Hiver 2018

Devoir

MAT-2910 : Analyse numérique pour ingénieurs

Consignes :

- Vous devez utiliser le logiciel MATLAB pour programmer les différentes méthodes.
- Donner tous les développements et calculs. Pour recevoir des points, toute réponse doit être convenablement justifiée.
- Pour tous les programmes MATLAB qu'on demande de coder, l'énoncé indiquera leur nom et les paramètres d'entrée et de sortie sous la forme du canevas suivant :

```
[sortie1, sortie2,...]=nomfonction(param1,param2,param3...)
```

Il faudra fournir pour chaque fonction un fichier MATLAB nommé `nomfonction.m` dans lequel sera codée la fonction. Les paramètres d'entrée et de sortie doivent correspondre à ceux du canevas fourni.

- Des annexes sont disponibles sur le site du cours. Ils sont uniquement là pour vous introduire un sujet, pour une compréhension plus approfondie, on fera preuve d'autonomie en regardant la littérature sur le sujet en question.
- Vous ne pouvez pas obtenir d'aide du CDA pour ce devoir.
- Des points seront alloués pour la qualité de la présentation et du français. On portera attention à la lisibilité des figures (qualité, légendes, titres,...).
- Votre rapport, au format `.pdf`, ainsi que les fichiers MATLAB (`.m`) doivent être déposés dans la boîte de dépôt sur le site du cours.
- Le rapport ne contiendra aucun programme ou script MATLAB en son corps. Vous devrez les fournir en pièces jointes et y référer dans le texte.
- Support : 4 séances d'information sont prévues. Pour les lieux et heures voir sur le site du cours. Pour toute autre question, vous pouvez (on vous conseille fortement) contacter les professeurs ou poser des questions sur le forum.
- Équipe d'au plus deux personnes (sans restriction sur les sections). Vous avez jusqu'au 28 mars pour former les équipes.
- **Date de remise :** mardi 17 avril.

Description du problème :

Dans tout le problème, on considère une corde en tension. On suppose la tension T_0 de la corde indépendante du temps et la densité linéique ρ_0 uniforme et constante (c'est à dire indépendante du temps et de l'espace). Lorsque l'on modélise le déplacement d'une telle corde, d'Alembert a montré que le déplacement u vérifiait l'équation des ondes :

$$\frac{\partial^2 u(x,t)}{\partial t^2} - c\frac{\partial^2 u(x,t)}{\partial x^2} = 0, \tag{1}$$

avec $c = \frac{T_0}{\rho_0}$. On note que c, bien que désignant habituellement une vitesse, est ici homogène au carré d'une vitesse (on pourra le vérifier en faisant une analyse dimensionnelle de l'égalité). La variable d'espace est la variable x et la variable de temps est t. La corde est de longueur finie L et occupe l'espace $[0, L]$. On considère le mouvement de la corde pendant un certain intervalle de temps $[0, T]$. On précise aussi que $\frac{\partial^2 u}{\partial t^2}$ est la dérivée partielle seconde de u par rapport à t et de même $\frac{\partial^2 u}{\partial x^2}$ est la dérivée partielle seconde de u par rapport à x. Notez que (1) ne suffit pas pour trouver une unique solution. Il faut mettre **des conditions aux limites** (c'est à dire en $x = 0$ et $x = L$) et **des conditions initiales** (c'est à dire en $t = 0$, donner $u(x, 0)$ et $\frac{\partial u}{\partial t}(x, 0)$).

Le but de l'étude est de déterminer s'il est possible, en prenant une corde quelconque, tendue entre deux points P_1 et P_2, dont on ne connaît pas les paramètres T_0 et/ou ρ_0, de retrouver la vitesse de l'onde en observant le déplacement en un point de la corde, lorsque l'on agite la corde à un bout. Pour cela nous devons dans un premier temps être en mesure de résoudre numériquement l'équation des ondes. Une fois cette étape complétée on pourra s'attaquer au problème de retrouver la vitesse c (on nommera ce problème, problème inverse). La première chose que l'on va faire est de trouver des solutions particulières de l'équation des ondes appelées modes propres. Ceux-ci nous permettront de tester la validité de la résolution de l'équation des ondes.

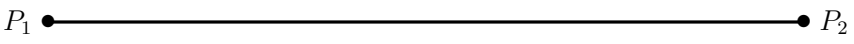

FIGURE 1: Une corde au repos entre les points P_1 et P_2

Partie 1 (théorique) : Modes propres d'une corde

Exercice 1 :

Dans cette partie on considère le système suivant :

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2} - c\dfrac{\partial^2 u}{\partial x^2} = 0, \\ \forall t \in [0,T],\ u(0,t) = u(L,t) = 0, \end{cases} \tag{2}$$

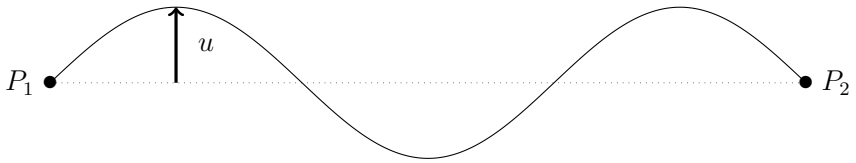

FIGURE 2: Une corde en mouvement entre les points P_1 et P_2.

Sur la figure 2, on a tracé un déplacement à un temps t fixé. Le déplacement tracé vérifie les conditions dites aux limites (c'est à dire au bord de la corde). En effet en P_1 et P_2 le déplacement est nul ($u(0,t) = u(L,t) = 0$) : la corde est fixée en ces points.

On ne prescrit pour l'instant pas de conditions initiales. On cherche des solutions sous la forme :

$$u(x,t) = (\alpha\cos(\omega_t t) + \beta\sin(\omega_t t))\sin(\omega_x x), \tag{3}$$

avec α et β dans $\mathbb{R}$ deux constantes. ω_t et ω_x sont appelés pulsations, et ne dépendent pas de x et t.

a) Calculer les dérivées partielles secondes de u par rapport à t et x.

b) Déterminer une condition nécessaire sur ω_t, ω_x et c pour que u, donné par (3) soit solution. Cela suffit-il pour que u soit solution ?

c) Donner les valeurs de ω_x telles que u vérifie les conditions aux limites $u(0,t) = u(L,t) = 0$. Justifier soigneusement, vous devez en trouver une infinité.

d) Finalement donner les fonctions u en fonction de α, β, L, c, x et t.
Remarque : Ces solutions sont les **modes propres** de l'équation. Une solution générale du système (2) peut s'écrire comme la somme (infinie) de ces solutions.

e) On appelle «nœud» un point de la corde dont le déplacement $u(x,t)$ est nul pour tout temps t. En plus des deux points situés à chaque extrémité de la corde, déterminer le nombre de nœuds de ces modes propres.

f) Comme on l'a fait remarquer en introduction, pour compléter (2), il faut ajouter des conditions initiales ($u(x,0)$ et $\frac{\partial u}{\partial t}(x,0)$). Pour chacun des modes propres u trouvés, donner les conditions initiales associées.

Partie 2 : Résolution numérique de l'équation des ondes - Problème direct

Dans cette partie on veut résoudre un système du type :

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2} - c\dfrac{\partial^2 u}{\partial x^2} = 0, \\ \forall t \in [0,T],\ u(0,t) = f(t) \text{ et } u(L,t) = g(t), \\ \forall x \in [0,L],\ u(x,0) = u_0(x) \text{ et } \dfrac{\partial u}{\partial t}(x,0) = u_1(x) \end{cases} \tag{4}$$

Les fonctions u_0 et u_1 sont les conditions initiales, ce sont des données.

Il existe plusieurs façons de résoudre numériquement l'équation des ondes, on choisit ici d'utiliser une méthode dite de différences finies. La méthode s'appuie sur une discrétisation de l'espace et du temps. C'est à dire que l'on va séparer les intervalles de temps et d'espace en petits sous-intervalles. Cela nous donne une suite (x_j) telle que :

$$\begin{aligned} &x_0 = 0, \\ &x_n = x_0 + n\Delta x \text{ pour } 0 \leq n \leq N_x - 1 \end{aligned}$$

où Δx est un paramètre qu'on appelle "pas de discrétisation en espace" et N_x est le nombre de points de la discrétisation en espace.

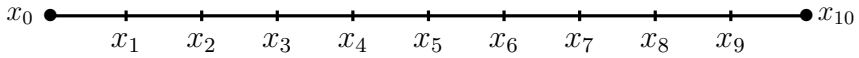

FIGURE 3: Discrétisation en espace de la corde au repos avec $N_x = 11$.

De même on définit une suite (t_n) telle que :

$$\begin{aligned} &t_0 = 0, \\ &t_n = t_0 + n\Delta t \end{aligned}$$

où Δt est un paramètre qu'on appelle "pas de discrétisation en temps" et N_t est le nombre de points de la discrétisation en temps.

L'objectif est de calculer des approximations de u aux points (x_j, t_n). On notera l'approximation de u au point (x_j, t_n) : u_j^n et numériquement le code produira une suite à double indice (u_j^n). Les figures 4 et 5 montre pour $N_x = 11$ deux pas de temps successifs. L'algorithme de résolution fournit uniquement les valeurs des u_j^n (en rouge) à chaque pas de temps.

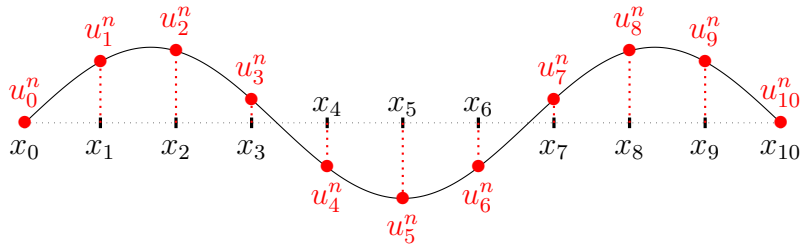

FIGURE 4: Discrétisation d'une corde en mouvement entre les points P_1 et P_2 au temps t_n.

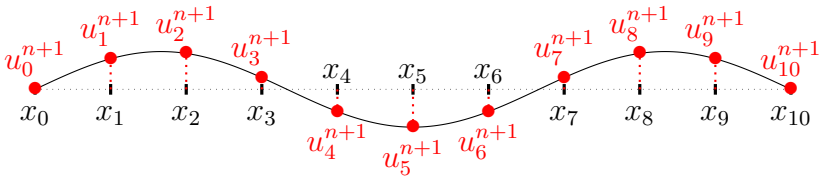

FIGURE 5: Discrétisation d'une corde en mouvement entre les points P_1 et P_2 au temps t_{n+1}.

Pour résoudre (4), on approche la dérivée partielle seconde de u par rapport à x à t fixé par :

$$\frac{\partial u^2}{\partial x^2}(x,t) \approx \frac{u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)}{\Delta x^2}. \tag{5}$$

On peut faire de même pour la dérivée en temps. La formule (5) peut s'écrire en fonction des termes de la suite (u_j^n) si on prend $t = t_n$:

$$\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2}.$$

Exercice 1 :

a) Déterminer l'ordre d'approximation de la dérivée seconde en utilisant la formule (5) (on rappelle que cela correspond au degré du terme d'erreur). On pourra pour cela effectuer deux développements de Taylor à l'ordre 2 de u en x (à t constant).

Pour résoudre l'équation des ondes, on propose de discrétiser l'équation des ondes par :

$$\begin{aligned} &\frac{u_j^{n+1} - 2u_j^n + u_j^{n-1}}{\Delta t^2} + \theta c \frac{-u_{j+1}^{n+1} - u_{j-1}^{n+1} + 2u_j^{n+1}}{\Delta x^2} \\ &+ (1-2\theta)c\frac{-u_{j+1}^n - u_{j-1}^n + 2u_j^n}{\Delta x^2} + \theta c \frac{-u_{j+1}^{n-1} - u_{j-1}^{n-1} + 2u_j^{n-1}}{\Delta x^2} = 0 \end{aligned} \tag{6}$$

pour j de 1 à $N_x - 2$ et n de 1 à $N_t - 1$. L'idée étant que connaissant les valeurs aux deux pas de temps précédents (n et $n-1$) on est capable de calculer le déplacement au pas de temps suivant.

b) Pour chacune des fractions dans (6), déterminer la dérivée partielle approchée (et le point où elle est calculée).

Le schéma (6) n'est pas complet puisqu'il ne permet pas de calculer u_0^{n+1} et $u_{N_x-1}^{n+1}$. On introduit le vecteur $U^n = (u_0^n, \cdots, u_{N_x-1}^n)$. Le vecteur U^n est le vecteur des valeurs de l'approximation de $u(x, t_n)$ aux points x_j. On veut trouver des matrices A, B_0 et B_1 telles que le système (6) s'écrive sous la forme :

$$AU^{n+1} = B_0 U^n + B_1 U^{n-1}.$$

Les matrices (tridiagonales) que l'on trouve sont :

$$A = \begin{bmatrix} 1+2\theta C & -\theta C & 0 & \dots & 0 \\ -\theta C & 1+2\theta C & -\theta C & \ddots & 0 \\ 0 & -\theta C & 1+2\theta C & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\theta C \\ 0 & 0 & 0 & -\theta C & 1+2\theta C \end{bmatrix} \tag{7}$$

$$B_0 = \begin{bmatrix} 2-2(1-2\theta)C & (1-2\theta)C & 0 & \dots & 0 \\ (1-2\theta)C & 2-2(1-2\theta)C & (1-2\theta)C & \ddots & 0 \\ 0 & (1-2\theta)C & 2-2(1-2\theta)C & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & (1-2\theta)C \\ 0 & 0 & 0 & (1-2\theta)C & 2-2(1-2\theta)C \end{bmatrix} \tag{8}$$

$$B_1 = \begin{bmatrix} -1-2\theta C & \theta C & 0 & \dots & 0 \\ \theta C & -1-2\theta C & \theta C & \ddots & 0 \\ 0 & \theta C & -1-2\theta C & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \theta C \\ 0 & 0 & 0 & \theta C & -1-2\theta C \end{bmatrix} \tag{9}$$

en notant $C = c\frac{\Delta t^2}{\Delta x^2}$. La définition de ces matrices ne permet pas de respecter les conditions aux limites $u(0, t_n) = f(t_n)$ et $u(L, t_n) = g(t_n)$ pour tout n. Pour cela, on doit modifier la matrice A et le vecteur $b^n = B_0 U^n + B_1 U^{n-1}$. On modifie A de la

manière suivante

$$A(1,1) = 1 \text{ et } A(1,k) = 0, \ \forall\ k \geq 2$$
$$A(N_x, N_x) = 1 \text{ et } A(N_x, k) = 0, \ \forall\ k < N_x$$

et l'on modifie b^n de la façon suivante (ici b^n est vu comme un vecteur au sens MATLAB, les indices commençant donc à 1) :

$$b^n(1) = f(t_{n+1}),$$
$$b^n(N_x) = g(t_{n+1}).$$

Le problème à résoudre dans votre code devient :

$$AU^{n+1} = b^n.$$

avec la matrice A changée.

c) Justifier que de cette façon les conditions aux limites de (4) seront bien vérifiées au temps t_{n+1}.

d) Étant donné le grand nombre de 0 qui les compose, on peut les qualifier de matrices creuses. Quel est l'intérêt numérique des matrices creuses ?

e) À quoi est égal A si $\theta = 0$? Quel est l'avantage du cas $\theta = 0$ par rapport au cas où $\theta \neq 0$?

Remarque : Lorsque $\theta = 0$ on parle de méthode explicite tandis que $\theta \neq 0$ on parle de méthodes implicites.

Exercice 2 :

Dans cette partie on veut résoudre numériquement l'équation des ondes et notamment retrouver certains modes propres trouvés théoriquement. Numériquement, il est impossible d'imposer la condition $\frac{\partial u}{\partial t}(x,0) = u_1(x)$, on impose plutôt la valeur de u au temps Δt, c'est-à-dire $u(x, \Delta t) = \tilde{u}_1(x)$. On rappelle les modes propres trouvés en première partie (ici $\beta = 0$ et $\alpha = 1$) :

$$u(x,t) = \cos(\omega_t t)\sin(\omega_x x).$$

On classe les modes propres (3) dans l'ordre croissant de ω_t. On prendra $L = 1$ et $g(t) = 0$ (déplacement nul en $x = L$). On pourra créer un script `script_res_equation_onde.m` pour définir les paramètres et tester nos fonctions. On déclarera les paramètres T, α et ω_t en variable global dans le script et on pourra les utiliser dans la fonction.

a) Implémenter une fonction qui résout l'équation des ondes pour une condition aux limites

f en $x = 0$ quelconque :

```
[u,erreur]=resout_equation_onde(c,Nt,Nx,theta,f,u0,u1tilde)
```

La sortie `u` représente le déplacement au cours du temps. `u` sera la matrice u_j^n où j sont les lignes et n les colonnes (le temps est en colonne et l'espace en ligne). `f` est une fonction qui est égale à $f(t)$, la condition aux limites au point $x = 0$. `u0` et `u1tilde` sont des fonctions égales à $u_0(x)$ et $\tilde{u}_1(x)$, les conditions initiales. Dans le cas où l'on tente de retrouver un mode propre, la sortie `erreur` sera la différence en norme 2 vectorielle entre la solution approchée et la valeur exacte du mode propre approchée :

$$\|u_{\text{appro}} - u_{\text{exacte}}\|_2 .$$

sinon l'erreur sera un tableau vide (`erreur=[]`). On prend comme paramètres pour la suite :

$$c = 1,\ T = 1\ N_t = 100,\ N_x = 100,\ \theta = 0$$

(comme mode exacte, on pourra par exemple prendre $u_2(x,t) = \cos\left(\frac{2\pi}{L}t\right)\sin\left(\frac{2\pi}{L}x\right)$).

b) Écrire une fonction qui trace l'évolution temporelle de u trouvée à la question précédente (cela devrait générer une sorte de petit film). On pourra s'aider de l'annexe MATLAB.

```
trace_solution(u,Nt,Nx)
```

On implémentera de même une fonction :

```
trace_comparaison_mode(u,Nt,Nx,uexacte)
```

qui trace sur le même graphique la solution exacte pour le mode propre et son approximation trouvée par l'algorithme de la question précédente (on prendra garde à tracer `u` et `uexacte` de deux couleurs différentes).

c) À l'aide de cette fonction, retrouver le troisième mode propre (en excluant 0) donné par l'équation (3) ($f(t) = 0$). Les conditions initiales à appliquer sont les suivantes : $u_0(x) = \tilde{u}_1(x) = \sin(\omega_x x)$. Tracer l'erreur définie précédemment en fonction du temps.

d) On va faire varier les paramètres N_t, N_x et θ pour comprendre l'intérêt des schémas implicites ($\theta \neq 0$). On demande de calculer le déplacement approché correspondant au deuxième mode propre $\omega_t = 2\pi$ pour les paramètres suivants :

θ	N_x	N_t
0	100	95
0.1	100	78
0.1	100	70
0.5	100	50
0.5	100	30

Le phénomène observé pour $\theta = 0$, $N_x = 100$ et $N_t = 95$ est un phénomène d'instabilité. L'observez-vous dans le cas de $\theta = 0.1$? $\theta = 0.5$?

En fait on peut prouver que, pour des valeurs de θ comprises entre 0 et 1/4 (strictement), la discrétisation est stable seulement si :

$$\frac{\Delta_t}{\Delta_x} \leq \frac{1}{\sqrt{c}}\sqrt{\frac{1}{1-4\theta}} \tag{10}$$

et elle est stable pour θ entre 1/4 et 1/2 indépendamment des N_x et N_t. On prendra garde au fait que stable signifie que l'erreur n'explose pas au cours du temps mais cela n'assure pas une bonne précision. Dans tous les cas, plus N_x et N_t seront petits plus l'erreur sera petite.

e) À l'aide de (10), expliquer les résultats trouvés précédemment. Quel est l'avantage des discrétisations produites par $0.25 \leq \theta \leq 0.5$ par rapport à celle produite par $\theta = 0$? Est-ce qu'il y a un intérêt à utiliser $0 < \theta < 0.25$?

f) On prend $\theta = \frac{1}{2}$, $c = 10$, $T = 5$, $N_t = 500$ et $N_x = 100$. Résoudre (4) pour $g(t) = 0$ et $f(t) = 0.1\cos(2\pi\sqrt{10}t)$. Pour les conditions initiales, on choisit $u_0(x) = 0.1(1-x)$ et $\tilde{u}_1(x) = 0.1\cos\left(2\pi\sqrt{10}\Delta t\right)(1-x)$. Tracez l'évolution du déplacement. Que remarquez-vous ? Comment retrouver le troisième mode propre ? Vérifier votre réponse en effectuant la simulation.

Partie 3 : Problème inverse

NOTE : on fera **au choix** l'exercice 2 (méthode de la sécante) **ou** l'exercice 3 (méthode du gradient). Si le cœur vous en dit, vous pouvez faire les deux ! Les exercices 1 et 4 sont obligatoires.

À présent, on considère le problème dit inverse. Le problème est le suivant : à partir de données expérimentales u_{exp} - le déplacement de la corde en un certain point de l'espace au cours du temps - est-on capable de retrouver la valeur de c intervenant dans l'équation des ondes ? On notera cette valeur c_{exp}, c'est donc l'inconnu du problème inverse. Notre corde est toujours de longueur $L = 1$. On place un «capteur» au quatrième nœud de discrétisation en espace, mesurant le déplacement de la corde au cours du temps, on notera ce nœud $x_{capteur}$.

Afin de déterminer c_{exp} on introduit la fonction suivante, où u_c est la solution de l'équation des ondes pour le paramètre c :

$$J(u_c) = \int_0^T |u_c(x_{capteur}, t) - u_{exp}(x_{capteur}, t)|^2 \, dt \tag{11}$$

où u_{exp} sont les valeurs expérimentales. Remarquons tout de suite que la fonction J est minimale quand $c = c_{exp}$. On va tout d'abord regarder à quoi ressemble J en fonction de c et générer (artificiellement) un jeu de données expérimentales u_{exp} à l'aide du code précédemment écrit. Puis on présentera trois méthodes pour retrouver c_{exp}.

Créer un fichier `script_devoir_pb_inverse.m`. Ce script contiendra l'appel aux 3 méthodes que l'on va voir pour résoudre le problème inverse (sécante/différence finie et recuit simulé) et permettant de trouver c_{exp}. Dans toute la suite, pour toutes les méthodes, on prendra les paramètres suivants

$$\omega = 5, \quad \theta = \frac{1}{2}, \quad L = 1, \quad T = 1, \quad N_t = 500, N_x = 100,$$
$$g(t) = 0, \quad f(t) = \cos(\omega t), \quad u_0(x) = 0.1(1 - x), \quad \tilde{u}_1(x) = 0.1 \cos(\omega \Delta t)(1 - x)$$

On pourra déclarer ω, θ, L, T, N_t, N_x comme variables globales dans notre script `script_devoir_pb_inverse.m` (cf. annexe MATLAB). Pour générer des données expérimentales, le plus simple est de se donner c_{exp} et de résoudre l'équation des ondes pour trouver le déplacement u_{exp} associé. Le but du problème inverse est donc d'être capable de retrouver c_{exp} uniquement à partir de la connaissance de u_{exp}.

On fera attention que le paramètre c doit être positif en tout temps. Dans toutes les applications, on considère que si $c > 100$, on a divergé. Un critère de divergence pour les

algorithmes pourra être de s'assurer que $0 < c < 100$.

Exercice 1 :

Le but de cet exercice est d'être capable d'écrire un script calculant la valeur de l'intégrale (11) et d'afficher sa valeur pour différentes valeurs de c (et donc différents u_c associés).

a) Afin de calculer $J(u_c)$, écrire une fonction :

```
J=calcul_valeur_integrale(u_c,u_ex)
```

Les paramètres d'entrée sont u_c (le déplacement correspondant à une certaine valeur de c), trouvé à l'aide de la fonction de résolution de l'équation des ondes, et u_{exp} la donnée expérimentale. Pour calculer numériquement l'intégrale, on utilisera l'annexe à ce sujet.

b) On souhaite également tracer la valeur de J pour différentes valeurs de c. Pour cela, écrire une fonction :

```
trace_fonction_objectif(u_exp,c_intervalle,f,u0,u1tilde)
```

qui prend un intervalle de valeur de vitesse `c_intervalle` et les données expérimentales cibles `u_exp`. Dans cette fonction, on bouclera sur les différentes valeurs de c, puis pour chaque valeur de c, on calculera le déplacement u_c associé, puis on calculera la valeur de l'intégrale correspondant à ce u_c. On finira par afficher la valeur de J en fonction des valeurs de c. Dans l'appel de la fonction, les paramètres `f,u0,u1tilde` représente toujours, respectivement, la condition à la limite en $x = 0$ et les conditions initiales au temps 0 et Δt permettant de calculer u_c.

c) Au début du script `script_devoir_pb_inverse.m`, générer les données expérimentales u_{exp} en utilisant la fonction `resout_equation_onde_melde`, avec $c_{exp} = 10$.

d) Tracer l'objectif $J(u_c)$ en fonction de c pour des valeurs de c comprises allant de 7 à 30 avec un pas de 0.5.

Exercice 2 :

Le but de cet exercice est de voir une première méthode pour déterminer c_{exp}. On va utiliser le fait que $J(u_c)$ s'annule quand $c = c_{exp}$. On cherche donc une racine de $J(u_c)$, c'est à dire r telle que $J(u_r) = 0$.

a) Peut-on utiliser la méthode de la bissection pour trouver r ?

b) Implémenter une fonction qui retrouve c_{exp} à l'aide de la méthode de la sécante :

```
[cfinal,ufinal,tab_err]=pb_inv_secante(u_ex,nmax,precision,c0,c1,f,u0,u1tilde)
```

Les paramètres d'entrées sont les suivant : `u_ex` est la valeur du déplacement expérimental cible, `nmax` le nombre d'itérations maximal de la méthode de la sécante, `precision` la précision pour le critère d'arrêt, `c0` et `c1` les valeurs initiales pour les vitesses. Les paramètres de sortie sont : `cfinal` la vitesse finale trouvée par l'algorithme, `ufinal` le déplacement associé au `cfinal`, et enfin `tab_err` le tableau des erreurs absolues successives.

c) Tester votre algorithme en partant des valeurs suivantes :

c0	c1
3	3.1
15	15.1
17	17.1

Pour chacun des cas on tracera l'erreur absolue ($|J(u_c) - J(u_{exp})|$) en fonction des itérations de la sécante. En vous aidant des courbes de l'objectif en fonction de la vitesse précédemment tracées, expliquez la divergence dans le troisième cas.

Exercice 3 :

Le but de cet exercice est de voir une seconde méthode pour déterminer c_{exp}. Dans cette partie on va utiliser le fait que quand $c = c_{exp}$, la fonction $J(u_c)$ atteint un minimum global. On cherche alors à trouver ce minimum. On se reportera à l'annexe sur la minimisation de fonction pour une explication rapide des méthodes dites de descente.

Un minimum de la fonction J est forcément atteint quand la dérivée de $J(u_c)$ par rapport à c s'annule, i.e $\frac{dJ}{dc} = 0$ (attention cela peut correspondre à un minimum local de la fonction J). Trouver le point où la dérivée s'annule nécessite donc d'être en mesure de calculer la dérivée de J par rapport à c.

a) Pour calculer le gradient de J, on propose de faire :

$$\frac{dJ}{dc} \approx \frac{J(u_{c+\varepsilon}) - J(u_{c-\varepsilon})}{2\varepsilon}.$$

Écrire une fonction

```
gradient=calc_grad_diff_ini(u_ex,c,f,u0,u1tilde)
```

qui pour un c donné, donne le gradient $\frac{dJ}{dc}$. On devra donc à l'intérieur de cette fonction, résoudre l'équation des ondes avec $c + \varepsilon$ et $c - \varepsilon$ afin de déterminer les déplacements associés et pouvoir calculer $J(u_{c+\varepsilon})$ et $J(u_{c-\varepsilon})$. Comment coder cette méthode en évitant les opérations numériques dangereuses ? On pourra prendre $\varepsilon = 10^{-5}$.

b) Écrire une fonction

`[cfinal,ufinal]=pb_inv_minimisation(u_ex,nmax,precision,cini,f,u0,u1tilde)`

qui, partant d'une valeur $c = c_{ini}$, résout le problème inverse en utilisant une méthode de descente, basée sur le gradient (un bon critère d'arrêt est que la valeur absolue du gradient devienne très petite, la variable `precision` sert à contrôler cela). On utilisera la fonction écrite à la question précédente.

c) Lancer l'algorithme en partant des c initiaux : 3, 15, 17, 21, 23. Que se passe-t-il dans le cas de la dernière valeur ($c = 23$) ? On pourra tracer le graphe de $J(u_c)$ en fonction de c pour des valeurs de c comprises entre 10 et 30 avec un pas de 0.1 afin d'expliquer la non convergence de l'algorithme.

Exercice 4 :

Pour trouver le minimum global de la fonction J, on propose une dernière approche, stochastique, dite de «Recuit Simulé» (annexe 3).

a) Implémenter l'algorithme de recuit simulé présent dans l'annexe en question. La signature de la fonction sera la suivante

`function[cfinal,iter]=pb_inv_recuit_simule(u_ex,cini,vois,nmax,f,u0,u1tilde)`

`vois` représente «la distance» à laquelle on va chercher un voisin (cf. algorithme), on pourra prendre une valeur entre 15 et 20. Le nombre d'itérations est donné par `nmax`. Pour les paramètres du recuit simulé définis dans l'annexe 3, on propose d'utiliser les valeurs suivantes $\alpha = 0.8$, $R = 100$, `vois`$= 20$ Lancer l'algorithme à partir des valeurs de c suivantes $c = 3, 15, 17, 21, 23$. Que pouvez-vous dire quand à la précision et le coût de cette méthode ?

Attention, comme mentionné dans l'annexe, s'agissant d'un algorithme stochastique, pour chaque valeur de c on fera plusieurs lancements pour garantir le résultat !

b) Comment pourrait-on coupler les différentes méthodes afin d'assurer une convergence vers c_{exp} avec une bonne précision (en partant de n'importe quel c_{ini}) ?