République Algérienne Démocratique Et Populaire
Ministère de L'enseignement Supérieur et de la Recherche Scientifique
Université Dr. Yahia Fares, Médéa
Faculté des Sciences
Département de Mathématiques et Informatique



MÉMOIRE DE FIN D'ÉTUDES POUR L'OBTENTION DU DIPLÔME DE MASTER
SPÉCIALITÉ : **ANALYSE ET MODÉLISATION MATHÉMATIQUES**

Présenté par :

DERIADI Fatma

Intitulé :

La résolution des équations differentielles partielles par SCILAB

Soutenu publiquement le 12 juin 2017 à Médéa devant le jury composé de

M. BENCHAA	M.A.A	Université de Médéa	Président
K. TOUMACHE	M.A.A	Université de Médéa	Examinateur
N. MEKDACHE	M.A.A	Université de Médéa	Examinateur
K. MAALOUM	M.A.A	Université de Médéa	Encadreur

Table des matières

Résumé vii

Notations viii

Introduction viii

1 Scilab 1

1.1 Vue d'ensemble . . . 1

1.1.1 Présentation du logiciel . . . 1

1.1.2 Installation du logiciel . . . 2

1.1.3 L'environnement et utilisation du logiciel Scilab . . . 2

1.2 Utilisation de Scilab comme une calculette . . . 4

1.3 Définition et manipulation des objets . . . 5

1.3.1 Objets Scilab . . . 5

1.3.2 Objets vectoriels et matriciels . . . 8

1.3.3 Construction de vecteurs et de matrices . . . 8

1.3.4 Extraction et insertion . . . 11

1.3.5 Opérations vectorielles et matricielles . . . 13

1.4 Programmation en Scilab . . . 16

1.4.1 L'éditeur SciNotes . . . 16

1.4.2 Structures conditionnelles . . . 17

1.4.3 Boucles itératives . . . 19

1.5 Les fonctions . . . 20

1.6 Les graphiques avec scilab . . . 21

1.6.1 Les fenêtres graphiques 21
1.6.2 Modifier un tracé 22
1.6.3 L'instruction plot 22
1.6.4 L'instruction plot2d 24
1.6.5 L'instruction plot3d 25
1.7 L'aide en ligne 26

2 Les équations aux dérivées partielles linéaires classiques 28
2.1 Introduction 28
2.2 Rappels sur la dérivation partielle 29
2.3 Équations aux dérivées partielles linéaires 31
2.3.1 Conditions aux bord (conditions aux limites) 32
2.3.2 Probléme bien posé 32
2.3.3 Notions de solution 33
2.4 Modélisation 33
2.5 Quelques résolutions théoriques d'E.D.P. 35
2.6 L'équation de transport 37
2.7 Équation de la chaleur 38
2.8 Équation des ondes 41
2.9 Équations de Laplace et Poisson 45
2.10 Classification des EDP d'ordre 2 47

3 Exemples d'applications 49
3.1 Résolutions (E.D.) d'ordre 1 (la fonction ode()) 49
3.1.1 Point de terminologie et de forme 49
3.1.2 L'appel de la commande ode de Scilab 49
3.2 Transformer une E.D. d'ordre 2 en E.D. vectorielle d'ordre 1 52
3.2.1 La méthode illustrée sur une E.D. Linéaire 52
3.2.2 Traduction en Scilab 52
3.3 Résolution d'une EDP parabolique 54
3.4 Discrétisation de l'espace et du temps 55
3.4.1 Discrétisation de l'espace 55

3.4.2 Discrétisation du temps 55
3.5 Programmation de la solution analytique 56

Conclusion **58**

Bibliographie **59**

Remerciements

Avant tout je remercie **ALLAH** tout puissant de m'avoir donné le courage et la patience pour terminer ce travail.

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner toute ma reconnaissance.

Je voudrais tout d'abord adresser toute ma gratitude à Monsieur **K. MAALOUM**, Maître de Conférences à l'université Dr. Yahia Farès de Médéa, d'avoir accepté d'encadrer mon mémoire, ainsi pour sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter ma réflexion.

Je remercie également Monsieur **M. BENCHAA**, Maître Assistant à l'université Dr. Yahia Farès de Médèa, pour l'honneur qu'il me fait en acceptant de présider le jury de ce mémoire.

Je tiens à exprimer mes sincères remerciements à Monsieur **K. TOUMACHE**, Maître Assistant à

l'université Dr. Yahia Farès de Médéa, pour son aide, sa disponibilité, et ses bons conseils.

Je remercie également Monsieur **N. MEKDACHE**, Maître Assistant à l'université Dr. Yahia Farès de Médèa, pour son aide, sa disponibilité.

Je lui témoigne ma gratitude pour avoir pu prendre part à ce jury.

Enfin, merci d'avance à ceux ou celles qui voudront bien me lire ou m'écouter et surtout m'aider à progresser.

Dédicace

Je dédie ce travail

À ma très chère **mère**

Tu es l'exemple de dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Puisse **ALLAH**, le tout puissant, te préserver et t'accorder santé, longue vie et bonheur.

À mon très chère **père**

Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être. Ce travail est le fruit de tes sacrifices que tu as consentis pour mon éducation.

À mes frères : **Oussama**, **Mohamed Rayan**.

À mes surs : **Ahlem** et **Imane**.

Vous vous êtes dépensés pour moi sans compter. En reconnaissances de tous les sacrifices consentis par tous et chacun pour me permettre d'atteindre cette étape de ma vie.

À mes neveux, meilleurs vux de succès dans vos études.

À toute la famille **DERIADI**.

À mes enseignants de l'école primaire jusqu'à l'université dont les conseils précieux m'ont guidé, qu'ils trouvent ici l'expression de ma reconnaissance.

À mes **amies** et **À** leurs familles.

Je vous remercie de votre patience vous m'a aidée toujours à avancer vous êtes tous des grandes amies si gentilles, merci d'être toujours près de moi, amies avec lesquelles je souris.

À toute la promotion d'analyse et modélisation mathématiques.

À tous ce qui m'aiment. À tous ce que j'aime surtout Mouna et Koko.

2016-2017

FATMA

Résumé

Ce travail consiste à étudier la résolution des équations défferentielles partielles par SCILAB, il s'appuie essentiellement sur le livre [2] publié en 2007/2008 par Ch. Billon, on a aussi tiré grand profit du livre [7] publié par P. Ferreira, Sylvie, Mas-Gallic en 2001.

Le mémoire est composé de trois chapitres : dans le premier on donne : Présentation du logiciel, Installation du logiciel,Utilisation de Scilab comme une calculette, Programmation en Scilab, Les graphiques avec scilab.

Dans le second chapitre, on rappelle Les équations aux dérivées ordinaire et partielles par exemple ; L'équation de transport, Equation de la chaleur, Equations de Laplace et Poisson, Solution fondamentale de l'équation de la chaleur.

Exemples d'applications elle sera faite au chapitre 3.

Notations

.

$\mathbb{N}$: l'ensemble des nombres naturels.

$\mathbb{R}$: l'ensemble des nombres réels.

$\mathbb{R}^n$: espace euclidien de dimension $n, n \geq 1$.

$\mathbb{R}_+^*$: le demi-plan ouvert.

Ω : ouvert quelconque de $\mathbb{R}^n$.

$\overline{\Omega}$: la fermeture de Ω.

$\Gamma = \partial\Omega$: la frontière de Ω.

$x = (x_1, \ldots, x_n)$: est un vecteur de $\mathbb{R}^n$.

η : la normale extérieure à l'ouvert Ω.

$\frac{\partial u}{\partial \eta}$: la dérivéee normale extérieure.

D_i : la dérivée partielle $\frac{\partial}{\partial x_i}$, pour tout multi-indice $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$,

on pose $D^\alpha = D_1^{\alpha_1} \ldots D_n^{\alpha_n} = \frac{\partial^{|\alpha|}}{\partial_1^{\alpha_1} \ldots \partial_n^{\alpha_n}}$ telle que $|\alpha| = \alpha_1 + \ldots + \alpha_n$

$x^\alpha = x_1^{\alpha_1} \ldots x_n^{\alpha_n}$ pour tout $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$.

$\mathcal{C}^m(\Omega)$: l'espace des fonctions m fois continûment différentiables sur Ω, $(m \geq 0)$

$\mathcal{C}^m(\bar{\Omega})$: fonction m fois continument differentiables sur $\bar{\Omega}$

Introduction

Dans le monde scientifique, l'utilisation des logiciels de calcul numérique est aujourd'hui indispensable. Des chimistes qui simulent leurs modèles de processus aux ingénieurs qui conçoivent des systèmes de régulation, des spécialistes des télécommunications qui mettent au point des algorithmes de traitement du signal aux spécialistes de finance qui manipulent des modèles statistiques, tous ont des besoins importants en calcul numèrique. Scilab, qui est largement diffusé dans le monde universitaire et industriel depuis quelques années, est un logiciel libre de calcul numérique.

Pendant de nombreuses années, Scilab a été essentiellement développé par des chercheurs de l'INRIA et de l'ENPC avec de nombreuses contributions extérieures, souvent sous forme de boîtes à outils. Scilab est maintenant pris en charge par un consortium auquel participent financièrement une vingtaine de grandes sociétés.

Scilab n'est pas qu'une super-calculette : c'est un puissant environnement logiciel pour le développement d'applications scientifiques. Ces dernières nécessitent le plus souvent la manipulation des matrices et vecteurs et c'est pour cela que Scilab, comme d'autres logiciels tels que Matlab ou Octave, utilise une syntaxe adaptée à la manipulation de ces objets. En outre, Scilab propose des centaines des fonctions de calcul numérique regroupées en bibliothèques qui couvrent des domaines comme la simulation, l'optimisation, l'automatique ou le traitement du signal, ce qui facilite le développement d'applications particulières.

Bien que la plupart des fonctions soient de type mathématique, on peut aussi bien utiliser Scilab comme un langage de programmation généraliste. Il est par exemple tout à fait possible de développer un tableur dans Scilab.

Chapitre 1

Scilab

1.1 Vue d'ensemble

1.1.1 Présentation du logiciel

Scilab (contraction de scientific laboratory) est un logiciel écrit à l'origine par l'Institut National de Recherche en Informatique et en Automatique (INRIA), ce logiciel a une licence proche de la licence GPL, il est libre et gratuit. Il est maintenant développé par un consortium comprenant l'INRIA et l'ENPC. Vous pouvez le télécharger gratuitement sur http : www.scilab.org/download, ainsi qu'une nombreuse documentation gratuite. Scilab disponible sous UNIX, LINUX, Windows, et Mac OS.

On peut considérer que Scilab est un langage de script puisque il peut servir à une illustration algorithmique en quelques lignes. Mais c'est aussi un langage de programmation et la librairie Scilab contient presque 1200000 lignes de code de fonctions écrites en Scilab. La syntaxe de Scilab est simple, les données propres aux calculs scientifiques (Matrices) sont d'un maniement aisé. Cela conduit à des programmes compacts et lisibles souvent beaucoup courts que des programmes C ou le Fortran.

On peut facilement rajouter de nouvelles fonctionnalités a Scilab avec des librairies extérieures existantes.

1.1.2 Installation du logiciel

L'installation dépend du système d'exploitation (Windows, Unix, Linux). On peut installer une version binaire ou compiler Scilab à partir du code source. Cette dernière opération nécessite des compilateurs C et FORTRAN. Pour la version Windows, deux voies sont possibles : utiliser Cygwin si on veut rester dans l'univers du logiciel libre ou utiliser les outils Microsoft : Visual C++ ou Visual Express qui est disponible gratuitement. La compilation sous Windows est parfois plus délicate que sous Linux. Toutes les distributions Linux contiennent les compilateurs C et FORTRAN. Sur les stations de travail Unix, si les compilateurs natifs ne sont pas disponibles, on peut utiliser les compilateurs GNU.

Vous pouvez être averti des sorties de nouvelles versions du logiciel Scilab en vous inscrivant sur notre canal de notification a l'adresse suivante :
http ://lists.scilab.org/mailman/listinfo/release

1.1.3 L'environnement et utilisation du logiciel Scilab

Scilab se lance dans l'environnement WINDOWS. Dans un premier temps, pour se familiariser avec la syntaxe, il est plus facile d'utiliser Scilab comme une super calculatrice évoluée.
Par un double clique sur l'icône ad hoc, située généralement sur le bureau. On obtient alors l'ouverture d'une nouvelle fenêtre comme sur la ≪ FIG 1.1.3 ≫ mais il faut attendre l'apparition d'une flèche −−> avant d'utiliser le logiciel.

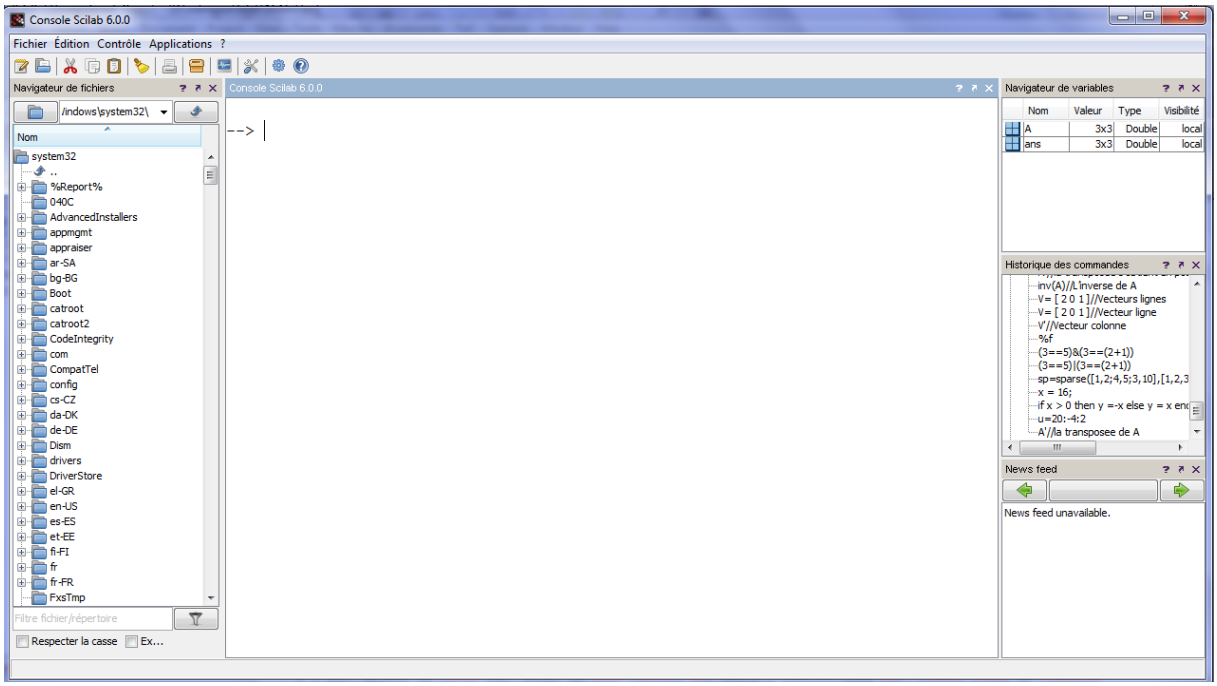


FIG 1.1.3 : La fenêtre principale de Scilab.

La fenêtre principale est composée de quatre sous-fenêtres :

•**Console Scilab :** c'est la fenêtre dans laquelle on tape les commandes pour exécution immédiate, et dans laquelle sont affichées les réponses.

•**Navigateur de variables :** contient le nom, la taille et le type de variables reconnues par la console.

•**Navigateur de fichiers :** précise le répertoire courant (celui dans lequel on travaille), et permet de naviguer dans le système de fichiers.

•**Historique des commandes :** contient toutes les lignes de commande validées dans la console.

En haut de la fenêtre, se trouve une barre de menu :

Fichier Édition Contrôle Application ?

À l'intérieur de cette fenêtre, on peut rentrer une suite d'instructions sur chaque ligne de commande commence par l'invite $-->$. Cette invite n'apparaît pas lors de l'exécution d'instructions ou lorsque le logiciel attend une saisie de l'utilisateur au clavier.

On a ensuite deux possibilités :

– si la ligne de commande se termine par le symbole $\ll ; \gg$, l'instruction est exécutée mais le résultat n'est pas affiché à l'écran . On dit que l'exécution s'est faite **sans écho**.

– si la ligne de commande ne se termine pas par le symbole « ; » (elle se termine par un espace ou par le symbole « , ») l'instruction est exécute et le résultat est affiché.

Une ligne de commande peut être modifiée tant qu'elle n'est pas validée. Une fois validée, on ne peut par contre plus la corriger. Cependant, on peut rappeler toute ligne de commande précédemment validée dans la console en utilisant les flèches haut et bas du clavier : ▲ et ▼.
Enfin, pour effacer les données en mémoire, il suffit d'utiliser la commande « clc ».

1.2 Utilisation de Scilab comme une calculette

Scilab peut être utilisé comme une calculette, en prenant en compte les règles habituelles de priorité relatives aux opérations mathématiques classiques :
se notent « + » pour l'addition, « − » pour la soustraction, « ∗ » pour la multiplication, « / » pour la division, « ^ » pour les exposants. La virgule des nombres décimaux est notée avec un point. Par exemple :

```
--> 2 + 3.4
 ans  =

   5.4

--> 4*5-6
 ans  =

   14.
```

• Il est nécessaire de bien respecter la casse (majuscules et minuscules) pour que les calculs s'effectuent correctement. Par exemple, avec la commande **sqrt** qui permet de calculer la racine carrée :

```
--> sqrt(9)
 ans  =

   3.
```

alors que :

```
--> SQRT(9)

Variable non définie : SQRT

    '
```

• Les résultats sont donnés à "epsilon machine" (%eps) près. Il s'agit d'un calcul numérique et non d'un calcul symbolique.

• Avec cette "calculette", on dispose d'un grand nombre de fonctions prédéfinies :

log( ) (logarithme népérien ln)

exp( ) (exponentielle)

floor( ) (partie entière)

abs( ) (valeur absolue)

sqrt( ) (racine carrée)

sin( ) (sinus)

cos ( ) (cosinus) etc...

• Dans Scilab, les calculs sont essentiellement numériques. Les nombres réels ou complexes sont en double précision, avec une précision maximale de 16 chiffres décimaux et leur valeurs absolues est comprise entre 10^{-308} et 10^{+308}.

• Par défaut, les résultats sont affichés avec 10 caractères, comprenant le point décimal et le signe.

1.3 Définition et manipulation des objets

Le logiciel Scilab contient des opérateurs puissants permettant la manipulation des objets de base, lesquels sont nombreux et variés.

1.3.1 Objets Scilab

Scilab propose une importante collection d'objets de base. Les plus élémentaires d'entre eux sont les nombres flottants et les chaînes de caractères. Bien sûr, Scilab reconnaît bien d'autres objets comme les booléens, les polynômes, les listes, etc. Une variable est un objet ou un ensemble d'objets réunis dans un vecteur, une matrice, une hypermatrice .

• Par ailleurs, on distingue deux formes de objet : celles qui sont prédéfinies et celles qui sont définies par affectation.

• Les objets numériques prédéfinis

%**e**, %**eps** et %**pi** représentent respectivement exp, $\epsilon_{machine}$ et π :

```
--> %e
 %e  =

   2.7182818


--> %pi
 %pi  =

   3.1415927


--> %eps
 %eps  =

   2.220D-16
```

%i représente la variable complexe **i** en entrée et s'affiche **i** en sortie :

```
--> 3+2*%i
 ans  =

   3. + 2.i
```

• objet Les booléennes prédéfinis

Scilab définit deux variables booléennes $\%t$ (true) et $\%f$ (false) :

```
--> %f
 ans  =

  F


--> %t
 ans  =

  T
```

Les symboles de comparaison sont :

== (test d'égalité)

< (inférieur strict)

<= (inférieur ou égal)

> (supérieur strict)

>= (supérieur ou égal).

On peut également former des expressions logiques plus compliquées :

~ (négation logique)

| ("ou" inclusif)

& ("et" logique)

Exemple 1.1 *:*

```
--> (3==5)&(3==(2+1))
 ans  =

  F


--> (3==5)|(3==(2+1))
 ans  =

  T
```

Ces constantes boolénnes permettent d'évaluer la valeur logique d'une relation.

- **Les objets ou variables scalaires définies par affectation**

L'affectation est l'instruction qui consiste à donner une valeur à une variable que l'on crée

ou que l'on modifie. Sa syntaxe est simple : on choisit un nom qui peut être une lettre ou un groupe de lettres et on lui attribue une valeur à l'aide du signe = .

Exemple 1.2 *:*

```
Ma-donnee = sin(2*%pi);
disp(a)
```

1.3.2 Objets vectoriels et matriciels

La caractéristique essentielle de Scilab est de pouvoir stocker et traiter les objets de base sous forme de tableaux 1D, 2D ou même multidimensionnels (vecteurs, matrices et hypermatrices).

1.3.3 Construction de vecteurs et de matrices

Une matrice est un tableau contenant certains objets de base comme les nombres entiers, réels ou complexes, les chaînes de caractères, les booléens et les polynômes. Un vecteur est un cas particulier de matrice n'ayant qu'une seule ligne ou colonne. Les matrices et les vecteurs peuvent être définis à l'aide des opérateurs élémentaires : « [ ] » , « , » et « ; ». La virgule est utilisée pour séparer les colonnes et le point-virgule pour séparer les lignes. Les crochets servent à délimiter le contenu de la matrice. Les objets que l'on utilise pour fabriquer une matrice doivent être de même type ou de types compatibles. Par exemple :

```
--> A= [1 1 1;2 4 8;3 9 27]
 A  =

   1.    1.    1.
   2.    4.    8.
   3.    9.    27.

--> A'//la transposée de A
 ans  =

   1.    2.    3.
   1.    4.    9.
   1.    8.    27.
```

```
--> inv(A)//L'inverse de A
 ans  =

   3.   -1.5   0.3333333
  -2.5   2.   -0.5
   0.5  -0.5   0.1666667

--> V= [ 2 0 1 ]//Vecteur ligne
 V  =

   2.   0.   1.

--> V'//Vecteur colonne
 ans  =

   2.
   0.
   1.
```

Il existe des fonctions pour construire des matrices et vecteurs types, dont voici une première liste (il y en a bien d'autres dont nous parlerons ultérieurement ou que vous découvrirez avec le Help) :

• Pour obtenir une matrice identité de dimension (3,3) :

```
--> I= eye (3,3)
 I  =

   1.   0.   0.
   0.   1.   0.
   0.   0.   1.
```

• Les fonctions **zeros** et **ones** permettent respectivement de créer des matrices nulles et des matrices "de un". Comme pour la fonction **eye** leurs arguments sont le nombre de lignes puis de colonnes désirées. Par exemple :

```
--> C = ones (3,4)
 C  =

   1.   1.   1.   1.
   1.   1.   1.   1.
   1.   1.   1.   1.

--> Z = zeros (C)
 Z  =

   0.   0.   0.   0.
   0.   0.   0.   0.
   0.   0.   0.   0.
```

La fonction rand

permet de créer des matrices remplies de nombres **pseudo-aléatoires**

```
--> T = rand(3,3)
 T  =

   0.7263507   0.2320748   0.8833888
   0.1985144   0.2312237   0.6525135
   0.5442573   0.2164633   0.3076091
```

• Pour rentrer un vecteur (ligne) X à n composantes régulièrement réparties entre X_1 et X_n (c-à-d telles $X_{i+1} - X_i = \frac{X_n - X_1}{n-1}$, n "piquets" donc $n-1$ intervalles ...), on utilise la fonction $\mathbf{linspace(X_1, X_n, n)}$:

```
--> X = linspace(-1,0,9)
 X  =

  -1.  -0.875  -0.75  -0.625  -0.5  -0.375  -0.25  -0.125   0.
```

1.3.4 Extraction et insertion

La notation $A(i, j)$, si les indices i et j sont des nombres, désigne l'élément (i, j) de la matrice A. Cette notation est utilisée aussi bien pour l'insertion que pour l'extraction. Considérons la matrice aléatoire de taille 3×3 :

```
--> A=rand(3,3)
 A  =

   0.4826472   0.5015342   0.6325745
   0.3321719   0.4368588   0.4051954
   0.5935095   0.2693125   0.9184708
```

On peut extraire l'élément $(3, 1)$ de cette matrice par la commande :

```
--> A(3,1)
 ans  =

   0.5935095
```

Le symbole $\ll : \gg$ peut être utilisé à la place d'un indice pour désigner $\ll$ tout $\gg$ (toutes les lignes ou toutes les colonnes,en fonction du placement).

```
--> A(2,:)
 ans  =

   0.3321719   0.4368588   0.4051954
```

On peut ainsi insérer une sous-matrice de la façon suivante :

```
--> A ([2,3], [1,2]) =ones (2,2)
 A  =

   1.   0.5015342   1.
   1.   1.          1.
   1.   1.          0.9184708
```

L'insertion d'une matrice vide $\ll [\ ] \gg$ peut être utilisée pour supprimer une ou plusieurs lignes (ou colonnes) d'une matrice. Par exemple pour supprimer les deux premières lignes

de A, on peut faire

```
--> A ([1,2],:) =[]
 A  =

   0.5935095    0.2693125    0.9184708
```

L'insertion et l'extraction pour les vecteurs peuvent bien évidemment être définies par un seul indice $V(i)$

```
--> V= [1 4 6 8 3];

--> V ([1 2])
 ans  =

   1.    4.

--> V ([4 6])= [-1 -2]
 V  =

   1.    4.    6.   -1.    3.   -2.
```

On note ici, que le vecteur V est automatiquement redimensionné lorsque l'on insère des éléments au delà de sa taille courante. Le plus souvent l'usage d'un seul indice pour les matrices revient à mettre en correspondance la matrice avec un vecteur colonne formé par la concaténation des colonnes de la matrice.

```
--> A=[1 2
  >   3 4];

--> A(3)
 ans  =

   2.
```

```
--> A ([1,3])
 ans  =

   1.
   2.
```

Il faut noter que la syntaxe $X = A(:)$ retourne un vecteur colonne X formé par la concaténation des colonnes de la matrice A.

1.3.5 Opérations vectorielles et matricielles

Toutes les opérations usuelles sur les matrices sont disponibles : somme de deux matrices de mêmes dimensions, produit de deux matrices (si leurs dimensions sont compatibles $(n, m) * (m, n) \ldots$), produit d'un scalaire et d'une matrice, etc ... Voici quelques exemples :

```
--> B= [1 3 0;4 2 8;3 9 5];

--> G = [1 1 1;2 4 8;-1 6 10];

--> M = G + B
 M  =

   2.    4.    1.
   6.    6.    16.
   2.    15.   15.

--> E = M*B
 E  =

   21.    23.    37.
   78.    174.   128.
   107.   171.   195.
```

```
--> B^3 // La puissance 3-ième de B
 ans  =

   121.   273.    192.
   556.   788.    984.
   633.   1179.   1061.


--> M/2 // on peut diviser une matrice par un scalaire
 ans  =

   1.   2.    0.5
   3.   3.    8.
   1.   7.5   7.5


--> R = 4 + M // Addition des coefficients par un nombre
 R  =

   6.    8.    5.
   10.   10.   20.
   6.    19.   19.


--> D = 2 * G // Multiplication des coefficients par un nombre
 D  =

   2.   2.    2.
   4.   8.    16.
  -2.   12.   20.
```

Une autre caractéristique très interessante est que les fonctions usuelles s'appliquent aussi aux matrices « élément par élément » :

```
--> exp (B)
 ans  =

   2.7182818   20.085537   1.
   54.59815    7.3890561   2980.958
   20.085537   8103.0839   148.41316


--> sqrt (B)
 ans  =

   1.          1.7320508   0.
   2.          1.4142136   2.8284271
   1.7320508   3.          2.236068
```

Opérations "élément par élément"

Si A et B sont deux matrices ayant la même dimension en appliquant ces opérations "élément par élément" pour multiplier ≪.∗≫ et diviser ≪ ./ ≫. De même, on peut élever à la puissance chaque coefficient en utilisant l'opérateur ≪ˆ ≫

```
--> G./G
 ans  =

   1.   1.   1.
   1.   1.   1.
   1.   1.   1.

--> B.*B
 ans  =

   1.    9.    0.
   16.   4.    64.
   9.    81.   25.
```

1.4 Programmation en Scilab

Une succession de d'instructions est appelée un **script**. On peut les saisir directement dans la console Scilab, mais ceci a de nombreux inconvénients, car à chaque modification du script, il faut le réécrire en entier. De plus, il est difficile de le sauvegarder pour une utilisation ultérieure. C'est pour ces raisons qu'on utilise systématiquement l'éditeur intégré Scinotes.

1.4.1 L'éditeur SciNotes

À partir de la fenêtre principale de Scilab, cliquez sur la première icône  dans la barre d'outils ou sur **applications** > **Scinotes** dans la barre de menus.
On tape du texte dans l'éditeur comme dans n'importe quel traitement de texte.
Dans l'éditeur de texte, l'apparition des parenthèses, ouvrantes et fermantes et des commandes de fin de boucle, de fonction et de test est automatique.
On peut cependant désactiver ces deux fonctionnalités dans le menu **options** > **complétion automatique,** en cliquant sur les deux entrées ci-dessous activées par défaut :

- (, [, . . .
- if, function, . . .

En principe, il faut aller à la ligne aprés chaque instruction, mais il est possible de taper plusieurs instructions sur une même ligne en les séparant par un point virgule « ; ».
Un décalage de début de ligne appelé indentation se fait automatiquement lorsqu on commence une boucle ou un test.
Dans l'exemple suivant, on résoudre l'équation ordinaire $\frac{\mathbf{dy}}{\mathbf{dt}} = \mathbf{y^2} - \mathbf{y} * \mathbf{sin(t)} + \mathbf{cos(t)}$ avec la condition initial $\mathbf{y(0) = 0}$:

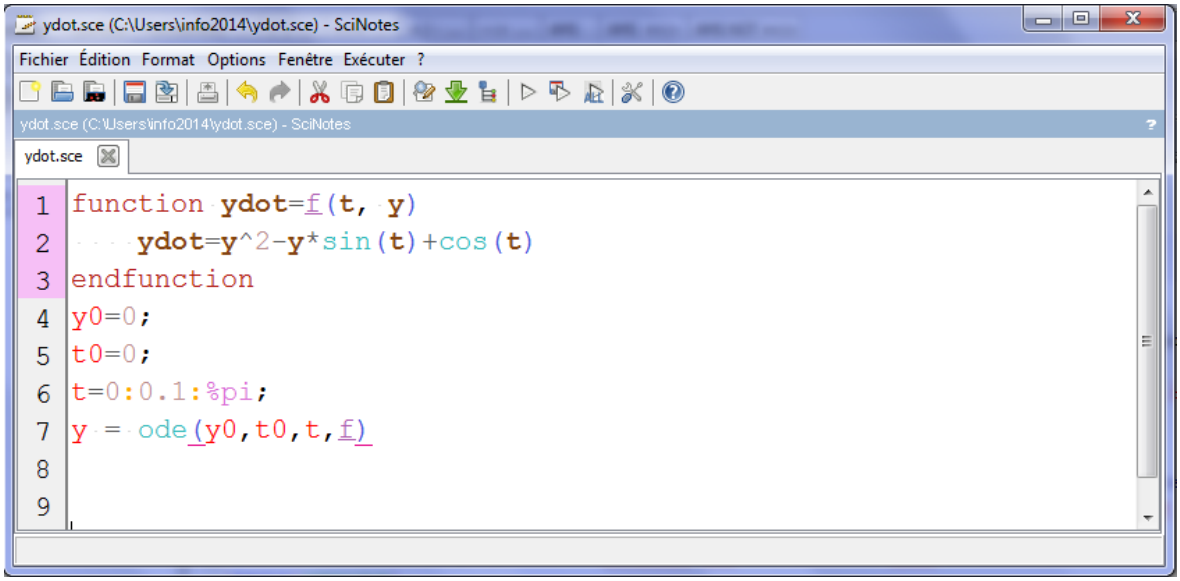


FIG 1.4.1 : La fenêtre de L'éditeur SciNotes .

Il est possible d'enregistrer tout fichier en cliquant sur **Fichier > Enregistrer sous**. l'extension ≪ .sce ≫ à la fin du nom de fichier déclenchera automatiquement le lancement de scilab à l'ouverture du fichier (excepté sous Linux et mac Os X).
Pour exécuter un programme en cliquant sur Exécuter dans la barre de menus.

Remarque 1.1 *:*

1) Pour écrire des commentaires qui ne seront pas pris en compte dans les calculs, les faire précéder de "//".

2) À l'écriture d'un programme, l'indentation est automatique. si toute fois cela n'était pas le cas, cliquez sur ***Format*** *>* ***corriger*** *l'indentation pour la rétablir (Ctrl I sous Windows et linux ou Commande I sous mac Os).*

1.4.2 Structures conditionnelles

Ces instructions permettent de conditionner l'exécution d'instructions à la valeur d'une condition. La forme la plus simple est la suivante :

```
if <condition> then <instructions> end
```

Lorsque "condition" booléenne est vraie (resp. fausse), la séquence "instructions" est (resp. n'est pas) exécutée. On notera que les instructions terminées par une virgule seront affichées,

celles terminée par un point-virgule ne seront pas affichées. Il est possible de donner une séquence alternative d'instructions à exécuter lorsque la condition test est fausse. Cette instruction est introduite par le mot-clef : else

```
if <expression > then <instructions1 > else <instructions1> end
```

La syntaxe est par exemple :

```
--> x = 16;

--> if x > 0 then y =-x else y = x end
 y  =

  -16.
```

Cette derniére syntaxe utilise le mot-clef elseif et permet d'effectuer une série de tests successifs, auxquels sont associés des actions spécifiques. Dans le cas où le test doit départager un "grand nombre" (supérieur à deux ...) de possibilités, on a aussi la possibilité d'utiliser le **select-case**. La syntaxe est par exemple :

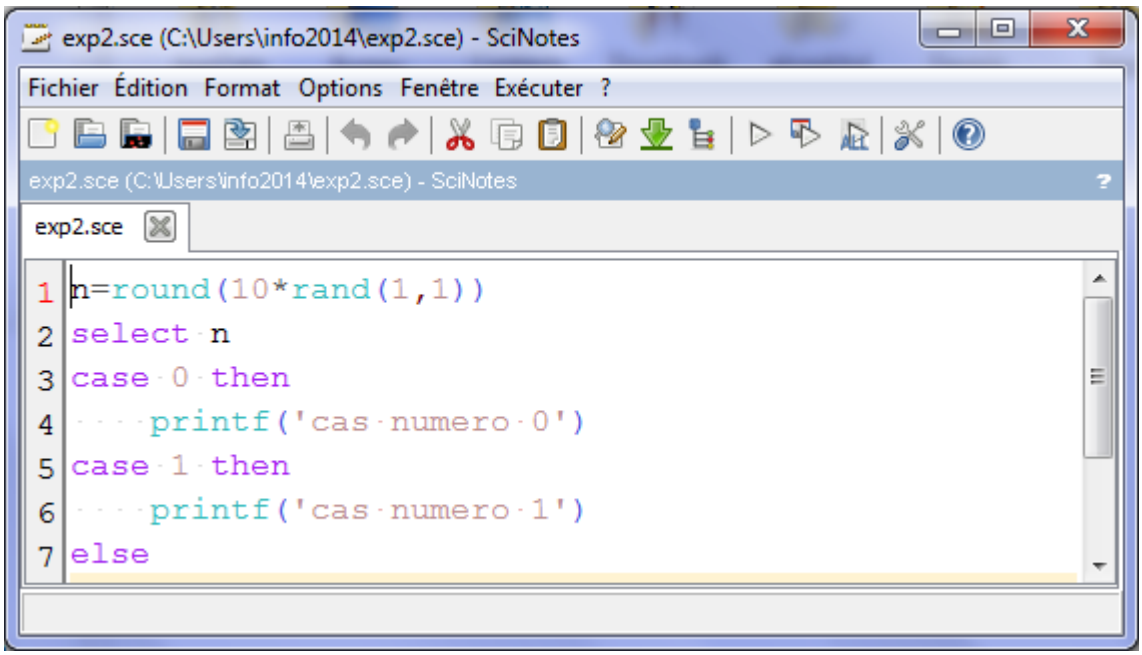


FIG 1.4.2 : La fenêtre de L'éditeur SciNotes .

1.4.3 Boucles itératives

L'incrémentation

L'opérateur ≪ : ≫ permet de définir des vecteurs de nombres dont les coordonnées sont en suite arithmétique. On donne ≪ la première valeur : le pas : la dernière valeur ≫.

Si le pas n'est pas mentionne, sa valeur est par défaut, 1.

Par exemple, pour définir un vecteur ligne d'entiers qui se décrémentent de 4 entre 20 et 2 :

```
--> u=20:-4:2
 u  =

   20.   16.   12.   8.   4.
```

For

La structure de boucle la plus simple pour répéter une suite d'instructions un nombre déterminé de fois, connu à l'avance, s'écrit avec **for** ... **end** qui signifie ≪ Pour ... fin de pour ≫.

Exemple 1.3 *:*

```
for i = 1:5
    disp(i)
end
```

While

La boucle while exécute son contenu tant qu'une condition est réalisée, on utilisera la forme **while** ... **end** qui signifie ≪ Tant que... fin de tant que ≫.

Exemple 1.4 *:*

```
i = 0
while i < 5 do
    disp(i);
    i= i+1;
end
```

Les deux types de boucles peuvent être terminés prématurément par l'instruction **break**. L'instruction **continue** permet de passer directement à l'itération suivante d'une instruction for ou while en sautant les instructions jusqu'au end.

1.5 Les fonctions

Définir de nouvelles fonctions dans Scilab se fait au moyen de la construction syntaxique suivante (à écrire dans un fichier de SciNotes) :

```
function [y1, y2,...] = <nom-de-fonction> (x1, x2,...)
<instructions>
endfunction
```

Dans la définition de la fonction on précise bien sûr le nom de la fonction, la liste de ses arguments d'entrée `(x1, x2,...)`, et la liste des arguments de sortie `(y1, y2,...)`. L'appel d'une fonction se fait de la façon suivante :

```
<nom-de-fonction> (<expr1>, <expr2>,...).
```

Lors de l'appel de la fonction les expressions fournies en arguments sont d'abord évaluées et leurs valeurs sont passées à l'environnement d'exécution de la fonction sous les noms `<x1>, <x2>,...`

Les arguments des fonctions sont donc passés par valeurs quel que soit leur type.

Les variables utilisées dans une fonction sont locales : elles sont effacées en fin d'exécution, et n'apparaissent pas dans le Navigateur de variables. Au contraire, les variables créées dans la console ou dans un programme sont appelées variables globales : on peut les utiliser et les modifier aussi bien dans la console que dans un programme.

Exemple 1.5 *: calcul de la factorielle d'un entier naturel. Cette fois, on veut créer une fonction fact de telle sorte que la commande fact(n) exécutée dans la console donne en*

réponse la factorielle de n.

```
function [J]=fact(n)
    J=1;
    for i= 1:n
        J=J*i
    end
endfunction
```

1.6 Les graphiques avec scilab

Les fonctions graphiques de Scilab sont basées sur la construction et la manipulation d'entités graphiques (objets). Il est ainsi possible de changer **a posteriori** l'aspect d'un graphique en modifiant les propriétés graphiques du dessin obtenu. Ces modifications peuvent être réalisées par des instructions Scilab.

1.6.1 Les fenêtres graphiques

En plus de la fenêtre principale et de l'éditeur SciNotes, il existe un troisième type de fenêtres : les fenêtres graphiques. Ces fenêtres sont nommées Figure suivi d'un numéro qui correspond à leur ordre de création.

On peut ouvrir plusieurs fenêtres graphiques simultanément. De plus, à l'intérieur d'une même fenêtre, on peut superposer des graphiques en les traçant les uns après les autres.

L'onglet Édition d'une fenêtre graphique permet de modifier le titre, d'ajouter une légende etc ...

Le même onglet permet aussi d'effacer la figure (sans fermer la fenêtre). Pour effacer simultanément tout le contenu de toutes les fenêtres graphiques, on exécute la commande clf ("clear figure" en anglais) dans la console. Pour jongler avec plusieurs fenêtres graphiques on utilise les commandes suivantes :

•**drawnow()** : dessinez la figure courante

• **drawlater()** : utilisée pour ne pas exposition immédiatement

• **xdel** : supprime des fenêtres graphiques

• **xname** : change le nom de la fenêtre graphique courante

1.6.2 Modifier un tracé

La loupe  permet de faire un zoom. Pour effectuer un zoom en deux dimensions, cliquez sur l'icône et avec la souris créez un rectangle qui constituera la nouvelle vue agrandie. Pour effectuer un zoom en trois dimensions, cliquez sur l'icône et créez le parallélépipède qui constituera la nouvelle vue agrandie. il est également possible de zoomer en utilisant la molette de la souris.
Pour revenir à l'écran initial, cliquez sur l'autre loupe .
l'icône  permet de faire tourner la figure (particulièrement utile en 3D) avec des actions de clic droit qui seront guidées par des messages en bas de la fenêtre graphique.
Pour des modifications plus précises, cliquez sur **Édition** > **Propriétés** de la figure ou **Propriétés des axes** et laissez-vous guider (cette option n'est pas encore disponible sous Mac Os).

1.6.3 L'instruction plot

Cette instruction, qui se veut une émulation de la fonction Matlab de même nom, est assez simple à utiliser. Voici quelques exemples :

```
x = linspace (-1, 1, 70);
y = x.^3;
plot(x, y)
```

Rajoutons maintenant une autre courbe :

```
yter = 2*y;
plot (x, yter)
```

Il est possible d'afficher les deux courbes simultanément, en précisant aussi la couleur utilisée :

```
drawlater()
clf() // pour effacer la fenêtre courante
plot(x, y, "b", x, yter, "r")
xtitle(" Les courbes ", "x", "y") // un titre plus une légende pour les deux axes
legend(x^3, 2x^3) // une légende pour les courbes
xgrid(15) //
drawnow();
```

et on obtient la figure suivante :

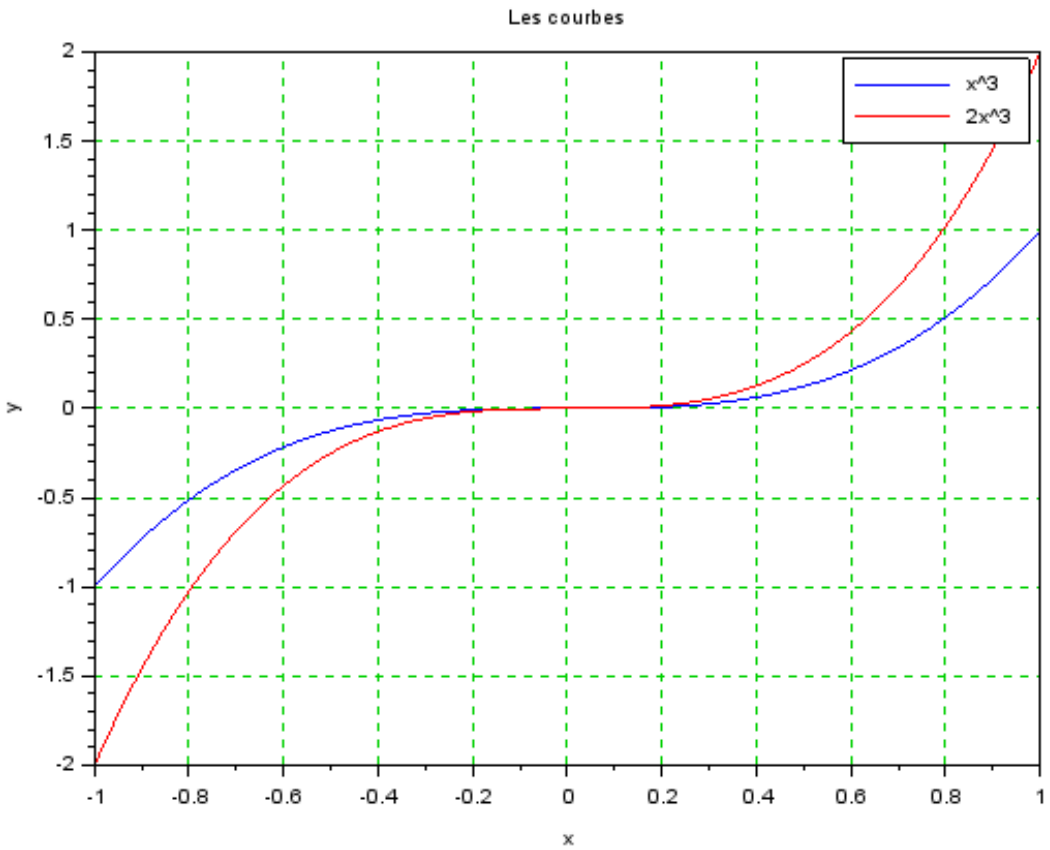


FIG 1.6.3 : les fonctions x^3, $2x^3$

Pour afficher simultanément plusieurs courbes, l'instruction prend la forme :

```
plot(x1, y1 [,style1], x2, y2 [,style2],...),
```

où **style** est une chaîne de caractères précisant la couleur. En rajoutant d'autres caractères on peut aussi spécifier le type du trait (trait plein, traits pointillés) ou bien encore le symbole utilisé pour dessiner les points (x_i, y_i).

1.6.4 L'instruction plot2d

Cette instruction est plus difficile à utiliser que `plot` mais elle permet de spécifier certains paramètres (réglages concernant l'échelle par exemple) sans avoir à utiliser le handle du système d'axes du graphique.
La syntaxe générale a la forme : `plot2d(X, Y <,opt_arg>*)`
où :

- `X` et `Y` sont respectivement la matrice des abscisses et celle des ordonnées, la $k^{\text{éme}}$ courbe étant constituée par les $k^{\text{éme}}$ colonnes de `X` et `Y` , d'autre part, si toutes les courbes ont les mêmes abscisses, on a pas à dupliquer le vecteur des abscisses : par exemple pour afficher `x, y1, x, y2, x, y3`, on utilisera `plot2d (x, [y1 y2 y3],...)` si bien sûr tous ces vecteurs sont des vecteurs colonnes.
- `<,opt_arg>*` désigne l'éventuelle séquence des arguments optionnels **opt arg** prenant la forme : **Mot- clé = valeur**
  la séquence pouvant être mise dans n'importe quel ordre :

```
clf();
x=[0 : 0.1: 2*%pi]';
plot2d([x  x  2*x], [sin(x)  2*sin(x)  cos(x)], style=[-1  20  15])
legend("sin(x)"," 2sin(x)", "cos(x)")
legend("Les courbes","x","y")
```

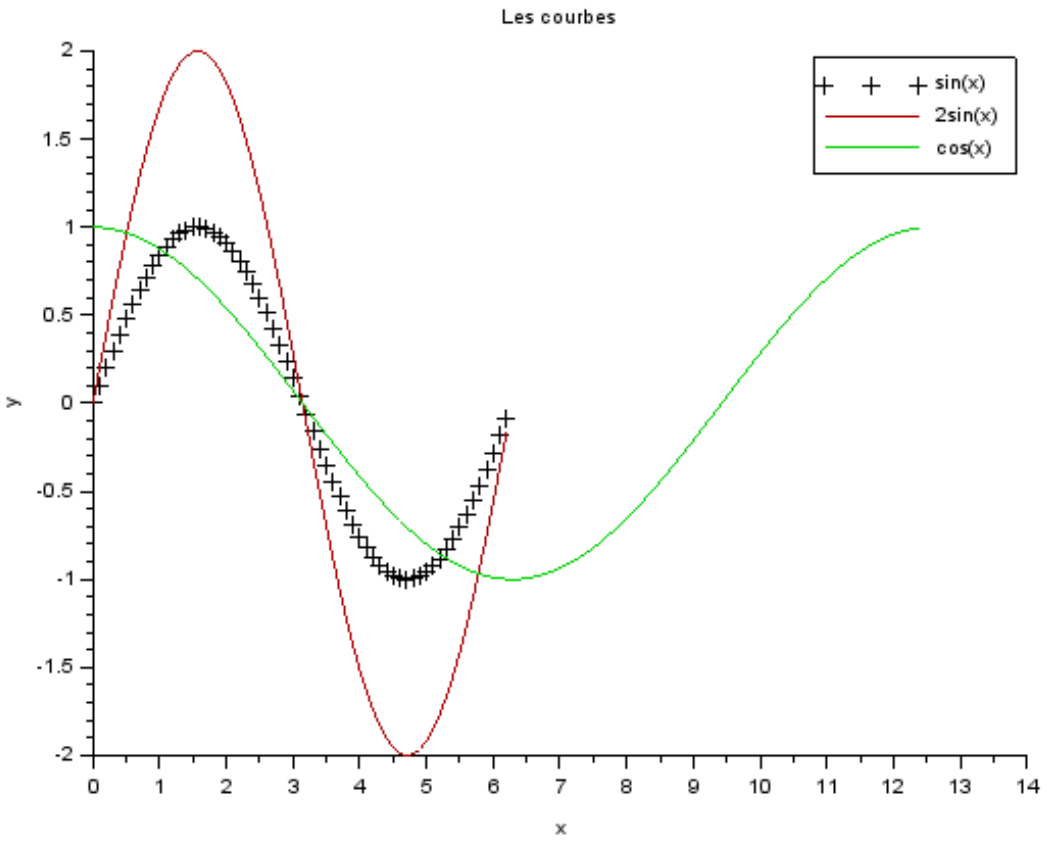


FIG 1.6.4 : Exemple d'utilisation de plot2d.

1.6.5 L'instruction plot3d

L'instruction générique pour dessiner des surfaces est `plot3d`. Avec une représentation de votre surface par facettes, on peut définir une couleur pour chaque facette. Depuis la version 2.6, on peut aussi, pour des facettes triangulaires ou quadrangulaires, spécifier une couleur par sommet et le rendu de la facette est obtenu par interpolation des couleurs définies aux sommets. L'appel le plus simple de plot3d est `plot3d (x, y, z)` où `x` est un vecteur (colonne) à n_x composantes, `y` un vecteur à n_y composantes et `z` une matrice à n_x lignes et n_y colonnes.

Exemple 1.6 *:*

```
//le multiple trace employer des couleurs
t = [0 : 0.3 : 2*%pi]';
z = sin(t)*cos(t');
//la même parcelle de terrain en utilisant des facettes calculées par genfac3d
[xx, yy, zz] = genfac3d(t, t, z);
plot3d([xx  xx], [yy  yy], list([zz  zz+4], [4*ones(1, 400)  5*ones(1, 400)]))
```

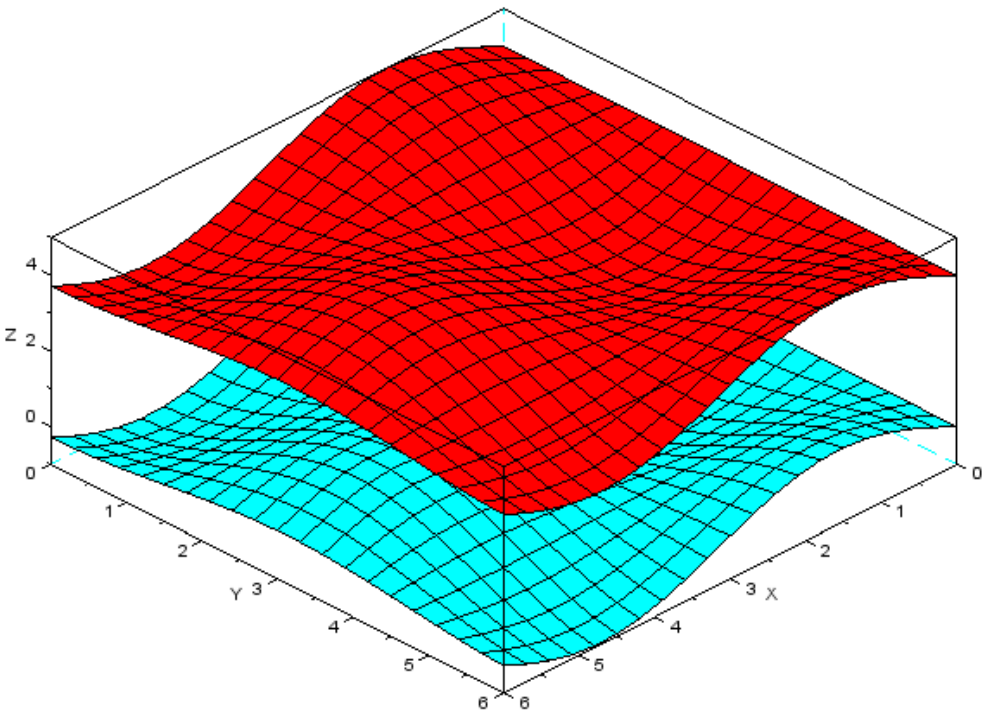


FIG 1.6.5 : Exemple d'utilisation de plot3d.

Le graphique de la sphère aurait plus être aussi généré directement sous formes de facettes grâce à la fonction **eval3dp**.

1.7 L'aide en ligne

Le menu d'aide Help est particulièrement important. Ce menu donne de façon interactive (en français ou en anglais) les propriétés de chacune des instructions, symboles, fonctions,

procédures Scilab prédéfinies. Pour accéder à l'aide en ligne, cliquez sur **? > aide Scilab** dans la barre de menus, ou tapez dans la console :

−−> help

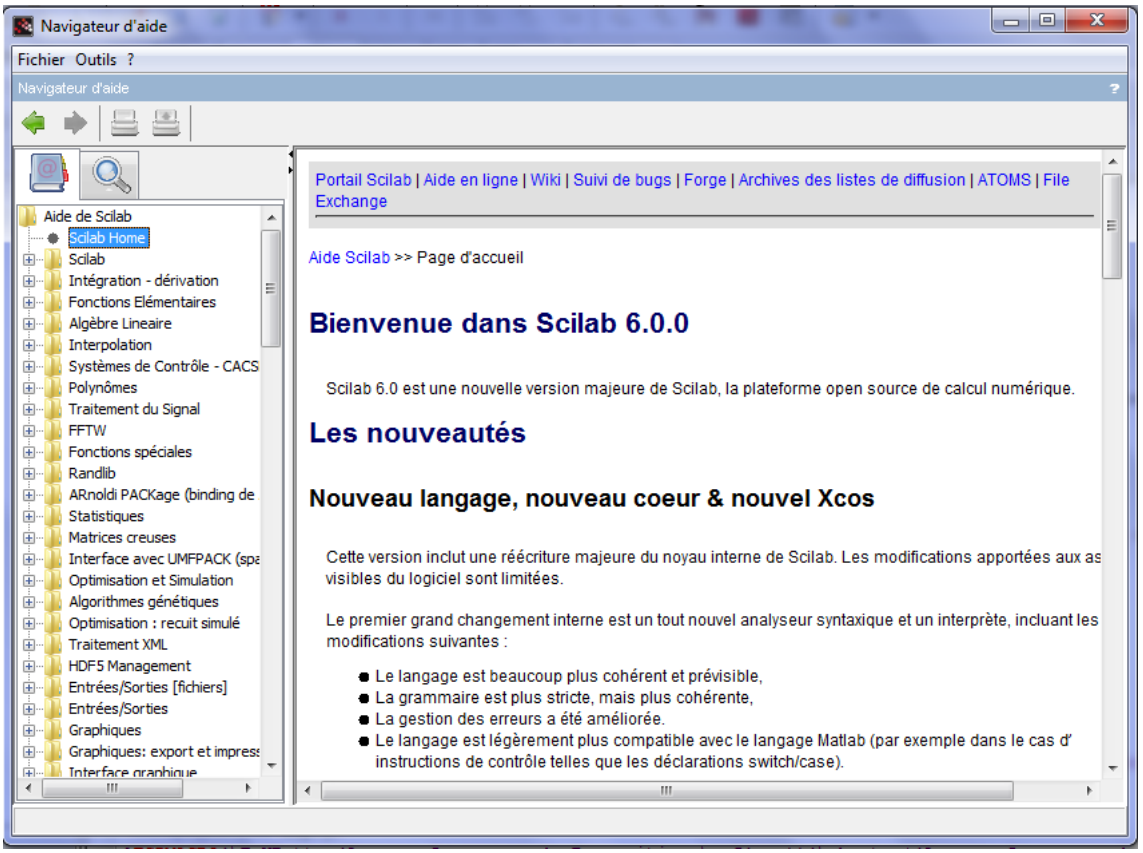


FIG 1.7 : La fenêtre d'aide

Pour obtenir de laide sur des fonctions, tapez dans la console help et le nom de la fonction souhaitée. Par exemple :

−−> help sin

affichera l'aide de la fonction sin (sinus).

Chapitre 2

Les équations aux dérivées partielles linéaires classiques

2.1 Introduction

La modélisation d'un problème réel la plupart utilise les lois de la physique (mécanique, thermodynamique, électromagnétisme, acoustique, etc. . .), ces lois sont, généralement, écrites sous la forme de bilans qui se traduisent mathématiquement par des Équations Différentielles Ordinaires ou par des Équations aux Dèrivées Partielles.
Les équations aux dérivées partielles interviennent dans beaucoup domaines : en chimie pour modéliser les réactions, en économie pour étudier le comportement des marchés et en finance pour étudier les produits dérivées (options et obligations).
Les équations aux dérivées partielles sont un sujet de recherche très actif en mathématiques et elles sont à l'origine de la création de beaucoup de concepts mathématiques.
Dans la plupart des cas il est très difficile, voir impossible d'exhiber les solutions d'une équation aux dérivées partielles. Dans certains cas on arrive à montrer que le problème est bien posé (c'est-à-dire qu'il admet une solution unique) et on peut, parfois, calculer des approximations numèriques des solutions.

2.2 Rappels sur la dérivation partielle

Fonction Scalaires

Soit u est une fonction définie sur Ω dans $\mathbb{R}$. A l'aide d'un repère on peut écrire un vecteur $x \in \Omega$ comme $(x_1, x_2, \ldots, x_n)$ où les x_i sont des réels.

Définition 2.1 *: La i-éme dérivée partielles de u au point $(x_1, x_2 \ldots, x_n)$ est définie par :*

$$\frac{\partial u}{\partial x_i}(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n) = \lim_{h \to 0} \frac{u(\bar{x}_1, \ldots, \bar{x}_i + h, \ldots, \bar{x}_n) - u(\bar{x}_1, \ldots, \bar{x}_{i-1}, \bar{x}_i, \bar{x}_{i+1} \ldots, x_n)}{h}$$

si elle existe. Un concept important est celui de gradient :

Définition 2.2 *:*
Si toutes les dérivées partielles de u sont définies en un point de l'espace $\mathbb{R}^n$ Alors on appelle gradient de u au point $x \in \Omega$ est donnée par :

$$\nabla u(x) = \left(\frac{\partial u}{\partial x_1}(x), \frac{\partial u}{\partial x_2}(x), \ldots, \frac{\partial u}{\partial x_n}(x) \right).$$

Définition 2.3 *: Soit $v = (v_1, v_2, \ldots, v_n)$ un vecteur non nule de $\mathbb{R}^n$ la dérivée directionnelle de u dans la direction du vecteur v au point $x \in \Omega$ est définie par :*

$$D_v u(x) = \lim_{h \to 0} \frac{u(x + hv) - u(x)}{h}.$$

Si toutes les dérivées partielles de u sont définies en un point $x \in \Omega$ alors

$$D_v u(x) = \nabla u(x).v$$

où "." est le produit scalaire de deux vecteurs. La dérivée partielle d'une fonction réelle est une fonction réelle, on peut ainsi définir facilement les dérivées d'ordre supérieure :

Définition 2.4 *: La dérivée partielle de la fonction u d'ordre j par rapport aux $x_{i1}, \ldots x_{ij}$ est définie par :*

$$\frac{\partial^j u}{\partial x_{i1}, \ldots, x_{ij}} = \frac{\partial}{\partial x_{i1}} \left(\frac{\partial^{j-1} u}{\partial x_{i2}, \ldots, x_{ij}} \right).$$

Un opérateur differentiel d'ordre 2 trés important est le Laplacien il intervient dans nombreuses équations aux dérivées partielles

Définition 2.5 *: Le Laplacien d'une fonction de N variable u est*

$$\Delta u(x) = \sum_{i=1}^{N} \frac{\partial^2 u}{\partial x_i^2}(x).$$

Fonction vectorielle

Une fonction vectorielle est une fonction à valeur en $\mathbb{R}^n$, on va étudier le cas particulier des fonctions de $\mathbb{R}^n$ en $\mathbb{R}^n$. Pour ces fonctions on définit une fonction scalaire, c'est un opérateur différentiel d'ordre 1 qu'on appelle la divergence.

Définition 2.6 *: La divergence de la fonction u de $\Omega \subset \mathbb{R}^n$ dans $\mathbb{R}^n$ est définie par :*

$$div(u(x)) = \sum_{i=1}^{n} \frac{\partial u_i}{\partial x_i}$$

la divergence est parfois notée comme :

$$divu = \nabla.u$$

Formellement l'opérateur divergence applique à un champ vecteur peut s'interpréter comme produit scalaire du vecteur nabla $\nabla = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \ldots, \frac{\partial}{\partial x_n}\right)^t$ par le vecteur $u(x) = (u_1(x), \ldots, u_n(x))$:

$$\nabla.u(x) = \sum_{i=1}^{n} \frac{\partial u_i}{\partial x_i}(x)$$

Remarque 2.1 *Si u une fonction de $\Omega \subset \mathbb{R}^n$ dans $\mathbb{R}$ alors*

$$\Delta u(x) = div(\nabla u(x))$$

Théorème 2.1 *: (Formule d'Ostrogradsky)*
Soit Ω un ouvert borne $\mathbb{R}^n$ de classe C^1 et Γ son bord soit F une fonction de classe $C^1(\overline{\Omega})$ à valeur dans $\mathbb{R}^n$ alors

$$\int_{\Omega} div(F(x))\, dx = \int_{\Gamma} F(x)\eta(x)\, d\Gamma,$$

Où η vecteur normal.

2.3 Équations aux dérivées partielles linéaires

Soit u une fonction définie sur Ω, à valeurs dans $\mathbb{R}$ et suffisamment régulière pour que les expressions qui suivent aient un sens.

Une équations aux dérivées partielles (E.D.P.) est une relation entre une fonction de plusieurs variables (réelles) u et un nombre fini des dérivées partielles.

$$F(x, u, \frac{\partial u}{\partial x_1}, \ldots, \frac{\partial u}{\partial x_n}, \frac{\partial u}{\partial x_1^2}, \ldots, \frac{\partial^\alpha u}{\partial x_n^m}) = 0, \quad x \in \Omega. \tag{2.1}$$

On dit que u est solution de l'équation aux dérivées partielles dans Ω si, après substitution de u et de ses dérivées partielles, F s'annule pour tout $x \in \Omega$.

L'ordre $m \in N$ d'une équation aux dérivées partielles est celui de la dérivée partielle d'ordre le plus élevé.

Un système d'équations aux dérivées partielles est formé de plusieurs équations aux dérivées partielles impliquant une ou plusieurs fonctions inconnues u_i.

Une équation aux dérivées partielles est linéaire si F est linéaire par rapport à u et ses dérivées partielles (l'ensemble de ses solutions forme un espace vectoriel). Si m est l'ordre de l'équation aux dérivées partielles linéaires, l'équation est de la forme :

$$\sum_{|\alpha| \leq m} A_\alpha(x) D^\alpha u(x) = B(x),$$

où $A_\alpha(x)$ sont des fonction $\mathcal{C}^1(\Omega)$ dans $\mathbb{R}$ pour $|\alpha| \leq m$, et la fonction $B(x)$ est continue. Si $B = 0$ on a une équation homogène et $Lu = \sum_{|\alpha| \leq m} A_\alpha D^\alpha u$ est un opérateur différentiel linéaire.

Propriété

- si u_1 et u_2 sont deux solutions d'une équation aux dérivées partielles linéaire homogène, alors pour α_1 et α_2 des réels quelconques, $\alpha_1 u_1 + \alpha_2 u_2$ est aussi solution ;
- si u_h est solution de l'équation linéaire homogène et u_p est solution de l'équation linéaire non homogène, alors $u_h + u_p$ est solution de l'équation complète.

La solution d'une équation aux dérivées partielles d'ordre m dépend en général de m fonctions arbitraires de $n - 1$ variables.

La solution générale d'une équation aux dérivées partielles est celle qui permet de trouver toutes les solutions de l'équation (sauf des cas de solutions singulières) en donnant des valeurs particulières aux fonctions arbitraires.

2.3.1 Conditions aux bord (conditions aux limites)

La recherche d'une solution d'une équation aux dérivées partielles ne peut se faire que lorsque cette solution est bien déterminée par des conditions fixées sur la frontière Γ du domaine Ω à l'intérieur duquel cette solution est recherchée.
Les conditions aux limites les plus courantes pour un problème de résolution d'une E.D.P. sont énoncées ci-dessous.

1) Conditions de Dirichlet où u est donnée sur le bord de Ω :

$$u|_\Gamma = g \quad \forall x \in \Gamma$$

2) Conditions de Neumann où la dérivée normale de u est donnée :

$$\frac{\partial u}{\partial n}|_\Gamma = g \quad \forall x \in \Gamma$$

3) Conditions de Robin ou mixtes :

$$\alpha(x)u(x) + \beta(x)\frac{\partial u}{\partial n} = g \quad sur \quad \Gamma$$

où $\alpha(x)$ et $\beta(x)$ étant deux fonctions régulières données sur Γ non nulles simultanément.

Si $g = 0$ on a des conditions homogènes au bord .

4) Conditions initiales :

Celle-ci précise qu'en chaque point de Γ, u et $\frac{\partial u}{\partial t}$ sont données. On l'utilise généralement dans le cas des équations hyperboliques, on parle aussi de conditions de Cauchy .

2.3.2 Probléme bien posé

Considérons une équation aux dérivées partielles sur un domaine Ω avec éventuellement des conditions auxiliaires sur la solution, on dit que le problème est bien posé (au sens d'Hadamard) si on a :

- existence d'une solution du problème .
- unicité de cette solution.
- stabilité par rapport aux données du problème.

2.3.3 Notions de solution

Soit u la solution d'un problème bien posé. Dans ce qui précède on n'a pas précisé la régularité de u, on a seulement supposé que u était suffisamment différentiable. Pour que les équation aient un sens, par exemple $u \in C^m(\Omega)$, si u est solution d'un équation aux dérivées partielles d'ordre m.

On dit alors que u est une solution au sens fort de l'équation aux dérivées partielle. Dans beaucoup de cas un problème n'admet pas de solutions régulières : l'équation des Ondes avec conditions initiales non continues, les loi des Ondes de choc.

Pour que le problème soit bien posé il faut élargir l'ensemble des u, par exemple admettre des solutions non continues. On doit transformer le problème de façon à lui garder un sens pour des fonctions non différentiable, on parle de formulation faible du problème.

Pour une solution forte il y a équivalence entre la formulation faible et le problème initial. Une solution du problème faible est appelée solution faible ou solution généralisée. Il est souvent plus facile de trouver une solution faible et de montrer a posteriori qu'elle est réguliére, c'est -à-dire que l'ou a bien une solution forte.

2.4 Modélisation

La modélisation est l'étape qui permet de mettre en équation un problème donné. Cette étape nécessite d'identifier les mécanismes mis en jeu et de formuler des hypothéses simplificatrices permettant à partir des principes fondamentaux de la physique, la chimie, l'économie . . . , d'obtenir une ou pleusieurs équations permettant de modéliser le problème (ie de rendre compte des phénomènes observés), de plus, il faut associe des conditions aux limites et des conditions initials pour obtenir le model complet. Dans la suite on présente un exemple qui va modéliser par une équation aux dérivées partielles elliptique :

Exemple d'une équation aux dérivées partielles elliptique

On va montrer comment, à partir des lois de la thermodynamique on peut déduire une équation aux dérivées partielles dont les solutions décrivent la température d'un corps en équilibre thermique.

Soit Ω un ouvert de $\mathbb{R}^3$ qui décrit la position dans l'espace du corps, soit $T(x)$ la fonction de Ω à valeurs dans $\mathbb{R}^3$ qui décrit la température en chaque point.
Soit $\Phi(x) = (\Phi_1(x), \Phi_2(x), \Phi_3(x))$ le champ vectoriel qui décrit le flux de chaleur au point x. Soit $f(x)$ la densité volumique de chaleur reçue au point x.
L'équilibre thermique d'un petit volume $\omega \subset \Omega$ signifie que le flux de chaleur qui traverse la surface de ω est égalà la quantité de chaleur reçue, si qui se traduit par :

$$\int_{\partial\omega} \Phi.\eta ds = \int_{\omega} f(x)\, dx,$$

où $\eta = \eta(x)$ est la normale extérieure unitaire à la surface $\partial\omega$ au point $x \in \partial\omega$. En utilisant la formule de d'Ostrogradsky, permet de transformer l'intégrale de surface en intégrale de volume, on conclut que

$$\int_{\partial\omega} \Phi(x).\eta(x)\, ds = \int_{\omega} div\Phi\, dx,$$

donc :

$$\int_{\omega} div\Phi\, dx = \int_{\omega} f(x)\, dx. \tag{2.2}$$

Cette équation est vérifiée pour tout petit volume $\omega \subset \Omega$, nous obtenons donc la relation valable en tout point du domaine Ω peut être modélisé par l'équation :

$$div\Phi(x) = f(x) \quad \forall\, x \in \Omega. \tag{2.3}$$

L'équation (2.3) est la forme locale (ou différentielle) de la conservation de la chaleur. Il s'agit d'une loi fondamentale de la physique. Il reste maintenant à relier le flux de chaleur Φ à la température T. C'est ce que l'on appelle une loi de comportement. Il est communément admis (dans un régime de températures pas trop élevées) que le flux de chaleur est proportionnel au gradient de la température. Il s'agit de la loi de Fourier. Dans un milieu supposé hétérogéne (la loi dépend du point considéré) et anisotrope (les directions de l'espace ne sont pas équivalentes), cette loi suppose l'existence d'un tenseur $\mu(x)$, appelé tenseur de conductivité thermique qui dépend exclusivement de la nature du matériel, ce qui s'écrit :

$$\Phi(x) = -\mu(x)\nabla T(x). \tag{2.4}$$

En remplaçant (2.4) dans 2.3 on obtient l'èquation

$$-\, div(\mu(x)\nabla T(x)) = f(x), \quad dans \;\; \Omega \tag{2.5}$$

qui est équivalent, dans le cas où $\mu(x)$ est constant, à

$$-\mu\Delta T = f \quad dans \;\; \Omega,$$

Pour que ce problème soit bien posé on doit connaître la température ou le flux de chaleur dans frontiére du domaine, on considére deux types de conditions :

- Une condition de Dirichlet où on impose la température sur le bord du domaine :

$$T(x) = T_0(x) \quad \forall\, x \;\in\; \partial\Omega,$$

  où T_0 est une fonction donnée sur $\partial\Omega$.
- Une condition de Neumann où on impose un flux de chaleur à travers la frontiére :

$$-\mu(x)\frac{\partial T}{\partial \eta}(x) = \theta_0(x) \quad \forall\, x \;\in\; \partial\Omega,$$

  où θ_0 est une fonction donnée sur $\partial\Omega$.

2.5 Quelques résolutions théoriques d'E.D.P.

Notations- Soit $\Omega \subset \mathbb{R}$. de frontière Γ. On note

$$\mathbf{Q} = \Omega\times]0,+\infty[, \quad \Sigma = \Gamma\times]0,+\infty[.$$

préliminaire

Convolution de deux fonctions et propriétés élémentaires

Définition 2.7 *Soient deux fonctions f, $g : \mathbb{R} \mapsto \mathbb{R}$ (ou $\mathbb{C}$). On appelle convolution de f et g la fonction que l'on note $f * g$, si elle existe, définie par*

$$(f * g)(x) = \int_{\mathbb{R}} f(x-y)g(y)\,dy = \int_{\mathbb{R}} g(x-t)g(t)\,dt,$$

c'est un application linéaire, et symétrique.

Remarque 2.2 *Une des caractéristiques de la convolution est un effet régularisant. Ici le produit de convolution de deux fonctions discontinues donne une fonction continue.*

Exemple 2.1 *Convolution d'une fonction par* $\frac{1}{2h}\chi_{[-h,h]}$.
On pose $g_h = \frac{1}{2h}\chi_{[-h,h]}$. *Ona*

$$\int_{\mathbb{R}} g_h(x)\,dx = 1, \quad \forall\, h > 0,$$

Le produit de convolution

$$(f * g_h)(x) = \int_{\mathbb{R}} f(x-y)\frac{1}{2h}\chi_{[-h,h]}\,dy = \int_{-h}^{h} f(x-y)\,dy.$$

Quand h *est petit,* $f * g_h(x)$ *est proche de* $f(x)$ *pourvu que* f *soit continue.*

Impulsion de Dirac δ_0

Reprenons l'exemple précédent de la fonction g_h. Que devient le produit de convolution $f * g_h$ quand h tend vers 0 ?. L'intervalle sur lequel on fait la moyenne de f autour de x devient de plus en plus petit, et si f est une fonction continue alors

$$\lim_{h\to 0}(f * g_h)(x) = f(x).$$

On se pose alors la question, puisque seule g_h dépend de h de ce qu'est la limite $\lim\limits_{h\to 0} g_h$. Si cette limite a un sens, alors la fonction limite, que l'on note $\delta_0 = \lim\limits_{h\to 0} g_h$ est l'élément neutre pour la convolution

$$\delta_0 * f = f.$$

Et comme $\int_{\mathbb{R}} g_h(x)\,dx = 1,\ \forall\, h$, on s'attend à ce que $\int_{\mathbb{R}} \delta_0(x)\,dx = 1$.
En conclusion, la notion de limite dans $\delta_0 = \lim\limits_{h\to 0} g_h$ ne peut pas être une limite ponctuelle et δ_0 n'est pas une fonction. On parle de distribution, qui est un concept généralisant les fonctions et d'une limite au sens des distributions.
La masse de Dirac en 0, notèe δ_0 sera pour nous la fonction en un sens généralisé qui vérifie la propriété suivante :

$$\int_{\mathbb{R}} \delta_0(x) f(x)\,dx = f(0), \quad \text{pourvu que f soit continue en 0.}$$

Par conséquence

$$\int_{\mathbb{R}} \delta_0(x)\,dx = 1,$$

$$\delta_0 * f = f, \quad \text{pourvu que f soit continue.}$$

Théorème 2.2 *(Fubini dans le plan).*

Soit $f : D \subset \mathbb{R}^2 \to \mathbb{R}$ continue sur un domaine élémentaire du type

$$D = \left\{(x,\, y) \in \mathbb{R}^2, a \leq x \leq b, c \leq y \leq d\right\},$$

avec $a \leq b$ et $c \leq d$.

Alors la fonction $g : x \in [a, b] \to \int_c^d f(x,\, y)\, dy$ est continue et on a

$$\int_a^b g(x)\, dx = \int_a^b \left(\int_c^d f(x,\, y)\, dy \right) dx.$$

2.6 L'équation de transport

L'équation de transport est le prototype des équations aux dérivées partielles linéaires du premier ordre dans le demi-plan. On cherche une fonction u à valeurs réelles de classe $\mathcal{C}^1$ (solution classique) définie sur $\mathbf{Q}$ qui vérifie :

$$\begin{cases} \frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 & sur \quad \mathbf{Q} \\ u(x,\, 0) = u_0(x) & sur \quad \Omega \end{cases} \tag{2.6}$$

ici la constante $c \in \mathbb{R}$ est donné : c'est par définition la vitesse de propagation associée à l'équation.

Soit $y \in \Omega$, posons pour tout $t \in \mathbb{R}_*^+$, $\gamma(t) = y + tc$, alors γ est une application de classe $\mathcal{C}^1$ de $\mathbb{R}_*^+$ dans $\mathbb{R}$ vérifiant

$$\frac{\partial \gamma}{\partial t}(t) = c.$$

Définition 2.8 *: L'ensemble $\{(t,\, \gamma(t))/t \in \mathbb{R}_*^+\}$ est une droite de $\mathbf{Q}$, appelée "courbe caractéristique issue de y ".*

Soit $u \in \mathcal{C}^1(\mathbf{Q})$ solution de L'équation de transport. L'application $t \mapsto u(t,\, \gamma(t))$ est de classe $\mathcal{C}^1$ sur $\mathbb{R}_*^+$ (comme composée des applications u et $t \mapsto (t,\, \gamma(t))$, toutes deux de classe $\mathcal{C}^1$), et on a

$$\begin{aligned} \frac{d}{dt}u(t,\, \gamma(t)) &= \frac{\partial}{\partial t}u(t,\, \gamma(t)) + \frac{\partial}{\partial x}u(t,\, \gamma(t))\frac{d\gamma}{dt}(t) \\ &= \frac{\partial}{\partial t}u(t,\, \gamma(t)) + c\frac{\partial}{\partial x}u(t,\, \gamma(t)) \\ &= (\frac{\partial}{\partial t}u + c\frac{\partial}{\partial x}u)(t,\, \gamma(t)) \\ &= 0 \end{aligned}$$

Théorème 2.3 *Soit $u_0 \in \mathcal{C}^1(\Omega)$. Le problème de Cauchy (2.6) d'inconnue u en (x, t), admet une unique solution $u \in \mathcal{C}^1(\mathbf{Q})$, elle est donnée par :*

$$u(t,\, x) = u_0(x - ct) \quad \forall\, (t,\, x) \in \mathbf{Q}.$$

Cette formule explicite pour la solution de l'èquation de transport.

Preuve. Si u est une solution de classe $\mathbf{C}^1(\mathbf{Q})$ de l'équation de transport, elle est constante le long des courbes caractéristiques, donc

$$u(t, y + tc) = u(0, y) = u_0, \quad \forall\, t > 0,\, y \in \Omega.$$

En posant $x = y + tc$, on trouve donc que

$$u(t, x) = u_0(x - tc) \quad \forall\, t > 0,\, x \in \Omega.$$

Réciproquement, pour $u_0 \in \mathcal{C}^1(\Omega)$, l'application $(t,\, x) \mapsto u_0(x - tc)$ est de classe $\mathcal{C}^1$ sur $\mathbf{Q}$ D'autre part on a

$$\partial_x(u_0(x - tc)) = (\partial u_0)(x - tc),$$

tandis que

$$\begin{aligned} \partial_t(u_0(x - tc)) &= -c(\partial_x u_0)(x - tc)) \\ &= -c(\partial u_0)(x - tc) = -c(\partial_x(u_0(x - tc), \end{aligned}$$

ce qui montre que la fonction $u : (t, x) \mapsto u_0(x - tc)$ est bien une solution de l'équation de transport. ∎

Même si u_0 n'est pas dérivable, la formule

$$u(t,\, x) = u_0(x - tc),$$

garde un sens, et on souhaiterait pouvoir dire qu'elle définit encore une solution de l'équation de transport.

2.7 Équation de la chaleur

Soit Ω un ouvert borné de $\mathbb{R}$ de frontière Γ "assez régulière". On considère le problème suivant : étant donnè des fonctions $u_0 : \Omega \to \mathbb{R}$, et $f : \mathbf{Q} \to \mathbb{R}$ il s'agit de trouver une

fonction $u(x,\ t)$ solution de l'équation :

$$\begin{cases} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = f & sur \quad \mathbf{Q} \quad \text{(équation de la chaleur)} \\ u = 0 & sur \quad \Sigma \quad \text{(condition aux limites)} \\ u(x,\ 0) = u_0 & sur \quad \Omega \quad \text{(condition initiale)} \end{cases} \tag{2.7}$$

L'équations (2.7) décrivent l'évolution au cours du temps de la température u d'un milieu continu $\overline{\Omega}$ homogène et isotrope soumis à une source de chaleur f lorsque les constantes physiques sont prises égales à 1, la température étant fixée nulle au cours du temps sur la frontière Γ du milieu continu et la température initiale étant donnée égale à u_0.

Solution fondamentale de l'équation de la chaleur

Le problème (2.7) est beaucoup plus difficil que l'équation de transport, en général, on n'a pas de solution explicite. On peut néanmoins trouver une solution élémentaire. On a alors le résultat suivant

Proposition 2.1 *le problème (sans terme source)*

$$\begin{cases} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = 0 & sur \quad \mathbf{Q} \\ u_0 = \delta_0(x) \end{cases}$$

où δ représente une mesure de Dirac en $x = 0$, admet une solution unique

$$u(x,t) = \frac{1}{2\sqrt{\pi t}} e^{-\frac{x^2}{4t}}.$$

On not $\theta(x,t)$

Preuve. La fonction $\theta(x,t)$ est appelée distribution gaussienne autour de 0. Elle vérifie

$$\int_\Omega \theta(x,t)\, dx = 1,$$

ce qui est compatible avec la conservation de la quantité de chaleur. On peut le vérifier par le calcul suivant

$$\begin{aligned}
\int_\Omega \exp\left(-\frac{x^2}{4t}\right) dx &= \int_\Omega \exp\left(-\frac{x^2}{K}\right) dx \quad \text{où} \quad (K = 4t) \\
&= \left(\int_{\Omega^2} \exp\left(-\frac{x^2+y^2}{K}\right) dx\, dy\right)^{\frac{1}{2}} \\
&= \left(\int_0^{2\pi}\int_0^{+\infty} \exp\left(-\frac{r^2}{K}\right) r\, dr\, d\theta\right)^{\frac{1}{2}} \\
&= \left(2\pi K \int_0^{+\infty} e^{-t} \frac{\sqrt{t}}{2\sqrt{t}}\, dt\right)^{\frac{1}{2}} \\
&= \sqrt{\pi K}.
\end{aligned}$$

On remarque que la fonction $\theta(t,x)$ admet son maximum en 0 qui vaut $\theta(t,0) = \frac{1}{2\sqrt{\pi t}}$ et qui tend vers 0 lorsque t augmente.

La fonction $\theta(t,x)$ a des propriétes trés similairesà la fonction $g_h = \frac{1}{2h}\chi_{[-h,h]}$.

comme g_h la fonction $\theta(t,\, x)$ est d'intégrale 1 et concentrée autour de 0. Pour une fonction continue $f(x)$, l'intégrale $\int_\Omega f(x)\theta(t,x)\, dx$ est une moyenne pondérée autour de 0 de la fonction $f(x)$ et donc

$$\lim_{t\mapsto 0^+} \int_\Omega f(x)\theta(t,x)\, dx = f(0),$$

alors

$$\lim_{t\mapsto 0^+} \theta(x,t) = \delta_0.$$

La condition initiale est vérifiée.

Vérifions maintenant que $\theta(t,x)$ est une solution. Alors on a

$$\begin{aligned}
\frac{\partial\theta}{\partial t}(t,x) &= -\frac{1}{4\sqrt{\pi}} t^{-3/2} \exp\left(\frac{-x^2}{4t}\right) + \frac{1}{2\sqrt{\pi}} t^{-1/2} \left(\frac{x^2}{4t^2}\right) \exp\left(\frac{-x^2}{4t}\right) \\
&= \frac{1}{4\sqrt{\pi}} t^{-3/2} \exp\left(\frac{-x^2}{4t}\right)\left(-1 + \frac{x^2}{2t}\right)
\end{aligned}$$

et

$$\begin{aligned}\frac{\partial\theta}{\partial x}(t,x) &= -\frac{1}{2\sqrt{\pi}}t^{-1/2}\exp\left(\frac{-x^2}{4t}\right)\left(\frac{2x}{4t}\right)\\ \frac{\partial^2\theta}{x^2}(t,x) &= \frac{1}{4\sqrt{\pi}}t^{-3/2}\left(-\exp\left(\frac{-x^2}{4t}\right)+x\exp\left(\frac{-x^2}{4t}\right)\left(\frac{2x}{4t}\right)\right)\\ &= \frac{\partial\theta}{\partial t}(t,x)\end{aligned}$$

■

Proposition 2.2 *La solution de l'équation de la chaleur*

$$\begin{cases}\frac{\partial u}{\partial t}-\frac{\partial^2 u}{\partial x^2}=0 & sur \quad \mathbf{Q}\\ u_0=u_0(x) & sur \quad \Omega\end{cases}$$

est

$$u(t,x)=u_0(x)*\theta(t,x)=\frac{1}{2\sqrt{\pi t}}\int_\Omega \exp-\frac{(x-y)^2}{4t}u_0(y)\,dy$$

Preuve. Ona

$$\lim_{t\mapsto 0^+}u(t,x)=u_0(x)*\lim_{t\mapsto 0^+}\theta(t,x)=u_0(x)*\delta_0(x)=u_0(x),$$

et

$$\begin{aligned}\frac{\partial u}{\partial t}(t,x)&=u_0(x)*\frac{\partial\theta}{\partial t}(t,x),\\ \frac{\partial^2 u}{\partial x^2}=u_0(x)*\frac{\partial^2\theta}{\partial x^2}(t,x)&=u_0(x)*\frac{\partial\theta}{\partial t}(t,x)=\frac{\partial u}{\partial t}(t,x)\end{aligned}$$

■

2.8 Équation des ondes

Soit Ω un ouvert borné de $\mathbb{R}$ de frontière Γ "assez régulière". L'équation des ondes est un exemple modèle d'équation hyperbolique en Ω s'écrit :

$$\begin{cases}\frac{\partial^2 u}{\partial t^2}-c^2\frac{\partial^2 u}{\partial x^2}=0 & sur \quad \mathbf{Q}\\ u(x,\,0)=u_0(x) & sur \quad \Omega\\ \frac{\partial u}{\partial t}(x,\,0)=u_1(x) & sur \quad \Omega\end{cases} \tag{2.8}$$

elle modèlise la propagation d'une onde (acoustique, èlectromagnètique, etc...) dans un milieu infini qui n'est soumise à aucune force extérieure, pour chaque $t \geq 0$. La constante c est la vitesse de propagation de l'onde et les fonctions $u_0, u_1 : \Omega \to \mathbb{R}$ sont respectivement, l'état la vitesse initiale.

On établira tout d'abord un lien entre cette équation et l'équation de transport. Introduisons deux nouvelles fonctions inconnues, v_1 et v_2, qui sont liées à u par les égalités :

$$\begin{cases} v_1 = \frac{\partial u}{\partial t} \\ v_2 = c\frac{\partial u}{\partial x} \end{cases} \tag{2.9}$$

et considérons le système de deux équations aux dérivées partielles du premier ordre suivant :

$$\begin{cases} \frac{\partial v_1}{\partial t} - c\frac{\partial v_2}{\partial x} = 0 \quad sur \quad \mathbf{Q} \\ \frac{\partial v_2}{\partial t} - c\frac{\partial v_1}{\partial x} = 0 \quad sur \quad \mathbf{Q} \end{cases} \tag{2.10}$$

où on a fait abstraction des conditions initiales. En remplacent les expressions pour v_1 et v_2 dans (2.10) on voit que la première équation est équivalente à l'équation des ondes et la deuxième à l'équation de Schwarz :

$$\frac{\partial^2 u}{\partial x \partial t} = \frac{\partial^2 u}{\partial t \partial x}$$

qui est satisfaite par toute fonction de classe $\mathcal{C}^2$. Le changement de variables qu'on a pris n'est le seul possible, en effet, on aurait aussi pu prendre $v_1 = \frac{\partial u}{\partial t}$ et $v_2 = -c\frac{\partial u}{\partial x}$.

Le système (2.10) est composé de deux équations de transport couplées, de vitesse constante $-c$, on peut le récrire en forme matricielle :

$$\begin{bmatrix} \frac{\partial v_1}{\partial t} \\ \frac{\partial v_2}{\partial t} \end{bmatrix} + c \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial v_1}{\partial x} \\ \frac{\partial v_2}{\partial x} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.11}$$

ce qui équivaut à :

$$\frac{\partial U}{\partial t} + cA\frac{\partial U}{\partial x} \quad \text{où} \quad U = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \quad et \quad A = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$$

On notera que si on diagonalise la matrice A alors les deux équations de transport seront découplées, ce qui nous permettra l'utilisation des résultats qu'on a montré pour l'équation de transport à vitesse constante. Cherchons les valeurs propres de A, ceux-ci sont solution de l'équation :

$$P_A = \lambda^2 - 1 = 0$$

donc $\lambda_1 = 1$ et $\lambda_2 = -1$. Les vecteurs propres sont les solutions de :

$$x + y = 0$$

et de :

$$x - y = 0$$

on choisit alors les vecteurs propres de norme 1 : $(1/\sqrt{2})(1, -1)$ et $(1/\sqrt{2})(1, 1)$, ceci donne alors :

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

si on définit une nouvelle variable :

$$w = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Alors :

$$\begin{cases} w_1 = \frac{1}{2}(v_1 + v_2) \\ w_2 = \frac{1}{2}(-v_1 + v_2) \end{cases}$$

on note P l'inverse de la matrice :

$$\frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

donc

$$P = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

alors le système (2.11) s'écrite :

$$P\frac{\partial w}{\partial t} + cAP\frac{\partial w}{\partial x}$$

en multipliant par P^{-1} on obtient :

$$\frac{\partial w}{\partial t} + c \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \frac{\partial w}{\partial x} = 0$$

on écrit

$$\begin{cases} \frac{\partial w_1}{\partial t} + c\frac{\partial w_1}{\partial x} = 0 \\ \frac{\partial w_2}{\partial t} - c\frac{\partial w_2}{\partial x} = 0 \end{cases} \tag{2.12}$$

ceci nous montre tout de suite le besoin d'avoir deux conditions initiales.

On a ainsi obtenu deux équations de transport découplées à vitesse de propagation opposées.

La technique utilisée est similaire a celle qu'on utilise pour les équations différentielles ordinaires où on réduit une équation d'ordre 2 à un système de deux équations d'ordre 1. Pour $t = 0$ on a les conditions initiales $u(x,\,0) = u_0(x)$ et $\frac{\partial u}{\partial t}(x,\,0) = u_1(x)$, par définition de v_1 et v_2 ceci donne $v(x,\,0) = (u_1(x),\, cu_0^{'}(x))$. Passons en variables w :

$$\begin{cases} \frac{\partial w}{\partial t} + c \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \frac{\partial w}{\partial x} = 0 = \qquad (x,t) \in \mathbf{Q} \\ w(x,0) = \frac{1}{2} \begin{bmatrix} u_1(x) + cu_0^{'}(x) \\ -u_1(x) + cu_0^{'}(x) \end{bmatrix} \end{cases} \tag{2.13}$$

alors les solution de (2.13) sont :

$$\begin{cases} w_1(x,\,t) = w_{1,0}(x - ct) = \frac{1}{2}(u_1(x - ct) + cu_0'(x - ct)) \\ w_2(x,\,t) = w_{2,0}(x + ct) = \frac{1}{2}(-u_1(x + ct) + cu_0'(x + ct)) \end{cases}$$

en variables initiales on a :

$$\begin{aligned} v_1(x,t) = \tfrac{\partial u}{\partial t} &= \tfrac{1}{2}(u_1(x - ct) + cu_0'(x - ct)) - \tfrac{1}{2}(-u_1(x + ct) + cu_0'(x + ct)) \\ v_2(x,t) = c\tfrac{\partial u}{\partial x} &= \tfrac{1}{2}(u_1(x + ct) + cu_0'(x + ct)) + \tfrac{1}{2}(-u_1(x - ct) + cu_0'(x - ct)) \end{aligned}$$

On a résolu un problème en se ramenant à une situation connue.

Théorème 2.4 *(Formule de d'Alembert).*
Pour tout $u_0 \in \mathcal{C}^2(\Omega)$ et tout $u_1 \in \mathcal{C}^1(\Omega)$ l'équation des ondes (2.8) admet une solution unique $u \in \mathcal{C}^2(\mathbf{Q})$. Celle-ci est donnée par la formule de d'Alembert suivant :

$$u(t,x) = \frac{1}{2}(u_0(x + ct) + u_0(x - ct)) + \frac{1}{2c}\int_{x-ct}^{x+ct} u_1(y)\,dy.$$

Preuve. On a déjà vu que toute solution en écrit sous la forme :

$$u(x,t) = F(x + ct) + G(x - ct)$$

avec les conditions initiales on a :

$$u(x,0) = u_0(x) = F(x) + G(x)$$

et

$$\frac{\partial u}{\partial t}(x,0) = u_1(x) = c(F'(x) - G'(x))$$

D'après la dérivation on obtient le système :

$$\begin{cases} F'(x) + G'(x) = u_0'(x) \\ F'(x) - G'(x) = \frac{u_1(x)}{c} \end{cases}$$

qui a pour solution :

$$\begin{cases} F'(x) = \frac{cu_0'(x)+u_1(x)}{2c} \\ G'(x) = \frac{cu_0'(x)-u_1(x)}{2c} \end{cases}$$

ce qui donne :

$$\begin{cases} F(x) = c_1 + \frac{1}{2}u_0(x) + \frac{1}{2c}\int_0^x u_1(y)\,dy \\ G(x) = c_2 + \frac{1}{2}u_0(x) - \frac{1}{2c}\int_0^x u_1(y)\,dy \end{cases}$$

A l'aide de condition initiale à déterminer les constantes d'intégration :

$$u_0(0) = u(0,0) = F(0) + G(0) = c_1 + c_2 + u_0(0)$$

donc :

$$c_1 + c_2 = 0$$

et en sommant $F(x+ct)$ et $G(x-ct)$ on obtient le résultat. ∎

2.9 Équations de Laplace et Poisson

On s'intéresse à l'équation aux dérivées partielles linéaire d'ordre deux

$$-\Delta u = f,$$

c'est l'équation de Poisson, où Δ est l'opérateur de Laplace et f est une fonction continue donnée.

Cette équation est très importante à la fois en physique et en mathématiques.

Si $f = 0$, on a l'équation de Laplace et on dit que les solutions qui doivent être des fonctions de classe $\mathcal{C}^2$ sont appelées fonctions harmoniques.

Présentation du problème, existence et unicité

Soit Ω un ouvert borné de $\mathbb{R}^n$, de bord Γ régulier et $u \in \mathcal{C}^2(\bar{\Omega})$ On considère le problème de Poisson en dimension avec condition aux limites de Dirichlet homogène :

$$\begin{cases} -\Delta u(x) = f(x) & sur \quad \Omega \\ u_0 = u_1 = 0, & sur \quad \Gamma \end{cases} \tag{2.14}$$

où $f \in \mathcal{C}(\Omega)$

Proposition 2.3 *pour tout $f \in \mathcal{C}(\Omega)$, il existe un unique $u \in \mathcal{C}^2(\bar{\Omega})$ solution (classique) de (2.14) et donnée par*

$$u(x) = \int_\Omega G(x,y) f(y)\, dy, \quad \forall\, x \in \Omega, \tag{2.15}$$

où G s'appelle la fonction de Green du problème et est définie par

$$G(x,y) = \begin{cases} y(1-x), & si\, 0 \le y \le x, \\ x(1-y), & si\, x \le y \le 1 \end{cases}$$

telle que $\Omega = [0,1]$.

Preuve. On montre tout d'abord l'existence.

Soit u donnée par (2.15), alors pour tout $x \in [0,1]$

$$\begin{aligned} \frac{du}{dx}(x) &= \frac{d}{dx}\left(\int_\Omega G(x,y) f(y)\, dy\right) \\ &= \frac{d}{dx}\left(\int_0^x y(1-x) f(y)\, dy + \int_x^1 x(1-y) f(y)\, dy\right) \\ &= x(1-x) f(x) - \int_0^x y f(y)\, dy - x(1-x) f(x) + \int_x^1 (1-y) f(y)\, dy \\ &= \int_0^x f(y)\, dy - \int_x^1 y f(y)\, dy, \end{aligned}$$

et d'aprs l'intégration par partie on obtient

$$\frac{d^2 u}{dx^2}(x) = -f(x),$$

donc $u \in \mathcal{C}^2(\bar{\Omega})$ et est solution de (2.14). Ceci montre l'existence.

Pour montrer l'unicité, il suffit de montrer que si $f \equiv 0$ la seule solution de (2.14) est la fonction nulle. En effet, si u_1 et u_2 sont deux solutions pour f donnée, alors

$$\begin{cases} -\Delta(u_1 - u_2)(x) = 0, & sur \quad \Omega \\ u_0 = u_1 = 0 & sur \quad \Gamma \end{cases}$$

En effet, pour tout $x \in [0,1]$

$$\int_0^1 G(x,y) \times 0 \, dy = 0.$$

donc u solution de (2.14), en intègrant deux fois, on obtient

$$u(x) = -\int_0^x \int_0^s f(y) \, dy \, ds + ax + b,$$

avec a, b dans $\mathbb{R}$ déterminées par les conditions aux limites $u_0 = u_1 = 0$. On obtient $b = 0$ et

$$a = \int_0^1 \int_0^s f(y) \, dy \, ds.$$

D'où,

$$u(x) = x \int_0^1 \int_0^s f(y) \, dy \, ds - \int_0^x \int_0^s f(y) \, dy \, ds.$$

On utilise alors le théorème de Fubini pour calculer l'intégrale en la variable s. Alors

$$\begin{aligned} u(x) &= x \int_0^1 \int_y^1 f(y) \, dy \, ds - \int_0^x \int_y^x f(y) \, dy \, ds, \\ &= \int_0^1 x(1-y) f(y) \, dy - \int_0^x f(y)(y-x) \, dy, \\ &= \int_0^1 (x(1-y) + \chi_{[0,x]}(y)(1-y)) f(y) \, dy, \end{aligned}$$

où $\chi_{[0,x]}$ est la fonction caractéristique de l'intervalle $[0,x]$ donnée par

$$\chi_{[0,x]} = \begin{cases} 1 & si \; y \in [0,x], \\ 0 & sinon, \end{cases}$$

soit $G : [0,1]^2 \to \mathbb{R}$ définie par $G(x,y) = x(1-y) - (x-y)\chi_{[0,x]}(y)$ pour tout $(x,y) \in [0,1]^2$, alors si $y \in [0,x]$, $G(x,y) = x(1-y) - (x-y) = y(1-x)$ et si $y \in]x,1]$ alors $G(x,y) = x(1-y)$.

■

2.10 Classification des EDP d'ordre 2

Le classement des équations étudiées dans l'une des trois catégories qui sont les équations paraboliques, elliptiques, et hyperbolique, se fait par l'étude de l'équation homog ène, linéaire du ordre deux :

$$A\frac{\partial^2 u}{\partial x_2} + 2B\frac{\partial^2 u}{\partial x \partial t} + C\frac{\partial^2 u}{\partial t_2} = 0$$

Cette équation représente le cas général, si on définit le discriminant

$$\triangle = B^2 - AC,$$

on se trouve face à trois possibilitès :

- $\triangle = 0$: on dit que l'équation est parabolique,
- $\triangle < 0$: on dit que l'équation est elliptique,
- $\triangle > 0$: on dit que l'équation est hyperbolique.

Exemple 2.2 *:*

1) L'équation de la Chaleur est parabolique ,

2) L'équation de Laplace est elliptique ,

3) L'équation des Ondes est hyperbolique .

Chapitre 3

Exemples d'applications

3.1 Résolutions (E.D.) d'ordre 1 (la fonction ode())

3.1.1 Point de terminologie et de forme

Terminologie : Une équation différentielle (E.D.) comme nous les connaissons est une Equation Différentielle Ordinaire, en anglais Ordinary Différential Equation (ode).
Par opposition, pour les fonctions à plusieurs variables les èquations avec des dérivées partielles différentes seront appelées Equations aux Dérivées partielles (E.D.P. et en anglais P.D.E.).

Forme : Une équation différentielle du premier ordre normalisée peut toujours s'écrire sous la forme :

$$y' = f(t,\, y) \quad \text{c'est-à-dire aussi} \quad y'(t) = f(t,\, y(t)).$$

Par exemple :
$y^{'}(t) = -2y(t) + e^{-t} + 1$ s'écrira $y^{'} = f(t,\, y)$, avec $f(t,\, y) = -2y(t) + e^{-t} + 1$.

3.1.2 L'appel de la commande ode de Scilab

Description

`ode` est la fonction utilisée pour approcher la solution d'une équation différentielle ordinaire (E.D.O.) explicite du premier ordre en temps, définie par :

$$\begin{cases} \frac{dy}{dt} = f(t,y), \\ y(t_0) = y_0. \end{cases} \tag{3.1}$$

Il s'agit d'une interface vers diverses bibliothéques, en particulier ODEPACK. Le type du problème et la méthode utilisée dépendent de la valeur du premier argument optionnel `type`.

• L'appel le plus simple de `ode` est du type : `y=ode(y0,t0,t,f)` où y_0 est le vecteur des conditions initiales, t_0 est le temps initial, et t est le vecteur des instants (on nécessite l'entrée dans l'ordre de quatre arguments) où l'on veut une approximation de la solution y est calculée et y est la matrice $y = [y(t(1)), y(t(2)), \ldots]$.

•Le paramétre d'entrée f de ode défini le membre de droite de l'équation différentielle du premier ordre $\frac{dy}{dt} = f(t,y)$. C'est un external qui peut être une fonction Scilab, sa syntaxe doit être `ydot = f(t,y)` où t est un scalaire (le temps), y un vecteur (l'état). Cette fonction renvoie le second membre de l'équation différentielle $\frac{dy}{dt} = f(t,y)$.

Motivation : la résolution numérique d'un problème de Cauchy

Autrement dit on se donne $(t_0,\ y_0) \in \mathbb{R}_+^* \times \mathbb{R}$ (pour l'instant) et l'E.D. $y'(t) = f(t,\ y)$ et on cherche la solution du problème de Cauchy (3.1) associé par un algorithme d'approximation numérique. Par défaut `ode` choisit lui-même la méthode selon la nature du problème.
Pour cela on doit se donner aussi l'intervalle des t sur lequel on veut calculer des valeurs de la solution approchée.
Par exemple, on résoudre l'équation différentielle précidant sur $\mathbb{R}^+$:

$$y'(t) = -2y(t) + e^{-t} + 1, \quad \text{avec} \quad y(0) = 1. \tag{3.2}$$

Et la solution formelle de (3.2) est :

$$y(t) = \frac{1}{2}(1 - e^{-2t}) + e^{-t}$$

```
**************************************************
drawlater()
clf();
function yprim = f(t,y)
```

```
yprim = -2*y+exp(-t)+1;
endfunction
t0 = 0;
y0 = 1;
t = t0 : 0.1 :10;
sol = ode(y0,t0,t,f);
xtitle(" La solution calculée par ode() ", "t","y ")
xgrid(15)
clf()
plot2d(t,sol,5); // La solution calculée par ode()
plot2d(t,0.5*(1-exp(-2*t))+exp(-t),-7);// La solution formelle
drawnow();
```

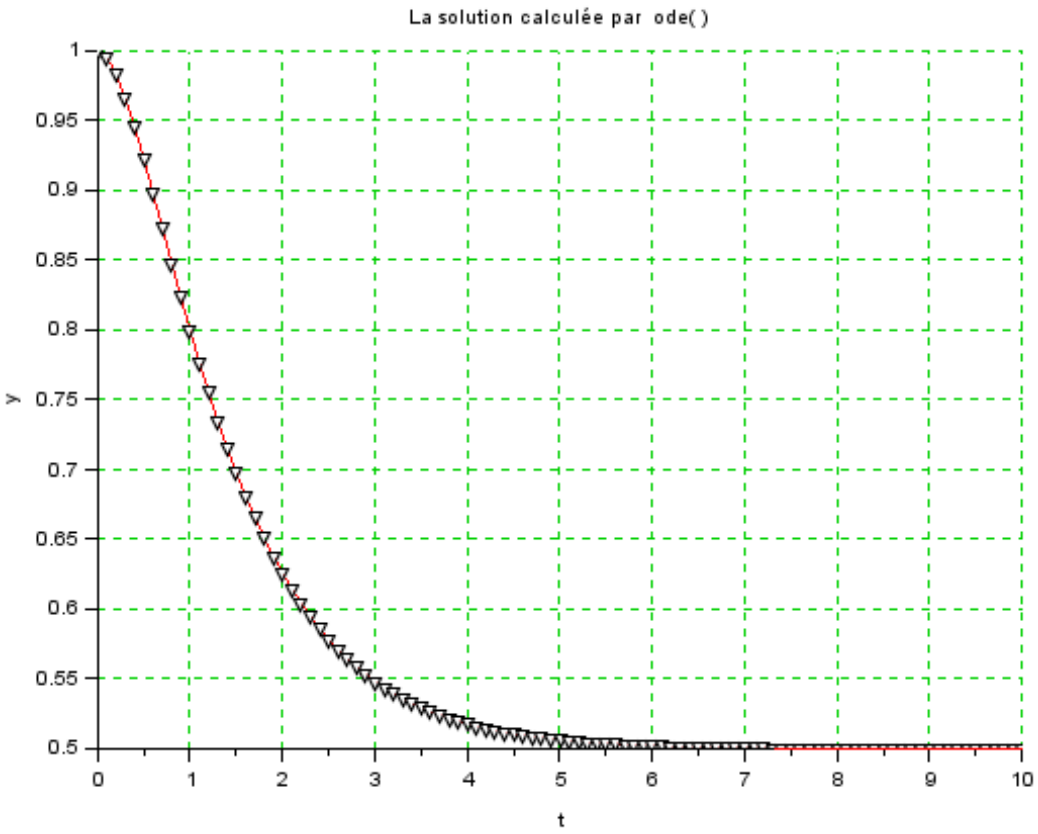


FIG 3.1.2 : La solution d'E.D. d'ordre 1

3.2 Transformer une E.D. d'ordre 2 en E.D. vectorielle d'ordre 1

3.2.1 La méthode illustrée sur une E.D. Linéaire

Intérêt mathématique plus général : On peut ramener toute la théorie des E.D.L. d'ordre quelconque à la théorie des E.D.L. d'ordre 1 dans le cadre vectoriel
Définition de la dérivée d'une fonction de $\mathbb{R}$ dans $\mathbb{R}^2$ coordonnées par coordonnées

$$\text{Si} \quad f : x \mapsto ((f_1(x), f_2(x)), \quad \text{alors} \quad f'(x) = (f_1'(x), f_2'(x)).$$

Motivation

On se donne une E.D.L. du second ordre

$$y''(t) = a(t)y'(t) + b(t)y(t) + c(t).$$

On va la ramener à une E.D. du premier ordre mais à inconnue une fonction vectorielle, avec le changement de variables :

$$Y_1(t) = y(t), \quad Y_2(t) = y'(t),$$

et on pose $Y(t) = (Y_1(t),\, Y_2(t))$ par la définition précédente de la dérivée $Y'(t) = (Y_1'(t),\, Y_2'(t))$.

L'E.D. initiale est équivalente au système : $\begin{cases} Y_1'(t) = Y_2(t) \quad \text{(c'est trivial)} \\ Y_2' = a(t)Y_2(t) + b(t)Y_1(t) + c(t) \end{cases}$

Finalement il s'agit donc de résoudre des systèmes d'équations différentielles du premier ordre :

$$Y'(t) = F(t,\, Y(t)),$$

où en notant $t \in \mathbb{R}_+^*$ et $Y = (Y_1,\, Y_2) \in \mathbb{R}^2$, $F : (t,\, Y) \mapsto (Y_2,\, a(t)Y_2 + b(t)Y_1 + c(t))$.
Ce système est dit explicite.

3.2.2 Traduction en Scilab

Le logiciel Scilab permet de résoudre de manière approchée toute équation (ou système d'équations) différentielle sous réserve d'existence et d'unicité de la solution.

Le `solveur` de systèmes différentielles explicites de scilab est la fonction `ode`.
On considère l'E.D. à coeficients constants

$$y''(t) + \alpha y'(t) + \beta y(t) = 0.$$

On peut la réécrire en posant $Y(t) = (y(t),\, y'(t))$ sous la forme $Y'(t) = F(t,\, Y(t))$ où pour $t \in \mathbb{R}_+^* Y = (Y_1,\, Y_2) \in \mathbb{R}^2$:

$$F(t,\, Y) = (Y_2, -\alpha y_2 - \beta y_1)$$

Application en Scilab : On veut tracer la solution pour $\alpha = 0.4$ et $\beta = 0.5$, au problème de Cauchy posé par cette E.D. $y''(t) + 0.4y'(t) + 0.5y(t) = 0$, avec $y(0) = 1$ et $y'(0) = 0$. On utilise le script suivant :

```
**************************************
drawlater()
clf()
function Yprim = F(t,Y);
Yprim(1) = Y(2);
Yprim(2) = -0.4*Y(2)-0.5*Y(1);
endfunction
t0 = 0; // temps initial
Y0 = [0 ;1]; // état initial rentrées comme un vecteur vertical
t = 0 :0.01 :10;
sol = ode(Y0,t0,t,F);// solution par ode
clf()
xtitle(" La solution calculée par ode() ", "t", "Y")
xgrid(15)
plot2d(t,sol(1, :),-15)
p = poly([0.5,0.4,1],"Y", "c")//on résout l'E.D. d'ordre 2 : y"+0.4y'+0.5y=0,
y(0)=0 et y'(0)=1
r = roots(p)// racines de la résolvante
im=imag(r(1))
```

```
reel=real(r(1))
z=(1/im)*exp(reel*t).*sin(im*t);// solution analytique
plot2d(t,z,5);
drawnow();
```

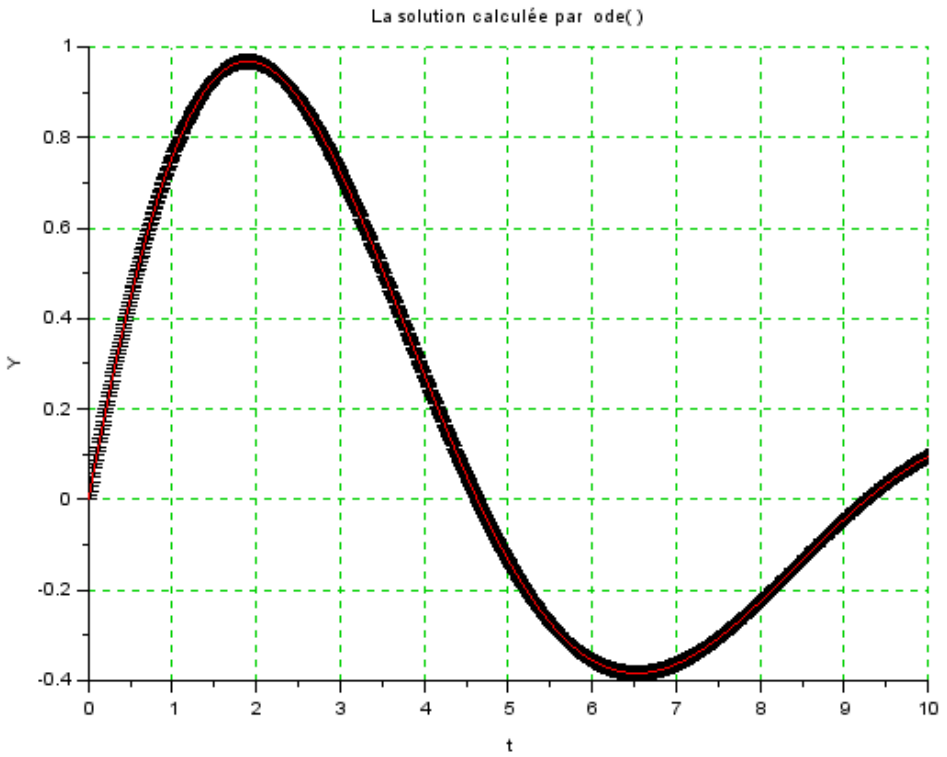


FIG 3.2.2 : La solution d'E.D. d'ordre 2.

3.3 Résolution d'une EDP parabolique

Enoncé :

Soit à résoudre l'équation de la chaleur $\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$ avec $\alpha \in \mathbb{R}$ fixe. Considérons le système :

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{\mu}{c\rho}\frac{\partial^2 u}{\partial x^2} \\ \\ u(x,0) = \begin{cases} 100x & si \quad x \in [0,1] \\ 100(2-x) & si \quad x \in [1,2] \end{cases} \\ \\ u(0,\,t) = u(2,t) = 0 \end{cases}$$

Prendre $\mu = 0.13$, $c = 0.11$, $\rho = 7.8g/cm^3$, $\Delta x = 0.25$. Δt sera donné par les conditions de stabilité d'efinies par l'inéquation : $r = \frac{\mu}{c\rho}\frac{\Delta t}{\Delta x^2}$. La solution analytique est donnée par :

$$u_{exact}(x,t) = 800 \times \sum_{n=0}^{\infty} \frac{1}{\pi^2(2n+1)^2} \times \cos\frac{\pi(2n+1)(x-1)}{2} \times e^{-0.3738(2n+1)^2 t} \quad [11]$$

3.4 Discrétisation de l'espace et du temps

3.4.1 Discrétisation de l'espace

Nous devons résoudre notre équation aux dérivées partielles dans l'espace compris entre 0 et 2 avec un pas Δx. Pour passer donc trouver le point x_i à partir du point x_{i-1}, nous devons ajouter à ce dernier le terme Δx, de sorte que nous avons :

$$\begin{cases} x_i &= x_{i-1} + \Delta x \\ x_{i-1} &= x_{i-2} + \Delta x \\ \vdots & \\ x_2 &= x_1 + \Delta x \\ x_1 &= x_0 + \Delta x \end{cases}$$

En sommant membres à membres les équations ainsi obtenues et après simplifications, nous trouvons : $x_i = x_0 + (i-1+1)\Delta x = x_0 + i\Delta x$ or, x_0 n'est autre que le point a de l'intervalle [a, b], finalement :

$$x_i = a + i\Delta x,\ 1 \leq i \leq n_x, \quad avec \quad n_x = \frac{b-a}{\Delta x} = \frac{2}{\Delta x}.$$

3.4.2 Discrétisation du temps

Notre équation, c'est à dire le procéssus de résolution doit se refaire dans un intervalle de temps égal à Δt jusqu'à arriver à T_{max}. Pour passer de t_j à t_{j+1} nous devons à chaque

fois ajouter Δt à t_j. Par un raisonnement par récurence comme précédemment, nous tirons : $t_j = j\Delta t$ car $t_0 = 0$ et correspond à l'instan $t = 0$.

$$t_j = j\Delta t, \quad 1 \leq j \leq n \quad \text{avec} \quad n_t = \frac{T_{max}}{\Delta t}.$$

3.5 Programmation de la solution analytique

Puisque nous sommes en analyse numérique et non en analyse fondamentale nous devons savoir que nous ne travaillons qu'avec des points d'un maillage et non dans tout le domaine comme on aurait cru le faire. Pour évaluer la valeur numérique de u_{exact} pour tout point appartenant dans la grille, nous devons faire une boucle qui, pour chaque point (x_i, t_j), calcule $u_{exact}(x_i, t_j)$. Posons $\infty = 100$ acceptable en analyse numérique, nous avons donc le programme suivant qui calcule les éléments de la matrice v représentant éxactement les valeurs de $u(x_i, t_j)$ dans le cadrillage

```
**********************************************************
clear;
k = 0.13; c = 0.11;
p = 7.8; dx = 0.25;
r = 1/4;
dt = dx*dx*c*p*r/k;
Tmax = 100*dt;// Nous avons fait Tmax un multiple de dt pour qu'en le divi-
sant par dt, nt soit toujours un entier, car sinon le programme ne compilera pas
cla = 0; clb = 0;
a = 0; b = 2;
nx=(b-a)/dx;
nt = Tmax/dt;
x = 0 : dx : b;
t = 0: dt :Tmax;
********** la solution analytique ********************
v = zeros(nx+1, nt+1);
```

```
n = 0;
while(n <= 100)
for i = 1 : nx+1
for j= 1 : nt+1
u(i,j) = v(i, j)+800*1/(%pi^2*(2*n+1)^2)*cos(%pi*(2*n+1)*(x(i)-1)/2)
*exp(-0.3738*(2*n + 1)^2 * t(j));
v(i,j) = u(i,j);
end
end
n = n+1;
end
mesh(t,x,v).
```

Voici la courbe que nous avons obténue pour $r = \frac{1}{4}$:

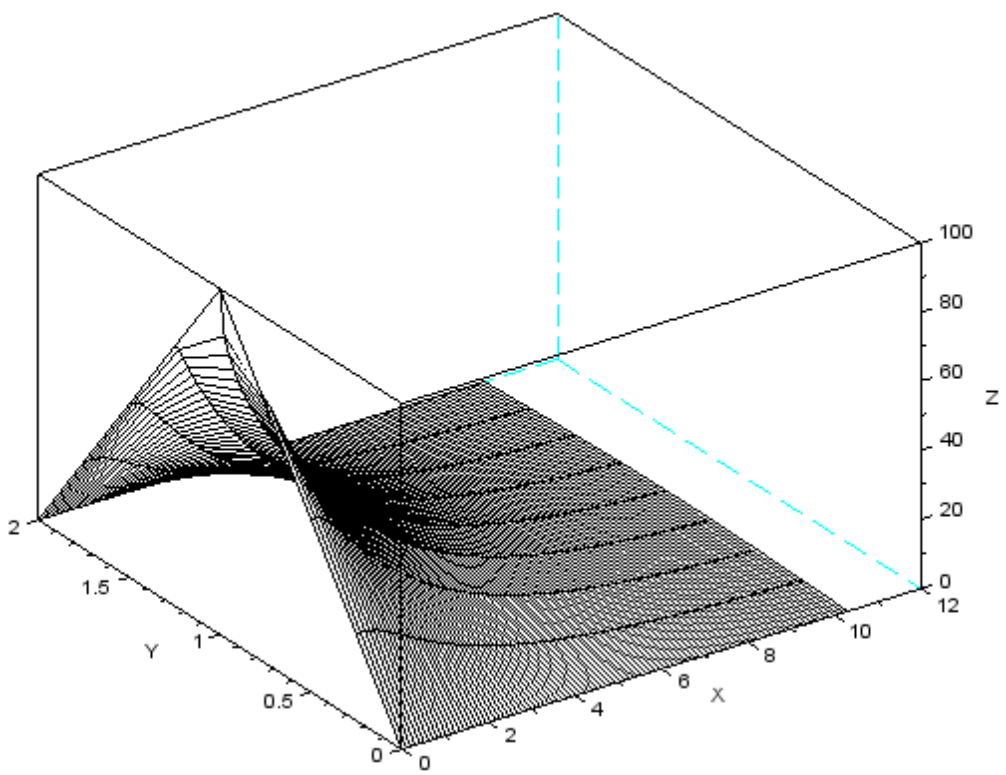


FIG 3.5 : Représentation graphique de la solution analytique $u_{exact}(x_i, t_j)$.

Conclusion

Ce travail a été pour nous l'occasion de découvrir le monde du calcul scientifique. De la modélisation physique au traitement des résultats numériques, ce travail nous a permis de comprendre et de saisir toute la complexité tant théorique en pratique que ce type de problème revêt. La résolution des équations aux dérivées partielles nécessite à la fois un bagage mathématique théorique relativement conséquent mais aussi de bonnes connaissances algorithmiques et informatiques. Nous avons bien sûr rencontre de nombreuses difficulté, de même, un tel projet nécessite, de par sa taille, une architecture et une organisation informatique que nous n'avions pas l'habitude de rencontrer. Il nous a donc fallu apprendre à évoluer dans une telle complexité et, plus d'une fois, il nous a fallu se plonger en profondeur dans les algorithmes et dans le fonctionnement de SCILAB pour résoudre les problèmes rencontrés.
Enfin, d'un point de vue purement technique, la rédaction de rapport a aussi été l'occasion de découvrir le formidable outil de traitement de texte qu'est LATEX.

Bien sûr, le monde de la résolution des E.D.P. étant tellement vaste et complexe, nous n'avons qu'effleuré ce dernier et bien d'autres aspects de ce projet auraient pu se révéler intéressants a développer. Par exemple, nous avons choisi de nous restreindre plus à la résolution effective numérique du système plutôt que de se pencher sur la théorie et justifier l'existence de solutions faibles pour ce problème. De même, il existe d'autres méthodes de résolution de tels systèmes (méthode du point fixe par exemple), Malheureusement, faute de temps et aussi face aux difficultés rencontrées, nous n'avons pas eu l'opportunité de développer en profondeur ces aspects. Néanmoins, tout l'intérêt de ce projet réside justement dans l'ouverture vers ces aspects. Par le travail effectué, par les lectures, nous avons abordé un monde vaste et passionnant qui a éveillé tout notre intérêt.

Bibliographie

[1] B. Arnaud, Informatique et algorithmique avec le logiciel Scilab en ECS 1^{re} année. Lycée Fermat Toulouse, 22/01/2014.

[2] Ch. Billon, Initiation Scilab I. Universitè Pierre et Marie Curi, 2007-2008.

[3] H. Bresis, Analyse fonctionnelle : Théorie et appliccations, Masson, Paris, 1987.

[4] J. Ph. Chancelier, Scilab : une introduction, Version 1.0. École Nationale des Ponts et Chaussées, 31/08/2004.

[5] J.-P. Chancelier, F. Delebecque, C. Gomez, M. Goursat, R. Nikoukhah, S. Steer, Introduction à Scilab Deuxième édition. France, 2007.

[6] R. Chill, Introduction aux équations aux dérivées partielles. Laboratoire de Mathématiques et Applications de Metz, 2010-2011.

[7] P. Ferreira, Sylvie, Mas-Gallic , Équations aux Dérivées Partielles. Pari, 11/12/2001.

[8] C. Gomez, Scilab pour l'enseignement des mathèmatiques. lycée Descartesà Antony, 2013.

[9] G. Koeper, Équation aux Dérivées partielles. Université René Descartes, Paris, 2001.

[10] S. Nicaise, L'analyse numérique et équations aux dérivées partièlles, cours et problèmes résolus. Dunod, Paris, 2000.

[11] N. Lantas, J.planchon, Résolution de l'équation istationnaire de la chaleur 1D. Université pierre et mrie-paris6, 09/03/2015.

[12] P.A. Raviart, J.M. Tomas, Introduction à l'analyse numérique des équations aux dérivées partiélles, Masson, Paris, 1983.

[13] M. Postel, Introduction au logiciel Scilab. Paris, janvier 2009.

[14] A. Rondepierre, A. Rouchon, Introduction aux Équations aux Dérivées Partielles, Étude théorique. Département STPI, 2012-2013.

[15] B. Ycart, Démarrer en Scilab. Université René Descartes, Paris.