Conseils pour l'épreuve de modélisation

Adrien Laurent
École Normale Supérieure de Rennes

2015 - 2016

Table des matières

Chapitre 1

Préparation à l'épreuve de modélisation

Souvent oubliée, l'épreuve de modélisation n'est pas à négliger. Elle représente un cinquième des points de l'agrégation ! Pour s'y préparer, il faut à la fois avoir une culture scientifique et connaître parfaitement les méthodes au programme. Bien sur, savoir programmer rapidement et efficacement est aussi nécessaire.
Je présente dans ce document la plupart des choses à retenir pour aller à l'épreuve "sereinement" si on a choisi l'option Calcul Scientifique.

• Premiers conseils :

1. Les textes de modélisation passés en juin sont les mêmes sur les deux semaines d'oraux. Il est donc malin de regarder les oraux des premiers à passer si nos oraux sont à la fin. On a ainsi une chance de tomber sur un texte sur lequel on a déjà réfléchi.
2. On ne porte pas de vêtements trop courts. Le jury n'a pas envie de voir le slip du candidat pendant 40 minutes...
3. L'épreuve de modélisation est surement l'épreuve qui se rapproche le plus d'un cours réel. Le jury attend donc beaucoup de pédagogie. Il faut jouer le jeu et aborder cette épreuve de manière scolaire, en expliquant tout clairement.

• Critères de notation : La note de modélisation se répartit en 5 critères, tous aussi importants :

1. La modélisation du problème (pédagogie et clarté attendues)
2. La programmation
3. Les connaissances
4. Les questions
5. L'organisation (faire un plan clair et intelligent)

Chapitre 2

L'épreuve de modélisation

• Pendant la préparation :

1. Il est demandé au candidat de rajouter des connaissances personnelles au texte de départ (preuves des propriétés non prouvées, modélisations, ...), mais il ne faut surtout pas faire de hors-sujet ! Il est très simple de trop se concentrer sur un élément que l'on connaît bien et d'oublier le reste. Attention donc ! Néanmoins, si on change la modélisation ou qu'on change une preuve, ce sera très bien vu.
2. Le plan est un élément important de l'oral, il ne faut pas le négliger. On ne peut pas arriver avec deux parties nommées modélisation et simulations. Il faut chercher une problématique et y répondre. Il faut aussi mettre des sous-sections. On peut mettre un titre original si on en trouve un.
3. Il faut répartir les simulations dans notre plan. Une simulation en introduction fait une TRÈS bonne impression !
4. Si les simulations ne marchent pas, ce n'est pas grave. On pourra les corriger pendant l'oral avec le jury souvent. De manière générale, il vaut mieux ne pas se lancer dans des programmes trop compliqués de manière à avoir des choses à présenter au jury.
5. Il faut tirer le maximum de nos programmes. Parfois, on peut faire des schémas en plus même s'ils ne sont pas nécessaires. Par exemple, on peut tracer les trajectoires données par une EDO en plus du portrait de phase habituel.
6. On est souvent obligé de faire toutes les simulations sur un exemple. C'est une bonne chose d'étudier à fond cet exemple et de le montrer tout au long de l'oral. C'est de la valeur ajouté !
   Typiquement, on peut penser à la matrice du laplacien qui apparaît partout, ou à un schéma d'oscillateur harmonique, etc...
7. Au cas où nos programmes ne compileraient pas face au jury, on fait quelques impressions écran de nos graphiques.
   Si ce sont les exemples du texte qui ne marchent pas (typiquement pour tracer l'ordre d'un Newton), alors on l'applique sur des exemples simples. Sans ça, c'est un programme perdu.
8. Il est bon de revenir au problème réel donné par le texte au cours de la présentation. Il faut trouver une utilité aux théorèmes prouvés, aux simulations.
9. Le jury attend que le candidat exploite les simulations. Il veut des résultats. Du coup, plutôt que d'implémenter une méthode déjà programmée dans Scilab, on peut l'utiliser et le dire au jury (typiquement, les décompositions LU, QR, etc...). On insistera alors sur des applications de l'algorithme utilisé et on tendra une perche au jury en donnant des noms de méthodes à utiliser.
   Dans tous les cas, il vaut mieux parler de toutes les méthodes numériques qu'on aurait pu implémenter. Cela donne plein de perches...
10. 10 à 20 minutes avant l'oral, on arrête tout et on réfléchit sur comment présenter nos résultats, comment faire une introduction aguicheuse, bien répartir les simulations, etc... On fait les liens entre les parties pour que la présentation soit fluide.
    Cela rend beaucoup mieux à l'oral et ça met à l'aise.
11. Il faut prendre le temps d'écrire un théorème et sa preuve entièrement et proprement. Si on arrive à faire une vraie preuve bien claire, c'est le jackpot !
12. Une partie/sous-partie sur l'aspect numérique du problème n'est JAMAIS hors-sujet.
13. On termine sur une conclusion où on résume ce qu'on a fait et avec d'autres pistes de recherche. Cela donne des points en plus facilement et c'est mieux que de terminer sur un "Voilà, j'ai fini !".

• Sur le tableau :

1. On commence par découper le tableau avec la craie. Il est important que le tableau ne soit pas en bazar.
2. Il faut écrire le plan complet à gauche du tableau pendant l'introduction, puis ne pas l'effacer.
3. Mettre une astérisque devant chaque sous-partie où des simulations sont présentes pour que le jury puisse interroger dessus.
4. Si on a besoin d'une fonction pour plus tard, on l'écrit dans un coin du tableau et on la garde ici pour ne pas l'effacer.
5. Il faut écrire tous les théorèmes non triviaux utilisés, même s'ils ne sont pas dans le texte. On n'est pas obligé de tout démontrer (c'est conseillé d'ailleurs.). Néanmoins, si on n'a pas le temps, il vaut mieux oublier les théorèmes...
6. Souvent, un dessin fait bien meilleure impression que des calculs et des phrases complexes.

• À l'oral :
Il y a quatre grandes choses qu'attend le jury et elles sont :

1. **une introduction claire qui présente nos attentes/nos buts**, avec une simulation si possible,
2. **une modélisation propre et pédagogique**,
3. **une partie/sous-partie mathématique rigoureuse**,
4. **des simulations nombreuses, expliquées et mises en valeur**, bien présenter les méthodes numériques utilisées, les données de départ...
   Contrairement à une idée reçue, on a le droit et il est même conseillé de **montrer son code** pour expliquer notre démarche.

Bien sur, il y a quelques autres choses à faire/ne pas faire :

1. Parler fort et bien regarder le jury.
2. Il faut être clair : écrire bien, bien expliquer, montrer que l'on sait être pédagogue. Si on présente quelque chose de fabuleux d'une mauvaise manière, on ruine tout.
3. Ne surtout pas parler du texte. Le jury n'est pas censé le connaître. On est censé faire un cours à des non-initiés.
4. Pour éviter d'aller trop vite sur les simulations, on les présente avant de les montrer en disant ce qu'elles vont montrer. Ça réveille le jury.
   Pour ma part, j'écris "simulation" au tableau et je fais une pause dans la présentation en expliquant bien mon programme.
5. Quand on admet quelque chose, il faut le dire ! Si on prouve trois implications sur quatre dans un théorème, on doit le dire ou l'écrire avant de commencer la preuve. De même, si on fait un cas particulier dans une preuve, on le dit.
6. Cela semble évident, mais on ne finit pas une présentation par un "je n'ai pas réussi à faire ...". On doit finir sur une note positive.
   De même, on ne commence pas la présentation en disant ce qui ne marche pas. Il faut parler des problèmes au tout dernier moment pour ne pas faire mauvaise impression (mais il faut en parler tout de même).
7. Au début de la présentation, on doit impressionner le jury, d'où la stratégie de mettre une simulation en intro.
8. Pour éviter de se mélanger dans ses feuilles, on les **numérote** et on les relit.

Chapitre 3

Méthodes basiques de programmation

• Répartition/Organisation des programmes :
Le plus simple est de faire un script (en .sci) contenant toutes les fonctions. Puis on fait différents fichiers (des .sce) qu'il suffit de lancer pour montrer les courbes/ résultats.
Ceux-ci commenceront par la ligne suivante pour s'éviter des problèmes récurrents de variables déjà définies :

```
clear; exec "fonctions.sci"
```

Cette simple ligne fait déjà bonne impression sur le jury.

• Faire un beau plot :

```
plot2d(X,[Y1,Y2,Y3],style=[color("green"),color("blue"),color("red")])
title('Beau graphique')
xlabel('Bel axe des abscisses');ylabel('Bel axe des ordonnées')
legend(['fonction1';'fonction2';'fonction3'])
```

• Faire des tests d'arrêts et les connaître :
Pour éviter les singularités, ou juste pour ne pas faire des calculs inutiles, on met des boucles *while* avec des tests d'arrêt. Par exemple, pour une méthode de gradient, il faut vérifier à chaque itération que $\|x_{k+1} - x_k\| > \varepsilon$, avec ε une certaine tolérance.

• Faire une pause : Pour faire une pause entre deux plot dans le même script, et pour afficher les choses petit à petit, on peut utiliser la commande *xclick*. Sinon il y a aussi la commande *pause*.

• Faire une animation :
Le principe est de faire une boucle où on calcule ce qu'il faut afficher puis on fait le plot après. Si on fait cela naïvement, on obtient une animation qui flash. Pour éviter ce problème, on rajoute les lignes de code suivantes :

1. `drawlater()`
   au début de la boucle,
2. `set(gca(),"auto_clear","on")`
   juste avant le plot,
3. `plot2d(X,Y,rect=[xmin,ymin,xmax,ymax])`
   pour fixer les axes,
4. `drawnow`
   après le plot.

• Le plot3d :
Cette fonction demande trois arguments : un vecteur abscisse x, un vecteur temps t et **une matrice** y contenant la fonction à tracer. On tape simplement les lignes suivantes :

```
plot3d(x,t,y)
xtitle('Evolution de l approximation numérique','abscisse x','temps t','ordonnée y')
```

Si on veut en plus des couleurs pour mieux repérer les différentes valeurs, on peut utiliser à la place la superbe commande *plot3d1* (à consommer sans modération [1]).
Bien sur, une animation est bien mieux si on a le temps de la faire.

• Intégrer une fonction simplement :
Pour calculer vite, $X = \int_a^b f(x)\mathrm{d}x$, on fait

```
X=integrate('f(x)','x',a,b)
```

• Les matrices faciles !

1. La transconjuguée de A est A'.
2. Calculer $A^{-1}b$ s'écrit $A\backslash b$ et non $Inv(A) * b$.
3. Créer une matrice par bande :
   Il faut utiliser la fonction *diag*. Ici on fait l'exemple de la matrice laplacien en dimension 1.
   ```
   function A=Lap(n)
       A=2*diag(ones(n,1))-diag(ones(n-1,1),1)-diag(ones(n-1,1),-1)
   endfunction
   ```
4. Extraire la diagonale d'une matrice :
   C'est encore la fonction *diag* !
5. Des matrices aléatoires pour des tests plus crédibles :
   Si on veut une matrice de taille $n \times n$, on tape $A = rand(n, n)$. Par contre tous les coefficients seront entre 0 et 1. Pour corriger cela, on peut multiplier par une constante.
6. Obtenir une matrice réelle aléatoire avec spectre réel :
   On se donne A avec la méthode précédente puis on pose $B = A + A'$ pour obtenir une matrice symétrique réelle. =)
7. Spectre et vecteurs propres d'une matrice :
   La commande *spec* est celle qu'il faut utiliser.
   Si on écrit $[a, b] = spec(A)$, alors b est une matrice diagonale contenant les valeurs propres de A et a est une matrice contenant en colonne les vecteurs propres de A associés aux valeurs propres précédentes (dans l'ordre).
   Si on écrit $c = spec(A)$, alors c est un vecteur contenant les valeurs propres de A.
8. Quelques autres fonctions à ne pas reprogrammer pour ne pas perdre de temps :
   le rang avec *rank*, la trace avec *trace*, le noyau avec *kernel*, la décomposition QR avec *qr*, la décomposition LU avec *lu*...

• Visualiser un champ de vecteurs :
La fonction *fchamp* permet de faire cela. Pour l'utiliser, il faut lui fournir en argument une fonction de la forme $f(t, x)$ puis le temps auquel on veut l'évaluer. Enfin on rajoute un vecteur d'abscisses des points à étudier, ainsi qu'un vecteur d'ordonnées.

• Calculer la valeur d'une pente :
Pour cela, on fait une régression linéaire en utilisant *reglin*. Attention, la fonction est capricieuse : elle renvoie trois arguments dont le premier est la pente, et si on a le malheur de mettre des vecteurs colonnes au lieu de vecteurs lignes, elle renvoie une matrice remplie de 0...

```
[pente, b, s]=reglin(X,Y)
```

• Les singularités du log :
On tombe toujours sur ce genre d'erreur due à la l'erreur machine. Du coup, au lieu de calculer $log(m)$, on calcule $log(m + \%eps)$ ($\%eps$ est l'erreur machine).

• Trier des objets :
La fonction à utiliser est *gsort*. Pour inverser tous les éléments d'un vecteur v, on pourra utiliser $v(n : -1 : 1)$ avec n la taille du vecteur.

• Faire afficher une valeur à Scilab :
Si a est la variable à afficher (typiquement, la pente d'une droite), il suffit d'écrire

1. Dans le cas où vous seriez épileptique, demandez d'abord conseil à votre médecin.

```
disp("La valeur est ");disp(a)
```

Pour afficher une valeur dans un titre, on peut faire de cette manière :

```
title(["Log de l'erreur en fonction de n. La pente est de ", string(42), ". Génial!"])
```

• Ne pas s'embêter avec les tailles des objets et les indices :
L'épreuve de modélisation n'est pas un exercice de combinatoire ou de dénombrement... Il y a des astuces à connaître pour s'éviter les dizaines de minutes perdues à corriger des indices.

1. Le dernier élément d'un vecteur v est donné par $v(\$)$. Pour une matrice, on peut l'adapter en $m(:, \$)$.
2. Pour calculer la longueur d'un vecteur, on utilise *length*. Pour la taille d'une matrice, il y a *size*.
   On peut écrire $size(A, 1)$ pour le nombre de lignes et $size(A, 2)$ pour le nombre de colonnes.
3. Les abscisses faciles : $X = a : h : b$
4. Il n'y a pas que le $for \quad i = 1 : n$, on peut utiliser $for \quad i = a : pas : b$ avec a et b entiers et un pas entier pouvant être négatif !
5. Si on veut trouver le maximum des valeurs d'un vecteur/ d'une matrice **réelle** ainsi que sa position, on utilise $[m, n] = max(A)$. Le maximum est m et sa place est n.
   Si on veut le maximum en module, on écrit $[m, n] = max(abs(A))$.
6. Si on veut une fonction qui à un vecteur x renvoie un vecteur constant à c de la même taille, il ne faut pas renvoyer c, il faut renvoyer $c + 0 * x$ pour conserver la taille.

• Les choses à ne pas faire :

1. Ne jamais inverser une matrice pour calculer $A^{-1}b$, on écrit $A\backslash b$. On passe ainsi d'une longue inversion de matrice à une simple décomposition LU.
2. Créer des fonctions dans d'autres fonctions.
   Il y a une loi empirique à ce sujet, ça ne marche jamais. Personne ne sait pourquoi, donc il vaut mieux ne même pas essayer.
3. Donner des noms trop simples à ses variables.
   Tout le monde le fait et cela créé toujours les mêmes erreurs. Si une fonction prend en argument une fonction f et qu'une autre fonction définie avant s'appelle aussi f, on peut être sur que ça ne va pas marcher.
4. Si on fait une boucle *if*, la commande *then* doit être sur la même ligne que le *if*.

Chapitre 4

Analyse numérique matricielle

Tous les détails peuvent être trouvés dans le Ciarlet, le Rappaz-Picasso ou le Quarteroni.

- Trouver des valeurs propres et des vecteurs propres :
  1. Méthode QR :
     C'est une méthode un peu magique. Il "suffit" de trouver la décomposition QR et d'inverser l'ordre des deux matrices. En itérant ce processus, on tend vers une matrice triangulaire supérieure dot la diagonale contient les valeurs propres.

     ```
     function An=QRmethode(A,N)
         An=A
         for i=1:N
             [Q,R]=qr(An)
             An=R*Q
         end
     endfunction
     ```

  2. Méthode de la puissance :
     Elle permet de trouver la valeur propre de plus grand module d'une matrice diagonalisable, ainsi qu'un vecteur propre associé.
     Soit A une matrice diagonalisable ayant pour valeurs propres $|\lambda_1| > |\lambda_2| \geqslant ... |\lambda_n|$ et comme vecteurs propres associés les (e_i), alors si x_0 est tel que $x_0 = \sum \alpha_i e_i$ avec $\alpha_1 \neq 0$, et si $(x_n)_n$ est définie par $x_{n+1} = \dfrac{Ax_n}{\|Ax_n\|}$, on a $x_n \to e_1$ et $Ax_n \to \lambda_1$. La vitesse de convergence est $\dfrac{\lambda_2}{\lambda_1}$.

     En pratique, on met un critère d'arrêt sur la convergence de $\|Ax_n\|$. Voici un exemple.

     ```
     function X=puissance(A,x0,N,tol)
         X(:,1)=x0
         i=1
         while (i<=N & (i==1 | abs(norm(y)-norm(A*X(:,i)))>tol))
             y=A*X(:,i)
             X(:,i+1)=y/norm(y)
             i=i+1
         end
     endfunction
     ```

  3. Méthode de la puissance inverse :
     C'est la même méthode sauf qu'on l'applique à A^{-1}. On a ainsi en retour l'inverse de la valeur propre minimale de A, ainsi qu'un vecteur propre associé.
     On fera bien attention à ne pas calculer l'inverse de A à l'intérieur mais plutôt à inverser des systèmes.
  4. Méthode de la puissance inverse translatée :
     Si on connaît une approximation de chaque valeur propre, on peut avoir accès aux vecteurs propres associés. Pour cela, il suffit de décaler le spectre de notre matrice A de chaque valeur propre, puis d'appliquer la puissance inverse.
     Attention, il faut rajouter un ε au décalage sinon la matrice n'est plus inversible !

     Dans l'algorithme suivant, j'utilise une méthode QR pour approximer les valeurs propres puis j'applique ma méthode de puissance inverse n fois. Remarquez que j'ai décalé mon estimation de valeur propre de $\varepsilon = 0.0001$ qui n'est pas trop petit pour éviter les problèmes de conditionnement.

```
function [valp,vecp]=spectre(A,x0,Nqr,Npuissance,tol)
    n=size(A,1)
    An=QRmethode(A,Nqr)
    V=gsort(diag(An)) //trie les valeurs propres
    valp=V(n:-1:1) //met le vecteur à l'envers pour coïncider avec la fonction spec de Scilab
    for i=1:n
        aux=puissancetranslatee(A,valp(i)+0.0001,x0,Npuissance,tol)
        vecp(:,i)=aux(:,$)/norm(aux(:,$))
    end
endfunction
```

5. Méthode de déflation :
   Supposons que l'on connaisse le premier vecteur propre e_1. Alors on peut trouver la seconde valeur propre λ_2 et son vecteur propre associé si A est symétrique réel.
   Pour cela il suffit d'appliquer la méthode de la puissance à $B = A - \frac{\lambda_1}{\|e_1\|^2} e_1 {}^{\mathrm{t}}e_1$. En effet, comme A est symétrique, ses vecteurs propres sont orthogonaux, donc les valeurs propres de B sont λ_2, ..., λ_n et 0.

• Les méthodes d'inversion, de résolution de systèmes linéaires :
On cherche à résoudre un problème du type $Ax = b$. Il y a trois types de méthodes : les **méthodes directes**, les **méthodes itératives** et les **méthodes de descente**.

1. La méthode directe naïve est d'utiliser les formules de Cramer, mais c'est terriblement long ! On utilise alors d'autres méthodes pour se ramener à une **complexité en** $O(n^3)$ (on rappelle qu'une multiplication naïve de matrices est déjà en $O(n^2)$).
   La méthode fondamentale est **le pivot de Gauss** : on multiplie A par des matrices de transvection pour se ramener à une matrice triangulaire. Avec une matrice triangulaire, le problème est simple.
   Si on garde en mémoire les opérations effectuées, on obtient **la décomposition LU**. En scilab, elle est déjà implémentée : c'est la fonction *lu*.
   On peut citer **la méthode de Cholesky**, puis **la décomposition QR** (*qr* en Scilab), qui n'est autre que la méthode de Gram-Schmidt écrite de manière matricielle. Pour cette dernière, il faut d'ailleurs faire attention à bien utiliser l'algorithme avec les matrices de Householder, sinon on peut obtenir un algorithme instable.
2. L'idée des méthodes itératives est de décomposer A sous la forme $M-N$ avec M inversible (décomposition régulière de A). On construit la méthode itérative par
   $$\begin{cases} x_0 \in \mathbb{R}^n \\ x_{k+1} = M^{-1}(Nx_k + b) \end{cases}.$$
   Alors la méthode converge si et seulement si $\rho(M^{-1}N) < 1$, car les erreurs valent $e_k = (M^{-1}N)^k e_0$.
   Les différentes décompositions régulières conduisent à différentes méthodes : **Jacobi**, **Gauss-Seidel**, **relaxation**. Pour des détails, je renvoie au Rappaz-Picasso qui décrit bien toutes ces méthodes.
   Je précise juste qu'ici la convergence ne dépend pas du conditionnement, ce qui est assez exceptionnel pour qu'on le remarque. En pratique néanmoins on n'utilise jamais ces méthodes.
   Si on veut les tester, un super exemple est la matrice de Laplacien car si on fait croître sa taille, le rayon spectral $\rho(M^{-1}N)$ des différentes méthodes va tendre vers 1.
3. On verra les méthodes de descente au chapitre sur l'optimisation. Pour les utiliser, il faut que $A \in \mathcal{S}_n^{++}(\mathbb{R})$.
   On utilise en pratique soit du **Newton** appliqué à $x \mapsto Ax - b$, soit des **méthodes de gradient à pas fixe ou optimal** tentant de minimiser $x \mapsto \frac{1}{2}(Ax, x) - (b, x)$.
   Ces méthodes dépendent bien sur du conditionnement.

• Sur quoi tester nos programmes ?

1. On teste les algorithmes sur des matrices et des vecteurs aléatoires, cela fait gagner du temps et ça marche souvent.
2. La matrice typique avec un mauvais conditionnement : la matrice de discrétisation du laplacien en dimension 1. Il faut y penser tout de suite ! On connaît ses valeurs propres donc tous les calculs peuvent être explicités. Attention cependant, ses valeurs propres sont comprises entre 0 et 4 (et non entre 0 et 2).
   On peut aussi citer les matrices de Hilbert dans les exemples de matrices ayant un (très) mauvais conditionnement : $B_{i,j} = \frac{1}{i+j-1}$.

3. La matrice des splines cubiques est typiquement une matrice bien conditionnée :

$$\begin{pmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{pmatrix}.$$

On peut néanmoins en prendre d'autres plus simples (diagonales...) car les calculs pour un bon conditionnement ne sont pas très intéressants, tout marche !

Chapitre 5

Approximation d'intégrales

• Les réflexes :

1. Calculer l'erreur d'une méthode de quadrature avec *Taylor-Lagrange*, ainsi que son ordre.
2. Les voir numériquement avec un graphe log-log et en testant contre des polynômes de degré allant jusqu'à $2n+1$.
3. Penser aux changements de variables pour se ramener aux fonctions poids connus.

• Les méthodes :
Tout est dans le Demailly ou le Crouzeix-Mignot. On se donne un intervalle que l'on subdivise, puis on somme toutes les approximations d'intégrale sur les subdivisions.

1. Méthode des rectangles (ordre 0) : $I(f) = (b-a)f(a)$ ou $I(f) = (b-a)f(b)$
2. Méthode du point milieu (ordre 1) : $I(f) = (b-a)f\left(\dfrac{a+b}{2}\right)$
3. Méthode des trapèzes (ordre 1) : $I(f) = (b-a)\dfrac{f(a)+f(b)}{2}$
4. Méthode de Simpson (ordre 3) : $I(f) = \dfrac{b-a}{6}\left(f(a) + f\left(\dfrac{a+b}{2}\right) + f(b)\right)$
5. Méthode de Monte Carlo pour la culture : on se donne une loi uniforme U sur $[a,b]$, alors l'idée est d'écrire $\displaystyle\int_a^b f(x)\mathrm{d}x = \mathbb{E}[f(U)]$. On prend donc N points x_i au hasard dans $[a,b]$, puis on écrit $I_N(f) = \dfrac{b-a}{N}\sum f(x_i)$. La consistance de cet estimateur est donnée par la LGN ; on peut aussi avoir un intervalle de confiance en choisissant bien N grâce au TCL. On montre que cette méthode est d'ordre 1. Son intérêt réside dans le fait qu'on peut la généraliser aux dimensions supérieures sans réduire son ordre ! De plus, elle est insensible à la régularité de la fonction f !
6. Méthode de Gauss : on se donne une fonction poids et on cherche les polynômes orthogonaux à celle-ci (difficile en pratique). Puis on pose $I_n(f) = \sum \lambda_i f(x_i)$ où les x_i sont les zéros du $(n+1)$-ième polynôme orthogonal. Les λ_i sont définis par $\lambda_i = \displaystyle\int_a^b L_i(x)w(x)\mathrm{d}x$ avec L_i polynôme d'interpolation de Lagrange de degré n tel que $L_i(x_j) = \delta_{i,j}$. Ainsi on obtient une méthode d'ordre $2n+1$.

• Les pièges :

1. Quand on fait un changement de variable en dimensions plus grandes que 1, il est conseillé de bien vérifier que le changement de variable est associé à un $\mathcal{C}^1$-difféomorphisme. De plus, il ne faut pas oublier de rajouter la **valeur absolue** du jacobien dans l'intégrale.

Chapitre 6

Équations différentielles ordinaires

- Les réflexes :

1. Dans la modélisation physique, on explicite la régularité nécessaire des fonctions et on dit quand on l'utilise.
2. Cauchy-Lipschitz dès le départ
3. On tente de prouver que la solution est globale. Souvent il suffit juste d'appliquer le théorème de sortie de tout compact avec Gronwall, mais parfois, on peut avoir à trouver une fonction de Lyapounov/une intégrale première (chercher une énergie) et à appliquer la méthode précédente.
   Si on a un compact à champ de vecteur rentrant (stable par le flot), alors la solution est bornée dans celui-ci, donc globale.
4. Théorème de Lyapounov pour la stabilité, étude des valeurs propres de la jacobienne (penser aux disques de Gershgörin pour localiser les valeurs propres). Si on n'a pas de résultat avec cette méthode, on peut simuler pour voir à peu près ce qui se passe.
5. La solution du système linéarisé approche souvent la vraie solution ; on peut la tracer pour le voir.
6. Quand les coefficients de l'équation sont périodiques, on peut **moyenner** l'équation : cela permet d'enlever les oscillations. On peut alors voir le comportement des solutions. Il "suffit" ensuite de rajouter des oscillations pour obtenir les vraies solutions.
7. Si on voit l'existence d'une intégrale première, on cherche un schéma numérique qui la conserve ! On vérifie numériquement si ça marche, etc ... Un bon exemple est celui de l'oscillateur harmonique

$$\left\{ \begin{array}{rcl} x' & = & y \\ y' & = & -x \end{array} \right. .$$

   L'intégrale première est $I(x,y) = x^2 + y^2$.
   D'autres bons exemples à connaître sont donnés par le pendule, certains circuits RLC, des systèmes de mécanique céleste, le modèle de Lotka-Volterra et bien d'autres systèmes physiques...
   Pour montrer qu'une fonction est une intégrale première, on la dérive et on remplace les dérivées.
   Les schémas numériques ne vérifient pas toujours exactement les intégrales premières. Parfois ils vérifient une intégrale approchée qu'il est bon d'étudier.
   Attention, certains schémas sont si précis qu'on a l'impression qu'ils vérifient les intégrales premières (typiquement RK4) alors que ce n'est pas le cas.

- Les gros théorèmes liés à utiliser parfois :

1. Le flot est différentiable en tous les paramètres ! Donc si on bouge les conditions initiales, ça ne dégénère pas trop. Le flot φ vérifie de plus l'équation suivante, où on note $\psi_t(y_0) = \dfrac{\partial \varphi}{\partial y_0}(t,0,y_0)$ et où l'équation différentielle étudiée est autonome,

$$\left\{ \begin{array}{l} \dfrac{\mathrm{d}\psi_t}{\mathrm{d}t}(y_0) = \dfrac{\partial f}{\partial y}(\varphi(t,0,y_0))\psi_t(y_0) \\ \psi_0(y_0) = Id \end{array} \right.$$

   La preuve est complexe. Il faut utiliser la résolvante, un cylindre de sécurité, les accroissements finis...
   On peut la trouver sur la page web de Philippe Chartier (cours d'EDO de L3 - chapitre 5).
2. Le théorème des fonctions implicites peut être appliqué notamment dans le cas précédent, pour montrer l'existence d'une trajectoire privilégiée avec un champ de vecteurs allant vers elle.

• Les astuces :

1. Pour calculer une différentielle d'une fonction à beaucoup de composantes, on peut calculer la différentielle de chaque composante plutôt que d'écrire une jacobienne monstrueuse !
2. Pour trouver les valeurs propres du laplacien, on prend le vecteur propre $V = \left(\sin\left(\frac{kp\pi}{N+1}\right)\right)_{k\in[\![1,N]\!]}$ pour $p \in [\![1, N]\!]$ fixé. Alors on trouve les valeurs propres $\lambda_p = 2\left(1 - \cos\left(\frac{p\pi}{N+1}\right)\right)$. En particulier, cette matrice est inversible.
3. Parfois on étudie une solution d'une équation différentielle en $\pm\infty$. Par exemple, on cherche une certaine trajectoire qui part d'un point instable et va jusqu'à un point d'équilibre stable.
   On connaît des méthodes pour montrer qu'on converge vers le point stable. Pour le point en $-\infty$, on écrit **l'équation rétrograde**. Ainsi le point instable devient stable et on reste en terrain connu.

• Les méthodes de tir :
Il est bon de savoir quoi faire quand on remplace une condition type Cauchy-Lipschitz par une condition type Dirichlet. Par exemple, on veut la solution de $u'' = f(u, u')$ avec $u(0) = u(1) = 0$.
L'idée est de poser $F(\theta)$ qui envoie θ sur $x_\theta(1)$ où x_θ est l'unique solution donnée par Cauchy Lipschitz avec conditions initiales $u(0) = 0$ et $u'(0) = \theta$.
On peut alors chercher les zéros de F par des méthodes de type Newton.

• Les problèmes raides :
Parfois quand les coefficients d'une équation différentielle ordinaire sont trop petits ou trop grands, les méthodes numériques ne convergent plus aussi bien, car il faut prendre un pas tout petit pour ne pas faire exploser la solution.
Un exemple typique d'une telle équation est $\varepsilon u' = -u$. La solution exacte est $u(t) = e^{-t/\varepsilon}u_0$.
Le schéma d'Euler explicite donne $u_{n+1} = \left(1 - \frac{\tau}{\varepsilon}\right)u_n$. Donc si le pas de temps τ n'est pas inférieur à ε, le schéma diverge !
Pour ce genre de problèmes, la stabilité est une question importante. Ici il sera plus intelligent de faire un schéma d'Euler implicite par exemple.
On peut trouver plus de détails dans le Filbet.

• Les pièges :

1. Dans le théorème de sortie de tout compact, il faut bien identifier les compacts ! Rester dans les compacts de $\Omega \subset \mathbb{K}^n$ veut dire être borné seulement si Ω est simplement connexe (n'a pas de trous).

• Les méthodes numériques associées :

1. Pour étudier les méthodes numériques, il vaut mieux prendre les définitions horribles du Demailly et les noter quelque part. Si le jury demande de présenter les notions de consistance/stabilité/convergence, le faire sur Euler explicite.
2. La méthode d'Euler explicite : la plus simple, la moins stable mais d'ordre 1 quand même. Il faut savoir prouver vite fait qu'elle explose.
3. La méthode RK4 (par ♡) :
   Elle n'est pas trop dur à apprendre (dans le cas autonome) et fait un peu plus classe que Euler explicite...
   Voici la version autonome, on peut trouver la version générale dans le Rappaz-Picasso.

```
function y=RK4(f,y0,T,N)
    y(:,1)=y0
    h=T/N
    for i=1 : N
        yi=y(:,i)
        k1=f(yi)
        k2=f(yi + k1*h/2)
        k3=f(yi + k2*h/2)
        k4=f(yi + k3*h)
        y(:,i+1)=yi + (h/6)*(k1 + 2*k2 + 2*k3 + k4)
    end
endfunction
```

4. Les méthodes implicites : Euler implicite (ordre 1, stable), point milieu (ordre 2, c'est une méthode symplectique !), trapèzes (ordre 2)... Si rien n'est précisé, on fait un point fixe rapide à chaque itération pour approximer le prochain terme de la suite. Sinon, on fait un Newton.

Chapitre 7

Équations aux dérivées partielles

La plupart des textes de modélisation traitent d'EDP. L'étude classique est de prouver l'existence et l'unicité d'une solution à une EDP puis de la modéliser avec un schéma numérique. On prouve alors la consistance/la stabilité/la convergence du schéma. On définira tout ça avec l'équation de la chaleur.

D'abord il faut tout de même parler de classification des EDP. Elle permet de classifier certaines des EDP d'ordre 2.
On note A la matrice contenant les coefficients devant les differentielles d'ordre 2.
Par exemple pour $\frac{\partial^2\Phi}{\partial x^2} + \frac{\partial^2\Phi}{\partial x\partial y} + \frac{\partial^2\Phi}{\partial y^2}$, on a la matrice $A = \begin{pmatrix} 1 & 1/2 \\ 1/2 & 1 \end{pmatrix}$.
Alors si toutes les valeurs propres sont de même signe et non nulles, l'EDP est dite elliptique (équation de Laplace). Si elles sont toutes non nulles et une est de signe opposée, l'EDP est dite hyperbolique (équation des ondes). Enfin si toutes les valeurs propres sont non-nulles sauf une et qu'elles sont de même signe, alors l'EDP est dite parabolique (équation de la chaleur).
Certaines propriétés de régularisation, de diffusion à l'infini sont caractéristiques de ces équations. C'est pourquoi on a coutume de dire que les EDP de transport sont hyperboliques malgré leur ordre 1. Leur comportement ressemble à celui des équations hyperboliques (termes en $x - ct$, propagation à vitesse finie...).

7.1 Équation de la chaleur

7.1.1 Modélisation physique

Dans toutes les modélisations, il faut rajouter la régularité des fonctions, de manière à justifier le sens des EDP écrites, et des calculs faits entre deux (dérivée sous l'intégrale,...).

On étudie la température u (ou la concentration chimique,...) dans un domaine U et on prend $V \subset U$ un sous-domaine. Alors si on note F la densité de flux, on a

$$\frac{\partial}{\partial t}\int_V u\mathrm{d}x = -\int_{\partial V} F.\vec{n}\mathrm{d}S.$$

Cette expression veut juste dire que la variation de quantité de u est le bilan de ce qui sort et ce qui rentre dans le domaine.
On a alors toujours une loi phénoménologique magique (ici la loi de Fourier) qui dit $F = -a\nabla u$ avec $a > 0$.
On a donc par Green-Ostrogradsky :

$$\int_V \frac{\partial u}{\partial t}\mathrm{d}x = \int_{\partial V} a\nabla u.\vec{n}\mathrm{d}S = \int_V a\,\mathrm{div}(\nabla u)\mathrm{d}x = \int_V a\Delta u\mathrm{d}x.$$

Comme V est arbitraire, il vient sur U :

$$\frac{\partial u}{\partial t} - a\Delta u = 0.$$

La constante a est la conductivité thermique. Elle dépend du matériau. Si on a un matériau composite, elle peut varier, donc prudence !

7.1.2 Étude théorique de la solution

• Existence de la solution sur $\mathbb{R}^n$:
On passe l'équation en Fourier et l'équation devient

$$\partial_t \widehat{u} = -4\pi^2 |\xi|^2 \widehat{u}.$$

On trouve

$$u(x,t) = u_0 * K_t(x), \text{ avec } K_t(x) = \frac{\exp(-\frac{|x|^2}{4t})}{(4\pi t)^{n/2}}.$$

K_t est appelé le noyau de la chaleur.
Pour faire ces calculs, on peut trouver une solution élémentaire dans l'espace de Schwarz, c'est plus simple (voir Bony).
Par contre, l'unicité n'est pas vérifiée si on ne se fixe pas un cadre fonctionnel adéquat. L'unicité est fausse dans $\mathcal{C}_1^2(\mathbb{R}^n\times]0,\infty[)$ (voir Evans), mais elle est vraie dans $\mathcal{C}(\mathbb{R}^+,\mathcal{S}'(\mathbb{R}^n))$ (la transformée de Fourier marche bien mieux dans $\mathcal{S}'$.).

• Existence de la solution sur Ω borné :
Dans certains cas très particuliers, on peut séparer les variables et écrire la solution comme une somme de série de Fourier. (voir dans mes développements).
Dans le cas général, on a un problème très difficile dont l'issue dépend de la condition au bord que l'on s'est donné. Il y a les conditions de Dirichlet où on fixe la fonction au bord, et les conditions de Neumann où on fixe la valeur de la "dérivée" au bord.
Pour résoudre tout cela, on utilise les espaces de Sobolev et notamment l'espace H_0^1 qui détermine tout de suite la valeur de notre fonction au bord.
Le seul vrai problème qu'il faut savoir résoudre le jour de l'oral est plutôt celui du Laplacien. On étudiera donc ça plus tard. Mais il faut savoir que l'on peut aussi faire des Sobolev avec l'équation de la chaleur ! La dérivée devient juste une condition de type Neumann.

• Principe du maximum (Ω borné) :
Voir Evans

7.1.3 La méthode des différences finies

• Schéma aux différences finis :
on prend l'équation, on se donne des points à peu près bien répartis dans l'espace et on approxime les dérivées par... des différences finies. Par exemple, u'' est approximé par $\frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2}$ pour une répartition uniforme des x_i dans $[0,1]$ avec h l'écart entre deux x_i consécutifs.

→ Schémas d'Euler :
Pour Euler explicite, on approxime l'équation par

$$\frac{u(x_i, t_{n+1}) - u(x_i, t_n)}{\tau} - \frac{u(x_{i+1}, t_n) - 2u(x_i, t_n) + u(x_{i-1}, t_n)}{h^2} = 0.$$

Ainsi le schéma est donné par $u^{n+1} = (I + \frac{\tau}{h^2} A_\Delta) u^n$.
Pour Euler implicite, on approxime par

$$\frac{u(x_i, t_{n+1}) - u(x_i, t_n)}{\tau} - \frac{u(x_{i+1}, t_{n+1}) - 2u(x_i, t_{n+1}) + u(x_{i-1}, t_{n+1})}{h^2} = 0.$$

Le schéma est $(I - \frac{\tau}{h^2} A_\Delta) u^{n+1} = u^n$. La matrice est bien inversible car les valeurs propres de $(I - \frac{\tau}{h^2} A_\Delta)$ sont toutes positives strictement.

→ Schéma de Crank–Nicolson :
Les schémas d'Euler sont du type $\frac{u_i^{n+1} - u_i^n}{\tau} = f(u^n)$ ou $\frac{u_i^{n+1} - u_i^n}{\tau} = f(u^{n+1})$. Le but des θ-méthodes est de produire un schéma du type

$$\frac{u_i^{n+1} - u_i^n}{\tau} = \theta f(u^n) + (1-\theta) f(u^{n+1}).$$

Pour $\theta = \frac{1}{2}$, on a le schéma de Crank–Nicolson.

→ Problèmes au bord :
Dans le cas Dirichlet, il faut fixer les valeurs de la solution aux bords.
Si la condition est de type Neumann, il faut refaire une approximation sur le bord pour dire que la dérivée est fixée. Par exemple, si la condition est $u'(L) = 0$, alors on peut approximer cela par $\frac{u_{J+1} - u_J}{dx} = 0$.
La matrice de discrétisation gagne en taille et la dernière ligne change :

$$A = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{pmatrix}$$

L'étude est alors différente et si l'approximation est aussi simpliste que celle que l'on a fait ici, alors l'ordre du schéma peut baisser ! Il faut être prudent avec ces conditions.

7.1.4 Étude des schémas numériques

Pour étudier les schémas associés à l'équation de la chaleur, il est nécessaire de connaître la matrice du Laplacien et ses valeurs propres $\lambda_p = 2\left(1 - \cos\left(\frac{p\pi}{n+1}\right)\right)$ associées aux vecteurs propres $x_p = \left(\sin\left(\frac{kp\pi}{n+1}\right)\right)_k$.

$$A_\Delta = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}$$

Parfois on étudie son opposé, mais il faut au moins garder en tête qu'elle est définie négative ou définie positive selon le cas !

• **L'erreur de consistance** (ou de troncature) est l'erreur commise en remplaçant l'opérateur différentiel par l'opérateur discrétisé. Ce faisant, il ne faut pas oublier que la solution exacte vérifie l'équation étudiée !
Pour l'équation de la chaleur avec Euler explicite en dimension 1, on a

$$\varepsilon_i^n = \frac{u(x_i,t_{n+1})-u(x_i,t_n)}{\tau} - \frac{u(x_{i+1},t_n)-2u(x_i,t_n)+u(x_{i-1},t_n)}{h^2}.$$

Le schéma est dit **consistant** si $\max_n \|\varepsilon^n\|_\infty \underset{\tau,h\to 0}{\longrightarrow} 0$. Attention les pas de temps et d'espace varient, donc n et i aussi ! Ils peuvent aller vers l'infini.
Si il existe $C>0$ tel que $\max_n \|\varepsilon^n\|_\infty \leqslant C(\tau^p + h^q)$ alors la méthode est dite d'ordre p en temps et q en espace.
Pour Euler explicite ou implicite, on montre facilement que l'on a un ordre de 1 en temps et de 2 en espace.

• La **stabilité** du schéma exprime le fait que l'accumulation des erreurs commises en remplaçant $u(x_i,t_n)$ par u_i^n reste bornée. Cette notion dépend de la norme choisie.
On dit que le schéma est stable pour la norme $\|.\|$ (l^2 ou l^∞ en pratique) s'il existe $C>0$ tel que pour τ et h asse petits, on aît $\|u^n\| \leqslant C\|u^0\|$ et ceci pour tout n et pour toute condition initiale u^0. On dit alors que le schéma est **inconditionnellement stable**. Si il faut que τ et h vérifie une relation - appelée **condition CFL** (Courant–Friedrichs–Lewy) - alors le schéma est dit **conditionnellement stable**.

A savoir : pour Euler explicite, la CFL est $\dfrac{\tau}{h^2} \leqslant \dfrac{1}{2}$ pour l^2 et l^∞.

En effet, si $A = \dfrac{1}{h^2}A_\Delta$, alors $u^{n+1} = (I+\tau A)u^n$. Si on suppose la CFL vérifiée, comme toutes les valeurs propres de $(I+\tau A)$ sont inférieures à 1 strictement, on a la stabilité l^2. Puis on remarque que u_i^{n+1} est une combinaison convexe des u_j^n, ce qui donne la stabilité l^∞.
Pour montrer que l'on n'a pas stabilité si la CFL n'est pas vérifiée, on montre que l'on a une valeur propre de module plus grand que 1 strictement. Alors en prenant pour u^0 un vecteur propre, on explose.
Euler implicite est inconditionnellement stable (stabilité l^2 en regardant les valeurs propres).

Dans d'autres cas, on ne pourra pas mettre le schéma sous une forme si simple (typiquement les équations hyperboliques), il convient alors d'utiliser la transformée de Fourier, en sachant que la norme l^2 est préservée. On fait les calculs normalement et on dit ensuite que c'est de la transformée de Fourier discrète, mais que tout marche pareil.
On pourra regarder le Di Menza à ce sujet.

• La **convergence** du schéma est vérifiée pour la norme $\|.\|$ si $\max_n \|v^n - u^n\| \underset{\tau,h\to 0}{\longrightarrow} 0$ avec v la solution exacte.
Le théorème de Lax donne quasiment dans tous les cas que le schéma converge si on a stabilité et consistance. Néanmoins, il faut le reprouver à chaque fois qu'on l'utilise !

7.2 Équation de Laplace

En stationnaire, l'équation de la chaleur devient une équation de type Laplace. Cela peut apparaître dans certaines modélisations physiques.
L'équation de Laplace apparaît dans de nombreux domaines de la physique : astronomie, électrostatique, mécanique des fluides, propagation de la chaleur, diffusion, mouvement brownien, mécanique quantique...

7.2.1 Résolution, espaces de Sobolev

• L'équation de Laplace $-\Delta u = f$ (muni de conditions type Dirichlet ou Neumann) que l'on peut généraliser en $-(pu')' + qu = f$ avec $p \in \mathcal{C}^1$, $q, f \in \mathcal{C}^0$, $p \geqslant \alpha > 0$ et $q \geqslant 0$.
Pour la résoudre **en dimension 1**, on peut le faire avec la théorie des EDO et on a une unique solution $\mathcal{C}^2$.
Pour la résoudre **en dimension 2**, on peut le faire avec la théorie des fonctions harmoniques. On se souviendra notamment que la partie réelle d'une fonction holomorphe est harmonique.
En dimensions supérieures, c'est difficile en général ! Si le domaine est simple (typiquement Ω est un pavé), alors on peut séparer les variables comme avec l'équation de la chaleur et tout marche de la même manière.
Si $\Omega = \mathbb{R}^n$ (donc on étudie le problème sans conditions de bord), on peut appliquer les méthodes de transformée de Fourier et trouver le noyau de Laplace comme nous avons fait pour l'équation de la chaleur.
Si Ω est juste borné, on passe à une formulation variationnelle dans les Sobolev, et en appliquant le **théorème de Lax-Milgram**, on trouve une unique solution (voir Brézis/Hirsch Lacombe).

• Théorème de Lax-Milgram
Soit H un espace de Hilbert, a une forme bilinéaire, continue et coercive (tend vers l'infini en l'infini), et l linéaire continue, alors il existe un unique élément u de H tel que

$$a(u, v) = l(v), \quad \forall v \in H.$$

De plus, si a est symétrique, alors u est le minimum sur H de la fonctionnelle J définie pour tout $v \in H$ par

$$J(v) = \frac{1}{2}a(u, v) - l(v).$$

• Inégalité de Poincaré
Si Ω est borné, alors il existe $C > 0$ tel que pour tout $u \in H_0^1$, $\|u\|_2 \leqslant C \|u'\|_2$. On peut le prouver par l'absurde avec le théorème de Rellich.
Si on se restreint aux u d'intégrale nulle, alors on a une nouvelle inégalité dite de Poincaré-Wirtinger.
En dimension 1, sur un intervalle de longueur d, on connaît la constante de l'inégalité de Poincaré-Wirtinger : c'est $C = \dfrac{d}{2\pi}$. Pour le démontrer, on utilise les séries de Fourier. On sait que $c_n(f') = \dfrac{2\pi}{d} i n c_n(f)$, puis Bessel-Parseval donne (comme $c_o(f) = 0$)

$$\|f'\|_2^2 = 2\pi \left(\frac{2\pi}{d}\right)^2 \sum |nc_n(f)|^2 \geqslant 2\pi \left(\frac{2\pi}{d}\right)^2 \sum |c_n(f)|^2 = \left(\frac{2\pi}{d}\right)^2 \|f\|_2^2 .$$

7.2.2 Méthode des éléments finis

On peut appliquer ici des méthodes de différences finies comme pour l'équation de la chaleur, mais on va changer un peu en présentant des éléments finis. Ceux-ci sont quasiment toujours présenté pour une équation de Laplace en pratique car il lui est associé une des formulations variationnelles les plus simples. Pour plus de détails, on pourra consulter le Allaire.
Les éléments finis sont une méthode complexe et il n'est pas grave de ne pas savoir tout démontrer dessus. Il faut juste en avoir entendu parler.

• Le but est d'approximer $u \in V$ défini de manière unique (grâçe à Lax-Milgram) par

$$a(u, v) = L(v), \quad \forall v \in V.$$

V est un espace de Hilbert (H_0^1 en pratique), a est bilinéaire continue coercive et L est linéaire continue.
Pour cela, on se donne des sous-espaces de dimension finie V_h de V où on peut trouver "facilement" u_h vérifiant

$$a(u_h, v_h) = L(v_h), \quad \forall v_h \in V_h.$$

• Le résultat important à retenir est le lemme de Céa. Il donne pour M la constante de continuité de a et ν sa constante de coercivité :

$$\|u - u_h\| \leqslant \frac{M}{\nu} \inf_{v_h \in V_h} \|u - v_h\| .$$

Il est simple de le prouver. Mettre la preuve dans une présentation est assez enrichissant.

• La seule chose à faire à présent est de trouver de bons espaces V_h où on peut calculer efficacement u_h.
En dimension 1, pour résoudre le problème du laplacien sur $[0, 1]$, on peut se donner à n fixé, $h = \dfrac{1}{n+1}$ et la subdivision (x_i) associée, l'espace

$$V_h = \{v \in \mathcal{C}^0, v_{|[x_i, x_{i+1}]} \in \mathbb{P}_1\}.$$

On se donne alors une bonne base et on peut se ramener à la résolution d'un système du type $K_h U_h = b_h$ avec K_h la matrice de discrétisation du laplacien, et b_h un vecteur constitué d'intégrale à approcher.

7.3 Équation de transport

7.3.1 Modélisation et heuristique physique

On fait le raisonnement en une dimension.
Prenons par exemple un nuage qui se promène (en 1D, ouioui). On note $u(x,t)$ la densité de masse du nuage au point x et au temps t. On étudie un espace infinitésimal $[x, x+dx]$, alors la masse de ce bout de nuage est

$$m(t) = \int_x^{x+dx} u(y,t)\mathrm{d}y.$$

On note $c(x,t)$ la vitesse du nuage au point x et au temps t, alors la différence de masse entre t et $t+dt$ est la différence de la masse de nuage entrée dans $[x, x+dx]$ et de celle qui en est sortie. On a donc

$$m(t+dt) - m(t) = c(x,t)u(x,t)dt - c(x+dx,t)u(x+dx,t)dt.$$

D'où il vient

$$\frac{\mathrm{d}m}{\mathrm{d}t} = -\left[(cu)(.,t)\right]_x^{x+dx} = -\int_x^{x+dx} \frac{\partial(cu)}{\partial x}(y,t)\mathrm{d}y.$$

Or sous certaines hypothèses de régularité, on a

$$\frac{\mathrm{d}m}{\mathrm{d}t} = \int_x^{x+dx} \frac{\partial u}{\partial t}(y,t)\mathrm{d}y.$$

On en déduit

$$\int_x^{x+dx} \left(\frac{\partial u}{\partial t} + \frac{\partial(cu)}{\partial x}\right)(y,t)\mathrm{d}y = 0.$$

Donc en faisant tendre dx vers 0, on a l'équation suivante dite de transport

$$\frac{\partial u}{\partial t} + \frac{\partial(cu)}{\partial x} = 0.$$

Pour la trouver en dimensions supérieurs, il suffit d'utiliser la formule de Green-Ostrogradski pour obtenir

$$\frac{\partial u}{\partial t} + \mathrm{div}_x(cu) = 0.$$

7.3.2 Résolution théorique

Le problème classique de transport est

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) + c(t,x).\nabla_x u(t,x) = a(t,x)f(t,x) + b(t,x), & (t,x) \in \mathbb{R}\times\mathbb{R}^d \\ u(0,x) = u_0(x), & x \in \mathbb{R}^d \end{cases}$$

On reconnaît l'équation précédente en utilisant $\mathrm{div}_x(cu) = c.\nabla_x u + \mathrm{div}_x(c)f$.
On a alors le théorème suivant :

Théorème.

Si u_0, a, b et c sont $\mathbb{C}^1$ et si de plus c est **globalement lipschitzienne**, alors le problème de transport a une unique solution donnée par

$$u(t,x) = u_0(X(0,t,x))\exp\left(\int_0^t a(s,X(s,t,x))\mathrm{d}s\right) + \int_0^t b(s,X(s,t,x))\exp\left(\int_s^t a(\tau,X(\tau,t,x))\mathrm{d}\tau\right)\mathrm{d}s.$$

où X est l'unique solution de

$$\begin{cases} \dfrac{\partial X}{\partial t}(s,t,x) = c(t,X(s,t,x)), & \forall t \in \mathbb{R} \\ X(s,s,x) = x \end{cases}$$

Les conditions a et b de classe $\mathbb{C}^1$ ne sont pas inutiles. En effet, pour dériver en t la fonction u, il faut aussi dériver sous l'intégrale car $a(s, X(s,t,x))$ et $b(s, X(s,t,x))$ dépendent de t.

Pour démontrer cela, il suffit de vérifier que l'expression du théorème vérifie l'équation de transport et il faut utiliser la propriété de groupe vérifiée par le flot X.
La fonction X est appelée la caractéristique et on doit juste utiliser le fait que la solution de l'équation homogène est en fait constante sur les caractéristiques (càd $u(t, X(0,t,x)) = u(0, X(0,0,x)) = u_0(x)$ pour tout t).
Puis on trouve l'expression générale par la méthode de variation des constantes.

On retiendra notamment la solution de l'équation homogène :

$$u(t,x) = u_0(X(0,t,x)).$$

Et si c est constant, alors on a simplement :

$$u(t,x) = u_0(x + ct).$$

Il est aussi intéressant d'étudier cette équation pour des vitesses seulement localement lipschitziennes, ou sur un domaine borné. Il faut alors vérifier que l'on a bien une solution, et sous quelles conditions.

7.3.3 Résolution numérique

On fait des schémas aux différences finies. Pour cela, on étudie l'équation de transport en 1D sur un domaine borné du type $[0,T] \times [-L, L]$ et on se donne une subdivision régulière avec un pas de temps τ et un pas d'espace h. On note (x_j) et $(t_n)_n$ nos subdivisions.

Considérons d'abord le cas où c est constant.
Le but est que l'on calcule nos données dans le même sens que vont les caractéristiques. C'est pourquoi on utilise deux schémas en pratique, que l'on choisit suivant le signe de c.

- Le schéma décentré à gauche/upwind :

$$u_j^{n+1} = u_j^n - c\frac{\tau}{h}(u_j^n - u_{j-1}^n)$$

Ce schéma marche pour des vitesses c **positives** et si la condition CFL $\left|c\frac{\tau}{h}\right| < 1$ est vérifiée (condition de stabilité).
Le problème pour programmer ce genre de schémas est sur les bords. En effet, on est censé calculer ce schéma pour $j \in \mathbb{Z}$, mais comme on ne le fait que sur un nombre fini de termes, il faut faire attention à enlever les termes de bords. Pour faire cela simplement, on peut juste écrire la matrice A telle que $u^{n+1} = Au^n$.
Voici un exemple de programmation du schéma décentré gauche (DFg) :

```
function u=schemagauche(c,u0,T,L,Nt,Nx)
    dt=T/Nt;dx=L/Nx
    u(:,1)=u0(-L:dx:L)
    A=(1-c*dt/dx)*diag(ones(2*Nx+1,1))+c*dt/dx*diag(ones(2*Nx,1),-1)
    for n=1:Nt
        u(:,n+1)=A*u(:,n)
    end
endfunction
```

- Le schéma décentré à droite/downwind :

$$u_j^{n+1} = u_j^n - c\frac{\tau}{h}(u_{j+1}^n - u_j^n)$$

Ce schéma marche pour des vitesses c **négatives** et si la condition CFL $\left|c\frac{\tau}{h}\right| < 1$ est vérifiée.

- Schéma décentré variable :

A présent, on peut traiter le cas où c est une fonction non constante : il suffit de choisir l'un ou l'autre schéma suivant le signe de c.

$$u_j^{n+1} = \begin{cases} u_j^n - c(x_j, t_n)\frac{\tau}{h}(u_j^n - u_{j-1}^n) \text{ si } c(x_j,t_n) \geqslant 0 \\ u_j^n - c(x_j, t_n)\frac{\tau}{h}(u_{j+1}^n - u_j^n) \text{ si } c(x_j,t_n) < 0 \end{cases}$$

• Pourquoi ne pas prendre le schéma centré ?
Le schéma centré est défini par

$$u_j^{n+1} = u_j^n - c(x_j, t_n)\frac{\tau}{2h}(u_{j+1}^n - u_{j-1}^n)$$

Ce schéma ne doit surtout pas être utilisé car il est instable !
Pour le voir, il faut décomposer u_0 en série de Fourier et regarder ce que donne le schéma.
Si $u_0 = \sum_{m\in\mathbb{Z}} \alpha_m e^{imx}$, alors

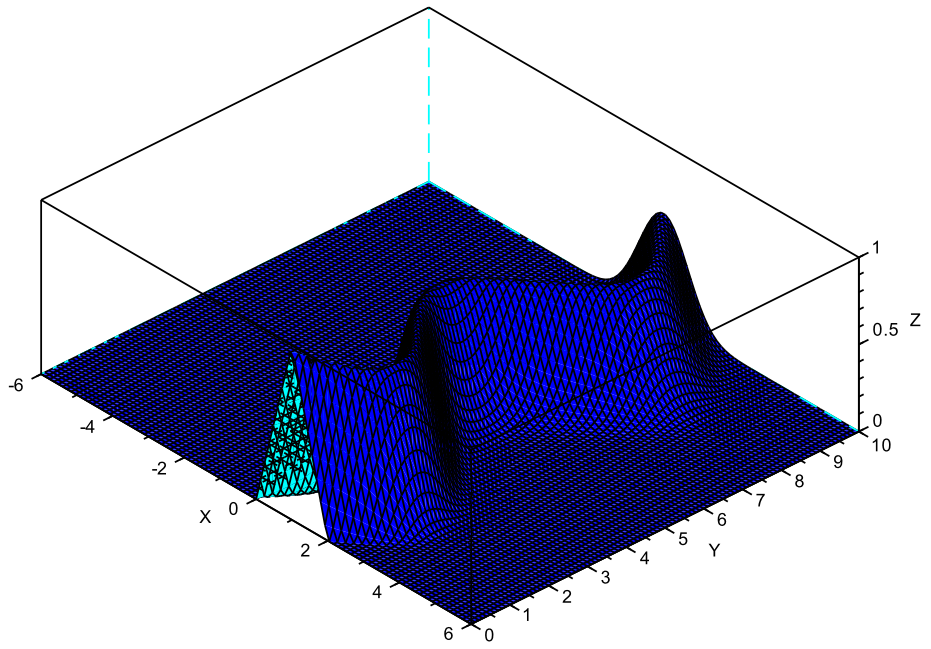
$$u_j^n = \sum_{m\in\mathbb{Z}} \alpha_m e^{imjh}\left(1 - \frac{c\tau}{h} i\sin(mh)\right).$$

Or $\left|1 - \frac{c\tau}{h} i\sin(mh)\right|^2 = 1 + \left(\frac{c\tau}{h}\sin(mh)\right)^2 > 1$, donc l'approximation numérique calculée explose en norme l^∞ et l^2 (par Bessel-Parseval).
On peut trouver tous les détails sur la stabilité dans le Rappaz-Picasso.

• La fonction à transporter u_0 est souvent prise comme une "fonction chapeau". Attention à lui permettre de prendre un vecteur.

```
function y=u0(x)
    for i=1:length(x)
        if x(i)>0 & x(i)<1 then
            y(i)=x(i)
        elseif x(i)<2 & x(i)>=1 then
            y(i)=2-x(i)
            else y(i)=0
        end
    end
endfunction
```

• Pour l'épreuve de modélisation, il faut programmer ces schémas, regarder leur comportement suivant la vérification ou non de la condition CFL. On peut tracer le logarithme de l'erreur en fonction du pas de temps τ pour montrer qu'ils sont d'ordre 1. Ensuite, on peut faire des animations où on voit le déplacement des courbes. Enfin un *plot3d* est une bonne idée si on veut vraiment visualiser la perte de matière au cours du temps.
Voilà un exemple que j'ai programmé. La vitesse est $c = \cos(t)$.



7.4 Équation des ondes

On pourra trouver des détails sur cette équation dans un pdf de Benjamin Boutin : `https://perso.univ-rennes1.fr/benjamin.boutin/Docs/WaveEquation2012.pdf`.

7.4.1 Modélisation physique

On va juste trouver l'équation en dimension 1. Cela marche de la même manière en dimensions supérieures. On pourra trouver quelques éléments de modélisation dans le Evans.

On se donne une corde évoluant dans un repère (O, x, y). Ses deux extrémités sont fixées en 0 et en 1. On pose alors $u(t, x)$ l'ordonnée de la corde au temps t et à l'abscisse x (en supposant que la corde ne fait pas de retours bizarres en arrière, etc...).
Si on note ρ la masse linéique de la corde (supposée constante ici), et si on note $T(t, x)$ la tension de la corde, alors le principe fondamental de la dynamique donne

$$(\rho dx)\frac{\partial^2 u}{\partial t^2} = T(t, x + dx) - T(t, x).$$

La loi (empirique) de Hooke donne

$$T(t, x) = k\nabla_x u(t, x).$$

On a donc bien l'équation des ondes (avec une condition de Dirichlet)

$$\frac{\partial^2 u}{\partial t^2} - \frac{k}{\rho}\frac{\partial^2 u}{\partial x^2} = 0.$$

On voit aussi cette équation en électromagnétisme. Pour la trouver, il suffit de se placer dans le vide et d'appliquer les lois de Maxwell.

$$\begin{aligned}\frac{\partial^2 E}{\partial t^2} &= \frac{1}{\mu_0\varepsilon_0}\frac{\partial(\mathrm{rot}(B))}{\partial t}\\ &= \frac{1}{\mu_0\varepsilon_0}\,\mathrm{rot}\left(\frac{\partial(B)}{\partial t}\right)\\ &= -\frac{1}{\mu_0\varepsilon_0}\,\mathrm{rot}(\mathrm{rot}(E))\\ &= -\frac{1}{\mu_0\varepsilon_0}(\nabla\,\mathrm{div}(E) - \Delta E)\\ &= \frac{1}{\mu_0\varepsilon_0}\Delta E\end{aligned}$$

Celle-ci modélise le comportement de rayonnement électromagnétique dans le vide. On interprète ce rayonnement comme une somme d'ondes électromagnétiques.

7.4.2 Résolution mathématique

- Si on se donne le problème de Cauchy suivant (en dimension 1) avec g de classe $\mathcal{C}^2$ et h de classe $\mathcal{C}^1$.

$$\begin{cases}\dfrac{\partial^2 u}{\partial t^2} - c^2\dfrac{\partial^2 u}{\partial x^2} = f\\ u(0, x) = g(x)\\ \dfrac{\partial u}{\partial t}(0, x) = h(x)\end{cases}$$

Alors on sait le résoudre.
Pour cela, on remarque que $\dfrac{\partial^2}{\partial t^2} - c^2\dfrac{\partial^2}{\partial x^2} = \left(\dfrac{\partial}{\partial t} - c\dfrac{\partial}{\partial x}\right)\left(\dfrac{\partial}{\partial t} + c\dfrac{\partial}{\partial x}\right)$.
On pose donc $v = \dfrac{\partial u}{\partial t} + c\dfrac{\partial}{\partial x}$. Et on résout juste $\dfrac{\partial v}{\partial t} - c\dfrac{\partial v}{\partial x} = f$.
En résolvant ces deux équations de transport, on trouve

$$u(t, x) = \frac{g(x + ct) + g(x - ct)}{2} + \frac{1}{2c}\int_{x-ct}^{x+ct} h(\sigma)\mathrm{d}\sigma.$$

Cette solution est $\mathcal{C}^2$ et elle est unique. Cette expression est nommée la formule de D'Alembert.

A partir de cette formule et de la méthode utilisée pour l'obtenir, on voit tous les liens entre l'équation des ondes et les équations de transport : vitesse finie de propagation, pas de régularisation, dépendance stable aux conditions initiales...

• On peut aussi la résoudre avec des séries de Fourier si on fait ça sur un domaine gentil (un pavé). Cette manière de faire est plus approprié si on met des conditions aux bords de type Dirichlet ou Neumann. Pour la méthode, on pourra s'inspirer de mon développement sur l'équation de la chaleur.

• On a conservation de l'énergie dans le cas homogène. Cela permet de prouver l'unicité avec les séries de Fourier.

• Bien sur, on peut aussi la résoudre sur l'espace entier en appliquant la transformée de Fourier.

7.4.3 Résolution numérique

Ce sont les mêmes méthodes et CFL que pour l'équation de transport. C'est juste plus compliqué. Il n'est pas nécessaire de savoir étudier ces schémas pour l'agrégation ; ceux pour les équations de transport suffisent.

Si on veut étudier la consistance de tels schémas, il faut retenir qu'il faut calculer $\frac{u_{n+1} - u_n}{dt}$. Cela rend les calculs plus complexes car on a un laplacien en temps dans l'équation...

En pratique, si on cherche une solution exacte simple pour les modélisations, on se donne une sinusoïde et on rajoute des coefficients comme on veut.

Chapitre 8

Optimisation

- Théorie :

1. Existence/unicité du minimum
   Pour l'existence du minimum, on peut utiliser que la fonction est continue sur un compact ou de la coercivité. Un minimum local devient global si on est convexe. Si on est strictement convexe, on a unicité. Si on est fortement convexe, on a existence ET unicité !
   Pour les caractérisations de la convexité pour des fonctions de $\mathbb{R}^n$, on peut voir le Hirriart-Urruty, Lemaréchal et le Allaire.
2. Dans le **cas sans contrainte**, le minimum est caractérisé par l'annulation de la dérivée.
3. Dans le **cas avec contrainte**, on a le théorème des extrema liés qui s'applique parfois. Dans le cas général, il y a les inéquations d'Euler ou les conditions de Kuhn-Tucker (voir Allaire).
   Pour ce type de problème, on utilise souvent des méthodes de gradient projeté ou de dualité. Elles ne sont pas exigibles au niveau de l'agrégation. Il faut juste savoir qu'elles existent.
4. Sur un Hilbert, on a les théorèmes de Stampacchia et de Lax-Milgram (que l'on peut appliquer au problème du laplacien, voir Brézis). On a aussi les moindres carrés (voir Rouvière).

- Méthodes numériques (tout dans Rouvière ou Demailly) :

1. Méthode de Newton-Raphson
   Pour trouver le minimum avec Newton, on cherche le zéro de $\nabla f = g$ (qui est bien une fonction de $\mathbb{R}^n$ dans $\mathbb{R}^n$). On suppose g de classe $\mathcal{C}^2$ et de dérivée ne s'annulant pas sur un voisinage du zéro a. On suppose connu une approximation grossière du zéro située dans le voisinage précédent (pas évident en théorie, mais marche toujours en pratique). Alors la méthode est définie récursivement par $x_{n+1} = \varphi(x_n)$ avec $\varphi : x \mapsto x - (\nabla g(x))^{-1}.g(x)$. On montre que le point fixe a est super-attractif, c'est à dire que $(x_n)_n$ converge et $\varphi'(a) = 0$.
   Pour montrer cela, on commence par faire des développements limités

$$\varphi(x) = a + \frac{1}{2}g'(a)^{-1}(g''(a)(x-a, x-a)) + o(\|x-a\|^2)$$

   Ainsi on a $\varphi'(a) = 0$ et $\|\varphi(x) - a\| \leqslant \frac{1}{2}(\|\varphi''(a)\| + \varepsilon(x-a)) \|x-a\|^2$.
   Donc en se plaçant sur un petit intervalle autour de a, on prend C constante telle que $\|\varphi(x) - a\| \leqslant C\|x-a\|^2$, alors $C\|\varphi(x) - a\| \leqslant (C\|x-a\|)^2$, donc par récurrence, on a

$$\|x_n - a\| \leqslant \frac{1}{C}(C\|x_0 - a\|)^{2^n}$$

   .
   Si on rajoute une hypothèse de convexité, alors peu importe où on prend le point de départ, la suite convergera.
   Si, au contraire, on enlève l'hypothèse $\mathcal{C}^2$, on aura toujours convergence mais elle ne sera plus quadratique.
   L'inconvénient de cette méthode est qu'il faut inverser une matrice. Si la dérivée est difficile à calculer, on peut l'approximer comme dans la méthode de la sécante, quitte à réduire un peu l'ordre. Un autre moyen d'approximer simplement la dérivée est juste de calculer $\dfrac{f(x+\sqrt{\%eps}) - f(x)}{\sqrt{\%eps}}$.
   Le point fort de cette méthode est sa convergence quadratique.

2. Méthodes de gradient
   L'idée est de poser la suite $x_{n+1} = x_n - \alpha_n \nabla f(x_n)$, puis de trouver le α_n qui convient. Il y en a trois à retenir : celle à pas constant, celle à pas optimal (voir mon développement dessus pour tous les détails !) et celle du gradient conjugué. Cette dernière n'est pas vraiment de la forme annoncée, car elle minimise f sur l'espace vectoriel engendré par les directions prises jusque-là à chaque itération.
   Ces méthodes sont d'ordre 1 et celle du gradient conjugué se finit en n étapes. Une manière classique d'étudier leurs performances est de les appliquer sur des formes quadratiques définies positives. On voit alors que la vitesse de l'algorithme dépend du conditionnement de la matrice, ce qui n'est pas terrible car dans ce cas là, trouver le minimum revient exactement à inverser une matrice (voir la section suivante pour ce problème).

   Voici un exemple de programmation de la méthode de gradient à pas optimal pour trouver le point minimisant $x \mapsto \frac{1}{2}(Ax, x) - (b, x)$.

```
function X=gradient_optimal(A,b,x0,N,tol)
    X(:,1)=x0
    i=1
    while (i<=N & (i==1 | norm(X(:,i)-X(:,i-1))>tol))
        dk=-(A*X(:,i)-b)
        rhok=norm(dk)^2/((A*dk)'*dk)
        X(:,i+1)=X(:,i)-rhok*(A*X(:,i)-b)
        i=i+1
    end
endfunction
```

   Il y a une chose à retenir sur le gradient à pas fixe : il faut que la tolérance soit supérieure au pas, sinon l'algorithme ne terminera surement pas : on oscillera autour du minimum sans pouvoir l'atteindre car le pas est trop grand.
   Pour plus de détails de convergence, on peut regarder le Allaire.

3. Moindres carrés
   Elle permet des minimiser la distance d'un élément à un espace vectoriel de dimension finie. On l'utilise sur un Hilbert. Cela revient à se donner une famille libre, puis à l'orthonormaliser. On a alors le projeté donné par la formule $p(f) = \sum (f, e_n) e_n$.
   Celle-ci peut se voir comme une application des méthodes de gradient. En effet, si on se donne $\{(x_i, y_i)\}$ les points à interpoler et n le degré voulu du polynôme interpolateur, alors cela revient à résoudre les équations normales ${}^tVVa = {}^tVy$ avec a l'inconnue et $V = (x_i^j)_{i \in [1,N], j \in [1,n+1]}$ la matrice de Vandermonde. On peut résoudre cela avec une méthode de gradient à pas optimal classique sur les fonctions quadratiques (ou juste résoudre le système linéaire obtenu, voir chapitre sur l'analyse numérique matricielle). On a alors le polynôme $p_n = a_0 + ... + a_n x^n$ qui approche le mieux (en norme L^2) les (y_i). Pour $n = 1$, on retrouve la régression linéaire.
   Le problème de résoudre les équations normales directement est que la matrice ${}^tVV$ a souvent un mauvais conditionnement (elle double celui de V.). Pour résoudre ce problème, on peut appliquer une décomposition QR à V, puis ${}^tVV = {}^tRR$ est beaucoup plus facile à inverser.
   Pour plus de détails, voir Rappaz, Picasso ou Filbet.

Chapitre 9

Culture physique/chimique/économique

Il est toujours bon de faire le lien entre nos modélisations, nos théorèmes et la réalité réelle de la vie véritable. Du coup, voici quelques éléments de culture à connaître.

• Physique :

1. **Le principe fondamental de la dynamique** consiste à faire le bilan des forces sur un système fermé. Si on étudie le mouvement d'un point matériel M de masse m sur lequel agissent les forces $\vec{F_i}$, alors
$$m\frac{\mathrm{d}^2M}{\mathrm{d}t^2} = \sum_i \vec{F_i}.$$
On peut bien sûr l'appliquer à un solide : si on note G son centre d'inertie, alors
$$m\frac{\mathrm{d}^2G}{\mathrm{d}t^2} = \sum_i \vec{F_i}.$$
2. **Les circuits électriques** permettent d'obtenir des EDO très facilement. Il suffit de connaître les relations idéales liant l'intensité et le courant passant par les dipôles classiques en convention récepteur.
Pour la bobine, $u = L\frac{\mathrm{d}i}{\mathrm{d}t}$; pour le condensateur, $i = C\frac{\mathrm{d}u}{\mathrm{d}t}$; pour la résistance, $u = Ri$.
3. Il est aussi bon de connaître vaguement **les lois de Maxwell**, ainsi que **la loi de conservation de la charge** $\mathrm{div}\left(\vec{j} + \frac{\partial\rho}{\partial t}\right) = 0$. Elles peuvent s'avérer utiles.
$$\begin{cases} \mathrm{div}(E) = \dfrac{\rho}{\varepsilon_0} \\ \mathrm{div}(B) = 0 \\ \mathrm{rot}(E) = -\dfrac{\partial B}{\partial t} \\ \mathrm{rot}(B) = \mu_0 j + \mu_0\varepsilon_0\dfrac{\partial E}{\partial t} \end{cases}$$

• Chimie :

1. On étudie pour expliquer une relation du type suivant
$$\nu_A A + \nu_B B \to \nu_C C + \nu_D D.$$
Les ν_X sont appelés les coefficients stœchiométriques.
Alors cela veut dire que pour créer ν_C molécule(s) de C et ν_D de D, on en consomme ν_A de A et ν_B de B. Donc si on note n_X la quantité de matière de X, on a
$$-\frac{1}{\nu_A}\frac{\mathrm{d}n_A}{\mathrm{d}t} = -\frac{1}{\nu_B}\frac{\mathrm{d}n_B}{\mathrm{d}t} = \frac{1}{\nu_C}\frac{\mathrm{d}n_C}{\mathrm{d}t} = \frac{1}{\nu_D}\frac{\mathrm{d}n_D}{\mathrm{d}t} = \frac{\mathrm{d}\xi}{\mathrm{d}t}.$$
La quantité ξ - définie au départ par $\xi(0) = 0$ - est appelée l'avancement de la réaction. C'est ce qu'il est bon d'approcher. Sa dérivée est appelée la vitesse molaire de réaction.
En pratique, on connaît la vitesse de réaction d'un ou de plusieurs éléments et cela donne des équations différentielles ordinaires que nous pouvons étudier.

2. **Le principe des états stationnaires** dit que lorsque que l'on a une réaction composée $A \to X \to B$, si X est très réactif, alors on peut supposer $x' = 0$ dans les équations.

• Économie :

1. **Le modèle d'entrée-sorties de Léontieff** vise à caractériser l'économie d'un pays en évaluant la quantité de biens qui doit être produite dans chaque secteur et leur prix afin d'assurer une situation d'équilibre.
   On commence par partitionner l'économie en n secteurs d'activités (énergie, agriculture, chocolateries, ...). Chaque secteur produit une quantité $x_i \geqslant 0$ du bien numéro i, mais pour faire cela, il consomme une quantité $a_{i,j} \geqslant 0$ du bien j. Enfin, comme le peuple a besoin de tous ces trucs, on a un vecteur demande du public $c \geqslant 0$.
   Ainsi si on fait un bilan de ce qui est produit et de ce qui est consommé, on a

$$Ax + c = x.$$

   Si on connaît c et A, alors le problème est de trouver x le vecteur production. En posant $B = I - A$, on obtient l'équation classique

$$Bx = c.$$

   Le problème revient donc à inverser la matrice B.
   On peut l'inverser avec des méthodes type LU/pivot de Gauss, mais comme n est souvent grand, on préférera des approximations du type gradient à pas optimal ou autre.

• Dynamique des populations :

1. LE modèle à connaître par cœur est le système différentiel de Lotka-Volterra. Il y a tous les détails sur sa périodicité, son intégrale première dans le FGN analyse 4.
2. Si on rajoute un terme en $(-x_i^2)$ dans Lotka-Volterra, on obtient les équations de compétition de Lotka-Volterra. On a des équations du type logistique, ce qui donne des comportements complexes. On peut encore compliquer cela en mettant des coefficients périodiques.