Stage d'assistant ingénieur : « Recherche de la Cordelière »
CI - 2019

ENSTA Bretagne

2 rue François Verny
29806 Brest Cedex 9
France

Olivier Laurendin
olivier.laurendin@ensta-bretagne.org

Department of Autonomous Control and Systems Engineering (ACSE), university of Sheffield

Mappin Street
Sheffield, S1 3JD
United Kingdom

Lyudmila. S. Mihaylova,
l.s.mihaylova@sheffield.ac.uk

Remerciements

J'aimerais remercier mes deux encadrants, professeur Luc Jaulin du Lab-STICC de l'ENSTA Bretagne ainsi que professeur Lyudmila S. Mihaylova du département ACSE de l'université de Sheffield pour leur soutien pendant le stage, ainsi que les services techniques de l'ACSE pour leur efficacité. Un grand merci à tous les étudiants-chercheurs, doctorants et post-doctorants de l'ACSE qui m'ont accueilli à bras ouverts pendant ces trois mois et dont les témoignages m'ont aiguillé quant à mes choix de carrière.

I would like to thank my two supervisors, professor Luc Jaulin from the Lab-STICC department at the ENSTA Bretagne, and professor Lyudmila S. Mihaylova from the ACSE department of the university of Sheffield for their guidance and support throughout the internship, and the technical team of the ACSE department for their efficiency. I would also like to thank the doctoral students and post-doctoral researchers I met during this internship for their heart-warming welcome and whose stories helped me understand better the academic world.

Résumé

Ce rapport présente les résultats de mon stage qui s'est déroulé pendant l'été 2018 à l'université de Sheffield. Le but de ce stage était la constitution d'un programme de krigeage de données magnétométriques recueillies par un robot bateau autonome "Boatbot" en Juillet 2018 dans la baie de Brest dans le but d'identifier la zone du naufrage de "La Cordelière". Le programme a été codé en python pour des questions de facilité de programmation et d'utilisation par un utilisateur tiers. Après une période de recherche autour de l'algorithme de krigeage et d'état de l'art des programmes déjà existants, un programme python a été développé selon un cahier des charges constitué avec mon encadrante, Dr. L. Mihaylova, professeure à l'université de Sheffield du département ACSE (Autonomous Control and Systems Engineering). Une étude préliminaire des résultats du programme développé sera aussi présentée.

Abstract

This paper aims at showing the results of my 3-month internship at the University of Sheffield. This internship aimed at programming a kriging algorithm to be applied to magnetometric data gathered by an autonomous boat "Boatbot" in July 2018 near Brest, France, in order to narrow down the wrecking site of the ship "La Cordelière". This program was coded in the python programming language for its programming and use simplicity. After a search time about the kriging algorithm and a state of the art of existing algorithms, a python program was developped under the guidance of Dr L Mihaylova, professor at the University of Sheffield in the Autonomous Control and Systems Engineering (ACSE) department. A study of the results of the program will further be presented.

Table des matières

Introduction

Ce projet porte surtout sur l'implémentation d'un algorithme de krigeage pour l'estimation de données magnétométriques en une grille de points régulière à partir d'un ensemble de données magnétométriques recueillies par un magnétomètre relié à un bateau autonome. L'utilisation du krigeage dans ce contexte est relativement nouvelle dans la mesure où celui-ci est initialement tiré du domaine des géostatistiques, et nécessitera quelques ajustements à notre problème.

Après une présentation générale du thème du stage et du système de recueillement des données mis en place par l'équipe du Lab-STICC, nous nous intéresserons aux lois physiques de la magnétométrie et aux incertitudes de mesure et de calcul auxquelles nous avons affaire. Nous décririons par la suite dans le détail les principes mathématiques et les étapes de paramétrage d'un algorithme de krigeage puis l'implémentation en python réalisée ainsi que les résultats expérimentaux.

1. Présentation du thème du stage

1.1.Présentation de la Cordelière

Le Marie-la-Cordelière ou plus simplement « La Cordelière » est un bateau breton construit en 1487 à la demande de Francis II, duc de Bretagne. It s'agissait du plus grand navire de la flotte bretonne avec près de 200 canons à bord. Il participa le 10 Août 1512 à la bataille de Saint-Mathieu contre les flottes anglaises au large de Brest. It coula en emmenant avec lui le navire anglais « The Regent » dans une grande explosion en mettant le feu à ses propres poudres sous l'ordre de Hervé de Portzmoguer.

1.2.Présentation du problème

Présentation du système de récupération des données magnétométriques

Une première session de collecte de données magnétométriques a été effectuée en juillet dernier par une équipe composée de professeurs et d'élèves ingénieurs de l'ENSTA Bretagne ainsi que d'archéologues sous-marins du **Glossaires** des termes techniques. L'objectif était de créer un bateau autonome capable de tracter un magnétomètre selon des droites prédéfinies. L'idée étant de rendre ce robot utilisable par des non-roboticiens et de démontrer l'utilité de l'automatisation des taches de recherches d'épaves sur de grandes étendues. Les données utilisées dans ce rapport proviennent directement des données recueillies par cette session.
Le matériel de recueillement des données a été mis en place selon le schéma suivant :

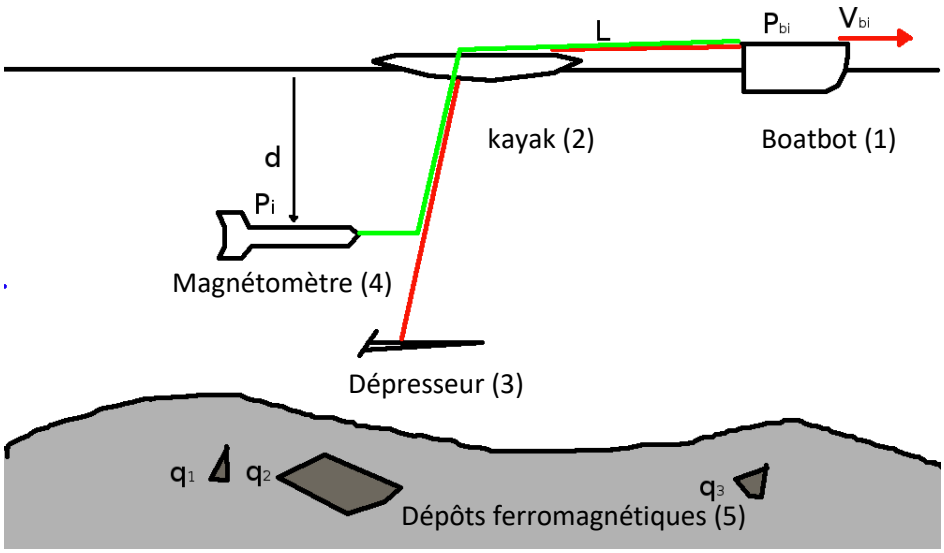


Figure 1 : Schéma du système de récupération des données

Le zodiac autonome baptisé **Boatbot** (Boatbot, s.d.) suit alors des chemins préétablis appelés “briques” du fait de leur forme rectangulaire de 1km par 10m. Le bateau va effectuer 10 allers-retours sépares de 1m sur cette brique. Le bateau autonome tracte un dépresseur par le biais d’un kayak. Le magnétomètre est quant à lui relié assez lâchement par son propre câble de communication.
Le dépresseur a pour fonction de conserver une profondeur constante. Ce faisant, il exerce une tension sur le câble le reliant à la surface. Le kayak sert justement à compenser la tension exercée sur le câble par le dépresseur tout en conservant le magnétomètre et le dépresseur éloignés du bateau autonome. Le but étant de réduire au maximum les interférences magnétiques provoquées par la présence du bateau et de maintenir le câble de communication du magnétomètre à distance des pâles du bateau.

Figure 3 : Schéma des 52 briques de l’emplacement des fouilles (OpenCPN) (gauche) et carte google earth des briques 4,5,6 et 11 recueillies le 13 juillet 2018 (droite).

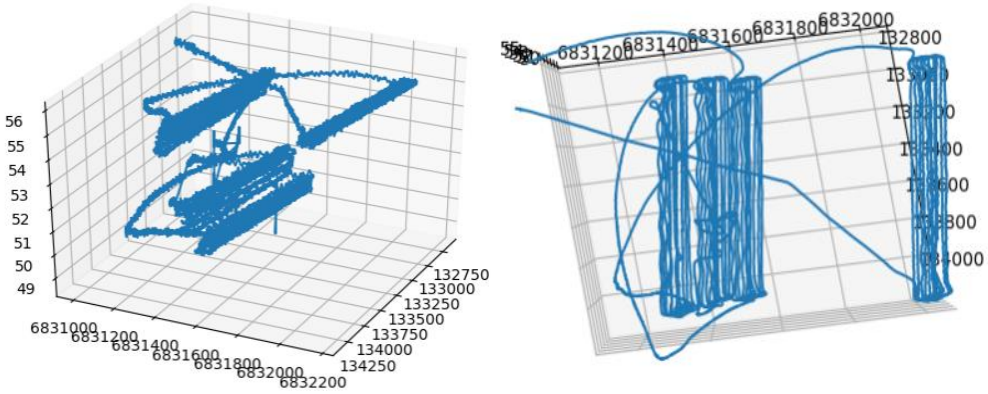


Figure 2 : Vue de biais et du dessus des briques 4,5,6 et 11

On constate bien les 10 allers-retours effectues par le robot pour chaque brique. La différence d’altitude entre les briques s’explique par un changement du niveau de l’eau entre les prises de ces briques. Les points géographiques sont exprimés en Lambert 93 et la profondeur est en mètres.

Description du problème :

Soit $\vec{P} = [\overrightarrow{p_0}^T, \cdots, \overrightarrow{p_{tmax}}^T]^T where \overrightarrow{p_t} = [p_{t,x}, p_{t,y}, p_{t,z}]^T \forall t \in [\![0, tmax]\!]$ l’ensemble des points d’entrainement, c’est-à-dire l’ensemble des points de la trajectoire du robot autonome. Et $\vec{Q} = [\overrightarrow{q_0}^T, \cdots, \overrightarrow{q_N}^T]^T where \overrightarrow{q_n} = [q_{n,x}, q_{n,y}, q_{n,z}]^T \forall n \in [\![0, N]\!]$ l’ensemble des sources magnétiques présentes dans les alentours de la trajectoire du robot de valeur et de dimensions inconnues.

Nous cherchons à établir une cartographie magnétique des fonds marins à partir des données d'entrainement recueillies. Cela se concrétise par une estimation des grandeurs magnétiques en chaque point d'une grille serrée projetée sur le fond marin. Les zones de plus forte activité magnétique seront alors considérées comme potentiellement porteuses de dépôts de fer pouvant provenir d'épaves. Cette grille est représentée par le vecteur $\overrightarrow{P^*}$. **Voir Figure 1**

$$\overrightarrow{P^*} = [\overrightarrow{p_0^{*T}}, \cdots, \overrightarrow{p_J^{*T}}]^T where \overrightarrow{p_j^*} = [p_{j,x}^*, p_{j,y}^*, p_{j,z}^*]^T \forall j \in [\![0, J]\!]$$

.

Description des lois magnétiques, modèle du dipôle magnétique vu à longue distance

Le champ de force à longue distance B d'un créé par un dipôle est donné par la formule :

Formule 1 :

$$B(m, r, \lambda) = \frac{\mu_0}{4\pi} \frac{m}{r^3} \sqrt{1 + 3\sin^2(\lambda)}$$

Avec :[1]

B, la force du champ, calculée en teslas
r, la distance du centre du dipôle, en mètres
λ, la latitude magnétique mesurée en radians ou degrés
m, le moment dipolaire, mesuré en ampère carré mètre ou joule par tesla
μ_0, la perméabilité du vide, en henry par mètre

Ici, nous cherchons à caractériser le champ magnétique créé par un dépôt de matière ferromagnétique tel qu'un canon enfoui sous le fond de la mer. Ces dépôts sont composés d'un grand nombre de dipôles magnétiques dont les moments magnétiques sont de direction aléatoire. On peut donc considérer le champ magnétique produit comme **Glossaires** des termes techniques. Dès lors, ce dernier ne dépend que de la distance r entre le dipôle et le magnétomètre, et se résume donc à la formule suivante :
Formule 2 :

$$B = \frac{C * m}{r^3}$$

Où C est une constante en henri par mètre. Il peut aussi être noté que le niveau magnétique détecté en un point de l'espace est linéaire en fonction du nombre de sources magnétiques en présence. Dès lors, le niveau magnétique détecté en un point est la somme de toutes les sources magnétiques en ce point.

Estimation des incertitudes en jeu

Le problème que nous cherchons à résoudre ici est celui d'un problème de régression pour un lot de données en 4D. Les trois premières dimensions étant spatiales et constituant la position du magnétomètre. La quatrième étant la valeur magnétique recueillie en ce point exprimée en Teslas. Nous cherchons à estimer en chaque point de la brique étudiée les valeurs magnétiques ainsi que l'erreur de mesure commise.
L'erreur commise sur les mesures magnétiques découle en grande partie des erreurs de mesure sur la position du magnétomètre à chaque instant. On peut alors modéliser la position du magnétomètre par l'équation suivante :

[1] **Source : https://en.wikipedia.org/wiki/Dipole**

Formule 3 : **Voir Figure 1**

$$\overrightarrow{P_t} = \overrightarrow{Pb_t} - L\frac{\overrightarrow{Vb_t}}{\|\overrightarrow{Vb_t}\|} + \vec{d} + \overrightarrow{\epsilon_t}$$

Où :

-$\overrightarrow{P_i}$et $\overrightarrow{Pb_i}$sont respectivement les vecteurs position a un instant i du magnétomètre et du bateau autonome par rapport à un repère galiléen quelconque R. A noter que dans le cas présent ces vecteurs sont des vecteurs en trois dimensions.

- L est la distance séparant le bateau et le kayak et $\vec{V_i}$est le vecteur vitesse du bateau a un instant i. Le second terme de l'équation représente alors une translation due à la trainée du kayak a une distance L dans la direction opposée au sens de déplacement du bateau (~10m en arrière). Cette erreur dépend de la direction du bateau à chaque instant d'où son indice i.

- $\vec{d}$est le vecteur profondeur du magnetometre par rapport a la surface.

- $\overrightarrow{\epsilon_i}$quant à lui represente la somme de l'erreur de position du bateau, c'est à dire l'erreur de mesure du GPS embarque, et de la derive du depresseur vis-a-vis du kayak. L'erreur du GPS peut être estimée à moins de 2m d'erreur de position horizontale et moins de 4m d'erreur de position verticale avec un intervalle de confiance a 95%[2]. L'erreur provoquée par la dérive du dépresseur étant en revanche très dure à modéliser, nous nous contenterons de surévaluer l'erreur du GPS notamment l'erreur horizontale puisque le dépresseur aura tendance à être plus efficace pour maintenir sa profondeur.

De cette position du magnétomètre, nous en déduisons les niveaux magnétiques enregistrés par ce dernier à chaque instant $B_i$par la formule suivante :

Formule 4 : **Voir Figure 1**

$$B_t = \sum_{n=0}^{N} \frac{Cm_t}{{r_{tn}}^3} + \beta_t = \sum_{n=0}^{N} \frac{Cm_t}{\|\overrightarrow{q_n} - \overrightarrow{P_t}\|^3} + \beta_t = \mu_t + e_t$$

Où :

- L'ensemble des q_j forme l'ensemble des sources magnétiques supposées quasi-ponctuelles dans les alentours du magnétomètre.

- $\beta_i$symbolise l'erreur de mesure du magnétomètre. Celle-ci étant très faible (de l'ordre de 0.1nT d'après la documentation du SeaSPY de Marine Robotics [3], nous ne la prendrons pas en compte dans les calculs d'erreur.

De plus, nous cherchons à estimer les valeurs magnétiques en chaque point d'une grille régulière recouvrant toute la brique étudiée. Quel que soit l'estimateur utilise, celui-ci provoquera son lot d'incertitudes qui cette fois-ci dépendront de la distance entre le point de la grille estime et l'ensemble des points de donnée utilisés pour l'estimation.

[2] **Précision du GPS :** données provenant du site officiel du gouvernement des Etats-Unis d'Amérique d'information sur le GPS : https://www.gps.gov/systems/gps/performance/accuracy/

[3] **Documentation du magnétomètre SeaSPY de Marine Robotics** : https://www.ashtead-technology.com/datasheets/Marine_Magnetics_SeaSPY.pdf

2. Choix d'un algorithme d'interpolation

2.1.Algorithmes déterministes classiques

Un problème de régression consiste en l'interpolation d'une grandeur physique en un ensemble de points dits de test à partir de mesures connues en un autre ensemble de points, dites données d'entraînement. Ces mesures peuvent prendre un ensemble infini de valeurs, tel que l'ensemble des valeurs réelles dans un intervalle. Soient 2 ensembles de points N dimensionnels : P, et P* et un ensemble de réels B. P et B sont le vecteur position des données d'entrainement et la valeur mesurée en ce point tandis que P* est l'ensemble des données de test dont on cherche à estimer la grandeur physique mesurée en chaque point. Dans le cas d'algorithmes déterministiques classiques, la valeur physique en un point est estimée grâce à la similarité entre son vecteur position et ceux des points qui lui sont les plus proches. Cette similarité se résumant souvent à la distance entre le point de test considéré et les points d'entraînement les plus proches.
Enfin, l'estimée au point de test peut être par exemple la valeur du point d'entrainement le plus proche ou une combinaison linéaire des mesures aux points d'entrainement les plus proches pondérées par la distance à ses points. Les résultats de ces deux algorithmes de régression appliqués à des données en 1D sont présentés en **Figure 4**.

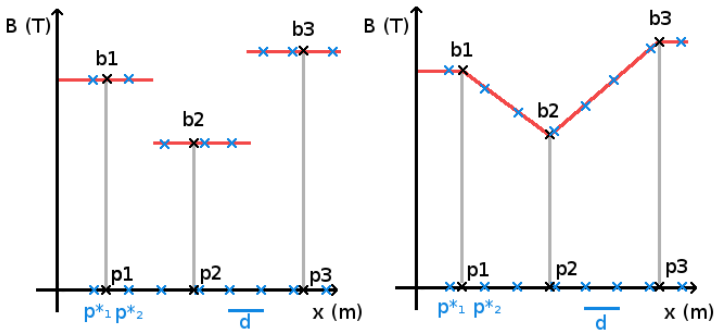


Figure 4 : Exemple d'une estimée en un ensemble de points de test avec un algorithme déterministe en 1D.

Les données magnétiques sont ici représentées en fonction de x pour les points d'entrainement Pi (en noir) et aux points de test Pi* (en bleu) pour les deux algorithmes présentés plus haut. En prenant un ensemble de points de test continu, la fonction magnétique estimée tendrait vers la courbe en rouge.

2.2.Krigeage

(Jean-Paul Chilès, 1999), p154 « Kriging with a known mean »
Le krigeage est un autre algorithme de régression qui permet d'estimer simultanément la valeur moyenne et l'écart-type d'une grandeur physique à partir d'un ensemble de données. Cet algorithme est aussi appelé procédés gaussiens à variogramme en opposition aux procédés gaussiens dits « standards » décrits dans [3] qui eux sont non-déterministes. Le krigeage, a la particularité de ne pas nécessiter de sélectionner des hyperparamètres vis-à-vis des données avant de le lancer. En revanche, la fonction de kernel des**Glossaires des termes** techniquesest ici remplacée par un variogramme. Ce variogramme définie lui aussi la covariance entre les points d'entrainement, soit la régularité de la fonction d'estimation.

En reprenant les définitions de P, B et P* fournies plus haut, le krigeage (au même titre que les Procédés Gaussiens Standards) est le meilleur estimateur linéaire exact. Ce qui signifie que la surface extrapolée par krigeage passera par les points d'entrainement de manière exacte à la condition de supposer l'exactitude des données d'entrainement. La valeur du poids a un point de test B^*s'effectue par pondération des valeurs aux points d'entrainement $B_\alpha$par la formule suivante :

$$B^* = \sum_\alpha \lambda_\alpha B_\alpha + \lambda_0$$

(*0)

Où les $\lambda_a$sont les poids associes aux points d'entrainement et $\lambda_0$est la valeur moyenne de la grandeur physique étudiée, potentiellement inconnue. Dans cet exemple, nous considerons cette valeur constante et connue ce qui, comme démontré dans (Jean-Paul Chilès, 1999), revient à totalement ignorer cette valeur des calculs.

Le calcul des poids $\lambda_a$decoule de la minimisation de l'erreur moyenne au carré, ou Mean Square Error en anglais (MSE) entre la valeur exacte au point de test B_0(inconnue) et l'estimee B^*. Celle valeur coïncide avec la variance de l'estimée en ce point $Var(B^*)$dans le cas ou on neglige la valeur moyenne de la grandeur physique etudiee. On obtient alors :

$$MSE = E[(B^* - B_0)^2] = Var(B^* - B_0) + [E[B^* - B_0]]^2$$
$$= \sum_\alpha \sum_\beta \lambda_\alpha \lambda_\beta \sigma_{\alpha\beta} - 2\sum_\alpha \lambda_\alpha \sigma_{\alpha 0} + \sigma_{00}$$

(*1)

La détermination des poids $\lambda_a$permettant de minimiser la MSE se fait en egalisant la derivee de la MSE et 0 par ces meme poids. On obtient alors :

$$\frac{\partial}{\partial \lambda_a}(E[(\overrightarrow{B^*} - \overrightarrow{B_0})^2]) = 2\sum_\beta \lambda_\beta \sigma_{\alpha\beta} - 2\sigma_{\alpha 0} = 0 \Leftrightarrow \sum_\beta \lambda_\beta \sigma_{\alpha\beta} = \sigma_{\alpha 0}$$

(*2)

Ou sous format matriciel :

$$\Sigma\lambda = \sigma_0$$

(*3)

Avec :

$$\Sigma = \begin{pmatrix} \sigma_{\alpha 1 \alpha 1} & \cdots & \sigma_{\alpha 1 \alpha \tau} \\ \vdots & & \vdots \\ \sigma_{\alpha \tau \alpha 1} & \cdots & \sigma_{\alpha \tau \alpha \tau} \end{pmatrix} \vec{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_\tau \end{pmatrix} \overrightarrow{\sigma_0} = \begin{pmatrix} \sigma_{\alpha 1,0} \\ \vdots \\ \sigma_{\alpha \tau,0} \end{pmatrix}$$

Σ, appelee la matrice de krigeage, ou matrice d'autocorrelation, n'est rien d'autre que la matrice de covariance entre les points d'entrainement. Elle fait état des relations de correlation entre les points d'entrainement. Elle est déterminée grâce à un outil statistique appelé le variograme, d'où le nom de Procédés Gaussiens avec Variograme.

$\overrightarrow{\sigma_0}$ est le vecteur de covariance entre le point de test considéré et les points d'entraînement. De même, ce vecteur dépend uniquement de la proximité du point de test avec les points d'entraînement.

$\vec{\lambda}$ enfin est le vecteur des poids des points d'entrainement. Il est déterminé par résolution de l'équation (*3). Une fois ces poids déterminés, il devient possible de calculer l'estimée au point d'entrainement en remplaçant les poids dans l'équation (*0). De la même manière, en remarquant qu'en multipliant par les poids et en sommant selon ces derniers l'équation (*2) on obtient $\sum_\alpha \sum_\beta \lambda_\alpha \lambda_\beta \sigma_{\alpha\beta} - \sum_\alpha \lambda_\alpha \sigma_{\alpha 0} = 0$ ce qui en remplaçant dans l'équation (*1) nous permet d'obtenir le calcul de la variance de l'estimée :

$$Var(B^*) = \sum_{\alpha} \lambda_\alpha \sigma_{\alpha 0} + \sigma_{00}$$

$$\Leftrightarrow \sigma_{SK} = \sigma_{00} - \overrightarrow{\lambda^T} \cdot \overrightarrow{\sigma_0}$$

Ou σ_{SK} est appelée la variance de krigeage et represente la variance de l'estimée par rapport à la valeur réelle. A noter qu'encore celle-ci ne dépend que de la position relative du point de test et des points d'entrainement.

Ainsi on constate que le krigeage permet d'estimer la valeur de la grandeur physique étudiée en un point de test ainsi que l'erreur commise par l'estimation en résolvant le système d'équations (*3). Comme nous le verrons par la suite, il est aussi possible lors du calcul de l'estimée en un point de test de ne considérer que les points d'entrainement les plus proches de celui-ci. On parle alors de fenêtre mouvante. Cela permet de réduire considérablement la complexité de l'algorithme tout en conservant une bonne précision puisque les points d'entrainement les plus proches, et donc les plus significatifs pour l'estimation, sont conservés. Bien qu'il reste indéniable que les Procédés Gaussiens Standards offrent une précision accrue par rapport au krigeage (voir source (Yuxin Zhao, 2018) pour plus de précisions), cela se fait au détriment d'une complexité de calcul plus élevée que le krigeage permet de contourner. C'est pourquoi nous avons privilégié cet algorithme pour notre implémentation.

3. Description de l'algorithme de krigeage

3.1. Historique

Le krigeage, du nom de D. G. Krige, un statisticien et ingénieur minier sud-africain, est un terme utilise pour la première fois par G. Mathéron en 1963 pour désigner un algorithme premièrement utilisé en géostatistiques dans le cadre d'estimation de ressources minières. Cet algorithme a été créé initialement pour des raisons économiques. L'idée était d'obtenir une estimation des réserves minières sur une zone de grande envergure tout en minimisant les opérations d'échantillonnages car celles-ci consistaient en des extractions très couteuses. De plus, les méthodes d'estimation déterministes simples ne parvenaient pas à rendre compte efficacement des fortes variations spatiales de concentration de minerais, appelées pépites (nuggets en anglais). Ces variations brusque de concentration de minerais étant très sporadiques par natures, mais faussaient toutes les mesures alentours dans le cas d'algorithmes déterministes simples. D'autre part, la nature irrégulière des emplacements d'échantillonnage dans le domaine de l'exploitation minière rendait aussi impossible la mise en place d'algorithmes de type analyse de Fourier ou autre reposant sur des répétitions de signaux périodiques.

3.2. Différentes formes de krigeage et leurs applications

Comme expliqué dans le chapitre précédant, le principe du krigeage repose sur la résolution d'un système d'équations de la forme de la formule (*3) dans lequel la matrice de krigeage est obtenue à l'aide d'un outil statistique appelé le variograme. Le variograme permet de caractériser la dépendance spatiale entre les points d'entrainement en fonction de leur distance relative. Là où des outils statistiques tels que la moyenne ou l'écart-type ne permettent que de se faire une idée globale des échantillons, le variograme permet d'étudier la granularité des données d'entrainement, qui est un phénomène local. Il est ainsi tout à fait possible d'observer des ensembles d'échantillons possédant des moyennes et écarts-types semblables avec des variogrames sensiblement différents.
Le variograme n'est donc rien d'autre que la fonction de la variance moyenne entre chaque paire de points d'entrainement en fonction de leur distance relative. Il est calculé de manière générale par la minimisation par la méthode des moindres carres du variograme dit "empirique", c'est à dire calculé sur les données d'entrainement selon la formule suivante :

$$\gamma(h) = \frac{1}{2} \cdot \frac{1}{U(h)} \sum_{i=0}^{U(h)} (B_i - B_{i+h})^2$$

Ou $B_i$represente la grandeur physique au ieme point d'entrainement. Le variograme est ici représenté en fonction de h, appelé "lag". En effet, les distances entre chaque paire de points d'entrainement pouvant prendre n'importe quelles valeurs réelles, il est nécessaire de les regrouper en des paquets de paires de distances équivalentes, les lags, avant de calculer la variance sur ceux-ci. $U(h)$ représente ainsi la population du lag h, c'est à dire le nombre de paires de points appartenant au lag h. Une illustration des lags est fournie en **Figure 5**.

On choisit alors un type de variograme parmi les profils les plus courants présentés en **Figure 6**. Le choix du profil se fait selon des observations préliminaires du variograme théorique ou selon des connaissances préalables des phénomènes physiques étudies. Par exemple, si le phénomène physique est connu comme très régulier, des profils offrant une dérivée nulle en zéro tel que le modèle gaussien seront préférés à un modèle sphérique par exemple. Chacun de ses modèles possède des hyperparamètres qui déterminent complètement le variograme. Ces hyperparamètres sont en général déterminés par la méthode des moindres carres en minimisant l'erreur entre le variograme empirique, tiré des données réelles, et le variograme théorique, découlant des hyperparamètres testés. Une fois des hyperparamètres optimaux fixes, il reste à mettre en place le système de krigeage comme présenté dans la formule (*3) et dont la résolution nous fournit les poids λ nécessaires à l'estimation en chaque point de la grille P^*.

Bien que le système d'équations du krigeage soit toujours de la forme de la formule 3, La matrice de krigeage Σ et le vecteur de covariance entre les points d'entrainement et le point de test $\overrightarrow{\sigma_0}$ dépendent des hypothèses simplificatrices mises en place et peuvent prendre les valeurs resumees dans le tableau 1 en **Annexe B** : Tableau récapitulatif des types de krigeage. Le choix du type de krigeage utilisé se fait selon des connaissances préalables sur la moyenne spatiale du phénomène physique étudié. Pour rappel, on suppose la grandeur physique étudiée être de la forme

$$B(\vec{p}) = \mu(\vec{p}) + e(\vec{p}) + w.$$

Le krigeage simple est utilisé dans l'hypothèse où la valeur moyenne $\mu(\vec{p})$de la grandeur physique étudiée sur l'ensemble du domaine Dest constante et connue : $\mu(\vec{p}) = \mu_0 \forall p \in D$. Le systeme d'equations du krigeage prend alors la même forme que celle développée dans l'équation (*3), avec $\sigma_{\alpha i \alpha j} = \gamma(\|P_i - P_j\|) = \gamma(l_{ij})$. L'hypothese de moyenne constante et connue etant cependant une hypothese tres lourde, elle est rarement remplie sur un ensemble de données à traiter. Il reste possible d'estimer une moyenne en calculant une moyenne statistique sur un très grand nombre de données d'entrainement. Cependant, (Jean-Paul Chilès, 1999) nous met en garde contre de telles pratiques car il est impossible d'estimer l'erreur produite par de telles approximations. En effet, rien ne permet de distinguer la moyenne μ des résidus que sont $e(\vec{p})$ et w. Ainsi toute estimation globale de la moyenne provoquera une pollution de celle-ci par les résidus. Au lieu de cela, il vaut mieux utiliser d'autres types de krigeage tels que le krigeage ordinaire et le krigeage universel qui sont des généralisations du krigeage simple.

L'utilisation du krigeage ordinaire présuppose une moyenne constante mais de valeur inconnue. Quant au krigeage universel, il suppose une moyenne polynomiale a coefficients constants inconnus de la forme : $\mu = \sum_{m=1}^{M} a_m f(x)^{(m)}$. Les coefficients $a_m$sont constants inconnus tandis que les $f(x)^{(m)}$sont des fonctions de base dont on connait la valeur et dont le degre n'excede pas 2 en pratique d'après (Jean-Paul Chilès, 1999).Il est d'ailleurs courant que ces fonctions soient des monômes, et le premier coefficient vaut $f_i^{(0)} = 1 \forall i \in [\![1, \tau]\!]$ car celui-ci fait référence à la valeur constante de la moyenne, qui est donc parfaitement représentée par la fonction constante $f(x) = 1 \forall x$. Un autre exemple, si on suppose que la moyenne est lineaire en x et y, soit $\mu(\vec{p}) = a_0 \cdot 1 + a_1 \cdot x(\vec{p}) + a_2 \cdot y(\vec{p})$, on peut alors poser comme fonctions de base : $f_i^{(0)}(\vec{p}) = 1, f_i^{(1)}(\vec{p}) = x(\vec{p}), f_i^{(2)} = y(\vec{p}) \forall i \in [\![1, \tau]\!]$ ce qui revient simplement a rentrer les coordonnées des points

d'entrainement dans la matrice de krigeage. (Jean-Paul Chilès, 1999) (voir tableau en **Annexe B** : Tableau récapitulatif des types de krigeage).
Dans le cas de ces deux méthodes, le calcul de la moyenne est incorporé au système d'équations du krigeage. Elle est donc résolue en même temps que les poids λ.

On remarque par ailleurs qu'une fois le variograme théorique fixé, le reste des calculs est purement déterministe. De plus, étant donné que les seuls paramètres à fixer dans l'intégralité de l'algorithme de krigeage sont les hyperparamètres du variograme théorique et que ces derniers sont déterminés par méthode des moindres carres, il devient clair que le krigeage est un algorithme ne nécessitant le choix d'aucun paramètre arbitraire au préalable autre que le modèle du variograme théorique et le type de krigeage sélectionnés.

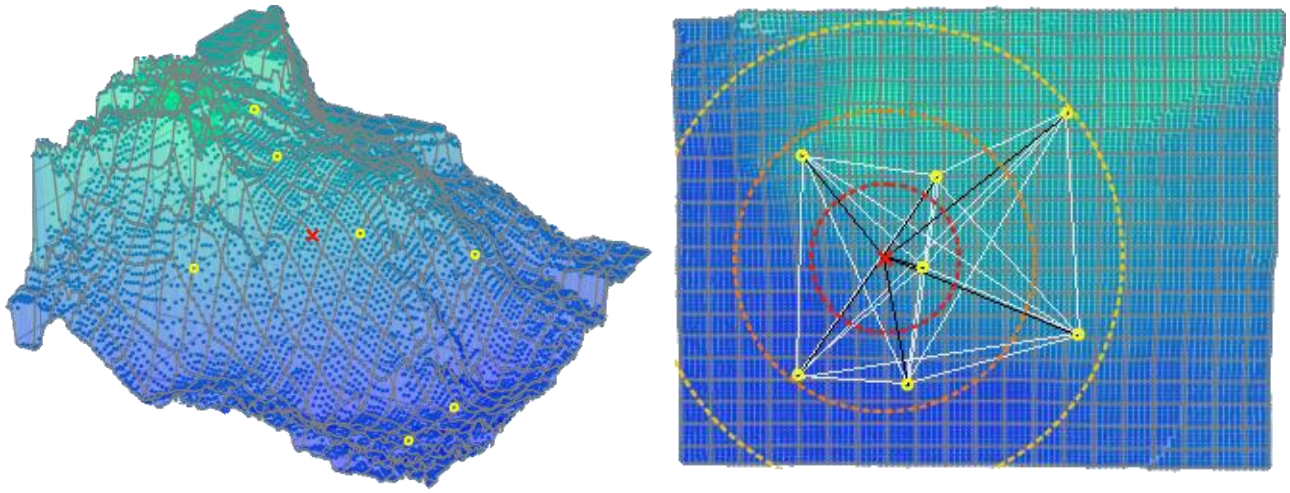


Figure 5 : Représentation en perspective et en vue du dessus d'une surface krigee et de quelques points particuliers.

Complément d'information à la figure 5 : On cherche ici à déterminer l'altitude en un point de test (en rouge) à partir des données en 7 points d'entrainement (en jaune). Les paires de points amenant au calcul du variograme empirique sont représentées en gris clair. Sont aussi représentés trois lags (entre cercles pointillés) autour du point de test. Pour classer les paires de points en lags, il suffit de se figurer ces trois cercles concentriques déplacés en chaque point d'entrainement. Pour chaque autre point d'entrainement, on regarde entre quels cercles il se situe et on le rajoute à la population du lag correspondant. Une fois ce travail effectué en chaque point d'entrainement, le calcul de la variance en chaque lag nous fournira le variograme empirique. A noter que les points figurant sur la surface 3D ne sont pas les données dont la surface est issue, mais ont été rajoutées après coup a des seules fins d'illustration du fonctionnement des lags.

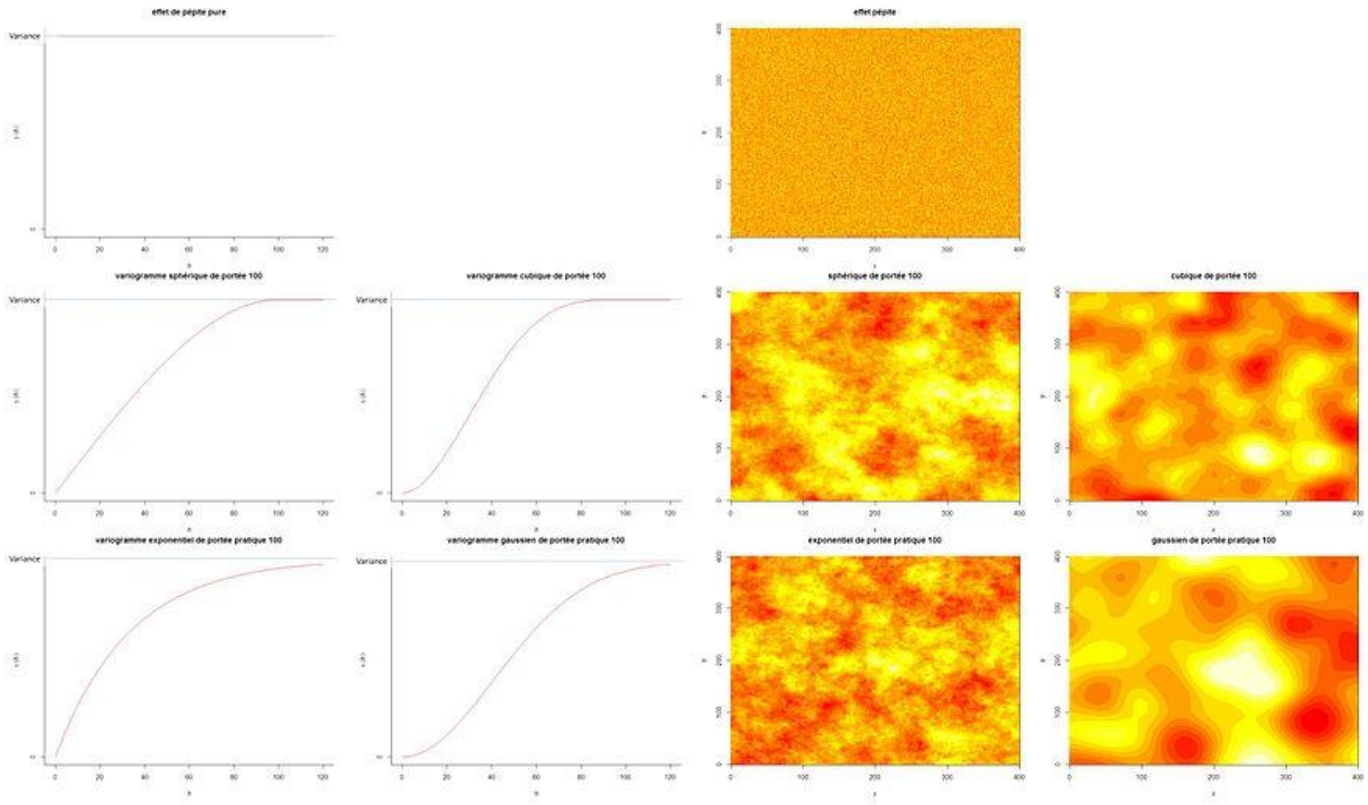


Figure 6 : Cinq modèles de variograme différents et l'estimation qui en découle.

Complément d'information à la figure 6 : On constate que chaque modèle nous fournit des résultats bien distincts qui partagent les mêmes moments d'ordre 1 et 2 (moyenne et écart-type).[4]

[4] **Source : Wikipedia : https://en.wikipedia.org/wiki/Variogram**

4. Implémentation

Du point de vue de l'implémentation finale, le code se sépare en deux parties indépendantes : la partie « extraction » en charge de l'extraction de données dans une base de données à partir de fichiers csv ou de rosbag, et la partie « krigeage » qui s'occupe du calcul du krigeage à proprement parler.

4.1. Extraction des données magnétométriques

Le cahier des charges pour la partie « extraction » du code peut se résumer dans la liste d'objectifs suivante :

- Doit être capable d'extraire des données en base de données à partir de fichiers csv/txt/rosbag.
- Doit être facile d'utilisation, ne nécessitant que des connaissances sommaires en python pour être utilisé.
- Doit prendre en charge les doublons éventuels dans les données extraites.
- Doit être modulable selon les besoins de l'utilisateur, dans le détail, les caractères suivants doivent être paramétrables :
    - Les informations de base de la base de données, à savoir le nom de la base de données sélectionnée ou encore le nom des colonnes de la table créée.
    - Le lien entre les colonnes du fichier CSV / les messages du rosbag et les colonnes de la table créée.
    - Etre capable de créer des colonnes comme le résultat d'opérations entre certaines colonnes déjà en base de données.
- Quelques fonctionnalités optionnelles :
    - Générer un fichier log rassemblant les éventuelles erreurs d'extraction afin de faciliter le débogage.
    - Générer des messages d'erreur clairs pour un utilisateur non averti à l'aide d'un mode verbose.

Le code fourni doit être capable d'effectuer ces opérations sans que l'utilisateur ne connaisse le langage SQL.

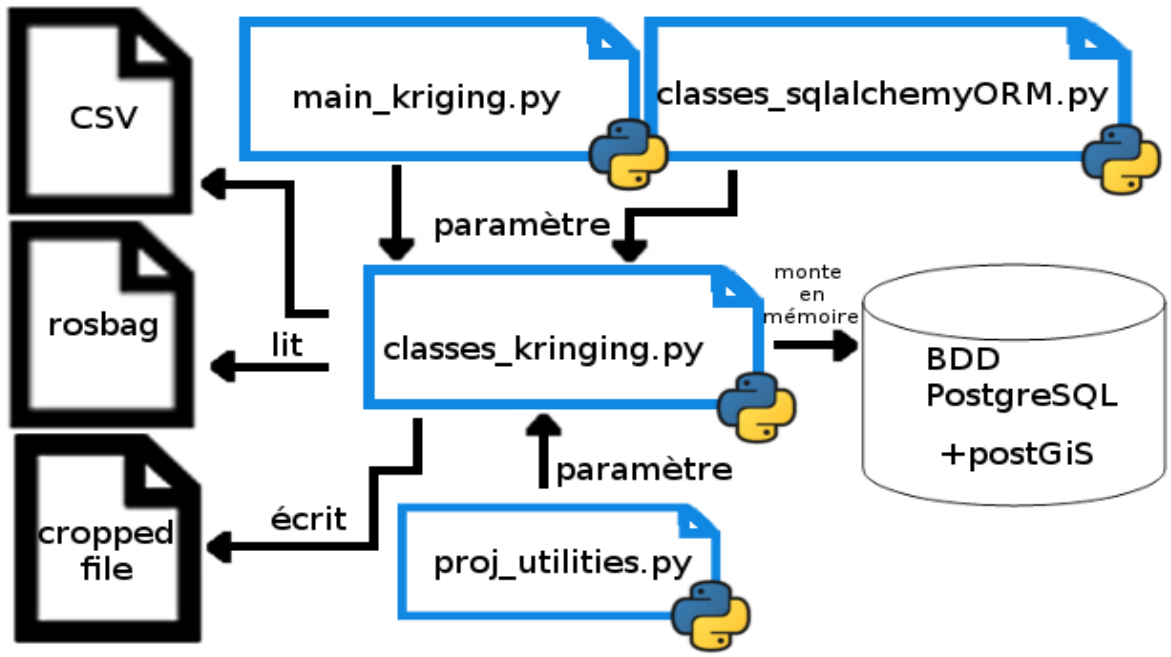


Figure 7 : Schéma de l'organisation des fichiers python de la partie « extraction » du code.

Une représentation schématique sommaire des fichiers python en charge de cette partie du code est fournie en **Figure 7.**

Dans le détail, fichier par fichier :

- **Fichier main_exctractdb.py** :

Fichier de paramètres pour la partie extraction. Fait office d'interface utilisateur pour la partie extraction. Il consiste en l'instanciation de la classe Ext_msg_db définie dans le fichier classes_extraction.py. Ce code peut gérer l'extraction en base de données d'un fichier csv/rosbag à la fois. L'utilisateur doit donc lancer ce code autant de fois qu'il y a de fichiers à extraire. Une tentative d'automatisation de ce procédé a été tenté. Bien que l'extraction de plusieurs fichiers CSV à la fois ne poserait aucun problème, l'extraction de plusieurs rosbag à la fois reste pour le moment impossible à cause d'une fuite mémorielle d'origine inconnue. Les fichiers rosbag traités pouvant excéder 1Go, l'extraction de plusieurs fichiers rosbag de cette taille consécutifs provoque un débordement de la RAM. En conséquence, l'utilisateur ne pourra pas extraire plusieurs fichiers en série.

- **Fichier classes_sqlalchemyORM.py :**

Définit les classes gérées par la librairie sqlalchemy pour faire le lien entre des variables python et les colonnes de la base de données. Pour chaque table en base de données, l'utilisateur doit créer en amont une classe héritant de la declarative_base de sqlalchemy. Cette classe est composée de :

- la variable _tablename_ qui définit le nom de la table générée

- D'un ensemble de variables de type sqlalchemy.Column. Chacune de ses variables définit une colonne de la table du même nom, le type de données stockées et les propriétés de la colonne étant déterminés par le type de sqlalchemy.Column sélectionné. L'utilisateur peut s'inspirer des quelques exemples fournis dans ce fichier, et peut partir de la classe ROSMsg_base, possédant tous les prérequis nécessaires. Les entrées ainsi générées seront reconnues de manière unique à l'aide de la clé primaire « seq » qui est réglée en auto-incrément et qui doit obligatoirement figurer parmi les colonnes de la table.

- Afin d'extraire efficacement des points enregistrés en mémoire en fonction de leur position géographique, il est courant d'utiliser des géoindex, supportés par PostgreSQL à l'aide de l'extension PosGIS. Le code fourni permet l'implémentation simple de Geoindex en rajoutant à la classe sqlalchemy une variable de type Column(geoalchemy2.Geometry) comme présenté dans la classe Msg_nav_data.

Toutes les tables ainsi définies seront initialisées par la classe Ext_msg_db dans le fichier classes_extraction.py. Le déroulement de la mise en mémoire des lignes d'un CSV sont représentées dans la **Figure 8** à titre d'exemple. La montée d'un rosbag en mémoire se fait de manière semblable pour chaque message du rosbag sélectionné. A noter qu'une table doit être initialisée pour chaque rosmsg.

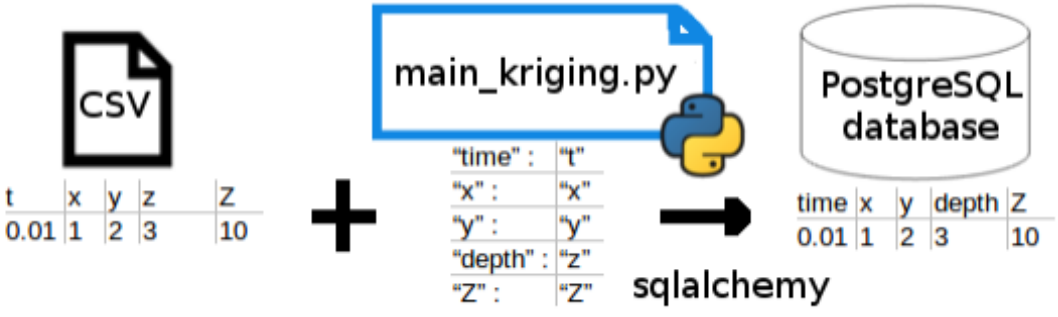


Figure 8 : Résumé de la montée en mémoire d'une ligne d'un CSV.

- **Fichier proj_utilities.py** :

Fichier en charge de la projection de coordonnées géographiques. Il peut être intéressant de projeter certaines colonnes représentant des coordonnées géographiques dans un autre repère. Dans le cas présent, c'est la projection de coordonnées GPS (WGS 84) vers Lambert 93. L'utilisateur peut sélectionner la projection de départ et d'arriver en modifiant les constantes globales epsg_input et espg_output présentes dans la section « PROJECTION CONSTRAINTS ». Pour ce faire, il suffit de remplacer la série de 4 chiffres représentant le numéro EPSG des projections sélectionnées. Ces derniers sont disponibles sur le site http://spatialreference.org. Il sera alors possible de créer un géoindex dans le nouveau repère dans le fichier main_extractdb.py. Pour le moment, seules les projections de repères en deux dimensions sont supportées.

- **Fichier classes_extraction.py** :

Définit la classe Ext_msg_db qui fonctionne selon la manière suivante :

- Sélectionne le fichier à extraire
- Se connecte à la base de données sélectionnée : génère une session de communication avec la base de données
- Initialise la (les) table(s) sélectionnées
- Fait le lien entre les colonnes du CSV / les messages du rosbag et les variables d'instance de la classe de sqlalchemy (voir classes_sqlalchemyORM.py)
- Applique les projections fournies par proj_utilities.py (optionnel)
- Monte en base de données la classe sqlalchemy une fois remplie
- Créé un fichier allégé contenant les n premières lignes du fichier pour un usage ultérieur (optionnel)
- Renvoie des informations sur le déroulement de l'extraction à l'utilisateur (non implémenté)
- Enregistre les messages d'erreur dans un fichier log (non implémenté)

4.2.Implémentation du krigeage

Le cahier des charges pour la partie extraction peut se résumer à la liste suivante :

- Doit être capable d'extraire les données présentes en base de données.
- Doit permettre une extraction rapide d'une fraction des données géographiques en mémoire appartenant à une région géographique donnée. En l'occurrence, doit permettre d'extraire les données d'une brique parmi les données recueillies au cours d'une journée.
- Doit afficher des informations préliminaires sur les données afin de permettre à l'utilisateur de juger de leur conditionnement.
- Doit afficher les résultats du krigeage de manière claire.
- Doit évaluer la performance de l'algorithme de krigeage en calculant les erreurs d'entrainement et de test et en les affichant de manière claire.
- Doit être modulable, les caractères suivants doivent notamment être paramétrables :
  - Le type de krigeage utilisé (ordinaire ou universel)
  - Le nom des colonnes à kriger
  - La liste d'affichages pertinents parmi ceux fournis.

Une représentation schématique sommaire des fichiers python en charge de cette partie du code est fournie en **Figure 9**:

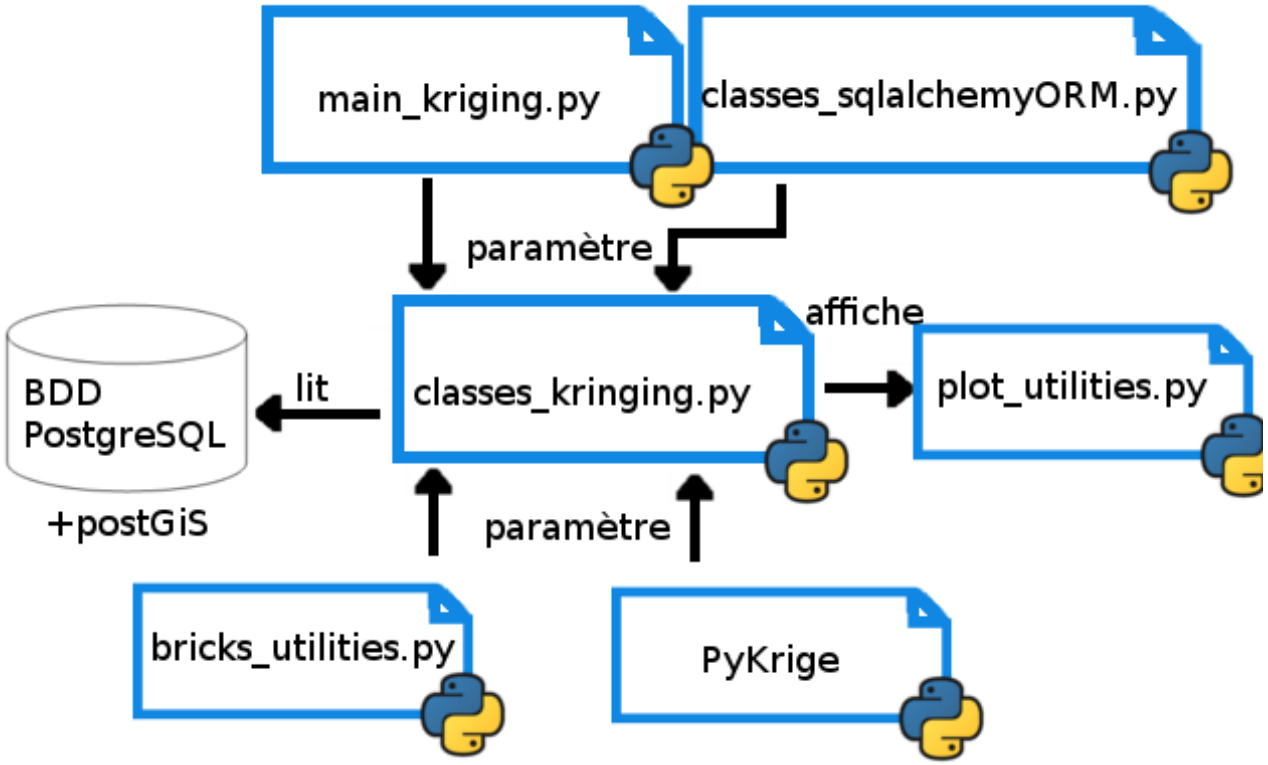


Figure 9 : Schéma de l'organisation des fichiers python de la partie « krigeage » du code.

Dans le détail, fichier par fichier :

- **Fichier main_kriging.py** :

Fichier de paramètres pour la partie krigeage. Fait office d'interface utilisateur pour la partie krigeage. Il consiste en l'instanciation de la classe Kriging définie dans le fichier classes_kriging.py.

- **Fichier bricks_utilities.py** :

Fichier en charge des fonctions d'extraction de points. Une fois les coordonnées 3D chargées en mémoire, et les géoindex générés, il devient possible de n'extraire qu'une partie des points stockés et de leur appliquer l'algorithme de krigeage. Nous sommes ici intéressés par l'extraction des points appartenant à une brique parmi les données récupérées au cours d'une journée de relevés. Cette brique, matérialisée par un parallélogramme, se résume à 6 valeurs correspondantes par aux 2 valeurs extrêmes du parallélogramme selon les 3 axes de l'espace. Ses valeurs extrêmes pour quelques briques exemple sont stockées dans le dictionnaire « bricks » présent dans le fichier bricks_utilities.py sous le format suivant :

« numéro de brique » = (xmin, xmax, ymin, ymax, zmin, zmax)

Les termes dans le tuple peuvent valoir False, auquel cas cela signifie que le parallélogramme sera non limité dans cette direction.
Libre à l'utilisateur de rajouter ses propres briques dans ce dictionnaire.

- **Fichier plot_utilities** :

Fichier rassemblant l'ensemble des fonctions d'affichage des informations fournies par le calcul de krigeage. Aussi bien les plots de prétraitement que ceux du calcul de krigeage ou des estimations d'erreur.

- **Fichier classes_kriging.py** :

Fichier de calcul de krigeage. Il contient la définition de la classe Kriging, en charge de l'extraction des données de la base de données, la génération de points aléatoires pour l'estimation de l'erreur de test… Pour ce qui est des calculs de krigeage, elle se rabat sur la librairie PyKrige, qui est une extension de scikit-learn dédiée au calcul par krigeage.

5. Résultats expérimentaux

En première approximation, nous avons commencé par étudier un problème en deux dimensions et ou les positions des points d'entraînement sont connues de manière exacte. Nous avons étudié pour cela un ensemble de données simulées fournies par Mr Jaulin qui consistent en des données de position en trois dimensions d'un **Glossaires** des termes techniques. Sa tâche était de déterminer la profondeur en chacun des points de ses parcours ainsi que ses données bathymétriques. Cet exemple a été utile pour mettre en place un premier environnement de développement pour le krigeage ainsi que de mettre en place les outils statistiques permettant de vérifier le bon fonctionnement de l'algorithme sur un cas simple.

5.1.Sur données 2D simulées

Dans le cas de cet exemple, la grandeur physique qui a été estimée par krigeage est la profondeur du fond marin. En effet, la profondeur du fond marin peut être calculée comme la somme de la profondeur de l'AUV, récoltée par un baromètre, et de la hauteur de ce dernier par rapport au fond marin, déterminée par un sonar latéral. De cette manière, la valeur de profondeur du fond marin est indépendante de la profondeur de l'AUV. La profondeur du fond marin peut dans ce cas être estimé par krigeage.

Prétraitement

Comme précisé précédemment, ce premier test a été l'occasion de mettre en place de premiers outils de vérification statistique. C'est dans cette optique qu'ont été créées des méthodes d'affichage, grandement inspirées ou reposant sur celles fournies par la librairie python **geostatmodels** ou inspirées des conseils pratiques fournis par (Jean-Paul Chilès, 1999). Ces méthodes sont regroupées dans le fichier plot_utilities.py. La première, la plus simple, est celle d'affichage des données d'entrainement effectuée par la fonction **plotscatterdata** et présentée en **Figure 10**.
On peut effectuer plusieurs remarques sur ces données d'entrainement. Tout d'abord, le fond marin n'est vraisemblablement pas plat avec une anisotropie allant de Nord-Est vers Sud-Ouest. Et, comme nous le verrons plus en avant (partie results, affichage de la surface 3D), la dérive de la profondeur du fond marin est fortement non linéaire. D'autre part, le robot s'est déplacé selon des rails en deux endroits distincts. Nous verrons que cela a des conséquences sur les populations des lags (voir preprocessing) et, par voie de conséquence, sur le variograme (Voir First use of the variogram : spherical).

Enfin, en regardant les emplacements ou le robot a croisé à plusieurs reprises son propre chemin, on constate que la profondeur recueillie en chacun de ses passages est différente. Or, comme nous l'avons vu précédemment, (Voir Estimate the depth of the seabed from the AUV data), la profondeur du fond marin collectée est censée être indépendante de l'altitude du robot. L'hypothèse la plus crédible est que, les données ayant été prises sur une étendue temporelle relativement importante (près de 1h40min), la marée a contribué à accroitre temporairement la profondeur du fond marin entre chaque prise de données. Cette nuisance aura, comme nous pourrons le constater au moment des résultats (Voir results), de lourdes conséquences sur la corrélation entre les données d'entrainement et les estimées en chaque point de test. Cette nuisance pouvant être analysée comme un bruit non gaussien. Nous proposerons cependant une méthode permettant de résoudre partiellement ce problème en **Annexe C** : ainsi qu'une analyse des résultats par cette méthode.

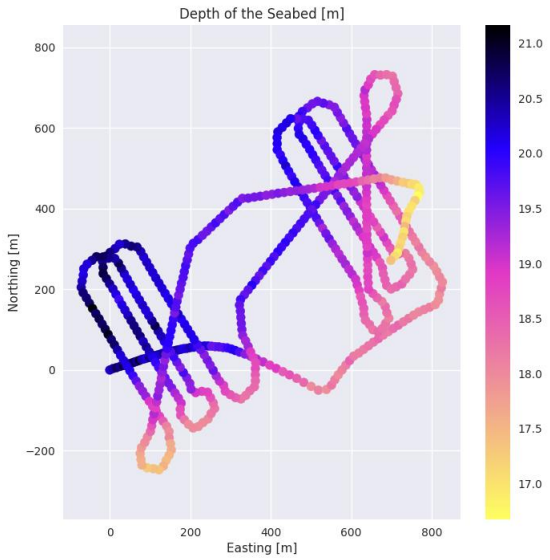


Figure 10 : Représentation de la profondeur du fond marin en chaque point d'entrainement.

Avant d'estimer la profondeur du fond marin en chaque point d'une grille par krigeage, il est important de vérifier que nos données d'entraînement ne soient pas trop éloignées d'un contexte gaussien afin que les axiomes de base du krigeage soient respectés. En supposant que la grandeur physique étudiée, ici la profondeur du fond marin, est suffisamment stable sur l'ensemble des données d'entraînement, les perturbations observées sont alors dues au bruit supposé gaussien qu'on souhaite quantifier. Pour vérifier si l'hypothèse du bruit gaussien est justifiée, il suffit dès lors de comparer un histogramme des données de profondeur du fond marin a une distribution gaussienne de même moyenne et écart-type. Ou plus exactement de comparer les quartiles théoriques d'une distribution gaussienne de mêmes caractéristiques avec la distribution réelle. Cette opération est effectuée par la fonction **probplot** de la librairie **scipy**.**stats** par le biais de la fonction **plotgaussiandist** dont le résultat est présenté dans la **Figure 12**.

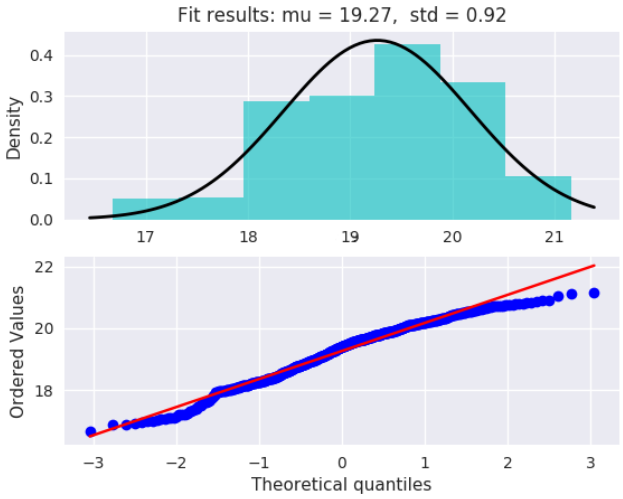


Figure 12 : Distribution des données et comparaison a une distribution gaussienne de même moyenne et écart-type.

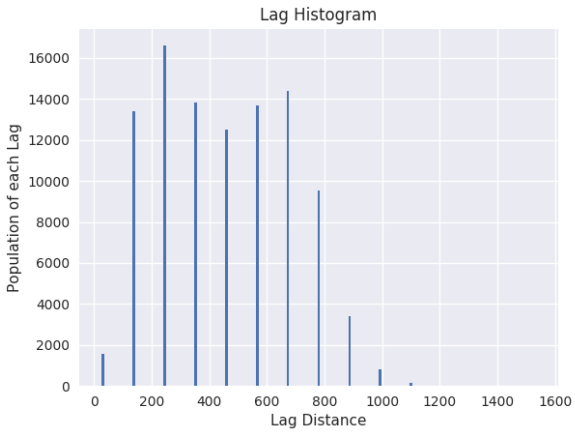


Figure 11 : Histogramme de population des lags

De la même manière , il peut être intéressant d'afficher les populations des différents lags utilisés pour le calcul du variogramme afin de s'assurer celles-ci ne soient pas trop faibles. Cela permet d'éviter des variations brutales de variance qui ne seraient pas dues qui seraient dues à un manque de population de certains lags et non propres à la granularité des données. D'autre part, une variation brutale de la variance peut être signe de la présence de différents clusters de données qui doivent être traités séparément. La question est d'autant plus légitime quand on se penche sur la répartition de nos données d'entrainement. Est-il valable de considérer l'ensemble de ces données comme étant un seul et même cluster ou est-il préférable de les distinguer en deux séries de rail distinctes ? Ce graphique est affiché sur la **Figure 11**. On peut alors voir qu'à l'exception des deux derniers points dont la population peut être considérée comme insuffisante, le reste des lags est suffisamment peuplé pour fournir des données viables. De plus, on peut constater deux pics de population distincts aux environs de 200 et 600m faisant ainsi état de deux clusters de données correspondant aux deux séries de rails. Mais vu que la population des lags entre ces deux pics reste conséquente et que l'exemple considéré n'est qu'un exemple introductif, nous avons préféré considérer l'ensemble des données comme un seul et même ensemble.

Choix du variogramme

Pour le calcul du variogramme théorique, le modèle sphérique a été choisi après comparaison des résultats fournis par les différents types de variogramme par validation croisée sur les données fournies pour un krigeage ordinaire. Les variogrames empirique et théorique des données d'entraînement, calculés par la librairie **pykrige**, sont fournis en **Figure 14**.

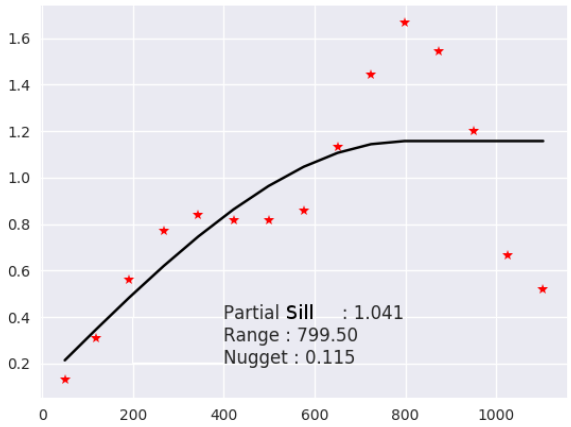


Figure 14 : Variogramme empirique (étoiles rouges) et théorique (courbe noire) des données d'entrainement sans prétraitement. Le variogramme théorique est basé sur le modèle sphérique dont les caractéristiques sont fournies sur le graphique.

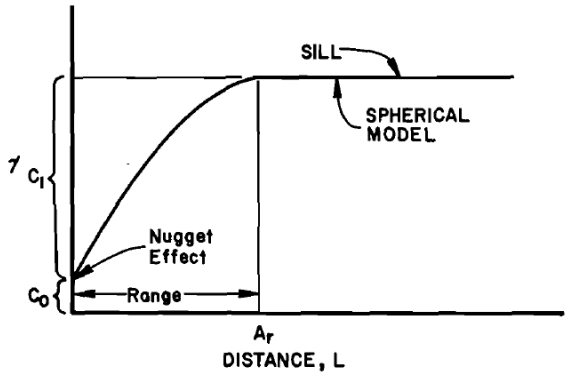


Figure 13 : Schéma explicatif d'un variogramme de modèle sphérique.

Interpolation par krigeage et estimation de l'erreur de krigeage

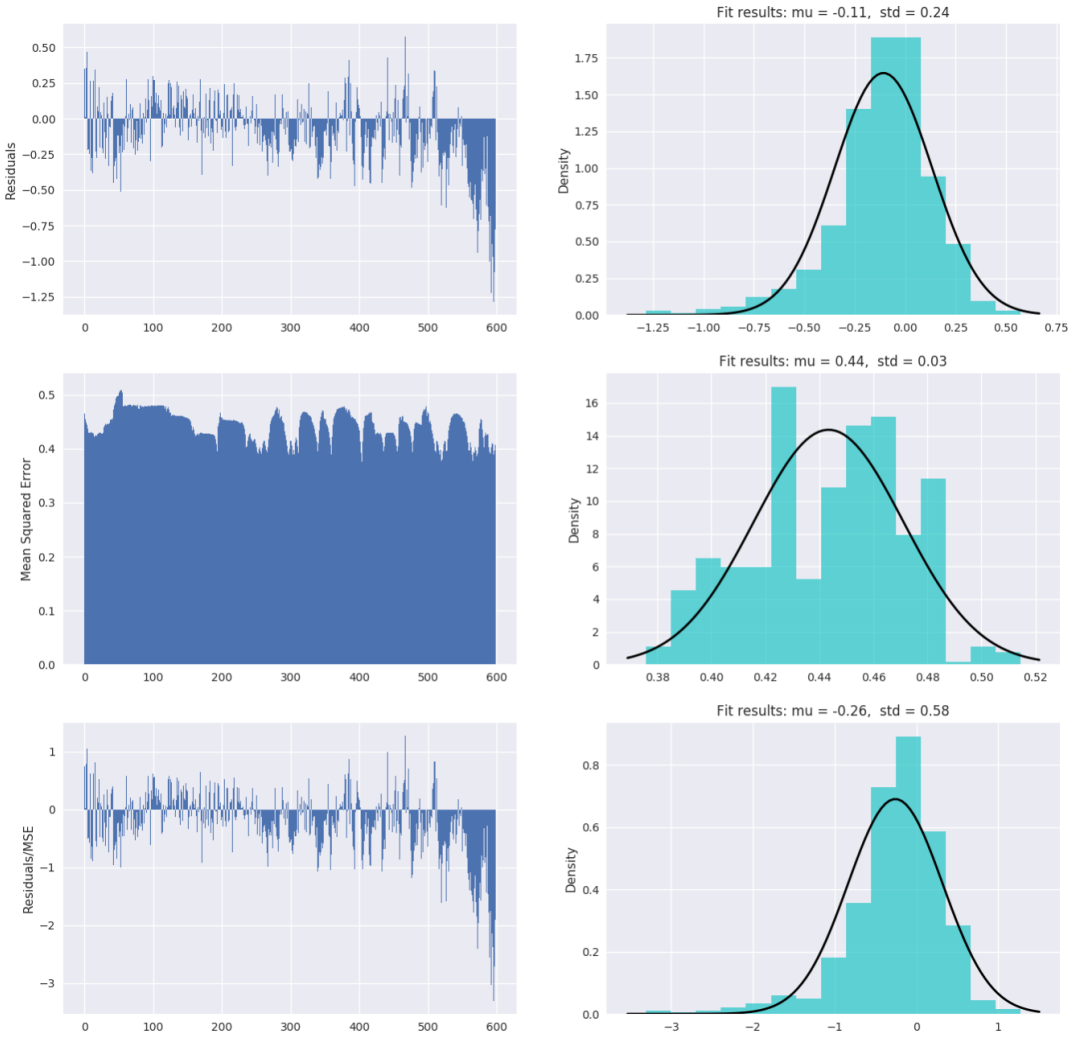
La gestion du calcul du krigeage en lui-même est pris en charge par la librairie pykrige qui est une extension de sklearn spécifiquement conçue pour le krigeage.[5] Sklearn étant une librairie python couramment utilisée pour l'apprentissage du machine learning. L'utilisation de la librairie est calquée sur celle de sklearn et consiste en deux étapes :

- L'entrainement de l'algorithme de krigeage. Il est réalisé par l'initialisation d'une instance de krigeage. On choisit d'abord le type de krigeage que l'on souhaite appliquer en instanciant une classe OrdinaryKriging ou UniversalKriging ou une des variantes a trois dimensions de ces deux algorithmes. Puis on fournit à cette instance le type de variogramme que l'on souhaite utiliser et les points d'entrainement. D'autres paramètres optionnels peuvent être fournis notamment les paramètres du variogramme théorique, l'angle et l'amplitude d'une anisotropie éventuelle des données ou encore le backend utilisé pour les calculs. A noter que dans le cas où les paramètres du variogramme théorique ne sont pas fournis, ceux-ci sont calcules automatiquement par la librairie par moindres carrés. A la fin de cette étape, la librairie a en mémoire le variogramme théorique, les positions des points d'entrainement ainsi que la valeur de la grandeur physique en chacun de ces points et une estimation de l'erreur en chaque point.
- L'application de l'algorithme sur les données de test via la méthode **execute** de l'instance de krigeage choisie. Il suffit de fournir les points de test a la méthode execute, en général une grille régulière de points. La librairie calcule alors l'estimation et l'écart-type en chaque point de test.

Les résultats du krigeage se résument d'une part à des estimations de l'erreur dite «**Glossaires des** termes techniques» via la fonction plotresidualsvario qui affiche simultanément les histogrammes des résidus, de la variance et des résidus standardisés entre les données d'entrainement et les estimées aux même points (**Voir Figure 15**). Sont aussi représentées les distributions associées à ces valeurs, ainsi que la distribution gaussienne la plus proche de celles-ci. A noter que la comparaison a une distribution gaussienne n'est pas forcement concluante la distribution de la variance, il s'agissait juste d'un raccourci de programmation. Ce graphique ne peut malheureusement pas non plus fournir d'information quant à la qualité de l'estimation sur les données de test, l'**Glossaires des termes** techniques étant totalement disjointe de l'erreur d'entrainement. En revanche, elle nous fournit des informations concernant d'éventuelles dérives des données d'entrainement. Comme on peut d'ailleurs le constater en regardant l'histogramme des résidus. En effet, on constate que les résidus sont de plus en plus négativement éloignés de la moyenne au fur et à

[5] Github de pykrige : https://github.com/bsmurphy/PyKrige

mesure que l'indice des points d'entrainement augmente. En rappelant que les points d'entrainement ont été pris par un robot en mouvement, ils sont donc classés par ordre temporel croissant. On en déduit donc la présence d'une dérive négative de la profondeur au fur et à mesure du temps, soit une diminution du niveau



de la mer due à la marée.

Figure 15 : Histogrammes des résidus, de la variance et des résidus standardisés entre les données d'entrainement et les estimées aux mêmes points avant prétraitement.

D'autre part, nous pouvons aussi nous faire une idée des résultats via la méthode plotcolormesh_eststd en deux affichages correspondant à l'estimée et la variance en chaque point de test. Les résultats pour notre premier exemple de test est fourni en **Figure 17** sur laquelle on peut voir dans le premier sous-graphique l'estimée en chaque point de test c'est à dire les couleurs qui représentent le fond coloré, et la valeur originale en chaque point d'entrainement se trouvant au milieu des cercles noirs.
En regardant attentivement les valeurs des estimées en chaque point de test dans le premier sous-graphique, on constate qu'elles sont sensiblement différentes de celles des points d'entrainement en plusieurs endroits. Comme on s'y attendait en voyant le prétraitement des données (Voir **Annexe C** : **Méthode de suppression de la marée)**, la marée a bel et bien eu un effet considérable sur la qualité de l'estimation finale.

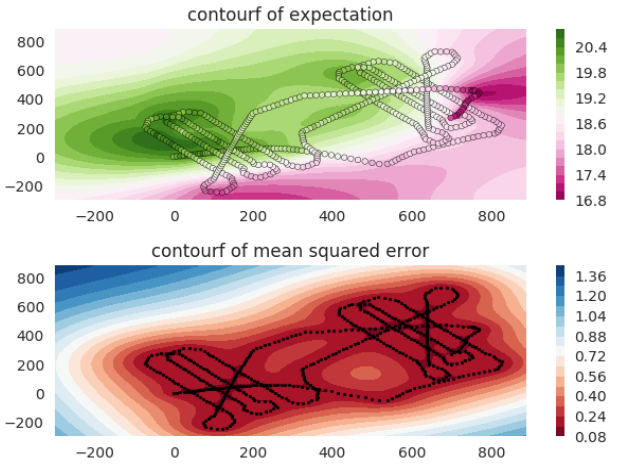


Figure 17 : Contours de l'estimée et de la variance en chaque point d'entrainement et de test sans prétraitement pour supprimer l'effet de la marée.

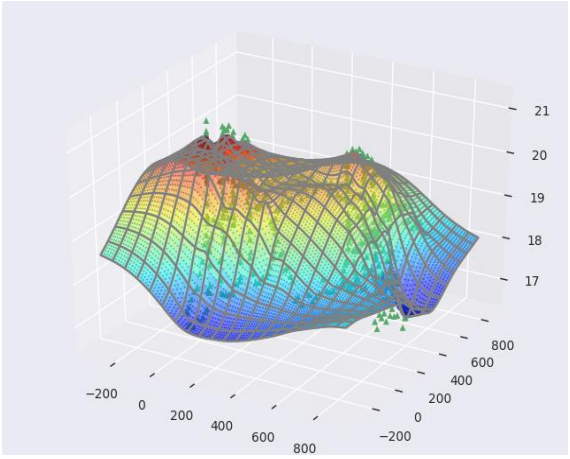


Figure 16 : Représentation 3D de la profondeur des fonds marins après krigeage sans prétraitement. Les triangles verts représentent les données d'entrainement. Les points de petite dimension représentent quant à eux la grille des points de test.

Enfin, en **Figure 16** est donnée une représentation 3D de la profondeur des fonds marins d'après les résultats du krigeage. A noter qu'étant donné que la profondeur est représentée selon l'axe montant, il faudrait inverser la figure selon l'axe z pour obtenir une cartographie des fonds marins.

Résultats sans l'effet de la marée

Comme présenté **Annexe C** : , nous avons implémenté une méthode pour tenter de supprimer au mieux les effets de la marée sur les estimations fournies par krigeage. Bien que cette technique ne permette pas d'estimer l'erreur qu'elle induit sur les données, on peut cependant se rendre compte de l'impact positif qu'elle a sur l'estimée. On peut s'en rendre compte visuellement en regardant les résultats présentes en **Figure 18**, mais aussi en regardant les résultats statistiques sur l'erreur d'entrainement fournis en **Figure 19**.

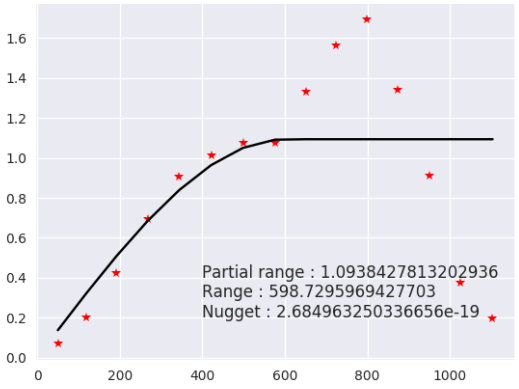


Figure 18 : Variograme empirique (étoiles rouges) et théorique (courbe noire) des données d'entrainement après prétraitement.

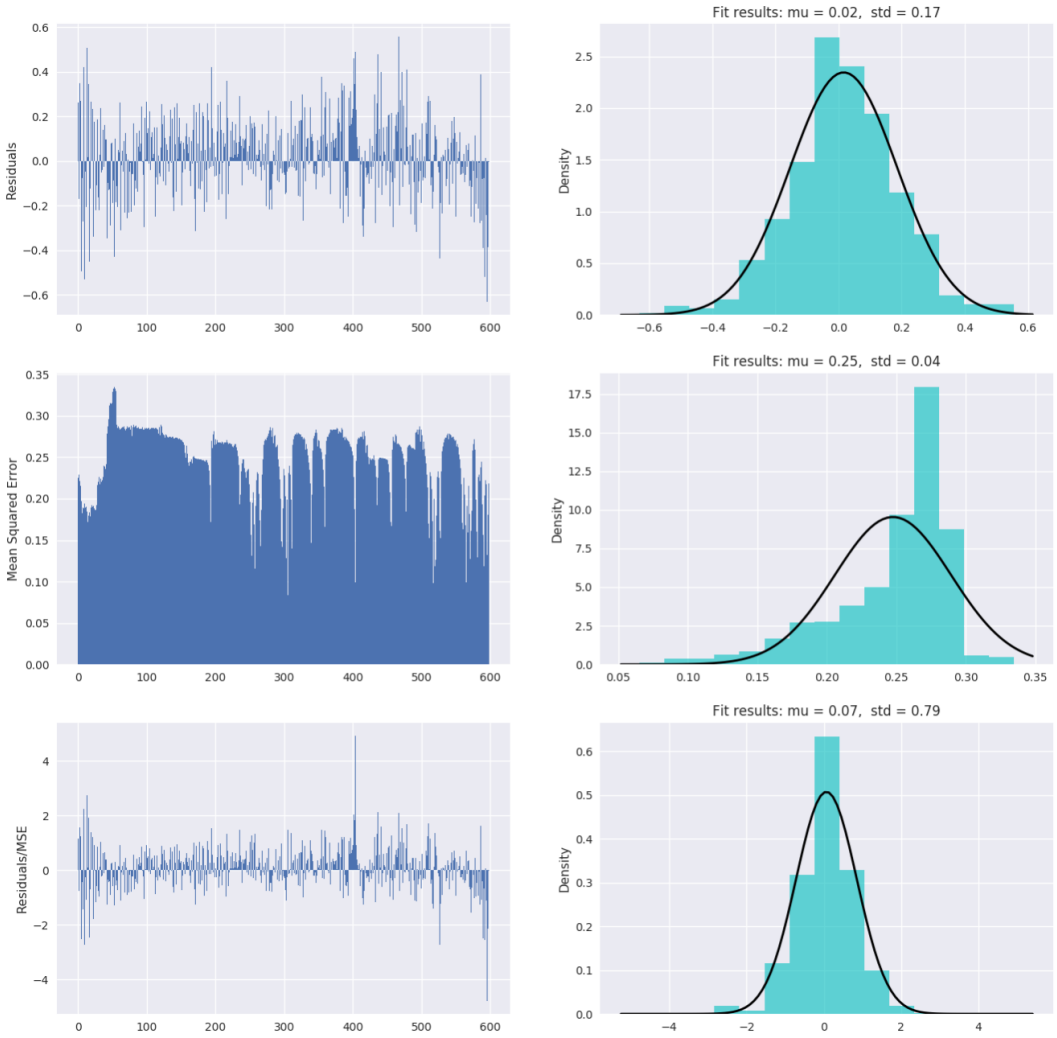


Figure 19 : Histogrammes des résidus, de la variance et des résidus standardisés entre les données d'entrainement et les estimées aux mêmes points après prétraitement.

La variance du résultat fourni n'est qu'une borne minimale de la véritable variance en chaque point de test.

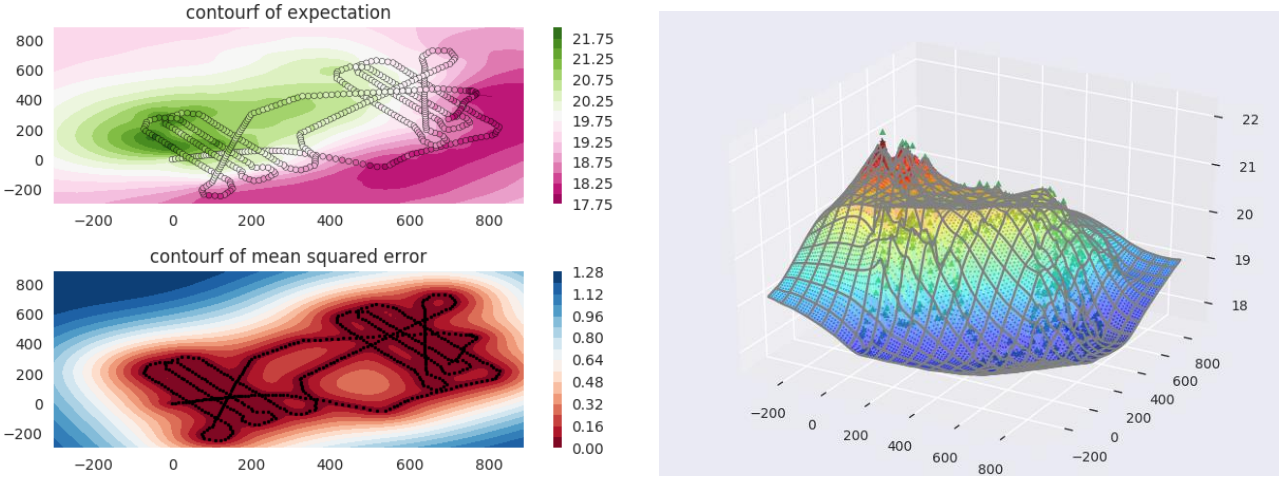


Figure 20 : Contours de l'estimée et de la variance en chaque point d'entrainement et de test et représentation 3D de la profondeur des fonds marins après prétraitement pour supprimer l'effet de la marée.

5.2.Sur données 3D partiellement simulées

A l'heure de l'écriture de ce rapport, le code utilisé pour l'estimation de la profondeur des fonds marins a été adapté pour des données en trois dimensions. Il a été testé sur un ensemble de données formées du déplacement réel du robot Boatbot du 11 Juillet 2018 mais avec des données magnétométriques simulées. Bien que les premiers résultats soient concluants, ils ne portent que sur l'erreur d'entrainement selon la méthodologie présentée dans l'exemple en deux dimensions. Les données magnétométriques ont été simulée comme la somme des interactions de dépôts ferromagnétiques placés à une profondeur constante. Il a été nécessaire de simuler ces données car celles recueillies par le magnétomètre réel n'étaient pas exploitables en raison d'une erreur dans les timestamp de ces dernières. La finalité du code serait de comparer les données estimées en chaque point d'une grille régulière avec et sans bruit additif. En répétant cette opération sur un grand nombre de répartitions de dépôts ferromagnétiques distincts, nous parviendrons à obtenir une estimation statistique de l'erreur de l'algorithme.

6. Analyse économique du ACSE

Le département de Automatic Control and Systems Engineering (ACSE) est une entité quasi indépendante de l'université de Sheffield. Elle possède ses propres services techniques et administratif. Elle tire une partie de ses fonds du département administratif principal, notamment pour l'achat de matériel ou les salaires du personnel non académique ou de certains doctorants, post-doctorants et chercheurs. Elle possède cependant plusieurs partenariats avec des entreprises qui lui fournissent une source de revenus, notamment en fournissant des thèses rémunérées.

7. Apport du stage à mes perspectives de carrière

Ce stage aura été pour moi ma première expérience dans le monde académique. J'ai eu la chance d'être accueilli par une équipe soudée et très chaleureuse qui m'auront fait partager leur quotidien pendant ces trois mois de stage. Ce stage aura été une opportunité pour moi de travailler en quasi-autonomie sous la direction de Dr L. Mihaylova sur un sujet qui m'était totalement inconnu. Cela a nécessité des efforts d'organisation de ma part, pour d'une part faire une recherche théorique sur le krigeage, et de faire l'état de l'art des programmes utiles.
J'ai aussi eu l'occasion de tester la vie dans un pays étranger pour une période prolongée.

Conclusion

Après une recherche bibliographique sur les principes du krigeage, ses domaines d'application et méthodes les plus courantes ainsi que sur le magnétisme et les incertitudes mises en jeu sur la position du magnétomètre, un programme python a été implémenté remplissant une majorité des points soulevés dans le cahier des charges. Bien que le code en charge de l'extraction des données soit terminé et testé, celui en charge du krigeage reste encore à terminer à l'heure de la rédaction de ce rapport. A noter cependant que les fonctions élémentaires constituant la partie krigeage existent déjà mais nécessitent une compréhension intime du code et des librairies python mises en jeu pour être prises en main. La partie du code manquante consiste à la mise en place d'une interface plus ergonomique pour l'utilisateur ainsi que la correction de quelques bugs.

Ce rapport passe aussi sous silence certaines pistes de recherche d'amélioration des librairies utilisées, notamment pykrige, comme la détection semi-automatique d'anisotropie dans les données ou encore la détermination des hyperparamètres du variograme théorique par validation croisée qui n'étaient pas primordiales au développement du code.

Points non abordés : anisotropies, validation croisée,

De nombreuses pistes d'amélioration du code existent, notamment la résolution de la fuite mémoire lors de l'extraction des rosbags, la parallélisation des calculs de krigeage à l'aide d'un cloud et la mise en place d'un krigeage par blocs.

Bibliographie

Alan E. Gelfand, P. J. (2010). *Handbook of Spatial Statistics.*

Boatbot. (s.d.). Récupéré sur https://www.ensta-bretagne.fr/jaulin/boatbot.html

Jean-Paul Chilès, P. D. (1999). *Geostatistics Modeling Spatial Uncertainty.*

Kiš, I. M. (2016). *Comparison of Ordinary and Universal Kriging interpolation techniques on a depth variable (a case of linear spatial trend), case study of the Šandrovac Field .*

Westra, E. (2013). *Python Geospatial Development, Second Edition.* Packt Publishing.

Williams, C. E. (2006). *Gaussian Processes for Machine Learning.*

Yuxin Zhao, C. L. (2018). Gaussian Processes for RSS Fingerprints Construction in Indoor Localization.

Glossaires des termes techniques

Isotropie	*Qui présente les mêmes caractéristiques physiques dans toutes les directions.*
DRASSM	*Département de recherches archéologiques subaquatiques et sous-marines*
Procédés Gaussiens Standards	*Ou plus simplement Processus Gaussiens : Processus stochastique sur un ensemble constitué d'une suite de variables gaussiennes.*
AUV	*Autonomous Underwater Vehicle, ou véhicule sous-marin autonome. Véhicule sous-marin non téléguidé.*
Erreur d'entrainement	*Erreur d'estimation commise par un algorithme sur les données qui ont servi à son entrainement. Des hyperparamètres déterminés uniquement par estimation de l'erreur d'entrainement sont soumis à de l'overfitting.*
Erreur de test	*Erreur d'estimation commise par un algorithme sur des données non incluses dans ses données d'entrainement (Out Of Sample data) .*
Overfitting	*La production d'une analyse qui est trop proche d'un ensemble de données particulier pour être réaliste, et risque de ce fait d'être incapable de donner des prédictions correctes pour tout autre ensemble de données.*

Table des figures

Annexe A : Précision d'un GPS

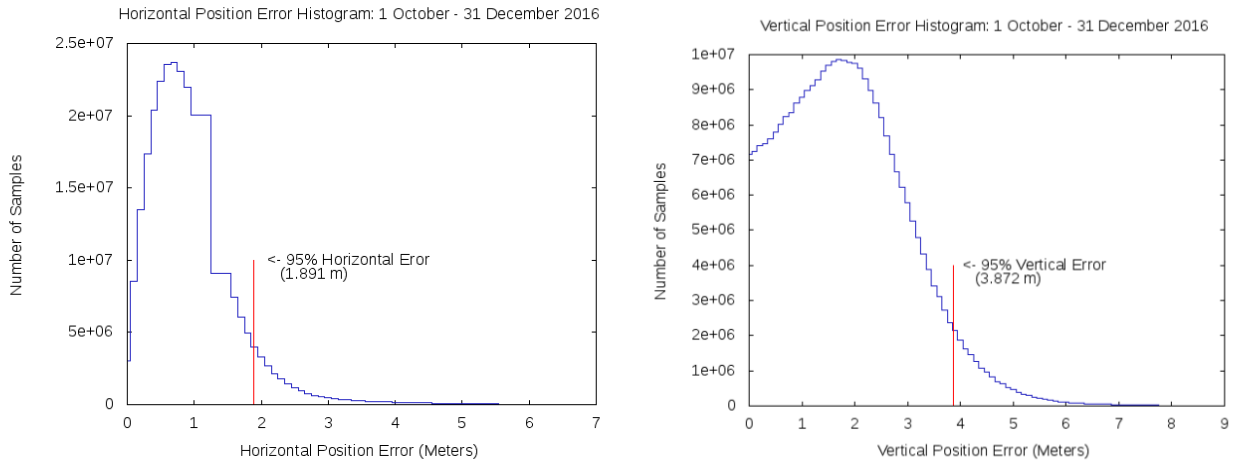


Figure 21 : Courbe de précision d'un GPS vendu dans le commerce.

Source : Site officiel du gouvernement des Etats-Unis d'Amérique d'information sur le GPS : https://www.gps.gov/systems/gps/performance/accuracy/[6]

6

Annexe B : Tableau récapitulatif des types de krigeage

	Σ	λ	σ_0	μ	σ_K
Krigeage Simple	$\Sigma = \begin{pmatrix} \gamma(l_{11}) & \cdots & \gamma(l_{1\tau}) \\ \vdots & & \vdots \\ \gamma(l_{\tau 1}) & \cdots & \gamma(l_{\tau\tau}) \end{pmatrix}$	$\vec{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_\tau \end{pmatrix}$	$\vec{\sigma_0} = \begin{pmatrix} \gamma(l_{10}) \\ \vdots \\ \gamma(l_{\tau 0}) \end{pmatrix}$	$\mu = cte = \mu_0$	$\sigma^2_{SK} = \sigma_{00} - \lambda^T \vec{\sigma_0}$
Krigeage Ordinaire	$\Sigma = \begin{pmatrix} \gamma(l_{11}) & \cdots & \gamma(l_{1\tau}) & 1 \\ \vdots & & \vdots & \vdots \\ \gamma(l_{\tau 1}) & \cdots & \gamma(l_{\tau\tau}) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix}$	$\vec{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_\tau \\ L_0 \end{pmatrix}$	$\vec{\sigma_0} = \begin{pmatrix} \gamma(l_{10}) \\ \vdots \\ \gamma(l_{\tau 0}) \\ 1 \end{pmatrix}$	$\mu = cte$	$\sigma^2_{OK} = \sigma_{00} - \lambda^T \vec{\sigma_0} = \sigma_{00} - \sum_i \lambda_i \gamma(l_{i0}) - L_0$
Krigeage Universel	$\Sigma = \begin{pmatrix} \gamma(l_{11}) & \cdots & \gamma(l_{1\tau}) & 1 & f_1^{(1)} & \cdots & f_1^{(M)} \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \gamma(l_{\tau 1}) & \cdots & \gamma(l_{\tau\tau}) & 1 & f_\tau^{(1)} & \cdots & f_\tau^{(M)} \\ 1 & \cdots & 1 & 0 & \cdots & \cdots & 0 \\ f_1^{(1)} & \cdots & f_\tau^{(1)} & \vdots & \vdots & \vdots & \vdots \\ \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_1^{(M)} & \cdots & f_\tau^{(M)} & 0 & \cdots & \cdots & 0 \end{pmatrix}$	$\vec{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_\tau \\ L_0 \\ \vdots \\ L_\tau \end{pmatrix}$	$\vec{\sigma_0} = \begin{pmatrix} \gamma(l_{10}) \\ \vdots \\ \gamma(l_{\tau 0}) \\ 1 \\ f_0^{(1)} \\ \vdots \\ f_0^{(\tau)} \end{pmatrix}$	$\mu_i = \sum_{m=1}^{M} a_m f_i(x)^{(m)}$ $\forall\, i \in [\![1, \tau]\!]$	$\sigma^2_{UK} = \sigma_{00} - \lambda^T \vec{\sigma_0} = \sigma_{00} - \sum_i \lambda_i \gamma(l_{i0}) - \sum_{i=0}^{\tau} L_i f_0^i$

Annexe C : Méthode de suppression de la marée

Présentation du problème :

L'AUV effectue des relevés bathymétriques selon des rails et croise souvent sa propre trajectoire afin de passer d'un rail un a un autre. Ce faisant, il enregistre à plusieurs reprises des données de profondeur du fond marin très proches spatialement mais qui semblent varier sensiblement entre chaque passage. Pour supprimer cet effet vraisemblablement dû a la marée, nous commençons par extraire ces points qui sont proches spatialement mais pas temporellement (Voir figure 3.1). La différence de profondeur qui est visible entre ces points est supposée directement corrélée à la différence de temps entre ceux-ci. Ainsi, en effectuant la régression linéaire de la différence de profondeur par la différence de temps entre chaque paire de ces points, on pourra estimer l'influence de la marée dans nos données dans le temps et ainsi la supprimer de nos données d'entrainement (voir figure 3.2).

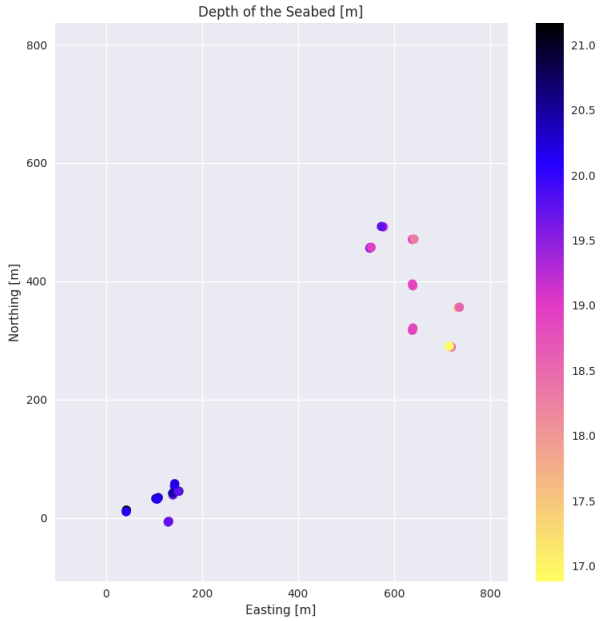


Figure 231 : Extraction des points proches spatialement mais éloignes temporellement. On constate que pour bon nombre d'entre eux, les profondeurs recueillies sont sensiblement différentes.

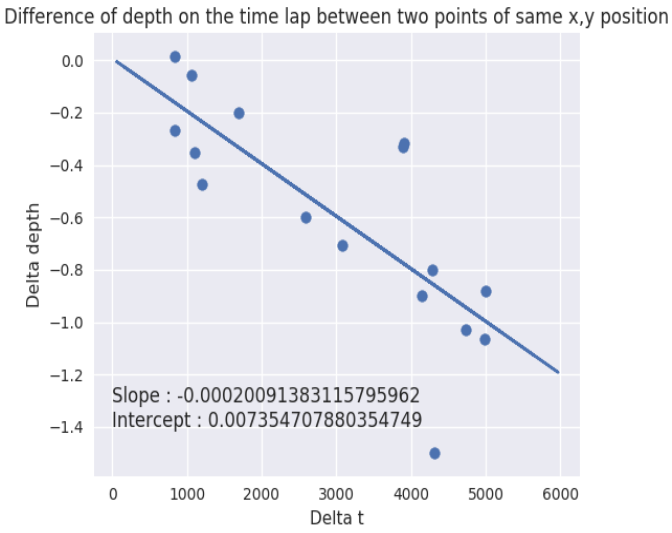


Figure B22 : Régression linéaire de la différence de profondeur en fonction de la différence de temps entre chaque paire de point présentes dans la figure B1.

Les effets de ce prétraitement sont présentés en **Figure B24** : Comparaison entre les données d'entrainement avant et après prétraitement de suppression des effets de la marée.

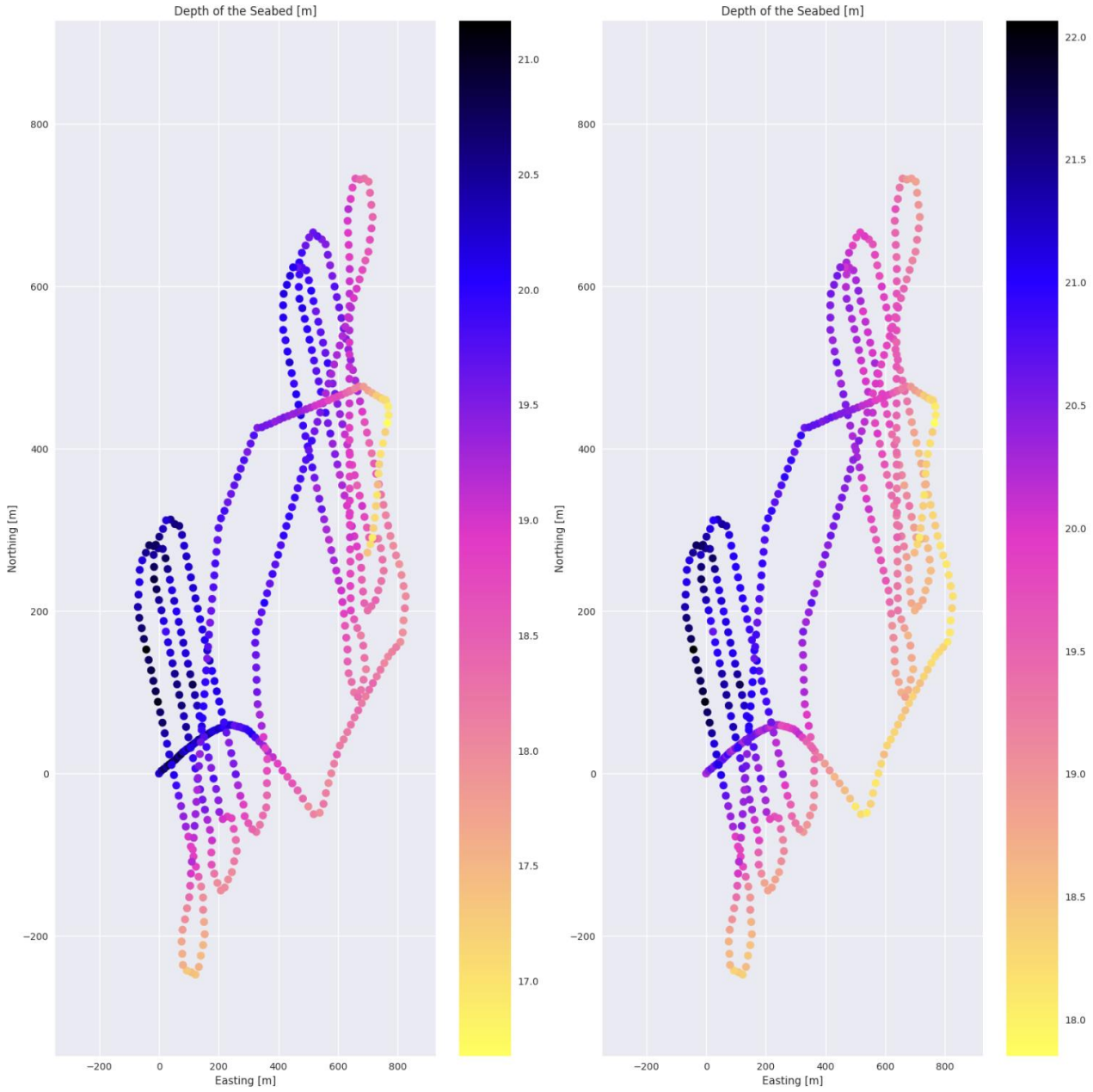


Figure B24 : Comparaison entre les données d'entrainement avant et après prétraitement de suppression des effets de la marée.

On constate une diminution de la variation des données de profondeur recueillies aux points de croisement de plusieurs trajectoires. A noter cependant que les deux schémas n'ont pas la même échelle de couleur.

Plusieurs défauts à cette technique subsistent cependant :

- La marée est un effet sinusoïdal dans le temps de période 12h. L'ensemble des données ayant été recueillies sur 1h40min, la linéarité du phénomène sur cette étendue temporelle reste discutable.
- Puisque cette méthode repose sur une régression linéaire, il n'est pas possible d'estimer précisément l'erreur induite par cette méthode, à la manière de ce qui est possible de faire avec des procédés gaussiens par exemple. De ce fait, les variances en chaque point de test qui seront données après krigeage ne seront que des bornes minimales des véritables variances en chaque point.
- Etant donné que l'on supprime la pente du phénomène de marée, l'ensemble des données de profondeur est donne relativement à la profondeur au début de la marée. Il faudrait connaitre l'heure exacte à laquelle a commencé l'expérience pour connaitre la véritable profondeur de chaque point de test.

Rapport d'évaluation

ENSTA Bretagne

RAPPORT D'EVALUATION
ASSESSMENT REPORT

Merci de retourner ce rapport en fin du stage à :
Please return this report at the end of the internship to :

ENSTA Bretagne – Bureau des stages - 2 rue François Verny - 29806 BREST cedex 9 – FRANCE
☎ 00.33 (0) 2.98.34.87.70 - Fax 00.33 (0) 2.98.38.87.90 - stages@ensta-bretagne.fr

I - ORGANISME / *HOST ORGANISATION*

NOM / *Name* UNIVERSITY OF SHEFFIELD

Adresse / *Address* AMY JOHNSON BUILDING, AUTOMATIC CONTROL & SYSTEMS ENGINEERING, MAPPIN STREET, SHEFFIELD S10 1QR

Tél / *Phone (including country and area code)* 00 44 114 222 5675

Fax / *Fax (including country and area code)* _____

Nom du superviseur / *Name of placement supervisor* LYUDMILA MIHAYLOVA

Fonction / *Function* PROFESSOR

Adresse e-mail / *E-mail address* L.S.MIHAYLOVA@SHEFFIELD.AC.UK

Nom du stagiaire accueilli / *Name of trainee* [signature]

II - EVALUATION / *ASSESSMENT*

Veuillez attribuer une note, en encerclant la lettre appropriée, pour chacune des caractéristiques suivantes. Cette note devra se situer entre **A (très bien)** et **F (très faible)**
*Please attribute a mark from **A (very good)** to **F (very weak).***

MISSION / *TASK*

- La mission de départ a-t-elle été remplie ? A B C D E F
  Was the initial contract carried out to your satisfaction?
- Manquait-il au stagiaire des connaissances ? ☑ oui/*yes* ☐ non/*no*
  Was the trainee lacking skills?

  Si oui, lesquelles ? / *If so, which skills?* _____

ESPRIT D'EQUIPE / *TEAM SPIRIT*

- Le stagiaire s'est-il bien intégré dans l'organisme d'accueil (disponible, sérieux, s'est adapté au travail en groupe) / *Did the trainee easily integrate the host organisation? (flexible, conscientious, adapted to team work)*

  (A) B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

9

Version du 03/05/2017

COMPORTEMENT AU TRAVAIL / *BEHAVIOUR TOWARDS WORK*

Le comportement du stagiaire était-il conforme à vos attentes (Ponctuel, ordonné, respectueux, soucieux de participer et d'acquérir de nouvelles connaissances) ?
Did the trainee live up to expectations? (Punctual, methodical, responsive to management instructions, attentive to quality, concerned with acquiring new skills)?

(A) B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here*
VERY DILIGENT AND FOCUSED.

INITIATIVE – AUTONOMIE / *INITIATIVE – AUTONOMY*

Le stagiaire s'est –il rapidement adapté à de nouvelles situations ? (A) B C D E F
(Proposition de solutions aux problèmes rencontrés, autonomie dans le travail, etc.)

Did the trainee adapt well to new situations? (A) B C D E F
(eg. suggested solutions to problems encountered, demonstrated autonomy in his/her job, etc.)

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here*
OLIVIER CAN WORK INDEPENDENTLY AND ACHIEVE INTERESTING RESULTS

CULTUREL – COMMUNICATION / *CULTURAL – COMMUNICATION*

Le stagiaire était-il ouvert, d'une manière générale, à la communication ? (A) B C D E F
Was the trainee open to listening and expressing himself /herself?

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here*
VERY SOCIAL AND COOPERATIVE

OPINION GLOBALE / *OVERALL ASSESSMENT*

❖ La valeur technique du stagiaire était : A B C D E F
Evaluate the technical skills of the trainee:

III - PARTENARIAT FUTUR / *FUTURE PARTNERSHIP*

❖ Etes-vous prêt à accueillir un autre stagiaire l'an prochain ?
Would you be willing to host another trainee next year? ☑ oui/*yes* ☐ non/*no*

Fait à 3/10/2018, le _____
In _____, on _____

Signature Entreprise _____ Signature stagiaire
Company stamp _____ *Trainee's signature*

PROF. L. MIHAYLOVA

Merci pour votre coopération
We thank you very much for your cooperation

10

Version du 03/05/2017