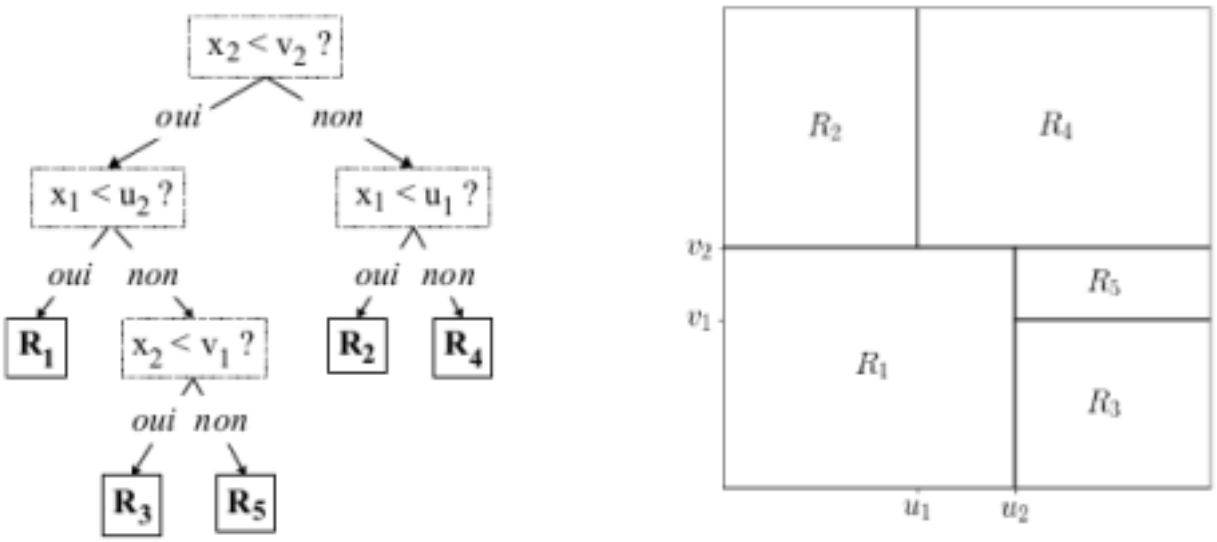


FIGURE 2.2 – Arbre de décision (à gauche) partitionne $\mathbb{R}^2$ en 5 zones (à droite)

Élaguer un arbre

Pour des soucis d'efficacité de nos algorithmes, il faut être capable de déterminer quand arrêter l'arbre de décision. Pour cela nous pouvons utiliser plusieurs méthodes :

— Nous pouvons définir un seuil minimum de point par feuille de l'arbre, ainsi nous pouvons limiter la profondeur de l'arbre

— Nous pouvons utiliser une méthode de régularisation, sachant que le coût en complexité de l'arbre est défini par le nombre de régions qu'il définit

Malheureusement, les arbres de décision ont tendance à donner des modèles trop simples et à avoir des performances de prédiction à peine supérieures à des modèles aléatoires et peu robustes aux variations dans les données. On les appelle apprenants faibles. Pour y remédier, nous devons utiliser des méthodes ensemblistes (section 2.2.2).

Forêts aléatoires

Les méthodes ensemblistes sont des méthodes très puissantes en pratique, qui reposent sur l'idée que combiner de nombreux apprenants faibles permet d'obtenir une performance largement supérieure aux performances individuelles de ces apprenants faibles, car leurs erreurs se compensent les unes les autres. Une particularité de ces méthodes est qu'à cause de leurs caractères aléatoires, nous pouvons obtenir des résultats différents sur le même calcul. Notre but sera donc de réduire leur erreur mais aussi de limiter la variance de leurs résultats.

La puissance des méthodes ensemblistes se révèle lorsque les apprenants faibles sont indépendants conditionnellement aux données, autrement dit aussi différents les uns des autres que possible, afin que leurs erreurs puissent se compenser les unes les autres. Pour atteindre cet objectif, l'idée des forêts aléatoires [15], est de construire les arbres individuels non seulement sur des échantillons différents , mais aussi en utilisant des variables différentes.

Random Forest construit donc plusieurs arbres de décision avec des frontières de décision initiale aléatoire (u_1, u_2, v_1 et v_2 sur la figure 2.3). Le partitionnement de l'espace (à partir des frontières de décision initiale aléatoire) est toujours réalisé de façon à optimiser l'erreur quadratique moyenne.Le partitionnement de l'espace est donc différent entre chaque arbre et par conséquent les sorties de chaque arbre lors d'une prédiction peuvent être différentes aussi. Pour créer une prédiction globale, l'algorithme moyenne les prédictions de chaque arbre pour un problème de régression, sinon l'algorithme fait voter chaque arbre et garde l'étiquette qui a le plus de votes pour un problème de classification.

Cette méthode est particulièrement efficace en termes de coût de calcul, de plus il est facile de gérer par le nombre d'arbres et leur profondeur le ratio temps de calcul et qualité. Pour plus d'information sur l'influence des hyperparamètres pour Random Forest le lecteur est invité à lire [21].

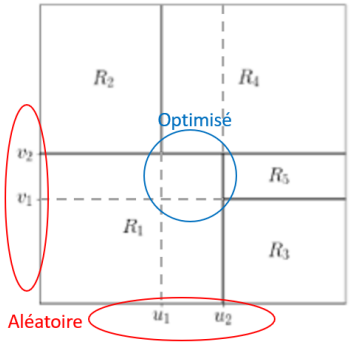


FIGURE 2.3 – Schéma partitionnement de l'espace avec Random Forest

Extra Trees

La méthode Extra Trees (EXTremely RAndomised Trees) [16], construit plusieurs arbres de décision avec des frontières de décision initiale aléatoire (u_1, u_2, v_1 et v_2 sur la figure 2.4). Le partitionnement de l'espace (à partir des frontières de décision initiale aléatoire) est cette fois-ci aussi réalisé aléatoirement d'où le nom : EXTremely RAndomised Trees. Le reste du fonctionnement de la méthode est complètement similaire à Random Forest, voir 2.2.2.

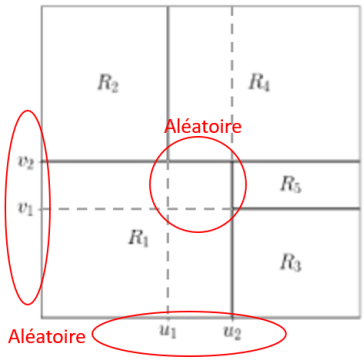


FIGURE 2.4 – Schéma partitionnement de l'espace avec Extra Trees

Ainsi le temps de calcul est encore réduit, ce qui nous permet d'ajouter plus d'arbres au modèle et ainsi réduire la variance de nos résultats. Nos observations montrent que cet algorithme de régression donne des résultats avec une variance significativement plus faible que Random Forest ou Adaboost. Le comportement de la méthode face à ses hyperparamètres et semblable à celui de Random Forest (section 2.2.2).

Adaboost

Adaboost [17] est un algorithme qui utilise les arbres de décision mais qui les force à travailler sur les erreurs du modèle (d'où le nom AdaBoost = Adaptive Boosting).

Supposons un jeu de données de classification binaire D, un nombre d'itérations M et comme algorithme d'apprentissage un arbre de décision. Étant donné un jeu de données S, on note f_S la fonction de décision retournée par cet algorithme.

Nous appelons jeu de données pondérées D' un jeu de données dans lequel un poids w_i a été affecté à la i-ème observation. Nous supposons ici que l'algorithme d'apprentissage que nous utilisons est capable d'intégrer ces pondérations. Dans le cas des arbres de décision, la pondération des exemples d'apprentissages se reflète par leur pondération dans le critère d'impureté, la décision est également prise par vote majoritaire pondéré. Autrement dit, on appelle AdaBoost la procédure de construction d'un ensemble d'apprenants suivante :

1. Initialiser les pondérations $w_1^1, w_2^1, ..., w_n^1$
2. Pour $m = 1, 2, ..., M$:

(a) Apprendre sur le jeu de données pondéré $D_m = \{(w_i^m, \overrightarrow{x_i}, y_i)\}_{i=1,...,n}$ la fonction de décision $f_m = f_{Dm}$

(b) Calculer l'erreur pondérée de ce modèle :

$$\epsilon_m = \sum_{i=1}^{m} w_i^m \delta(f_m(\overrightarrow{x}^i), y^i)$$

(c) En déduire la confiance que l'on peut lui associer :

$$\alpha_m = \frac{1}{2} \log \frac{1-\epsilon_m}{\epsilon_m}$$

α_m est d'autant plus élevé que l'erreur globale du modèle est faible : on pourra alors lui faire plus confiance.

(d) Actualiser les poids, de façon à donner plus d'importance à un exemple d'entraînement sur lequel f_m se trompe :

$$w_i^{m+1} = \frac{1}{Z_m} w_i^m \exp^{-\alpha_m y^i f_m(\overrightarrow{x}^i)} \text{ où } Z_m = \sum_{l=1}^{n} w_l^m \exp^{-\alpha_n y^i f_m(\overrightarrow{x}^i)}$$

Le rôle de Z_m est d'assurer que la somme des coefficients w_i^{m+1} vaut 1.

3. Retourner la fonction de décision finale :

$$f : \overrightarrow{x} \mapsto \sum_{m=1}^{M} \alpha_m f_m(\overrightarrow{x})$$

De nos expériences, nous avons remarqué qu'Adaboost est une méthode adaptée aux problèmes qui n'ont que très peu de samples d'entraînements, voir section 5.1.

2.3 Méthode d'optimisation

L'influence des hyperparamètres sur la qualité de l'interpolation est importante [21]. Chercher la qualité maximale de l'interpolation est un problème d'optimisation multidimensionnelle complexe. Tout d'abord, car le temps de calcul nécessaire pour l'interpolation et l'évaluation de sa qualité peut être important (donc nous évitons de couvrir tous les paramètres possibles). Mais aussi, car il existe de multiples extremums locaux. Pour obtenir les hyperparamètres adaptés aux problèmes, nous devons utiliser des méthodes d'optimisation que nous présentons dans cette section.

2.3.1 Black Box OPAL de Charles Audet

OPAL est une bibliothèque python créée par Charles Audet [22] qui utilise une méthode d'optimisation sans dérivation pour optimiser les paramètres de tout type d'algorithme, la méthode est non intrusive c'est-à-dire qu'elle ne nécessite pas de connaître le fonctionnement de l'algorithme.

Son algorithme est adapté pour la création de modèles de substitution dont le temps de calcul est lourd. A chaque calcul, OPAL calcule un nouveau set de paramètres par recherche directe.

Les algorithmes de recherche directe diffèrent les uns des autres par la façon dont ils construisent le prochain paramètre d'essai. L'une des méthodes les plus simples est la recherche de coordonnées, qui consiste à créer un second paramètre d'essai (où n est la dimension du vecteur p) dont l'espérance est d'améliorer le paramètre le plus connu actuel, dit p^{best}. Lors de cette seconde tentative les paramètres deviennent $p^{best} \pm \Delta e_i$ avec i = 1, 2, ..., n. Où e_i est le i-ème vecteur de coordonnées et $\Delta > 0$ est une taille de pas donnée, également appelée taille de maillage. Chacun de ces seconds paramètres d'essai sont déposé à son tour dans la boîte noire pour évaluation. Si l'un d'entre eux est réalisable et produit une valeur de fonction objective $\Phi(p) < \Phi^{best}$, alors p^{best} est réinitialisé à p et le processus est réitéré avec le nouveau meilleur paramètre. Sinon, la taille de pas Δ est réduite et le processus est réitéré.

Les modèles d'optimisation classique nécessitent de calculer le gradient de la fonction de coût. Si l'on ne connaît pas son gradient, ces méthodes ne fonctionnent plus. Alors que OPAL se sert du principe que dans une base positive, il existe toujours une direction de descente.

2.3.2 Algorithme de Recuit Simulé

L'algorithme de Recuit Simulé [23] est un algorithme d'optimisation, inspiré de la métallurgie. Il est particulièrement adapté aux problèmes d'optimisation où l'on risque de rester coincé dans un minimum local et ne jamais trouver le minimum global. Pour comprendre son fonctionnement, faisons l'analogie avec la métallurgie :

Considérons un morceau de métal que l'on chauffe. Les atomes de métal dans ce morceau sont organisés en un réseau dont le motif varie en fonction du métal. Chauffer le métal revient à faire vibrer plus ou moins fort les atomes autour de leur position moyenne. Le réseau se déforme et l'ensemble des atomes acquiert de l'énergie. Lorsque le morceau de métal refroidit, on s'attend à ce que les atomes reviennent sagement à leur position d'origine et que tout revienne dans l'ordre, c'est-à-dire que le morceau de métal revient à son énergie initiale. Mais cela ne se passe pas exactement comme ça... Si le refroidissement est brutal, le réseau métallique peut se bloquer dans des configurations qui ne sont pas la configuration initiale, que l'on posera comme celle d'énergie minimum. Des tensions se créent, qui ont généralement un effet néfaste sur les caractéristiques mécaniques du métal, sauf cas particulier où l'on cherche cet effet, comme une trempe par exemple. Une solution est de recuire le métal légèrement lors de son refroidissement et ainsi s'assurer de retomber dans l'état initial d'énergie minimum.

L'algorithme de Recuit Simulé est une application numérique du processus de métallurgie dont le but est de trouver le minimum global d'une fonction sans rester bloqué dans un minimum local. Voici l'algorithme du Recuit Simulé appliqué à l'optimisation d'une fonction f(x1,x2,...,xn) :

- Définir la fonction f, la valeur initiale des variables x_i et la valeur initiale T_0 de la température
- Tant que $T_i < T_{finale}$
  - Atteindre l'équilibre du système à T_i
  - Calculer l'énergie moyenne du système pour T_i
  - Traiter l'écart d'énergie $(E_i - E_i - 1)$ selon l'algorithme de Metropolis :
    - Soit x_i un état d'énergie E_i, dans notre système
    - Je fais varier x_i en lui ajoutant une grandeur aléatoire comprise entre 0 et 1, j'obtiens x_{i+1} d'énergie E_{i+1}
    - si $E_{i+1} \leq E_i$ alors x_{i+1} est le nouvel état de mon système, parce que son énergie diminue
    - sinon, je ne me bloque pas pour ne pas rester dans un minimum local éventuel, je décide que x_{i+1} devient mon nouvel état avec une probabilité égale à $\exp^{-\frac{E_{i+1}-E_i}{k_B T}}$
  - Sinon, on diminue la température selon la formule choisie

2.3.3 Algorithme d'évolution différentielle

L'évolution différentielle [24] est une méthode stochastique basée sur la population qui est utile pour les problèmes d'optimisation globale. À chaque passage dans la population, l'algorithme mute chaque solution candidate en se mélangeant à d'autres solutions candidates pour créer un candidat d'essai. Il existe plusieurs stratégies pour créer des candidats d'essai, chacune convient mieux à certains problèmes qu'à d'autres. La stratégie «best1bin» est un bon point de départ pour de nombreux systèmes. Dans cette stratégie, deux membres de la population sont choisis au hasard. Leur différence est utilisée pour muter le meilleur membre (le «meilleur» dans «best1bin»), jusqu'à présent :

$$b' = b_0 + mutation * (population[rand0] - population[rand1])$$

Un vecteur d'essai est ensuite construit. En commençant par un i^{eme} paramètre choisi au hasard, l'essai est rempli séquentiellement (en modulo) avec les paramètres de b' ou du candidat d'origine. Le choix d'utiliser b' ou le candidat d'origine est fait avec une distribution binomiale (le' bin 'dans' best1bin ') - un nombre aléatoire dans [0, 1) est généré. Si ce nombre est inférieur à la constante de recombinaison, le paramètre est chargé à partir de b', sinon il est chargé à partir du candidat d'origine. Le paramètre final est toujours chargé à partir de b'. Une fois le candidat à l'essai construit, son aptitude est évaluée. Si le candidat à l'essai est meilleur que le candidat d'origine, alors il prend sa place. S'il est également meilleur que le meilleur candidat global, il le remplace également.

2.3.4 Évaluation des modèles

Lors d'un calcul de régression, l'objectif est non seulement de réduire l'erreur du modèle de régression sur le jeu de données qui ont permis de construire ce modèle (pour éviter le sous-apprentissage ou underfitting), mais aussi de réduire l'erreur du modèle sur la totalité de l'espace. En effet, si nous évaluons notre modèle uniquement sur les données qui ont permis de le construire, nous risquons un surapprentissage (overfiting), voir figure 2.5.

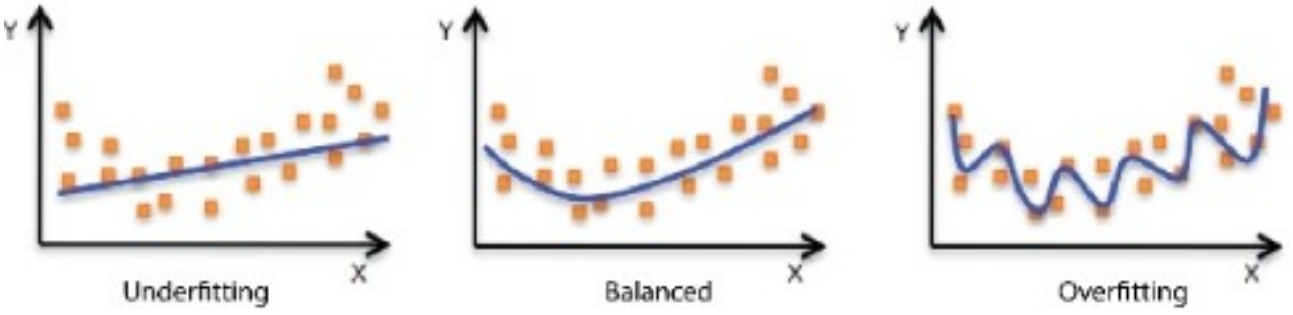


FIGURE 2.5 – Définition de sous et sur apprentissage

Pour ce faire, plusieurs méthodes existent pour tester la précision du modèle. Le plus important est de diviser nos données en set de données d'entraînement de test et de validation :

— données d'entraînements : l'ensemble $D_{training}$ utilisé pour créer le modèle prédictif
— données de test : l'ensemble D_{test} utilisé pour l'évaluation du modèle prédictif et l'optimisation des hyperparamètres
— données de validation : si nous devons choisir entre k modèles, le plus performant, nous ne pouvons pas évaluer leur performance sur le jeu de test qui a déjà servi à optimiser leurs hyperparamètres. Nous devons donc créer un 3^{eme} ensemble $D_{validation}$ pour sélectionner le meilleur modèle.

Pour ce faire nous nous servons d'un indicateur : le coefficient de détermination R^2 [25] et sa variante le coefficient de corrélation Q^2 :

$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$

Où :

— y_i sont les mesures du set de données d'entraînements
— $\widehat{y}_i$ sont les prédictions réalisées sur les points du set de données d'entraînements
— $\bar{y}$ est la moyenne des mesures du set de données d'entraînements

La différence avec Q^2 est que les données ne viennent pas du set d'entraînement qui ont servi à créer le modèle mais du set de test :

$Q^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$

Où :

— y_i sont les mesures du set de données de test
— $\widehat{y}_i$ sont les prédictions réalisés sur les points du set de données de test
— $\bar{y}$ est la moyenne des mesures du set de données de test

En observant l'expression de ces coefficients, nous comprenons que le but est de converger vers 1, qui est la valeur maximale. Il est d'usage de prendre 0.8 comme valeur de convergence satisfaisante. Si notre prédiction était totalement aléatoire, nous aurions la même espérance d'erreur "au dessus" que "en dessous" des set d'entraînements et de test, si nous appliquons cette information aux coefficients de déterminations, nous obtiendrons 0. Ce qui veut dire que tout résultat négatif est encore plus mauvais qu'une sélection aléatoire de points, ce qui n'est pas acceptable.

Si nous nous trouvons dans un cas de sous-apprentissage, le modèle ne capte pas assez d'informations des données d'entraînements. Ce qui signifie que R^2 et Q^2 seront faibles. Dans un cas d'apprentissage correct nous atteindrons un maximum pour Q^2 et R^2 sera grand. Dans le cas de surapprentissage, le modèle capte trop d'informations du set de données d'entraînement même son bruit, R^2 est donc très proche de 1 mais Q^2 est faible. Lors de la conception d'un modèle de régression, pour s'assurer de

sélectionner un modèle dont l'apprentissage est correct, nous pouvons tracer un graphique comme le montre la figure 2.6.

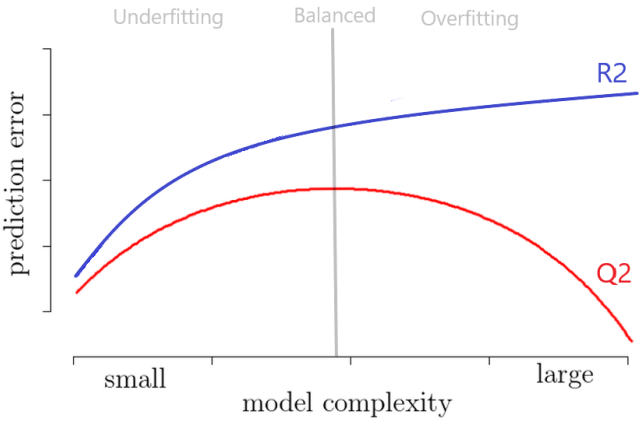


FIGURE 2.6 – Comparaison Q^2 et R^2

L'avantage du coefficient de détermination par rapport aux fonctions de pertes : Mean Square Error (MSE) Mean Absolute Error (MAE), est que grâce à la pondération par la différence des prédictions avec la moyenne des données, la comparaison de ces coefficients entre des prédictions réalisées avec des données différentes est possible. Nous avons de plus un critère de convergence d'usage à vérifier ($Q^2 = 0.8$).

2.4 Construction et Utilisation des modèles réduits non-intrusifs basés sur la POD

Comme nous l'avons vu dans 2, le modèle réduit non intrusif que nous construisons décompose l'espace par projection des données sur des sets de fonctions de bases de dimension plus petite, avec détermination des coefficients par apprentissage automatique. Ici nous allons nous intéresser à la construction de ces méthodes avec apprentissage supervisé, ainsi que leur utilisation dans la littérature scientifique.

2.4.1 Construction du modèle

Le modèle est construit de la manière suivante : Les matrices de sorties des calculs sont décomposées dans la base POD. Le but de cette décomposition est de conserver les structures cohérentes de l'écoulement et ainsi conserver un sens physique à notre modèle de substitution tout en réduisant les dimensions du problème. Nous nous servons des coefficients de la base POD comme entraînements pour le régresseur, pour ensuite demander au régresseur de prédire les coefficients de la base POD aux points d'intérêts. Enfin nous reconstruisons les prédictions dans l'espace initial. La méthode est décrite dans la figure 2.7.

2.4.2 Utilisation de la méthode

La méthode non intrusive basée sur une décomposition aux valeurs propres (POD) et régression dans l'espace réduit a été utilisée dans la littérature avec de bons résultats à plusieurs reprises.

Cette méthode de modèle de substitution a été utilisée dans des problèmes de "downscaling" [26], le but est de reconstruire les lames d'eau hivernale mensuelle dans la péninsule ibérique à partir du champ de pression au niveau de l'océan Atlantique Nord en hiver (décembre-février). Le modèle de substitution utilise la POD pour réduire la taille des données du champs de pression Atlantique Nord et extraire les composants principaux de l'écoulement (structures cohérentes), et ensuite par apprentissage automatique dans cet espace réduit prédit les précipitations dans la péninsule ibérique. La méthode a pu reproduire de manière très réaliste l'évolution des lames d'eau, même s'il a été observé que le krigeage sous-estime la variance d'observation en raison d'un effet lissant.

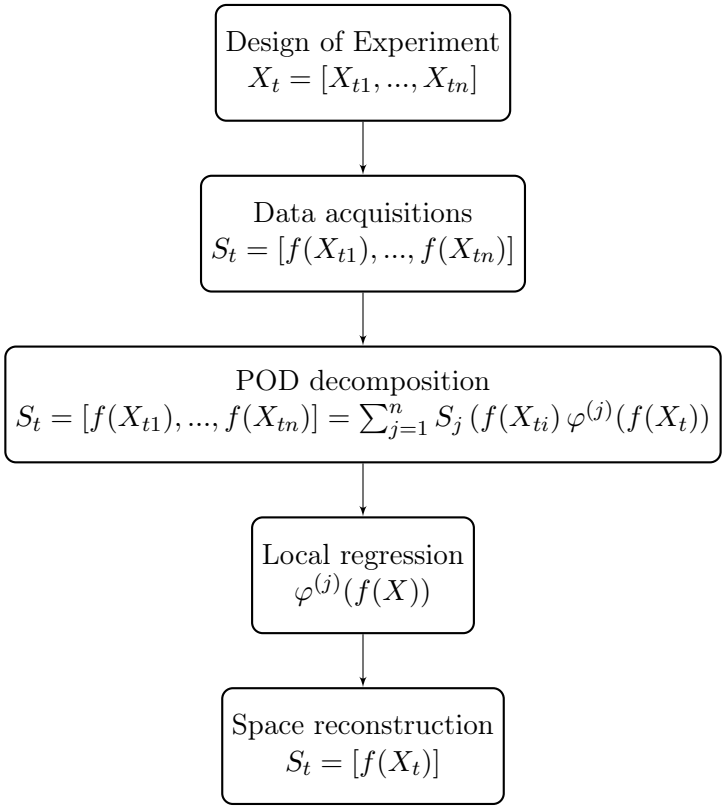


FIGURE 2.7 – Organigramme de la méthode non intrusive avec apprentissage supervisé

Cette méthode a aussi été utilisée pour réduire l'utilisation des simulations numériques lourdes de combustion en CFD [27] en faisant appel à un modèle de substitution créé à partir des sorties de codes CFD, décomposées dans une base POD et interpolées avec Krigeage. L'étude se concentre aussi sur la décomposition en modèles locaux, pour traiter les phénomènes physiques différents et ainsi mieux traiter les discontinuités. Il a été démontré que le modèle de substitution donnait des erreurs de prédictions inférieures à 10 %, et que l'utilisation de modèles locaux augmentait encore la qualité du modèle.

En médecine, les modèles de substitutions sont aussi utilisés [13]. Une méthode est proposée pour le diagnostic du cancer du col de l'utérus. Ce cancer se développe sans symptôme apparent et dès qu'il a atteint tous les organes commencent à se révéler, une multitude de paramètres peuvent être pris en compte (40 dans l'article) pour son dépistage mais il est difficile pour un médecin de tenir compte de tous ces paramètres pour diagnostiquer un risque. Une solution est de créer un système de machine learning avec réduction de l'espace des paramètres pour classer les patients selon le risque de contraction du cancer.

L'interpolation entre les données de puits est un problème bien connu en recherche sismique [14]. Pour réaliser des interpolations de données dans un espace d'ordre élevé, la méthode retenue est l'utilisation de Krigeage dans une base POD. Une part importante de l'étude est dédiée au coût de calcul et son optimisation, le modèle réduit final peut être utilisé sur un ordinateur portable classique avec un temps de calcul de l'ordre de la minute.

Nous pouvons citer aussi l'article de Cabrera [12], qui est l'article dont je suis le co-auteur, cet article reprend le travail de recherche réaliser sur les modèles de substitution avec apprentissage supervisé.

Chapitre 3

Modèles réduits locaux non intrusifs fondés sur l'apprentissage automatique non supervisé

Dans le cas où plusieurs phénomènes physiques sont représentés par les données (par exemple en aérodynamique avec des écoulements subsoniques, soniques et supersoniques), le modèle de substitution pour garder un maximum de sens physique peut être décomposé en modèles locaux. Chaque modèle local est un modèle de substitution d'un seul phénomène physique, les modèles locaux reprennent la construction des modèles non-intrusifs basés sur la POD dans un espace délimité appelé cluster. Cette méthode a prouvé être capable d'améliorer la qualité du modèle de substitution [27] [14] [28]. Pour créer ces clusters, nous pouvons faire appel à l'apprentissage automatique non supervisé (section 3.1).

3.1 Méthode d'apprentissage automatique non supervisé

3.1.1 Définition de l'apprentissage non supervisé

L'apprentissage non supervisé fait référence aux algorithmes de partition des données (clustering), c'est-à-dire créé des étiquettes aux données. Pour ce faire, différentes méthodes existent qui peuvent être basées sur la distance entre les données, l'optimisation de groupe de même taille et de même variance... Pour aller plus loin sur le sujet, le lecteur est invité à lire [29].

Par exemple, prenons le cas très connu des services de Streaming de vidéo. Le service récolte sur les utilisateurs de nombreuses données, qu'elles soient personnelles : âge, pays, langue ou d'utilisation : vidéos regardées, commentaires, like, recherche... Pour maintenir l'utilisateur sur la plateforme le plus longtemps possible, l'algorithme de suggestion doit affiner les résultats. Pour ce faire de nombreux groupes d'utilisateurs sont créés appelés cluster (environ 2000 sur Netflix selon l'entreprise) chaque cluster regroupe les utilisateurs qui ont des points communs basés sur l'étude de leurs données. Ainsi pour l'algorithme de suggestion, il suffit de retrouver les vidéos non vues par l'utilisateur parmi les vidéos aimées et qui ont fait réagir les autres utilisateurs du cluster.

Il faut retenir que le Cluster est un choix cartésien de l'algorithme d'apprentissage automatique qui ne fait pas appel à nos références subjectives de goût, mais juste une analyse statistique des données, l'étiquette créée est juste un numéro aléatoire.

3.1.2 Vue d'ensemble des méthodes de clustering

Dans cette partie nous nous intéresserons aux méthodes de clustering adaptés aux problèmes avec beaucoup de données et peu de clusters. En effet, dans notre modèle de substitution, nous ne souhaitons pas réaliser beaucoup de clusters car nous n'étudions pas des milliers de phénomènes physiques à la fois. Les données que nous traitons représentent un volume important d'informations et nous devons sélectionner l'algorithme de clustering qui est capable de traiter ces données en un temps suffisamment

court. Dans le tableau 3.1, nous trouverons les propriétés et caractéristiques de chaque méthode de Clustering pouvant nous intéresser.

Nom	Paramètres	Adaptabilité	Utilisation	Métrique utilisée
K-Means	Nombre de Clusters	Beaucoup de données peu de cluster	Usage général, même taille de cluster, géométrie plate, pas trop de clusters	Distance entre les points
Spectral clustering	Nombre de clusters	Peu de cluster	Peu de cluster, cluster de même taille, géométrie non plate	Graphe de voisin le plus proche
DBSCAN	Taille du voisinage	Beaucoup de données	Géométrie non plate, taille de cluster inégales	Distance entre points les plus proches

TABLE 3.1 – Propriété de chaque méthode de clustering

3.1.3 K-means

Nous utiliserons dans ce travail une méthode de groupement (clustering en anglais), appelée K-means [30]. Cet algorithme essaye de séparer les données d'entrée en n groupes de variance identique. Pour ce faire, la méthode minimise un critère d'inertie. Cet algorithme nécessite de spécifier le nombre de groupes. Il est particulièrement adapté aux cas où les données d'entrées sont de tailles importantes.

Prenons N données X, et essayons de les grouper en K groupe disjoint C_j, chacun des ces groupes a une moyenne μ_j. La moyenne des groupes est appelée centroïde. L'algorithme K-means cherche à minimiser le paramètre d'inertie défini comme :

$$\sum_{i=0}^{n} \underset{\mu_j \in C}{min}(||x_i - \mu_j||^2)$$

L'algorithme comporte quatre étapes :

— 1 : Choisir les centroïdes initiaux en prenant au hasard K données.
— 2 : Assigner aux données restantes le centroïde le plus proche.
— 3 : Créer de nouveaux centroïdes en prenant la moyenne de toutes les données associées aux précédents centroïdes.
— 4 : Calculer la différence entre les nouveaux et les anciens centroïdes et boucler sur les étapes 2, 3 et 4 jusqu'à atteindre une différence inférieure à une limite prédéfinie.

Le problème principal de la méthode K-Mean est qu'elle convergera vers un minimum local et dans certains cas complexes, les résultats peuvent être mauvais. Il y a aussi le fait que l'utilisateur doit définir le bon nombre de groupes sinon le regroupement n'aura aucun sens. Une illustration des limitations de la méthode est montrée figure 3.1.

Pour pallier aux problèmes de K-means, des algorithmes d'initialisation ont été créés pour choisir des centroïdes initiaux de manière à éviter les minimum locaux lors de la convergence. Par exemple K-means++ choisit les centroïdes initiaux de manière à garder une distance importante entre eux. Pour aller plus loin sur ce sujet, le lecteur est invité à lire [29].

3.1.4 Spectral clustering

SpectralClustering effectue une incorporation de faible dimension de la matrice d'affinité entre les données d'entraînements, suivi d'un regroupement. Il est particulièrement efficace en coût de calcul si la matrice d'affinité est clairsemée.

L'algorithme comporte cinq étapes :

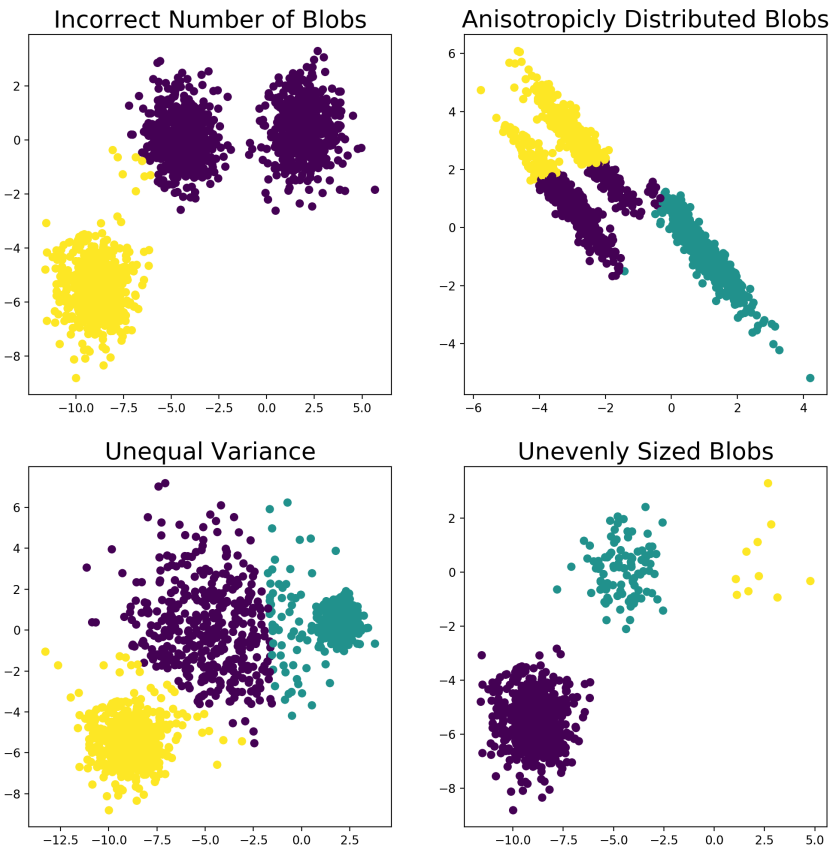


FIGURE 3.1 – Limitations de la méthode K-Mean

Entrée : données, nombre de clusters k
— Construit la matrice d'affinité S
— Calcule le Laplacien L
— Calcule les k premiers vecteurs propres de L et stocke les vecteurs en colonne dans la matrice U
— pour chaque ligne dans la matrice U
— Utiliser K-Means pour étiqueter les données de la ligne
Sortie : Clusters

3.1.5 DBScan

L'algorithme DBSCAN voit les clusters comme des zones de haute densité séparées par des zones de faible densité. En raison de cette vue plutôt générique, les clusters trouvés par DBSCAN peuvent avoir n'importe quelle forme, par opposition aux k-means qui supposent que les clusters sont de forme convexe. L'élément central du DBSCAN est le concept d'échantillons noyaux, qui sont des échantillons situés dans des zones à haute densité. Un noyau est donc un ensemble de données de base, chacune proche l'une de l'autre (mesuré par une mesure de distance) et un ensemble de données non noyau qui sont proche d'un noyau (mais ne sont pas eux-mêmes des noyaux). Il y a deux paramètres dans l'algorithme, min_samples et eps. Ces paramètres permettent d'adapter la définition de densité au problème.

L'algorithme comporte six étapes :
Entrée : données, min_samples, eps
— Pour tous points dans les données

— Chercher le nombre de voisins a une distance inférieure à eps : N (il existe plusieurs fonctions distance)
— Si N est inférieur à min_samples, Le point est considéré comme du bruit
— Sinon
  — S'il n'a pas de noyau comme voisin, il devient un noyau
  — S'il a un noyau comme voisin, il étend le cluster de ce noyau

3.2 Modèle réduit avec apprentissage non supervisé

3.2.1 Construction de la méthode

Cette fois-ci la différence se fait dès le début entre l'acquisition des données et la décomposition POD. Nous allons cette fois créer des groupes basés sur une ou plusieurs variables. Chacun de ces groupes permet d'isoler un comportement physique différent et ainsi optimiser la qualité du modèle de substitution. Pour optimiser le groupement, nous utilisons un senseur. Le senseur permet d'augmenter la variance des variables de groupements et ainsi d'obtenir un groupement de meilleure qualité. Enfin comme les variables de groupement et d'entraînements n'ont pas toujours le même nombre de points de calcul, nous devons utiliser une classification (ExtraTrees regressor pour son coût de calcul faible) pour conserver l'étiquette des groupes sur les variables d'entraînements. La méthode est décrite dans la miniature 2.7 et est inspirée des articles de Dupuis & al [28] ainsi que de Amsallem & al [31].

3.2.2 Utilsation de la méthode

La méthode non intrusive basée sur le clustering suivi d'une décomposition aux valeurs propres (POD) et régression dans l'espace réduit a été moins utilisée dans la littérature que la méthode précédente mais a présenté des résultats encourageants.

Les calculs de combustion sont très lourds en CFD et ont souvent des zones où la physique est différente, l'utilisation de modèle de substitution y est souvent présent [27]. Le but est de réduire l'utilisation des simulations numériques lourdes CFD en faisant appel à un modèle de substitution créé à partir des sorties de codes CFD, décomposé dans une base POD pour réduire la quantité de données et conserver un sens physique et interpolé avec Krigeage. L'étude se concentre aussi sur la décomposition en modèles locaux. En effet, la combustion peut suivre différents régimes qui ont des comportements physiques différents. Ce passage entre régimes crée des discontinuités et peut empêcher notre modèle de substitution de prédire correctement l'écoulement. L'utilisation de modèles locaux est alors envisagée pour pallier ce problème. Ces groupes locaux contrairement à la méthode présentée 3.2 sont créés par les chercheurs plutôt que d'utiliser un algorithme non supervisé.

Les prédictions dans ces groupes locaux ont une erreur inférieure à 7% comparé à l'erreur précédente inférieure à 10%, nous voyons bien que ces modèles locaux sont d'une grande utilité pour traiter les problèmes non linéaires.

L'interpolation entre les données de puits est un problème bien connu en recherche sismique [14], est décrite la méthode utilisée par des chercheurs pour réaliser des interpolations locales de données dans un espace d'ordre élevé. La méthode retenue est l'utilisation de Krigeage dans une base POD. Une part importante de l'étude est dédiée au coût de calcul et son optimisation. Les modèles réduits locaux peuvent être utilisés sur un ordinateur portable classique avec un temps de calcul de l'ordre de la minute.

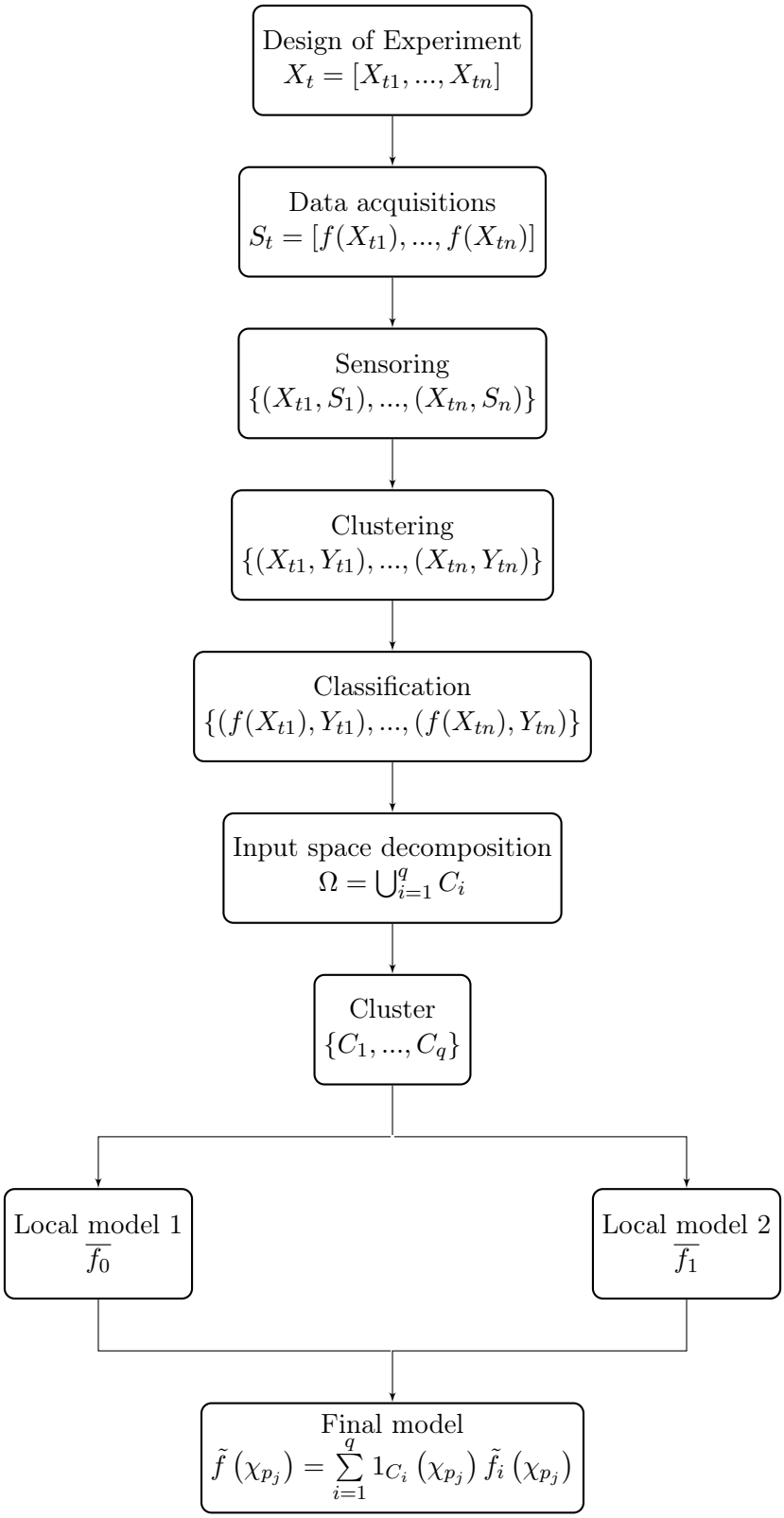


FIGURE 3.2 – Organigramme de la méthode non intrusive avec apprentissage non supervisé

Chapitre 4

L'outil BATMAN

Au cours de mon stage, une partie délicate de ma mission est d'intégrer mes développements dans l'outil BATMAN développé au CERFACS par un doctorant il y a maintenant 2 ans. L'outil conçu pour traiter les sorties de codes CFD et analyser les résultats commence à ne plus être compatible avec les dernières bibliothèques Python. De plus, dans le cadre de mes recherches, je dois utiliser d'autres méthodes (que l'analyse Bayésienne), basées sur la bibliothèque de machine learning SciKitLearn. Par la même occasion, je dois rajouter les fonctionnalités de SciKitLearn à l'outil BATMAN.

BATMAN est un logiciel open source licence CeCILL-B [1] disponible en ligne gratuitement sur Gitlab (voir section suivante). N'importe qui peut installer BATMAN, soit directement avec Python : "pip install" ou conda "conda install", les deux commandes vont automatiquement pointer vers un repository Gitlab : `https://gitlab.com/cerfacs/batman`. Pour mettre à jour le logiciel, il faut télécharger sa version de développement, créer une branche sur Gitlab, et répondre à tous les critères, pour que la mise à jour soit acceptée et puisse être "merge" dans la branche principale. Il est donc impératif de garder un code propre, de maintenir la documentation à jour et de tester au moins 80% des lignes du code, il ne doit pas y avoir la moindre erreur lors des tests. Cette méthode de développement garantit le maintien de la qualité du logiciel, alors même que le développeur initial ne travaille plus dessus depuis un certain temps.

4.1 Présentation de l'outil

BATMAN [4] est un outil développé au CERFACS, dont la signification est : Bayesian Analysis Tool for Modelling and uncertAinty quaNtification. Batman crée des modèles de substitutions pour les problèmes de calculs lourds. La procédure générale pour créer ces modèles de substitution est :

- Design Of Experiment (DOE) : BATMAN crée les points d'espace desquels les snapshots devront être pris. Différentes méthodes de sampling peuvent être utilisées : Halton [32], Sobol [33], Uniform, Latin Hypercube Sampling LHS [34], Faure [35] ou Saltelli [35].
- Réalise une analyse physique et une compression données en utlisant un POD 2.1 (optionnel).
- Construction du modèle de substitution : le modèle apprend les relations entre les variables, plusieurs méthodes sont disponibles : Gaussian Process 2.2.1 , Polynomial Chaos expansion [36] ou toute méthodes de scikit Learn [2] [5]. Si l'option POD est choisie le modèle de substitution apprend dans l'espace POD.
- Analyse de sensibilité (optionnel) : variabilité intrinsèque : calcule l'indice de Sobol [3] pour estimer la contribution de chaque variable sur les données de sortie. Ou quantification des incertitudes : calcule la fonction de densité de probabilité des données de sorties.
- Crée des graphes de visualisation des résultats.

1. CeCILL-B suit largement le modèle de la populaire licence BSD et de ses variantes (licences Apache, X11 ou W3C, parmi d'autres). En échange d'une forte obligation de citation (dans tout logiciel incorporant un logiciel sous CeCILL-B, mais aussi à travers un site Web dans ce cas), l'auteur autorise la réutilisation de son logiciel sans aucune autre contrainte.
2. Scikit Learn est une bibliothèque d'outils de machine learning open source
3. les indices de Sobol sont des indices de sensibilité d'une variable de sortie à une variable d'entrée.

4.2 Modifications apportées

- Nouveautés :
  - Ajout de la méthode de régression “mixture” : classification/ régression
  - Ajout des méthodes de régression de SciKitLearn : LinearRegression, LogisticRegression, LogisticRegressionCV, PassiveAggressiveRegressor, SGDRegressor, TheilSenRegressor, DecisionTreeRegressor, GradientBoostingRegressor, AdaBoostRegressor, RandomForestRegressor et ExtraTreesRegressor
  - Ajout d’une option relative aux paramètres des régresseurs
  - Ajout de la méthode de resampling sigma distance
- Améliorations :
  - Option visualization devient optionnelle pour réduire le temps de calcul quand cette dernière n’est pas nécessaire
  - Ajout options d’entrée schema.json relative aux nouveautés
  - Mise à jour de la documentation
- Résolution du bug :
  - Copie la classe multiprocessing de SciKitLearn pour augmenter la robustesse et s’assurer une compatibilité avec toutes les classes python
  - Résolution de problème lié à la création de dataframe vide lent
  - Mise à jour de la typographie relative aux dernières versions de numpy

4.2.1 Intégration complète de la classe Mixture

Un modèle de “Mixture” (mélange) gaussien est un modèle probabilistique qui suppose que tous les points de données sont générés à partir d’un mélange d’un nombre fini de distributions gaussiennes avec des paramètres inconnus. On peut penser aux modèles de mélange comme généralisant le regroupement des k-mean (voir section apprentissage non supervisé) pour incorporer des informations sur la structure de covariance des données ainsi que sur les centres des Gaussiens latents.

Le problème est que la classe mixture n’était pas reliée au fichier d’entrée settings.jon, en découle tout un travail de transmissions et vérifications de paramètres jusqu’à la classe mixture. Cette classe n’est pas développée par moi, mais puisque la documentation ne la mentionnait pas et que l’intégration n’était pas finie, la mise à jour apportée permet de la révéler.

4.2.2 Ajout des méthodes sklearn, et des paramètres associés

C’est la grande nouveauté apportée à Batman. Sklearn est une bibliothèque libre Python destinée à l’apprentissage automatique. Elle est développée par de nombreux contributeurs notamment dans le monde académique par des instituts français d’enseignement supérieur et de recherche comme Inria. Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue pour s’harmoniser avec d’autres bibliothèques libres Python, notamment NumPy et SciPy.

Les méthodes de sklearn ne peuvent pas réaliser de quantification d’incertitude comme les processus gaussiens, qui étaient à l’origine de Batman. Il est donc important de cloisonner l’utilisation de l’option sklearn à l’apprentissage et la prédiction. Toutes les méthodes développées dans sklearn sont dépendantes d’hyperparamètres ayant une influence importante sur la qualité du modèle de substitution final. Il aura donc fallu créer des tests et transmettre toutes les informations nécessaires depuis le fichier d’entrée settings.json. A noter que Batman appelle la bibliothèque sklearn qui doit être installée dans l’environnement de travail, ainsi l’intégration des mises à jour de sklearn est automatique et robuste.

4.2.3 Nouvelle méthode de resampling

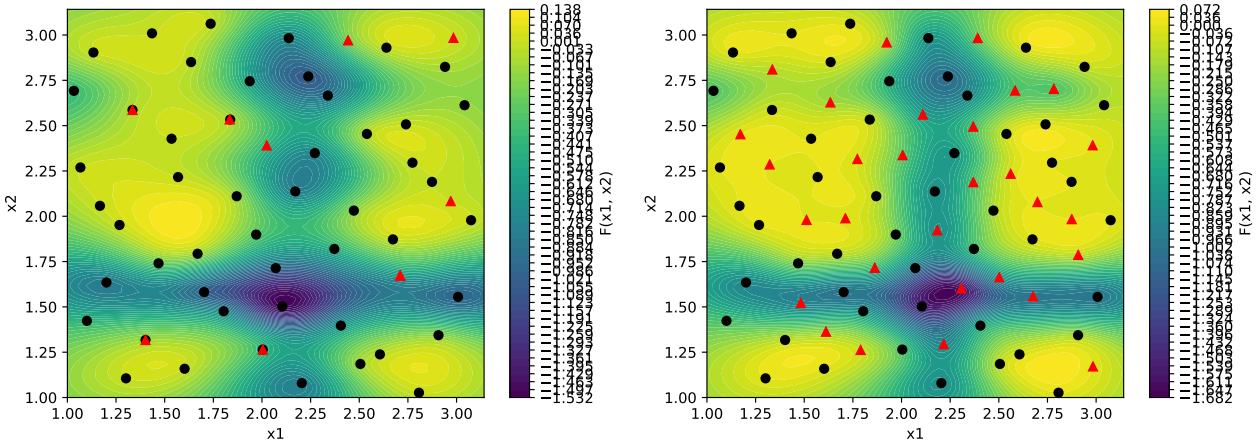
Le resampling est une technique essentielle de l’outil BATMAN. Le principe est de placer de nouveaux points de calcul à des endroits stratégiques :

- Au point qui maximise la répartition spatiale

— Au point où la variance est la plus élevée
— Au point où l'erreur est la plus importante
— Au point qui a la valeur maximale ou minimale en cas de problème d'optimisation

La technique la plus utilisée est la technique de variance ou technique appelée "sigma" , qui place un nouveau point sur l'endroit de la surface de réponse avec la variance la plus élevée. Le problème est que ce point de variance max peut être confondu avec les points de sampling initial voir figure 4.1a.

La recherche du point de variance maximum se fait par évolution différentielle, voir section 2.3.3. Dans Batman, la méthode d'optimisation par évolution différentielle recherche toujours un minimum, le problème d'optimisation est donc $min(-\sigma(x))$. Une solution apportée est de ne pas changer la fonction de resampling sigma mais de créer une nouvelle fonction qui empêche la superposition de points de resampling avec les points déjà présents. Pour ce faire nous transformons le problème d'optimisation et y ajoutons l'inverse de la distance au point le plus proche : $min(-\sigma(x) + \frac{1}{d_{plusproche}(x)})$ en y ajoutant des coefficients choisis par l'utilisateur pour équilibrer l'impact de sigma et de la distance, le problème devient : $min(-weight_1\sigma(x) + \frac{weight_2}{d_{plusproche}(x)})$. Grâce aux nouveaux termes si la distance au point le plus proche tend vers 0, $\frac{1}{d_{plusproche}(x)})$ tend vers $+\infty$ ce qui évite à la méthode d'optimisation d'évolution différentielle de garder ce point. Le résultat de la nouvelle méthode de resampling est montré figure 4.1b.



(a) Technique de resampling sigma

(b) Technique de resampling sigma distance

FIGURE 4.1 – Resampling technique comparaison, les points noirs représentent le sampling initial, les triangles rouges les points de resampling ajoutés

4.2.4 Nouvelle méthode de parallélisation

Lorsque dans un algorithme, une étape est longue, car elle a besoin de faire beaucoup d'itérations, si l'itération suivante ne requiert pas d'information sur l'itération précédente, alors nous pouvons paralléliser notre algorithme (=multiprocessing). Le but est de répartir le travail global en sous-tâches exécutées sur chaque unité logique disponible en parallèle. Par exemple, si un algorithme doit lire un fichier très long, plutôt que de le faire tourner sur un processeur qui va lire ligne par ligne le fichier, nous pouvons demander à l'algorithme d'attribuer la tâche de lecture d'un quart du fichier à quatre processeurs, et donc aller plus vite.

BATMAN est un outil qui utilise beaucoup la parallélisation pour accélérer les calculs. Cependant, entre les dernières versions de Python, la définition de multiprocessing a changé, et a donc rendu obsolète l'outil. Une solution apportée est l'intégration d'une nouvelle méthode de multiprocessing basée sur une bibliothèque open source maintenue à jour par d'autres développeurs : joblib [37]. Ainsi, l'outil est beaucoup plus stable et sera maintenu à jour sans avoir besoin d'une intervention d'un employé du CERFACS lors d'un changement de version sur le multiprocessing python.

Chapitre 5

Évaluation des méthodes et application à la prévision instantanée des lames d'eau

5.1 Benchmarks

Nous avons vu que l'outil BATMAN intègre la méthode de régression Kriging, et cette méthode avait déjà fait ses preuves dans de nombreuses études scientifiques (section 2.4). Dans le cadre des recherches pour Météo-France, nous devons tenir compte d'une contrainte de temps (résultat inférieur à la minute). Nous avons vu dans le chapitre 2 que d'autres méthodes plus efficaces en termes de temps de calcul pouvaient être utilisées : Random Forest et ses variantes ExtraTrees et AdaBoost. Nous allons donc tester ces méthodes sur des benchmarks, c'est-à-dire des fonctions connues difficiles à prédire.

5.1.1 Test kriging versus random forest

Les tests de comparaison seront portés sur les fonctions de Michalewicz et d'Ishigami :

$$\text{Michalewicz} : f(x_1, x_2) = -\sum_{i=1}^{2} \sin(x_i) \sin^{2*10}\left(\frac{ix_i^2}{\pi}\right) \text{ pour } x_1, x_2 \in [[1, 3.1415][1, 3.1415]]$$

$$\text{Ishigami} : f(x_1, x_2, x_3) = \sin(x_1) + 7\sin(x_2)^2 + 0.1x_3^4 \sin(x_1) \text{ pour}$$
$$x_1, x_2, x_3 \in [[-3.1415, 3.1415][-3.1415, 3.1415][-3.1415, 3.1415]]$$

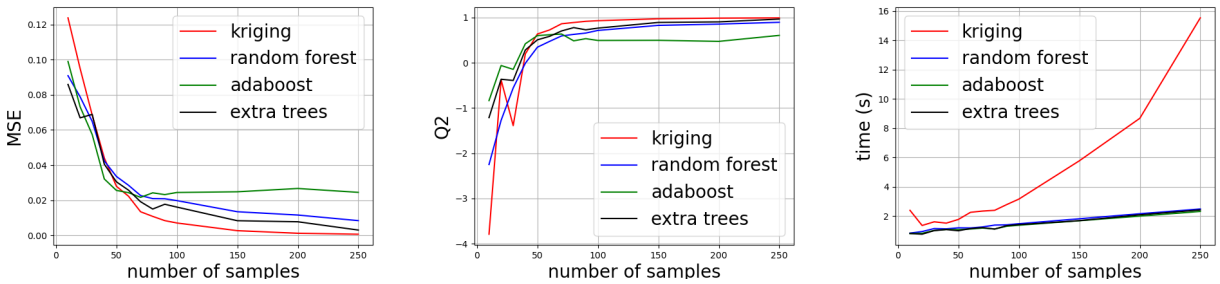
Le test se fera sur plusieurs échantillons d'entraînements, correspondant à un échantillonnage de sobol de 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250 samples. Nous testerons l'erreur (mse, q2) sur un échantillonnage uniforme (10 segmentations par axe).

Nous afficherons l'erreur (mse, q2), mais aussi le temps de calcul. Le tout sera affiché en fonction du nombre de training samples. Le but étant de chercher quel algorithme nous donne le meilleur résultat pour une limite de temps de calcul donnée, pour répondre aux exigences de Météo-France. Nous afficherons aussi les surface ou volume de prédiction de random forest ou ishigami. A noter que les résultats de Michalewicz seront représentés comme un nuage de points qui décrit une surface dont la dimension suivant z (notre prédiction) sera colorée selon une carte de couleur "hiver" pour mieux la distinguer. Il n'y aura pas de visualisation des résultats d'Ishigami car trop compliqués à observer en 3D.

5.1.2 Résultat des tests sur Michalewicz

Voici les résultats sur la fonction Michalewicz, ici nous comparons l'erreur MSE et Q2 mais aussi le temps de calcul, voir figure 5.1.

Les résultats nous montrent que Kriging comme utilisé avec Batman nous permet d'obtenir un meilleur coefficient de détermination sur notre échantillonnage de test par rapport à random forest (pour 250 sample q2 de kriging = 0.992 alors que q2 de random forest = 0.941) sachant que si l'on considère que le modèle a convergé lorsque q2 devient supérieur à 0.8, nous devons prendre environ 65 samples pour kriging pour un temps de calcul de plus de 7 secondes, alors que pour random forest,



(a) Comparaison de la MSE (b) Comparaison de Q^2 (c) Comparaison du temps de calcul

FIGURE 5.1 – Comparaison des performances des méthodes sur Michalewicz

nous devons prendre 120 samples mais avec un temps de calcul de 4.76 secondes. Pour comparer les résultats visuellement, vous trouverez ci-dessous la solution exacte suivie des prédictions de random forest et kriging, figure 5.2 et 5.3.

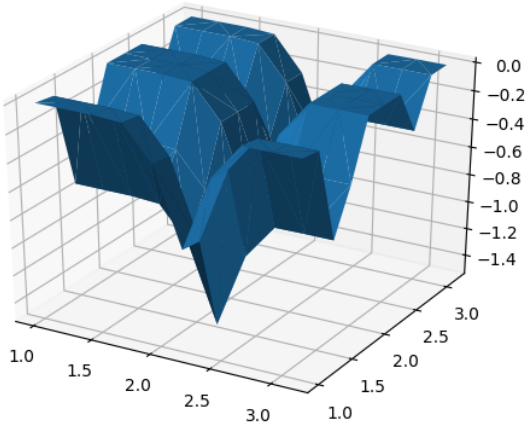


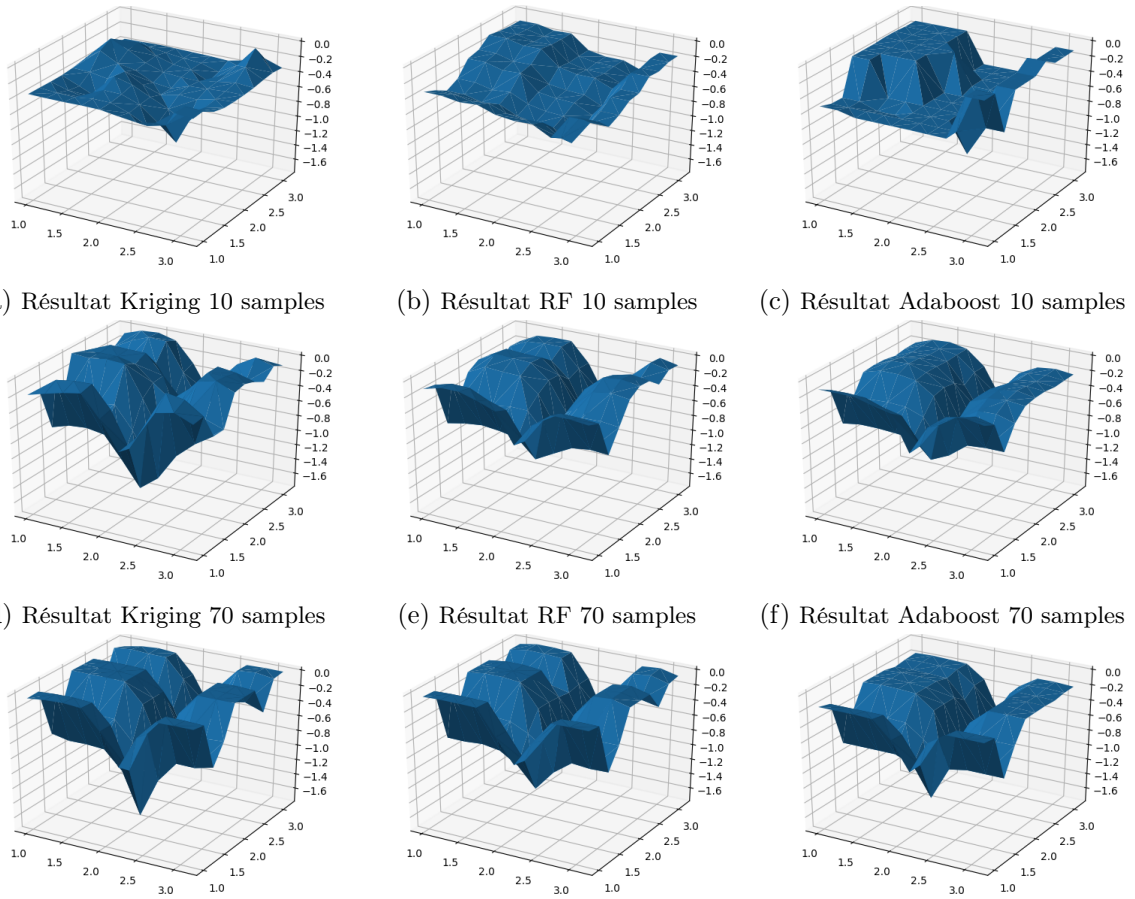
FIGURE 5.2 – Résultat exact Michalewicz

5.1.3 Résultat des tests sur Ishigami

Voici les résultats sur la fonction d'Ishigami, ici nous comparons l'erreur MSE et Q2 mais aussi le temps de calcul, voir figure 5.4.

Les résultats nous montrent que Kriging tel qu'utilisé avec Batman nous permet d'obtenir un meilleur coefficient de détermination sur notre échantillonnage de test par rapport à random forest pour 250 sample q2 de kriging = 0.913 alors que q2 de random forest = 0.700, sachant que si l'on considère que le modèle a convergé lorsque q2 devient supérieur à 0.8, nous devons prendre environ 175 samples pour kriging pour un temps de calcul de plus de 40 secondes, alors que pour random forest nous devons prendre 250 samples pour un temps de calcul de 3.91 secondes. Dans notre cas où nous avons un critère de temps de calcul à ne pas dépasser il semble intéressant d'utiliser Random Forest. Malheureusement, s'agissant d'une fonction 3d, les résultats ne sont pas facilement comparables à l'oeil nu.

Grâce à ces tests nous pouvons conclure que dans notre cas avec un critère de temps de calcul à respecter, il est préférable d'utiliser ExtraTrees Regressor, la variante de Random Forest même si les deux variantes ont des résultats similaires. Kriging est certes plus performante pour un nombre de données fixe sans limitation de temps de calcul, mais nous serions obligés de réduire les données

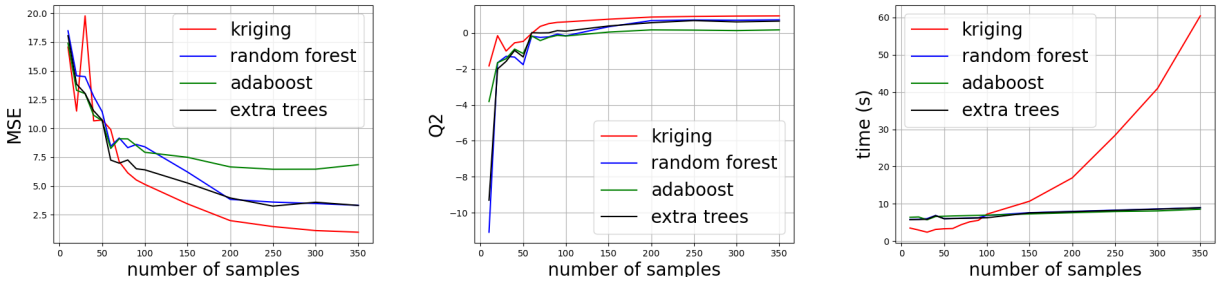


(a) Résultat Kriging 10 samples (b) Résultat RF 10 samples (c) Résultat Adaboost 10 samples

(d) Résultat Kriging 70 samples (e) Résultat RF 70 samples (f) Résultat Adaboost 70 samples

(g) Résultat Kriging 250 samples (h) Résultat RF 250 samples (i) Résultat Adaboost 250 samples

FIGURE 5.3 – Comparaison des résultats des méthodes sur Michalewicz



(a) Comparaison de la MSE (b) Comparaison de Q^2 (c) Comparaison du temps de calcul

FIGURE 5.4 – Comparaison des performances des méthodes sur Ishigami

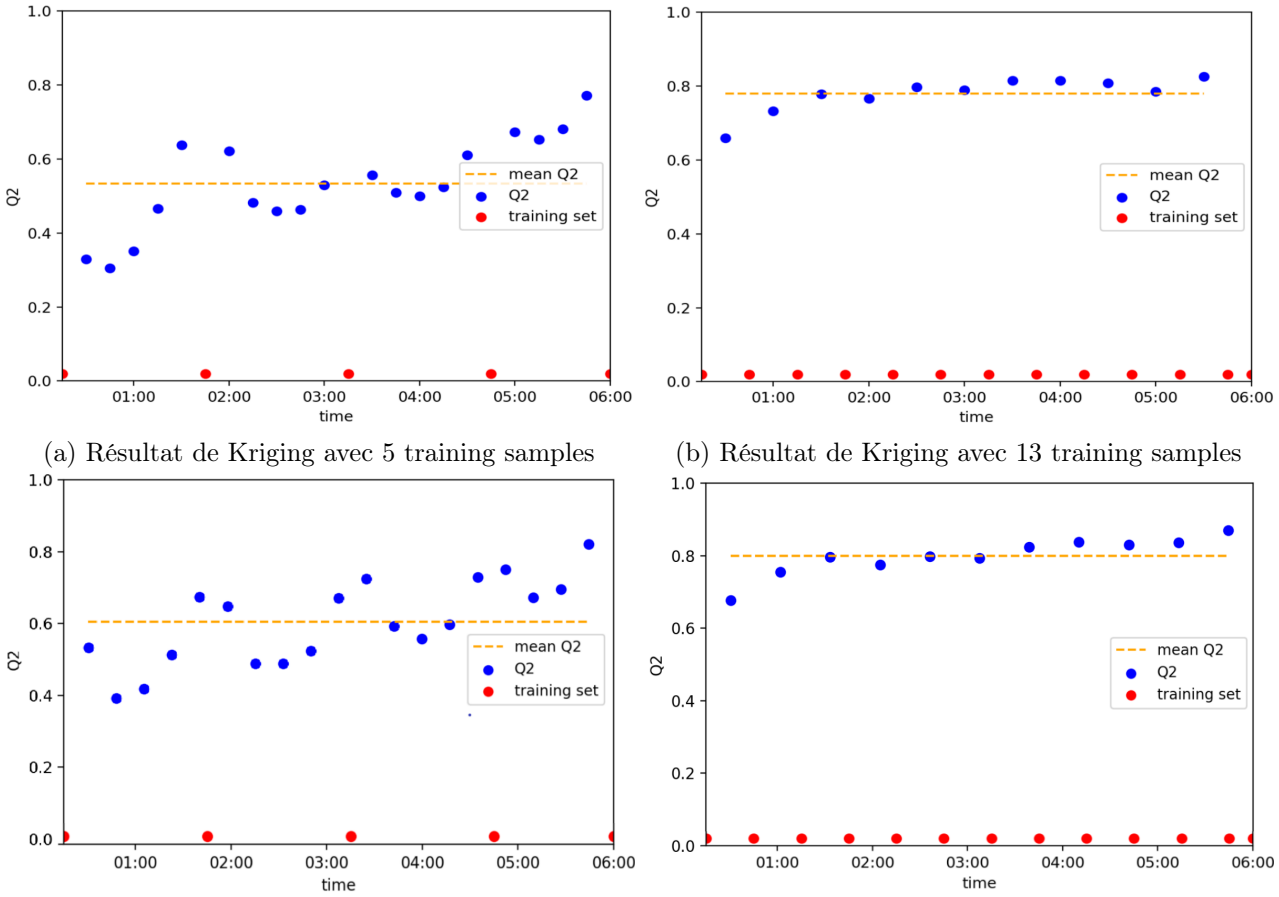
d'entraînements pour satisfaire les exigences de temps de calcul. La variante AdaBoost ne présente aucun intérêt dans notre cas.

5.2 Application à la prédiction instantanée des lames d'eau

5.2.1 Évaluation des modèles de substitution basés sur la POD

Dans cette partie, nous allons comparer les modèles construits à partir d'un POD et d'un régresseur : Kriging ou Random Forest. Pour ce faire nous allons entraîner nos modèles sur les prédictions d'Arome (qui contient 24 set de données comme cité précédemment) mais avec seulement 5, 7, 9 ou 13 données d'entraînements. Le but étant de comparer les performances de Random Forest et de Kriging dans des tests déjà réalisés avec Kriging. Nous évaluerons les performances de nos modèles sur les données d'Arome qui ne sont pas utilisés, nous évaluons donc le coefficient de détermination Q^2.

Cette fois, nous voulons effectuer une analyse statistique plus poussée, nous testerons donc chacun des modèles sur 10 tests. Ces tests correspondent à des prédictions d'Arome au début 2018 une période qualifiée de très pluvieuse sur le site de Météo France. Nous pouvons observer les résultats de nos modèles sur un test sur la figure 5.5



(a) Résultat de Kriging avec 5 training samples

(b) Résultat de Kriging avec 13 training samples

(c) Résultat de RF avec 5 training samples

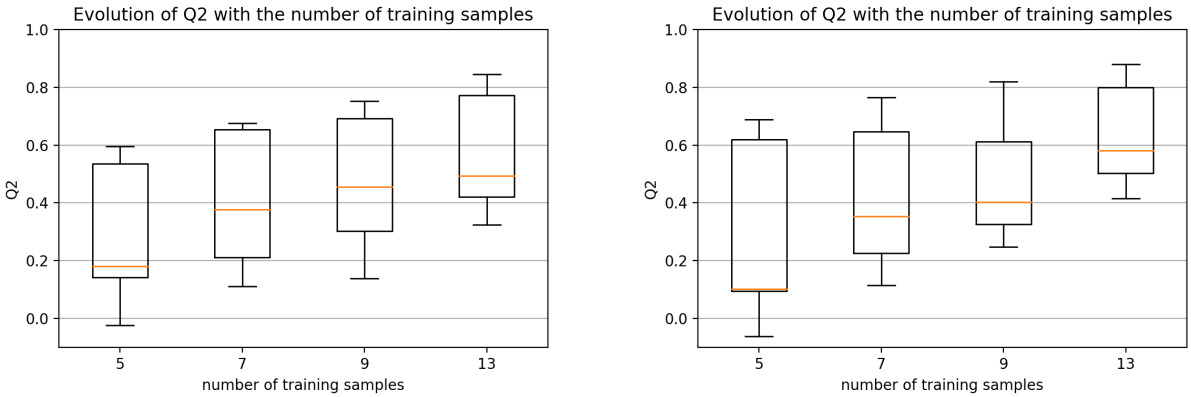
(d) Résultat de RF avec 13 training samples

FIGURE 5.5 – Évolution de Q^2 pour le test du 4 Janvier 2018 : les points bleus représentent Q^2, les points rouges sont les points d'entraînements et la ligne orange représente la moyenne des Q^2

Nous pouvons observer la répartition des résultats sur tous les tests dans ce graphique à moustache figure 5.6 (une explication sur le fonctionnement des graphiques à moustache est à retrouver en annexes 5.2.7).

Malheureusement nous ne pouvons pas évaluer le coefficient de détermination lorsqu'on utilise les 24 données d'entraînements, nous utiliserons donc la technique "LOO" : Leave One Out, pour tester notre modèle. Il s'agit de laisser un point "dehors" et ne pas l'utiliser dans l'entraînement, puis comparer la performance du modèle sur ce point. En répétant la technique plusieurs fois avec des points différents, on peut déterminer une approximation relativement proche même si on sous-évalue les performances du modèle avec 24 données d'entraînements.

Les résultats de la technique LOO sont présentés figure 5.7, nous pouvons observer que la valeur



(a) Résultat de Kriging

(b) Résultat de RF

FIGURE 5.6 – Evolution de Q^2 en fonction du nombre de samples entraînements (5, 7, 9 et 13)

de Q^2 augmente par rapport aux résultats de la figure 5.6, alors que nous savons que la méthode LOO sous-estime le Q^2. Nous voyons que le modèle de substitution utilisant Extra Trees présente de meilleurs résultats que Kriging.

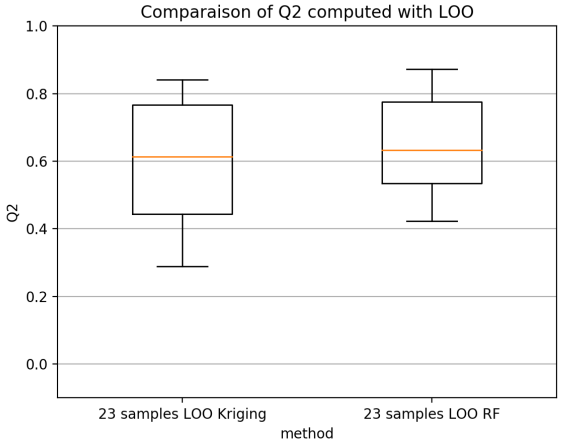


FIGURE 5.7 – Estimation du Q^2 par LOO, comparaison des méthodes Kriging et RF

Il est important de noter que la différence de temps de calcul est importante entre le modèle créé avec Random Forest et le modèle utilisant Kriging. Tous les calculs ont été effectués sur une machine du CERFACS appelée NEMO, via une connexion à un noeud de calcul disposant de 2 processeurs Intel 12 coeurs E5-2680-v3 à 2.5 Ghz et 64 GO de mémoire DDR4 1.3.2.

Les calculs avec 23 samples de données d'entraînements prennent un temps CPU [1] de 1m49.098s pour Kriging et 0m43.758 pour Random Forest. Mais le temps de calcul augmente de façon exponentielle suivant le nombre de données d'entraînements avec Kriging à cause de l'inversion de la matrice de covariance de dimension $N_{samples} \times N_{samples}$: pour 71 samples le temps CPU est de 95m57.972s, alors qu'il reste beaucoup plus raisonnable avec Random Forest : pour 71 samples le temps CPU est de 2m18.391s.

5.2.2 Utiliser d'autres données

Quelles données supplémentaires sélectionner ?

Pour augmenter le nombre de données d'entraînements, 3 approches vont être explorées :

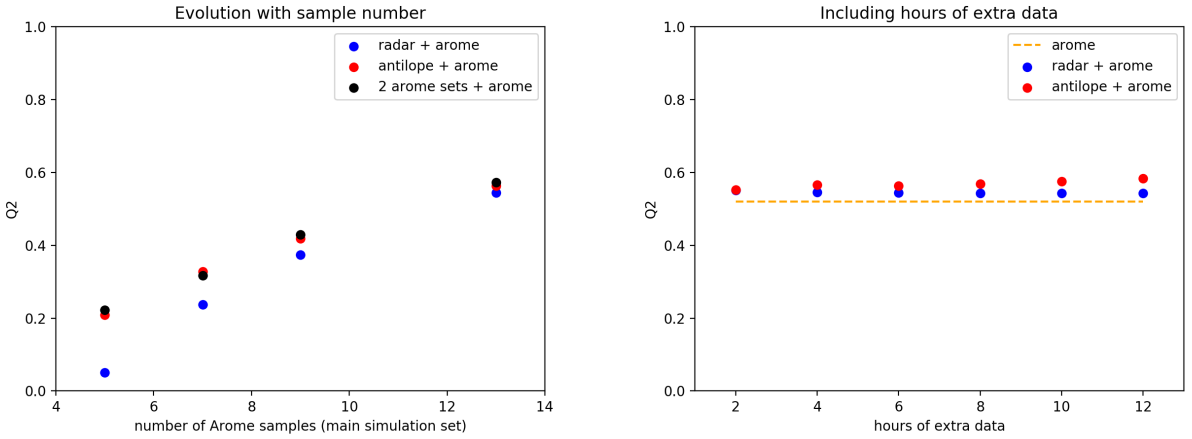
1. Le temps processeur (ou CPU) est la somme des intervalles de temps de chaque calcul élémentaire d'un processus. Il retire les interruptions provoquées par d'autres programmes prioritaires et somme les temps processeur des différents processus en parallèle.

— Utiliser les données précédentes d'Arome-NWC
— Utiliser les données précédentes provenant des radars
— Utiliser les données précédentes d'Antilope

Les données radar correspondent aux mesures de lames d'eau réalisées par Radar, nous utiliserons ici les données 2D. Des données 3D existent et sont même les conditions initiales du calcul Arome-NWC, il n'y a donc pas de discontinuité entre les données radar et les données Arome-NWC. Les données Radar sont disponibles toutes les 5 minutes.

Les données Antilope correspondent à une correction des données Radar par les données au sol. Elles sont disponibles toutes les 15 minutes et ont un temps de calcul de 9 minutes.

Nous allons donc comparer l'évolution de Q^2 dans un premier temps avec plus ou moins de données d'entraînements (comprenant toujours les données Arome plus les données radar ou antilope ou des extra données arome), figure de gauche. Puis sur la figure de droite l'effet de la quantité des sets précédents sur Q^2 (remontant sur 2 à 12h mais en utilisant toujours 13 données d'entraînements au total, nous utiliserons un sampling de type sobol) avec comme référence la ligne orange correspondant au test avec 13 données d'entraînements des prédictions Arome-NWC. Toutes ces expériences ont été réalisées avec le modèle de substitution POD + Kriging. Voir figure 5.8.



(a) Résultats avec données supplémentaires (b) Résultat en fonction du nombre de sets de données

FIGURE 5.8 – Évolution de Q^2 en fonction de données supplémentaires

Avec les résultats de la figure de droite, on remarque que les entraînements utilisant des données d'Antilope obtiennent de meilleurs Q^2, malheureusement comme cité précédemment le temps de calcul des données Antilope est de 9 minutes et nous ne pouvons pas nous permettre d'attendre 9 minutes quand le but de notre application est de donner des résultats dans la minute. De plus, dans la figure de gauche, on voit qu'utiliser des données Aromes ou Antillope précédentes donne relativement le même Q^2.

Dans l'ensemble l'utilisation des données radar a des résultats peu satisfaisants. Nous retiendrons donc dans le reste de nos expériences que l'utilisation de set de données Arome précédent est la meilleure option.

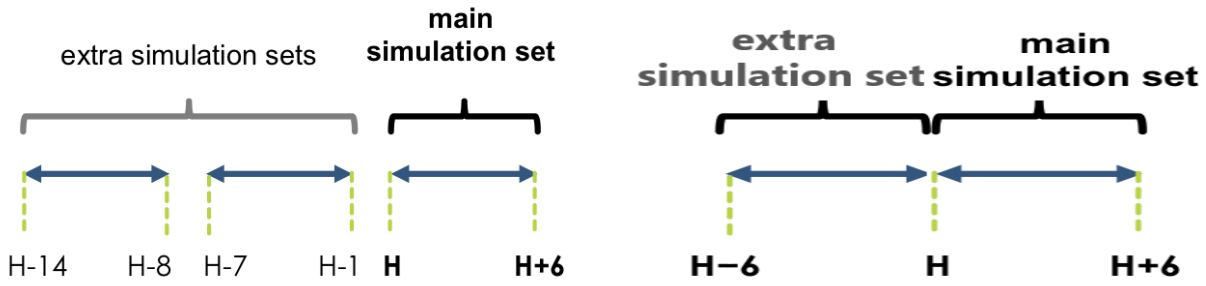
Prédictions avec des sets de données Arome-NWC supplémentaires

Puisque la méthode retenue est de prendre des sets précédents de données Arome-NWC, c'est-à-dire les sorties des calculs Arome-NWC réalisés avant la prédiction Arome-NWC que nous cherchons à interpoler, nous devons retenir deux points importants :

— Premièrement, les sets supplémentaires doivent être choisis de sorte qu'il n'y ait qu'une prédiction à chaque instant t (pas de chevauchement de sets de données)
— Secondement, entre deux sets de données Arome-NWC vu qu'il s'agit de prédiction inexacte il y a discontinuité entre les différents sets de données consécutifs

Kriging est une technique qui peut être biaisée pas la discontinuité des données d'entraînements vu qu'elle interprétera la discontinuité comme une variance importante et accordera un poids important à cet élément lors du calcul des prédictions. Nous avons donc laissé un gap d'une heure entre les sets de données utilisés avec Kriging, nous avons vu de nos expériences précédentes qu'utiliser deux sets de données (c'est-à-dire des données qui remontent jusqu'à H-14 donc) est la solution optimale pour la précision de Kriging (total de 71 données d'entraînements, car évaluer avec la technique Leave One Out). Nous pouvons observer la répartition des données d'entraînements figure 5.9.

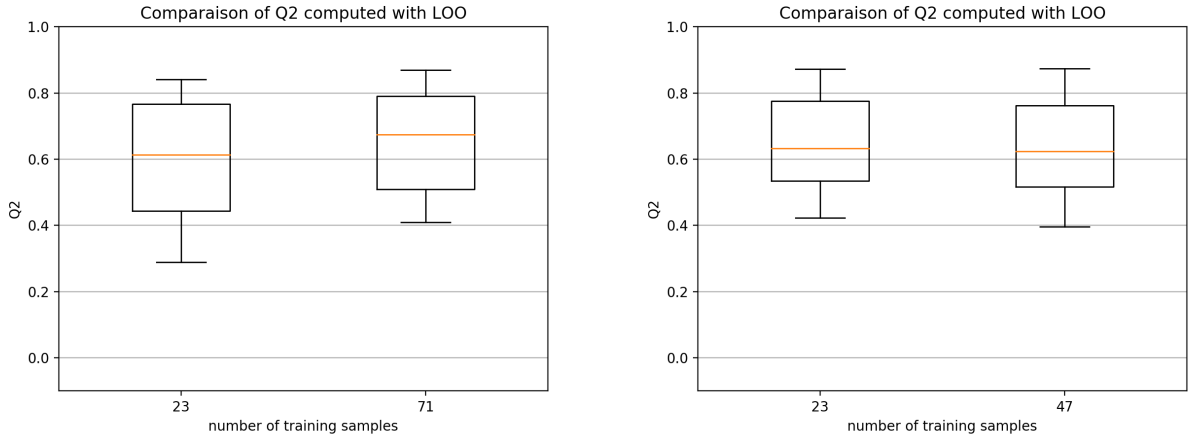
Quant à Random Forest, nous savons que la discontinuité dans les données d'entraînements ne pose pas de problème, nous ne choisirons qu'un seul set de données supplémentaire consécutif aux sets de prédiction (total de 47 données d'entraînements, car évaluer avec la technique Leave One Out).



(a) Données d'entraînements pour Kriging (b) Données d'entraînements pour RF

FIGURE 5.9 – Organisation des données d'entraînements pour tests avec sets supplémentaires.

Nous allons maintenant comparer les résultats. Pour ce faire, nous allons comparer les résultats LOO avec 23 données d'entraînements sur le set de données Arome à calculer, et les résultats LOO avec 23 données d'entraînements sur le set de données Arome à calculer plus 24 ou 48 données supplémentaires venant d'un ou deux sets de données supplémentaires Arome précédant notre calcul. Les résultats sont montrés figure 5.10.



(a) Résultats avec données supplémentaires et Kriging (b) Résultats avec données supplémentaires et RF

FIGURE 5.10 – Évolution de Q^2 en fonction du nombre de données avec données supplémentaires

Les résultats de Random Forest sont intéressants : on voit qu'il n'y a pas d'amélioration de Q^2 lorsque l'on utilise des sets supplémentaires de données. Si nous reprenons le fonctionnement de cette méthode, lors d'une prédiction un arbre retourne un résultat qui va dépendre uniquement des données d'entraînements proches de la prédiction, il n'y a pas d'apprentissage sur la durée et dans le cas même où la profondeur de l'arbre n'est pas limitée (comme ici) nous voyons même que le résultat d'une prédiction à un instant t ne dépend que des valeurs d'entraînements encadrant ce t. Il n'y a donc aucun apprentissage de la physique du modèle, on peut dire que Random Forest fait du traitement d'image sans réellement apprendre.

Contrairement à Random Forest, le modèle construit avec Kriging obtient de meilleurs résultats lors du calcul avec des sets supplémentaires de données Arome-NWC. Lorsque nous regardons de plus près le fonctionnement de Kriging nous constatons que toutes les données d'entraînements ont un poids sur la prédiction, nous concluons que Kriging est capable de capter beaucoup plus d'informations sur la physique du modèle d'où l'intérêt du couplage avec POD.

5.2.3 Analyse graphique des prédictions

Le modèle de substitution est testé dans différents cas représentant différentes conditions météorologiques. Les cas sont tous sélectionnés à partir d'une base de données de début 2018 qui présente une période de précipitation anormalement élevée. Ainsi les résultats montrés ci-dessous sont des représentations des cas les plus difficiles à prédire. Nous allons chercher quelles sont les situations où l'erreur est la plus importante.

Sur la figure 5.11, un test difficile pour le modèle de substitution avec des zones de pluies petites et éparpillées, les Q^2 sur ce test (avec 13 samples d'entraînements) sont de $Q^2(RF) = 0.47$ et $Q^2(Kriging) = 0.42$. La figure 5.11 représente les résultats avec Random Forest.

Sur la figure 5.12, un autre test difficile pour le modèle de substitution avec un front de pluie se déplaçant rapidement vers l'Est, les Q^2 sur ce test (avec 13 samples d'entraînements) sont de $Q^2(RF) = 0.41$ et $Q^2(Kriging) = 0.32$. La figure 5.11 représente les résultats avec Kriging.

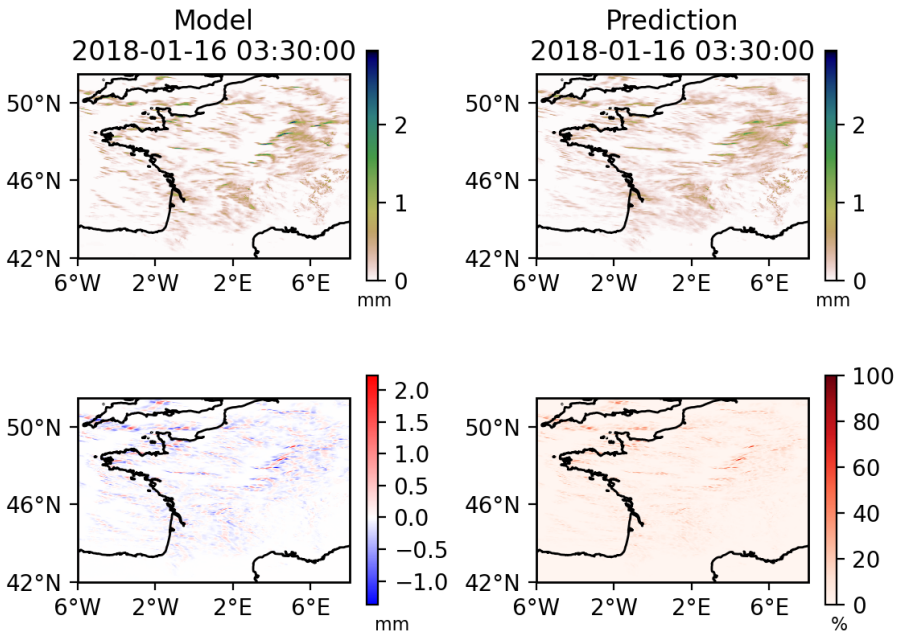


FIGURE 5.11 – Différence modèle/ prédiction test du 16/01/2018 à 3h30 en utilisant Random Forest

5.2.4 Évaluation des méthodes d'apprentissage non supervisé

Modèle de substitution non supervisé pour la prévision des lames d'eau

Comme nous l'avons vu dans le chapitre 3, une façon d'améliorer les résultats de régression dans les données qui ne sont pas linéaire est d'utiliser des modèles de substitution locaux. Or dans nos précédents calculs, nous avons vu que les fronts de pluies se déplaçant rapidement créent le plus d'erreurs, peuvent s'apparenter a une discontinuité dans les sets de données. Nous avons remarqué que Météo France classifie deux types de pluies qui ont des comportements différents :

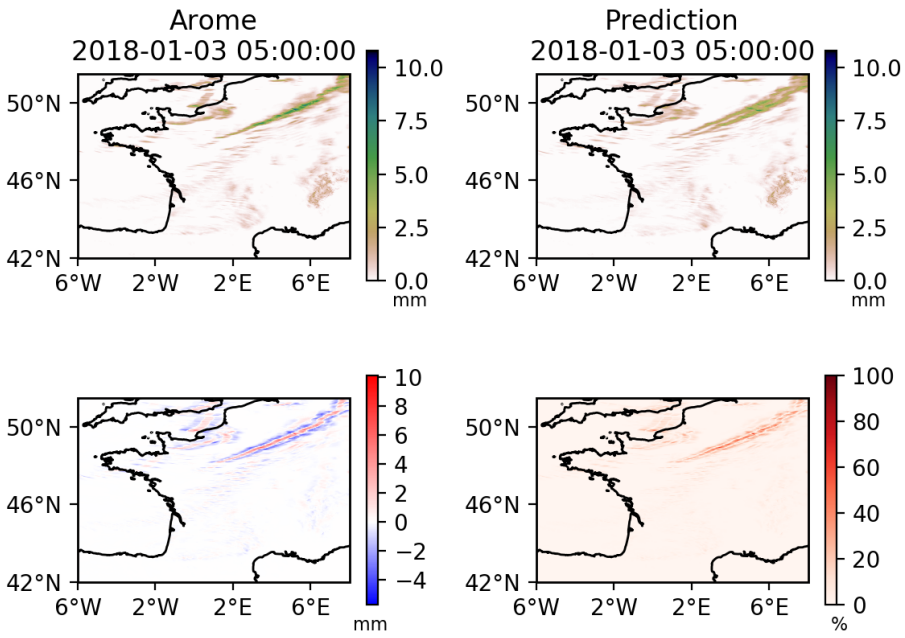


FIGURE 5.12 – Différence modèle/ prédiction test du 03/01/2018 à 5h00 en utilisant Kriging

— Les pluies convectives : Les précipitations convectives résultent de l'élévation rapide de masses d'air chargées d'humidité, par la poussée d'Archimède. Cette convection d'air humide est à l'origine de nuages de type cumuliformes avec une extension verticale pouvant dépasser les 10 km à nos latitudes. Les précipitations associées se caractérisent par : une intensité forte à très forte, une hétérogénéité spatiale, une durée souvent courte (de l'ordre de 30 minutes).
— les pluies stratifiées : dues au soulèvement lent et à grande échelle d'une masse d'air humide qui se condense uniformément. Leur nom provient des nuages associés qui sont de type stratiformes (nappes d'apparence grise et uniforme : nimbostratus, stratocumulus, stratus). Elles se manifestent dans le corps des perturbations pluvieuses associées aux fronts chauds et froids. Elles se caractérisent par : une intensité faible ou modérée (inférieure à 10 mm/h), un aspect continu, une relative homogénéité spatiale, une durée parfois importante, en cas de déplacement lent des perturbations ou bien lorsqu'elles accrochent les reliefs (blocages orographiques).

Pour plus de détails sur les différents types de pluies, le lecteur peut lire l'annexe 5.2.7 ou l'article [38]. Nous savons qu'il existe une variable permettant de séparer les types de pluies stratifiées des pluies convectives : le gradient de la température pseudo-adiabatique potentielle du thermomètre mouillé (grad(θ'_w)). Nous nous servirons donc de cette variable pour réaliser les clusters, nous nous servirons aussi d'un senseur sigmoïde pour prétraiter les données : $sigmoide(grad(\theta' w)) = \frac{1}{1+\exp^{-grad(\theta' w)}}$.

Puisque les données de grad(θ'_w) et de lames d'eau calculées par Arome-NWC ne sont pas de la même taille, nous devrons entraîner un classifieur sur les données de grad(θ'_w) pour étiqueter les données de lames d'eau. Nous utiliserons un classifieur Random Forest, et nous jouerons sur la profondeur des arbres pour lisser chaque cluster. Enfin nous pouvons aussi associer à chaque cluster un noyau (pour Kriging) adapté aux spécificités de la zone de précipitation (voir section 2.2.1) :

— Zones de pluies convectives : grad(θ'_w) élevé, sont instationnaires et non périodique par conséquent le noyau le plus adapté à leur prédiction est RationalQuadraticKernel, nous pouvons aussi ajoutés un noyau constant.
— Zones de pluies stratifiées : grad(θ'_w) faible, sont pratiquement stationnaire et anisotropiques, un noyau utilsant Matern ou RadialBasisFunction sera parfaitement adapté à la prédiction des pluies stratiformes, nous pouvons aussi ajoutés un noyau constant. A l'utilisation nous observons une légère amélioration en utilisant Matern, nous garderons donc ce noyau.

Le fait de créer des clusters crée un problème : les résultats sont totalement discontinus aux frontières des clusters, on obtient donc une mauvaise représentation de la réalité (surtout que les résultats ont vocation à apparaître dans des animations). Nous ajusterons un contour autour de chaque cluster en jouant sur le senseur sigmoïde :

$$f_1(grad(\theta' w)) = \frac{1}{1+\exp^{-grad(\theta' w)+cst1}}$$

$$f_2(grad(\theta' w)) = \frac{1}{1+\exp^{-grad(\theta' w)+cst2}}$$

Ainsi nous faisons le calcul avec Batman dans le cluster avec contour puis nous reconstruisons les résultats en nous appuyant sur les longitudes/ latitudes correspondantes dans le cluster sans contour. Vous pouvez observer les figures qui nous ont permis de développer chaque étape de la méthode dans la figure 5.13.

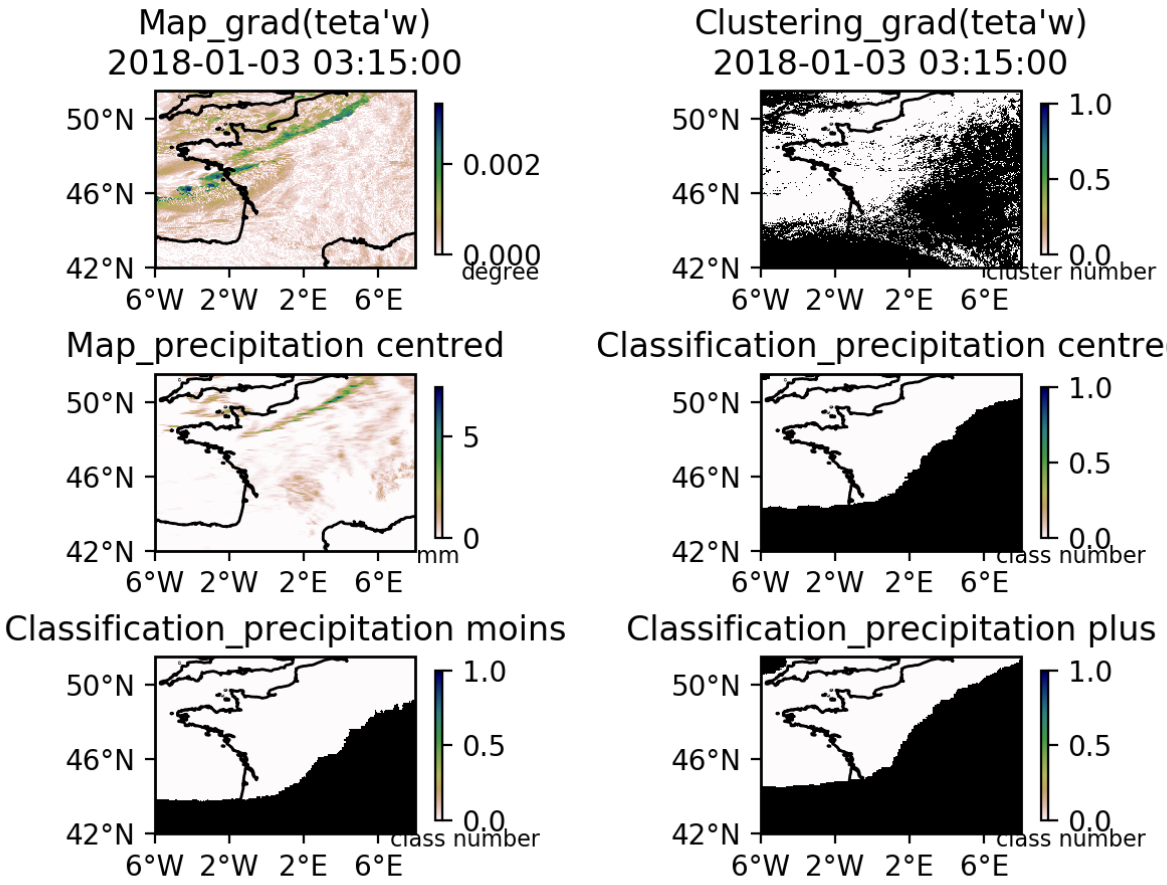


FIGURE 5.13 – Visualisation du clustering/ classification lors de l'utilisation de la méthode non supervisé. Classification centred correspond à la classification après avoir utilisé le sigmoïde centré, classification moins correspond à la classification pour avoir le contour du cluster de pluies convectives, classification plus correspond à la classification pour avoir le contour du cluster de pluies stratifiées

Résultat en utilisant RandomForest

Le comportement de Random Forest de traitement d'image sans influence des points éloignés en espace et en temps se confirme dans ce test. Il n'y a aucune différence entre les résultats de Random Forest lors du calcul avec clustering et classification par rapport aux calculs précédents où Random Forest était utilisé seul (les résultats sont identiques à 10^{-6} près, le caractère aléatoire de la méthode

crée des différences insignifiantes). Nous pouvons conclure que l'utilisation de clustering n'est pas adaptée à la méthode de régression Random Forest.

Résultats en utilisant Kriging

Pour tester cette méthode, nous procéderons en deux étapes : premièrement nous comparerons la méthode supervisée et non supervisée avec des tests suffisant pour déterminer la meilleure. Ensuite nous évaluerons sur un test plus poussé qui nécessitera plus de ressource informatique la meilleure des deux méthodes (kriging seul ou kriging dans cluster) face à Random Forest (dans sa variante ExtraTrees Regressor).

Nous allons évaluer la qualité de prédiction de la nouvelle méthode face à Kriging seul. Les résultats sur les 10 tests utilisés précédemment pour la méthode Kriging et la nouvelle méthode (Kriging dans cluster) sont montrés dans le tableau 5.1

	Kriging seul	Kriging dans cluster	différence
Moyenne Q^2	0.607	0.638	+ 1.44%
Mediane Q^2	0.587	0.589	+ 0.30%
Premier Quartile Q^2	0.440	0.449	+ 2.04%
Troisième Quartile Q^2	0.787	0.817	+ 3.8%
Ecart type	0.199	0.190	-3.06%

TABLE 5.1 – Analyse statistique de la qualité des résultats des méthodes Kriging et Kriging dans cluster

Nous remarquons une amélioration globale des résultats. Ce qui est intéressant est l'amélioration importante du troisième quartile et de l'écart type. Nous avons utilisé en tout 110 évaluations de Q^2, si l'on considère que Q^2 est une variable stochastique, on peut calculer l'intervalle de confiance à 95%. L'espérance est inconnue mais l'on connaît la variance ($\sigma = \sqrt{V} \Leftrightarrow V = \sigma^2$). On sait que :

Intervalle de confiance : $\bar{X} \pm z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}$

Pour un intervalle de confiance à 95% : $z_{1-\frac{\alpha}{2}} = 1.960$, l"intervalle de confiance de la moyenne de Q^2 est présenté dans le tableau 5.2.

Méthode	P(Q^2) 95% min	P(Q^2) 95% max
Moyenne Kriging seul	0.601	0.611
Moyenne Kriging dans cluster	0.633	0.642

TABLE 5.2 – Intervalle de confiance de la moyenne de Q^2

Nous voyons que la méthode de Kriging dans les clusters obtient des meilleurs résultats mais la différence reste faible. Nous avons dû vérifier que les intervalles de confiances à 95% ne se chevauchent pas et ainsi nous assurer que la différence est significative pour conclure que la méthode de Kriging dans cluster est meilleure que Kriging seul.

5.2.5 Comparaison de la qualité du modèle utilisant random forest et du modèle utilisant la méthode non supervisée

Protocole de test

Le but du stage est de trouver le meilleur modèle de substitution possible en termes de qualité et de coût de calcul pour Météo France. Nous avons vu que la méthode utilisant EXtraTrees Regressor était meilleure en termes de qualité et de coût de calcul que la méthode utilisant Kriging seul (voir section 5.2.1). Et nous avons vu que la méthode utilisant le clustering sur grad($\theta' w$) avant d'utiliser Kriging dans chacun des clusters (zone de pluies stratifiée et convective) était meilleure que la méthode utilisant Kriging seul (voir section 5.2.4). Mais nous n'avons pas encore conclu sur la meilleure méthode de manière définitive. De plus, les tests ont été réalisés de manière à pouvoir être certains que le Q^2

moyen d'une méthode n'était pas dans l'intervalle de confiance de l'autre, mais pas pour donner une estimation générale statistique de Q^2.

Nous allons donc revoir la méthode de tests. Pour ce faire, nous allons tester non pas sur 10 sets de données Arome, mais sur tous les sets disponibles du mois de Janvier 2018, un mois très pluvieux et difficile à prédire. La France a connu en janvier 2018 une succession de passages perturbés très actifs avec plusieurs épisodes tempétueux dans une ambiance exceptionnellement douce. Le mois de janvier a débuté avec le passage de deux tempêtes successives, Carmen le 1^{er} et Eleanor du 2 au 4. Puis, de forts coups de vent ont concerné les régions méditerranéennes notamment les 17 et 21 janvier. Au passage de la tempête Fiona le 17 janvier, les vents ont été extrêmement violents sur la Corse, avec jusqu'à 225 km/h au cap Corse (Haute-Corse), record absolu. La tempête David a concerné l'extrême nord de la France le 18 janvier.

La tempête Eleanor, qui a touché 25% du territoire, se classe au 19^{eme} rang des tempêtes les plus sévères depuis 1980.

Nous avons 709 sets de données Arome, contenant chacun 24 samples, nous utiliserons 13 samples pour l'entraînement du modèle de substitution et nous évaluerons le modèle sur les 11 restantes. Ce qui représente un total de 7799 samples évalués. Ce qui est largement suffisant pour donner une estimation générale de la qualité du modèle sur 13 samples d'entraînements. Bien sûr, Q^2 sera sous-estimé, car Météo France utilisera les 24 samples pour l'entraînement du modèle de substitution, mais dans ce cas, nous ne pouvons pas évaluer la qualité du modèle.

Analyse des résultats

Premièrement concernant le temps de calcul en temps CPU pour 13 samples :

— ExtraTreesRegressor : 0m43.758s

— Unsupervised + supervised : 2m31.614s

La méthode avec ExtraTrees est beaucoup plus efficace en terme de coût de calcul que la méthode avec kriging dans cluster.

Deuxièmement on peut observer le Q^2 pour 13 samples dans le tableau 5.3.

	ExtraTrees	Kriging dans cluster	différence
Moyenne Q^2	0.621	0.523	+ 18.7%
Médiane Q^2	0.657	0.576	+ 14.0%
Premier Quartile Q^2	0.534	0.421	+ 26.7%
Troisième Quartile Q^2	0.738	0.683	+ 8.05%
Écart-type	0.165	0.277	- 40.4%

TABLE 5.3 – Comparaison qualité des prédictions des méthodes Random Forest et kriging dans cluster

Ou sous une autre forme, voir figure 5.14.

5.2.6 Extrapolation du Q2 moyen avec 24 samples d'entraînements

Comme nous l'avons vu précédemment, il est impossible de calculer le Q^2 explicitement avec tous les samples utilisés dans l'entraînement du modèle de substitution. Nous savons que le Q^2 d'un modèle de substitution en fonction du nombre de samples d'entraînements utilisés suit une fonction asymptotique (voir section 5.1.2) de la forme :

$$Q^2(samplesnumber) = A + \frac{B}{samplesnumber}$$

Avec A et B des constantes inconnus, appartenant à $\mathbb{R}$. Pour extrapoler correctement la fonction précédente, la solution la plus simple est de la transformer de sorte qu'elle devienne linéaire. Pour ce faire, il suffit de multiplier la fonction par le nombre de samples :

$$Q^2(samplesnumber) * samplesnumber = A * samplesnumber + B$$

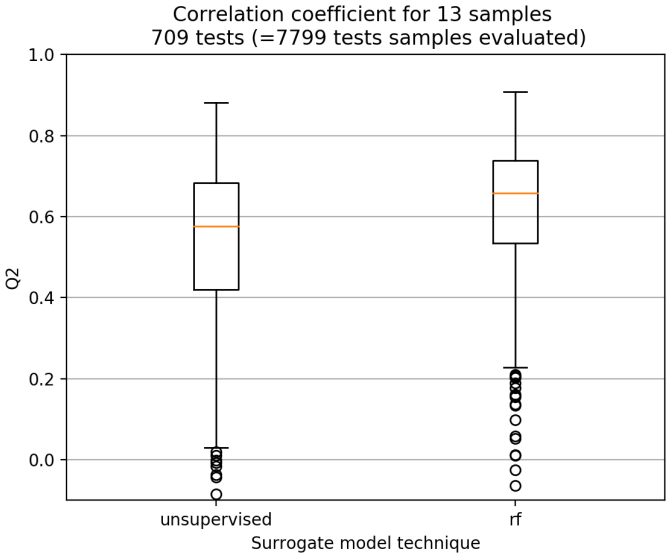


FIGURE 5.14 – Comparaison de Q^2 pour le modèle utilisant random forest et pour le modèle utilisant la méthode non supervisée

Puisque nous connaissons la valeur de Q^2 moyen pour 5, 7, 9 et 13 samples, nous pouvons maintenant utiliser une régression linéaire, extrapoler la valeur du Q^2 moyen pour 24 samples d'entraînements, en divisant les résultats en chaque point par le nombre de samples. L'extrapolation est montrée pour toutes les méthodes testées dans ce rapport figure 5.15

5.2.7 Meilleur modèle réduit pour la prédiction des lames d'eau

Malgré tout le travail de développement effectué pour la méthode de kriging dans cluster machine learning, il faut reconnaître que cette méthode est moins bonne que celle utilisant ExtraTreesRegressor, pour le cas de prévision des pluies. Même si nous avons réussi à améliorer les performances de Kriging, ce ne sera pas suffisant dans notre cas.

Nous proposerons donc à Météo France l'utilisation de BATMAN avec EXtraTrees Regressor pour la prédiction des lames d'eau immédiates. Les avantages de la méthode sont :

— Réduction du coût de calcul : grâce au multiprocess le temps réel est largement inférieur à la minute

— Qualité de résultats satisfaisante. Nous n'avons pas réussi à atteindre le critère de convergence de 0.8, mais nous nous en rapprochons. Puisque le mois de Janvier 2018 (mois d'où proviennent les tests) est un mois si actif au niveau des lames d'eau, nous pouvons considérer que dans un contexte plus classique, le critère de convergence est satisfait.

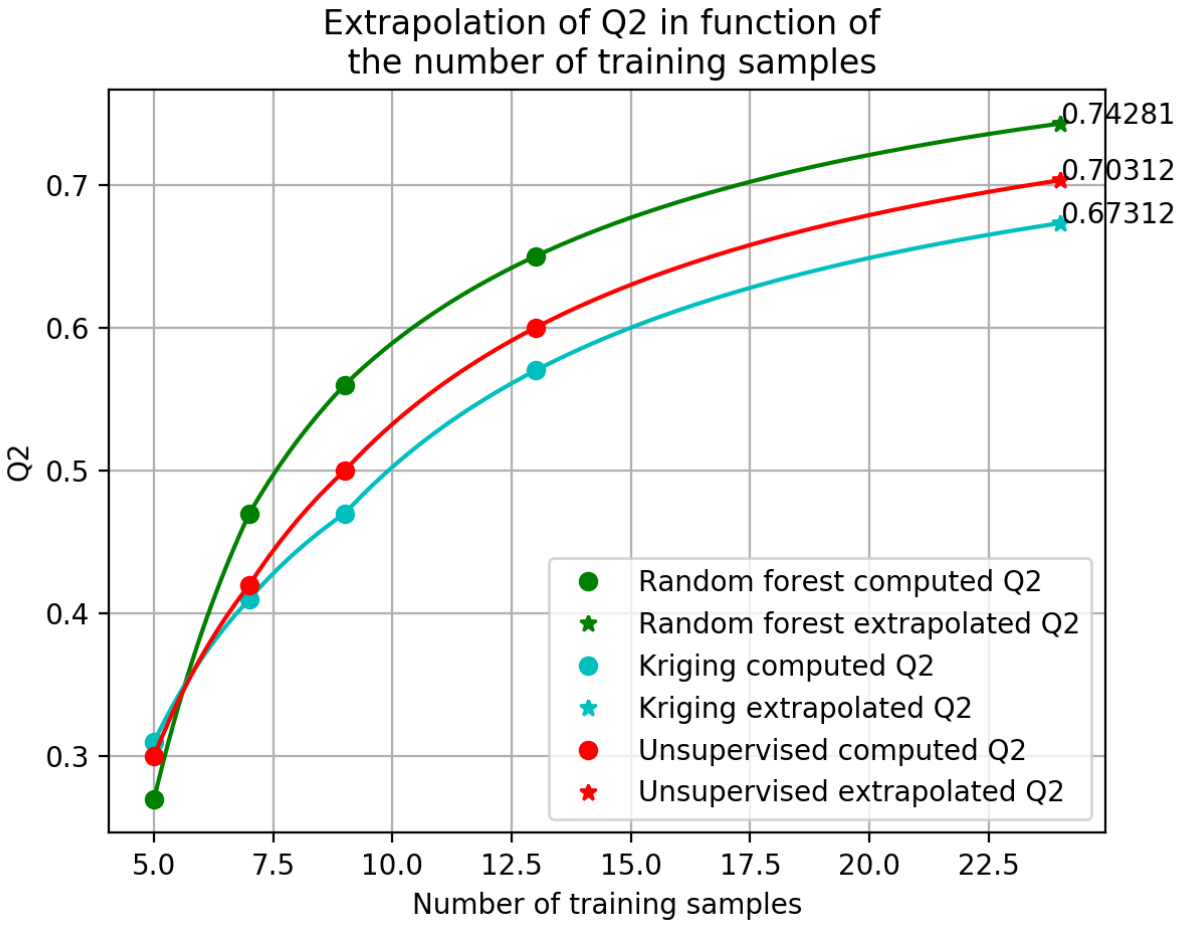


FIGURE 5.15 – Extrapolation du Q^2 de toutes les méthodes pour 24 samples

Conclusion

Les modèles de substitution sont des solutions applicables à la prévision immédiate des lames d'eau. Leurs performances principales sont :

— La conservation d'un sens physique
— La réduction du temps de calcul
— La possibilité de créer des prédictions en tous points (interpolation)

Nous aurons étudié deux méthodes de conception de modèle de substitution. La première réalise une interpolation dans un espace réduit (POD), l'interpolation est réalisée par des méthodes géostatistique avec Kriging ou d'apprentissage automatique avec les méthodes ensemblistes (Random Forest, Extra Trees, AdaBoost).

Le choix de la méthode d'interpolation dépend du problème à substituer et des contraintes de temps de calcul. L'avantage de la technique de kriging est sa capacité à transposer un sens physique au modèle de substitution, mais nécessite un temps de calcul important (l'utilisation de la POD sert aussi à réduire la taille des données d'entraînements et donc raccourcir le temps de calcul). Tandis que les méthodes ensemblistes effectuent un traitement d'image (plus efficace en termes de temps de calcul), c'est pourquoi il est nécessaire de les utiliser dans l'espace POD pour garder un sens physique.

Enfin nous avons développé une méthode de calcul basée sur la division de l'espace (non temporelle) en zones de pluie convective et stratifiée, ces types de pluies n'ayant pas le même comportement. Nous espérons ainsi gommer un défaut de Kriging qui est de prendre en compte en tous points, les valeurs d'autres points éloignés qui n'ont pas le même comportement. Ainsi même si nous rallongeons le temps de calcul, nous conservons le plus de sens physique possible. La division des zones de pluies stratifiées ou convectives s'appuie sur un clustering sur la variable $grad(\theta' w)$, nous devons aussi utiliser un algorithme de classification pour passer d'un clustering sur $grad(\theta' w)$ à des zones de lame d'eau car les données ne sont pas stockées sur les mêmes points (latitudes/ longitudes).

La méthode présentant la meilleure qualité de prédiction, ainsi que le temps de calcul le plus court, est la méthode utilisant la variante de Random Forest : Extra Trees dans l'espace POD. Le critère $Q^2 = 0.8$ convergence est approché (sur nos tests $Q^2_{moyen} \approx 0.74$), pour un coût de calcul d'environ 30 secondes. Vient ensuite la méthode utilisant le groupement qui a une qualité de prédictoion plus faible (sur nos tests $Q^2_{moyen} \approx 0.70$) et un coût de calcul d'environ 2 minutes 30 secondes. La méthode testée qui présente les plus mauvais résultats est la méthode utilisant Kriging dans l'espace POD avec une qualité de prédiction peu satisfaisante (sur nos tests $Q^2_{moyen} \approx 0.67$) pour un temps de calcul d'environ 1 minute 40 secondes.

La méthode utilisant un groupement obtient des résultats moins bons que la méthode d'interpolation dans la base POD. Cependant, nous avons vu dans la littérature que cette méthode permet généralement une amélioration des qualités de prédiction. Du travail reste à faire sur ce type de modèle de substitution, nous suspectons une erreur induite par les zones de pluies convectives/ stratifiées. En effet, lors de la définition des groupes nous associons à chaque point une valeur pluie convective ou stratifiée en fonction de la présence majoritaire du type de pluie en ce point dans les données Arome (24 données de temps sur 6 heures). Malheureusement, comme les frontières de pluies convectives / stratifiés bougent au cours du temps, le groupe attribué à certains points pour certains pas de temps n'est pas adapté. L'idéal serait de trouver le moyen de réaliser l'interpolation dans des groupes "mouvants".

Une autre opportunité est la création d'un réseau de neurones. Ce travail est réalisé en ce moment à l'IRT Saint-Exupéry dans le cadre du programme Deep4Cast. Les résultats sont pour le moment plutôt décevants avec une tendance au lissage des zones de précipitations très prononcé.

Enfin, toutes ces techniques étudiées ne représentent qu'un type de modèle de substitution : les

modèles non intrusifs. Nous pourrions aussi tester des modèles basés sur la projection des équations gouvernantes ou la minimisation des résidus, si nous avions accès à l'algorithme Arome. Malheureusement l'accès à Arome n'a pas été autorisé par Météo-France pour ce stage.

Bibliographie

[1] D. Krige. *A Statistical Approach to Some Mine Valuation and Allied Problems on the Witwatersrand*. Ph.D Thesis. University of Witwatersrand, Johannesburg, 1951.

[2] L. Auger, O.Dupont, S. Hagelin, P. Brousseau, and P. Brovelli. Arome – nwc : a new nowcasting tool based on an operationalmesoscale forecasting system. *Quarterly Journal of the Royal Meteorological Society*, 141 :1603–1611, 2015.

[3] J. Holmes, C. Mattingly, and W.Wittenberg. Low-dimensional models of coherent structures in turbulence. *Physics Reports*, 1997.

[4] P.T. Roy, S.Ricci, R. Dupuis, R. Campet, J.-C. Jouhaud, and C. Fournier. Batman : Statistical analysis for expensive computer codes made easy. *The Journal of Open Source Software*, 2018.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn : Machine learning in Python. *Journal of Machine Learning Research*, 12 :2825–2830, 2011.

[6] AVBP AVBP. Code : www. cerfacs. fr/cfd/avbp_code. php and www. cerfacs. fr/cfd. *CFDPublications. html*, 2013.

[7] Markus Baum. Ntmix/chemkim-release2. *User's Guide*, 1, 1995.

[8] Julien Weiss. *A Tutorial on the Proper Orthogonal Decomposition*. AIAA, 2019.

[9] F. D'Andrea and R. Vautard. Extratropical low-frequency variability as a low dimensional problem. *Quarterly Journal of the Royal Meteorological Society*, 127 :1357–1375, 2001.

[10] J. M. Beckers and M.Rixen. Eof calculations and data filling from incomplete oceanographic data sets. *American Meteorological Society*, 20 :1839–1856, 2003.

[11] N. J. Holbrook and N. L. Bindoff. A statistically efficient mapping technique for four-dimensional ocean temperature data. *Journal of atmospheric and oceanic technology*, 17 :831–846, 2000.

[12] Naty Cabrera-Gutiérrez, Hadrien Godé, and J.-C Jouhaud. Surrogate models for rainfall nowcasting, 06 2020.

[13] Kaliappan Geetha, Sivasubramanian. Cervical cancer identification with synthetic minority oversampling technique and pca analysis using random forest classifier. *Journal of Medical Systems*, 43 :286, 07 2019.

[14] K. Lange, T. Hansen, J. L. Fernández-Martínez, J. Frydendall, and K. Mosegaard. Kriging in high dimensional attribute space using principal component analysis. In *Kriging in high dimensional attribute space using Principal component analysis*, 10 2010.

[15] L. Breiman. Random forests. *Machine Learning*, 45 :5–32, 2001.

[16] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63 :3–42, 04 2006.

[17] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, page 148–156, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[18] B.V. Srinivasan, Ramani Duraiswami, and R. Murtugudde. Efficient kriging for real-time spatiotemporal interpolation. *Proceedings of the 20 th Conference on Probability and Statistics in the Atmospheric Sciences*, pages 228 – 235, 2010/// 2010.

[19] D. Courault and P. Monestiez. Spatial interpolation of air temperature according to atmospheric circulation patterns in southeast france. *International Journal of Climatology*, 19 :365–378, 1999.

[20] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 11 2014.

[21] S. Adam S. Bernard, L. Heutte. Influence of hyperparameters on random forest accuracy. *Lecture Notes in Computer Science*, 5519, 2009.

[22] C. Audet, K. Dang, and D. Orban. Optimization of algorithms with opal. *Math. Prog. Comp.*, 2012.

[23] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598) :671–680, 1983.

[24] Rainer Storn and Kenneth Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11 :341–359, 01 1997.

[25] François Robert. Pierre bailly et christine carrère, statistiques descriptives. l'économie et les chiffres. *Lectures*, 2015.

[26] G. Biau, E. Zorita, H. von Storch, and Hans Wackernagel. Estimation of precipitation by kriging in the eof space of the sea level pressure field. *Journal of climate*, 12 :1070–1085, 1999.

[27] G. Aversano, A. Bellemans, Z. Li, A. Coussement, O. Gicquel, and A. Parente. Application of reduced-order models based on pca & kriging for the development of digital twins of reacting flow applications. *Computers and Chemical Engineering*, 121 :422–441, 2019.

[28] R. Dupuis, J.-C. Jouhaud, and P. Sagaut. Surrogate modeling of aerodynamic simulations for multiple operating conditions using machine learning. *American Institute of Aeronautics and Astronautics*, 56 :3622–3635, 2018.

[29] C. Azencott. *Introduction au Machine Learning*. Dunod, 2019.

[30] Hans-Hermann Bock. Clustering methods : a history of k-means algorithms. In *Selected contributions in data analysis and classification*, pages 161–172. Springer, 2007.

[31] David Amsallem, Matthew Zahr, and Charbel Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92 :891–916, 12 2012.

[32] J. H. Halton. Algorithm 247 : Radical-inverse quasi-random point sequence. *Commun. ACM*, 7(12) :701–702, December 1964.

[33] Paul Bratley and Bennett L. Fox. Algorithm 659 : Implementing sobol's quasirandom sequence generator. *ACM Trans. Math. Softw.*, 14(1) :88–100, March 1988.

[34] R L Iman, J M Davenport, and D K Zeigler. Latin hypercube sampling (program user's guide). [lhc, in fortran]. 1 1980.

[35] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, 1992.

[36] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24(2) :619–644, 2002.

[37] Gaël Varoquaux and O Grisel. Joblib : running python function as pipeline jobs. *packages. python. org/joblib*, 2009.

[38] Robert F. Adler and Andrew J. Negri. A Satellite Infrared Technique to Estimate Tropical Convective and Stratiform Rainfall. *Journal of Applied Meteorology*, 27(1) :30–51, 01 1988.

Annexes

Les boîtes à moustaches

Le box-plot que l'on appelle aussi boîte à moustache pour sa forme originale est un graphique qui permet de résumer une variable aléatoire de manière simple et visuelle, d'identifier les valeurs extrêmes et de comprendre la répartition des observations. Nous proposons quelques détails sur ce graphique afin de l'utiliser simplement.

Un box-plot est un graphique simple composé d'un rectangle duquel deux droites sortent afin de représenter certains éléments des données.

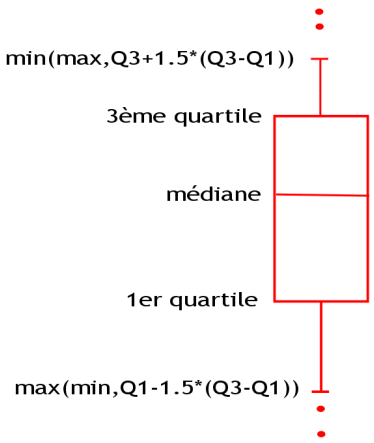


FIGURE 5.16 – Illustration boxplot

— La valeur centrale du graphique est la médiane (il existe autant de valeurs supérieures qu'inférieures à cette valeur dans l'échantillon).
— Les bords du rectangle sont les quartiles (Pour le bord inférieur, un quart des observations ont des valeurs plus petites et trois quarts ont des valeurs plus grandes, le bord supérieur suit le même raisonnement).
— Les extrémités des moustaches sont calculées en utilisant 1.5 fois l'espace interquartile (la distance entre le 1^{er} et le 3^{eme} quartile).

On peut remarquer que 50% des observations se trouvent à l'intérieur de la boîte.

Les valeurs à l'extérieur des moustaches sont représentées par des points. On ne peut pas dire que si une observation est à l'extérieur des moustaches alors elle est une valeur aberrante. Par contre, cela indique qu'il faut étudier plus en détail cette observation.

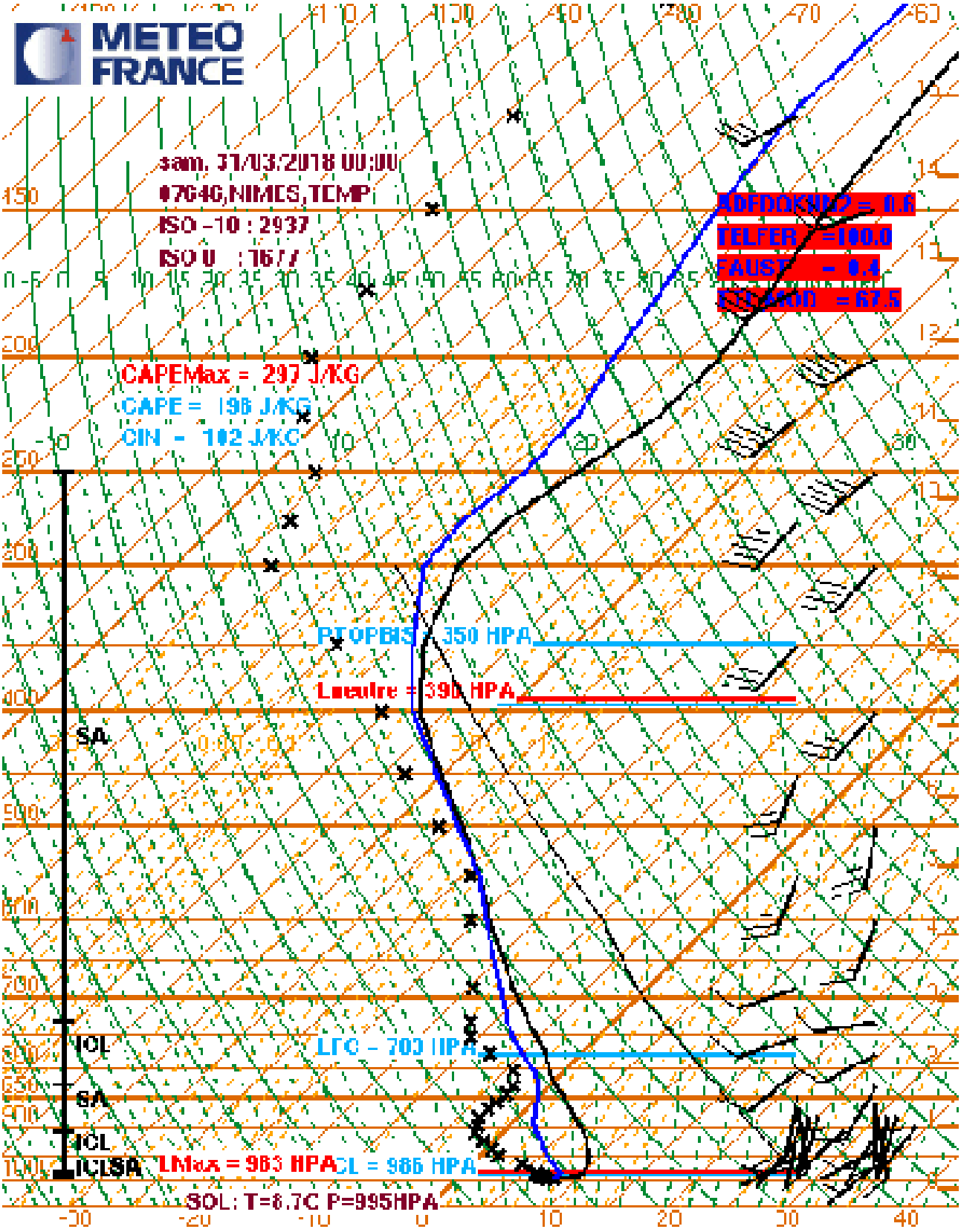
Les différents types de pluies

La pluie vient de l'évaporation de l'eau, dans un premier temps la vapeur d'eau se mélange à l'air et monte en altitude. Lors de ce mouvement, la parcelle de gaz subit une transformation adiabatique [2], si son ascension se poursuit, la pression diminuant en altitude la parcelle arrivera à un point de

2. En thermodynamique, une transformation est dite adiabatique si elle est effectuée sans qu'aucun transfert thermique n'intervienne entre le système étudié et le milieu extérieur

condensation (à une température t_c, et une pression p_c), on dit aussi que la parcelle devient saturée. La transformation de la parcelle suit une adiabatique (ou adiabatique sèche) tant qu'elle n'est pas saturée, mais lorsque la parcelle est saturée la transformation devient pseudo-adiabatique (ou adiabatique saturé).

Un outil utilisé par les météorologues et qui permet de suivre les parcelles est l'émagramme. L'émagramme permet de lier la pression atmosphérique, la température, la vitesse et la direction du vent, et enfin l'humidité : ces cinq grandeurs, en effet, déterminent les facteurs de stabilité ou d'instabilité des couches atmosphériques successives, leurs mouvements horizontaux et verticaux éventuels ainsi que les possibilités d'évaporation, de condensation ou de précipitation qui en résultent. Un émagramme n'est rien de plus qu'un graphique montrant l'état de l'atmosphère pour un lieu donné et à différents niveaux, ci-dessous résultat du sondage effectué le 31/03/2018 à Nîmes par Météo-France :



En abscisse : pour que la courbe d'état soit sensiblement verticale, la température est représentée par des lignes isothermes [3] (marron) inclinées à 45° vers la droite elle est exprimée en °C.

En ordonnée : la pression atmosphérique exprimée en hectopascals (hPa) est représentée par des lignes horizontales isobares [4] (marron).

En ordonnée : l'altitude est elle aussi représentée sur l'émagramme. Elle s'exprime en kilomètres

3. isotherme = température constante
4. isobare = pression constante

ou en pieds.

Les adiabatiques sèches sont représentées sur un émagramme par des courbes continues en vert, inclinées vers la gauche et creusées. Une particule d'air sèche (ne contenant pas d'eau sous forme liquide) qui s'élève en altitude voit sa température suivre la courbe sur laquelle elle est positionnée au départ. Ce gradient de température vaut à peu près 1° C/100 m du sol à la tropopause [5].

Les pseudo-adiabatiques saturées sont représentées sur un émagramme par des courbes discontinues en vert qui sont souvent plus pentues que les adiabatiques sèches. Une particule d'air qui s'élève en altitude avec condensation voit sa température suivre la courbe sur laquelle elle est positionnée au point où apparaît la condensation de la vapeur d'eau. Sa température diminue à un taux de 5° C/1 000 m . En fait ce gradient est plutôt entre 1° C/1000 m et 8° C/1 000 m car sa valeur varie avec l'altitude et la température ambiante.

Les courbes du ratio de mélange représentent l'évolution dynamique de l'humidité d'une masse d'air dans un mouvement vertical, des courbes de rapport de mélange sont ajoutées sous la forme de lignes orange pointillées. Le rapport de mélange est la quantité de vapeur d'eau contenue dans l'air. Il est mesuré en grammes d'eau par kilogramme d'air sec et sa pente est de 2° C/1 000 m. Par exemple si l'humidité d'une bulle d'air au sol est de 10° C (température du point de rosée), alors à une altitude de 1 000 m, elle sera : 10° C - 2° C = 8° C. Si l'on connaît la température et le point de rosée initiaux, les courbes du rapport de mélange peuvent être utilisées pour déterminer l'altitude à laquelle se produit la condensation, c'est-à-dire les bases des nuages.

Une parcelle saturée se caractérise par la variable $grad(\theta'_w)$, qui correspond au gradient du point d'intersection de la pseudo-adiabatique suivie par la parcelle avec l'absice (L'axe des absices correspond à la température). θ'_w aussi appelé la température pseudo-adiabatique potentielle du thermomètre mouillé de la parcelle, permet de distinguer deux types de pluies :

- les pluies convectives (grad(θ'_w) élevé) : Les précipitations convectives résultent de l'élévation rapide de masses d'air chargées d'humidité, par la poussée d'Archimède. Cette convection d'air humide est à l'origine de nuages de type cumuliformes avec une extension verticale pouvant dépasser les 10 km à nos latitudes. Les précipitations associées se caractérisent par : une intensité forte à très forte, une hétérogénéité spatiale, une durée souvent courte (de l'ordre de 30 minutes).
- les pluies stratifiées (grad(θ'_w) faible) : dues au soulèvement lent et à grande échelle d'une masse d'air humide qui se condense uniformément. Leur nom provient des nuages associés qui sont de type stratiformes (nappes d'apparence grise et uniforme : nimbostratus, stratocumulus, stratus). Elles se manifestent dans le corps des perturbations pluvieuses associées aux fronts chauds et froids. Elles se caractérisent par : une intensité faible ou modérée (inférieure à 10 mm/h), un aspect continu, une relative homogénéité spatiale, une durée parfois importante, en cas de déplacement lent des perturbations ou bien lorsqu'elles accrochent les reliefs (blocages orographiques).

Le comportement de ces pluies étant différents il sera étudié dans ce rapport une méthode qui permet de classer les pluies puis de résoudre un problème de régression pour chacun de ces types de pluies. Le comportement de l'un ne venant pas perturber les données d'entraînements de l'autre.

5. La tropopause est une zone de l'atmosphère terrestre qui fait la transition entre la troposphère (au-dessous) et la stratosphère (au-dessus). Elle se situe à une altitude d'environ 11km en France