Projet de programmation scientifique

Déplacements tectoniques de stations GNSS

Décembre 2025

julien.barneoud@ign.fr,
jacques.beilin@geodata-paris.fr,
leonie.leroux@ign.fr

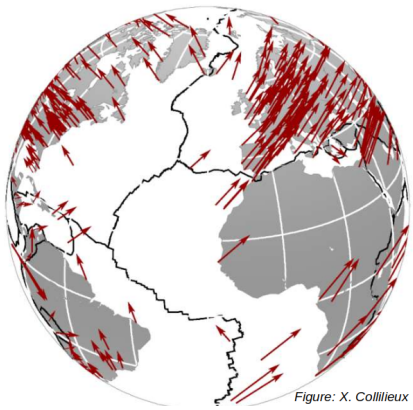

Figure: X. Collilieux

Ce projet a pour objet l'apprentissage de la programmation scientifique avec Python. Il se place également dans le cadre de la *programmation verte* : au vu des données manipulées, il sera nécessaire d'optimiser votre code -notamment en passant par du calcul matriciel- pour que le temps d'exécution reste raisonnable et ne mobilise pas inutilement les ressources de votre machine. Les bibliothèques *numpy* et *pandas* seront donc largement utilisées.

Contexte géodésique

La surface de la Terre n'est pas immobile : elle est découpée en grandes plaques lithosphériques qui se déplacent lentement les unes par rapport aux autres, avec des vitesses typiques de quelques millimètres à quelques centimètres par an. Ces déplacements tectoniques contrôlent de nombreux phénomènes naturels tels que l'ouverture des océans, la formation des chaînes de montagnes ou l'activité sismique. Grâce aux réseaux modernes de stations GNSS (Global Navigation Satellite Systems), il est possible de mesurer ces mouvements avec une précision millimétrique. Pour que ces mesures soient cohérentes à l'échelle mondiale et dans le temps, un repère géodésique de référence commun est nécessaire : l'International Terrestrial Reference Frame (ITRF). Il est calculé par l'équipe de recherche en géodésie de l'IGN - IPGP, dont une nouvelle version sort en moyenne tous les 6 ans, avec parfois des mises à jour annuelles.

L'ITRF fournit les coordonnées des stations et leurs vitesses, mais pour interpréter ces mouvements, on distingue le comportement rigide des plaques du comportement local (déformation). Dans un modèle cinématique global, on considère qu'une plaque se déplace comme un corps rigide autour d'un pôle de rotation : un point situé à la surface de la Terre défini par une latitude et une longitude, autour duquel la plaque tourne avec une vitesse angulaire constante. Cette rotation peut être représentée par un vecteur $\vec{\omega}$ (appelé vecteur de rotation), dont les composantes permettent de prédire la vitesse théorique d'un point situé sur la plaque. Le modèle ITRF2020 Plate Motion Model (PMM), publié dans Altamimi et al. (2023), fournit les pôles de rotation et vitesses angulaires des 13 principales plaques tectoniques. En comparant les vitesses prédites par ce modèle aux vitesses

observées, on peut identifier les régions stables, les zones de déformation, et mieux comprendre la dynamique tectonique actuelle.

Objectifs

L'objectif de ce projet est de manipuler des données géodésiques réelles (stations GNSS ITRF2020, limites de plaques tectoniques, modèle de déformation, pôles de rotation du modèle ITRF2020-PMM), et de mettre en œuvre plusieurs algorithmes fondamentaux :

— conversion entre différents types de coordonnées ;
— détermination de l'appartenance d'un point à un polygone ;
— calculs de distances géographiques ;
— mise en œuvre d'un modèle cinématique de plaques rigides ;

Vous serez amenés à écrire vos propres fonctions, à évaluer leurs performances, et à comparer les résultats avec ceux obtenus par des bibliothèques standards (notamment `geopandas`). Le projet est conçu pour être progressif : chaque étape prépare la suivante.

Ce projet est divisé en 4 parties principales :

1. Lecture et préparation des données
2. Calcul d'appartenance à une plaque tectonique
3. Filtrage des stations se trouvant dans des zones de fortes déformations
4. Calcul des déplacements des stations se trouvant hors zone de déformation

Vous pourrez faire divers cartes pour présenter vos résultats (voir 5).

Données

Le dossier *data* contient l'ensemble des données utiles pour traiter le sujet. Nous retrouvons :

— *Tectonic_Plates.geojson* : limites de plaques tectoniques.
— *ITRF2020_GNSS.SSC.txt* : coordonnées des stations GNSS de l'ITRF2020. Les coordonnées sont en coordonnées cartésiennes, à l'époque 2015.0. Une station peut voir plusieurs coordonnées fournies (i.e. plusieurs lignes dans ce fichier), valables sur une plage temporelle particulière appelée "SOLN" (voir dernières colonnes *SOLN, DATA_START,DATA_END*). Ces discontinuités et changements de coordonnées correspondent essentiellement à des changements de matériel des stations GNSS (antennes et/ ou récepteurs).
— *GSRM_strain.txt* : modèle global de déformation. Voir l'Annexe 7.3.
— *pmm_itrf.txt* : table du modèle ITRF2020-PPM, décrit dans l'article Altamimi, Z. et al. (2023). ITRF2020 plate motion model.

Librairies à utiliser

En raison du format des données (grille, nombreux points...), il sera nécessaire de travailler matriciellement, afin d'optimiser votre code pour éviter de mobiliser inutilement les ressources de l'ordinateur ("programmation verte").

Les librairies indispensables :

— *numpy* et *pandas* pour le calcul matriciel et manipulation des données tabulées
— *matplotlib* pour les graphiques
— *cartopy* pour le tracé de cartes
— *datetime* pour la gestion des dates
— *tqdm* utile pour l'affichage de bars de progressions dynamiques et connaître efficacement l'avancement des boucles, en évitant les "print" excessifs dans la console

Sauf mention explicite dans certaines questions, l'utilisation de `geopandas` est interdite, car l'objectif du sujet est de comparer et de comprendre les fonctions sous-jacentes utilisées par cette librairie.

1 Lecture et formatage des données

Cette première partie vise à charger les différents jeux de données, à les mettre en forme, et à effectuer les conversions nécessaires pour le reste du projet.

1.1 Stations GNSS ITRF2020 (XYZ et vitesses)

Vous disposez d'un fichier *ITRF2020_GNSS.SSC.txt* contenant pour chaque station :
— les coordonnées cartésiennes (X, Y, Z) en mètres dans le repère terrestre ITRF2020 ;
— les vitesses (V_X, V_Y, V_Z) en m/an ;
— l'époque de validité (2015.0 pour ITRF2020).

Ces coordonnées sont cartésiennes, ce qui n'est pas adapté pour effectuer des tests d'appartenance géographique où les frontières de plaques sont connues avec leur latitude (φ) et longitude (λ). Pour calculer les coordonnées géographiques (λ,φ) d'une station à partir de ses coordonnées cartésiennes géocentriques (X,Y,Z), il faut utiliser les formules suivantes :

$$r = \sqrt{X^2 + Y^2 + Z^2}$$

$$\mu = \arctan\left[\frac{Z}{\sqrt{X^2+Y^2}}.\left[(1-f_e) + \frac{a_e.e_e^2}{r}\right]\right]$$

$$\lambda = 2\arctan\left[\frac{Y}{X + \sqrt{X^2+Y^2}}\right]$$

$$\varphi = \arctan\left[\frac{Z(1-f_e) + e_e^2 a_e \sin^3\mu}{(1-f_e)\left(\sqrt{X^2+Y^2} - e_e^2 a_e \cos^3\mu\right)}\right]$$

$$h = \sqrt{X^2+Y^2}\cos\varphi + Z\sin\varphi - a_e\sqrt{1 - e_e^2\sin^2\varphi}$$

Pour appliquer ces formules de transformation classiques, vous devez utiliser les valeurs du demi-grand axe a_e, de l'aplatissement f_e (et de l'excentricité e_e, donnée par $e_e^2 = f_e(2-f_e)$) de l'ellipsoïde WGS84, fournies dans le fichier Earth_parameters.dat.

Travail demandé

1. Charger les coordonnées des stations GNSS *ITRF2020_GNSS.SSC.txt* dans un `DataFrame`. *Au vu de la structure irrégulière des colonnes du fichier, la méthode `pandas.read_fwf()` pourra être utilisée*. Dans le cas où une station a plusieurs plages temporelles de validité (i.e. plusieurs lignes),vous ne garderez que la dernière (DATA_END="00 :000 :00000").
2. Vous ferez de même avec les vitesses Vx, Vy, Vz de chaque station, que vous pourrez charger dans un deuxième `DataFrame`. Ces vitesses seront notamment utiles à la fin du sujet pour comparer et analyser vos résultats avec les vitesses de plaques calculées avec le modèle ITRF2020-PMM (voir 4.2)
3. Implémenter une fonction Python `xyz_to_llh(X, Y, Z)` retournant (λ, φ) en radian.
4. Ajouter ces colonnes (λ, φ) à votre DataFrame stations (radian & degré).

Exemples numériques pour vérifier votre fonction :
— BRAZ [4115014.0770 -4550641.6224 -1741443.8395]
=> lon=-47.877869 ; lat=-15.947474 ; h=1106.018257
— GRAS [4581690.7667 556115.0275 4389360.9122]
=> lon=6.920576 ; lat=43.754739 ; h=1319.313196

1.2 Modèle de déformation (GEM Strain Rate Model)

Pour déterminer les zones de déformation sur l'ensemble de la surface de la planète, nous nous appuyons sur le modèle du GEM **(voir Annexe 7.3)**. Les données sont fournies sous forme d'un fichier texte contenant pour chaque point :

$$\text{lat, lon, } e_{xx},\ e_{yy},\ e_{xy},\ \ldots$$

Ces points constituent une grille globale. Vous utiliserez les coordonnées (λ, φ) pour déterminer si vos stations GNSS se trouvent à proximité d'une zone de déformation.

Travail demandé.

1. Charger les points du modèle GEM dans un `DataFrame`.
2. Effectuer un filtrage en ne gardant que les points ayant une déformation significative (voir Annexe 7.3)
3. Supprimer les doublons éventuels : certaines coordonnées (lat, lon) apparaissent plusieurs fois dans le fichier, correspondant à plusieurs estimations de déformation pour un même point. Pour chaque point, ne conserver que la ligne correspondant à la déformation maximale.

1.3 Limites de plaques tectoniques (GeoJSON)

Le fichier `Tectonic_Plates.geojson` contient les géométries de différentes plaques tectoniques au format :

— `type: Polygon` ou `MultiPolygon` ;
— coordonnées en `CRS84`, soit **(longitude, latitude)** en degrés.

Vous allez extraire pour chaque plaque une liste ordonnée des sommets, par exemple :

$$[(\lambda_1, \varphi_1),\ (\lambda_2, \varphi_2), \ldots, (\lambda_n, \varphi_n)].$$

Travail demandé.

1. Lire le fichier GeoJSON (à l'aide de la librairie `json`).
2. Extraire les noms des plaques et les sommets de leurs polygones.
3. Stocker le tout dans une structure exploitable, par exemple :

```
dict_plates = {
    'Amur'    : DataFrame(columns=['lat', 'lon', 'h']),
    ...
    'Somalia' : DataFrame(columns=['lat', 'lon', 'h']),
    }
```

4. Par souci d'optimisation pour la suite, vous ne garderez que les 13 plaques décrites dans ITRF2020-PMM. La liste de ces plaques (ID et nom) se trouve dans le fichier *pmm_itrf.txt* que vous pourrez aussi charger dans un `DataFrame`

Remarques : Dans le geojson, pour une plaque donnée, le nom de la plaque se trouve dans l'attribut `dict["features"]["properties"]["PlateName"]` ; le polygone (liste de points) : `dict["features"]["geometr`
Une plaque peut contenir plusieurs polygones (ex : 'Autralia'). Dans ce cas, il faudra les concaténer (i.e. mettre les points à la suite).

2 Appartenance à une plaque tectonique

L'objectif de cette partie est de déterminer, pour chaque station GNSS, la plaque tectonique à laquelle elle appartient. Vous développerez progressivement une méthode pour résoudre un problème fondamental de géométrie algorithmique : **l'appartenance d'un point à un polygone**. *Dans la suite du sujet, les "points" désigneront les "stations GNSS" et les "polygons" les "plaques tectoniques".*

Pour mesurer le temps d'exécution de votre script, vous pouvez vous appuyer sur la librairie `datetime` :

```
import datetime as dt

t_start = dt.datetime.now()  # date depart
# .
# ... vos executions de fonctions....
# .
t_end = dt.datetime.now() # date finale
print("Temps d'execution : ", t_end - t_start)
```

2.1 Méthode Ray Casting

Sur le plan algorithmique, la méthode la plus couramment utilisée pour déterminer si un point appartient ou non à un polygone est l'algorithme dit du Ray Casting (ou « test de la demi-droite »). En pratique, lorsqu'on utilise des bibliothèques comme Shapely ou GeoPandas, il n'est généralement pas nécessaire d'implémenter soi-même cette méthode : ces outils fournissent déjà des prédicats topologiques (comme `contains`) qui exécutent ce test de façon optimisée. Cependant, il est important d'en comprendre le principe, car il s'agit du fondement de la plupart des méthodes d'appartenance point-dans-polygone.

Principe général L'algorithme consiste à tracer, depuis le point à tester, une demi-droite (ou « ray ») vers une direction arbitraire, généralement vers l'est, et à compter le nombre d'intersections entre cette demi-droite et les arêtes du polygone.

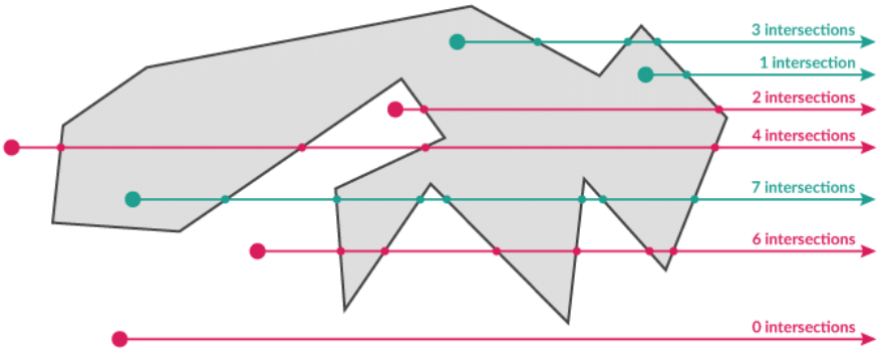

FIGURE 1 – Exemple du principe de Ray Casting. Image extraite de l'article de Vacek et al., 2017

— Si le nombre d'intersections est impair, le point est considéré comme à l'intérieur du polygone.
— S'il est pair, le point est à l'extérieur.

Cette idée repose sur un principe simple : chaque fois que la demi-droite « entre » dans le polygone puis « sort » de celui-ci, elle traverse une arête. Un nombre impair d'intersections signifie donc que le point est piégé à l'intérieur ; un nombre pair indique qu'il est à l'extérieur.

Avantages et limites La méthode fonctionne aussi bien pour les polygones convexes que concaves.

Certains cas particuliers (comme un point situé exactement sur une arête ou un sommet) nécessitent un traitement spécifique pour éviter les ambiguïtés.

Travail demandé Implémentez la méthode standard Ray Casting :

1. Tracer une demi-droite horizontale à partir du point ;
2. Compter les intersections avec les segments du polygone ;
3. Si le nombre impair : le point est à l'intérieur.
4. **Attention à la gestion des cas limites** : si le point est un sommet du polygone, si le point est sur une arête du polygone...
5. Attribuer finalement à chaque station GNSS sa plaque (ajouter par exemple une colonne 'plate' dans votre DataFrame).
6. Optimisation : pour réduire le temps de calcul, vous pourrez mettre en place votre méthode ray-casting sous forme matricielle, notamment pour s'affranchir des tests d'intersection côté par côté du polygone dans une boucle `for`.

À l'issue de cette étape, il est normal que certaines stations n'aient pas de plaque attribuée. En effet, le modèle ITRF2020-PMM ne renseigne que les 13 principales plaques, les stations se trouvant notamment sur la plaque Coco ou Philippine ne sont donc pas modélisées dans l'ITRF2020-PMM.

Cette méthode est utilisée dans la majorité des bibliothèques géométriques (y compris GEOS, utilisée par `geopandas`).

2.2 Optimisation par boîte englobante

Pour améliorer la performance de votre algorithme, vous vérifierez le test d'appartenance *avant* d'appliquer le ray-casting :

$$\lambda_{\min plaque} \leq \lambda_{station} \leq \lambda_{\max plaque} \qquad \text{et} \qquad \varphi_{\min plaque} \leq \varphi_{station} \leq \varphi_{\max plaque}.$$

Cela permet d'éliminer immédiatement certaines plaques.

2.3 Comparaison avec `geopandas`

Vous comparerez vos deux méthodes précédentes (i.e. ray-casting, boite englobante + ray-casting) avec la librairie **geopandas** (voir documentation officielle), notamment en regardant :

— la fiabilité (la même plaque attribuée à chaque point ?)

— le temps d'exécution ;

Avec geopandas, vous pourrez en particulier regarder les méthodes `within`, `contains` ou `sjoin` qui permettent de faire des tests sur les géométries.

> À titre indicatif, le temps d'exécution sur l'ensemble des 1330 stations GNSS par les différentes méthodes :
>
> — ray-casting basique (boucle for sur les côtés) : $\approx$ 10s
>
> — boîte englobante + ray-casting basique : $\approx$ 5s
>
> — ray-casting vectorisé : $<$ 1s
>
> — boîte englobante + ray-casting vectorisé : $<$ 0.5 s
>
> — geopandas : $<$ 0.1s

2.4 (bonus) Méthode par test sur la somme des angles

Une méthode conceptuellement simple pour déterminer si un point appartient à un polygone consiste à calculer la somme des angles formés entre ce point et chaque paire de sommets consécutifs du polygone.

On note le point testé P et les sommets successifs du polygone A_i et A_{i+1}. Pour chaque paire (A_i, A_{i+1}), on calcule l'angle orienté entre les vecteurs :

$$\vec{u}_i = A_i - P, \qquad \vec{v}_i = A_{i+1} - P.$$

L'angle correspondant vaut :

$$\theta_i = \operatorname{atan2}\Big(\|\vec{u}_i \times \vec{v}_i\|,\ \vec{u}_i \cdot \vec{v}_i\Big).$$

On additionne ensuite tous les angles :

$$\Theta = \sum_{i=1}^{N} \theta_i.$$

Le critère est alors le suivant :

$$\text{le point est à l'intérieur si } \Theta \approx 2\pi, \qquad \text{et à l'extérieur si } \Theta \approx 0.$$

Ce test fonctionne pour les polygones convexes et concaves. En revanche, il est coûteux car il implique un calcul trigonométrique (via `atan2`) pour chaque arête.

Travail demandé (bonus). Écrire une fonction `point_in_polygon_angle(lat, lon, polygon)` qui applique cette méthode de la somme des angles, puis comparer son temps d'exécution et résultats à ceux de votre méthode principale.

3 Proximité à une zone de déformation

Après avoir identifié la plaque tectonique de chaque station (partie 2), le but est maintenant de déterminer si certaines d'entre elles se situent dans une région de déformation active. En effet, si tel est le cas, le comportement des stations ne pourra pas être prédit par le modèle de plaque PPM (voir la partie 4).

Comme vu précédement, le modèle de déformation utilisé est celui du GEM (Annexe 7.3)

3.1 Distance géodésique : formule de Haversine

La formule de Haversine permet de calculer la distance la plus courte entre deux points à la surface d'une sphère, en tenant compte de la courbure de la Terre. Contrairement à une distance euclidienne classique dans un plan, elle utilise les latitudes et longitudes (en radians) pour déterminer la distance géodésique le long du grand cercle reliant les deux points. C'est l'une des méthodes les plus courantes pour estimer des distances sur le globe à partir de coordonnées géographiques.

Voici la formule :

$$d = 2R_T \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos\varphi_1 \cos\varphi_2 \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right), \qquad R_T = 6378 \text{ km}.$$

où :

— φ_1, λ_1 : latitude et longitude du premier point (en radians)
— φ_2, λ_2 : latitude et longitude du second point (en radians)
— R_T : Approximation du rayon terrestre (sphère)
— d : distance géodésique le long de la surface terrestre entre les deux points

3.2 Détection de zone de déformation

Vous déclarerez :

$$\text{station en zone de déformation} \quad \Longleftrightarrow \quad d_{\min} < 50 \text{ km}.$$

Travail demandé.

1. Écrire une fonction `distance_haversine()`.
2. Calculer la distance minimale entre chaque station et les points de déformation de la grille GEM préparée en 1.2.
3. Ajouter une colonne `in_deformation` dans votre table de stations GNSS.

> Exemple numérique pour vérifier votre fonction d'Haversine :
> d(GRAS - TOUL) = 438.55 km
> => cohérent avec la distance Grasse - Toulouse.

4 Détermination des déplacements de stations

Dans cette dernière partie, vous appliquerez le modèle cinématique des plaques rigides ITRF2020-PMM à l'ensemble de vos stations GNSS. Ce modèle est valable pour les **stations situées hors zone de déformation** i.e. `in_deformation=False`. Vous pourrez notamment comparer les écarts de vitesses prédites vs. observées à la fin de cette partie

4.1 Pôles de rotation ITRF2020

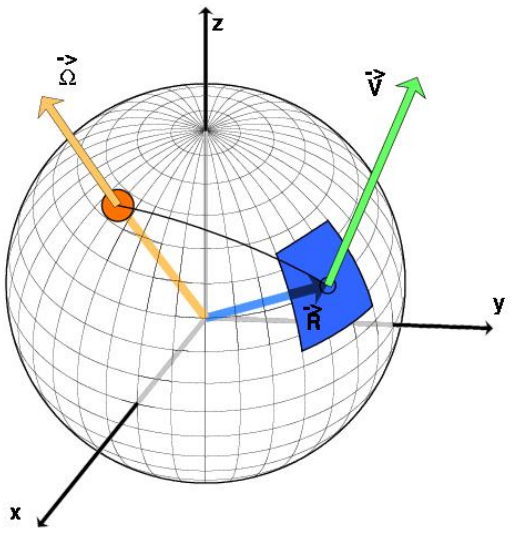

FIGURE 2 – Mouvement d'une plaque à partir de son pôle de rotation. Figure extraite de https ://www.geologie.ens.fr

Pour chaque plaque tectonique, le PMM fournit un vecteur de rotation :

$$\vec{\omega} = (\omega_x, \omega_y, \omega_z) \text{ [mas/an]}.$$

La conversion radians/an s'exprime :

$$\omega_{\text{rad/an}} = \omega_{\text{mas/an}} \times 4.848136811 \times 10^{-9}.$$

Composante de translation : Origine Rate Bias (ORB) Contrairement à d'autres modèles de plaques, le PMM ITRF2020 inclut une translation globale :

$$\vec{T} = (T_X, T_Y, T_Z) \text{ [mm/an]}.$$

Par l'ajout de cette composante ORB, les consignes officielles recommandent d'ignorer la composante verticale de la vitesse prédite.

Vitesse prédite Pour une station sur une plaque donnée, la vitesse prédite s'exprime par le *produit vectoriel* :

$$\vec{v}_{\text{pred}} = \vec{\omega} \times \vec{r} + \vec{T},$$

où $\vec{r}$ est le vecteur unitaire de direction :

$$\vec{r} = \frac{1}{\sqrt{X^2 + Y^2 + Z^2}}(X, Y, Z).$$

Travail demandé.

1. Charger les pôles de rotation ITRF2020 depuis *pmm_itrf.txt* (tables de l'Annexe 7.2) dans un DataFrame.
2. Implémenter une fonction donnant $\vec{v}_{\text{pred}}$.

Exemples numériques de la station GRAS :
- Vitesse prédite PMM : (m/an) : [-0.01270 0.01889 0.01204]
  norme : 0.026 m/year
- Vitesse *ITRF2020_GNSS.SSC.txt* (m/an) : [-0.01364 0.01893 0.0113]
  norme : 0.026 m/year

4.2 Analyse des vitesses prédites par le modèle de plaques ITRF2020-PMM

- Quelles sont les 10 stations avec les vitesses prédites (normes) les plus importantes ? Où se trouvent-elles ?
- Même question pour les 10 vitesses les plus faibles.

Comparer vos $\vec{v}_{\text{pred}}$ PMM avec les vitesses "observées"[1] ITRF2020 qui sont dans *ITRF2020_GNSS.SSC.txt* (voir 1.1). Quelles sont les 10 stations avec les différences les plus élevées ? Dans cette analyse, vous pourrez notamment regarder :

- la cohérence des vitesses par plaque ;
- l'influence des zones de déformation ;
- les cas où les stations sont proches de frontières de plaques.

Remarque "vitesses "observées"[1] : les vitesses ITRF2020 fournies dans le fichier *ITRF2020_GNSS.SSC.txt* ne sont pas uniquement déterminées à partir des séries temporelles de positions des stations. Elles

résultent en fait d'un ajustement global de l'ensemble du réseau, nettement plus complexe, dans lequel des contraintes — notamment des contraintes d'égalité sur les vitesses des stations distantes de moins de 10 km — sont appliquées.

5 Représentations cartographiques

Pour représenter vos divers résultats sur une carte, vous pourrez par exemple :
— tracer les limites de plaques ;
— représenter les stations GNSS ;
— visualiser les zones de déformation et les distances seuils ;
— afficher les vecteurs de vitesse observés et modélisés (voir Annexe 7.4 ;
Un zoom pourra être fait en particulier sur la plaque Eurasie.

6 À rendre...

Le projet sera réalisé sous Python sous forme de fonctions et de scripts permettant d'exécuter le code facilement. Il sera accompagné d'un compte-rendu synthétique (format pdf).

Le travail sera réalisé en binôme.

Le rendu est attendu par mail sous forme d'un dossier compressé (archive zip ou tar.gz) portant les noms des membres du groupe selon le modèle : progSci25_nom1_nom2.zip contenant :

1. un dossier *src* avec le code source python :
   — vos scripts *commentés* avec des noms explicites (partie_x.py, main.py, ...), contenant les appels aux fonctions. Chacun de ces scripts devra nous permettre de vérifier le bon fonctionnement du code produit ;
   — les scripts python complémentaires éventuels avec les fonctions ou classes appelées (toolbox, lecture, calcul, dessin...) ;
2. un rapport (format pdf) décrivant en quelques pages :
   — la reformulation des thématiques/ objectifs du sujet ;
   — la recherche bibliographique s'il y a lieu ;
   — pour chaque partie :
      (a) la description et structure des algorithmes choisis (documentation développeur), avec une mise en évidence des choix adoptés dans la recherche d'optimisation du code
      (b) sorties graphiques et comparaison/analyse des résultats intermédiaires (se questionner sur la pertinence et les ordres de grandeur des distances et vitesses calculées...).

Concernant les codes sources, un accent particulier sera porté sur :
— l'organisation et la structure du code (import, définition des fonctions, main),
— les commentaires (entêtes de fonctions et corps '#'),
— le fonctionnement des programmes,
— la bonne utilisation des bibliothèques *numpy* et *pandas*
— l'optimisation et le temps d'exécution (quelques secondes max, affichages pertinents dans la console utilisateur pour l'informer de l'avancement),
— la présentation des résultats (2D/3D, légendes et titres des graphes, figures non pixelisées...)

7 Annexes

7.1 Glossaire

— **ITRF** : *International Terrestrial Reference Frame*. Site officiel : https ://itrf.ign.fr

— **GSRM** : *Global Earthquake Model Strain Rate Model* Modèle de vitesse de déformation "strain rate". Voir site UNAVCO et site C. Kreemer
— **GNSS** : *Global Navigation Satellite Systems*
— **PPM** : *Plate Motion Model.* Modèle de plaque tectonique
— **IGS** : *International GNSS Service*. L'IGS traite et diffuse les données de stations GNSS sur un réseau international (cf https ://igs.org)

7.2 Modèle ITRF2020-PPM

Source article : https ://doi.org/10.1029/2023GL106373

TABLE 1 – Absolute Plate Rotation Poles Defining the ITRF2020-Plate Motion Model and Their Standard Deviations (1-Sigma)

Plate	N^a	ω_x	ω_y	ω_z	ω	WRMS_E	WRMS_N
		(mas/yr)			(°/Ma)	(mm/yr)	
AMUR	3	-0.131	-0.551	0.837	0.281	0.17	0.18
		±0.009	±0.014	±0.015	±0.002		
ANTA	15	-0.269	-0.312	0.678	0.220	0.16	0.24
		±0.003	±0.003	±0.004	±0.001		
ARAB	3	1.129	-0.146	1.438	0.509	0.15	0.11
		±0.025	±0.027	±0.016	±0.007		
AUST	118	1.487	1.175	1.223	0.627	0.19	0.17
		±0.003	±0.003	±0.003	±0.001		
CARB	5	0.207	-1.422	0.726	0.447	0.18	0.56
		±0.076	±0.174	±0.061	±0.053		
EURA	143	-0.085	-0.519	0.753	0.255	0.19	0.21
		±0.003	±0.003	±0.003	±0.001		
INDI	4	1.137	0.013	1.444	0.511	0.41	0.33
		±0.008	±0.040	±0.016	±0.005		
NAZC	3	-0.327	-1.561	1.605	0.629	0.05	0.10
		±0.006	±0.018	±0.008	±0.003		
NOAM	108	0.045	-0.666	-0.098	0.187	0.23	0.31
		±0.003	±0.003	±0.003	±0.001		
NUBI	31	0.090	-0.585	0.717	0.258	0.18	0.21
		±0.003	±0.003	±0.004	±0.001		
PCFC	20	-0.404	1.021	-2.154	0.672	0.32	0.30
		±0.003	±0.003	±0.004	±0.001		
SOAM	59	-0.261	-0.282	-0.157	0.115	0.30	0.29
		±0.004	±0.004	±0.003	±0.001		
SOMA	6	-0.081	-0.719	0.864	0.313	0.41	0.21
		±0.014	±0.015	±0.005	±0.004		
ITRF2020-PMM overall fit						0.21	0.24

[a] Number of sites.

TABLE 2 – Components of the Origin Rate Bias of the ITRF2020-Plate Motion Model and Their Standard Deviations (1-Sigma)

mm/yr		
0.37	0.35	0.74
±0.08	±0.10	±0.09

7.3 Le modèle de déformation GSRM

Le modèle **GSRM v2.1** (Global Strain Rate Model) fournit un champ global de vitesses et de taux de déformation horizontaux de la croûte terrestre, dérivé principalement de données géodésiques (GPS, VLBI, DORIS). Chaque ligne du fichier ASCII correspond à un point de la grille avec les colonnes suivantes :

- **lat, lon** : coordonnées géographiques du centre de la cellule de grille (en degrés)
- **exx, eyy, exy** : composantes du tenseur de taux de déformation horizontale (nanostrain/an)
- **vorticity** : taux de rotation locale (nanoradians/an)
- **RL-NLC, LL-NLC** : composantes non-linéaires de cisaillement, utilisées pour l'analyse sismotectonique
- **e1, e2** : valeurs propres du tenseur de déformation (e1 : extension principale, e2 : compression principale)
- **azi_e1** : azimut de l'axe de déformation principale e1, en degrés depuis le Nord

Pour déterminer si un point donné se situe dans une *zone de déformation*, on peut :

- localiser le point le plus proche dans la grille GSRM,
- calculer la *magnitude de déformation* :

$$\text{strain magnitude} = \sqrt{exx^2 + eyy^2 + 2\,exy^2}$$

- utiliser des seuils indicatifs : **>50 nanostrain/an** correspond à une zone fortement déformée, **20–50** à une zone modérément déformée, et **<20** à une région stable.

.

Pour plus d'informations, voir site de l'UNAVCO et site de C. Kreemer.

7.4 Conversion des vitesses cartésiennes en variations angulaires.

Les vitesses calculées à partir du modèle de mouvement des plaques (PMM) sont exprimées dans un repère cartésien terrestre X, Y, Z en m/an (V_x, V_y, V_z).

Pour représenter ces vecteurs sur une carte en latitude φ et longitude λ, il est pratique de les projeter dans un repère topocentrique tangent à la surface de la Terre au niveau de la station, appelé repère ENU (East-North-Up). La transformation s'écrit :

$$\begin{bmatrix} V_E \\ V_N \\ V_U \end{bmatrix} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\varphi\cos\lambda & -\sin\varphi\sin\lambda & \cos\varphi \\ \cos\varphi\cos\lambda & \cos\varphi\sin\lambda & \sin\varphi \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}.$$

Ici :

- V_E, V_N, V_U sont les composantes horizontales et verticale dans le repère local (m/an),
- φ, λ sont la latitude et longitude de la station,
- V_E correspond à la vitesse vers l'Est, V_N vers le Nord, et V_U verticalement.

Ensuite, pour l'affichage sur une carte en coordonnées angulaires, on approxime :

$$V_\varphi \approx \frac{V_N}{R}, \qquad V_\lambda \approx \frac{V_E}{R\cos\varphi},$$

où $R \approx 6371$ km est le rayon moyen terrestre. Cette approximation est suffisante pour **la représentation graphique des vecteurs vitesse** ; elle ne fournit pas une conversion géodésique précise.

7.5 Références

— Altamimi, Z., Metivier, L., Rebischung, P., Collilieux, X., Chanard, K., & Barnéoud, J. (2023). ITRF2020 plate motion model. Geophysical Research Letters, 50(24), https ://-doi.org/10.1029/2023GL106373

— ALTAMIMI, Zuheir, REBISCHUNG, Paul, COLLILIEUX, Xavier, et al. ITRF2020 : an augmented reference frame refining the modeling of nonlinear station motions. Journal of Geodesy, 2023, vol. 97. https ://doi.org/10.1007/s00190-023-01738-w

— Site web de l'ITRF : https ://itrf.ign.fr

— Kreemer, C., G. Blewitt, E.C. Klein, 2014, A geodetic plate motion and Global Strain Rate Model, Geochemistry, Geophysics, Geosystems, 15, 3849-3889, https ://doi.org/10.1002/2014GC005407.

— Tectonic plates : Arcgis layer

— VACEK, Lukas, ATTER, Edward, RIZO, Pedro, et al. sUAS for deployment and recovery of an environmental sensor probe. In : 2017 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, 2017. p. 1022-1029.