Dépannage

Patrice Courchesne

Aide pour la partie 2 du devoir

Structure du dépannage

Problème regardé

On s'intéresse à l'équation des ondes :

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

où les fonctions suivantes sont données :

Conditions aux limites

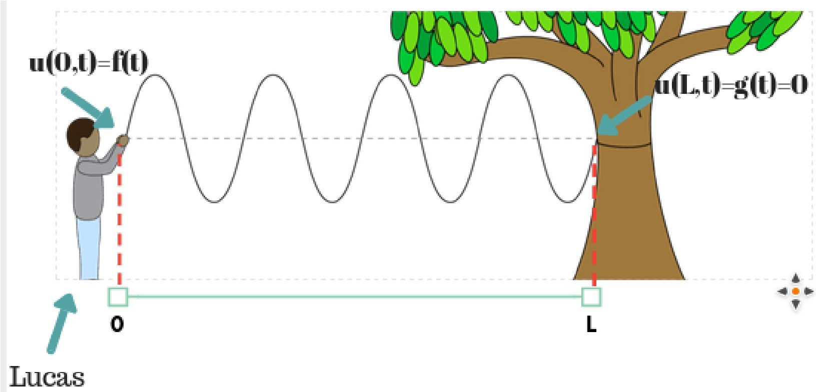
- **f(t)** (mouvement de la particule du début de la corde)
- **g(t)** mouvement de la particule au bout de la corde)

Conditions initiales

- $\mathbf{u_0(x)}$ (représente l'état initiale de la corde)
- $\mathbf{u_1(x)}$ (la vitesse initiale de la corde au point x)

Exemple

- $\mathbf{f(t)}$ (mouvement induit par Lucas)
- $\mathbf{g(t)} = 0$ (la corde est fixée sur l'arbre)
- $\mathbf{u_0(x)}$ (la fonction sinusoïdale représentée)
- $\mathbf{u_1(x)} = 0$ (rien n'est en mouvement pour l'instant)



Rappel théorique utile

Afin de résoudre numériquement, on approxime les dérivées par ce que l'on appelle des "différences finies". Voyons voir à l'aide d'un exemple pourquoi cette technique de discrétisation nous permet bel et bien d'approximer les dérivées.

Il est possible d'approximer la dérivée première de u par rapport à x par une différence finie centrée :

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

Vérifions que cette approximation en est bel et bien une.

Développement de Taylor

Développement de Taylor centré en x :

$$\begin{aligned} u(x+\Delta x,t) &= u(x,t) + \frac{\partial u}{\partial x}(u,t)\Delta x + \frac{\partial^2 u}{\partial x^2}(u,t)\frac{\Delta x^2}{2} + O(\Delta x^3) \\ u(x-\Delta x,t) &= u(x,t) - \frac{\partial u}{\partial x}(u,t)\Delta x + \frac{\partial^2 u}{\partial x^2}(u,t)\frac{\Delta x^2}{2} - O((\Delta x)^3) \end{aligned}$$

En soustrayant les deux équations on obtient alors :

$$u(x+\Delta x,t) - u(x+\Delta x,t) = 2\frac{\partial u}{\partial x}(u,t)\Delta x + \underbrace{O(\Delta x^3) + O((\Delta x)^3)}_{O(\Delta x^3)}$$

En isolant la dérivée, on a donc :

$$\frac{\partial u}{\partial x}(x,t) = \frac{u(x+\Delta x,t) - u(x-\Delta x,t)}{2\Delta x} + \underbrace{\frac{O((\Delta x)^3)}{2\Delta x}}_{O(\Delta x^2)}$$

Ainsi,

$$\frac{u(x+\Delta x, t) - u(x-\Delta x, t)}{2\Delta x}$$

est une approximation d'ordre 2 de la dérivée partielle première de u par rapport à x.

Trucs Matlab utiles

Discrétisation d'un intervalle [a,b] par bond de h

- Entrée : **a** : **h** : **b**
- Sortie : un **vecteur ligne** contenant ce que nous voulons

- Exemple :

Command Window

New to MATLAB? See resources for Getting Started.

```
>> x=0:2:10

x =

     0     2     4     6     8    10
```

Cela nous sera utile pour discrétiser notre intervalle de temps et notre intervalle d'espace.

Trucs Matlab utiles

Création de fonction et évaluations de fonctions

- NOMDELAFONCTION=@(x)EXPRESSION ALGÉBRIQUE
  Pour f une fonction Matlab et x un vecteur (ou même une matrice) Matlab, on a
- si l'entrée est f(**x**)
- la sortie est un **vecteur** (respectivement une matrice) de la même dimension que **x**

Attention : Il est important que toutes les multiplications et les exponentiations définies dans f aient un point juste avant (.*), sinon la fonction ne pourra pas prendre en argument des vecteurs !

Command Window

New to MATLAB? See resources for Getting Started.

```
>> x=0:2:10

x =

     0     2     4     6     8    10

>> f=@(x)x.*3.^2

f =

  function_handle with value:

    @(x)x.*3.^2

>> fx=f(x)

fx =

     0    18    36    54    72    90
```

On a alors que **fx** est le vecteur d'évaluation de f au point correspondant de x.

Trucs Matlab utiles

Extraction de la $i^{ème}$ ligne ou de la $j^{ème}$ colonne d'une matrice

Pour une matrice A Matlab préalablement définie ;

- Entrée : $A(i,:)$
- Sortie : **vecteur ligne** contenant la $i^{ème}$ ligne de A
- Entrée : $A(:,j)$
- Sortie : **vecteur colonne** contenant la $j^{ème}$ colonne de A

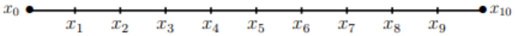
Cela est utile lorsque l'on veut définir une colonne d'une matrice. Par exemple, si l'on souhaite que la première colonne de B soit le vecteur **colonne** v, on écrit

$$A(:,1) = v$$

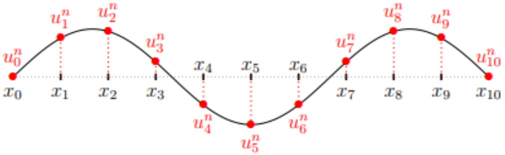
```
>> A=zeros(3,3)

A =

     0     0     0
     0     0     0
     0     0     0

>> v=[1;3;5]

v =

     1
     3
     5

>> A(:,1)=v

A =

     1     0     0
     3     0     0
     5     0     0
```

But de : resout_equation _onde

Pour une discrétisation en espace donnée, disons :



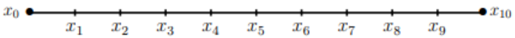
On veut calculer la hauteur de la fonction u au temps t_n et «garder en mémoire» ces valeurs dans une matrice (voici une visualisation) :



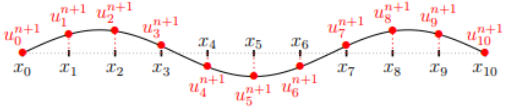
$$\begin{bmatrix} \vdots & u_0^n & \vdots \\ & u_2^n & \\ & u_3^n & \\ \vdots & \vdots & \vdots \\ & u_9^n & \\ \vdots & u_{10}^n & \vdots \end{bmatrix}$$

But de : resout_equation _onde

Pour une discrétisation en espace donnée, disons :



On continue à calculer la hauteur de la fonction u, cette fois-ci au temps suivant t_{n+1} et de «garder en mémoire» les valeurs dans la matrice (seconde visualisation) :



$$\begin{bmatrix} \vdots & u_0^n & u_0^{n+1} & \vdots \\ & u_1^n & u_1^{n+1} & \\ & u_2^n & u_2^{n+1} & \\ \vdots & \vdots & \vdots & \vdots \\ & u_9^n & u_9^{n+1} & \\ \vdots & u_{10}^n & u_{10}^{n+1} & \vdots \end{bmatrix}$$

Autrement dit, on veut avoir une matrice de la forme :

$$\begin{bmatrix} u_0^0 & u_0^1 & \dots & u_0^n & \dots & u_0^{N_t-1} \\ u_1^0 & u_1^1 & \dots & u_1^n & \dots & u_1^{N_t-1} \\ u_2^0 & u_2^1 & \dots & u_2^n & \dots & u_2^{N_t-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_9^0 & u_9^1 & \dots & u_9^n & \dots & u_9^{N_t-1} \\ u_{10}^0 & u_{10}^1 & \dots & u_{10}^n & \dots & u_{10}^{N_t-1} \end{bmatrix}$$

Il reste à savoir comment calculer les colonnes de cette matrice.
C'est le rôle de resout_equation _onde !

Esquisse de resout_equation _onde

resout_equation _onde

- **Entrée** : Ensemble des paramètres nous permettant de résoudre l'équation des ondes
- **Sortie 1** : Une matrice (dim. $Nx \times Nt$), u, dont la j^{ieme} colonne représente l'approximation de $u(x, j\Delta t)$
- **Sortie 2** un vecteur (dim. $Nt \times 1$) dont la j^{ieme} entrée est l'erreur entre le vecteur de solution exacte et le vecteur de la solution approximée en norme 2 au temps $j * \Delta t$.

La méthode de résolution se base sur la formule suivante,

$$AU^{n+1} = B_0 U^n + B_1 U^{n-1} \quad \text{avec } A, U, B_0, B_1 \text{définit dans le devoir}$$

Il s'agit d'une méthode à **deux pas**. Ainsi, avant de résoudre, on doit fournir $u(x, 0)$ et $u(x, \Delta t)$. Il s'agira des deux premières entrées de la matrice de sortie.

Stratégie de résolution

Dans le script :

1. Initialiser les paramètres globaux (L, T, α, ω_x) du problème ;
2. Initialiser les autres paramètres dans le même script (c, Nt(à calculer) ,Nx (à calculer) ,theta ,f ,u_0,utilde) ;
3. faire appel à la fonction

Dans la fonction Matlab,

1. discrétisation de l'intervalle ;
2. création des deux première colonnes de la matrice u ;
3. création des matrice A, B_0 et B_1 ;
4. création d'une boucle qui nous permettra de résoudre les systèmes matricielles $\underbrace{A}_{connu} U^{n+1} = \underbrace{\left(B_0 U^n + B_1 U^{n-1}\right)}_{\text{connu et à modifier légèrement}}$

Stratégie de résolution (suite)

Également, il est possible d'ajouter une variable globale (disons coefferr) qui indiquera si nous voulons ou non calculer l'erreur.

- Si coefferre==0, erreur=[]
- Si coefferr==1 :
  - On définit une fonction anonyme de la solution exacte (voir la partie 1 lettre d)
  - On calcule la solution exacte pour tous les points (x_i, t_j) de notre discrétisation (on obtiendra une matrice de dim. $(Nx \times Nt)$ disons matsolexacte)
  - On calcule la norme 2 de chaque colonne de la matrice de soustraction matsolexacte-u.