

IMT Atlantique

Technopôle de Brest-Iroise - CS 83818

29238 Brest Cedex 3

URL : www.imt-atlantique.fr



Rapport de projet 3A

Projet n°45

Tourbillons océaniques synthétiques : détection, incertitudes et assimilation de données

Guillaume Ghienne

Alexandre Lefebvre

Antoine Lerosey

Luc Ménard

Encadrants : Stéphane Raynaud (SHOM), Pierre Tandéo (IMT Atlantique)

Date d'édition : 25 mars 2021

Version : 1.0



IMT Atlantique

Bretagne-Pays de la Loire

École Mines-Télécom

Sommaire

Résumé	3
Introduction	4
1. Création d'un catalogue d'observations	4
1.1. Les données	4
1.2. Détection	4
1.2.1. Calcul des lignes de courants	5
1.2.2. Regroupement des lignes en tourbillons	7
1.2.3. Estimation des paramètres d'un tourbillon	8
1.3. Tracking	12
1.3.1. Mesure de similitude : indice de Jaccard	12
1.3.2. Tracking avec un K-means sur une fenêtre de temps	12
1.3.3. Algorithme retenu	13
1.4. Création du catalogue	14
2. Assimilation de données	15
2.1. La librairie AnDA	15
2.2. Préparation des données	17
2.2.1. Le catalogue	17
2.2.2. Les observations	17
2.3. Modification de la librairie	18
2.4. Assimilation sur les tourbillons	19
2.4.1. Dans le cas d'un unique tourbillon	19
2.4.2. Extension à l'ensemble des tourbillons	20
2.4.3. Utilisation du code	21
3. Conclusion	23
Annexes	24
Références	25

Liste des figures

1.	Données brutes de courant au jour 0	5
2.	Lignes de courants obtenues avec les trois fonctions.	6
3.	Lignes de courants complètes et tronquées avec le ratio (1).	7
4.	Regroupement des lignes de courant par tourbillons.	8
5.	Résultats du K-means pour le tracking des tourbillons.	13
6.	Illustration d'une itération de l'algorithme	14
7.	Résultats de l'algorithme implémenté pour le tracking des tourbillons	15
8.	Illustration du fonctionnement de l'opérateur de prévision analogique. [2]	16
9.	Représentation visuelle des trois méthodes de régression proposées. [2]	16
10.	Construction simplifiée des structures composant le catalogue.	17
11.	Construction simplifiée des structures d'observations.	18
12.	Exemple de résultat sur un tourbillon.	20
13.	Simulation du résultat final obtenu avec l'assimilation.	21

Résumé

Ce document présente les outils développés et les résultats obtenus à l'issue de notre projet réalisé dans le cadre de notre troisième année à l'IMT Atlantique. En partenariat avec le SHOM (Service Hydrographique et Océanographique de la Marine), ce projet a pour but d'appliquer une méthode d'assimilation de données orientée données à la correction de modèles de prédictions de l'évolution des tourbillons océaniques. Il s'agira dans un premier temps de construire un catalogue de tourbillons synthétiques à partir de données réelles de courants afin de, dans un second temps, l'utiliser au sein d'une méthode d'assimilation de données appelée Analog Data Assimilation [2].

Introduction

Les tourbillons océaniques sont des structures approximativement elliptiques dans les courants marins qui s'observent également via les anomalies de températures, de niveau et de composition de l'eau qu'ils contiennent. Leur taille et leur durée de vie varient de quelques dizaines à plusieurs centaines de kilomètres et de quelques jours à plusieurs mois. En raison de la puissance des courants associés et des grandes masses d'eau anormalement froide ou chaude qu'ils transportent, la prévision de ces tourbillons est un enjeu important pour tous les secteurs d'activité impactés par les courants océaniques. Les applications concrètes comptent, entre autres, le camouflage acoustique des sous-marins, l'optimisation des routes commerciales ou encore l'étude du développement du plancton et de la faune qui en découle en raison du transport de nutriments vers la surface.

Les modèles actuels peinent à simuler précisément les courants. La résolution et la précision insuffisante des grilles de données, les approximations nécessaires pour que les simulations tournent en un temps raisonnable et plus généralement la nature chaotique des phénomènes en jeu conduisent à des prévisions imparfaites avec un décalage par rapport à la réalité. L'idée explorée lors de notre projet est l'utilisation de l'Analog Data Assimilation [2](AnDA), une méthode d'assimilation de données orientée données utilisant un catalogue d'observations passées afin de corriger ou prédire des observations futures.

La première partie présentée dans ce rapport traite de la création d'un catalogue de tourbillons en partant de données réelles de courants brutes. Plus précisément, on y présentera les outils implémentés pour obtenir dans un premier temps les lignes de courants puis pour détecter les tourbillons océaniques à une date donnée et leur associer une incertitude de mesure. La deuxième partie présente rapidement les principes de la librairie AnDA [2] et comment nous l'avons adaptée à la prévision de l'évolution des tourbillons océaniques sur plusieurs jours. Dans cette partie seront également présents quelques exemples généraux sur le fonctionnement théorique de notre implémentation. On finira ce rapport par une petite conclusion revenant brièvement sur quelques points qui nous semblent importants.

1. Création d'un catalogue d'observations

Un catalogue d'observations sur lequel se base une assimilation de données consiste en un ensemble d'observations datées et auxquelles sont associées des incertitudes. Ces incertitudes caractérisent, pour une date et un paramètre donnés, le niveau de confiance que l'on peut accorder à chaque observation. Dans le cas présent, les observations correspondent aux caractéristiques quotidiennes de tourbillons océaniques sur une durée d'un peu moins de trois ans.

1.1. Les données

Les données utilisées pour créer le catalogue sont les moyennes journalières de vitesse en m.s^{-1} des courants océaniques vers le Nord et vers l'Est. Elles sont réparties sur une grille Arakawa de type C. Ces moyennes de vitesse sont nommées respectivement u_o et v_o . Toutes les détails liés au format des données peuvent être trouvés dans le [manuel utilisateur](#) de Copernicus. Le domaine d'étude est la mer d'Arabie sur un peu moins de trois années avec une résolution de $1/12^\circ$:

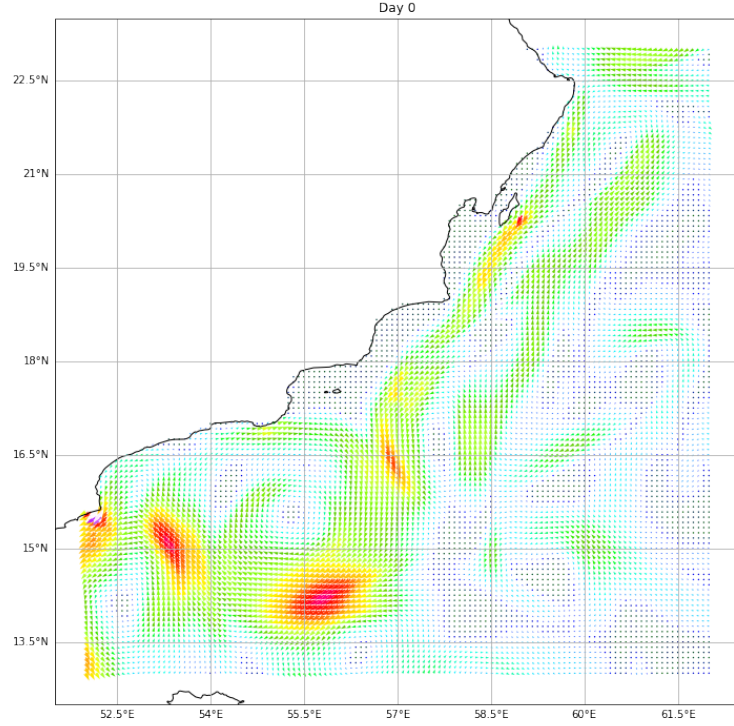
time : du 2018-01-01 au 2020-10-14

longitude : de 52°E à 62°E

latitude : de 13°N à 23°N

1.2. Détection

La première étape est la détection des tourbillons pour chaque jour observé. À ce stade, chaque jour de données brutes est traité indépendamment des autres.



L'ensemble des données brutes est constitué d'une double grille (une par coordonnée des vecteurs vitesse) pour chaque jour de données.

FIGURE 1 – Données brutes de courant au jour 0

1.2.1. Calcul des lignes de courants

Le format des données brutes (un champs discret de vitesses pour chaque jour) n'est pas le plus adapté pour extraire les informations pertinentes. Les lignes de courant sont visuellement plus faciles à interpréter, et une caractéristique facile à utiliser pour repérer un tourbillon est leur enroulement. Elles sont obtenues par intégration de la vitesse du courant $\vec{v}(x, y, t)$ figé à la date d'observation d_{obs} . Pour une ligne de courant $l : s \mapsto l(s) = (x(s), y(s))$ paramétrée par un temps s , on a donc la relation :

$$l(s) = l(0) + \int_0^s \vec{v}(x(t), y(t), d_{obs}) dt$$

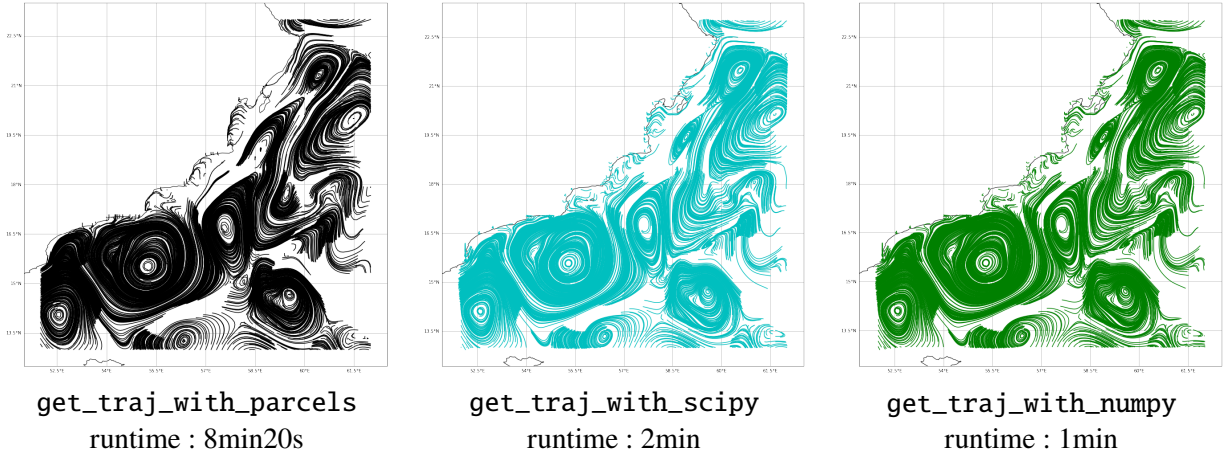
Le temps réel d_{obs} étant constant, le temps s qui apparait dans la relation n'a pas de réalité physique, mais il peut être interprété comme la durée nécessaire pour qu'une particule initialement en $l(0)$ dérive avec le courant jusqu'en $l(s)$ si le courant était invariant avec le temps.

Le fichier `./code/eddies_detection.py` contient `get_traj_with_parcel`, `get_traj_with_scipy` et `get_traj_with_numpy`, trois fonctions qui effectuent l'interpolation du champs discret de vitesses à la date fixée et l'intégration de plusieurs ligne de courant selon la méthode de Runge-Kutta d'ordre quatre (RK4) à partir de positions initiales $l(0)$ disposées en grille. Les paramètres principaux de ces fonctions sont :

- `date` : Date d'observation en nombre de jours relativement au premier jour contenu dans le dataset.

- **runtime** : Durée d'intégration en heures. La longueur moyenne des lignes est proportionnelle à cette durée.
- **delta_time** ou **max_delta_time** : Pas de temps dt en heures pour l'intégration.
- **particle_grid_step** : Pas en indice de la grille de positions initiales $l_{0 \leq k \leq K}$ par rapport à la grille de données de courant. Permet de régler la densité des lignes de courant. Influence le temps d'exécution en $O(n^2)$

Les lignes obtenues sont quasiment identiques quelle que soit la fonction mais elles sont tout de même légèrement moins régulières pour la fonction utilisant la librairie *parcels* (Figure 2). En revanche la fonction utilisant *numpy* uniquement est bien plus rapide que les 2 autres, les résultats qui seront présentés par la suite sont donc issus de celle-ci.



Les temps d'exécution ont été mesurés avec les mêmes conditions dans les 3 cas.
date=0, runtime=600, delta_time=5, particle_grid_step=2

FIGURE 2 – Lignes de courants obtenues avec les trois fonctions.

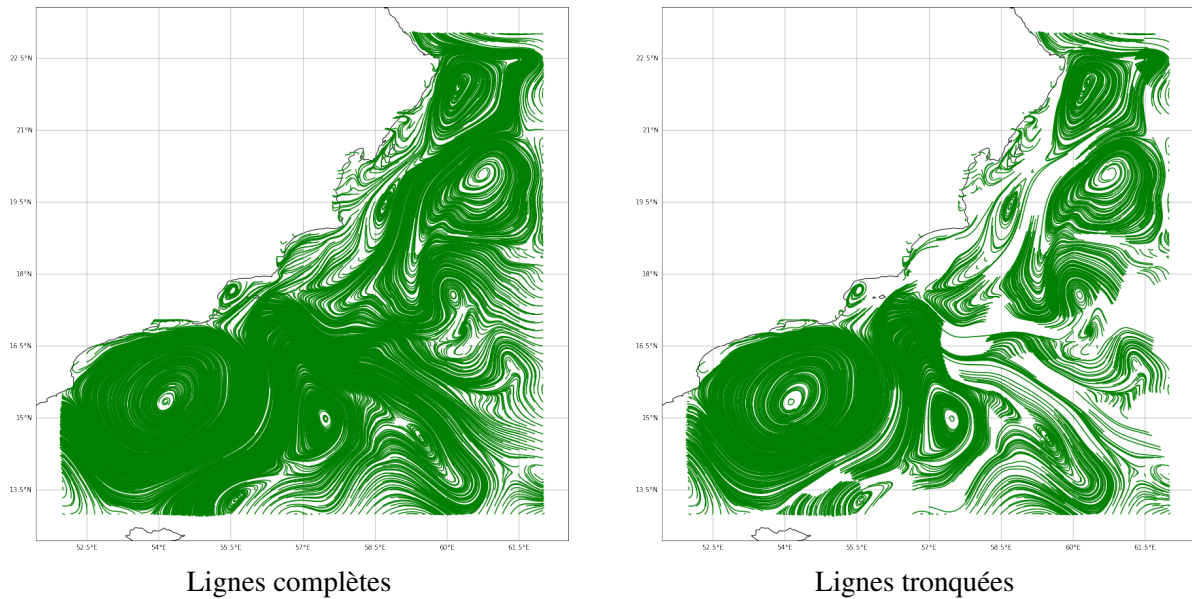
Les objets retournés par ces fonctions sont des listes de *StreamLine* définies dans *./code/classes.py*. Elles sont construites à partir d'une liste de coordonnées représentant une ligne de courant. Lors de l'initialisation d'une instance, la liste de coordonnées est tronquée pour ne garder que la partie qui ressemble le plus à une boucle en minimisant le ratio suivant :

$$r(i, j) = \frac{\|l(j) - l(i)\|}{\int_i^j \|\vec{dl}\|} \quad (1)$$

Si une partie de la ligne de courant entre les coordonnées d'indice i et j est rectiligne, ce ratio sera proche de 1. À l'inverse si la partie de la ligne entre i et j est une boucle, avec $l(i)$ et $l(j)$ proches l'un de l'autre, $r(i, j)$ sera proche de 0. Ce traitement permet de réduire le biais des calculs qui nécessitent d'étudier la répartition spatiale des points dans la ligne, et éventuellement dans le tourbillon si la ligne est retenue par la suite. La Figure 3 compare les lignes obtenues avec et sans ce traitement. Une autre méthode consiste à minimiser la longueur de la ligne tout en conservant un angle d'enroulement de 2π (lorsque c'est possible, ce qui n'est pas le cas des lignes qui ne s'enroulent pas) :

$$(i, j) = \begin{cases} \text{if } \Pi_l \neq \emptyset \text{ then } \min_{(i,j) \in \Pi_l} \int_i^j \|\vec{dl}\| \\ \text{if } \Pi_l = \emptyset \text{ then } (0, \text{maxi}) \end{cases} \quad \text{where } \Pi_l = \left\{ (i, j) \mid \left\| \int_i^j \theta \left(\frac{\vec{dl}}{dt}(t), \frac{\vec{dl}}{dt}(t + dt) \right) \right\| > 2\pi \right\}$$

Cette méthode n'a pas été retenue car elle est plus coûteuse en calculs et les résultats produits n'étaient pas toujours ceux attendus (la mesure de l'enroulement est très sensible aux déformations qui ne rendent pas la boucle parfaitement elliptique).



date=14, runtime=600, delta_time=5, particle_gride_step=2

FIGURE 3 – Lignes de courants complètes et tronquées avec le ratio (1).

1.2.2. Regroupement des lignes en tourbillons

Les lignes de courant obtenues pour le jour d'observation donné sont retournées par une des fonctions précédentes sous forme d'une liste d'objet `StreamLine`. Cette liste est divisée en plusieurs parties pour regrouper les lignes appartenant à un même tourbillon. Cette division est effectuée par la fonction `find_eddies` définie dans `./code/eddies_detection.py` qui prend en paramètre une liste de `StreamLine` et retourne une liste d'Eddy (définie également dans `./code/eddies_detection.py`).

En plus de la liste de coordonnées constituant la ligne de courant, un objet `StreamLine` contient des attributs calculés à son initialisation et utiles pour définir l'algorithme de partage de la liste :

- `coord_list` : La liste des coordonnées
- `mean_pos` : La moyenne des coordonnées. Pour une ligne avec un enroulement de 2π , cette position est une estimation du centre du tourbillon.
- `winding_angle` : L'angle orienté d'enroulement de la ligne.

Il contient également une méthode `get_mean_radius` qui retourne l'écart-type des coordonnées. Pour une ligne avec un enroulement de 2π , cette distance représente le rayon moyen de la boucle.

Avec ces caractéristiques, les lignes de courant peuvent être assignées à des clusters en utilisant un algorithme basé sur le K-means. Pour chaque cluster on garde à jour des attributs `centroid` et `max_radius` qui correspondent respectivement à la moyenne des points à et la distance maximale du centroïde avec des lignes de courant déjà assignées au cluster. Si une ligne de courant est trop éloignée des centroïdes existants, un nouveau cluster est créé. Lorsque l'algorithme se termine, ces attributs représentent le centre et le rayon du tourbillon.

Algorithme de clustering des lignes de courant en tourbillons

```
list_eddies = []
for l in list_streamlines:
```



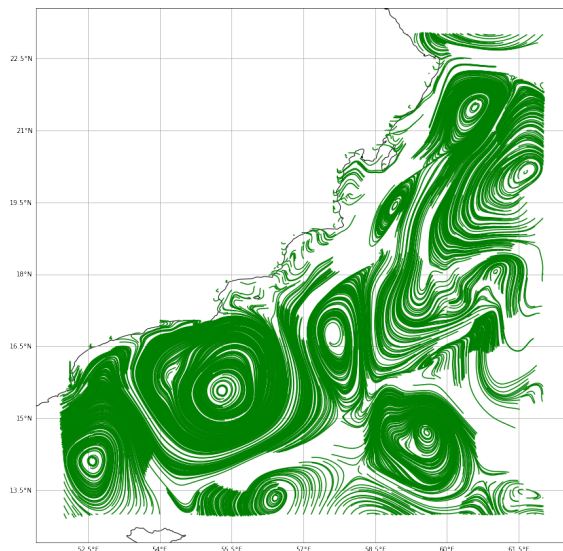
```

if l.winding_angle <= 2*pi*alpha:
    delet l
else
    distances = [|l.mean_pos - eddy.centroid|
                 for eddy in list_eddies]
    d = min(distances)
    eddy = list_eddies[argmin(distances)]

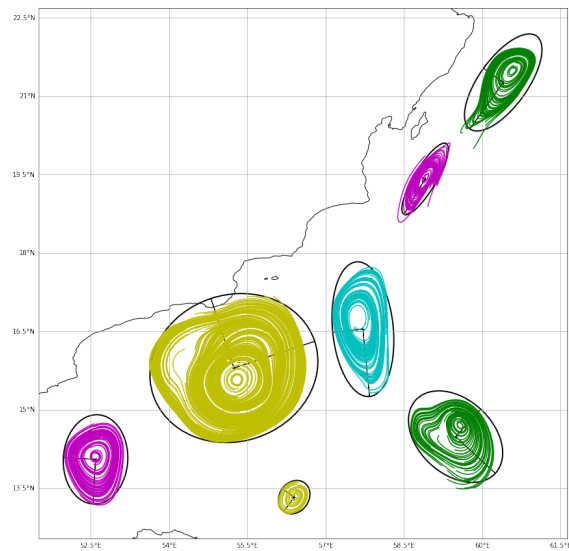
    if d < l.get_mean_radius or d < eddy.max_radius
        eddy.append(l)          # Add the streamline to
    else:
        list_eddies.append([l]) # Create a new eddy

```

Le coefficient réel α permet d'ajuster une tolérance sur l'enroulement minimum des lignes de courant qui sont retenues pour représenter les tourbillons. Ce coefficient est actuellement fixé à 0.9 dans le code.



Lignes de courants obtenues avec numpy.



Résultat de la fonction find_eddies

Les calculs permettant d'obtenir les paramètres des ellipses superposées aux lignes de courant sont présentés dans le paragraphe (1.2.3). numpy, date=0, runtime=600, delta_time=5, particle_gride_step=2

FIGURE 4 – Regroupement des lignes de courant par tourbillons.

Avant d'instancier les objets Eddy à partir des clusters obtenus avec cet algorithme, ceux qui contiennent trop peu de lignes sont supprimés. Les tourbillons dont les axes de l'ellipse qui les représente sont trop courts sont également supprimés. Les valeurs de ces deux paramètres qui permettent de filtrer les "tourbillons-bruits" d'observation peuvent être modifiées dans le fichier `.code/constants.py`.

1.2.3. Estimation des paramètres d'un tourbillon

Les tourbillons océaniques sont des structures au sein du courant liées à la hauteur de l'eau et dont la modélisation découle - dans le cas idéal - d'une gaussienne bidimensionnelle. Cette modélisation permet de représenter chaque tourbillon observé avec peu de paramètres dans le catalogue. Dans un premier temps nous avons utilisé une représentation elliptique des tourbillons, sans utiliser les équations liants la structure gaussienne et les courants induits. Dans un second temps, nous avons essayé de compléter le modèle elliptique pour obtenir tous les paramètres d'une modélisation gaussienne.

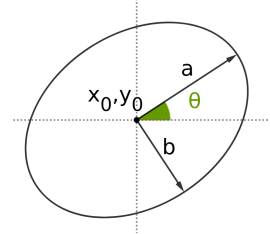
Modélisation par une ellipse

La modélisation d'un tourbillon dans le cas idéal étant lié à une gaussienne (2), on peut également utiliser une représentation elliptique où la direction des axes est identique à ceux de la gaussienne et où leurs longueurs sont proportionnelles aux paramètres σ_x et σ_y .

$$\eta = \eta_0 e^{\frac{-(x-x_0)^2}{2\sigma_x^2} - \frac{-(y-y_0)^2}{2\sigma_y^2}} \quad (2)$$

La représentation d'un tourbillon se réduit alors à ces paramètres :

- (x_0, y_0) : les coordonnées du centre.
- (a, b) : les longueurs des axes.
- θ : l'angle entre le premier axe (de longueur a) et l'axe des longitudes (parallèle à l'équateur).



Coordonnées du centre

Les coordonnées du centre correspondent au centroïde défini dans la partie 1.2.2.

Angle des axes

Les directions (\vec{u}, \vec{v}) des axes peuvent être obtenues en calculant les vecteurs propres de la matrice de covariance de l'ensemble des points des lignes de courant du tourbillon. L'angle retenu est celui entre \vec{u} et l'axe des longitudes (parallèle à l'équateur). Les valeurs propres ne sont pas proportionnelles à la longueur des axes et ne peuvent pas être utilisées pour les déterminer.

Longueur des axes

Pour passer des coordonnées (x, y) dans le repère incliné d'un angle θ aux coordonnées (X, Y) dans le repère cartésien et inversement, on définit les matrices de rotation suivantes :

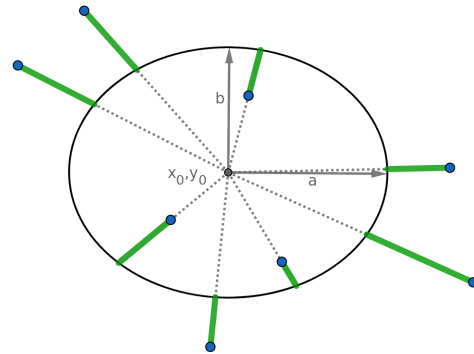
$$r_\theta = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}$$

Cette matrice permet de passer des coordonnées (X, Y) aux coordonnées (x, y) en effectuant une rotation d'angle θ . On a donc :

$$\begin{pmatrix} x \\ y \end{pmatrix} = r_\theta \begin{pmatrix} X \\ Y \end{pmatrix} \text{ et } \begin{pmatrix} X \\ Y \end{pmatrix} = r_{-\theta} \begin{pmatrix} x \\ y \end{pmatrix}$$

On suppose que le changement de repère a été effectué et que les axes de l'ellipse sont alignés avec les axes du repère. La longueur des axes est déterminée à l'aide d'une régression, en minimisant la somme des distances au carré des points p du tourbillon e avec une courbe (en l'occurrence une ellipse) :

$$(a, b) = \min_{(c, d)} \sum_{p \in e} \left(\sqrt{\frac{x_p^2}{c^2} + \frac{y_p^2}{d^2}} - 1 \right)^2$$



Ce minimum est obtenu en effectuant une descente de gradient sur les longueurs des deux axes lors de l'initialisation des objets Eddy.

Modélisation par un tourbillon géostrophique gaussien

Un tourbillon géostrophique gaussien parfait est caractérisé par les paramètres suivants :

- (x_0, y_0) : les coordonnées du centre peuvent être estimées avec la moyenne des points qui constituent les lignes de courant du tourbillon.
- θ : l'angle des axes de la gaussienne avec les axes cartésiens peut être estimé avec les vecteurs propres de la matrice de covariance des points qui constituent les lignes de courant du tourbillon.
- η_0 : le niveau de la mer au centre du tourbillon.
- σ_x, σ_y : les dimensions caractéristiques de la gaussienne.

Il reste donc à estimer η_0, σ_x et σ_y . On peut toutefois noter que la longueur des axes (a, b) de la modélisation elliptique et (σ_x, σ_y) sont liés linéairement.

On peut donner l'expression théorique des courants u (zonal) et v (méridional) d'un tourbillon en un point de coordonnées (X, Y) en fonction de la valeur du niveau de la mer en ce point.

$$\begin{pmatrix} v \\ -u \end{pmatrix} = \frac{g}{f} \vec{\nabla} \eta \quad (3)$$

où :

- $f = 10^{-4} \text{rad.s}^{-1}$ est le paramètre de Coriolis.
- $g = 9,82 \text{m.s}^{-2}$ est l'intensité locale de la pesanteur.

Puisque l'on connaît les coordonnées de l'ensemble des points qui constituent un tourbillon dans le référentiel de ce tourbillon (référentiel orienté d'un angle θ), on cherche à exprimer le niveau de la mer en fonction des coordonnées dans le repère cartésien. On trouve l'expression suivante :

$$\eta = \eta_0 e^{-(a(X-X_0)^2 - 2b(X-X_0)(Y-Y_0) + c(Y-Y_0)^2)} \quad (4)$$

où :

$$\begin{cases} a = \frac{\cos^2 \theta}{2\sigma_X^2} + \frac{\sin^2 \theta}{2\sigma_Y^2} \\ b = \frac{-\sin 2\theta}{4\sigma_X^2} + \frac{\sin 2\theta}{4\sigma_Y^2} \\ c = \frac{\sin^2 \theta}{2\sigma_X^2} + \frac{\cos^2 \theta}{2\sigma_Y^2} \end{cases}$$

En exprimant la dérivée de η avec (3) et (4) on trouve les expressions des courants suivantes :

$$\begin{cases} u = -2\frac{g}{f} [b(X - X_0) - c(Y - Y_0)] \eta(X, Y) \\ v = 2\frac{g}{f} [b(Y - Y_0) - a(Y - Y_0)] \eta(X, Y) \end{cases} \quad (5)$$

Pour calculer la valeur de ces courants théoriques, il nous manque la valeur du niveau de la mer au centre du tourbillon, η_0 . On fait l'hypothèse d'un tourbillon circulaire et on passe en coordonnées cylindriques. L'expression du niveau de la mer devient alors, après changement de repère :

$$\eta(r, \alpha) = \eta_0 e^{-\frac{r^2}{R^2}} \quad (6)$$

avec :

$$\frac{1}{R^2} = a \cos^2 \alpha - 2b \cos \alpha \sin \alpha + c \sin^2 \alpha \quad (7)$$

Sachant que $\mathbf{v} = \|\vec{u} + \vec{v}\|$, le changement de repère en coordonnées cylindriques donne :

$$\mathbf{v}(r, \alpha) = 2 \frac{g}{f} r \eta(r, \alpha) \sqrt{(-b \cos \alpha + c \sin \alpha)^2 + (a \cos \alpha - b \sin \alpha)^2} \quad (8)$$

On cherche le rayon r pour lequel la vitesse est maximale, on trouve :

$$\frac{\partial \mathbf{v}(r, \alpha)}{\partial r} = 0 \Leftrightarrow r = \frac{R}{\sqrt{2}} \quad (9)$$

En prenant arbitrairement $\alpha = \theta + \frac{\pi}{2}$ on mesure la valeur \mathbf{v}_{max} en R_Y .

On a alors :

$$\mathbf{v}_{max} = \frac{g}{f R_Y} |\eta_0| e^{-0.5} \quad (10)$$

ce qui nous permet de déduire $|\eta_0|$:

$$|\eta_0| = \frac{f}{g} R_Y e^{0.5} \mathbf{v}_{max} \quad (11)$$

Pour déterminer le signe de η_0 on utilise le sens de rotation du tourbillon précédemment déterminé ainsi que le fait que l'on se trouve dans l'hémisphère nord. Dans l'hémisphère nord les tourbillons chauds (anomalie positive du niveau de la mer) tournent dans le sens horaire, alors que les tourbillons froids (anomalie négative) tournent le sens anti-horaire.

Dès que l'on a estimé le niveau de la mer au centre du tourbillon, on peut en dériver les courants géostrophiques u et v induits par le tourbillon en utilisant les relations (3). On peut alors définir une distance entre les courants théoriques calculés et les courants mesurés :

$$L_1 = \sum_{i=1}^N (||u_{eval,i} - u_{measured,i}||^2 + ||v_{eval,i} - v_{measured,i}||^2) \quad (12)$$

où :

- N : nombre de points sur lequel on effectue l'étude
- u_{eval}, v_{eval} : courants théoriques calculés à l'aide des paramètres
- $u_{measured}, v_{measured}$: courants réels mesurés (valeurs après interpolation)

Dans un second temps on effectue une descente de gradient sur cette distance par rapport aux paramètres que l'on veut optimiser. Dans notre cas, on cherche à optimiser les paramètres R_x et R_y :

$$\begin{pmatrix} R_x \\ R_y \end{pmatrix} \leftarrow \begin{pmatrix} R_x \\ R_y \end{pmatrix} - \lambda * \begin{pmatrix} \frac{\partial L_1}{\partial R_x} \\ \frac{\partial L_1}{\partial R_y} \end{pmatrix} \quad (13)$$

où λ est un réel positif correspondant au pas de descente du gradient.

Lorsque l'on effectue la descente de gradient sur les différents tourbillons on observe bien une diminution du coût L_1 mais l'affichage des tourbillons sur la carte avec les paramètres calculés ne donne pas les résultats espérés. Nous avons identifié plusieurs possibilités pour expliquer cette erreur :

- Erreur dans le code ou l'algorithme de la descente de gradient ;
- La fonction de coût L_1 que l'on utilise pourrait ne pas être adaptée. On pourrait également prendre en compte des données SSH pour réduire l'incertitude liées aux mesures ;
- L'interpolation faite de la valeur des courants u et v en tout point à partir des valeurs discrètes à partir desquelles nous travaillons pourrait ne pas être assez précise ;
- Les autres paramètres du tourbillon pourraient également être optimisés simultanément. Les longueurs des axes (a, b) de la modélisation elliptique peuvent être utiliser au cours de l'optimisation, (σ_x, σ_y) et (a, b) étant linéairement liés ;

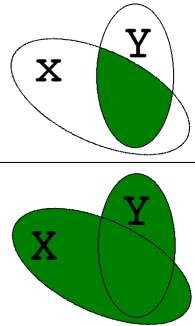
Malheureusement, par manque de temps nous n'avons pas eu l'opportunité d'explorer toutes ces possibilités. Cependant notre code étant commenté et un certains nombre de fonctions étant déjà implémentées, notre travail pourrait servir de base de travail pour résoudre ce problème.

1.3. Tracking

L'intérêt du catalogue est de fournir des exemples d'évolution dans le temps des paramètres d'un tourbillon. Il est donc nécessaire, pour chacune des dates d'observation, de faire correspondre les tourbillons observés avec ceux des dates précédentes et déterminer éventuellement quels tourbillons viennent de se former ou de disparaître. Des problèmes se posent, notamment pour gérer les cas où un tourbillon n'est pas détecté pendant un voire plusieurs jours consécutifs.

1.3.1. Mesure de similitude : indice de Jaccard

L'indice de Jaccard est une mesure de similitude entre deux ensembles X et Y à valeur dans $[0, 1]$ et qui vaut 0 pour deux ensembles disjoints et 1 pour deux ensembles identiques. Il est définie par :

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} \quad \text{J} (X, Y) = \frac{\text{Diagram 1}}{\text{Diagram 2}}$$


Pour la modélisation elliptique des tourbillons, les ensembles X et Y représentent les surfaces des ellipses. L'indice est calculé par la fonction `eddies_jaccard_index` définie dans `./code/metrics.py` et prend en paramètre deux objets `Eddy`. L'aire des deux ellipses est connue ($A = \pi ab$) et l'aire de l'intersection est estimée avec une grille.

1.3.2. Tracking avec un K-means sur une fenêtre de temps

Algorithme K-means

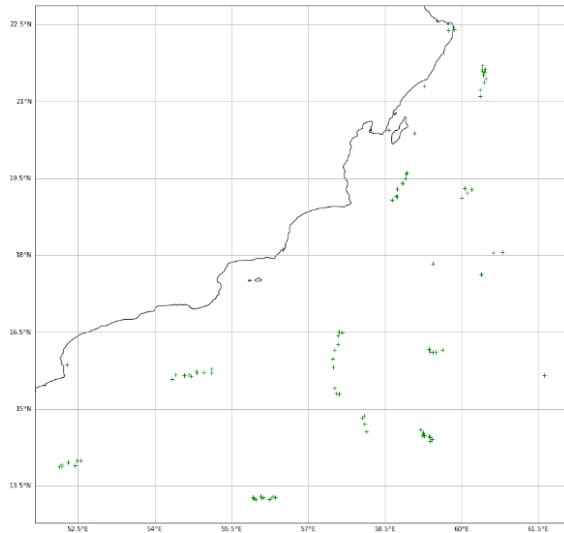
```
Initialiser les  $c_{0,k}$ 
Tant que  $l_i \neq l_{i-1}$  :
|    $i += 1$ 
|    $\forall x_n, l_i(x_n) = \operatorname{argmin}_k d(x_n, c_{i,k})$ 
|    $\forall k$ , recalculer  $c_{i,k}$ 
```

Le K-means est un algorithme itératif de clustering parmi les plus basiques. L'idée derrière cette méthode est de partitionner les points $x_{n1 \leq n \leq N}$ dans K clusters de façon à minimiser la distance d entre chaque point et le barycentre c_k (ou centroïde) du cluster auquel le point a été assigné. Le cluster auquel est assigné un point est noté $l_k(x_n)$.

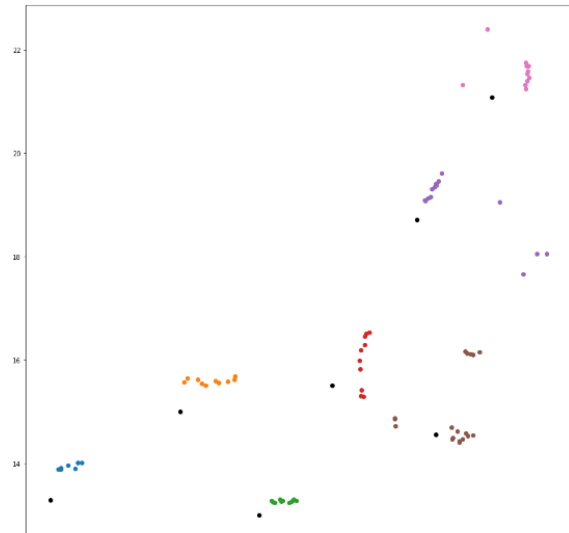
Dans notre cas, on choisit un nombre de clusters K égale au nombre de tourbillons observés au premier jour de la fenêtre de temps. Les points sont les coordonnées du centre des tourbillons observés dans la fenêtre, les $c_{0,k}$ sont les positions des tourbillons au premier jour de la fenêtre et la distance d est la distance euclidienne classique.

Utiliser une distance qui tient compte de plusieurs autres paramètres des tourbillons (comme l'indice de Jaccard que nous avons utilisé par la suite) aurait été probablement plus efficace. Mais le K-means a mis en avant certains inconvénients qui nous ont amenés à trouver un autre algorithme de tracking avant de chercher à remplacer la distance :

- Le nombre de cluster K fixé au premier jour de la fenêtre de temps implique qu'il n'y ait pas de formation ou de disparition de tourbillon sur toute la durée de la fenêtre, ce qui est rarement le cas (cluster rose de la Figure 5).
- Les 'outsiders' sont traités comme n'importe quel autre point et il n'est pas possible de distinguer les observations pertinentes du bruit et des petits tourbillons qui ne durent qu'un ou deux jours comme l'illustre le cluster rose de la Figure 5.



Centres de tous les tourbillons détectés pendant une fenêtre de 10 jours.



Clusters obtenus avec le K-means.

En approximation on peut considérer que chaque couleur sur la figure de droite correspond à un tourbillon. On peut noter que le cluster marron est partagé en deux voire trois groupes distincts parce qu'il correspond probablement à plusieurs tourbillons. Le cluster violet contient quelques points très éloignés du reste du groupe qui ne correspondent pas au même tourbillon.

FIGURE 5 – Résultats du K-means pour le tracking des tourbillons.

- La fenêtre de temps ne doit pas être "trop grande" pour éviter que les trajectoires de deux tourbillons ne se croisent pendant la durée de la fenêtre. Si c'est le cas, les points seront trop proches pour être assignés à des clusters distincts.
- Avec cette méthode on pourrait avoir des résultats aberrants, comme deux observations à la même date assignées au même cluster, ce qui signifierait qu'un tourbillon a été observé à deux endroits à la même date.

1.3.3. Algorithme retenu

L'algorithme retenu est implémenté avec la fonction `eddies_tracker` dans `./code/eddies_tracking.py` et tient compte des inconvénients mentionnés précédemment :

- Le nombre de clusters est variable et ne dépend que des observations passées en argument, bien que certains paramètres puissent suffisamment influencer les résultats pour créer/supprimer un cluster.
- Un tourbillon est observé au plus une fois par jour.
- Des paramètres permettent de fixer une durée de vie minimum pour qu'un tourbillon soit retenu et éviter que les trajectoires ne se croisent (ou se ne se rapprochent trop) à cause d'une fenêtre de temps trop grande tout en prenant en compte qu'un tourbillon pourrait ne pas être détecté pendant un ou plusieurs jours consécutifs.

L'algorithme est itératif sur le nombre total de jours d'observation sur lesquels on l'exécute. Il est initialisé à $j = 0$, répète la même étape jusqu'au dernier jour puis quelques opérations de filtrage sont effectuées sur les clusters obtenus.

Initialisation Pour chaque tourbillon observé à $j = 0$, un cluster est initialisé avec ce tourbillon.

Les cercles représentent des tourbillons observés à différentes dates. Chaque colonne représente un jour, les couleurs représentent un cluster (un même tourbillon observé une ou plusieurs fois à différentes dates). Les flèches permettent de visualiser la similitude entre un tourbillon observé et ceux des jours précédents : une flèche épaisse indique une forte ressemblance (indice de Jaccard relativement élevé) et l'absence de flèche indique, lorsque l'indice de Jaccard a été calculé au cours de l'algorithme, que les tourbillons sont éloignés.

À l'itération j , les tourbillons gris vont être assignés à un cluster.

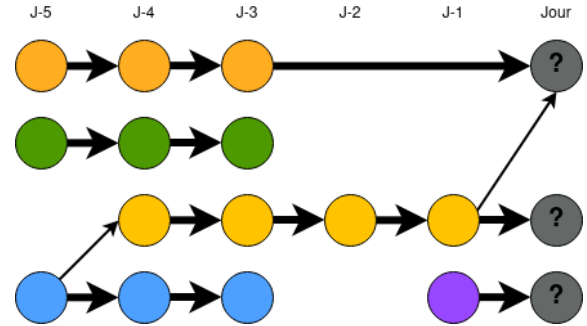


FIGURE 6 – Illustration d'une itération de l'algorithme

Itération j Toutes les observations des jours précédents ont été assignées à un cluster. Pour chaque cluster existant, on cherche à $j' = j - 1$ puis $j' = j - 2$ etc l'observation $E_{j'}$ antérieure à j (si elle existe) telle qu'il existe au moins une observation E_j à la date j vérifiant :

$$\begin{cases} E_j \text{ n'est pas encore assignée à un cluster} \\ E_j = \operatorname{argmax}_e J(e, E_{j'}) , e \in \text{observations de } j \end{cases}$$

où J est l'indice de Jaccard. Si plusieurs observations E_j vérifient ces conditions, on sélectionne celle qui est la plus similaire à $E_{j'}$. On assigne E_j au même cluster que $E_{j'}$. Après avoir traité tous les clusters existants, s'il reste des observations du jour J qui n'ont pas été assignées, on initialise un nouveau cluster pour chacune d'elles. En résumé :

- Pour chaque cluster existant, essayer de trouver une observation antérieure à J qui ressemble à une observation de J , en commençant à $J - 1, J - 2, \dots$ et l'assigner au cluster si elle existe.
- Initialiser des nouveaux clusters pour toutes les observations du jour J qui n'ont pas été assignées.

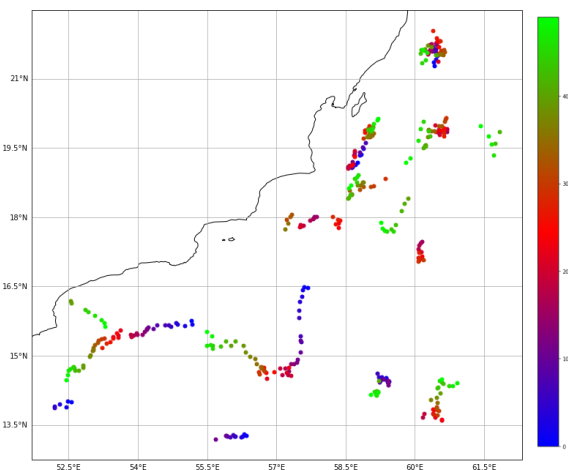
Postprocessing Les tourbillons observés trop peu de fois (c'est à dire les clusters avec un nombre d'observation inférieur à une limite paramétrable) sont considérés comme étant trop petits pour être intéressants ou susceptible d'être du bruit et sont supprimés.

La figure 7 montre le résultat obtenu pour une exécution sur 50 jours. On remarque qu'il n'y a plus d'outsider et de tourbillon-cluster orphelin comme dans le premier cas.

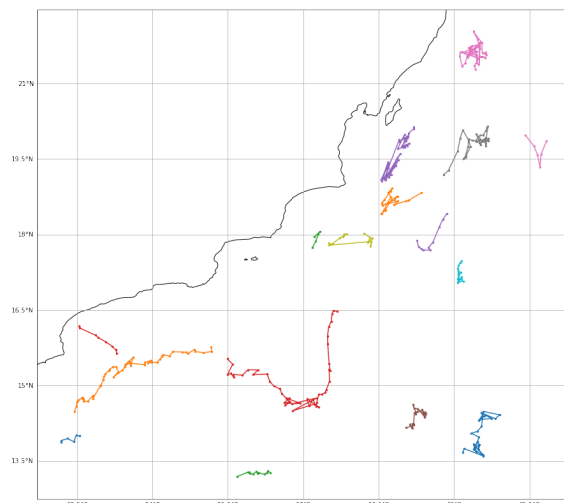
1.4. Création du catalogue

Les données du catalogue sont sauvegardées grâce à la fonction `write_catalog` définie dans le fichier `./code/catalog.py`. Cette fonction prend en paramètre la sortie de la fonction `eddies_tracker` et sauvegarde ces données dans un fichier `.csv`. Chaque ligne du tableau `csv` correspond à l'observation d'un tourbillon à une date donnée. Une observation est constituée des attributs suivants :

- *date* : date de l'observation en nombre de jours écoulés depuis le premier jour des données brutes.
- *id* : identifiant unique qui permet de retrouver toutes les observations du même tourbillon.
- x_0 : la longitude du centre du tourbillon.
- y_0 : la latitude du centre du tourbillon.
- a, b : les longueurs des axes.
- θ : l'angle entre le premier axe (de longueur a) et l'axe des longitudes (parallèle à l'équateur).



Centres de tous les tourbillons détectés pendant une période de 50 jours. Les couleurs indiquent la date à laquelle l'observation a été faite.



Clusters-trajectoires obtenus avec l'algorithme implémenté.

Résultats sur une période de 50 jours. numpy, nb_prev_day=8, date=0 to 50, runtime=600, delta_time=5, particle_gride_step=2

FIGURE 7 – Résultats de l'algorithme implémenté pour le tracking des tourbillons

- ω vitesse angulaire orientée moyenne des courants. Ce paramètre est le seul qui n'est pas réellement utilisé dans le projet, mais le signe permet de déterminer si le tourbillon est une anomalie positive ou négative du niveau de l'océan.

2. Assimilation de données

2.1. La librairie AnDA

La librairie AnDA est une librairie Python qui implémente les modèles proposés dans The Analog Data Assimilation [2]. L'idée clé du schéma d'assimilation proposé dans ce papier repose sur l'utilisation d'un modèle dynamique «data-driven» au lieu du modèle dynamique habituellement utilisé.

Pour résumer, la représentation discrète d'état utilisée est la suivante :

$$\begin{aligned} x(t) &= A[x(t-1), \eta(t)], \\ y(t) &= H[x(t)] + \epsilon(t), \end{aligned}$$

où x et y sont respectivement les états réels et les états observés, A est le modèle dynamique « data-driven » qu'on appellera opérateur de prévision analogique, η une perturbation aléatoire qu'on peut assimiler à des incertitudes du modèle. H est supposé linéaire, et l'erreur d'observation ϵ assimilée à un bruit blanc gaussien de covariance R .

L'opérateur A requiert un catalogue C formé de paires de vecteurs prédécesseurs/successeurs toutes séparées par le même intervalle de temps Δ . Son fonctionnement (pour une unique réalisation) est représenté dans la Figure 8 et se décrit de manière simplifiée de la manière suivante :

- recherche dans la catalogue de k plus proches voisins
- attribution d'un poids à chacun en fonction de leurs distances respectives
- régression sur N membres ou particules

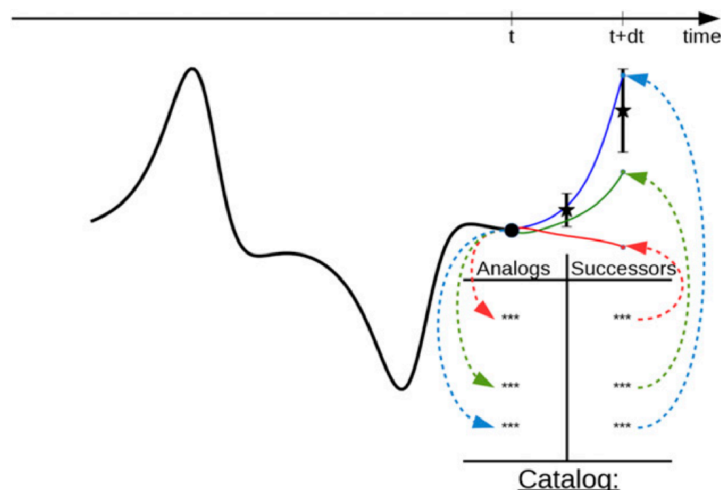


FIGURE 8 – Illustration du fonctionnement de l'opérateur de prévision analogique. [2]

- échantillonnage.

On reviendra plus en détails sur les notions de distances qui seront utilisées pour la recherche des plus proches voisins et la pondération de ces derniers. L'implémentation de la librairie propose trois sortes de régression : localement constante, localement incrémentale et localement linéaire (voir Figure 9). De même, plusieurs possibilités d'échantillonnages sont offertes : échantillonnages gaussiens ou multinomiales.

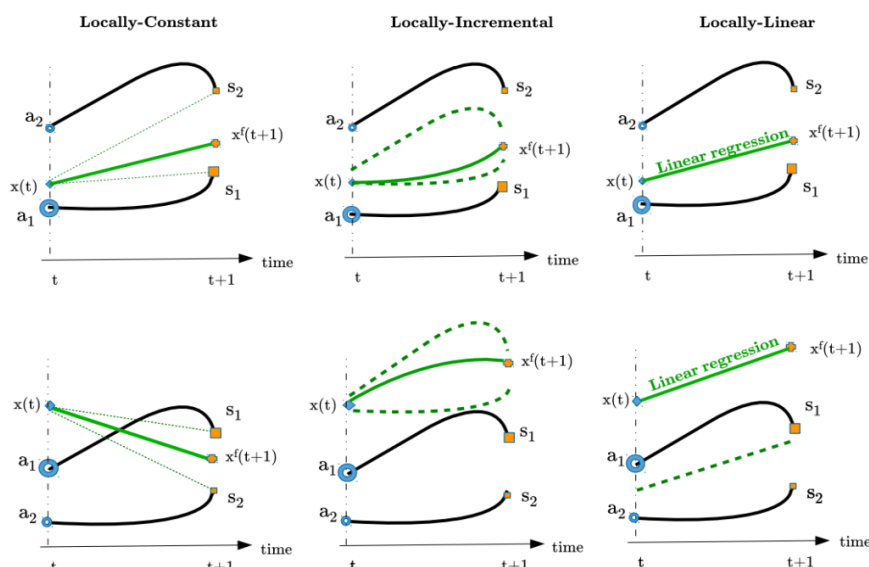


FIGURE 9 – Représentation visuelle des trois méthodes de régression proposées. [2]

La librairie implémente ensuite trois techniques classiques d'assimilation de données qui sont des variantes analogiques des méthodes classiques que sont le filtre de Kalman, le « smoother » de Kalman ainsi qu'un filtre particulaire.

En assimilant les résultats de la prédiction des tourbillons océaniques par des modèles existants à des états observés, l'Analog Data Assimilation, combinée au travaux sur la conception d'un catalogue précédemment présentés, va nous permettre d'essayer de corriger ces modèles. Pour commencer, ne disposant pas facilement de tels résultats de prédiction nous allons considérer que des courants observés

dans le passé seront des courants simulés par un modèle.

Dans la partie suivante, nous allons décrire comment nous avons préparé nos données afin qu’elles puissent être utilisées sans modifications importantes de la structure de la librairie.

2.2. Préparation des données

2.2.1. Le catalogue

En sortie de notre modèle de détection, un tourbillon E repéré par son identifiant est, comme on l’a vu, assimilé à chaque date à une structure elliptique que l’on va représenter par six paramètres. A chaque instant, un tourbillon est donc représenté par un vecteur :

$$\vec{V}_E = (x_0, y_0, a, b, \theta, \omega).$$

Parmi l’ensemble des tourbillons détectés lors de la première phase de conception du catalogue, on recherche alors tous les couples de tourbillons de même identifiant (censé donc représenté le même tourbillon) observés à deux dates consécutives. Cela nous permet de créer deux structures ordonnées de taille n appelées respectivement *Analogs* et *Successors* qui composeront le catalogue C :

$$C = \{Analogs, Successors\}.$$

Pour tous entier naturel $i < n$, *Successors*[i] est donc un vecteur à six dimensions représentant le même tourbillon que *Analogs*[i] à la date suivante. On montre sur la Figure 10 un exemple simplifié de la construction de ces structures. Il est à noter que certains tourbillons pourtant détectés ne seront pas pris en compte car non-détectés à la date suivante (mis en évidence en gris sur la figure).

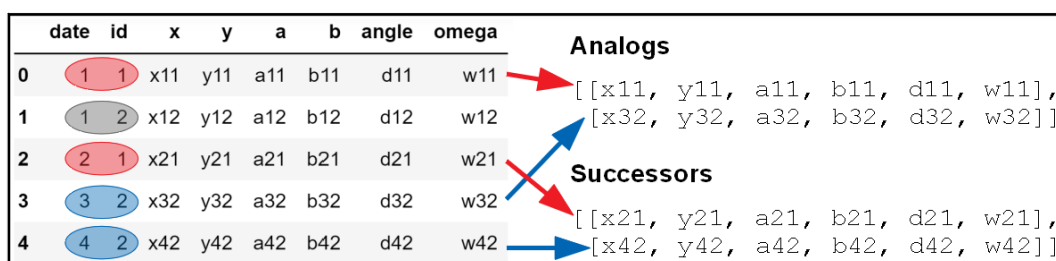


FIGURE 10 – Construction simplifiée des structures composant le catalogue.

2.2.2. Les observations

Les observations (représentant en fait plutôt dans notre cas les prédictions d’un modèle à améliorer) peuvent être traitées d’une manière similaire que précédemment : chaque tourbillon observé (ou prédit ici) est représenté par un vecteur à six dimensions et donc ses observations peuvent être représentées par une structure ordonnées contenant l’ensemble de ces vecteurs.

Cependant, il est nécessaire d’y ajouter les dates d’observation ce qui est fait de la manière suivante : une structure ordonnée *time* va contenir l’ensemble des dates entre sa première et sa dernière observation (l’intervalle de temps entre deux dates consécutives est modulable) et une autre structure parallèle *values* de taille m contiendra l’ensemble des observations (et non-observations aux dates où elles n’existent pas) du tourbillon modélisé par son vecteur de paramètres. Ainsi :

$$O = \{values, time\}$$

et pour un tourbillon donnée, *value*[i] sera le vecteur modélisant l’observation de ce tourbillon à la date *time*[i] si il y a observation et un vecteur indéfini sinon.

On peut voir cette construction sur la Figure 11, chaque tourbillon (le $id = 1$ en bleu et le $id = 2$ en rouge ici par exemple) possède cette structure. Il est important de préciser qu'il y a donc potentiellement autant de structures qu'il y a d'identifiants différents dans les observations.

	date	id	x	y	a	b	angle	omega
0		1 1	x11	y11	a11	b11	d11	w11
1		3 2	x32	y32	a32	b32	d32	w32
2		3 1	x31	y31	a31	b31	d31	w31
Obs1	time	[1, 2, 3]						
	values	[[x11, y11, a11, b11, d11, w11],						
		[nan, nan, nan, nan, nan, nan]]						
Obs2	time	[2]						
	values	[[x32, y32, a32, b32, d32, w32]]						

FIGURE 11 – Construction simplifiée des structures d'observations.

2.3. Modification de la librairie

Outre l'allègement du code par le retrait des fonctions qui ne nous étaient pas utiles, une seule grosse modification de la librairie a été effectuée lors de ce projet : elle concerne la recherche des k plus proches voisins dans le catalogue, les métriques y étant implémentées n'étant pas adaptées à notre cas d'utilisation.

On considère dans la suite deux tourbillons E et F représentés respectivement par leurs vecteurs de paramètres respectifs :

$$\vec{V}_E = (x_E, y_E, a_E, b_E, \theta_E, \omega_E) \text{ et } \vec{V}_F = (x_F, y_F, a_F, b_F, \theta_F, \omega_F)$$

Comme vu précédemment, à chaque réalisation l'opérateur de prévision analogique A commence par rechercher un nombre prédéfini k de tourbillons ressemblants à la réalisation en cours. Cela nécessite de définir une métrique d de distance entre deux tourbillons. Trois possibilités ont été envisagées et implémentées.

Distance euclidienne des centres

Une possibilité extrêmement simple consiste à simplement à assimiler la ressemblance de deux tourbillons à la distance qui sépare leurs centres. On a donc :

$$d(E, F) = \sqrt{(x_E - x_F)^2 + (y_E - y_F)^2}$$

Utiliser cette distance sous-entend que le comportement d'un tourbillon serait semblable à celui des tourbillons présents aux mêmes endroits dans le passé. Cette métrique a l'avantage d'être très rapide à calculer mais pose quelques problèmes d'interprétation. En effet, elle a du sens dans la mesure où les conditions locales ont un rôle sur l'agrandissement, l'accélération de la rotation ou le déplacement du tourbillon, mais il est possible que ces conditions aient une influence différente sur des tourbillons de formes différentes (taille, vitesse), ce que cette métrique ne prend pas en compte.

Norme L_2 pondérée

Pour prendre en compte tous les paramètres susceptibles d'influer sur le comportement du tourbillon on peut imaginer une métrique calculant la norme L_2 de la différences des vecteurs représentatifs des paramètres pondérés des tourbillons. Il s'agit donc d'attribuer à chacun des 6 paramètres $\{x_0, y_0, a, b, \theta, \omega\}$ des poids $\{w_x, w_y, w_a, w_b, w_\theta, w_\omega\}$ représentant l'importance que l'on donne à chaque paramètre dans le calcul de ressemblance entre tourbillons. On aurait alors :

$$d(E, F) = \sqrt{\sum_{p \in \{x, y, a, b, \theta, \omega\}} w_p^2 (p_E - p_F)^2}$$

Cela nécessite au préalable, afin de rester cohérent, de normaliser paramètre par paramètre l'ensemble du catalogue C et d'appliquer la même transformation au vecteur paramètre de chaque tourbillon considéré dans la suite.

Utilisation de l'indice de Jaccard

Une troisième idée est d'utiliser la distance de Jaccard $J(E, F)$ entre les deux tourbillons. Cette métrique permet une mesure fiable de la ressemblance entre deux tourbillons synthétiques elliptiques. Cependant, elle est coûteuse en temps de calcul, ce qui pose problème dans notre cas puisque cette distance est calculée à de nombreuses reprises sur l'ensemble du catalogue.

On propose donc une approximation qui consiste à chaque itération à réduire le catalogue aux $r \times k$, $r > 1$ analogues les plus proches selon une distance plus rapide à calculer, puis d'appliquer ensuite la recherche des k plus proches voisins avec la distance de Jaccard. Même ainsi, le temps de calcul reste bien supérieur aux autres méthodes ce qui rend cette distance inutilisable sur de trop longues périodes de temps.

2.4. Assimilation sur les tourbillons

Afin de procéder à l'assimilation de donnée sur des tourbillons il nous faut un catalogue et une banque de tourbillons observés (prédits) dans le futur. Dans notre cas, il nous faut choisi une date t_0 qui séparera virtuellement le passé du futur. Le travail présenté dans la partie «Détection» nous donne donc une banque de tourbillons observés dans le passé virtuel (à partir de laquelle on construit notre catalogue C) ainsi qu'une banque de tourbillons observés dans le futur virtuel (qui représente donc les prévisions d'un modèle de détection) de laquelle on retire aléatoirement des tourbillons afin de simuler des prédictions imparfaites.

2.4.1. Dans le cas d'un unique tourbillon

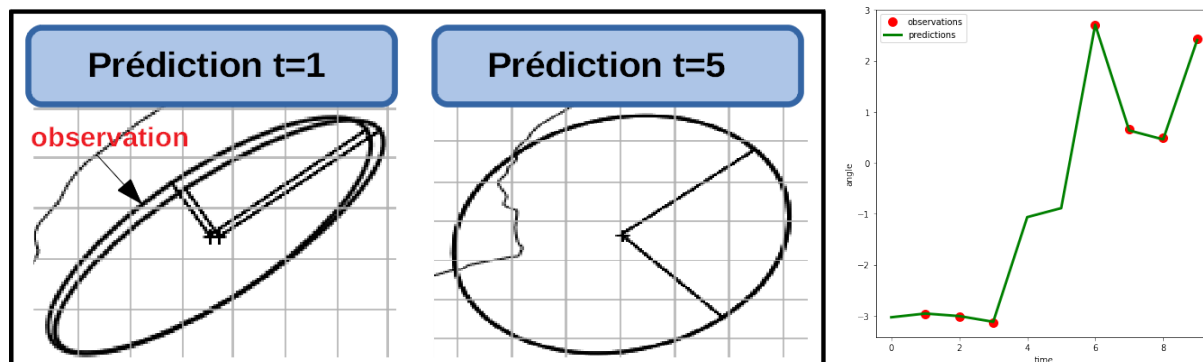
Afin de procéder sur un tourbillon E , on construit la structure d'observations qui lui est associée, on choisit le nombre N de membres à utiliser lors de l'assimilation, puis k de plus proches voisins à considérer à chaque réalisation ainsi que la distance d à utiliser pour la recherche de ces derniers. On peut également renseigner le filtre (lisseur de Kalman par défaut) à utiliser ainsi que les méthodes de régression (linéaire par défaut) et de sampling (gaussien par défaut).

On associe enfin aux observations la matrice de covariance du bruit de mesure R dépendante donc du modèle de prédiction à corriger. Ainsi en supposant que pour chaque paramètre p on note ϵ_p le bruit de mesure, on a classiquement :

$$R = \begin{pmatrix} \text{Var}(\epsilon_x) & \dots & \dots & \dots & \dots & \text{Cov}(\epsilon_x, \epsilon_\omega) \\ \vdots & \text{Var}(\epsilon_y) & & & & \vdots \\ \vdots & & \text{Var}(\epsilon_a) & & & \vdots \\ \vdots & & & \text{Var}(\epsilon_b) & & \vdots \\ \vdots & & & & \text{Var}(\epsilon_\theta) & \vdots \\ \text{Cov}(\epsilon_\omega, \epsilon_x) & \dots & \dots & \dots & \dots & \text{Var}(\epsilon_\omega) \end{pmatrix}$$

N'ayant pas accès à ces données, on utilisera dans la suite des matrices constantes à u ou encore parfois $R = u \times I_6$ avec $u > 0$.

On récupère en sortie de l'assimilation une structure de prédiction associé au tourbillon E construite sur le même principe que la structure d'observation ($\{values, time\}$). On peut alors facilement en extraire la prédiction de chacun des six paramètres à tout instant considéré. Ainsi, pour tout i le vecteur paramètres du tourbillons E à l'instant $time[i]$ est donné par $values[i]$ et nous permet de reconstruire notre tourbillon synthétique.



A gauche, le résultat graphique des prédictions (et observations si disponibles) d'un tourbillons à deux dates différentes. A droite, les variations de l'orientation du même tourbillon au cours du temps (prédictions et observations).

FIGURE 12 – Exemple de résultat sur un tourbillon.

Sur la partie gauche de la Figure 12 on voit le résultat de l'assimilation sur le même tourbillon pour deux dates différentes. Pour certaines dates ($t = 1$ ici) l'assimilation agit comme une correction de l'observation (ou de la prédiction d'un modèle dans notre cas), pour d'autre ($t = 5$ ici), où aucune observation n'est disponible, cela revient à compléter les prédictions du modèle. LA partie droite de la figure indique bien cela en montrant comment va agir l'assimilation au niveau de chaque paramètre du tourbillon (ici l'angle d'orientation).

2.4.2. Extension à l'ensemble des tourbillons

Notre implémentation peut bien sûr s'étendre dans le cas de la présence de plusieurs tourbillons. Prenons par exemple dans le cas où l'on désire procéder à l'assimilation sur l'ensemble des observations disponibles après notre date limite t_0 .

On commence par regrouper toutes les observations selon l'identifiant des tourbillons afin d'associer à chaque tourbillon sa propre structure d'observation. On étend les structures $values$ et $time$ de façon à ce que pour chaque tourbillon toutes les dates entre t_0 et une date t_{max} (qui peut-être aussi bien inférieur que supérieur à la dernière date d'observation) soit renseignées.

On définit ensuite éventuellement une durée t_{vanish} (0 par défaut) représentant le temps au bout duquel on considère qu'un tourbillon non-observé a en fait disparu ainsi que le temps avant lequel on considère qu'un tourbillon pas encore observé est en fait présent. Ainsi, si un tourbillon observé de manière irrégulière entre t_1 et t_2 verra son état prédit sur un interval de temps discrétisé T_E tel que :

$$T_E = [\min(t_0, t_1 - t_{vanish}), \max(t_{max}, t_2 + t_{vanish})].$$

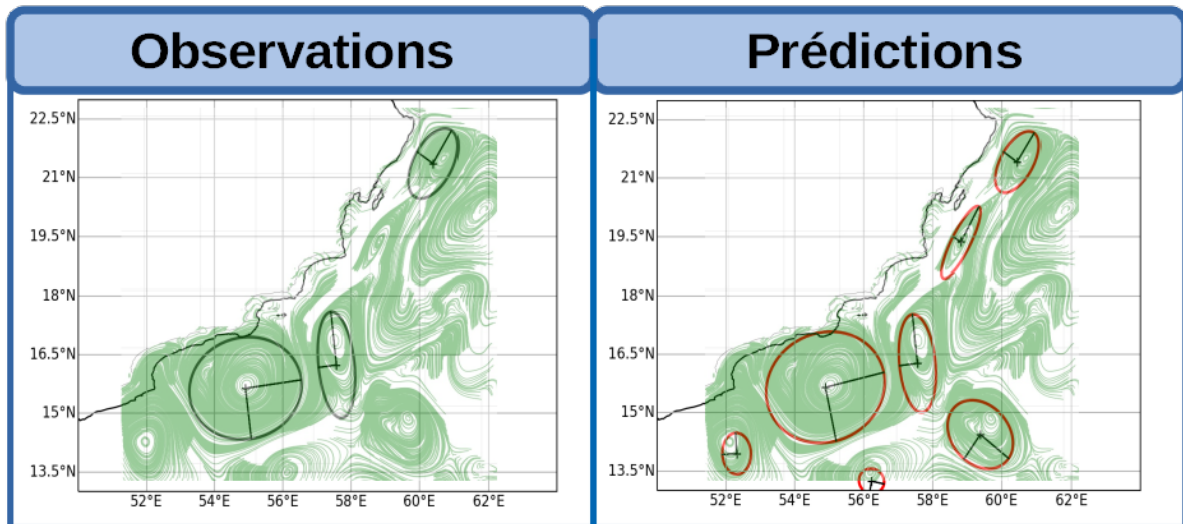


FIGURE 13 – Simulation du résultat final obtenu avec l'assimilation.

On voit sur la Figure 13 le résultat général à la date $t_0 + 4$ obtenu après assimilation. Bien que plus visuel au format vidéo, on voit clairement ici, notamment grâce à la superposition avec les lignes de courant à la même date, les corrections apportées aux observations ainsi que la complétion avec des tourbillons assimilés sur cette date. On voit néanmoins qu'il manque quelques tourbillons (à droite de l'image principalement). Cela est dû au fait que ces tourbillons ne sont soit jamais détectés soit détectés à une date ne rentrant pas dans la fourchette induite par le t_{vanish} . Augmenter ce paramètre risquerait malheureusement de faire apparaître à certaines dates des tourbillons qui n'existent pas encore ou ont déjà disparu. Il y a là un axe d'étude à approfondir afin d'améliorer notre implémentation.

2.4.3. Utilisation du code

Bien qu'il soit modulable en profondeur (modification du filtre, du sampling ou des regression à utiliser), nous proposons dans notre code (voir dépôt git en annexes) une interface relativement simple d'utilisation.

Afin d'éviter la surcharge, et comme indiqué précédemment, des choix par défaut ont été fait concernant le filtre (smoother), la régression (linéaire) et le sampling (gaussien). Pour les modifier, il suffit de changer leur valeur dans les classes **ForecastingMethod** et **FilteringMethod** présentes dans le fichier **classe.py**.

Le fichier **eddies_prediction.py** contient les fonctions permettant de construire les structures définitives du catalogue ou des observations ainsi que quelques fonctions utiles à l'interface. Il est possible d'utiliser ces fonctions de manière indépendantes afin d'obtenir des résultats intermédiaire ou d'enrichir et améliorer les résultats finaux. Néanmoins, la fonction principale **eddies_prediction** permettra une utilisation simple de l'Analog Data Assimilation [2] appliquée à nos structures de tourbillons. Voici sa documentation :

Documentation de la fonction principale de l'assimilation de donnée

```
def predict_eddies(
    catalog, observation, R, k=20, N=50, t_vanish=10, search="fast",
    Tmax=None
):
    """Compute prediction of the parameters of eddies.
    Args:
```

```
catalog(pandas.DataFrame) : bank of past observed eddies parameters.
observation(pandas.DataFrame) : bank of futur observations of eddies.
R (np.array(6,6)) : observation noise covariance matrix.
k (int) : number of analogs to use during the forecast.
N (int) : number of members.
t_vanish (int) : time after wich (and before witch) we consider an
    unobserved eddy has vanished.
search (string) : "complet" or "fast" to use either Jaccard Index
    based or just Euclidian based metric.
Tmax (int): time during which we want predictions (max time of
    observation.date if None).
Returns:
    eddies (dict[time][list(eddy)]) : list of eddies for each used time.
    ""
    .
    .
    .
    return eddies
```

De nombreuses autres options sont disponibles dans le code. On peut citer de manière non-exhaustive des modifications de la précision (et donc du temps de calcul) de l'indice de Jaccard ou encore des fonctions d'affichage graphique des tourbillons prédits sur une carte ainsi qu'une classe permettant l'animation vidéo des prédictions. Une petite documentation du dépôt git est présente en annexes.

3. Conclusion

Lors de la réalisation de ce projet nous avons implémenté tout un travail s'inscrivant dans des travaux de prédictions de l'évolution des tourbillons océaniques. En premier lieu, des outils permettant la détection des tourbillons et leur approximation elliptique afin de construire une banque de tourbillons synthétiques observés dans le passé. Cela passe par le traitement des données brutes de courant, l'obtention des lignes de courants, la détection des tourbillons, leurs approximations ainsi que l'identification des tourbillons par des méthodes de tracking. Ensuite a été implémentée l'adaptation de la librairie AnDA [2] à l'assimilation analogique de données sur les paramètres des tourbillons synthétiques et son intégration dans le reste du projet. Cela incluait un travail de préparation des données mais également des modifications au coeur du processus d'assimilation.

Notre implémentation est simple d'utilisation et permet la visualisation graphique de nos résultats sous forme de cartes de tourbillons ou animations des prédictions. On s'aperçoit que les prédictions que l'on obtient sont cohérentes et prometteuses pour une éventuelle suite de nos travaux. Ces prédictions ne sont cependant que virtuelles dans la mesure où elles consistent en fait, pour des raisons pratiques, à des simulations de prédictions ayant eu lieu dans le passé. Il faudrait notamment se pencher sur le calcul des incertitudes liées à la détection et à la prédiction des tourbillons afin de pouvoir utiliser notre implémentation dans des cas de prédictions réelles de tourbillons et, dans la suite, de correction des modèles de prédiction des courants océaniques.

Annexes

Lien du dépôt git contenant le code du projet : https://github.com/lordastorios/Projet_3A_IMTA.git

A la racine se trouve un dossier **code** qui constitue le corps de notre projet. Il contient :

- **AnDA** : un dossier contenant la librairie AnDA [2] adaptée pour notre projet.
- **data** : un dossier qui contient les données brutes de courant qui ont servi pour tester notre code.
- **tutos** : un dossier dans lequel se trouve des Notebook et fichiers python illustrant l'utilisation de nos implémentations.
- **catalog.py** : contient une fonction de stockage de la banque de tourbillons au format csv.
- **classes.py** : contient l'ensemble des classes (hors affichage graphique :) utilisées dans le code.
- **constant.py** : définit les constantes utilisées.
- **eddies_detection.py** : contient les fonctions utilisées dans le cadre de la détection et synthèse des tourbillons à partir des données de courant.
- **eddies_prediction.py** : contient les fonctions qui servent lors de l'assimilation de données.
- **eddies_tracking.py** : contient les fonctions permettant l'identification par tracking des tourbillons détectés.
- **metric.py** : contient la fonction de calcul de la distance de l'indice de Jaccard.
- **plot.py** : contient deux classes servant respectivement à l'affichage graphique de divers résultats et à l'affichage des résultats d'assimilation sous forme d'animation.
- **tools.py** : contient diverses fonctions utilisées dans le reste du code.

Références

- [1] I. Ari Sadarjoen et Frits H. Post, *Geometric Methods for Vortex Extraction*, Data Visualization'99, janvier 1999.
- [2] R. Lguensat, P. Tando, P. Ailliot, M. Pulido, R. Fablet, *The Analog Data Assimilation*, Monthly Weather Review n°145 pages 4093–4107, octobre 2017.

OUR WORLDWIDE PARTNERS UNIVERSITIES - DOUBLE DEGREE AGREEMENTS

3 CAMPUS, 1 SITE



IMT Atlantique Bretagne-Pays de la Loire – <http://www.imt-atlantique.fr/>

Campus de Brest

Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 3
France
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes

4, rue Alfred Kastler
CS 20722
44307 Nantes Cedex 3
France
T +33 (0)2 51 85 81 00
F +33 (0)2 99 12 70 08

Campus de Rennes

2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
France
T +33 (0)2 99 12 70 00
F +33 (0)2 51 85 81 99

Site de Toulouse

10, avenue Édouard Belin
BP 44004
31028 Toulouse Cedex 04
France
T +33 (0)5 61 33 83 65



IMT Atlantique

Bretagne-Pays de la Loire
École Mines-Télécom

© IMT Atlantique, 2021
Imprimé à IMT Atlantique
Dépôt légal : Mars 2021
ISSN : n° ISSN