Analyse de données en géosciences : Projet Tsunami

Matthieu COURCELLES
Maxime SOARES CORREIA

Master 1 — Géologie

Université Paris-Cité
IPGP - Institut de Physique du Globe de Paris

Encadrants : Éric GAYER, Clément NARTEAU
Paris, Oct. - Dec. 2025

Analyse de données en géosciences : Projet Tsunami

par

Matthieu Courcelles
Maxime SOARES CORREIA

Master 1 — Géologie

Encadrants : E. Gayer, C. Narteau
Institution : IPGP — Institut de Physique du Globe de Paris
Durée du projet : Octobre - Décembre 2025

Table des matières

1

Introduction

1.1 Contexte géophysique

Les tsunamis sont des ondes de gravité de très grande longueur d'onde, générées par un déplacement brutal de la colonne d'eau océanique. Leur origine la plus fréquente est un *séisme sous-marin* dans une zone de subduction, où le fond marin se soulève ou s'affaisse brusquement, transférant de l'énergie à la masse d'eau sus-jacente. Ce phénomène déclenche une onde qui se propage à travers tout le bassin à une vitesse pouvant atteindre plusieurs centaines de kilomètres par heure dans les grandes profondeurs. En s'approchant des côtes, la diminution de la profondeur entraîne un ralentissement et une amplification de la vague, pouvant provoquer des inondations majeures.

Plusieurs événements historiques ont illustré la puissance destructrice des tsunamis : celui du 26 décembre 2004 au large de Sumatra, responsable de plus de 200 000 victimes ; celui du 11 mars 2011 au Japon (séisme de Tohoku, $M_w = 9.0$), qui a entraîné la catastrophe nucléaire de Fukushima ; et plus récemment, celui du 29–30 juillet 2025 au large du *Kamtchatka* ($M_w = 8.8$), dans la ceinture de feu du Pacifique.

La compréhension et la modélisation de la propagation des tsunamis constituent un enjeu majeur pour la mise en place de systèmes d'alerte efficaces. Déterminer la position et le temps d'origine de la source à partir des temps d'arrivée observés dans différentes stations côtières est une étape clé pour améliorer la réactivité et la fiabilité de ces dispositifs.

1.2 Théorie et méthodes de localisation

Le principe général de localisation d'une source de tsunami repose sur la mesure des *temps d'arrivée* de l'onde dans plusieurs stations réparties sur le globe. En comparant ces temps, il est possible de *trianguler* la position de la source en supposant une propagation le long des *grands cercles* reliant la source aux stations.

Dans un modèle d'eau profonde, la vitesse de phase de l'onde dépend uniquement de la profondeur locale h selon la relation :

$$v = \sqrt{g\,h}$$

où $g = 9.81\ \mathrm{m/s^2}$ est l'accélération de la pesanteur. Ce modèle, dit *non dispersif*, néglige les effets de réfraction, de dispersion et de friction, mais fournit une première approximation réaliste du champ de propagation à l'échelle océanique.

Plusieurs approches existent pour résoudre le problème inverse consistant à estimer la position de la source à partir des observations. Dans ce projet, nous retenons une méthode fondée sur la *comparaison des retards de détections entre les stations*. En éliminant le temps d'origine inconnu t_0 par différenciation, on évalue un indicateur d'erreur (ou *misfit*) sur une grille de candidats potentiels à la surface du globe. La source du tsunami est alors identifiée comme le point minimisant cet écart, tandis que le temps d'origine est ensuite estimé à partir des temps d'arrivée observés.

Cette approche, bien que simplifiée, permet de relier directement les observations aux paramètres physiques du modèle.

2

Méthodologie et modélisation numérique

2.1 Inconnues et données d'entrée

Inconnues. On cherche les paramètres de la source

$$\mathbf{x}_s = (\lambda_s,\ \varphi_s) \quad \text{et} \quad t_0,$$

où λ_s et φ_s désignent respectivement la latitude et la longitude de la source (en degrés), et t_0 la date d'origine (en UTC).

Stations et observations. On dispose d'un ensemble de stations côtières indexées par $i = 1, \ldots, N$, chacune caractérisée par ses coordonnées

$$\mathbf{x}_i = (\lambda_i, \varphi_i)$$

et par une *date d'arrivée observée* t_i^{obs} . Dans le projet, les horodatages sont convertis en secondes à partir de la première détection, ainsi la référence temporelle sera l'arrivée du tsunami à Los Angeles.

Bathymétrie (ETOPO5). La profondeur de la mer est fournie par la grille globale **ETOPO5** de résolution $5'$ d'arc (environ $9\ \mathrm{km}$ à l'équateur). On note $h(\lambda, \varphi)$ la profondeur d'eau positive (en mètres), obtenue par interpolation bilinéaire de la grille (les altitudes terrestres sont tronquées à $h = 0$). Cette bathymétrie sert à définir (i) un *masque océanique* pour exclure les trajets passant à terre et (ii) la vitesse locale des ondes de tsunami dans l'approximation d'eau profonde.

Modèle d'observation. Chaque station fournit une date d'arrivée t_i^{obs}. Pour une source candidate $\mathbf{x}_s = (\lambda_s, \varphi_s)$, le modèle prédit un temps de propagation $T_i(\mathbf{x}_s)$ calculé le long du grand cercle. Les données sont reliées au modèle par

$$t_i^{\mathrm{obs}} = t_0^* + T_i(\mathbf{x}_s)$$

où t_0^* est l'origine temporelle. Dans l'inversion utilisée ici, t_0^* est estimé pour chaque candidat par la médiane des différences $t_i^{\mathrm{obs}} - T_i(\mathbf{x}_s)$, ce qui fournit un décalage temporel sans recourir à une régression en pente libre. Les résidus $r_i = t_i^{\mathrm{obs}} - (t_0^* + T_i)$ servent ensuite à calculer le misfit quadratique.

Pré-traitements. Les horodatages des stations sont convertis en secondes. Pour chaque paire source–station, la géodésique est discrétisée et filtrée à l'aide d'un masque océanique dérivé d'ETOPO5 afin d'exclure tout point terrestre. La profondeur $h(\lambda, \varphi)$ est interpolée bilinéairement sur ETOPO5 et tronquée à $h \geq 0$. La vitesse locale $v = \sqrt{g\,h}$ est évaluée le long du trajet, permettant de calculer les temps de propagation utilisés dans l'inversion.

Unités et conventions. Sauf mention contraire : h en mètres, v en $\mathrm{m \cdot s^{-1}}$, T_i et t_i^{obs} en secondes, latitudes Nord et longitudes Est positives (longitudes Ouest négatives).

2.2 Modélisation physique

La propagation du tsunami est décrite par le modèle des *ondes de gravité en eau profonde*. Dans ce régime, la longueur d'onde λ est très supérieure à la profondeur h, et la vitesse de phase dépend uniquement de la profondeur moyenne selon :

$$v(h) = \sqrt{g\,h}, \qquad g = 9.81\ \mathrm{m\,s^{-2}}.$$

Le *temps de trajet théorique* d'une onde entre la source $\mathbf{x}_s$ et une station i s'écrit :

$$T_i(\mathbf{x}_s) = \int_0^{L_i} \frac{ds}{\sqrt{g\,h(s)}},$$

où s est la coordonnée le long du grand cercle reliant la source à la station avec $s \in [0, L_i]$, et $h(s)$ la profondeur interpolée sur ce trajet. La propagation du tsunami étant considérée comme rectiligne, on peut assimiler le trajet entre la source et un point d'observation à une ligne droite. Pour décrire cette ligne droite, une fraction de grand cercle est définie. Ainsi peut être calculé le temps de trajet de l'onde entre deux points d'un grand cercle en intégrant la vitesse locale le long de ce trajet.

Hypothèses principales.

- **Terre sphérique :** les distances sont mesurées le long des grands cercles d'une sphère de rayon $R = 6371\ \mathrm{km}$.
- **Pas de réfraction :** les effets de déviation dus aux gradients bathymétriques sont négligés.
- **Pas de dispersion ni de friction :** toutes les fréquences se propagent à la même vitesse $v(h)$ sans perte d'énergie.
- **Onde linéaire :** l'amplitude de la perturbation est supposée faible devant la profondeur moyenne.

2.3 Schéma d'algorithme (pipeline)

La Figure 2.1 illustre l'ensemble du pipeline mis en œuvre pour l'inversion de la source de tsunami. Le processus débute par le chargement de la bathymétrie ETOPO5, interpolée afin de fournir une fonction de profondeur continue $h(\lambda, \varphi)$ ainsi qu'un masque océan/terre. Les dates d'arrivée fournies pour les stations côtières sont ensuite lues et converties en secondes.

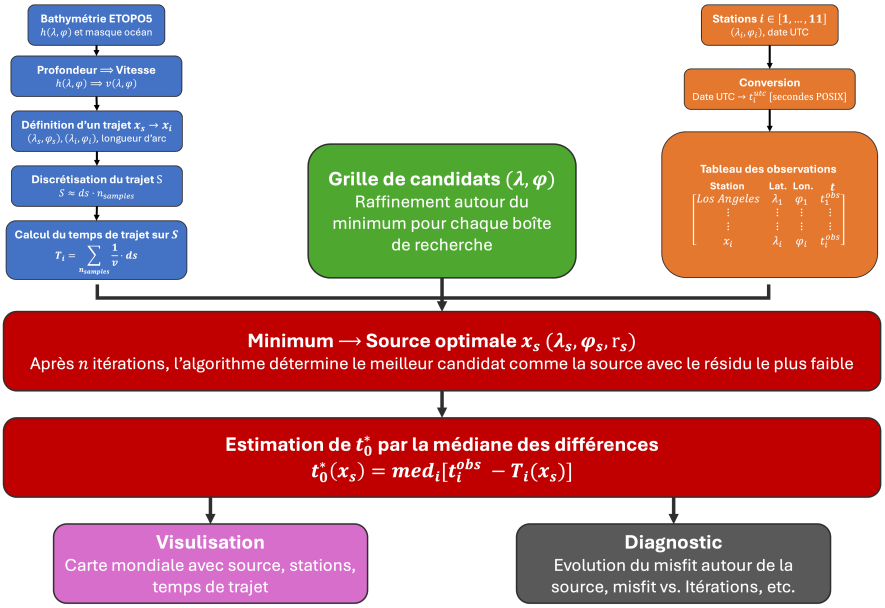


Figure 2.1 – Schéma d'ensemble du pipeline d'inversion : calcul des temps de trajet à partir de la bathymétrie, exploration de la grille de candidats, détermination de la source optimale et estimation du temps d'origine, suivis de la visualisation et des diagnostics.

Une grille de candidats (λ, φ) est construite sur la surface du globe. Pour chaque point candidat, le trajet océanique entre la source supposée et chaque station est calculé le long du grand cercle, avant d'être discrétisé afin d'échantillonner la profondeur locale. La vitesse d'onde $v(s) = \sqrt{g\, h(s)}$ est évaluée le long du trajet et intégrée pour obtenir les temps de propagation théoriques T_i^{mod}.

Les différences de temps modélisées sont alors comparées aux différences observées. Le point présentant le plus faible misfit est alors considéré comme le meilleur candidat. Une nouvelle grille est alors construite et centrée autour de ce point et la même méthode est appliquée. Cette étape peut être répétée jusqu'à ce que la précision soit considérée comme suffisante (L'efficacité du raffinement successif de la grille est illustrée par l'évolution du misfit au cours des itérations *(voir Fig. C.1 en Annexe)*, qui montre une convergence rapide vers un minimum stable)

Le minimum global de ce misfit fournit la position estimée de la source. Une fois cette localisation déterminée, le temps d'origine t_0^* est estimé grâce à la médiane des différences entre les temps observés et les temps modélisés.

Enfin, les résultats sont visualisés sous forme de cartes : trajectoires des rayons géodésiques, positions des stations, source optimale et temps de propagation. En parallèle, des *diagnostics* sont effectués sur le modèle.

3

Résultats et interprétation

3.1 Localisation de la source et trajectoires de propagation

L'inversion absolue appliquée aux temps d'arrivée de onze stations marégraphiques du Pacifique conduit à une position optimale située dans le Pacifique Nord-Est, au large de la façade ouest américaine :

$$\lambda_s \approx 24.48^\circ\mathrm{N}, \qquad \varphi_s \approx 212.13^\circ\mathrm{E}.$$

La Fig. 3.1 montre la bathymétrie ETOPO5, la position estimée de la source, les stations utilisées et les trajets de propagation théoriques évalués le long des grands cercles. Les premières arrivées observées sur Los Angeles, Manzanillo et Acapulco sont cohérentes avec une source située relativement près de la côte américaine.

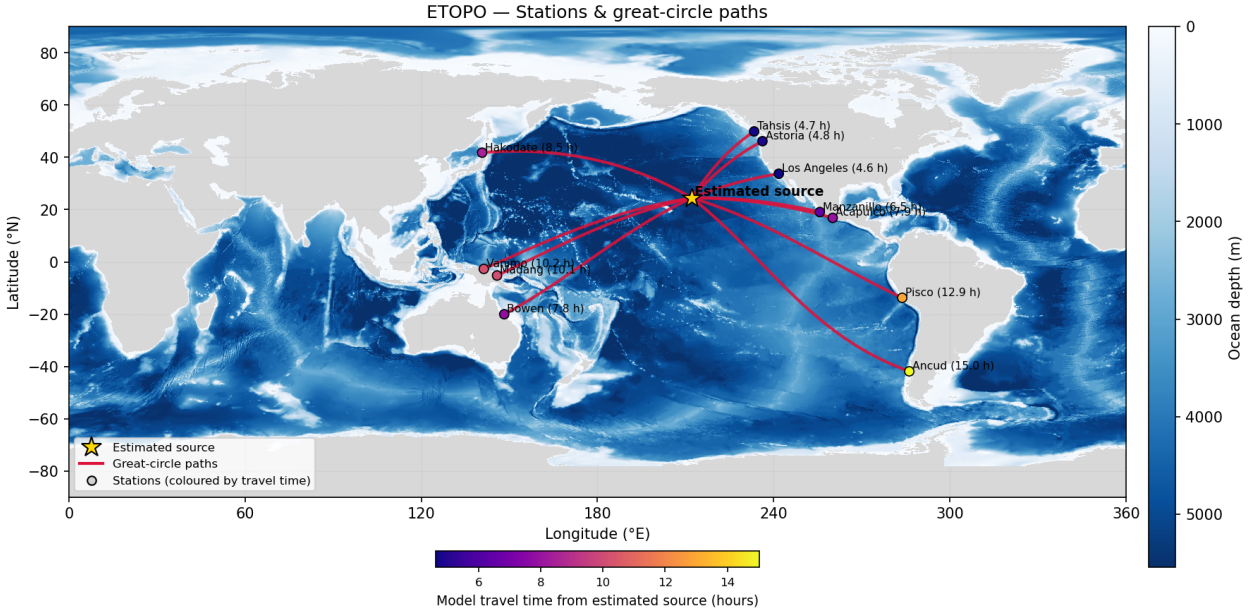


Figure 3.1 – Carte ETOPO5 montrant la source estimée (étoile), les stations utilisées et les trajets de propagation calculés avec leur temps de trajet le long du grand cercle.

3.2 Temps d'origine estimé et diagnostic modèle–données

Pour chaque candidat, le temps d'origine est déterminé comme la médiane des différences entre temps observés et temps modélisés. Pour la source optimale, on obtient

$$t_0^* \approx 1\,323\,790\,329 \text{ s} \;\Rightarrow\; \text{2011-12-13 15 :32 :09 UTC.}$$

3.3 Incertitudes sur la localisation

L'incertitude sur la position de la source résulte de plusieurs facteurs : distribution très asymétrique des stations, bathymétrie utilisée (ETOPO5 à 5' d'arc) et simplifications du modèle de propagation. La bathymétrie lissée tend notamment à moyenner les faibles variations locales de profondeur.

Les profils unidimensionnels du misfit présentés à la Fig. 3.2 permettent d'observer la stabilité locale de la solution. Une coupe en latitude et une coupe en longitude, réalisées autour de la source optimale, montrent un creux bien marqué mais relativement large, dans lequel plusieurs positions voisines produisent des RMSE très similaires. La cuvette est plus étroite en latitude qu'en longitude, ce qui traduit une meilleure contrainte nord–sud liée aux stations américaines, tandis que la direction est–ouest reste moins contrainte en raison de l'absence de stations dans le centre du Pacifique.

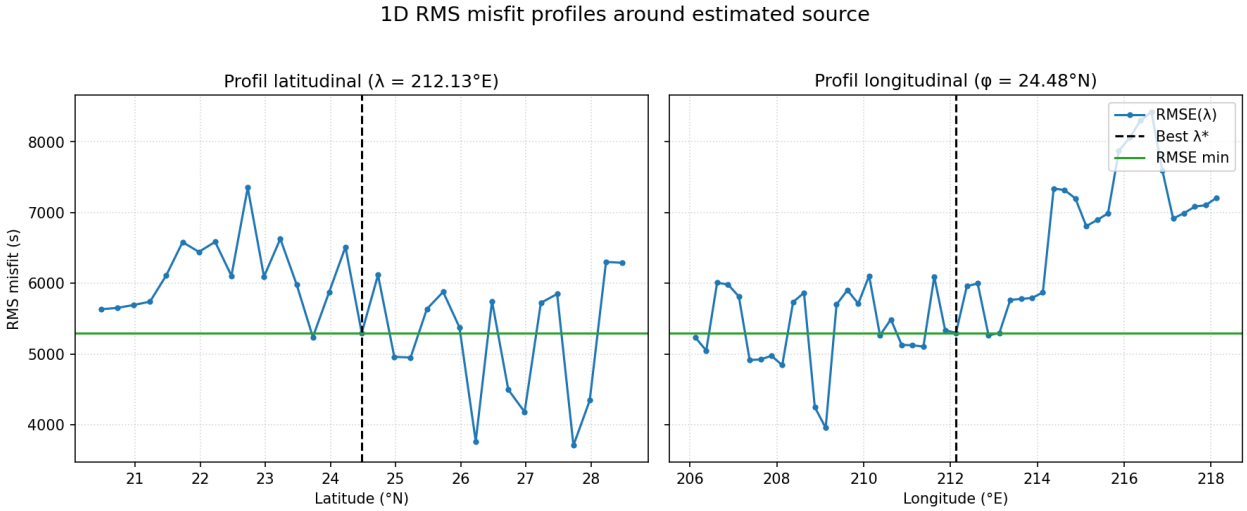


Figure 3.2 – Profils 1D du misfit en latitude (gauche) et en longitude (droite) autour de la solution optimale. La courbe bleue correspond au RMSE, la ligne verte au minimum local et la ligne verticale noire à la source estimée.

Ces deux profils montrent que la région où le misfit reste proche de son minimum s'étend sur plusieurs degrés en latitude comme en longitude. La cuvette latitudinale apparaît un peu plus resserrée que la cuvette longitudinale, ce qui reflète le fait que la position nord–sud de la source est davantage contrainte par les premières arrivées observées sur la façade américaine (Los Angeles, Manzanillo, Acapulco). À l'inverse, la zone centrale du Pacifique est peu instrumentée, ce qui laisse subsister une incertitude plus large dans la direction est–ouest. L'asymétrie du réseau de stations, concentré le long des côtes américaines, est ainsi directement visible dans la forme des deux courbes, la longitude étant significativement moins contrainte que la latitude.

4

Discussion

4.1 Validité et robustesse du modèle

La Fig. 4.1 (A) compare les temps de trajet modélisés et les temps d'arrivée observés. La ligne rouge correspond au modèle contraint $t_i^{\text{obs}} = t_0^* + T_i$, tandis que la régression libre (ligne noire) sert uniquement de diagnostic. Le coefficient de corrélation $R^2 \approx 0.81$ indique une bonne cohérence globale, malgré une dispersion notable. Le RMSE associé à la source optimale vaut $\approx 5\,300$ s (88 min) et le MAE $\approx 2\,574$ s (43 min).

Le choix de la fréquence d'échantillonnage est central dans la durée de calcul. Il est possible d'agir sur différents paramètres, tels que le nombre de points utilisés pour discrétiser le grand cercle, le pas de sommation pour l'obtention du temps de trajet, ainsi que la taille de la grille et la précision imposée. Le programme a déjà été optimisé en utilisant la vectorisation. Néanmoins, en imposant un échantillonnage important sur l'ensemble des paramètres, le calcul peut durer une dizaine de minutes. L'effet d'un mode « lite » a été mesuré. Il apparaît qu'en utilisant un nombre réduit de points pour chaque paramètre *(voir Fig. 4.1)*, la durée de calcul est fortement réduite tandis que le résultat n'est que faiblement dégradé. Le coefficient R^2 passe de 0.811 à 0.756 et la localisation estimée de la source fluctue d'environ 2 degrés autour de la localisation retenue. D'autre part, en augmentant fortement l'échantillonage, les résultats ne semblent pas s'améliorer et convergent vers ceux obtenus initialement avec des variations inférieures au degré pour la localisation et aucune évolution notable du coefficient R^2.

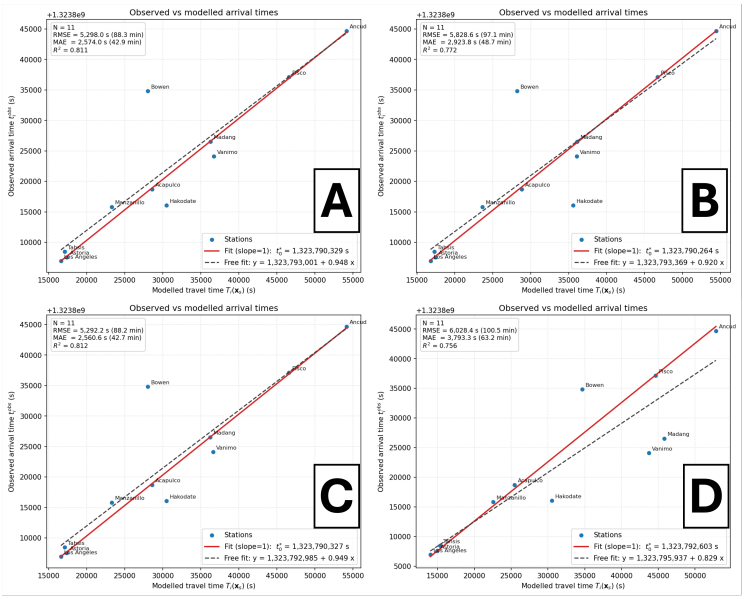


Figure 4.1 – Comparaison temps observés face au temps modélisés et influence des paramètres numériques sur la relation entre temps d'arrivée observés et modélisés. Les quatre panneaux comparent les mêmes données avec des choix différents de résolution de grille et de discrétisation. (A) Paramètres de référence : $n_{\text{samples}} = 2000$, $n_{\text{pts}} = 400$, $n_{\text{grid}} = 15$. (B) Grille plus fine : $n_{\text{grid}} = 30$. (C) Grille plus grossière : $n_{\text{grid}} = 8$. (D) Version « lite » avec paramètres réduits (calcul rapide).

Les stations de la façade américaine (Los Angeles, Acapulco, Astoria, Tahsis) sont parmi celles qui s'ajustent le mieux. Bowen, en revanche, présente un résidu très positif, indiquant un temps modélisé trop court, tandis que Hakodate et Vanimo montrent des résidus négatifs plus importants. Ces écarts reflètent les limites du modèle de propagation. En effet, le trajet entre Bowen et la source traverse plusieurs zones émergées, constituées d'un chapelet d'îles. Ces zones côtières peuvent influencer les résultats en induisant des mécanismes d'ombrage, compliquant la résolution des trajets, ainsi que des effets ondulatoires qui ne sont pas pris en compte dans le modèle de propagation. Il s'agit ici d'une des problématiques majeures de la modélisation.

4.2 Limites et perspectives

Les résultats de la simulation sont dans l'ensemble satisfaisants ; néanmoins, afin d'affiner la modélisation, plusieurs améliorations seraient pertinentes. Il a été retenu d'appliquer le programme à d'autres tsunamis (tel que Tohoku 2011, Tonga 2022) afin de valider la robustesse de la méthodologie. Ce retour d'expérience serait crucial, notamment au regard des écarts importants observés pour les stations australiennes, qui suggèrent des limites fortes dans la représentation des trajectoires de propagation et des interactions côtières. Un élargissement des tests permettrait ainsi de mieux identifier les conditions dans lesquelles le modèle reste fiable et celles pour lesquelles des ajustements sont nécessaires. Comme évoqué précédemment, il apparaît nécessaire d'intégrer un modèle ondulatoire afin de mieux représenter la propagation en zones côtières et insulaires.

A

Codes Python du projet

L'ensemble du code Python développé pour ce projet est disponible en accès libre sur le dépôt GitHub :

`https://github.com/MaxSC4/project-tsunami`

Ce dépôt contient tous les scripts nécessaires au pipeline d'inversion : chargement de la bathymétrie ETOPO5, lecture des observations, calcul des temps de propagation le long des grands cercles, inversion de la source et du temps d'origine, diagnostics et visualisation cartographique. Cette annexe en propose une vue d'ensemble, afin de documenter la structure du projet et le rôle des principaux modules. Les codes complets, avec leurs docstrings détaillées, sont consultables directement sur GitHub.

A.1 Structure du dépôt GitHub

L'organisation du dépôt est la suivante :

```
project-tsunami/

  data/                      # Données d'entrée
     etopo5.grd              # Bathymétrie globale (ETOPO5, ASCII)
     data_villes.csv         # Dates d'arrivée aux stations

  tsunami/                   # Cœur numérique du modèle
     geo.py                  # Géométrie de grand cercle, distances, coordonnées
     speed_model.py          # Modèle de vitesse v(h) = sqrt(g h)
     speed_integrator.py     # Intégration du temps de propagation le long du trajet
     io_etopo.py             # Chargement & interpolation de la bathymétrie ETOPO5
     observations.py         # Lecture des stations et parsing des temps observés
     inverse.py              # Outils génériques d'inversion / misfit
     absolute_inversion.py # Inversion absolue (source + t0* sur grille adaptative)

  plotting/                  # Visualisation et diagnostics
     world_map.py            # Carte globale, stations, source, grands cercles
     diagnostics.py          # Graphiques temps observés / modélisés, métriques globales
     uncertainty.py          # Profils de misfit
     table_arrival_times.py # Tableau station, t_obs, T_mod, résidu

  scripts/
     run_inversion.py        # Script principal : exécution du pipeline complet

  outputs/                   # Résultats numériques et figures générées
```

Le package `tsunami` regroupe ainsi le cœur numérique de la méthode (géométrie, vitesse, intégration, inversion), tandis que le répertoire `plotting` rassemble les outils de visualisation utilisés dans le Chapitre 3. Le script `run_inversion.py` orchestre l'ensemble du pipeline à partir des fichiers de données du répertoire `data` et écrit les résultats dans `outputs`.

A.2 Modules principaux et rôle fonctionnel

Le Tableau A.1 résume le rôle des modules les plus importants dans le pipeline d'inversion utilisé dans ce rapport.

Table A.1 – Principaux modules Python du projet et rôle dans le pipeline numérique.

Composant	Rôle principal	Fichier
Bathymétrie	Chargement ETOPO5, interpolation $h(\lambda, \varphi)$	`tsunami/io_etopo.py`
Géométrie	Grands cercles, distances, coordonnées le long du trajet	`tsunami/geo.py`
Observations	Lecture des stations, parsing des temps	`tsunami/observations.py`
Modèle de vitesse	Vitesse $v(h) = \sqrt{g\,h}$	`tsunami/speed_model.py`
Intégration	Calcul des temps de trajet le long des géodésiques	`tsunami/speed_integrator.py`
Inversion (outils)	Fonctions génériques de misfit et de recherche	`tsunami/inverse.py`
Inversion absolue	Recherche sur grille adaptative, estimation de t_0^*	`tsunami/absolute_inversion.py`
Carte globale	Carte ETOPO5, source, stations, grands cercles	`plotting/world_map.py`
Diagnostics	Graphiques temps observés/modélisés, régression, métriques	`plotting/diagnostics.py`
Incertitudes	Profils 1D de misfit	`plotting/uncertainty.py`
Tableau des temps	Génération du tableau t_i^{obs}, T_i, r_i	`plotting/table_arrival_times.py`
Script principal	Exécution complète : lecture données, inversion, figures	`scripts/run_inversion.py`

Dans la pratique, l'utilisateur n'interagit que avec un nombre limité de points d'entrée : le script `run_inversion.py` permet de lancer l'inversion complète en ligne de commande, tandis que les modules du package `tsunami` peuvent être réutilisés de manière plus fine dans un notebook ou un autre projet (par exemple pour tester d'autres domaines de recherche ou d'autres scénarios d'inversion).

A.3 Accès et réutilisation du code

Les codes complets sont consultables et téléchargeables directement via GitHub :

`github.com/MaxSC4/project-tsunami`

Chaque module comporte une en-tête décrivant sa fonction et son rôle dans le pipeline, ainsi que des docstrings pour les principales fonctions (chargement de la bathymétrie, calcul des temps de trajet, inversion, diagnostics). Cette organisation modulaire permet de réutiliser facilement les composants du projet, par exemple pour tester d'autres modèles de vitesse, changer le domaine géographique ou adapter l'inversion à un autre jeu de stations.

B

Extraits du code Python

Cette annexe rassemble plusieurs extraits significatifs du code Python développé dans le cadre du projet. Ils illustrent les différentes briques du pipeline : géométrie sphérique, calcul des temps de propagation, intégration le long du grand cercle, recherche de la source par inversion, et exécution complète via le script principal. Les versions complètes de ces modules sont disponibles sur le dépôt GitHub du projet.

`github.com/MaxSC4/project-tsunami`

B.1 Calcul de la longueur d'arc sur la sphère (`geo.py`)

```python
def arc_length_m(lat1, lon1, lat2, lon2):
    if lat1 == lat2 and lon1 == lon2:
        return 0.0

    phi1, lam1, phi2, lam2 = map(np.deg2rad, (lat1, lon1, lat2, lon2))
    c = (np.sin(phi1) * np.sin(phi2) + np.cos(phi1) * np.cos(phi2) * np.cos(lam2 - lam1))

    c = np.clip(c, -1.0, 1.0)
    w = np.arccos(c)
    return R_EARTH * w
```

Figure B.1 – Fonction `arc_length_m` : calcul de la distance entre deux points géographiques à l'aide de la formule du cosinus sphérique. Cette opération constitue la base des calculs de trajectoires et distances dans le modèle.

B.2 Calcul du temps de propagation sur un grand cercle (`speed_integrator.py`)

```python
def travel_time_seconds(lat1, lon1, lat2, lon2, depth_fn, n_samples=2000, h_min=0.0):
    """
    Calcule le temps de propagation d'un tsunami (en secondes)
    entre deux points (lat, lon) donnés.

    Paramètres
    ----------
    lat1, lon1 : float
        Point de départ (degrés).
    lat2, lon2 : float
        Point d'arrivée (degrés).
    depth_fn : callable
        Fonction profondeur(lat, lon) → profondeur d'eau (m, POSITIVE en mer).
    n_samples : int, optionnel
        Nombre de points d'échantillonnage le long du grand cercle
        (plus grand = plus précis, mais plus lent).
    h_min : float, optionnel
        Profondeur minimale (m) utilisée pour éviter v = 0
        dans les zones très peu profondes. h effective = max(h, h_min).

    Retour
    ------
    float
        Temps de trajet en secondes.

        - 0.0 si les deux points sont identiques
        - +inf si le trajet ne possède aucun segment océanique valable (tout sur la terre ou très fragmenté).
    """
    # Cas trivial : même point
    dist = arc_length_m(lat1, lon1, lat2, lon2)
    if dist == 0.0:
        return 0.0

    # 1) Points le long du grand cercle
    n_samples = max(3, int(n_samples))
    pts = great_circle_points(lat1, lon1, lat2, lon2, npts=n_samples)
    lats = pts[:, 0]
    lons = pts[:, 1]

    # 2) Profondeurs correspondantes
    depths = _safe_depth(depth_fn, lats, lons)

    # 3) Masque océan (profondeur d'eau > 0)
    ocean = np.isfinite(depths) & (depths > 0.0)

    # 4) Vitesse locale
    v = speed(h)
    if not np.all(np.isfinite(v)):
        return float("inf")

    # 5) Intégration numérique du temps total
    ds = dist / (len(lats) - 1)
    inv_v = 1.0 / v
    T = np.sum(inv_v[:-1]) * ds
    return float(T)
```

Figure B.2 – Fonction `travel_time_seconds`. Elle calcule le temps de propagation d'un tsunami entre deux points en intégrant numériquement $1/v(h)$ le long d'un grand cercle discrétisé. Cette fonction est au cœur du modèle direct utilisé par l'inversion.

B.3 Raffinement de la grille d'inversion (`absolute_inversion.py`)

```python
# --- boucle de raffinement de la grille ---
    while (
        (phi_max - phi_min) > precision_deg
        and (lam_max - lam_min) > precision_deg
        and tries < max_iter
    ):
        # 1. Grille de candidats
        lats = np.linspace(phi_min, phi_max, n)
        lons = np.linspace(lam_min, lam_max, n)
        dlat = (phi_max - phi_min) / n
        dlon = (lam_max - lam_min) / n
        grid_lat, grid_lon = np.meshgrid(lats, lons, indexing="ij")
        Np = grid_lat.size

        # 2. Temps de trajet modèles Tmod[p, i]
        Tmod = np.full((Np, Ns), np.nan, float)

        for j in range(Ns):
            lat_s, lon_s = stations[j]
            for p in range(Np):
                Ti = travel_time_seconds(
                    grid_lat.flat[p],
                    grid_lon.flat[p],
                    lat_s,
                    lon_s,
                    depth_fn,
                )
                Tmod[p, j] = np.nan if not np.isfinite(Ti) else float(Ti)

        # 3. Misfit en temps relatifs par rapport à la station ref_index
        #   ΔT_model[p, i] = Tmod[p, i] - Tmod[p, ref_index]
        # On ne garde que les stations i pour lesquelles les deux trajets (src→i, src→ref)
        # sont marins (Tmod finite).
        model_ok = np.isfinite(Tmod)
        ref_ok   = model_ok[:, ref_index][:, None]   # (Np,1)
        valid    = model_ok & ref_ok                 # (Np, Ns)

        # Temps relatifs modélisés
        T_ref = Tmod[:, ref_index][:, None]          # (Np,1)
        dt_model = Tmod - T_ref                      # (Np, Ns)

        # Résidus relatifs
        dt_obs_mat = np.broadcast_to(dt_obs, (Np, Ns))
        resid = dt_obs_mat - dt_model
        resid[~valid] = np.nan

        # Misfit = RMS des résidus relatifs (en secondes)
        rmse = np.sqrt(np.nanmean(resid**2, axis=1))
        valid_counts = np.sum(valid, axis=1)

        # On met rmse=inf là où il n'y a pas assez de stations marines
        rmse = np.where(valid_counts ⩾ min_valid_stations, rmse, np.inf)

        # 4. Sélection du meilleur candidat de cette grille
        if not np.any(np.isfinite(rmse)):
            if verbose:
                print(f"[Iter {tries+1}] Aucun candidat valide (rmse=inf partout), arrêt.")
            break

        idx_best = int(np.nanargmin(rmse))
        cand = dict(
            lat=float(grid_lat.flat[idx_best]),
            lon=float(grid_lon.flat[idx_best]),
            rmse_rel=float(rmse[idx_best]),
            valid=int(valid_counts[idx_best]),
            iter=int(tries + 1),
        )

        # Mise à jour du meilleur global
        if cand["rmse_rel"] < best["rmse_rel"]:
            best.update(cand)

        # 5. Raffinement autour du meilleur point global courant
        phi_min = best["lat"] - dlat
        phi_max = best["lat"] + dlat
        lam_min = best["lon"] - dlon
        lam_max = best["lon"] + dlon
        tries += 1

    # Position finale de la source
    best_lat = best["lat"]
    best_lon = best["lon"]
```

Figure B.3 – Boucle principale de raffinement de la grille dans l'inversion absolue. Cette portion de code génère une grille de candidats, calcule les temps de trajet, mesure le misfit relatif pour chaque point, sélectionne le meilleur candidat et resserre la boîte de recherche jusqu'à convergence.

B.4 Pipeline complet d'inversion (`run_inversion.py`)

```python
def run_pipeline(
    etopo_path="data/etopo5.grd",
    stations_csv="data/data_villes.csv",
    lon_mode="180",                 # '180' ([-180,180)) ou '360' ([0,360))
    search_box=None,                # (lat_min, lat_max, lon_min, lon_max) ou None → auto
    grid_n=15,                      # résolution initiale de la grille d'inversion
    precision_deg=0.2,              # critère d'arrêt (taille de boîte < precision_deg)
    max_iter=6,                     # itérations max de raffinement
    robust=True,                    # inversion robuste (médiane, clipping outliers)
    make_map=True,                  # tracer la carte finale
    save_map_path="outputs/final/world_map_inversion.png",
    make_diagnostics=False,
    diag_path="outputs/final/obs_vs_model.png",
    use_absolute=True,              # True = absolute_inversion
):
    """
    Exécute le pipeline d'inversion et (optionnellement) produit une carte.
    Retourne un dict de résultats (lat, lon, t0, misfit, etc.)
    """
    # --- 1) Bathymétrie ---
    print("→ Loading ETOPO grid...")
    lats, lons, H, _ = load_etopo5(etopo_path)
    depth_fn = make_depth_function(lats, lons, H, order=1)

    # --- 2) Observations ---
    print("→ Loading stations & observed arrival times...")
    df, t_obs_s = load_arrival_times(stations_csv)
    stations = _stations_array(df)
    station_list = _to_station_list(df)
    Ns = len(stations)
    print(f"   → {Ns} stations loaded.")

    # --- 3) Boîte de recherche ---
    if search_box is None:
        lat_min, lat_max, lon_min, lon_max = _auto_search_box(stations, margin_deg=10.0)
        print(f"→ Auto search box: {lat_min:.1f}–{lat_max:.1f}°N / {lon_min:.1f}–{lon_max:.1f}°E")
    else:
        lat_min, lat_max, lon_min, lon_max = map(float, search_box)
        print(f"→ User search box: {lat_min:.1f}–{lat_max:.1f}°N / {lon_min:.1f}–{lon_max:.1f}°E")

    # --- 3bis) Trouver l'index de la station de référence (Los Angeles) ---
    name_col = "Ville/Port" if "Ville/Port" in df.columns else "name"
    names_raw = df[name_col].astype(str)

    ref_index = 0
    try:
        ref_index = names_raw.tolist().index("Los_Angeles")
    except ValueError:
        try:
            ref_index = names_raw.str.replace("_", " ").tolist().index("Los Angeles")
        except ValueError:
            print("[run_pipeline] Warning: 'Los_Angeles' non trouvée, station[0] utilisée comme référence.")
            ref_index = 0

    # --- 4) Inversion ---
    print("→ Running inversion...")

    if use_absolute:
        # inversion "centrée Los Angeles"
        best_lat, best_lon, t0_hat, stats = absolute_inversion(
            stations=stations,
            t_obs_s=t_obs_s,
            depth_fn=depth_fn,
            phi_min=lat_min, phi_max=lat_max,
            lam_min=lon_min, lam_max=lon_max,
            ref_index=ref_index,
            n=grid_n,
            precision_deg=precision_deg,
            max_iter=max_iter,
            min_valid_stations=10,
            robust=robust,
            verbose=True,
        )

    print("\n=== Inversion result ===")
    print(f"Source latitude   : {best_lat:.3f}°")
    print(f"Source longitude  : {best_lon:.3f}°")

    # Conversion du temps (secondes) en date UTC
    try:
        t0_datetime = datetime.datetime.fromtimestamp(t0_hat, tz=datetime.timezone.utc)
        t0_str = t0_datetime.strftime("%Y-%m-%d %H:%M:%S %Z")
        print(f"Estimated t0*     : {t0_hat:.1f} s (POSIX time)")
        print(f"                  → {t0_str}")
    except (OSError, OverflowError, ValueError):
        print(f"Estimated t0*     : {t0_hat:.1f} s (not a valid POSIX timestamp)")
```

FIGURE B.4 – Extrait du script principal `run_inversion.py`, qui orchestre l'ensemble du pipeline : chargement des données, appel à l'inversion absolue, calcul des diagnostics, génération des figures et export des résultats.

C

Figures intermédiaires

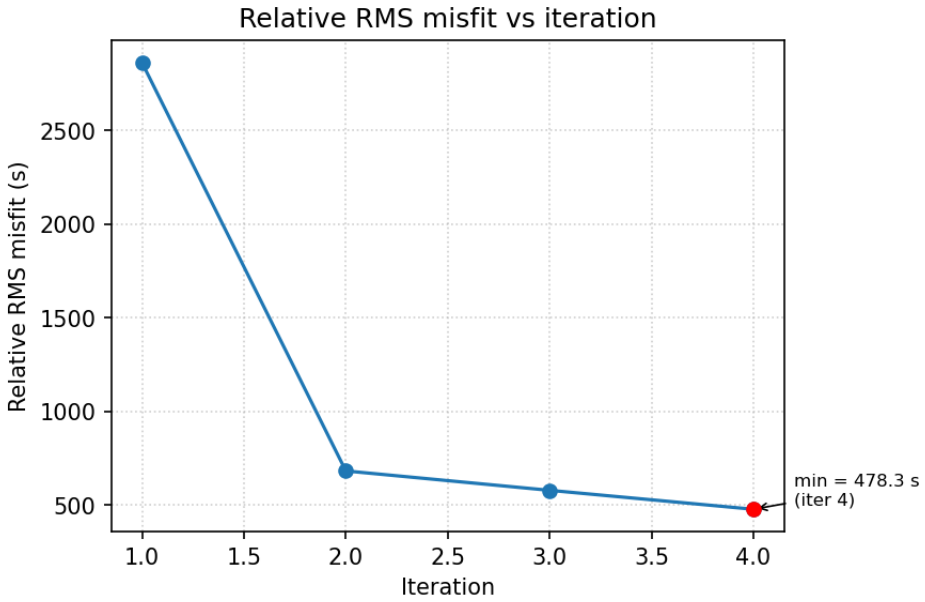
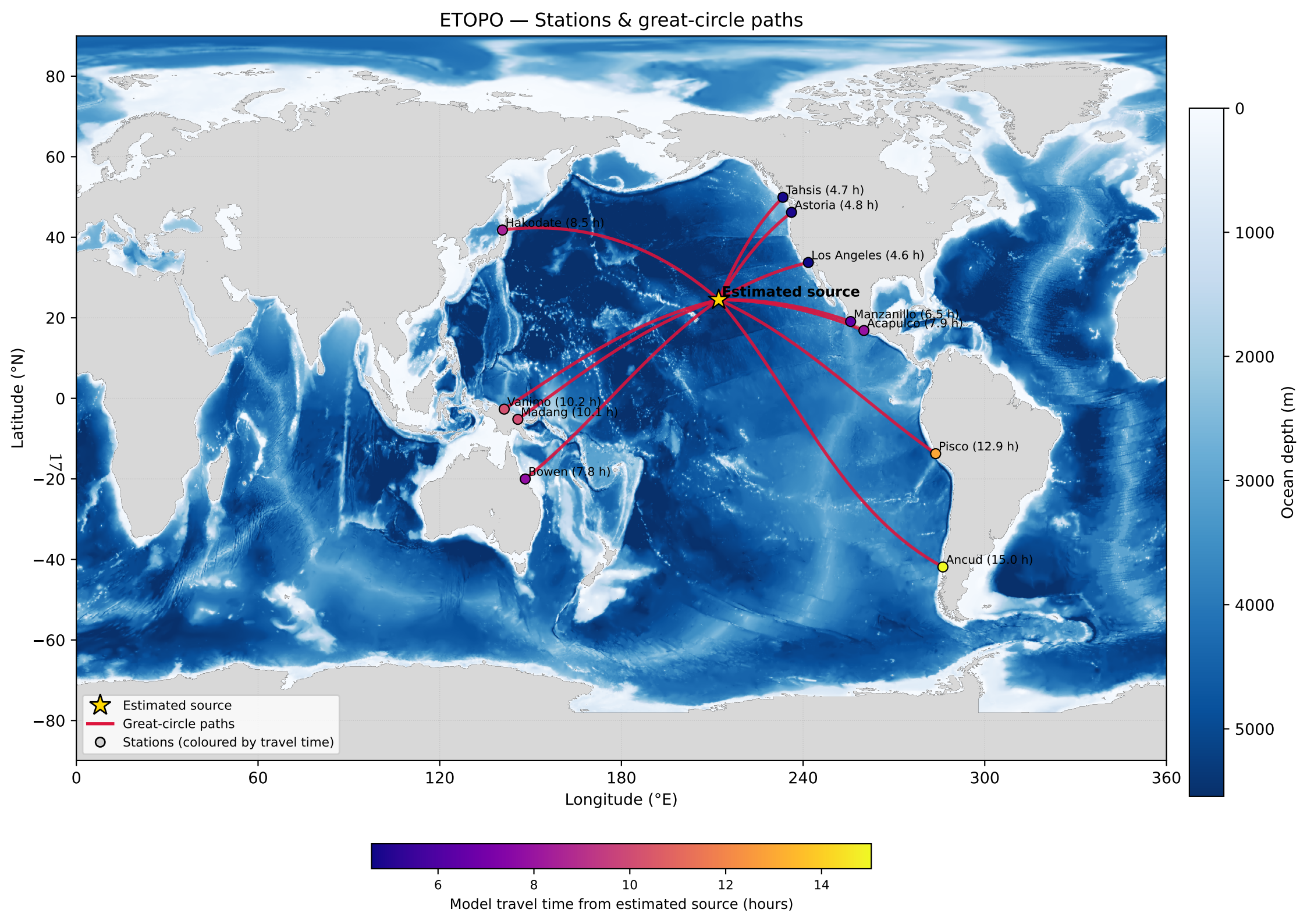


Figure C.1 – Évolution du misfit relatif (RMSE) au cours des itérations de la recherche sur grille. Chaque point correspond au meilleur misfit obtenu pour une boîte de recherche donnée. La forte diminution entre les itérations 1 et 2 illustre l'efficacité du premier raffinement de la grille, tandis que les itérations suivantes montrent une convergence progressive vers un minimum stable (~ 480 s à l'itération 4). Cette courbe fournit un diagnostic sur la robustesse de l'inversion et confirme que la localisation finale n'est pas sensible à la taille de la boîte initiale.


ETOPO — Stations & great-circle paths
Tahsis (4.7 h)
Astoria (4.8 h)
Hakodate (8.5 h)
Los Angeles (4.6 h)
Estimated source
Manzanillo (6.5 h)
Acapulco (7.9 h)
Vanimo (10.2 h)
Madang (10.1 h)
Pisco (12.9 h)
Bowen (7.8 h)
Ancud (15.0 h)
Estimated source
Great-circle paths
Stations (coloured by travel time)
Latitude (°N)
Longitude (°E)
Ocean depth (m)
Model travel time from estimated source (hours)

D

Tableau des résultats numériques

TABLE D.1 – Temps d'arrivée observés t_i^{obs}, temps modélisés $T_i(\mathbf{x}_s)$ et résidus $r_i = t_i^{\text{obs}} - (t_0^* T_i(\mathbf{x}_s))$ pour la source optimale avec les paramètres de référence (voir 4.1 (A)). Un résidu positif indique un temps d'arrivée observé plus tardif que prévu (modèle trop rapide), un résidu négatif indique un modèle trop lent.

Station	t_i^{obs} **(s)**	$T_i(\mathbf{x}_s)$ **(s)**	r_i **(s)**
Bowen	1323834827	28020	16478
Madang	1323826532	36266	-63
Vanimo	1323824088	36710	-2951
Hakodate	1323816071	30477	-4735
Manzanillo	1323815812	23286	2197
Astoria	1323807546	17384	-167
Acapulco	1323818717	28603	-215
Los Angeles	1323806934	16605	0
Ancud	1323844691	54134	228
Pisco	1323837127	46570	228
Tahsis	1323808468	17087	1052

Table des figures

Liste des tableaux