Rapport de soutenance intermédiaire

ST7 - Simulation à haute performance pour la réduction d'empreinte

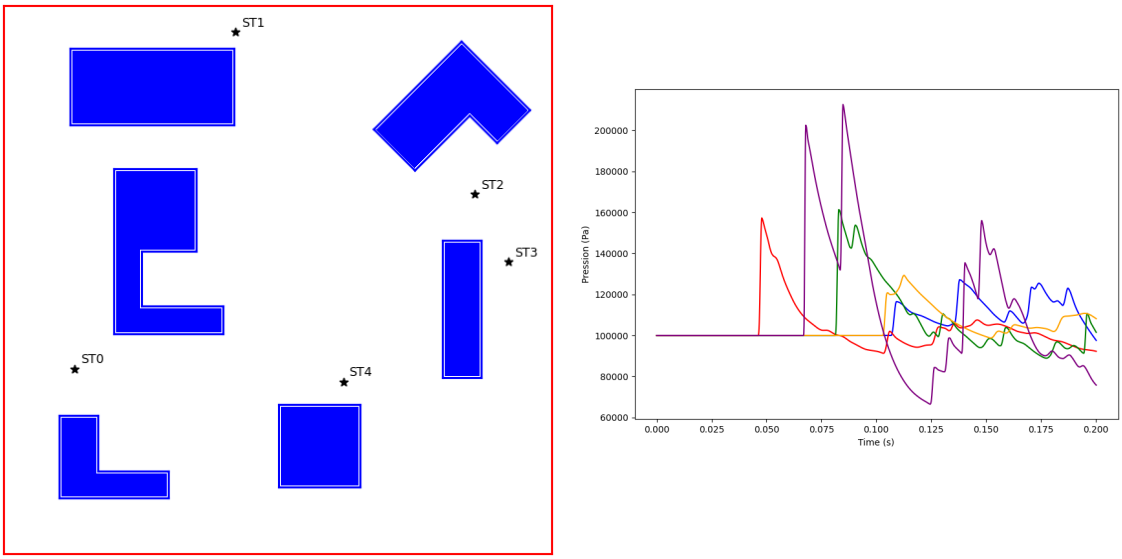
CEA-DAM : Localisation de source

Jonathan POLI
Tom LABIAUSSE
Mathis PASTORELLI
Pierre LABOURE
Thibault ROUSSET

16 Mars 2022

Contents

1 Introduction



(a) Terrain 2D

(b) Relevés de pression des 5 capteurs

Figure 1: Problème de localisation de source

2 Etude d'un cas simplifié

Hypothèses simplificatrices

Afin de réaliser une première étude mathématique, nous allons faire plusieurs hypothèses très réductrices :

- L'onde propagée par l'explosion est isotrope et homogène, se déplaçant à une vitesse v connue, à savoir celle du son dans l'air : $v = 340m.s^{-1}$
- L'étude est réalisée en 2D et en négligeant la présence de bâtiments

Idée générale

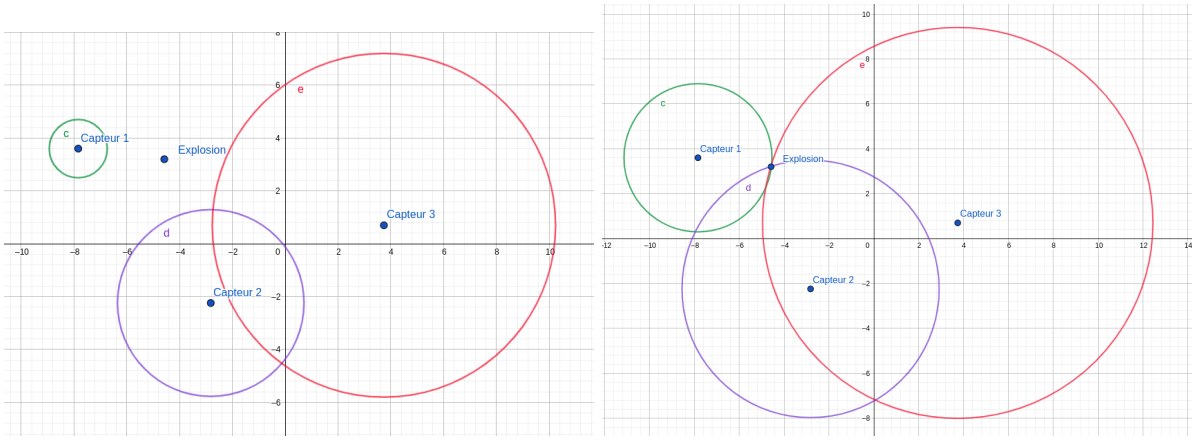
Bien que le temps initial $t_{explosion}$ ne puisse pas être directement lue sur un capteur, il semble convenable de penser que 3 capteurs suffisent pour connaître l'emplacement exacte de la source, par triangularisation.

Démarche et preuve

Lors de l'explosion, l'onde va se propager dans toutes les directions à la vitesse v et ainsi être détectée par chacun des capteurs lorsque l'onde aura parcourue la distance la séparant du capteur.
En prenant le problème à l'envers, pour un capteur, c'est similaire à considérer que l'explosion à lieu au niveau du capteur, et attendre qu'elle parcourt la distance à la source d'origine.

Ainsi, en notant (x_i, y_i) les coordonnées de chaque capteur, (x, y) les coordonées de la source de l'explosion, t_i la durée entre l'explosion et l'instant où l'explosion est détectée par chaque capteur, t_i^{mes} l'instant où l'explosion est détectée par chaque capteur et t_{exp} l'instant de l'explosion, on obtient :

$$\forall i, \begin{cases} (x_i - x)^2 + (y_i - y)^2 = (vt_i)^2 \\ t_i = t_i^{mes} - t_{exp} \end{cases}$$



(a) Cercles obtenus pour un certain t_{exp} non solution (b) Obtention de la solution par intersection des cercles

Figure 2: Recherche de la solution par inversion du problème avec 3 capteurs

Ensuite, par méthode graphique, on se rend compte qu'il exsite une infinité de solutions pour 2 capteurs et un nombre fini non nul de solutions pour 3 capteurs (sauf s'ils sont alignés). Nous ne considérons pas le cas de 4 capteurs qui risque de nous diminuer le nombre de solutions à zéro à cause des hypothèses effectuées. Nous choisissons donc d'utiliser à chaque fois des trio de capteurs.
On obtient donc un système d'équations à trois inconnues : x, y et t_exp.

$$\begin{cases} (x_1 - x)^2 + (y_1 - y)^2 = v^2(t_1^{mes} - t_{exp})^2 \\ (x_2 - x)^2 + (y_2 - y)^2 = v^2(t_2^{mes} - t_{exp})^2 \\ (x_3 - x)^2 + (y_3 - y)^2 = v^2(t_3^{mes} - t_{exp})^2 \end{cases}$$

Résultats

Algorithme Nous allons sélectionner plusieurs trio de capteurs successivement. Pour chaque groupe de trois capteurs, nous allons résoudre le système d'équations obtenues par méthode numérique, en imposant $t_{exp} \leq t_i^{mes}$ afin de limiter les solutions physiquement invalides.
Les différents points (x, y) obtenus sont des possibles solutions approchées au problème initial.

Simulations Nous avons réaliser 10 simulations, 5 sans bâtiments et 5 avec des bâtiments afin d'observer la qualité des résultats.

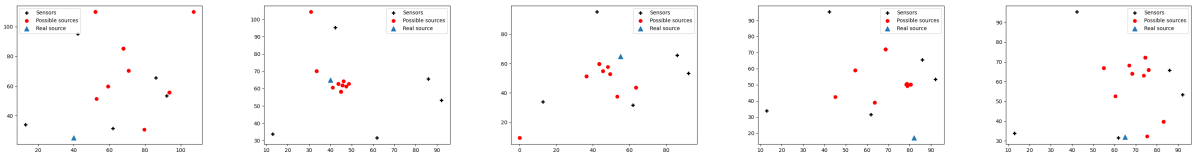


Figure 3: Simulations avec bâtiments

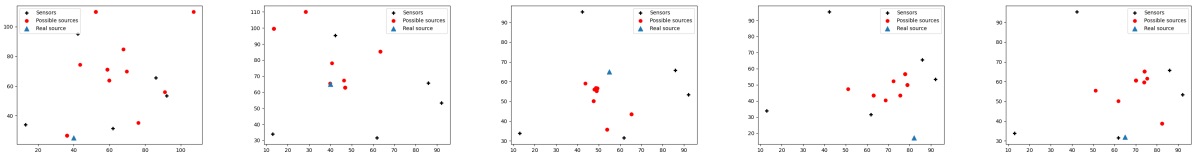


Figure 4: Simulations sans bâtiments

Analyse des résultats Les résulats sont parfois très bons, avec la grande majorité des points possibles proche de la véritable source, parfois un peu mauvais, avec au moins une source proche de la solution réelle

et enfin mauvais, avec aucun point à proximité de la solution.
Ensuite, de manière générale, les résultats sont plus précis lorsqu'il n'y pas de bâtiment, ce qui fait sens au vu des hypothèses.

Interprétation Plusieurs problèmes peuvent être à l'oeuvre :

- l'hypothèse de la vitesse constante $v = 340m.s^{-1}$ est possiblement très fausse.
- Le solveur numérique ne converge pas correctement vers la solution réelle
- Le solveur numérique converge possiblement vers une solution mathématiquement acceptable qui n'est pas la bonne physiquement.

3 Ecart entre deux signaux - fonction de coût

Afin d'évaluer la pertinence d'une estimation (X, Y) de la position de la source, nous n'avons d'autre choix que d'effectuer une simulation avec une source positionnée en (X, Y). On compare ensuite les relevés de pression en simulation à ceux obtenus dans le cas réel. Il s'agit donc de pouvoir déterminer si deux points de l'espace 2D considéré sont proches en comparant uniquement deux séries de relevés de pression. Nous devons donc déterminer une fonction de coût permettant d'évaluer la "distance" entre deux relevés de pression.

Pour cela, il faut tout d'abord synchroniser les signaux. Nous nous basons donc sur une station qui sert de référence. De là, nous repérons les positions des premiers maxima des deux signaux sur chacune des station, ce qui permet, une fois que cela est fait, d'effectuer une comparaison sur les autres stations avec un même point de départ pour les signaux. Ainsi, deux signaux de formes identiques mais ayant été translatés temporellement n'auront pas une distance importante.

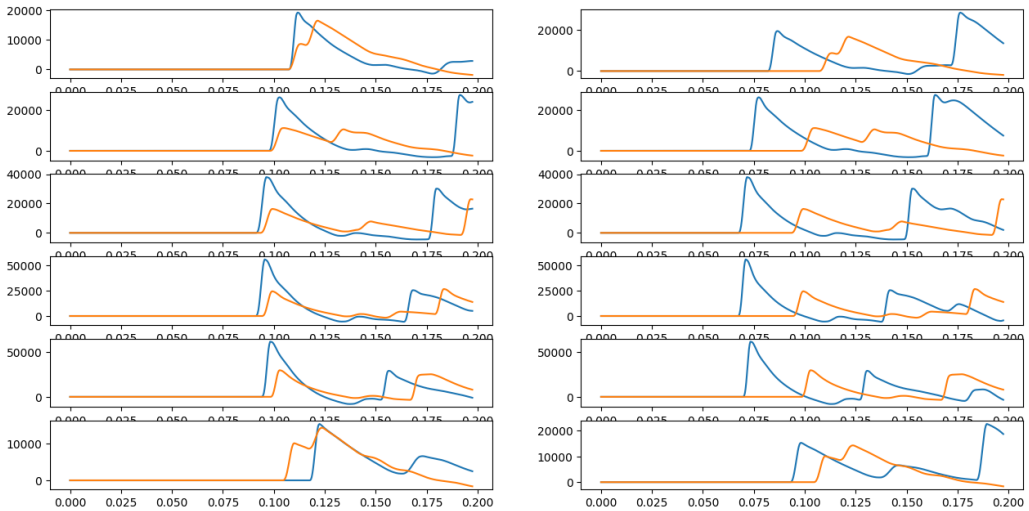


Figure 5: Synchrnonisation des signaux oranges sur les signaux bleus à partir du retard sur la première station

Il convient, une fois cela fait, de définir une distance entre deux signaux. Ce qui nous intéresse principalement ici est la position relative des formes (et maximas) des deux signaux. Pour effectuer une telle comparaison, nous calculons l'intercorrélation des deux signaux. Nous nous intéressons à ces caractéristiques car, a priori, deux signaux acoustiques captés par des sources espacées spatialement ne devraient (en l'absence de bâtiments) différer que par leurs abscisses. En présence de bâtiments, l'on peut supposer que si les deux points sont raisonnablement espacés, l'influence des bâtiments est minime sur la distorsion des signaux (sauf dans le cas d'un passage line-of-sight à non-line-of-sight, ce qui peut se produire sur certains capteurs mais,

a priori, pas sur tous les capteurs en même temps)

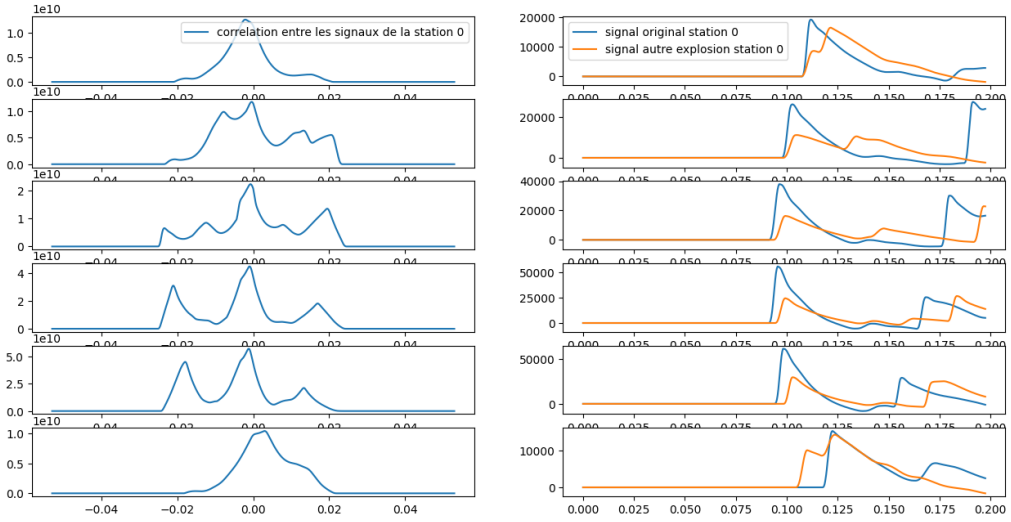


Figure 6: Calcul des corrélations entre les signaux synchronisés

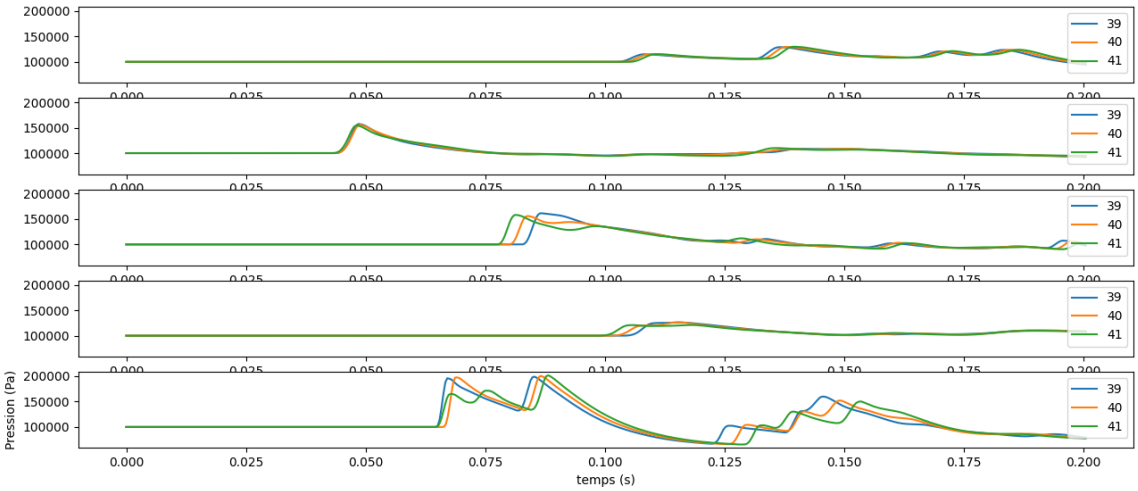


Figure 7: Explosions décalées d'un mètre les unes des autres (coordonnées : $y = 65m, x \in 39m, 40m, 41m$

On constate ici que la seule variation majeure observable entre les signaux est une différence temporelle : leur forme est similaire, mais plus deux singaux sont éloignés, plus le retard est important.

Nous estimons ensuite la distance entre les deux signaux via la norme 2 de la différence de l'intercorrélation et de l'autocorrélation du signal considéré comme le signal source. Ceci permet de mesurer la "ressemblance" du signal simulé vis-à-vis du signal mesuré.

$$d(x_{sim}, x_{mes}) = \int (\gamma_{x_{sim},x_{sim}}(t) - \gamma_{x_{sim},x_{mes}}(t))^2 dt$$

$$\text{où } \gamma_{x,y}(\tau) = \int x(t)y(t-\tau)dt$$

4 Approche par un algorithme de recherche locale

La première approche que nous avons décidée de mettre en oeuvre est un algorithme de recherche locale. Cette méthode s'appuie sur la notion de voisinage définie dans l'ensemble des solutions, qui est dans notre cas 2D un sous-ensemble borné de $\mathbb{R}^2$. Afin de définir un tel concept de voisinage qui conviendrait à notre problème, nous avons discrétisé l'espace à l'aide d'un quadrillage régulier dont la finesse peut varier en ajustant la taille des mailles. Chaque noeud représentant ainsi un point de l'espace. La **figure 8** présente un exemple d'une telle discrétisation.

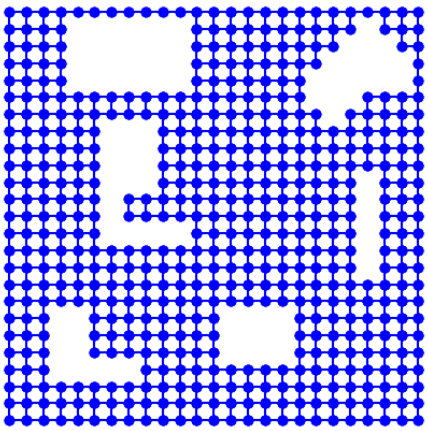


Figure 8: Discrétisation de l'espace 2D par une grille

Pour chaque point (X, Y) du maillage, on choisit ensuite de définir son voisinage comme étant l'ensemble des points avec lesquels il se trouve directement lié sur le quadrillage. La **figure 9** donne un exemple où le point orange posséde huit voisins verts. La présence de bâtiment supprimant des noeuds du maillage, un point peut posséder entre 2 et 8 voisins.

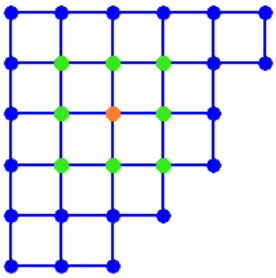
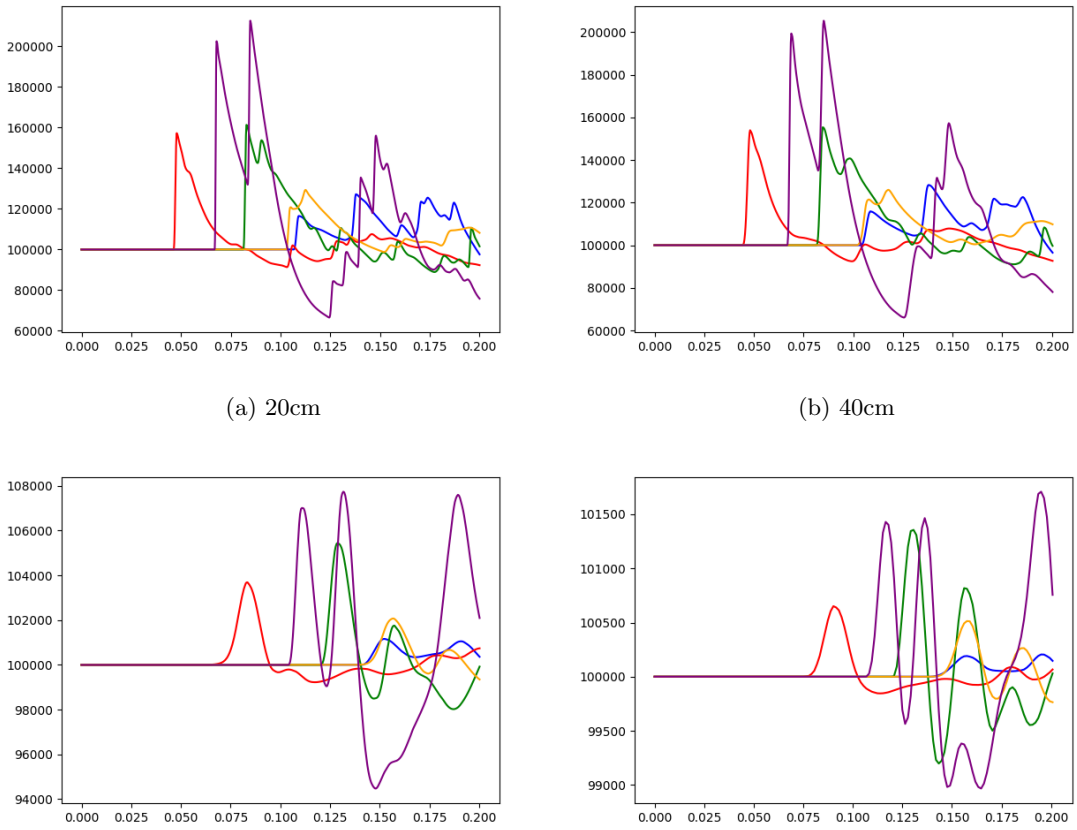


Figure 9: Exemple de voisinage

Le premier algorithme testé consiste à enchaîner des étapes de recherches exhaustives de la meilleure estimation de source parmi tous les voisins d'un point initial. Autrement dit on implémente une *greedy local search*. Cette procédure nécessite donc d'effectuer jusqu'à 8 simulations pour identifier le meilleur voisin d'un point. Une grande importance est donc accordée au temps d'exécution de chaque simulation. Nous avons donc tenté d'augmenter le pas spatial afin d'accélérer les simulations. Ainsi, à titre d'exemple, le temps d'exécution varie de $38s$ à $15s$ lorsqu'on augmente le pas de $0.4m$ à $0.8m$. Cependant, comme on peut le voir en **figure 10**, les signaux relevés sont affectés de manière non négligeable. Trop augmenter le pas risquerait donc de fausser les calculs de coût d'une solution empêchant ainsi la convergence de l'algorithme vers une solution optimale. On se limitera donc à un pas spatial de $60cm$.

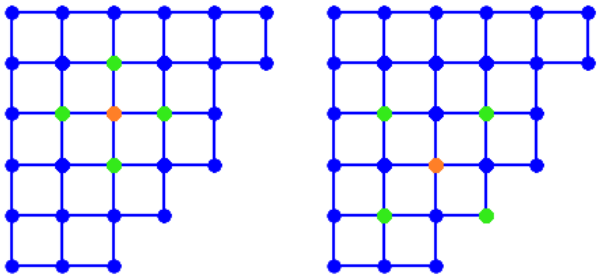


(a) 20cm (b) 40cm

(c) 60cm (d) 80cm

Figure 10: Exemples de relevés de pression avec différents pas spatiaux de simulation

Un autre moyen d'accélérer l'exécution de l'algorithme consiste à n'évaluer qu'un sous-ensemble des voisins d'un point donné. A titre d'exemple, au lieu d'évaluer les 8 voisins les plus proches, on peut se contenter d'en évaluer seulement 4 en choisissant (ou alternant) les deux styles d'évaluation suivants :



(a) Evaluation "droite" (b) Evaluation "diagonale"

5 Approche par un algorithme génétique

5.1 Présentation du processus global

L'idée derrière l'utilisation d'un algorithme génétique est d'éviter au maximum les minima locaux de la fonction de coût précédemment définie à l'aide de la similarité entre signaux. Les étapes de l'algorithme génétique que nous avons développé sont les suivantes :

1. Initialisation $n = 0$: Génération aléatoire de g_0 individus dans l'espace de recherche $[0, 100] \times [0, 100]$. Evaluation du coût associé à chacun des g_0 points (i.e. estimation de la qualité de chaque point en tant que source recherchée).

2. Etape de reproduction : Sélection aléatoire de αg_n couples d'individus parmi les g_n individus de la population. Création d'un point "enfant" à partir de chacun des αg_n couples. Evaluation du coût associé à chacun des αg_n points créés.

3. Etape d'élimination : Elimination des $\beta(\alpha + 1)g_n$ individus dont le coût est le plus élevé.

4. Bilan de population : $g_{n+1} = (1 - \beta)(\alpha + 1)g_n$. Si $n + 1 < n_{lim}$ alors retour à l'étape (2) sinon passage à l'étape (5).

5. Estimation finale : Selection du point de coût minimal parmi les $g_{n_{lim}}$ points finaux.

La complexité en temps de cet algorithme repose presque exclusivement sur l'évaluation du coût des points de $\mathbb{R}^2$ visités. On ne réalise cette évaluation que lors de l'étape de reproduction lors de laquelle de nouveau points sont pris en compte. Les individus survivants d'une génération à l'autre ne sont pas réévalués car demeurent inchangés. La complexité temporelle de l'algorithme dépend donc grandement du taux de reproduction dans la population et du nombre d'itération imposé.

5.2 Détails techniques

Afin de mettre en oeuvre le processus décrit plus haut, nous avons du surmonter quelques difficultés en partie lié à la nature de problème posé.

5.2.1 Etape de reproduction : genèse d'un individu

Lors de la reproduction à l'étape n, on créé un nouveau point à partir de deux individus tirés au sort parmi l'ensemble de la population de cette étape. Notons $A : (x_A, y_A)$ et $B : (x_b, y_B)$ ces deux parents.

La première idée fut de créer le point médian entre A et B. Cependant, la topologie du terrain nous oblige à prendre en compte les bâtiments lors de l'étape de reproduction. En effet, la méthode du point médian présente un problème lorsque le point créé se trouve être à l'intérieur d'un bâtiment.

Pour remédier à cela, nous avons decidé, dans le cas où le point médian n'est pas acceptable (i.e. à l'extérieur d'un bâtiment), de discrétiser le segment AB en un nombre fixe de points. On tire ensuite aléatoirement chacun de ces points jusqu'à trouver un point acceptable. Dans le cas où tous les tirages aléatoires échouent à localiser un point acceptable, on choisit par défaut un point acceptable aléatoirement sur l'ensemble du domaine $[0, 100] \times [0, 100]$. Ce dernier cas ne se produit que lorsque les points A et B sont tous deux très rapprochés d'un bâtiment et est donc une situation assez rare en général.

On choisit de fixer à **20** le nombre de points permettant de discrétiser le segment AB en cas de non-acceptabilité de point médian. Cela correspond dans le pire des cas à une discrétisation d'environ 7m sur la zone de recherche et permet d'assurer un bon compromis entre rapidité d'exécution et fiabilité de la méthode.

5.2.2 Etape de reproduction : mutation d'un individu

Nous avons vu comment créer un nouveau point acceptable à partir de deux individus parents. Cependant, il convient d'ajouter une certaine souplesse à cette méthode permettant de faire légèrement dévier le nouvel individu du segment AB. C'est ce qu'on nomme une mutation.

Ainsi, lorsqu'un point acceptable est identifié pour un nouvel individu, on choisit de lui appliquer un bruit gaussien centré en 0 et de déviation standard fixée à 3m. A titre d'exemple, si le point choisi pour être le nouvel individu est $C : (x_C, y_C)$, alors on tire ϵ_x et ϵ_y suivant une loi $N(0, 3)$ et construisons le point $\big(x_C + \epsilon_x, y_C + \epsilon_y\big)$. On s'assure bien entendu que ce nouveau point est acceptable. Si ce n'est pas le cas, on recommence en tirant de nouvelles déviations ϵ_x et ϵ_y.

Si, après 20 tirages, le processus ne permet toujours pas d'aboutir à un point acceptable, on décide de conserver le point C initial.

5.2.3 Etape d'élimination

Lors de l'élimination à l'étape n, un pourcentage fixe de la population va être supprimé. Cette proportion correspond aux individus apportant la plus mauvaise estimation de la source à l'étape courante (i.e. de coût le plus élevé).

Cependant, comme dans tout algorithme d'optimisation non trivial, il est important d'assurer une part de diversification dans la méthode employée. Ainsi, on définit un taux d'immunité β' permettant la survie de certains des pires indivius mais entraînant aussi la mort de certains des meilleurs. En pratique, après avoir décidé quels individus allaient survivre ou mourir à l'aide du paramètre β, on offre à chaque individu condamné la possibilité de survivre avec une probabilité de β'. De même chaque individu destiné à survivre a une probabilité β' d'être éliminé.

L'ajout d'un unique paramètre β' régissant à la fois la mort et la survie des individus permet à la fois d'assurer un minimum de diversification dans l'algorithme tout en évitant de surcharger le nombre de paramètres devant être fixés pour son éxecution.

5.2.4 Estimation du coût des individus

Comme il a déjà été précisé, l'algorithme ne nécessite d'estimer que le coût des nouveaux individus créés lors de l'étape de reproduction. Dans l'architecture du code, on choisit de conserver à tout instant une liste des individus composant la population triée selon leur coût. Ainsi, créer un nouvel individu revient à insérer un élément dans une liste triée, ce qui peut être réalisé en $log(|liste|)$. Pour notre implémentation, nous nous contentons seulement d'une complexité linéaire du fait de la durée négligeable de la tâche face à l'estimation du coût des nouveaux individus.

5.2.5 Population initiale

La population initiale est formée par tirage aléatoire dindividus sur $[0, 100] \times [0, 100]$ dont on ne conserve que les points acceptable. La taille de cette population initiale peut varier mais **50 individus** semble être un bon compromis entre le temps de simulation et qualité des résultats obtenus.

Lors des tests réalisés sur l'algorithme, nous avons pu constater que l'ensemble des individus (toutes générations confondues) avaient tendance à rester à l'intérieur de l'enveloppe convexe formée par les points de la population initiale. Celà n'a rien de surprenant et résulte simplement de la méthode de reproduction utilisée. En absence de mutation et de choix aléatoire d'enfant, ce résultat devient une propriété exacte de l'algorithme.

Pour encourager l'exploration des recoins de l'espace de recherche, nous avons alors décidé d'imposer les individus situés aux sommets de $[0, 100] \times [0, 100]$ dans la popualtion initiale. L'enveloppe convexe des individus de g_0 se retrouve alors être exactement égale à $[0, 100] \times [0, 100]$. Cette astuce a permi d'améliorer la convergence de l'algorithme vers une solution globale dans le cas où cette dernière est localisée proche des frontière du domaine de recherche.

5.3 Test de l'algorithme avec une fonction de coût artificielle

Nous avons ensuite choisi de tester cet l'algorithme avec une fonction de coût artificielle. Cette dernière n'est pas issue des signaux de pressions mais d'une simple combinaison linéaire de gaussiennes définies sur $\mathbb{R}^2$. La **figure 12** ci-dessus donne un aperçu d'une telle fonction.

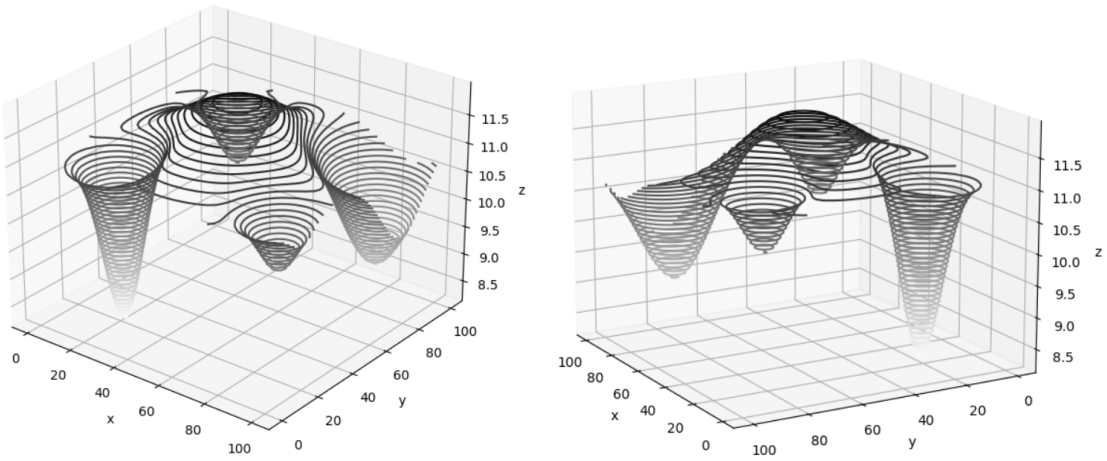


Figure 12: Exemple de fonction de coût artificiellement construite pour tester l'algorithme

La **figure 13** présente les premières étapes du déroulé de l'algorithme génétique appliqué à la minimisation de la fonction présentée en **figure 12**.

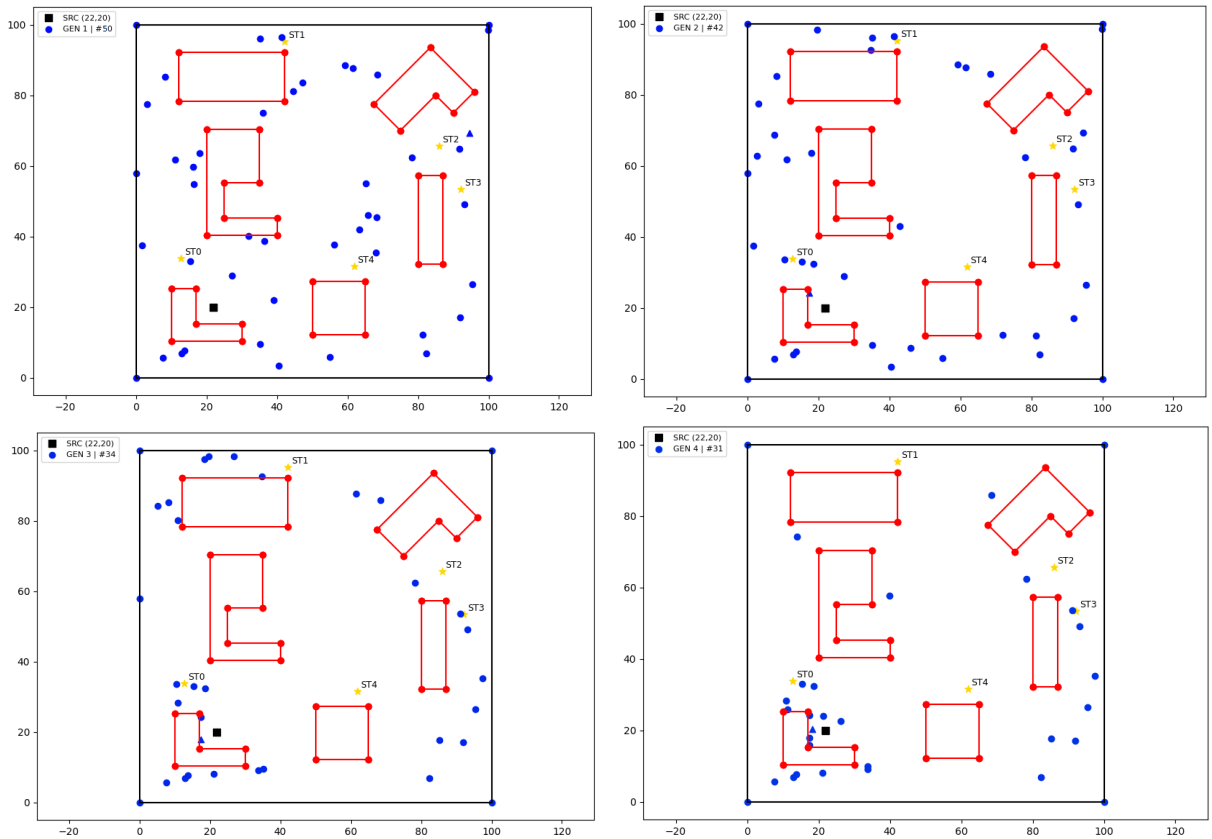


Figure 13: Exemple d'exécution de l'algorithme génétique avec la fonction de coût artificielle de la **figure 12**

Le paramétrage de l'éxécution est rappelée par le tableau ci-dessous :

Paramètres	**Valeurs**
g_0	50
corner points	True
n_{lim}	20
α	0.5
β	0.4
β'	0.05
mutation std	3

Nous avons pu observer une convergence assez rapide de l'algorithme vers le point "source" (ici le minimum global de la fonction de coût). La **figure 14** permet de le constater en visualisant l'évolution de la population toutes les 5 générations.

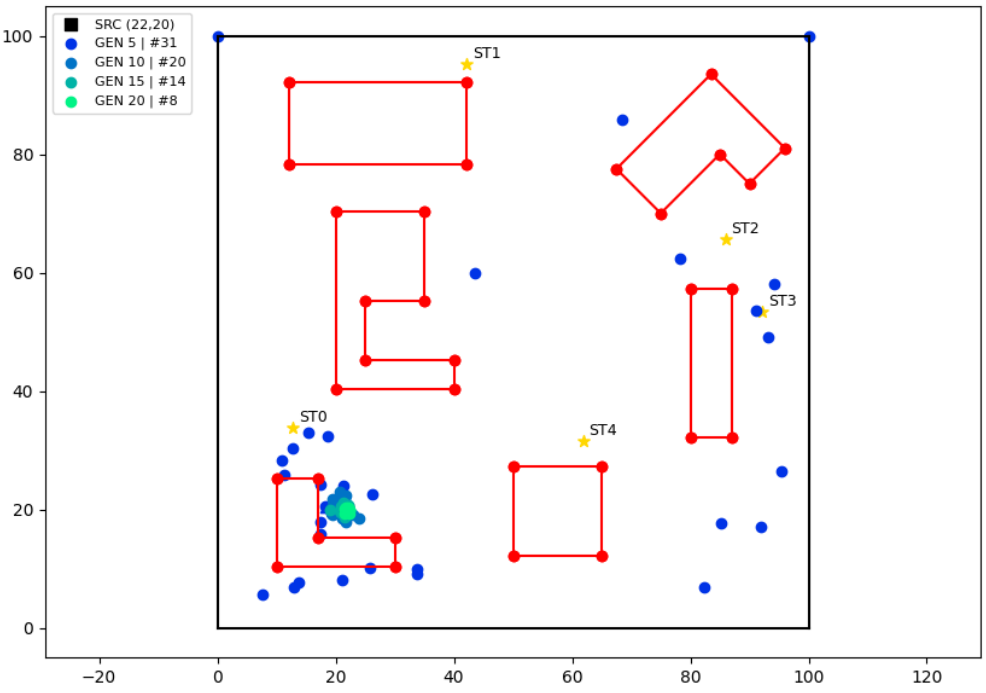


Figure 14: Exemple de voisinage

Cependant, sans modifier les conditions d'exécution, il est possible d'observer une convergence dé l'algorithme vers un des minimum locaux au lieu du minimum global. Cela pose un sérieux problème...

Mais la fonction de coût est bien différentiable et tout ici sur R2...

5.4 Test en situation réelle

ANALYSE DES RESULTATS DE L'ALGO.

5.5 Améliorations proposées

Arrêt de l'algo génétique lorsque le coût passe en-dessous une certaine borne.
Arrêt de l'algo lorsque tous les points sont regroupés dans une boule de rayon $< \rho$.