

TP1 – SIMULATION NUMERIQUE

1. METHODE D'EULER

On souhaite déterminer numériquement le temps de réponse à 5% de la vitesse angulaire d'un moteur d'entraînement d'une hélice, soumis à une entrée de type échelon unitaire. Cette étude va se faire en trois temps : résolution numérique de l'équation différentielle (alors que la solution analytique existe et qu'il est préférable dans tous les cas de l'utiliser), test de convergence de la résolution et détermination du temps de réponse à 5%.

L'étude a conduit à l'équation différentielle suivante, liant l'angle de rotation du moteur $\theta(t)$ à la vitesse de consigne $\theta_{const}(t)$:

$$\frac{1}{\omega_0^2} \cdot \frac{d^2 \theta_m(t)}{dt^2} + \frac{2m}{\omega_0} \cdot \frac{d\theta_m(t)}{dt} + \theta_m(t) = K \cdot \theta_{const}(t)$$

Dans la suite, on s'intéressera uniquement à la réponse indicielle unitaire du système dans les conditions de Heaviside, ce qui revient à prendre $\theta_{const}(t) = 1$ pour $t \geq 0$, avec $\theta_m(0) = 0$ et $\frac{d\theta_m(0)}{dt} = 0$.

La réponse $\theta_m(t)$ recherchée sur l'intervalle $[0; T_{max}]$ sera obtenue par la méthode d'Euler explicite.

Le pas de temps, noté pas, sera choisi constant. L'intervalle de temps discrétisé est alors représenté par le tableau $T = [t_0 = 0, t_1, \dots, t_{N-1} = T_{max}]$.

1. Fonction scalaire, équation différentielle du 1^{er} ordre

La méthode d'Euler explicite, utilisée sur un intervalle $[a, b]$, donne une approximation de $y(b)$, où y est une solution de $y' = F(t, y)$ telle que $y(a) = y_0$, condition initiale de l'équation différentielle.

La méthode consiste à discrétiser l'intervalle de temps $[a, b]$ avec un pas de temps h et d'approximer la dérivée par $y[k+1] = y[k] + h \cdot F(t, y[k])$

Q1. Ecrire la fonction Python `Euler(F, a, b, y0, h)`, qui réalise le schéma numérique d'Euler explicite pour une fonction scalaire à une variable notée $F(t, y)$.

Q2. Tester en renvoyant la valeur de $y(1)$ où y est la solution de l'équation différentielle $y' = y$ telle que $y(0) = 1$. On pourra essayer plusieurs pas, par exemple dans la liste $L = [1/2, 1/10, 1/100, 1/1000, 10^{-6}]$. Vous pouvez utiliser le programme `ordre1.py` et remplacer les parties **A COMPLETER**.*

Q3. Vous pouvez également tracer la courbe et la comparer à la solution réelle.

2. Avec des vecteurs

Rappel sur le procédé de vectorisation :

L'équation différentielle initiale est une équation différentielle du second degré, il est possible d'appliquer deux schémas d'Euler explicite et/ou implicite (sur la vitesse puis sur le déplacement). On peut également poser le

vecteur Y tel que $Y = \begin{pmatrix} \theta_m(t) \\ \frac{d\theta_m(t)}{dt} \end{pmatrix}$ et on pourra noter $\frac{d\theta_m(t)}{dt} = \dot{\theta}_m(t)$

L'équation différentielle peut alors se mettre sous la forme d'une équation différentielle vectorielle du premier ordre : $\dot{Y} = F(t, Y(t))$

Si on peut linéariser $F(t, Y(t))$, alors l'équation prend la forme $\dot{Y} = A \cdot Y(t) + B$. Ce qui est le cas ici !

Q4. Donner la forme de A (matrice 2×2) et B (matrice 2×1)

Pour chaque pas de temps, une valeur approchée Y_i de la solution $Y(t_i)$ de l'équation différentielle est recherchée. L'ensemble des Y_i représente N vecteurs de dimensions 2, qui seront stockés en mémoire sous la forme d'un tableau.

$$SY = \begin{pmatrix} \theta_m(0) & \dot{\theta}_m(0) \\ \theta_m(t_1) & \dot{\theta}_m(t_1) \\ \vdots & \vdots \\ \theta_m(T_{max}) & \dot{\theta}_m(T_{max}) \end{pmatrix}$$

Q5. Écrire une fonction $f1(ti, yi)$, qui prend en arguments la valeur du temps discrétisé ti et la valeur du vecteur $Y_i = Y(t_i)$ et qui retourne la valeur de $F(t, Y(t))$.

Q6. Donner la relation de récurrence qui lie Y_{i+1} à Y_i et à $F(t_i, Y_i)$ en fonction du pas de temps h .

Q7. Écrire une fonction **EulerExplicite(**Yini, h, Tmax, F**) qui prend en arguments **Yini**, un tableau de dimension 2 contenant la condition initiale de $Y(t)$, h le pas de temps, T_{max} l'instant final du calcul, et F la fonction du problème de Cauchy.**

Cette fonction renverra le tableau **SY**. L'appel à cette fonction dans le programme se fera avec la commande **SY = EulerExplicite(Y0, pas, tmax, f1)**.

Q8. Si le pas de temps est divisé par un facteur de 10, comment évolue l'erreur de calcul ?

Q9. Donner la complexité de cette méthode pour T_{max} fixé et indiquer comment évolue le temps de calcul quand le pas de temps est divisé par un facteur 10.

On suppose que la quantité de mémoire nécessaire pour réaliser le calcul se limite au stockage des tableaux **SY** et **T**. Ces éléments sont représentés en mémoire sous la forme de tableaux de flottants en double précision.

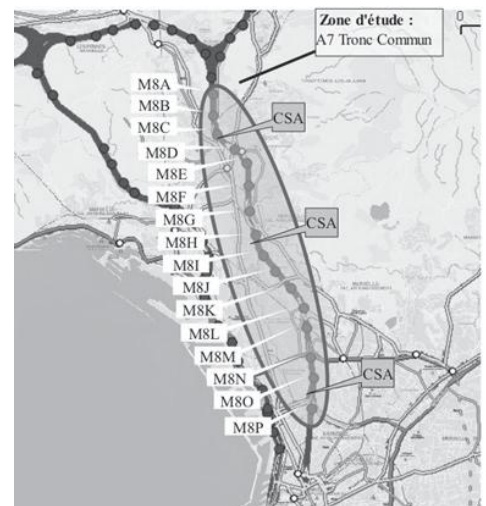
Q10. Déterminer le nombre d'octets nécessaire en mémoire pour réaliser cette simulation numérique avec un nombre de pas de temps $N = 10000$.

2. ETUDE DE LA CAPACITE ET DE LA CONGESTION DE L'AUTOROUTE A7

Il existe plusieurs formes de congestions routières, selon leur cause : la congestion récurrente, la congestion « prévisible » (travaux, manifestations, météo) et la congestion due aux incidents et accidents, par définition imprévisibles. On s'intéresse ici au niveau de congestion récurrente qui peut être défini comme le surplus de demande qui amène la congestion.

Cette étude se focalise sur la congestion routière de l'auto- route A7 en France. La section étudiée ne comporte ni entrée ni sortie. La longueur de l'axe est de 8,5 km et comporte 3 voies sauf au niveau de la dernière station (non étudiée ici). Un seul sens de circulation est étudié.

La cartographie présentée sur la figure 1 donne l'implantation des différentes stations de mesure, de la station M8A à M8P, et des contrôles de sanction automatique (CSA). Ces stations de mesure font partie du système de recueil automatique des données (RAD) présent sur les autoroutes.



Pour réaliser les mesures, la chaussée est équipée de boucles électromagnétiques connectées aux stations qui remontent l'information vers un système central. Ces boucles permettent de compter le nombre de véhicules qui passent sur les routes et, dans le cadre de cette étude, il s'agit de boucles doubles qui permettent aussi d'estimer les vitesses.

La connaissance et le suivi du niveau de congestion récurrente permettent ensuite d'envisager des actions de régulation et d'aménagement de voirie. La simulation numérique est alors employée pour étudier différentes solutions d'aménagement ou proposer des solutions de régulation dynamique du trafic.

L'objectif est de mettre en place une simulation numérique adaptée au tronçon étudié à partir d'une première simulation du trafic routier par la mécanique des fluides

Il est rappelé ici que dans toute la suite de l'étude, l'autoroute est assimilée à une seule voie. Le modèle continu revient à négliger le caractère discret de la matière. Pour le modèle routier, cela revient donc à regarder l'évolution du trafic sur des distances grandes devant la taille des véhicules, notée L_0 . On appelle $c(t,x)$, en véhicules par mètre, la concentration de véhicules par unité de longueur de route à l'instant t et à la position x . Sur une longueur L_0 , il y a, au plus, un seul véhicule, soit $\leq c_{max} = \frac{1}{L_0}$.

On appelle $q(t,x)$, en véhicules par seconde, le débit de véhicules, c'est-à-dire le nombre de véhicules par unité de temps traversant la section de la route située à la position x . La vitesse $v(t,x)$, en mètres par seconde, ne représente pas la vitesse de chacun des véhicules mais la vitesse moyenne du trafic à la position x . On considère que les véhicules se déplacent selon l'axe x dans le sens des x croissants.

On rappelle la relation $q(t,x) = c(t,x) \cdot v(t,x)$. Désormais, q , v et c représentent les grandeurs simulées et non plus des données expérimentales. On travaille dans la suite avec les unités du système international.

En considérant une portion d'autoroute dx pendant une durée dt et en supposant qu'il n'y a ni perte, ni création de véhicule, il est possible de montrer que $q(t,x)$ et $c(t,x)$ vérifient l'équation aux dérivées partielles suivante :

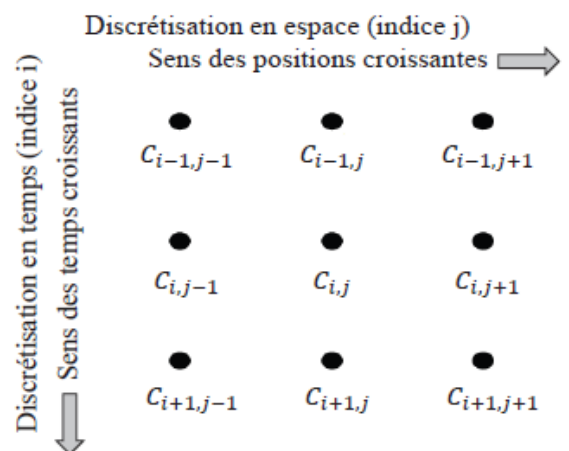
$$\frac{\partial q(t,x)}{\partial x} + \frac{\partial c(t,x)}{\partial t} = 0 \quad (1)$$

Pour comprendre comment évoluent la concentration, la vitesse moyenne ou le débit de véhicules au cours du temps le long de l'autoroute, il convient donc de résoudre cette équation aux dérivées partielles à partir de la situation initiale.

2.1 Discrétisation

Afin de résoudre numériquement cette équation aux dérivées partielles, nous avons besoin de la discrétiser. On choisit les paramètres suivants :

- longueur de l'autoroute : L_a
- durée de simulation : T_{emps}
- pas d'espace (en mètres) : dx
- pas de temps (en secondes) : dt



Q1. Soit C le tableau de valeurs contenant les concentrations en tous les points x et à tous les instants t discrétisés, comme représenté sur la figure ci-dessus. L'approximation numérique de la concentration au temps t_i et à la position x_j sera notée $C_{i,j}$ avec i désignant l'indice de temps et j désignant l'indice d'espace. Quelles sont les dimensions de C ?

2.2 Un modèle de diagramme fondamental

L'équation (1) possède deux inconnues. Il faut donc ajouter une deuxième équation pour pouvoir la résoudre. On propose tout d'abord de relier la vitesse et la concentration par le modèle de Greenshield établi à partir des analyses suivantes :

- lorsque la concentration en véhicules tend vers 0, les conducteurs peuvent rouler à la vitesse maximale autorisée, v_{max} en mètres par seconde ;
- lorsque les véhicules sont pare-choc contre pare-choc, la concentration est égale à c_{max} en véhicules par mètre : ils n'avancent plus.

Une relation linéaire entre vitesse et concentration est choisie dans le modèle de Greenshield, qui est ainsi défini par la relation suivante :

$$v(t, x) = v_{\max} \left(1 - \frac{c(t, x)}{c_{\max}} \right) \quad (2)$$

On souhaite concevoir une fonction `diagramme` permettant de réaliser le tracé du **diagramme fondamental** (**page 3**) pour un instant donné t_i (soit pour une ligne de C). Cette fonction fait appel à une fonction `debit` permettant de calculer les valeurs de débit à un instant t_i en utilisant la relation de Greenshield (2). Ces valeurs sont stockées dans un vecteur Q . Le tracé du diagramme fondamental est réalisé et le tableau Q est retourné.

Q2. Ecrire une fonction `debit(v_max, c_max, C_ligne)` qui prend en arguments la vitesse maximale ($v_{\max}$), la concentration maximale ($c_{\max}$) et un tableau contenant les concentrations à un instant donné (soit les éléments d'une ligne du tableau C) nommé ici C_ligne et qui renvoie un tableau de valeurs contenant les débits (en véhicules par seconde) aux différentes positions à ce même instant.

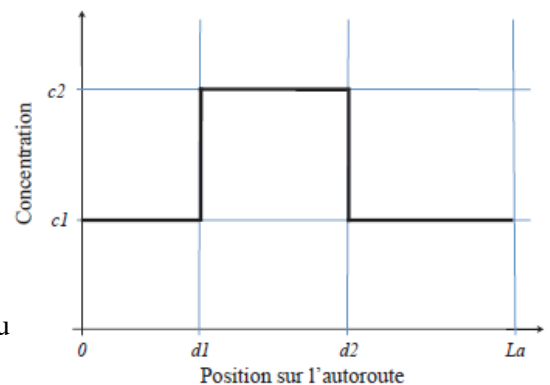
Q3. Spécifier les arguments d'entrée (et leur type) de la fonction `diagramme`. L'écriture du code de la fonction n'est pas demandée. Préciser les unités des différents termes. Tracer l'allure du diagramme fondamental obtenu. L'allure du diagramme dépend-elle du temps t_i auquel on se place (soit du choix de la ligne de C)?

2.3 Résolution de l'équation

Situation initiale

On considère une situation de départ ($t = 0$), comme indiqué sur la figure ci-contre. On se place dans une configuration où l'on a un profil de concentration dont on veut étudier l'évolution. La concentration $c1$ la plus faible passe à $c2$, plus forte, à la distance $d1$ et revient à $c1$ en $d2$.

Chacune des distances sera discrétisée. L'approximation numérique entière de la distance par le pas.



Q4. Concevoir une fonction `C_depart` qui permet d'initialiser la première ligne du tableau C (correspondant à $t = 0$). L'écriture du code correspondant n'est pas demandée ; en revanche, on précisera toutes les valeurs de j pour lesquelles $C_{0,j}$ sera égale à $c1$ et toutes les valeurs pour lesquelles $C_{0,j}$ sera égale à $c2$. L'entête ou spécification de la fonction devra être précisé(e) (et comporter les arguments d'entrée et leur type), ainsi que le résultat renvoyé par la fonction.

Résolution

Le tableau C contient des zéros, exceptée la première ligne qui a été remplie de valeurs grâce à la mise en œuvre de la fonction `C_depart`. On souhaite à présent écrire la fonction `resolution(C, dt, dx, c_max, v_max)` permettant de résoudre l'équation et de remplir complètement le tableau C .

Connaissant pour tout indice j les valeurs de $C_{i,j}$, on cherche à déterminer $C_{i+1,j}$.

Dans le schéma d'Euler « avant », la dérivée d'une fonction f par rapport à la variable x , au point x_j ,

$\frac{df}{dx}(x_j)$, est approximée (en utilisant ce point et le point situé « devant » lui) par $\frac{f_{j+1} - f_j}{dx}$.

Q est un vecteur contenant les valeurs de débits Q_j aux différentes positions x_j et à l'instant t_i (l'approximation du débit au temps t_i et à la position x_j sera donc notée Q_j). À chaque instant t_i , Q devra être recalculé.

Q5. A partir de l'équation (1) et en utilisant des schémas d'Euler « avant » pour l'écriture des dérivées, donner la relation de récurrence donnant $C_{i+1,j}$ en fonction de $C_{i,j}$, Q_{j+1} , Q_j , dx et dt .

Pour que le nombre de voitures soit constant sur la longueur de la route, il faut se fixer des conditions aux limites périodiques. Ainsi, quand un véhicule arrive en bout d'autoroute, il est replacé au début de celle-ci. On considère donc que le véhicule situé en $x = La$, se déplaçant vers la droite, a pour voisin de droite le véhicule situé en $x = 0$.

Q6. L'initialisation a été effectuée avec la fonction `C_depart(dx, d1, d2, c1, c2, C)`. Ecrire une fonction `resolution(C, dt, dx, c_max, v_max)` qui prend en arguments le tableau C , les pas dt et dx , la concentration maximale $c_{\max}$ et la valeur de la vitesse maximale $v_{\max}$ et qui

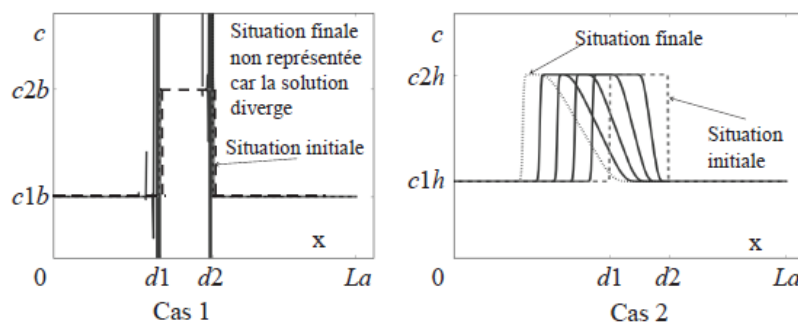
renvoie le tableau C rempli au cours de la résolution. On pourra faire appel à la fonction $\text{debit}(v_max, c_max, C \text{ ligne})$ définie à la question Q2.

2.4 Etude des solutions trouvées et modification du schéma

On étudie deux situations de départ basées sur le même profil que celui proposé à la situation initiale (figure ci-dessus). Dans le cas 1, on choisit $c1$ et $c2$, respectivement notées $c1b$ et $c2b$, correspondant à des concentrations faibles. On peut montrer que, pour ces concentrations, le créneau se déplace vers la droite (dans le sens des x croissants) au cours du temps. Cette démonstration n'est pas demandée.

Dans le cas 2, on choisit $c1$ et $c2$, respectivement notées $c1h$ et $c2h$, correspondant à des concentrations fortes. On peut montrer que, pour ces concentrations, le créneau se déplace vers la gauche (dans le sens des x décroissants) au cours du temps. Cette démonstration n'est pas demandée.

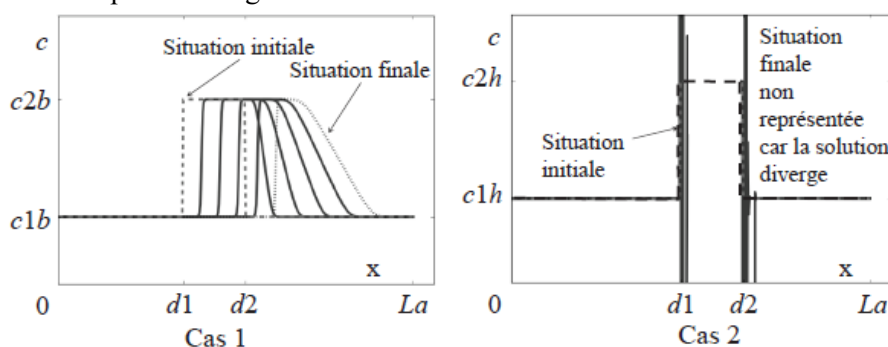
On applique la fonction `resolution` à ces deux situations de départ et on représente la concentration en fonction de la position à différents instants (figure suivante). La situation initiale est en traits interrompus (- -), les situations intermédiaires en traits continus (-) et la situation finale en pointillés (:).



On remarque qu'avec la situation de départ du cas 1, la solution diverge. Avec la situation de départ du cas 2, le créneau initial semble se déplacer vers la gauche.

Une modification est alors apportée à la fonction `resolution`. Dans la discrétisation en espace (c'est-à-dire lors de l'écriture des dérivées par rapport à la variable d'espace), le schéma d'Euler « avant » en espace est remplacé par un schéma d'Euler « arrière » en espace. Dans ce schéma, la dérivée d'une fonction f par rapport à la variable x au point x_j , $\frac{df}{dx}(x_j)$ est approximée (en utilisant ce point et le point situé « derrière » lui) par $\frac{f_j - f_{j-1}}{dx}$.

On obtient alors les résultats présentés figure suivante.



Cette fois-ci, avec la situation de départ du cas 1, le créneau initial se déplace vers la droite et avec la situation de départ du cas 2, la solution diverge.

Q7. On se place à l'itération $i + 1$; les calculs des itérations précédentes ont déjà été réalisés. Le calcul des termes de Q (vecteur des débits à l'instant t_i) est fait par la fonction `debit`, Q_j étant déterminé à partir de la valeur de $C_{i,j}$.

Sur la grille de discrétisation donnée figure page 4 (qui sera reproduite sur la copie), tracer des flèches partant des points déjà calculés aux itérations précédentes et allant vers le point à calculer $C_{i+1,j}$ (au pas d'espace j et à l'itération $i + 1$) dans le cas du schéma d'Euler « avant » pour la discrétisation en espace. Procéder de même dans le cas du schéma d'Euler « arrière » pour la discrétisation en espace.

Q8. En déduire un argument permettant de choisir le schéma d'Euler adapté à la situation de départ.

On cherche maintenant un schéma fonctionnel pour les deux situations. On propose celui de Lax- Friedrichs qui :

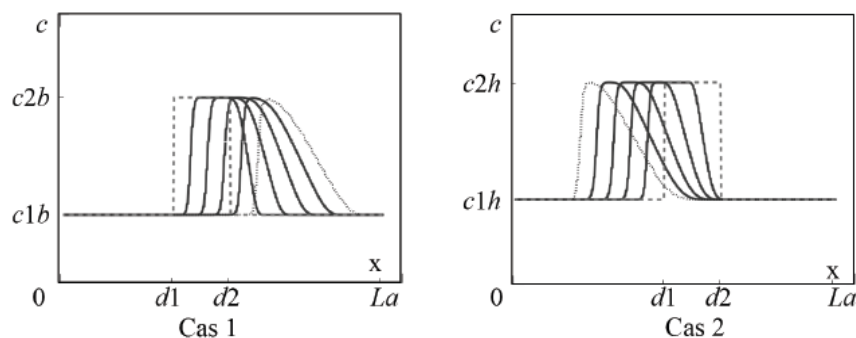
- utilise un schéma centré pour l'approximation des dérivées par rapport à la variable d'espace.

Dans ce schéma, la dérivée d'une fonction f par rapport à la variable x au point x_j , $\frac{df}{dx}(x_j)$ approximée (à partir du point précédent et du point suivant) par $\frac{f_{j+1}-f_{j-1}}{2dx}$;

- approxime les dérivées par rapport à la variable de temps en remplaçant la valeur $C_{i,j}$ par la moyenne de la valeur prise au point précédent $C_{i,j-1}$ et de la valeur prise au point suivant $C_{i,j+1}$.

Q9. Proposer les modifications de la fonction `resolution` (définie à la question Q6) nécessaires pour utiliser le schéma de Lax-Friedrichs.

La mise en œuvre de la fonction `resolution` permet, ensuite, d'effectuer le tracé des solutions obtenues dont le résultat est donné figure ci-dessous. **Les instructions permettant de réaliser le tracé ne sont pas demandées.**



2.5 Amélioration du programme : retour sur le choix du diagramme fondamental

Afin de s'assurer de la stabilité de la solution trouvée, il convient, une fois le schéma d'approximation déterminé, de définir les valeurs des paramètres tels que les pas de temps et d'espace. On suppose ici que ce travail a été réalisé. Cependant, cela ne signifie pas que l'on peut simuler fidèlement des phénomènes réels. En effet, le diagramme fondamental utilisé et tracé à la question Q7 est assez différent du nuage de points expérimentaux.

La nouvelle approximation choisie pour obtenir les paramètres caractéristiques du diagramme fondamental q en fonction de c est une régression d'ordre 3 : $q = a_3 c^3 + a_2 c^2 + a_1 c + a_0$, a_i étant les constantes d'ajustement de la courbe sur le nuage de n points. La fonction `regression(q_exp, c_exp)`, qui prend en arguments les vecteurs `q_exp` et `c_exp` obtenus expérimentalement (ayant servi à tracer le nuage de points de la figure page 3) et qui renvoie les coefficients a_0, a_1, a_2, a_3 , a été réalisée (on ne demande pas d'écrire cette fonction).

Q10. Expliquer en quelques phrases ce qu'il faut modifier dans la fonction `resolution` définie à la question Q6 pour résoudre l'équation (1) en prenant en compte ce nouveau diagramme fondamental basé sur l'expérimentation.

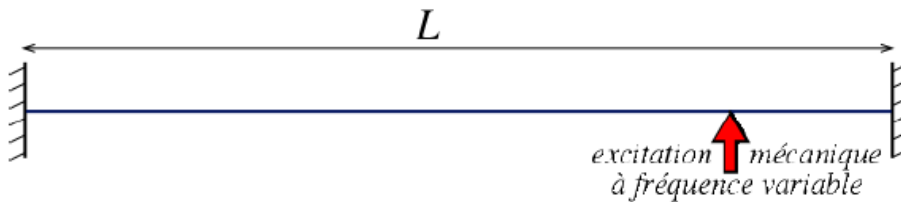
La simulation que nous avons mise en place permet ainsi de déterminer les caractéristiques d'évolution d'un embouteillage. On peut notamment en déduire la vitesse de propagation de l'embouteillage, ou sa vitesse de résorption. Cependant, on aimerait à présent mettre en place une simulation permettant de modéliser les comportements individuels des conducteurs.

3. EXCITATION D'UNE CORDE DE PIANO

On étudie le son produit par une corde de piano.

Après une perturbation initiale, une corde tendue, de longueur L fixée sur ses deux extrémités se met à vibrer. La tension et la densité de la corde sont notées σ et ρ .





Ce mouvement vibratoire est décrit par l'équation suivante :

$$\frac{\sigma}{\rho} \cdot \frac{d^2 y(x)}{dx^2} + \omega^2 y(x) = 0$$

Les pulsations ω_k sont les valeurs de ω correspondant aux solutions non nulles du problème. On montre que la solution générale $y(x)$ et la plus basse fréquence propre ω_0 sont :

$$y(x) = A e^{i\omega \sqrt{\frac{\rho}{\sigma}} x} + B e^{-i\omega \sqrt{\frac{\rho}{\sigma}} x}$$

Q1. Mettre cette équation sous forme matricielle suivante $([M] + \omega^2 [I])(y)$ en utilisant le schéma des différences finies centrées (dérivée partielle d'ordre 2) avec un pas d'espace $h = \frac{L}{4}$

Q2. Expliquer la résolution

4. CONDUCTION STATIONNAIRE 2D

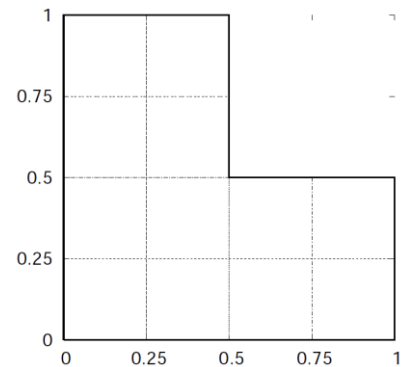
On s'intéresse à connaître la conduction dans un domaine en forme de L. (afin de comprendre la conduction du son dans une des parties du piano)

Les lois de la physique nous ont permis d'obtenir l'équation différentielle suivante :

$$\frac{\partial^2 u(x, y)}{\partial x^2} + \frac{\partial^2 u(x, y)}{\partial y^2} = 0$$

Les conditions aux limites sont les suivantes :

- sur la ligne $x = 0$: $u(0, y) = y(1 - y)$
- sur la ligne $y = 0$: $u(x, 0) = x(1 - x)$
- sur le reste de la frontière : $u = 0$.



Q3. Discrétiser l'équation de Laplace par les différences finies centrées en utilisant un pas constant $h = 0.25$ en x et en y .

Q4. En utilisant les conditions aux limites et la symétrie du problème, expliciter le système linéaire (3×3) qui régit ce problème. On se ramènera à un problème du type $Ax=b$

Q5. Considérer la décomposition de A amène à la résolution de 2 systèmes linéaires, qui sont moins coûteux à résoudre que le système initial. Décrire ces 2 phases de résolution.

Q6. Programmer la résolution de $Ax = b$ par la méthode du pivot de Gauss en descente lorsque A est une matrice triangulaire inférieure.

On utilise dans les cas général la méthode du pivot de Gauss pour résoudre ce système.

Cependant, il est plus efficace d'utiliser des librairies que de reprogrammer soi-même des routines existantes. D'un point de vue pratique, l'usage de librairie permet d'augmenter la lisibilité du code. De plus les librairies qui sont programmées de façon plus efficace sont souvent plus performantes comme on va le voir ici.

Le sous-module *linalg de numpy* permet la résolution de systèmes linéaires. En remplaçant toute la partie résolution par "`x=np.linalg.solve(A,b)`".

On peut aussi illustrer le gain que l'on peut avoir en utilisant à la fois des librairies, mais aussi le stockage *sparse* avec *scipy.sparse.linalg.spsolve* (cependant il faut bien noter qu'une différence d'optimisation dans la programmation des librairies utilisées pourrait aussi expliquer quelques écarts). Il s'agit de stocker les matrices creuse M de taille m.n sous la forme de trois tableaux unidimensionnels.

Si on cherche l'algorithme de résolution utilisé dans cette librairie il est appelé *SuperLU 4.0*, celui-ci est basé sur une *décomposition LU* apparenté à la méthode du pivot de Gauss.

On importe la bibliothèque pour avoir le solveur *dsolve* mais aussi la possibilité de transformer automatiquement une matrice pleine en creuse.

Ici le stockage est au format CSR qui est identique au format *Yale Sparse Matrix*, avec un stockage un peu différent des indices de colonne.

```
#pour avoir des outils
import scipy . sparse as sps
#importe le solveur
from scipy . sparse . linalg . dsolve import linsolve

#Pour transformer la matrice en une matrice sparse (format csr)
Ab=sps. lil_matrix (A)
Ab = Ab. tocsr ()
```

*Q7. Dans le cas général d'une matrice (3*3), écrire un programme effectuant la décomposition $A=LU$*