Travaux pratiques sous Maple™ et Matlab®

Bertrand Meyer & Lisl Weynans

8 septembre 2008

Modalités des contrôles des connaissances :

Examen : coefficient 0,4
Note d'exposé : coefficient 0,3
Contrôle continu : coefficient 0,3.

Organisation :

1. Les sujets sont à traiter par groupe de deux au plus tard une semaine à l'issue de la séance. Les travaux rendus hors délais ne seront pas corrigés.
2. Vous êtes encouragés à réfléchir aux problèmes par vous même mais rien ne vous empêche de chercher vos réponses dans la littérature. Quitte à utiliser des sources documentaires, prenez l'habitude de travailler avec des ouvrages imprimés, seules sources autorisées pour les concours de la fonction publique.
3. *Donnez vos sources* lorsque vous citez et reformulez avec vos mots autant que possible.
4. Vous pouvez discuter des sujets entre étudiants mais rédaction des réponses doit être *propre à chaque binôme*.
   Le plagiat ou la triche est une pratique dont les condamnations peuvent être lourdes : 5 ans d'interdiction de se présenter à un diplôme ou concours de l'Etat.
5. N'hésitez pas à poser vos questions par courrier électronique pendant la semaine ou visiter vos chargés de TP dans leur bureau.
6. Indiquez vos noms de famille dans le fichier et dans le nom de fichier.

Conseils :

1. Faites des réponses *concises*.
2. Vérifiez vos programmes en donnant plusieurs exemples pertinents d'utilisation de vos fonctions, et en vous donnant les moyens de le vérifer. Comparez lorsque c'est possible votre programme aux réponses des fonctions du logiciel. Exemple : Calculer `> gauss(A,b) - A\b` est le meilleur moyen de tester un pivot de Gauß. Calculer `expand(u*p+v*q-pgcd)` est le meilleur moyen de vérifier une relation de Bezout.
3. *Commentez* abondamment vos programmes, expliquez ce que vous faîtes. Les comptes rendus doivent être lisible sans le sujet et doivent être prêts à l'emploi.

4. Commencez vos feuilles de travail par `restart` en Maple ou `clear` en Matlab.
5. Dans Matlab, il faut systématiquement enregistrer les fonctions dans des fichiers indépendants. Constituez un fichier principal *TPn_NomDeFamille.m* pour répondre à l'ensemble du sujet.

Table des matières

1 Premier pas avec Maple™

Ce document a été conçu pour vous guider lors d'une première utilisation de Maple™ au CREMI (sous environnement linux) dans le cadre du module MHT304. Vous pouvez également utiliser la version de Maple™ installée à l'espace α (sous windows). Il reprend l'essentiel d'une version plus complète qui est disponible à l'espace α et de sujets de T.P. rédigés par J. Y. Boyer pour le précédent module MAP304.

On peut travailler avec Maple™ 10 de deux façons : soit en utilisant le mode *Document*, soit en utilisant le mode *Worksheet*. Le mode *Document* permet d'utiliser le logiciel de façon relativement intuitive, mais seule une partie restreintes des fonctions Maple™ sont disponibles, celles des données par les menus et les palettes. Le mode *Worksheet* donne accès à toute la puissance de Maple™ et reprend la syntaxe des versions précédentes. En particulier, toute feuille écrite dans une version antérieure doit pouvoir s'ouvrir et être utilisée dans ce mode. Nous ne donnons ici qu'une initiation à Maple™ 10 en mode *Worksheet* ; pour une introduction au mode document, on pourra consulter le cours ou le manuel de référence par les menus : *Help* →*Manuals, ...* →*Manuals* →*User Manual*.

Exercice 1.1 Pour commencer votre travail,

1. Connectez vous en donnant votre nom et votre mot de passe (comme pour vous connecter sur Ulysse).
2. Sur la fenêtre du terminal, créez un répertoire *ModuleMHT304* et un sous répertoire *TPMaple10* en utilisant la commande `mkdir nom_repertoire` à l'endroit où vous souhaitez le créer. Pour se déplacer d'un répertoire à l'autre, on utilise la commande `cd ..` ou `cd nom_repertoire`. Par ailleurs, l'instruction `ls` permet de connaître le contenu du répertoire.
3. Ouvrez une session Maple™ en tapant `xmaple10`.
4. Enregistrez cette session en
   - cliquant sur l'icône disquette,
   - en cliquant sur le bandeau de la fenêtre qui s'ouvre,
   - en cliquant dans la colonne de droite sur *TPMaple10* puis ok.
   - Dans la colonne de gauche, écrivez à la place de l'étoile `TD1.mws` puis en cliquant sur ok.

Vérifiez que le nom du fichier est inscrit dans le bandeau.

Ouvrir une page en mode *Worksheet*

Lorsqu'on lance Maple™ 10, la page qui s'ouvre par défaut est une page en mode document. Ici, nous allons travailler en mode *Worksheet*.

Pour ouvrir une *feuille de travail* (*Worksheet*) après avoir lancé Maple™ 10, exécutez *File* →*New* →*Worksheet Mode*. La page qui s'ouvre doit commencer par une prompt (>) rouge (sinon, c'est une page en mode commande ou une zone de texte).

Si l'on veut que la page qui s'ouvre lorsqu'on lance Maple™ 10 soit toujours une feuille de travail au lieu d'une page en mode document : exécutez sous linux (au CREMI) ou sous windows (à l'espace α), *Tools* →*Options* →*Interface*

→*Defaut format for new worksheet* →*Worksheet*, ou bien sous Mac OS X, *Maple10* →*Preference* →*Interface* →*Defaut format for new worksheet* →*Worksheet*. Dans tous les cas, cliquez sur *Apply globally* pour que ceci s'applique désormais toujours.

Dans le bandeau supérieur (première ligne), on trouve, entre autres, les *symboles T et* [>. Si l'on clique sur T, on ouvre une zone de texte, on peut écrire des phrases et y placer des symboles mathématiques (mais ils sont inertes, les calculs ne sont pas effectués). Si on clique sur [>, on ouvre une ligne Maple™ . Elle débute par un prompt (>) rouge à la suite duquel on écrit les instructions qui seront interprétées.

Sur le côté gauche de la fenêtre, on ouvre les *palettes* : *Symbol recognition, Expression, Units, ...*. On ouvre les palettes en cliquant sur le triangle. La palette *Expression* contient des icônes permettant de calculer des intégrales, des sommes, des produits, des dérivées, des limites, des valeurs de certaines fonction, afficher des fonctions ; la palette de *Common symbols* permet d'appeler les nombres π, e, i, ...ou certains symboles. Si on clique sur une de ces icônes, elle s'affiche dans la page à l'endroit où se trouve le curseur. On peut aussi les faire glisser à l'aide de la souris. Il est recommandé cependant d'apprendre par cœur les noms des commandes plutôt que de les rechercher dans la palette.

En mode *worksheet*, après un prompt (>) rouge, on peut travailler en mode *math* ou en mode *text* (voir section suivante), ceci est indiqué sur la ligne 2 du bandeau, à gauche. Pour passer de l'un à l'autre on clique sur Math ou sur Text ou sur la touche F5 du clavier.

En mode *Math* les instructions peuvent s'afficher soit avec des icônes (par exemple $\int_a^b f \, \mathrm{d}x$, pour le calcul d'une intégrale définie) soit par une ligne de commande (par exemple pour le calcul d'une intégrale définie : `int(f,x=a..b)` ; c'est l'instruction que l'on trouve en jaune lorsque l'on pointe sur une icône des palettes). Utiliser ce mode est très fortement déconseillé lorsque l'on souhaite utiliser sérieusement Maple™ , car dans ce cas, Maple™ n'affiche pas tout et il n'est pas possible de réellement contrôler les instructions que l'on a fourni en commande.

En mode *Text* on affiche uniquement des lignes de commandes (par exemple pour le calcul d'une intégrale définie : `int(f,x=a..b)`). Ce mode est le plus sûr et limite les erreurs.

Pour enregistrer la feuille de travail cliquer sur : *File* →*Save* ou bien, sur la première ligne du bandeau cliquer sur la disquette (Save the active file). Pensez à *sauvegarder régulièrement.*

Pour fermer le document, exécutez *File* →*Close Document* ou pour fermer Maple™ 10, *Maple 10* →*Quit Maple 10*. Pour clore la session, cliquez sur *Démarrer* →*Fermer la session.*

Utilisation de Maple™ en mode *Text* [C-Maple Input]

Ouvrez une page en mode worksheep et vérifiez que lorsque l'on place le curseur après un prompt rouge, on a dans le bandeau (deuxième ligne, à gauche) : *Text - C Maple Input* ou alors cliquer sur Text (ou sur F5).

Pour que lorsqu'un prompt se crée, on soit toujours en position *Text - C Maple Input*, executez avec linux ou windows (c'est le cas de l'espace alpha)

Tools →Options →Display →Input display →Maple notation ou avec Mac OS X, *Maple 10 →Preference →Display →Input display →Maple notation*.

Pour la sortie, il vaut mieux garder 2-D Math Notation. (Pour cela vérifier que dans la fenêtre précédente on a : *Output display -¿ 2-D math notation*) Cliquez sur *Apply to session* (pour que ceci s'applique pour cette page) ou sur *Apply Globally* (pour que ceci s'applique pour cette page et les prochaines).

Apres le prompt les instructions sont en rouge. (S'il n'y a pas de prompt, c'est une zone de texte.)

On peut soit utiliser les palettes, en cliquant sur une icône (ou en la faisant glisser) elle affiche à l'endroit du curseur la commande maple correspondante (et non pas une icône), soit écrire directement après le prompt la ligne de commande si on la connait. Ceci explique la terminologie 1-D.

La ligne doit se terminer par un point virgule (ou par un double point si on ne veut pas que le resultat s'affiche).

Si on place le curseur à la suite d'un résultat (en bleu) on passe en Math-2D (si ce choix est celui indiqué dans le menu preference sinon faire : *Tools →Options →Display →Output display →2-D Math Notation*) Dans ce cas, en plaçant le curseur à la suite d'un résultat (en bleu) on peut utiliser le menu déroulant.

Lorque l'on hésite sur une commande maple, on peut compléter le début d'une ligne de commande maple par ctrl+Space (sous windows, c'est le cas de l'espace alpha) ou ctrl+Shift+Space (sous Linux) ou pomme+Shift+Space (sous mac).

Exercice 1.2 1. Faites en sorte que tous les nouveaux documents que l'on ouvrira par la suite soit en mode *Worksheet* et que lorsqu'on crée un prompt, on soit en mode *Text*. Cliquez sur les symboles T et > pour voir ce qui se passe.

Exercice 1.3 1. Tapez sur la feuille de calcul les exemples suivants pour étudier le rôle de `;` `:` `:=` `%` `#` `restart`

```
> P:= 1*2*3*4*5;
> S:= 1+2+3+4+5:
> P+S;
> S;
> A:= % ; b := %%% / 15 ;
> (P+S)/15; # on vérifie que b=(P+S)/5
> c = 2+3;
> c; # Pourquoi n'a-t-on pas 15 comme réponse ?
> ?%
> p; P;
> restart;
> p; P; # Quel est le rôle de restart ?
```

2. Allez placer le curseur sur la ligne de P et remplacez P par `P :=1*2*3*4*5*6 ;` et de même `S := 1+2+3+4+5+6 ;`. Puis validez les lignes qui suivent.
3. Tapez les commandes suivantes et expliquez ce qui se passe

```
> u := < 1, 2, 3> ; v := < 4 | 5 | 6>;
> whattype(u); whattype(v);
```

```
> A := <<1, 0, -1> | <2, 3, 0> | <-1, 5, 3>>;
> A.u; u.A; A.v; v.A;
```

4. Tapez `plot(sin(x)/x, x=-12..12) ;` et expliquez ce qui se passe.

Il est plus élégant et plus clair d'ajouter un titre en tête de votre document et de présenter les exercices dans des sections distinctes. Pour cela, utilisez les instructions *Insert →Section*.

Exercice 1.4 En utilisant la palette à gauche ou le menu obtenu par clic droit,

1. Calculez la limite en $\frac{\pi}{4}$ de la fonction cosinus, puis une approximation à 10 chiffre du résultat (utilisez un clic droit sur le premier résultat). Lisez le fichier d'aide (`?evalf`). A quoi sert cette fonction ?
2. Calculez la limite en 0 de $\dfrac{1-\cos(x)}{x^2}$ (utilisez la commande `limit`).
3. Calculez l'intégrale suivante : $\displaystyle\int_{e^2}^{\pi} \frac{\mathrm{d}x}{x}$ (utilisez la commande `int`).
4. Calculez la dérivée par rapport à x de l'expression $\ln\left(\dfrac{1}{x^2}\right)$ (utilisez la commande `diff`) et évaluez-la en 2 (avec la commande `subs` ou `eval`).
5. Donnez une valeur approchée de $e+\pi$.
6. Calculez la somme des entiers de 1 à n et factorisez cette somme (cherchez `factor` dans l'aide). Evaluez la somme pour $n = 1234$. Cherchez `eval` et `subs` dans l'aide.
7. Calculez le quotient du produit des entiers pairs entre 1 et $2n$ par le produit des impairs entre 1 et $2n$. Evaluez ce quotient lorsque $n = 1000$ et donnez-en une valeur approchée. Calculez la limite de ce quotient lorsque n tend vers l'infini. Calculez la limite de ce même quotient multiplié par $\sqrt{n}$ lorsque n tend vers l'infini.
8. Calculez le module et l'argument des nombres complexes $1+i$ et $2+3i$ (utilisez un clic droit pour trouver les commandes). Donnez leur forme polaire.
9. Vérifiez que $j = e^{i2\pi/3}$ est une racine cubique de l'unité. Calculez $1+j+j^2$.
10. Tracez le graphique de la fonction $x \mapsto \cos\left(\frac{1}{x}\right)$ pour $x \in [-5,5]$ puis pour $x \in [-1,1]$.
11. Introduisez une fonction h définie par $h(x) = \cos(x^3)$ si $x < -3$, $h(x) = |x|$ si $-3 \leqslant x \leqslant 3$. Demandez le calcul de $h(-10\sqrt[3]{\pi})$, $h(-1)$, $h(0)$ et $h(10)$. Tracez le graphique de h
12. Résoudre les équations ou inéquations $x^2+x+1=0$, $x^3+x+1=0$, $x^2+3x+1>0$, $\sin(x)=0$, $x^5-2x^4+3x^3-2=0$ (chercher de l'aide sur la fonction `?solve`). Donnez des valeurs approchées dans le dernier cas.

---

2 Assignations, boucles logiques

MOTS CLÉS : Assignations, evaluation, instructions if, for, while.

Exercice 2.1 1. Entrez les commandes suivantes

```
> y := x^2;
> x := 2;
> y;
> z := x^2;
> x := 3;
> y; z;
```

2. Donnez une méthode pour libérer les affectations de x et y (pour l'instant on doit avoir $x = 3$ et $y = 9$).
3. Entrez l'expression `E := 3*x^2*y+x*y`. Les variables x, y et E sont-elles affectées (assigned) ?
4. Entrez les commandes suivantes et expliquez le résultat.

```
> E:= 'E';
> x:=0; y:=0; E;
```

5. Pouvez vous prévoir le résultat des commandes suivantes ?

```
> restart ;
> y:=x;
> x:=4;
> x+y+2*(y+1);
> 'x+y+2*(y+1)';
> restart;
> x:=y:  y:=4: y:=5: x;
> restart ;
> y:=4: x:=y: y:=5: y=6; x;
```

Les objets Maple™ sont représentés par des arbres dont la racine est étiquetée par le *type* et les branches sont les *opérandes*. L'instruction `op` permet de connaître les opérandes d'un objet Maple™, l'instruction `whattype` de connaître le type.

Exercice 2.2 1. Construisez l'arbre correspondant aux expression $A := x^3 + 5xy - cos(x + y)$ en utilisant les fonctions Maple™ appropriées.

2. Evaluez A lorsque $x + y = 10$ et $x = 10$ en utilisant les fonctions `subs` et `eval`.

Rappelons que les boucles conditionnelles se mettent en œuvre sous Maple™ avec la syntaxe (les passages à la ligne peuvent être omis)

```
if condition_1 then
conséquence_1 ;
elif condition_2 then
conséquence_2 ;
else
conséquence_j ;
end if ;
```

Exercice 2.3 (La boucle if) 1. Que permet de faire la suite d'instruction suivante ?

```
>a:=2;
>b:=3;
>if a >= b then
   "a est plus grand que b";
 else
   "b est strictement plus grand que a";
end if;
```

Modifiez les valeurs de a et de b.

2. Programmez une boucle de résolution de l'équation $ax^2 + bx + c = 0$ et appliquez–la sur plusieurs exemples.

Rappelons comment se programme une boucle de calculs

```
for i from d to f by p while condition do
   instruction_1 ;
   ⋮
   instruction_j ;
end do ;
```

ou bien

```
for s in S while condition do
   instruction_1 ;
   ⋮
instruction_j ;
end do ;
```

On peut omettre chaque élément de la syntaxe. Par exemple `x :=x0 ; to n do x :=f(x) od`.

Exercice 2.4 (L'instruction do) 1. Que se passe-t-il lorsque l'on tape `for i from 0 to 7 by 2 do [i, i*i] ; ifactor(i !) ; '------' ; end do ;` ?

2. Ecrivez une boucle qui calcule $\sum_{k=1}^{100} k^2$.
3. Ecrivez une boucle qui calcule la somme des entiers impairs compris entre 0 et 200.
4. Ecrivez une boucle qui permet de trouver le plus petit entier n tel que $\sum_{k=1}^{n} k^3 \geqslant 10^{20}$.

Exercice 2.5 (La conjecture de Syracuse) Soit $\phi : \mathbb{N}^* \to \mathbb{N}^*$ l'application telle que $\phi(n) = \dfrac{n}{2}$ si n est pair et $\phi(n) = 3n + 1$ si n est impair. On appelle suite de Syracuse toute suite $(s_n)_{n\in\mathbb{N}}$ définie par son premier terme $s_0 \in \mathbb{N}^*$ et par la relation $\forall n \in \mathbb{N}$, $s_{n+1} = \phi(s_n)$.
La conjecture de Syracuse postule que $(s_n)_{n\in\mathbb{N}}$ finit toujours par atteindre la valeur 1 et répète le motif 1, 4, 2, 1, 4, 2...

1. Ecrivez une suite de commandes pour tester la conjecture sur des petites valeurs de s_0.
2. Combien de temps faut-il pour atteindre 1 et quelle est la valeur maximale de la suite lorsque $s_0 = 15$ et $s_0 = 127$?

---

3 Prise en main de Matlab

MOTS CLÉS : Assignations, fonctions, contrôle de flot if, for, while, switch

Travaux de préparation :

1. Rappelez la syntaxe qu'il faut utiliser pour programmer des boucles `for`, `while`, `if` et `switch`.

* *
*

Matlab®est un logiciel qui permet d'effectuer des calculs numériques sans avoir besoin de programmer dans un langage traditionnel de type C, C++, Fortran ou Pascal. Matlab®signifie « Matrix laboratory »ce qui résume toute la philosophie de Matlab®. L'élement de base est la matrice et les fonctions sont optimisées pour s'exécuter rapidement sur ce type d'objet. En particulier, un scalaire est une matrice de taille 1×1.

C'est un langage interprété, ce qui signifie que l'on peut taper des commandes qui s'exécutent directement sans avoir à passer par une phase de compilation. Néanmoins, il est possible d'écrire des programmes qui sont alors une suite de commandes que l'on met dans un fichier.

A la différence de Maple™ , les instructions sont effectuées les unes après les autres. Il n'est pas possible de revenir en arrière. Pour accéder à l'aide sur la fonction ***, on peut taper `help ***`, ou bien lookfor pour rechercher un mot-clé.

Pour démarrer Matlab®, il faut taper `> matlab` dans une ligne de commande du terminal (Le caractère `>` est le prompt du terminal, il ne faut pas le taper).

Une fenêtre Matlab®se divise en quatre espaces affichés : *Command Window* où s'affichent les commandes exécutées et leur résultat, *Command History* où s'affiche l'historique des commandes, *Current Directory* qui affiche les dossiers et fichiers du répertoire courant, *Workspace* qui recense les variables affectées.

Les commandes Unix restent valables en tant que commande Matlab®. La ligne de commande peut être éditée à l'aide des commandes suivantes.

Commande	Effet
↑ ou Ctrl + P	rappeler la ligne précédante
↑ ou Ctrl + N	rappeler la ligne suivante
← ou Ctrl + B	déplacer le curseur vers la gauche
→ ou Ctrl + F	déplacer le curseur vers la droite
Ctrl + L	déplacer le curseur d'un mot vers la gauche
Ctrl + R	déplacer le curseur d'un mot vers la droite
Ctrl + A	placer le curseur en début de ligne
Ctrl + E	placer le curseur en fin de ligne
Ctrl + U	effacer la ligne courante
Ctrl + T	passer du mode insertion au mode écrasement et vice-versa
Ctrl + D	effacer le caractère sous le curseur
Ctrl + K	effacer la fin de la ligne après le curseur

Remarque : en tapant `> : mat` ↑, la dernière ligne commençant par *mat* s'affiche.

Lors de ce T.P. et des suivants, vous serez amenez à créer de nombreux fichiers. Pour ne pas disperser vos fichiers, n'oubliez pas de créer un sous-répertoire `TP7`. A la fin, vous pourrez rassembler tous les fichiers en un seul (appelé archive) en utilisant `tar cvzf ArchiveTP7_NomDeFamille.tar.gz TP7`. Le fichier créé s'appelle ArchiveTP7_NomDeFamille.tar.gz ; on lui a accolé les extensions .tar et .gz pour signaler qu'il est compressé au format zip et qu'il rassemble des fichiers sous le format tar. Pour décompresser un tel fichier, on peut utiliser l'instruction `tar xvzf ArchiveTP7_NomDeFamille.tar.gz TP7`

Exercice 3.1

1. Avant toute chose, créez un répertoire *TPMatlab* puis un sous répertoire *TP3* dans votre répertoire ModuleMHT304. Vérifiez dans la barre horizontale que le répertoire courant est TP3, sinon placez-vous-y.
2. Tapez les instructions suivantes dans l'espace de travail et expliquez ce qu'elles produisent.

```
> a = 2+i
> a
> a ;
> b = [1 2-2*i 3+i 4*i]
> c = [1+i ; -2*i ; 3+i ; 4]
> b'
> b.'
> b*c
> d = [1 2 3 ; 4 5 6 ; 7 8 9]
> c(1)
> d(2,3)
> d(3,3) = -10
> d(1:2,:)
> d
> size(b)
> size(c)
> size(d)
> e =  'abcd'
> g = [1 : .3 : 9]
> h = [1 : 1 : 9];
> h.*h
> zeros(3,4)
> one(2,5)
> eye(3)
> [d , zeros(3,2) ; eye(2,4) , ones(2,1) ]
```

   Quelles sont les classes des objets construits (instruction `class`) ?
3. En utilisant `who` et `whos`, indiquez quelles variables ont été affectées. Désaffectez la variable `a` en utilisant la commande `clear a`. On peut sauvegarder l'ensemble des variables par la commande `save nom_fich` et charger ledit fichier par la commande `load nom_fich`. Sauvegardez puis effacez toutes les données en mémoire. Rechargez-les.

4. Comparez le temps nécessaire à la définition de la matrice $A = \begin{pmatrix} 1 & -2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 7 \end{pmatrix}$ par les instructions `clear A ; A(1,1)=1 ; A(1,2)=-2 ; A(3,3) = 7` et les instructions `clear A ; A=zeros(3) ; A(1,1)=1 ; A(1,2)=-2 ; A(3,3) = 7`. Expliquez.

Pour garder une trace des calculs que l'on effectue et éventuellement les exécuter à nouveau, on peut inscrire à la suite les instructions dans un *script*. Pour cela, cliquez *File* → *New* → *M-file*. N'oubliez pas d'enregistrer le fichier, par exemple sous le nom `TP3.m`. La touche F5 du clavier permet de compiler l'ensemble des instructions du fichier, c'est-à-dire envoyer les instructions dans la fenêtre de commande, la touche F9 n'envoie que la partie sélectionnée du fichier.

Pour inscrire une ligne de commentaire, utilisez le caractère `%`. Ceci vous permettra également de signaler le début d'un exercice ou de placer un titre en tête de votre document.

Plusieurs commandes peuvent être écrites dans la même ligne, pour cela il suffit de les séparer d'une virgule.

Exercice 3.2 Créez un fichier `TP3.m` dans lequel vous continuerez le tp.

1. Soit $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$. Que représente `B = A+2`, `C = A*3`, `D = A*A`, `E = A.*A`, `F=A^3`, `G = exp(A)`, `H = sqrt(A)` ? En particulier quel est le rôle du point avant le symbole * ou / ?
2. Que représentent `x = [-1 : .1 : 1] ;` et `y = sin(x)./x ;` ? Quel est l'effet de l'instruction `plot(x,y)` ? Représentez sur le même graphique les représentations des fonctions $x \mapsto x^2$ (en rouge) et $x \mapsto x^2 \sin(x)e^{-x}$ (en bleu).
3. Tracez la courbe paramétrique du plan définie par $x(t) = \cos^3(t)$, $y(t) = \sin^3(t)$ pour $t \in [0, 2\pi]$.
4. En utilisant l'exemple donné dans l'aide de `meshgrid`, construisez la surface d'équation $z = \dfrac{\sin(\sqrt{x^2+y^2+1})}{\sqrt{x^2+y^2+1}}$.
5. Tapez `ans`, `eps`, `realmax`, `realmin`, `i`, `j`. Ce sont des variables de Matlab® que l'on peut modifier mais non effacer. A quoi correspondent-elles ?

Pour écrire une fonction élaborée, il faut enregistrer dans un fichier le texte de la fonction. Voici un exemple, pour lequel il convient de créer le fichier *disc.m* qui contient

```
function d=disc(a,b,c)
% disc(a,b,c) calcule le discriminant de $ax^2+bx+c$
% par la formule $d=b^2-4ac$
d = b^2-4*a*c
```

Exercice 3.3 1. (For) Affichez la liste des carrés des entiers entre 1 et 10.

2. (While) Ecrivez fonction `function n = pluspetit(k,epsilon)` qui détermine le premier entier n suffisamment grand pour que $\frac{1}{n^k} \leqslant epsilon$.
3. (Switch) Ecrivez une suite d'instruction avec l'instruction `switch` qui affiche le type d'équation, linéaire ou quadratique, et donne les racines de l'équation $ax + b = 0$ ou $ax^2 + bx + c + 0$ selon que V est le vecteur $[a\, b]$ ou $[a\, b\, c]$.

Exercice 3.4 On souhaite trouver toutes les substitutions de lettres par des chiffres qui rendent l'addition posée ci–dessous valable :

$$\begin{array}{cccccccc} & m & a & n & g & e & r \\ + & m & a & n & g & e & r \\ \hline g & r & o & s & s & i & r \end{array}$$

1. Combien de tests faut-il essayer ?
2. Montrez que $r = 0$ et $g = 1$.
3. Ecrivez un programme qui teste toutes les possibilités restantes et renvoie les valeurs de lettre possibles.

4 Structures de données et procédures en maple

MOTS CLÉS : Structures de données (listes, tableaux, ensemble), procédure, boucles logiques.

Maple™ connaît différentes *structures de données*. Les principales sont les suivantes :

- les *suites*, définies par `S := 1,2,3,4 ;`, qui peuvent admettre des doublons et sont ordonnées ;
- les *listes*, définies par `L := [1,2,3,4]`), qui peuvent admettre des doublons et sont ordonnées ;
- les *ensembles*, définis par `E := {1,2,3,4}`, sans doublons et sans ordre.

Pour accéder au $n$ième élément d'une suite L, on peut utiliser l'instruction `L[n]`. La fonction `op` renvoie les *opérandes* : elle permet de se débarrasser des crochets ou accolades. La fonction `nops` compte les opérandes. La fonction `member` teste l'appartenance.
Pour initialiser une suite vide, on peut taper `S :=NULL :` ; pour une liste vide, `L := []` ; pour un ensemble vide, `E := {}`.
Pour appliquer une fonction aux éléments d'une liste ou d'un ensemble, il faut utiliser `map`. Par exemple, `map(sin, [0, Pi/4, 1,Pi])` applique la fonction sinus à la liste des quatre éléments.

Pour écrire un *programme* autonome, on utilise les commandes

```
Nom_Procedure := proc(argument1, ..., argumentk) \\
instructions ;
...
end proc;
```

La procédure renvoie toujours le dernier résultat évalué. Il vaut mieux signaler dans la procédure quelles sont les variables locales, c'est-à-dire celles dont l'usage se limite à l'intérieur du programme.

Exercice 4.1 (Suites classiques)

1. Avec la commande `seq`, construisez la suite `S`, la liste `E` et l'ensemble `E` des 10 premiers termes de la suite géométrique de raison 2 et de terme initial 3.
2. Recherchez le $7^{\text{ième}}$ élément de `L` et le cardinal de E. Appliquez la fonction logarithme à `L`.

Exercice 4.2 Que permettent de faire les lignes suivantes ?

```
> Binome := proc(a,b,c)
local d;
d := b^2 - 4*a*c;
if d > 0 then
  (-b - sqrt(d))/(2*a), (-b + sqrt(d))/(2*a) ;
elif d = 0 then
  -b/(2*a);
else
  "pas de solutions";
end if;
end proc;
>Binome(1,-2,1);
```

Exercice 4.3 Approximation de $\sqrt{a}$
Etant donné un réel strictement positif a, on définit la suite réelle $(x_n)_{n\in\mathbb{N}}$ de la manière suivante :

$$\begin{aligned} x_0 &= a \\ x_{n+1} &= \frac{x_n^2 + a}{2x_n} \end{aligned}$$

On admet que cette suite converge vers $\sqrt{a}$. Ecrivez une fonction `racine = proc(a,n)` qui lit un réel positif a et un entier n et qui renvoie une approximation de $\sqrt{a}$ en utilisant x_n. Comparez avec la réponse que donne Maple en utilisant `sqrt`. Ecrivez un second programme `racineliste = proc(a,n)` qui renvoie la liste des réels $(x_p)_{0\leqslant p\leqslant n}$.

Exercice 4.4 Deux suites
On définit deux suites par :

$$\begin{aligned} u_0 &= 1 \\ v_0 &= 2 \\ \forall n \in \mathbb{N},\quad u_{n+1} &= \frac{u_n + v_u}{2} \\ \forall n \in \mathbb{N},\quad v_{n+1} &= \sqrt{u_{n+1} v_n} \end{aligned}$$

On admet que que les suites (u_n) et (v_n) sont adjacentes de limite $\sqrt{27}/\pi$.

1. Ecrivez un programme qui lit un entier n et qui affiche l'approximation du nombre π obtenue à partir de v_n.
2. Ecrivez un programme qui prend en argument un nombre ϵ, et retournera l'approximation du nombre π obtenue à partir de v_n, dès que la condition $\left|\frac{u_n - v_n}{u_n + v_n}\right| \leq \epsilon$ est satisfaite.

Exercice 4.5 Programmez une procédure `SousListe := proc(L,M)` qui cherche si la liste M est une sous liste de L et revoie le booléen correspondant.

5 Représentation graphique

MOTS CLÉS : fonction plot

Travaux de préparation :

1. Traitez les 3 premières questions de l'exercice 2.

---

* *
*

La représentation graphique en Maple™ s'effectue avec la commande `plot`. Par exemple, `plot(sin(x)/x,x=-3..3) ;` détermine le graphique de $x \mapsto \frac{\sin(x)}{x}$ entre -3 et 3.

Plusieurs fonctions peuvent être tracées simultanément. Par exemple : `plot([sin(x), cos(x)],x=-10..10, -1.2..1.2, legend = ["Sinus", "Cosinus"]) ;`.

Diverses options peuvent être rajoutées ensuite pour modifier le graphique. Par exemple, `plot( sin(x), x, view=[-3..3, -5..5], color=blue, linestyle = 3, scaling = constrained, title = "Courbe sinusoïdale") ;`.

Lorsque la fonction à tracer est discontinue, on peut l'indiquer à Maple™ par l'option `discont=true`. Par exemple `plot(x + 1/(x-1) , x=-1..2, -8..8, discont = true) ;`.

Une bibliothèque de fonctions graphiques supplémentaires est disponible sous Maple™ . Il faut la charger en utilisant l'instruction `with(plots) ;`. Pour superposer des graphiques, on peut utiliser la commande `display`. Par exemple :

```
Log := plot(ln(x),x=0..10,-5..5, color = blue):
Tang := plot( x-1 , x=0..10, -5..5, color = cyan , linestyle = 3):
display([Log, Tang], title="Le logarithme népérien et sa tangente en 1");
```

Exercice 5.1

1. Essayez les exemples présentés ci-dessus.
2. Représentez dans une même fenêtre les fonctions $x \mapsto x \sin\left(\frac{1}{x}\right)$, $x \mapsto x$, et $x \mapsto -x$ pour des abscisses comprises entre $-0,2$ et $0,2$.
3. Tracez la fonction tan sur l'intervalle $[-\pi, \pi]$.
4. En utilisant la commande `implicitplot` de la bibliothèque `plots`, tracez la courbe d'équation $y^2 = x^3 - 3 * x + 1$ dans des intervalles adaptés.
5. En utilisant la commande `plot3d` de la bibliothèque `plots`, représentez le graphe de la fonction $(x, y) \mapsto \dfrac{\sin(x^2 + 3y^2)}{0,1 + r^2} + (x^2 + 5y^2)\dfrac{\exp(1 - r^2)}{2}$ pour $-1 \leqslant x \leqslant 1$ et $-1 \leqslant y \leqslant 1$.
6. En utilisant les commandes `plot3d`, `implicitplot3d` et `display3d` de la bibliothèque `plots`, tracez dans un même repère la sphère unité de $\mathbb{R}^3$ et son plan tangent passant par le point $(0, 0, 1)$.
7. En cherchant dans l'aide en ligne `?plot[parametric]`, représentez la courbe donnée par $x(t) = 3\cos(t) - cos(3t)$, $y(t) = 3\sin(t) - \sin(3t)$ pour t variant entre $-\pi$ et π.

8. Que permet de faire la commande `plot([1+cos(theta), theta,theta = 0..2*Pi], coords=polar )` ?
9. Représentez la conchoïde d'équations $\begin{cases} x(u,v) &= k^u(1+\cos v)\cos u \\ y(u,v) &= k^u(1+\cos v)\sin u \\ z(u,v) &= k^u \sin v - ak'^u \end{cases}$ en choisissant les paramètres suivants $k = 1,2$, $k' = 1,2$, $a = 1,5$ et $u \in [0, 6\pi]$, $v \in [0, 2\pi]$.
10. En utilisant la commande `spacecurve`, représentez la courbe donnée en coordonnées cylindriques par $\begin{cases} \rho(t) &= 2+\cos\left(\frac{7}{3}t\right) \\ \phi(t) &= t \\ z(t) &= \sin\left(\frac{7}{3}t\right) \end{cases}$ où t varie entre 0 et 2π.

Exercice 5.2 (Famille de courbes) Pour tout entier naturel n non nul, on donne $f_n(x) = x\cos^n(x)$, on note x_n l'unique maximum de f_n sur l'intervalle $[0, \frac{\pi}{2}]$ et $y_n = f_n(x_n)$. Le but de cet exercice est d'étudier graphiquement le comportement de (x_n, y_n) et de son approximation au premier ordre.

1. Vérifiez que x_n existe et est unique. Montrez que $x_n \tan x_n = \frac{1}{n}$.
2. En observant que $x \mapsto x\tan x$ est strictement croissante, bijective et continue, montrez que $(x_n)_{n\in\mathbb{N}}$ converge vers 0, puis que $x_n^2 \underset{n\to+\infty}{\sim} \frac{1}{n}$.
3. En notant $y_n = f_n(x_n)$, justifiez que $\ln\left(\frac{y_n}{x_n}\right) = n\ln\cos x_n$, et par un développement limité de ln, montrez que $\ln\left(\frac{y_n}{x_n}\right) \to -\frac{1}{2}$. Concluez que $y_n \underset{n\to+\infty}{\sim} \frac{1}{\sqrt{en}}$
4. Entrez la boucle suivante : `for k from 1 to 3 do g[k] := x -> x^k ; od ;`. Que se passe-t-il ? Remédiez au problème en utilisant la commande `subs`.
5. Tracez dans un même graphique les vingt cinq premières fonctions f_k (utilisez la commande `seq`).
6. L'instruction `fsolve(x*tan(x)=1,x,0..Pi/2)` permet la résolution numérique d'équation. Construisez la liste $PtsEx := [[x_1, y_1], [x_2, y2], \ldots, [x_{25}, y_{25}]]$ à l'aide de cette commande.
7. Avec la commande `pointplot` de la bibliothèque *Plots*, représentez la liste `PtsEx` dans un graphique.
8. Construisez la liste `PtsAp` des points $[[\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{en}}]_{1\leqslant n\leqslant 25}]$ et représentez la dans un graphique en utilisant un autre symbole que pour la question précédente.
9. Superposez les trois graphiques obtenus.

6 Programmation récursive

MOTS CLÉS : Récursivité, suite de Fibonacci, courbe fractale, algorithme de tri

---

Travaux de préparation :

1. Complétez : *La suite de Fibonacci est définie par ses termes initiaux* $F_0 = 1$, $F_1 = 1$ *et*

$$\forall n \in \mathbb{N}, \qquad F_{\dots} = \dots \qquad .$$

On dit que ce type de suite est une suite ... . Le terme général s'exprime, après résolution de l'équation ...

$$r^2 = \dots,$$

en fonction de n *par* $F_n = \dots$. Justifiez le calcul du terme général.

2. Préparez la question 1 de l'exercice 2.

---

* *
*

Une fonction récursive est une fonction qui fait appel à elle-même au cours du calcul. Lorsqu'on utilise ce type de programmation, il faut être très précautionneux et bien définir les cas initiaux, sans quoi le programme boucle indéfiniment. Par exemple, pour programmer une suite arithmétique de raison 4 et terme initial 3, on peut écrire :

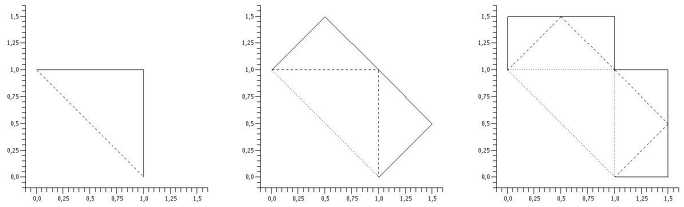
```
> suitearithm := proc (n)
> suitearithm(n-1) + 4 ;
> end proc;
> suitearithm(0) := 3;
```

Dans bien des cas, ce type de programme peut remplacer avantageusement des boucles `for` ou `while`.

Exercice 6.1 (Fibonacci)

1. Ecrivez une procédure récursive `Fibonacci := proc(n)` qui calcule de façon récursive le n-ième terme de la suite de Fibonacci. Testez votre procédure en calculant F_0, F_1, F_3 et F_5.
2. Chronométrez le temps nécessaire au calcul de chacun des 25 premiers termes (et plus si la machine le permet). On utilisera la commande `time()` et l'on se souviendra de la commande `seq` ou `for`. Représentez les résultats par un graphique ; on prendra une échelle logarithmique et on se souviendra de la commande `listplot`. Quelle conclusion en tirez vous ?
3. Soit $c(n)$ le nombre d'appels à `Fibonacci` nécessaire au calcul de F_n par cette procédure, montrez que $c(n)$ vérifie une relation de récurrence linéaire, et calculez $c(n)$, sans maple, en posant $\tilde{c}_n = \frac{c(n)+1}{2}$.
4. La commande `option remember` force Maple™ à retenir les résultats des appels intermédiaires. Ecrivez une procédure récursive `FibonacciBis := proc(n)` qui tient compte de cette option. Représentez le temps nécessaire en fonction de n sur un diagramme. (On utilisera la commande `forget` pour effacer la table de hachage entre chaque appel à FibonacciBis).

Exercice 6.2 (Courbe de Levy) La courbe de Levy est la courbe qui s'obtient par un processus de limite de lignes brisées construites par récurrence. On construit initialement un segment L_0. Puis pour chaque ligne brisée L_n construite, on en remplace chaque segment σ par les deux côtés d'un triangle rectangle isocèle dont σ est l'hypothénuse.



1. Montrez que si $\sigma = [A, B]$ est un segment de la ligne brisée L_n, z est l'affixe de A et z' celui de B, alors le point C à construire a pour affixe $\dfrac{z+z'}{2} + i\dfrac{z'-z}{2}$.
2. Ecrivez une boucle d'instructions qui transforme la liste ordonnée des affixes des points de L_n en la liste correspondant à L_{n+1}. On pourra employer la fonction `subsop`.
3. Ecrivez une procédure récursive `Levy := proc(T)` où T représente une liste d'affixe qui itère le processus de construction de la courbe et s'arrête si la distance entre les deux premiers points est inférieure à un certain pas que l'on fixera.
4. Représentez la courbe à l'aide de cette procédure en partant de $T = [I, 1, I]$ avec un pas de $0,03$. (On utilisera la fonction `complexplot`).

Exercice 6.3 (Quick sort) L'algorithme de tri rapide (ou quicksort) procède selon une technique de diviser pour régner : on choisit un pivot parmi les éléments de la liste, on isole dans une première sous-liste les éléments inférieurs au pivot et dans une seconde les éléments supérieurs au pivot. On applique l'algorithme à chaque sous-liste tant qu'elles ne se réduisent par à un élément.

1. Programmez l'algorithme et testez-le sur quelques exemples significatifs.
2. On considère une liste de longueur 2^n. En admettant que le pivot sépare à chaque étape la liste en deux listes de longueur égale, combien de fois chaque élément sera-t-il classé ? Déduisez-en le coût de l'algorithme en terme de nombre de comparaisons.

7 Problèmes de précision et de stabilité numérique

MOTS CLÉS : méthode de Newton, systèmes dynamiques.

Travaux de préparation :

1. Traitez les deux premières questions de l'exercice 1.
2. Citez des techniques pour montrer qu'une suite est croissante. Que peut-on dire d'une suite croissante majorée ?
3. Comment calcule-t-on la limite d'une suite convergente $(x_n)_{n\in\mathbb{N}}$ définie par son premier terme et une relation de la forme $\forall n \in \mathbb{N}$, $x_{n+1} = f(x_n)$?

---

* *
*

Avant de commencer, lisez (ou relisez) l'aide en ligne des fonctions `Digits` et `evalf`.

Exercice 7.1 On souhaite calculer une valeur approchée de $\sqrt{2}$ avec une précision arbitraire.

1. En utilisant les résultats sur la méthode de Newton, montrez que la racine de l'équation $x^2 - 2 = 0$ dans l'intervalle $[1, 2]$ est la limite de la suite définie par $r_0 = 1,4$ et $\forall n \in \mathbb{N}$, $r_{n+1} = \frac{r_n}{2} + \frac{1}{r_n}$. Justifiez également que $|\sqrt{2} - r_{n+1}| \leqslant |\sqrt{2} - r_n|^2$. Combien gagne-t-on en précision à chaque itération ?
2. Quel terme de la suite $(r_n)_{n\in\mathbb{N}}$ faut-il calculer pour déterminer $\sqrt{2}$ avec 1000 chiffres significatifs ?
3. Comparez le temps requis pour calculer r_{10} en fixant `Digits := 1001` et celui requis en commençant le calcul avec une précision initiale aussi faible que possible et en doublant la précision à chaque itération .

Exercice 7.2 On appelle système dynamique discret de dimension 1 une suite numérique $(x_n)_{n\in\mathbb{N}}$ définie par son premier terme x_0 et une relation de récurrence de la forme $\forall n \in \mathbb{N}$, $x_{n+1} = F(x_n)$, où F est une fonction numérique, dont on souhaite étudier le comportement asymptotique. On dit le système est *ultimement périodique de période k* si à partir d'un certain rang n, une suite finie de k valeurs itérées se répète.

1. Ecrivez une procédure `Orbite := proc(F,x0, n0, p)` qui renvoie la liste des itérés du système dynamique (x_n) défini par F et x_0 pour n compris entre n_0 et $n_0 + p$.
2. Etudiez numériquement le système défini par $f : x \mapsto 2,9\,x(1 - x)$ et $x_0 = 0,1$ en effectuant les calculs avec une précision de 10 chiffres. Quelle conclusion en tirez–vous ?
3. Reprenez l'étude en effectuant les calculs avec une précision de 15 chiffres mais en affichant les réponses avec 10 chiffres seulement. Quelle conclusion en tirez–vous ?

4. On peut démontrer que la suite admet une limite. Calculez sa valeur théorique et comparez–la avec les valeurs numériques obtenues dans les deux cas.

Le système étudié s'appelle la *suite logistique*.

Exercice 7.3 On se propose d'étudier la suite $(x_n)_{n\in\mathbb{N}}$ définie ci–dessous et de calculer son éventuelle limite.

$$\begin{cases} x_0 = \dfrac{11}{3} \qquad x_1 = \dfrac{43}{11} \\ \forall n \in \mathbb{N}, \quad x_{n+1} = 108 - \dfrac{815}{x_n} + \dfrac{1500}{x_n x_{n-1}} \end{cases}$$

1. Ecrivez une procédure récursive `xexact = proc (n)` qui calcule de façon exacte les termes de la suite $(x_n)_{n\in\mathbb{N}}$. Affichez par une boucle les valeurs approchées de x_0 jusqu'à x_{30}. Que remarque-t-on ?
2. Réécrivez une procédure récursive `xapproch = proc (n)` qui calcule les termes de la suite $(x_n)_{n\in\mathbb{N}}$ de façon approchée à chaque étape. Affichez par une boucle les valeurs de x_0 jusqu'à x_{30}. Que remarque-t-on ?
3. On pose $a_0 = 1$ et pour tout $n \in \mathbb{N}$, $a_{n+1} = \prod_{k=0}^{n} x_k$. Calculez a_1 et a_2. Montrez par un raisonnement que

$$\forall n \in \mathbb{N}, \quad a_{n+3} - 108a_{n+2} + 815a_{n+1} - 1500 = 0$$

4. Trouvez avec Maple les racines ρ_1, ρ_2 et ρ_3 du polynôme

$$P(x) = x^3 - 108x^2 + 815x - 1500$$

5. On cherche des nombres α, β, γ tels que pour tout $n = 0, 1, 2$, $a_n = \alpha\rho_1^n + \beta\rho_2^n + \gamma\rho_3^n$. Ecrivez le système d'équations et résolvez–le avec Maple. (On pourra charger le package `LinearAlgebra` et consulter l'aide de `LinearSolve`).
6. Montrez par un raisonnement par récurrence que pour tout $n \in \mathbb{N}$, $a_n = \dfrac{2}{3}3^n + \dfrac{1}{3}5^n$.
7. Définissez une fonction f qui à un entier n associe la valeur a_n. Exprimez x_n en fonction de f et simplifiez le résultat obtenu avec Maple.
8. Montrez avec Maple que la suite $(x_n)_{n\in\mathbb{N}}$ est croissante et majorée. Que peut-on en conclure ?
9. Déterminez avec Maple la limite de la suite $(x_n)_{n\in\mathbb{N}}$. Comparez avec les résultats des questions 1 et 2. Expliquez ce qui se passe.

8 PGCD, relations de Bézout, restes chinois

MOTS CLÉS : Relations de Bézout, algorithme d'Euclide, restes chinois

Travaux de préparation :

1. Sur le papier, appliquez l'algorithme d'Euclide à $a = 13$ et $b = 8$. Construisez à la main une relation de Bézout.
2. Soient a et b deux entiers naturels, d leur pgcd, q et r le quotient et le reste de leur division euclienne

$$a = bq + r.$$

   En supposant que r est non nul et qu'il existe deux entiers s et t tels que

$$s\,b + t\,r = d,$$

   expliquez comment construire u et v tels que $au + bv = d$ à partir des autres quantités.
3. Traitez la question 1. de l'exercice 2.

* *
*

Les fonctions Maple™ qui s'appliquent aux entiers commencent en général par la lettre i comme *integer*. Voici une liste de fonctions utiles :

Commande	Effet
`iquo`	Calcule le quotient de la division euclidienne
`irem`	Calcule le reste de la division euclidienne
`mod`, `modp`, `mods`	Calcule le reste modulo
`igcd`	Calcule le pgcd
`igcdex`	Trouve une relation de Bezout
`ifactor`	Calcule la factorisation en nombres premiers.

Les même fonctions existent pour les polynômes :

Commande	Effet
`quo`	Calcule le quotient de la division euclidienne
`rem`	Calcule le reste de la division euclidienne
`gcd`	Calcule le pgcd
`gcdex`	Trouve une relation de Bezout
`factor`	Calcule la factorisation en polynômes irréductibles.

Exercice 8.1 Construction d'une relation de Bézout (Etienne, 1730-1783) et algorithme d'Euclide (env. 300 av. J.C.)

1. Déterminez des procédures `pgcd := proc(a,b)`, `ppcm := proc(a,b)` et `bezout := proc(a,b)` qui prennent comme arguments les entiers a et b et qui renvoie le pgcd d, le ppcm m et les coefficients u et v d'une relation de Bézout $au + bv = d$. Vérifiez votre procédure en appelant `igcd`.
2. Pour quels nombres a et b, le coût de l'algorithme en nombre de divisions euclidiennes par rapport à la taille des entrées est-il le pire ? On réfléchira « à l'envers », c'est–à–dire qu'on partira de la dernière division euclidienne et on construira les plus petits entiers a et b possibles dans une instance de l'algorithme en n divisions euclidiennes.
3. Soit a et q deux nombres premiers entre eux. Comment peut-on calculer l'inverse de a modulo q à partir d'une relation de Bézout ?
4. Ecrivez une procédure `euclide2 := proc(a,b)` semblable à celle de la première question mais qui s'applique à des polynômes a et b.

Exercice 8.2 Lemme des restes chinois

Pour cette exercice, on pourra utiliser directement la fonction `igcdex`. Soient p, q deux entiers naturels premiers entre eux et a, b deux entiers naturels.

1. On considère le système $x \equiv \ a\,[mod\ p]$ et $x \equiv \ b\,[mod\ q]$ où $x \in \mathbb{Z}$. A partir d'une relation de Bézout $pu+qv = 1$ judicieusement choisie, donnez une solution explicite dans les cas suivants
   (a) $a = 1$ et $b = 0$,
   (b) a quelconque et $b = 1$
   (c) $a = 0$ et b quelconque
   (d) a et b quelconques
2. Ecrivez une procédure `resolution2 := proc(p,q,a,b)` qui prend comme arguments les entiers p, q, a et b et qui renvoie une solution du système. Quel est l'ensemble des solutions dans ce cas ?
3. Factorisez 247 puis résolvez l'équation $x^2 \equiv 118[247]$.

9 Système cryptographique RSA

MOTS CLÉS : Exponentiation rapide, algorithme RSA

Travaux de préparation :

1. Traitez la question 1. et 2. de l'exercice 1.

* *
*

La technique des *carrés répétés* permet de gagner beaucoup de temps lorsque l'on doit calculer la puissance n-ième d'un nombre a. Elle consiste à décomposer n en base 2, c'est-à-dire $n = 2^k + n_{k-1}2^{k-1} + \cdots + n_0$, puis calculer

$$a^n = (\cdots((a^2) \cdot a^{n_{k-1}})^2 \ldots)^2 a^{n_{k-2}} \ldots)^2 \cdot a^{n_0}$$

en partant de la parenthèse la plus intérieure. Par exemple, si $n = 11$, $a^{11} = ((a^2)^2 a)^2 a$.

Exercice 9.1 Exponentiation rapide

1. Comment décomposer a^{45} ?
2. Combien de carrés et de multiplications doit-on calculer au plus en fonction de n avec cette méthode ?
3. Ecrivez une procédure `Exponentiation :=proc(a,p,n)` qui calcule $a^n \mod p$ où $n \in \mathbb{N}$ avec la technique des carrés répétés. On pourra utiliser une boucle `for` ou bien écrire une procédure récursive.

La *méthode de cryptographie* R.S.A est très difficile à attaquer. Elle est devenue un standard dans le monde. Les lettres R.S.A. signifient Rivest-Shamir-Adleman du nom de ses inventeurs, qui ont mis au point en 1977. RSA est un algorithme dit « à clef publique »qui sert aussi bien au chiffrement de documents qu'à l'identification.

Le principe d'un algorithme à clef publique est que la méthode pour chiffrer le message est connue de tout le monde, mais qu'une seule personne sait déchiffrer. Dans le cas du codage RSA, tous les agents reçoivent la clef publique, un couple (e, n) tel que n est un nombre (très grand en général, et qui est le produit de deux facteurs premiers p et q), et e choisi tel que e et $(p-1)(q-1)$ sont premiers entre eux. La clé privée du décodeur est l'entier d tel que $1 < d < (p-1)(q-1)$ et

$$ed \equiv 1 \mod (p-1)(q-1)$$

La sécurité de ce code vient du fait qu'il est très difficile de trouver la décomposition en facteurs premiers $n = pq$ du nombre n.

Le message est exprimé à l'aide des entiers choisis dans $\Sigma = \{0, 1, \ldots, n-1\}$. Pour le chiffrer un caractère, on utilise l'application

$$\Theta : \sigma \in \Sigma \mapsto \sigma^d \mod n$$

Pour déchiffrer, on effectue

$$\Xi : \sigma \in \Sigma \mapsto \sigma^e \mod n.$$

Cette technique est justifiée par le petit théorème de Fermat, qui dans notre cas s'écrit : pour tout entier x et pour tout entier ℓ,

$$x^{\ell(p-1)(q-1)+1} \equiv x \mod n.$$

Ainsi la propriété $ed \equiv 1 \mod (p-1)(q-1)$ garantit que $\Xi \circ \Theta = \mathrm{Id}$.

Exercice 9.2

1. Ecrivez une procédure de chiffrement `RSA := proc(s, n, d)` qui chiffre l'entier s avec la clef publique (n, d).
2. Chiffrez le message 1234567890 en utilisant la clef publique $n = 21556703041$ et $d = 7$.
3. Déchiffrez le message 8509780874 en utilisant la clef privée $n = 34675796329$ et $e = 30595937633$.
4. Maple parvient-t-il à effectuer les calculs sans la procédure d'exponentiation rapide et en utilisant l'opérateur `^` seulement ? Que se passe-t-il si on remplace `^` par `&^` ?
5. Dans les deux cas précédents, cassez le cryptosystème, c'est-à-dire retrouvez les valeurs $\{p, q\}$, d et e.
6. Ecrivez une procédure `GenClef := proc()` qui génère aléatoirement une clef publique (n, d) et une clef privée (n, e) où n est le produit de deux nombres premiers p, q choisis au hasard parmi les 10^7 premiers nombres premiers et e est choisi uniformément entre 2 et $(p-1)(q-1)-2$. Regardez dans l'aide les fonctions `rand`, `isprime`.

---

10 Recherche des zéros de fonctions

MOTS CLÉS : Dichotomie, sécante , méthode de Newton

Soit f une fonction numérique dont on recherche un zéro ζ. Soit $(x_n)_{n\in\mathbb{N}}$ la suite des approximations de ζ obtenue à partir de termes initiaux en utilisant l'une des méthodes suivantes : méthode de dichotomie, de la sécante ou de Newton.

Travaux de préparation :

1. Exprimez, pour tout entier k, le terme x_{k+1} en fonction des termes précédents pour la méthode de la sécante et pour la méthode de Newton.
2. Rappelez, pour les trois méthodes, à quelle vitesse théorique la suite $(x_n)_{n\in\mathbb{N}}$ des approximations converge vers le zéro ζ, i.e. donnez l'exposant α tel que pour tout n entier, $|\zeta - x_{n+1}| \leqslant C.|\zeta - x_n|^\alpha$ pour une certaine constante C.
3. Traitez la question 1 de l'exercice 3 et la question 3 de l'exercice 4.

---

* *
*

Par défaut, Matlab effectue les calculs avec 16 chiffres mais n'affiche que 5 chiffres. La commande `format long` permet d'afficher les résultats à une précision de 15 chiffres. La commande `fzero` permet de trouver les valeurs approchées de zéros de fonction.

Des fonctions peuvent être créées en Matlab de deux manières, soit avec la commande `inline`, soit dans un fichier indépendant. Ainsi `g=inline('x^2-1')` affecte à la variable g la fonction $x \mapsto x^2 - 1$ et écrire dans le fichier $h.m$ les lignes

```
function y=h(x)
y=x^2-1;
```

crée le fichier $h.m$ contenant la fonction $x \mapsto x^2 - 1$. Pour utiliser g ou h en tant qu'argument d'une autre fonction, on écrit par exemple `fzero(g,0)` parce que g est une variable mais `fzero(@h,0)` parce que h n'est pas une variable mais le nom du fichier contenant une fonction. Pour contourner ce problème, on peut définir la variable k par `k = @h` et appeler `fzero(k,0)`.

Exercice 10.1

1. Ecrivez une fonction `function zero = dichoto(f, a, b, nbetap)` qui renvoie la suite des $nbetap+1$ premières approximations du zéro de f localisé entre a et b calculé par une méthode de dichotomie.
2. Ecrivez une fonction `function zero = secante(f, x0, x1, nbetap)` qui renvoie la suite des $nbetap+1$ premières approximations du zéro de f calculé à partir de x_0 et x_1 par la méthode de la sécante.
3. Ecrivez une fonction `function zero = newton(f, ff, x0, nbetap)` qui renvoie la suite des $nbetap+1$ premières approximations du zéro de f calculé à l'aide de la dérivée ff de f à partir de x_0 par la méthode de Newton.

Exercice 10.2 (Ordre de convergence) On définit les fonctions

$$f_1(x) = 3\cos(x) - 2\ln(x+1) - 1,$$
$$f_2 = 2x - 1, \quad f_3(x) = (2x-1)^3, f_4(x) = (2x-1)^5$$

1. Après avoir calculé les dérivées à la main, comparez sur un même graphique (n en fonction de k) le nombre n d'itérations nécessaires à chaque méthode pour obtenir le zéro des fonctions suivantes entre $a = 0$ et $b = 1$ avec une précision de 10^{-k} pour k entre 1 et 5.
2. Pour chaque fonction, représentez graphiquement les premiers termes de la suite $\left(\frac{\ln(|\zeta - x_{n+1}|)}{\ln(|\zeta - x_n|)}\right)_{n\in\mathbb{N}}$ où ζ représente la zéro de la fonction.

Un modèle simple de biologie, le modèle SIR, permet de simuler l'évolution d'une population affectée par une épidémie. Afin de calibrer un dispositif de santé publique, on souhaite connaître au début de l'épidémie combien de personnes seront touchées et combien resteront saines. Les équations du modèle conduisent à l'égalité :

$$S_\infty = S_0 \exp\left(-\frac{N - S_\infty}{\rho}\right) \tag{1}$$

où N est le nombre d'individus total, S_0 est le nombre d'individus initialement sains, ρ est un taux de guérison relatif (inférieur à S_0) et S_∞ est notre inconnue : le nombre d'individu restés sains en fin d'épidémie.

Exercice 10.3 Le modèle S.I.R.

1. Montrez que l'équation (1) admet une seule solution entre 0 et N.
2. Résolvez l'équation pour $N = 51$, $S_0 = 50$ et différentes valeurs admissibles de ρ. Tracez le graphique de S_∞ en fonction du rapport $\frac{S_0}{\rho}$.

Exercice 10.4 On considère la fonction $f : x \mapsto x^3 - 1$. On souhaite colorier le domaine du plan complexe $\mathscr{D} = [-2, 2] \times [-2, 2] \subset \mathbb{C}$ en fonction de la racine de f vers laquelle la méthode de Newton converge.

1. On divise $\mathscr{D}$ en petits carrés de longueur p (paramètre à choisir). Avec la commande `meshgrid`, construisez une matrice Z dont les coefficients représentent les affixes de ces carrés. Par exemple pour $p = 2$

$$Z = \begin{pmatrix} -2+2i & 2i & 2+2i \\ -2 & 0 & 2 \\ -2-2i & -2i & 2-2i \end{pmatrix}$$

2. Avec la commande `inline`, écrivez la fonction `Newt` qui effectue une étape dans la méthode de Newton. Cette fonction devra accepter des matrices en entrée.
3. Vérifiez que les parties imaginaires des racines de f sont distinctes.
4. Calculez Zn la n-ième itération de la méthode de Newton pour toutes les valeurs contenues dans Z.
5. Avec la commande `imagesc`, représentez la partie imaginaire de Zn. On prendra soin d'éliminer les valeurs de Zn supérieures à 1 ou inférieures à -1 en utilisant la commande `max` et `min`.

11 Racines de polynômes et de systèmes de polynômes

MOTS CLÉS : polynômes, division euclidienne, racines réelles, suite de Sturm.

Travaux de préparation :

1. Traitez la question 1 de l'exercice 1 et de l'exercice 2.
2. Expliquez pourquoi connaissant W, le résultat admis dans l'exercice 2. permet d'écrire un algorithme qui isole chaque racine d'un polynôme dans un intervalle distinct.

* *
*

Pour tout entier k, on note $\mathbb{Q}_k[X]$ l'espace vectoriel des polynômes de degré strictement inférieur à k. Soient $P = a_n X^n + \cdots + a_0$ et $Q = b_m X^m + \cdots + b_0$ deux polynômes de $\mathbb{Q}[X]$, où n est le degré de P et m celui de Q. On cherche à quelle condition P et Q admettent un facteur commun.

Proposition 1 *Les polynômes P et Q admettent un facteur commun si et seulement si l'application linéaire $\varphi : \mathbb{Q}_m[X] \times \mathbb{Q}_n[X] \to \mathbb{Q}_{n+m}[X]$ qui envoie (S,T) sur $S \cdot P + T \cdot Q$ n'est pas un isomorphisme, autrement dit si et seulement si le déterminant de φ*

$$\det \varphi = \begin{vmatrix} a_0 & & & & b_0 & & & & \\ a_1 & a_0 & & & b_1 & b_0 & & & \\ \vdots & \vdots & \ddots & & \vdots & \vdots & \ddots & & \\ \vdots & \vdots & & a_0 & b_{m-1} & \vdots & & \ddots & \\ \vdots & \vdots & & a_1 & b_m & \vdots & & & b_0 \\ a_n & \vdots & & \vdots & & b_m & & & \vdots \\ & a_n & & \vdots & & & \ddots & & \vdots \\ & & \ddots & \vdots & & & & \ddots & \vdots \\ & & & a_n & & & & & b_m \end{vmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \Bigg\} m+n$$

$$\underbrace{\qquad\qquad}_{m} \qquad \underbrace{\qquad\qquad\qquad}_{n}$$

est nul. On appelle $\det \varphi$ *le* résultant *de P et Q en X et on le note $Res_X(P,Q)$*

En effet, si les polynômes P et Q admettent le polynôme D comme facteur commun, l'image de φ est restreinte à aux multiples de D : φ ne peut pas être surjective, et encore moins bijective. Réciproquement, si P et Q sont premiers entre eux, soient U et V tels que $UP + VQ = 1$. Alors, pour tout $F \in \mathbb{Q}_{m+n}[X]$, F est l'image par φ de $S_0 = F \cdot U \mod Q$, $T_0 = FV \mod P$. Comme les dimensions sont finies et égales, la surjectivité équivaut à la bijectivité. Cette proposition est l'analogue du lemme chinois vu au T.P. 10.

Exercice 11.1 (Système d'équation polynomiale)

1. La matrice de φ décrite dans la proposition s'appelle la *matrice de Sylvester* (James Joseph, Londres 1814– Oxford 1897). Montrez qu'elle représente la matrice de l'application linéaire φ dans la base $(x^i, 0)_{0\leqslant i<m} \cup (0, x^i)_{0\leqslant i<n}$ de $\mathbb{Q}_m[X] \times \mathbb{Q}_n[X]$ vers la base $(x^i)_{0\leqslant i<m+n}$ de $\mathbb{Q}_{m+n}[X]$.
2. Ecrivez une procédure `result := proc(P,Q,x)` qui calcule le résultant en x de P et Q. On pourra s'aider des commandes `degree`, `CoefficientVector` de la bibliothèque `PolynomialTools` et `Determinant` de la bibliothèque `LinearAlgebra`.
3. Vérifiez que $A(X) = X^4 - 3X^2 + 2X$ et $B = X^3 - 1$ admettent un facteur commun.
4. La technique du résultant fonctionne encore avec des polynômes à deux variables $P, Q \in \mathbb{Q}[X, Y]$. On considère les polynômes

$$F(X,Y) = (Y^2+6)(X-1) - Y(X^2+1) \in \mathbb{Q}[X,Y]$$

$$G(X,Y) = (X^2+6)(Y-1) - X(Y^2+1) \in \mathbb{Q}[X,Y].$$

On cherche à résoudre le système

$$S = \{(x,y) \in \mathbb{R}^2; \qquad F(x,y) = 0, \quad G(x,y) = 0\}$$

Si (x_0, y_0) appartient à S, alors les polynômes à une variable X, $F(X, y_0)$ et $G(X, y_0)$, ont un facteur commun, puisque les deux s'annulent en x_0. Ceci nous indique que y_0 est racine de $R(Y) = \mathrm{Res}_X(F, G)$. Avec votre procédure ou bien l'instruction Maple `resultant`, calculez le résultant $R(Y) = \mathrm{Res}_X(F, G)$ de F et de G.

Avec la commande `solve`, déduisez-en les valeurs possibles de y puis en substituant les valeurs de y dans F et dans G (commande `subs`), calculez les valeurs possibles de x. Vérifiez graphiquement vos résultats pour les zéros réels en utilisant `implicitplot`.

Soit P un polynôme ne possédant que des racines simples. On définit la suite de polynômes $(P_k)_{k\geqslant 0}$ par :

$$P_0 = P \; ; \; P_1 = P' \; ; \; P_k = -(P_{k-2} \mod P_{k-1}) \text{ pour } k > 1$$

jusqu'à ce que P_k soit une constante. On note P_n soit le dernier terme (non nul) de la suite $(P_k)_{0\leqslant k}$. On dit que la suite $(P_k)_{0\leqslant k\leqslant n}$ est la *suite de Sturm* du polynôme P.

Soit α un réel tel que $P(\alpha) \neq 0$. On définit $W(\alpha)$ comme le nombre de variations de signe dans la suite de réels (sans tenir compte d'éventuels zéros) : $P_0(\alpha), P_1(\alpha), ..., P_n(\alpha)$. On admet le résultat suivant :

Proposition 2 (Sturm) *Soient $\alpha < \beta$ deux réels tels que $P(\alpha) \neq 0$ et $P(\beta) \neq 0$. La différence $W(\alpha) - W(\beta)$ majore le nombre de racines réelles de P dans l'intervalle $]\alpha, \beta]$.*

Exercice 11.2 Suite de Sturm (Jacques Charles François, 1803–1855)

1. Soit P un polynôme à coefficients réels $P = \omega \prod_{i=1}^{k} (x - \alpha_i)^{\nu_i}$ où $\omega \in \mathbb{R}$ est le terme dominant, où les $\alpha_i \in \mathbb{R}$ sont les racines distinctes et $\nu_i \in \mathbb{N}^{\times}$ leur ordre. Montrez que

$$\frac{P}{pgcd(P, P')} = \prod_{i=1}^{k} (X - \alpha_i)$$

2. Ecrivez une procédure `racinessimples := proc(P,x)` qui prend en argument un polynôme P d'inconnue x et renvoie un polynôme qui ne possède que des racines simples, qui sont les mêmes que celles de P. (Rappel, le pgcd s'appelle `gcd` en Maple).
3. Ecrivez une procédure `suitesturm := proc(P,x)` qui renvoie la suite de Sturm associée au polynôme P en x sous forme d'une liste.
   Vous pourrez vérifier votre procédure en la comparant à `sturmseq`.
4. Ecrivez une procédure `W := proc(P,x, α)` qui prend en arguments un polynôme P de variable x et un réel α et qui renvoie $W(\alpha)$.
5. Ecrivez une procédure récursive `encadrement := proc(P,x,α,β)`, qui procède par dichotomie et prend en arguments un polynôme P en x et deux réels α et β, avec $\alpha < \beta$, et (à condition que $P(\alpha) \neq 0$ et $P(\beta) \neq 0$) retourne un ensemble d'intervalles isolant les racines réelles de P comprises dans $]\alpha, \beta]$.
   Vous pourrez vérifier votre procédure en la comparant à `sturm`.

12 Polynômes : interpolation de Lagrange

MOTS CLÉS : Polynôme d'interpolation de Lagrange, points de Tchebychev, phénomème de Runge.

Travaux de préparation :

1. Soit f une application numérique définie sur un intervalle $[a, b]$ et $x_0, x_1, ..., x_n$ des points distincts de cet intervalle. Rappelez l'expression du polynôme d'interpolation de Lagrange de f aux points $(x_i)_{0\leqslant i\leqslant n}$.
2. Soient S le polynôme d'interpolation de Lagrange de f aux points $x_0, x_1, ..., x_{n-1}$, T celui qui interpole f aux points $x_1, x_1, ..., x_n$ et enfin P celui qui interpole f aux points $x_0, x_1, ..., x_n$. Montrer que :

$$P(x) = \frac{(x - x_0)T - (x - x_n)S}{x_n - x_0}$$

---

* *
*

Exercice 12.1 Interpolation par polynômes de Lagrange

1. Ecrivez une procédure non-récursive `lagrange := proc(n,f,a,b)` qui retourne le polynôme d'interpolation de Lagrange de f aux points équidistants $\left(x_i = a + \left(\frac{b-a}{n}\right) i\right)_{0\leqslant i\leqslant n}$.
2. En vous inspirant des travaux préparatoires, écrivez une procédure récursive `lagrangerecursif := proc(n,f, a, b)` qui retourne le polynôme d'interpolation de Lagrange de f aux points équidistribués $\left(x_i = a + \left(\frac{b-a}{n}\right) i\right)_{0\leqslant i\leqslant n}$.
3. Essayez vos algorithmes sur les fonctions suivantes définies sur $[-1, 1]$. Dans chaque cas on représentera graphiquement la fonction et quelques unes de ses interpolations pour différentes valeurs de n.

$$f_1 = x^7 + 3x^5 - 2x^2 + 1, \quad f_2 = x\cos(x) - \ln(x + 2) + 1$$
$$f_3 = \frac{1}{1 + x^2}, \qquad f_4 = (1 + x^2 - x^3)\exp\left(\frac{x}{5}\right)$$

Exercice 12.2 Phénomème de Runge et points de Tchebychev

1. Représentez graphiquement l'approximation par interpolation de Lagrange de la fonction $\varphi : x \mapsto e^{-x^2}$ sur l'intervalle $[-7, 7]$, pour différents nombres de points (par exemple $n = 5, 10, 20, 30, ...$). Que constatez-vous ? Ce phénomène s'appelle *phénomène de Runge*.
2. Interpoler une fonction sur des points équidistants n'est pas forcément le meilleur choix. Une alternative aux points équirépartis sur l'intervalle consiste à interpoler une fonction aux *points de Tchebychev* (Pafnouti Tchebychev, 1821–1894) :

$$\forall k \in [\![0, n]\!], \quad x_k = \frac{a + b}{2} + \frac{b - a}{2}\cos\left(\frac{2k + 1}{2(n + 1)}\pi\right).$$

On peut d'ailleurs démontrer que le choix de ces points est optimal en un certain sens.
Comparez pour différentes valeurs de n l'interpolation de φ avec les points équidistants et les points de Tchebychev.

13 Méthodes de quadrature et d'intégration numérique

MOTS CLÉS : Méthode des rectangles, métode des trapèzes, méthode de Simpson, ordre de convergence.

Soit f une fonction continue sur l'intervalle $[0,1]$. On cherche à approcher l'intégrale

$$I(f) = \int_0^1 f(x)\,\mathrm{d}x.$$

Le but des exercices suivants est d'illustrer l'efficacité des méthodes classiques (rectangles, trapèzes, Simpson) pour diverses fonctions. On s'intéresse en particulier à l'influence de la régularité de la fonction sur la vitesse de convergence de la méthode.

Travaux de préparation :

1. Rappelez les ordres de convergence théoriques des méthodes d'intégration numériques suivantes : rectangles, trapèzes, Simpson, c'est-à-dire l'exposant p tel qu'une inégalité du type

   $$|I(f) - I_n| \leq \frac{C}{n^p},$$

   où I_n est l'approximation numérique de $I(f)$ obtenue par n quadratures effectuées avec l'une des méthodes visées, est vérifiée pour une certaine constante C et précisez une condition suffisante de validité de l'inégalité.
2. De quelle classe est l'application f définie sur $[0,1]$ par $f(x) = |3x^4 - 1|$? Peut-on appliquer les résultats de la question précédente ?
3. On considère les fonctions suivantes, définies sur $\mathbb{R}$ par 1-périodicité par

   $$\begin{aligned} \forall x \in [0,1[, \quad f_1(x) &= \cos x \\ \forall x \in [0,1[, \quad f_i(x) &= x^i(1-x)^i \cos x \ (i = 2,3,4) \\ \forall x \in [0,1[, \quad f_5(x) &= \frac{\sin^2(2\pi x)}{1+\sin^2(2\pi x)} \end{aligned}$$

   Quelle est la régularité (la classe) de chacune de ces fonctions sur $\mathbb{R}$?

* *
*

Exercice 13.1 Ordre de convergence des méthodes classiques : rectangles, trapèzes, Simpson

Remarque : Pour programmer les formules d'intégration numérique, il est préférable d'utiliser les possibilités vectorielles de Matlab : en évitant les boucles, le temps de calcul est réduit sensiblement.

1. Ecrivez des fonctions `I=rect(f,n)`, `I=trap(f,n)` et `I=simpson(f,n)` qui calculent une approximation $I_n(f)$ de $I(f)$ avec respectivement la méthode des rectangles, des trapèzes et de Simpson avec en argument d'entrée n le nombre de subdivisions de l'intervalle $[0,1]$ et f.
2. Testez vos programmes avec la fonction $f : x \mapsto e^x$ pour différentes valeurs bien choisies de n et tracez sur un graphe logarithmique les courbes d'erreur $I - I_n$ en fonction du nombre de sous-intervalles n.

Exercice 13.2 1. De même que dans l'exercice précédent, tracez les courbes d'erreurs associées la fonction $f : x \mapsto |3x^4 - 1|$ (que l'on programmera de façon à ce qu'elle fonctionne avec des vecteurs). Quel ordre de convergence lit-on sur le graphique ?

2. Ecrivez une suite d'instructions qui permet de retrouver les ordres théoriques dans le cas de la fonction $f(x) = |3x^4 - 1|$. Tracez les courbes d'erreur.
Indice : il s'agit de découper l'intégrale en deux parties sur lesquelles la fonction est suffisamment régulière, c'est-à-dire de calculer $\int_0^{\frac{1}{\sqrt[4]{3}}} f(x)dx$ et $\int_{\frac{1}{\sqrt[4]{3}}}^1 f(x)dx$

Exercice 13.3 Intégration des fonctions périodiques
On s'intéresse ici au calcul de $I(f)$ avec la méthode des rectangles pour les fonctions 1-périodiques.

1. Ecrivez une suite d'instruction qui permet de comparer la vitesse de convergence de la méthode des rectangles pour chacune des fonctions $(f_i)_{1 \leqslant i \leqslant 5}$ définies dans la partie préparatoire.
2. Représentez graphiquement les résultats, commentez.

14 Interpolation par des polynômes trigonométriques.

MOTS CLÉS : Racines de l'unité, transformée de Fourier discrète, phénomène de Gibbs, FFT

Travaux de préparation :

1. Vérifiez qu'avec la définition (éq. 3) des coefficients $v = (v_n)_{-N/2 \leqslant n \leqslant N/2-1}$, le polynôme P défini par l'équation 2 vérifie la relation d'interpolation.
2. Décrivez la matrice Ω telle que $v = \Omega u$.
3. Vérifiez que la transformée de Fourier rapide coûte effectivement $\mathcal{O}(N \log N)$ opérations.

---

* *
*

Attention : quoiqu'il arrive, Maple numérote les éléments d'une liste en commençant à 1. A vous d'en tenir compte par rapport aux notations du TP.

Le but de ce T.P. est de présenter une technique d'interpolation d'une fonction $f : [0, 1[\to \mathbb{R}$ ou $\mathbb{C}$ par des polynômes trigonométriques, c'est à dire trouver les coefficients $(v_n)_{-N/2 \leqslant k \leqslant N/2-1}$ de sorte que le polynôme[1] trigonométrique P

$$P_N(x) = \sum_{n=-N/2}^{N/2-1} v_n \exp(2i\pi n x) \tag{2}$$

vérifie les relations d'interpolation[2] $P_N\left(\frac{k}{N}\right) = f\left(\frac{k}{N}\right)$ pour tout entier k entre 0 et $N-1$.

Nous supposerons toujours que N est pair. En pratique, N est en réalité toujours une puissance de 2.

Définition 3 *Pour tout $k \in [\![0, N-1]\!]$, on pose $u_k = f\left(\frac{k}{N}\right)$. On appelle $u = (u_k)_{k \in [\![0,N-1]\!]}$ la suite des échantillons de f. On note $\omega_N = \exp \frac{2i\pi}{N}$ la racine N-ième de l'unité. Enfin, pour $n \in [\![-\frac{N}{2}, \frac{N}{2} - 1]\!]$, on pose*

$$v_n = \frac{1}{N} \sum_{\ell=0}^{N-1} u_\ell \, \omega_N^{-n\ell} \tag{3}$$

On dit que v est la transformée de Fourier discrète de u.

---

[1] Il s'agit d'un « polynôme »dont l'inconnue est $\exp(2i\pi x)$

[2] Ce type d'interpolation est au cœur des méthodes mathématiques actuelles de traitement du signal (son, image, vidéo, etc.). Il est justifié par des raisons physiques : un signal périodique peut être représenté, sous réserve de certaines hypothèses, par une somme (infinie) de la forme $\sum_{n=-\infty}^{+\infty} v_n \exp(2i\pi n x)$ (appelée série de Fourier).
Toutefois, en pratique, les dispositifs d'acquisition et de restitution de signaux (de sons par exemple) ont une *bande passante*, c'est-à-dire un intervalle de fréquence captée ou reproduite ; les autres fréquences sont perdues ou tellement atténuées qu'elles peuvent être négligées.

Exercice 14.1 1. Ecrivez une procédure `Echant := proc (f, N)` qui renvoie la liste $u = (u_k)_{k\in[\![0,N-1]\!]}$ des échantillons de f. On se souviendra des instructions `seq` ou `for`. On utilisera `evalf` pour renvoyer le résultat sous forme approchée.

2. Ecrivez une procédure `TFD := proc (u, N)` qui calcule la transformée de Fourier discrète à partir d'un échantillon. On utilisera `evalf` pour renvoyer le résultat sous forme approchée.
3. Ecrivez une procédure `Interpol := proc( v, N)` qui calcule le polynôme d'interpolation trigonométrique de f à partir de sa transformée de Fourier.
4. Représentez sur un même graphique les fonctions $[0,1[\to\mathbb{R}$ suivantes et leurs interpolations successives : $f_1 : x \mapsto \sin(x)$, $f_2 : x \mapsto x^4 + x^3 - 2x + \frac{1}{2}$, $f_3 := \sin(2\pi x) + \sin(8\pi x)$ (Il s'avère que le polynôme d'interpolation est complexe. Pour les graphiques, on ne gardera que sa partie réelle). Commentez les résultats.

Le phénomène observé au cours de l'exercice 1 n'est malheureusement pas anecdotique. On peut démontrer mathématiquement que l'approximation d'une fonction qui « saute »en un point par des polynômes trigonométriques conduit toujours à l'apparition d'oscillations résiduelles, dont l'amplitude ne dépend pas du nombre de coefficients utilisés pour approcher la fonction. Ce phénomène porte le nom de phénomène de Gibbs [3].

Exercice 14.2 (Le phénomène de Gibbs) 1. Soit f l'application définie par morceau par $f(x) = 1$ si $x \in [0, \frac{1}{2}[$ et $f(x) = 0$ si $x \in [\frac{1}{2}, 1[$. Construisez une approximation de f pour $N = 2^\nu$ en faisant varier ν entre 1 et 6. Comparez les représentations graphiques. Qu'observe-t-on au voisinage des discontinuités de f ?

2. Faîtes de même avec l'application $x \mapsto 1 - 2x$. Comparez la valeur de $P_N\left(\frac{1}{2N}\right)$ avec $f(0)$.

Calculer $v = (v_n)_{-N/2\leqslant n\leqslant N/2-1}$ à partir de la formule (3) revient à évaluer un polynôme, en l'occurence $\frac{1}{N}\sum_{\ell=0}^{N-1} u_\ell\, X^\ell$ au point ω_N^{-n}. Les méthodes rapides (algorithme de Hörner par exemple) exigent $\mathcal{O}(N)$ opérations, soit $\mathcal{O}(N^2)$ opérations pour calculer v entièrement. L'algorithme de la *transformée de Fourier rapide* permet de résoudre le problème en $\mathcal{O}(N \log N)$.

On se place dans le cas où $N = 2^\nu$ et soit un polynôme $P(X) = \sum_{k=0}^{N-1} a_k X^k$.

On pose $Q(X) = \sum_{k=0}^{N/2-1} a_{2k} X^k$ et $R(X) = \sum_{k=0}^{N/2-1} a_{2k+1} X^k$. Ainsi

$$P(\omega_N^n) = Q\left((\omega_N^n)^2\right) + \omega_N^n R\left((\omega_N^n)^2\right)$$

[3] Un expérimentateur averti doit en avoir conscience lorsqu'il observe une série d'oscillations et se gardera bien de les interpréter comme faisant partie du signal. Le phénomène est dû au fait que les appareils de mesure tronquent nécessairement les hautes fréquences.

Comme $(\omega_N^n)^2 = \omega_{N/2}^n$, on est ramené à l'évaluation de deux polynômes Q et R de degrés $N/2$ aux racines de l'unité d'ordre $\frac{N}{2}$.

Exercice 14.3 (La transformée de Fourier rapide)

1. Ecrivez une procédure récursive `Ma_FFT := proc (u)` qui calcule la transformée de Fourier rapide de u.
2. Testez cette procédure sur les exemples du premier exercice.

---

15 Processus d'accélération de convergence de séries

MOTS CLÉS : Suites, séries, calcul numérique, méthode de Kummer, méthode de Richardson, méthode d'Aitken

Nombre de constantes mathématiques sont calculées à partir de l'évaluation numérique de suites ou de séries convergentes. Malheureusement, certaines d'entre elles convergent très lentement vers leur limite. Ce T.P. se propose de passer en revue quelques techniques d'accélération de convergence.

Pour fixer les notations, les séries étudiées seront généralement notées $S = \sum_{k=1}^{+\infty} a_k$, où $(a_k)_{k\in\mathbb{N}}$ est une suite numérique réelle, et leurs sommes partielles $s_n = a_1 + \cdots + a_n = \sum_{k=1}^{n} a_k$.

Travaux de préparation :

1. Rappelez la définition d'une série convergente.
2. Préparez les questions 1 et 2a de l'exercice 1.
3. Préparez les questions 1, 3 et 4a de l'exercice 2.

* *
*

On regardera avant de commencer à quoi servent la fonction `evalf` et la variable `Digits`.

Exercice 15.1 (La méthode de Kummer) La méthode de Ernst Kummer (1810-1893) date de 1837 et consiste à retrancher aux termes d'une série convergente $\sum_{k=1}^{+\infty} a_k$ les termes d'une autre série équivalente, dont la somme totale $C = \sum_{k=1}^{+\infty} b_k$ est connue. Précisément, on suppose que

$$\lim_{k\to+\infty} \left(\frac{a_k}{b_k}\right) = \rho \neq 0$$

La transformation devient donc

$$\sum_{k=1}^{+\infty} a_k = \rho \sum_{k=1}^{+\infty} b_k + \sum_{k=1}^{+\infty} \left(1 - \rho\frac{b_k}{a_k}\right) a_k = \rho C + \sum_{k=1}^{+\infty} \left(1 - \rho\frac{b_k}{a_k}\right) a_k$$

1. Pourquoi la seconde série du terme de droite converge-t-elle plus vite ?
2. On souhaite appliquer la méthode à la situation suivante : $a_k = \dfrac{1}{k^2}$ et $b_k = \dfrac{1}{k} - \dfrac{1}{k+1}$.
   (a) Calculez C (de tête) et déterminez ρ.
   (b) Avec Maple et en utilisant `sum`, calculez la valeur exacte de S.

(c) Comparez pour $n = 5$, 10, 100 et 1000, les valeurs approchées de s_n et $s'_n = \rho C + \sum_{k=1}^{n} \left(1 - \rho \frac{b_k}{a_k}\right) a_k$ avec S.

(d) Combien de termes de la série (s'_n) faut-il calculer pour obtenir S avec autant de précision que s_{1000} ?

Remarque : on peut gagner en rapidité en répétant le processus de Kummer plusieurs fois.

Exercice 15.2 (Le processus d'extrapolation de Richardson) Dans cet exercice, le but ultime est de donner une valeur approchée de S aussi bonne que possible à partir de la connaissance de quelques uns des premiers termes de la suite $(s_n)_{n\in\mathbb{N}}$. Supposons que la théorie prédise la taille de l'erreur commise entre s_n et S sous la forme un développement limité

$$s_n = S + \frac{\alpha_0}{n^p} + \frac{\alpha_1}{n^{p+1}} + \cdots + \frac{\alpha_k}{n^{p+k}} + o\left(\frac{1}{n^{p+k}}\right)$$

Le processus d'extrapolation de Lewis Fry Richardson (1881–1953) consiste à calculer la suite $(\sigma_n)_{n\in\mathbb{N}}$ définie par

$$\forall n \in \mathbb{N}^*, \quad \sigma_n = \frac{2^p\, s_{2n} - s_n}{2^p - 1}$$

1. Montrez que le procédé de Richardson permet de gagner un ordre dans la précision de la série, c'est-à-dire montrez que $\sigma_n = S + \frac{\alpha'}{n^{p+1}} + o\left(\frac{1}{n^{p+1}}\right)$ où $\alpha' = -\frac{\alpha_1}{2(2^p-1)}$.
2. Ecrivez une procédure `Richardson := proc (T, p)` qui calcule la liste des premiers termes de la suite $(\sigma_n)_{n\in\mathbb{N}}$ à partir de la liste `T` des premiers termes de $(s_n)_{n\in\mathbb{N}}$ et de l'ordre p.
3. Soit $(\tau_n)_{n\in\mathbb{N}}$ la suite obtenue par k itérations du processus de Richardson à partir de s_n. Avec quel ordre de précision, la suite τ converge-t-elle vers S ?
4. Soit α_n l'aire du polygone régulier à n côtés inscrit dans le disque unité. En calculant α_n pour $n = 6 \cdot 2^k$ avec $1 \leqslant k \leqslant 4$, Archimède de Syracuse (287–212 av. J.C.) a pu donner les premières approximations de la constante π.
   (a) Calculez α_n pour ces valeurs.
   (b) Combien de décimales supplémentaires Archimède aurait-il pu obtenir avec les mêmes valeurs de α_n en appliquant le processus de Richardson 4 fois (on partira avec $p = 2$ dans ce cas et on prendra soin de commencer par associer α_6 avec α_{12} et ainsi de suite) ?

Exercice 15.3 (Processus Δ^2 d'Aitken) On peut démontrer que sous l'hypothèse $\lim\limits_{k\to+\infty} \dfrac{a_k}{a_{k+1} - a_k} = 0$, le reste à l'ordre n de la série S est équivalent à $\displaystyle\sum_{k=n+1}^{+\infty} a_k \underset{n\to\infty}{\sim} \frac{a_{k+1}^2}{a_{k+2} - a_{k+1}}$. Le processus d'Alexander Aitken (1895–1967)

consiste à utiliser cet équivalent en introduisant une suite (σ_n) définie par

$$\sigma_n = s_{n+1} - \frac{a_{n+1}^2}{a_{n+2} - a_{n+1}}.$$

1. Exprimez σ_n en fonction de s_n, s_{n+1} et s_{n+2}.
2. Ecrivez une procédure `Aitken = proc(T)` qui renvoie la liste des premiers termes de la suite σ à partir de la liste `T` des premiers termes de la suite s.
3. La série $\lim_{n\to+\infty} \sum_{k=1}^{n} \frac{(-1)^{k+1}}{k}$ converge vers $\ln(2)$. Combien de décimales exactes obtient-on en en calculant s_5, s_{10}, s_{100} et s_{1000} ? Comparez avec σ_5, σ_{10}, σ_{100} et σ_{1000}.

---

16 Systèmes linéaires et inversion de matrice

MOTS CLÉS : Méthode de Cramer, pivot de Gauß, méthode de Jacobi, méthode de Gauß–Seidel, convergence, coût de calcul.

Travaux de préparation :

1. Combien d'opérations (additions de scalaires, multiplications, divisions) sont -elles nécessaires à la résolution d'un système $n \times n$ par la méthode de Gauß ?
2. Les méthodes de Jacobi et Gauß–Seidel sont des méthodes itératives où la solution du système linéaire est la limite d'une suite définie pour tout entier n par $x_{n+1} = \varphi(x_n)$. Quel terme initial peut-on prendre ? Rappelez quelle fonction φ est utilisée par chacune des méthodes.
3. Soit la matrice carrée A_0 de taille n, tridiagonale, définie par :

$$A_0 = \begin{bmatrix} 3 & -1 & 0 & \dots & 0 \\ -1 & 3 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 3 & -1 \\ 0 & \dots & 0 & -1 & 3 \end{bmatrix}$$

   En vous servant de l'une des conditions suffisantes (mais non nécessaire) vue en cours, montrez que les méthodes de Jacobi et Gauß-Seidel appliquées à cette matrice convergent.

---

* *
*

Vous trouverez la liste des connecteurs logiques en tapant directement dans la *command window* `help <` par exemple.

Vérifiez que des instructions telles que `M=[1 2 ; 3 4], M(2,1) = -7, M( :,2) = [0 ; 0]` permettent de modifier une partie d'une matrice.

Exercice 16.1 Résolution d'un système linéaire par la méthode de Carl Friederich Gauß (1777–1855)

Dans tout l'exercice, vérifiez au fur et à mesure que que vos programmes fonctionnent en les testant sur des exemples bien choisis.

1. Ecrivez une fonction `function i = pivot(A,j)` qui recherche le premier élément non nul de la $j$ème colonne de la matrice A et renvoie la ligne i où il se trouve.
2. Ecrivez une fonction `function [A,b] = echange(A,b,l1,l2)` qui échange les lignes l_1 et l_1 dans la matrice A et le vecteur b.
3. Ecrivez une fonction `function [A,b] = division(A,b,l,z)` qui divise la l-ième ligne de A et b par z.

4. Ecrivez une fonction `function [A,b] = elimine (A,b, l0, l, z)` qui retranche z fois la ligne l_0 à la ligne l.
5. Ecrivez une fonction `function x = gauss(A,b)` qui prend comme arguments une matrice A et un vecteur b, et résoud le système linéaire associé $Ax = b$ par la méthode du pivot de Gauß.
6. Appliquez cette fonction à la résolution du système formé par les égalités

$$\begin{cases} 13x + 4y + 3z &= 15, \\ -2x + 4y + 1z &= -4, \\ 3x - 7y + 18z &= 14 \end{cases}$$

Les méthodes de Jacobi et Gauß–Seidel sont des méthodes itératives de résolution d'un système d'équation linéaire $Ax = b$ où $A \in \mathcal{M}_n(\mathbb{R})$ et $b \in \mathbb{R}^n$, c'est-à-dire que l'on construit une suite de vecteurs

$$\begin{cases} \xi_0 &= b \\ \forall n \in \mathbb{N}, \quad \xi_{n+1} &= \varphi(\xi_n) \end{cases}$$

pour une certaine application φ bien choisie et dont le calcul est facile, rapide et numériquement stable. Lorsque la suite $(\xi_n)_{n\in\mathbb{N}}$ converge (ce qui n'arrive pas toujours), sa limite x vérifie alors $Ax = b$ et ses termes, pour un rang n assez grand, fournissent une approximation convenable de la solution du système.

Cherchez dans l'aide les fonctions Matlab permettant de générer la partie diagonale et les parties triangulaires supérieure et inférieure en tapant par exemple `lookfor diag`, `lookfor upper`, `lookfor lower`.

Exercice 16.2 Résolution d'un système linéaire par la méthode de Carl Gustav Jacob Jacobi (1804-1851)

1. Ecrivez une fonction `function x = jacobi(A,b,N)` qui prend comme arguments une matrice A, un vecteur b et un entier N, et résoud le système linéaire associé $Ax = b$ par la méthode de Jacobi en N itérations.
2. Appliquez cette fonction à la matrice A_0 donnée en préambule et au vecteur $b = (1, 0, 0, ..., 0, 1)^t$, pour $N = 3$, $N = 10$, $N = 100$, $N = 1000$.

Exercice 16.3 Résolution d'un système linéaire par la méthode de Gauß–Seidel (Philipp Ludwig von, 1821-1896)

1. Ecrivez une fonction `function x = gaussseidel(A,b,N)` qui prend comme arguments une matrice A, un vecteur b et un entier N, et résoud le système linéaire associé $Ax = b$ par la méthode de Gauß–Seidel en N itérations.
2. Appliquez cette fonction à la résolution du même système linéaire que dans l'exercice précédent.

Exercice 16.4 Contre-exemples
Dans chacun des cas suivants indiquez quelles méthodes de résolution fonctionnent :

$$B = \begin{bmatrix} 1 & 0.75 & 0.75 \\ 0.75 & 1 & 0.75 \\ 0.75 & 0.75 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{bmatrix}, \quad D = \begin{bmatrix} 2 & -1 & 1 \\ 2 & 2 & 2 \\ -1 & -1 & 2 \end{bmatrix}.$$

17 Conditionnement d'une matrice

MOTS CLÉS : Conditionnement d'une matrice, erreur relative, amplification d'erreur, préconditionnement

---

Exercice 17.1 Influence des erreurs d'arrondi sur la résolution d'un système linéaire

On cherche à résoudre le système $Ax = b$ où

$$A = \begin{pmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{pmatrix} \text{ et } b = \begin{pmatrix} 32 \\ 23 \\ 33 \\ 31 \end{pmatrix}$$

1. Calculez avec Matlab la solution exacte du système. On utilisera la commande `\`. Quelle est la différence entre l'instruction `A\b` et l'instruction `inv(A)*b` ?
2. Calculez la solution du système perturbé $(A + \Delta A)y = b$ où

$$\Delta A = \begin{pmatrix} 0 & 0 & 0.1 & 0.2 \\ 0.08 & 0.04 & 0 & 0 \\ 0 & -0.02 & -0.11 & 0 \\ -0.01 & -0.01 & 0 & -0.02 \end{pmatrix}$$

3. On note $\Delta x = y - x$. Déduisez-en le coefficient d'amplification d'erreur $\dfrac{\|\Delta x\|_2}{\|x + \Delta x\|_2} \times \dfrac{\|A\|_2}{\|\Delta A\|_2}$. On pourra utiliser la fonction `norm`. Que remarquez-vous ?
4. Calculez la solution du système perturbé $Az = b + \delta b$ avec

$$\delta b = \begin{pmatrix} 0.01 \\ -0.01 \\ 0.01 \\ -0.01 \end{pmatrix}$$

   On note $\delta x = z - x$. En déduire le coefficient d'amplification d'erreur $\dfrac{\|\delta x\|_2}{\|x\|_2} \times \dfrac{\|b\|_2}{\|\delta b\|_2}$. Que remarquez-vous ?
5. Calculez le conditionnement de la matrice A en norme 2, $\text{cond}_2(A) = \|A\|_2 \|A^{-1}\|_2$. Retrouvez ce résultat avec la fonction `cond`.
6. Calculez les valeurs propres de la matrice A. On note $\lambda_{\max}$ la plus grande d'entre elles en valeur absolue et $\lambda_{\min}$ la plus petite. Comparez le rapport $\dfrac{\lambda_{\max}}{\lambda_{\min}}$ avec le conditionnement.
7. Reprenez les questions précédentes en utilisant la norme $\|\cdot\|_\infty$ à la place de la norme $\|\cdot\|_2$.

Exercice 17.2 Erreurs relatives et conditionnement

On considère la matrice

$$A = \begin{pmatrix} 1 & 100 \\ 0 & 1 \end{pmatrix}$$

1. Calculez le conditionnement de la matrice A en norme 1, $\text{cond}_1(A) = \|A\|_1\|A^{-1}\|_1$.
2. On considère les systèmes suivants :

   $$Ax = b \text{ et } A(x+\delta x) = b+\delta b, \text{ avec } b = (100,1)^t \text{ et } b+\delta b = (100,0)^t$$

   Calculez la variation relative $\dfrac{\|\delta x\|_1}{\|x\|_1}$ de la solution et celle du second membre $\dfrac{\|\delta b\|_1}{\|b\|_1}$. Quel est le facteur d'amplification de l'erreur, c'est-à-dire $\dfrac{\|\delta x\|_1}{\|x\|_1} \times \dfrac{\|\delta b\|_1}{\|b\|_1}$?

Exercice 17.3 Déterminant et conditionnement

1. Calculez en fonction de n le déterminant et le conditionnement de la matrice carrée A d'ordre n définie par $\text{diag}(1, 10, ..., 10)$.
2. De même, calculez en fonction de n le déterminant et le conditionnement de la matrice carrée B d'ordre n définie par :

   $$\begin{cases} B_{i,i} &= 1, & 1 \le i \le n, \\ B_{i,i+1} &= 2, & 1 \le i \le n-1, \\ B_{i,j} &= 0, & \text{sinon .} \end{cases}$$

3. Qu'en concluez-vous ? Y a-t-il un lien entre déterminant et conditionnement ?

Exercice 17.4 Notion de préconditionnement

Soit $A = \begin{pmatrix} 1 & 0 \\ 0 & 10^{-6} \end{pmatrix}$ et $\Delta A = \begin{pmatrix} 10^{-8} & 0 \\ 0 & 10^{-14} \end{pmatrix}$

1. Calculez le conditionnement de A en norme 2.
2. Que vaut $\dfrac{\|\Delta A\|_2}{\|A\|_2}$? Déduisez-en une estimation de $\dfrac{\|\Delta x\|_2}{\|x\|_2}$, où x et $x+\Delta x$ sont solutions des systèmes $Ax = b$ et $(A+\Delta A)(x+\Delta x) = b$ avec b un vecteur quelconque.
3. Que vaut le conditionnement de $2A$, $-4A$, puis de αA avec α un réel quelconque ?
4. Soit $D = \begin{pmatrix} 1 & 0 \\ 0 & 10^6 \end{pmatrix}$. Exprimez DA et $D\Delta A$. Que vaut le conditionnement de DA en norme 2 ? Que vaut $\dfrac{\|D\Delta A\|_2}{\|DA\|_2}$? Déduisez-en une nouvelle estimation de $\dfrac{\|\Delta x\|_2}{\|x\|_2}$? Quelle conclusion en tirez-vous ?

18 Schémas numériques pour les équations différentielles

MOTS CLÉS : Méthode d'Euler explicite, d'Euler implicite, de Runge-Kutta d'ordre 2, de Runge-Kutta d'ordre 4.

Travaux de préparation :

1. Pour la résolution de l'équation différentielle

$$y'(x) = f(x, y(x)),$$

on s'intéresse aux méthodes numériques suivantes : Euler explicite, Euler implicite (également appelé Euler rétrograde), Runge-Kutta d'ordre 2, Runge-Kutta d'ordre 4. Présentez chaque schéma sous la forme générique :

$$\forall n \in \mathbb{N}, \quad y_{n+1} = y_n + \cdots$$

---

* *
*

Exercice 18.1 Résolution numérique avec les méthodes d'Euler explicite, Euler implicite et Runge-Kutta

1. Ecrivez une fonction `Y = eulerexplicite(f,y0,dx,n)` qui retourne les n premières valeurs de la résolution numérique à pas constants dx de l'équation différentielle munie de la condition initiale :

$$\begin{cases} y'(x) &= f(x, y(x)) \\ y(0) &= y_0 \end{cases}$$

avec la méthode d'Euler explicite.

2. Ecrivez une fonction `Y = rk2(f,y0,dx,n)` qui produit le même résultat en utilisant la méthode de Runge-Kutta à l'ordre 2.
3. Ecrivez une fonction `Y = rk4(f,y0,dx,n)` qui produit le même résultat en utilisant la méthode de Runge-Kutta à l'ordre 4.
4. Ecrivez une fonction `Y = eulerimplicite(f,y0,dx,n)` qui produit le même résultat en utilisant cette fois-ci la méthode d'Euler implicite.
Vous pourrez utiliser la commande `fzero` et créer une fonction `z = Phi(y)` à paramètres dans un fichier indépendant. Pour les paramètres, vous pourrez utiliser des variables *var* globales en indiquant `global var` partout où vous emploierez ces variables. Pour appeler votre fonction, pensez à utiliser `@` comme dans l'exemple suivant : `fzero(@Phi , . )` .

Exercice 18.2 Application

1. Appliquez les fonctions écrites dans l'exercice 1 à la résolution du problème

$$\left\{ \begin{array}{rcl} \dfrac{\mathrm{d}y}{\mathrm{d}x} & = & -2xy \\ y(0) & = & 1 \end{array} \right.$$

   avec un pas de résolution $\mathrm{d}x$ et un nombre d'itérations n que vous jugerez appropriés. Quelle est la solution exacte de ce problème ?

2. Comparez graphiquement vos résultats numériques à la solution exacte de ce problème.

A Construction de relations de Bezout et applications

MOTS CLÉS : Relations de Bézout, algorithme d'Euclide, algorithme de Yun, décomposition en éléments simples

---

Travaux de préparation :

1. Soit P le polynôme

$$P(x) = \prod_{i=1}^{k} (X - \alpha_i)^{\nu_i} \in \mathbb{C}[X]$$

où les $\alpha_i \in \mathbb{C}$ sont les racines distinctes de P et $\nu_i \in \mathbb{N}^\times$ leur multiplicité. Montrez que

$$\frac{P}{pgcd(P, P')} = \prod_{i=1}^{k} (X - \alpha_i)$$

2. Traitez la question 1. de l'exercice 1 et la question 2. de l'exercice 2.

---

* *
*

Dans le cadre de nombreux algorithmes fonctionnant avec des polynômes, il est parfois nécessaire d'écrire un polynôme $g \in \mathbb{Q}[X]$ selon une décomposition sans facteurs carrés, c'est-à-dire, de trouver les uniques polynômes unitaires g_1, $\ldots g_m \in \mathbb{Q}[X]$, sans facteurs carrés, deux à deux premiers entre eux, tels que $g = g_1 \cdot g_2^2 \cdots g_m^m$. Cette décomposition est obtenue par l'algorithme de Yun suivant :

Algorithme 4 (Yun) Entrée : $g \in \mathbb{Q}[X]$.
Sortie : décomposition sans facteurs carrés de g.

1. $u \leftarrow pgcd(g, g')$; $\quad v_1 \leftarrow \dfrac{g}{u}$; $\quad w_1 \leftarrow \dfrac{g'}{u}$

2. $i \leftarrow 1$
   répéter
   $h_i \leftarrow pgcd(v_i, w_i - v_i')$; $\quad v_{i+1} \leftarrow \dfrac{v_i}{h_i}$; $\quad w_{i+1} \leftarrow \dfrac{w_i - v_i'}{h_i}$
   $i \leftarrow i + 1$
   jusqu'à ce que $v_i = 1$
   $k \leftarrow i - 1$

3. retourner $(h_1, h_2, \ldots, h_k)$.

On peut vérifier qu'au cours de l'algorithme, $v_i = \prod_{i \leqslant j \leqslant m} g_j$ et que $w_i = \sum_{i \leqslant j \leqslant m} (j - i) g_j' \dfrac{v_{i+1}}{g_j}$

Exercice A.1 Algorithme de Yun

1. Soit le polynôme $p = (x-1)(x-2)(x-3)^2(x-5)^3$. Donnez sans Maple sa décomposition sans facteurs carrés.
2. Ecrivez une procédure `Yun := proc(g)` qui renvoie la décomposition sans facteurs carrés de g. La dérivation d'une expression s'obtient par `diff`.
3. Essayez votre procédure sur les polynômes suivants :

$$p_1 = (x-1)(x-2)\cdot(x-3)^2\cdot(x-5)^3$$

$$p_2 = x\cdot((x-2)(x+4))^2\cdot\left(x^2-2x+9\right)^5$$

$$p_3 = 39366+72171x+1458x^2-64098x^3-8316x^5-40986x^4+28x^8+242x^7+134x^6+x^9$$

Une fraction rationnelle est un quotient $f = \dfrac{p}{q}$ de deux polynômes p et q, où q est de plus unitaire. Nous supposerons que $p, q \in \mathbb{Q}[X]$, que $\deg p < \deg q$ et que q se factorise dans $\mathbb{Q}[X]$ en un produit de polynômes irréductibles $q = \prod_{i=1}^{m} q_i^{\alpha_i}$ où chaque facteur q_i appartient à $\mathbb{Q}[X]$ et $\alpha_i \in \mathbb{N}$.

Décomposer $f = \dfrac{p}{q}$ en éléments simples revient à écrire f sous la forme :

$$\frac{p}{q} = \sum_{i=1}^{m}\sum_{j=1}^{\alpha_i}\frac{h_{i,j}}{q_i^j}$$

où $\deg h_{i,j} < \deg q_i$, et peut être réalisé par maple à l'aide de `convert( f , parfrac)`.

Supposons pour simplifier que q ne compte que deux facteurs irréductibles : $q = q_1\cdot q_2$, $\alpha_1 = 1$ et $\alpha_2 = 1$. Soient s_1 et s_2 deux polynômes tels que $s_1q_1+s_2q_2 = 1$ (relation de Bézout). Alors

$$\frac{p}{q} = \frac{p(s_1q_1+s_2q_2)}{q} = \frac{ps_1}{q_2}+\frac{ps_2}{q_1}$$

Après division euclidienne de ps_1 par q_2 et ps_2 par q_1, en ne gardant que les restes u_2 et u_1, on obtient

$$\frac{p}{q} = \frac{u_1}{q_1}+\frac{u_2}{q_2}$$

avec $\deg u_1 < \deg q_1$ et $\deg u_2 < \deg q_2$.

Lorsque $\alpha_1 \neq 1$, on peut comme précédemment construire u_1 et u_2 de sorte que

$$\frac{p}{q} = \frac{u_1}{q_1^{\alpha_1}}+\frac{u_2}{q_2}$$

avec $\deg u_1 < \alpha_1 \deg q_1$ et $\deg u_2 < \deg q_2$, puis diviser u_1 par les puissances successives de q_1 pour trouver la décomposition en éléments simples.

Exercice A.2 (Décomposition en éléments simples) Les notations restent identiques à celles du paragraphe précédent.

1. Ecrivez une procédure `Decomposition :=proc(p, q1, q2)` qui calcule les polynômes u_1 et u_2 lorsque g n'a que deux facteurs irréductibles et $\alpha_1 = \alpha_2 = 1$.
2. Expliquez comment trouver la décomposition en éléments simples si $\alpha_1 = 2$, 3 et $\alpha_2 = 1$.
3. Ecrivez une procédure `DecompositionBis :=proc(p, q1, q2)` qui calcule les polynômes u_1 et u_2 lorsque g n'a que deux facteurs irréductibles et $\alpha_1 = 2$, $\alpha_2 = 1$.

B Techniques d'intégration symbolique

MOTS CLÉS : Relation de Bézout, méthode d'Hermite, résultant de Rothstein et Trager

---

Travaux de préparation :

1. Rappelez la définition du résultant. Que peut-on dire lorsque le résultant s'annule ?

---

* *
*

Le but de ce T.P. est d'étudier une technique d'intégration symbolique de fractions rationnelles, c'est-à-dire de fonctions de la forme $\frac{p}{q}$ où p et q sont tous deux des polynômes. La notation $\int f$ désignera improprement une des primitives de f.

Nous avons vu au cours du TP 11 comment transformer des fractions rationnelles en somme de termes de la forme $\dfrac{f}{g^n}$ où n est un entier non nul, f et g sont deux polynômes tels que $\deg f < \deg g$ et g est un polynôme sans facteur carré.

Lorsque $n > 1$, la première étape consiste à trouver à l'aide d'une relation de Bézout $v_1 g + v_2 g' = 1$, des polynômes $u_1 = f v_1 \mod g'$ et $u_2 = f v_2 \mod g$ qui vérifient $f = u_1 g + u_2 g'$ et $\deg u_1 < \deg g'$, $\deg u_2 < \deg g$. Par intégration par partie, on obtient,

$$\begin{aligned}\int \frac{f}{g^n} &= \int \frac{u_1}{g^{n-1}} + \int \frac{u_2 g'}{g^n} = \int \frac{u_1}{g^{n-1}} + \frac{-u_2}{(n-1)g^{n-1}} + \int \frac{u_2'}{(n-1)g^{n-1}} \\ &= \frac{u_2}{(1-n)g^{n-1}} + \int \frac{u_1 + u_2'/(n-1)}{g^{n-1}}.\end{aligned}$$

L'exposant $n' = n-1$ qui apparaît dans la nouvelle intégrale à calculer a baissé d'une unité, ce qui justifie l'intérêt de la transformation. Cette transformation s'appelle *transformation d'Hermite*.

Exercice B.1

1. Ecrivez une procédure `Hermite := proc(f,g,n)` qui renvoie le résultat de la transformation d'Hermite, c'est-à-dire la partie intégrée et le nouveau dénominateur à intégrer.
2. Ecrivez une procédure `HermiteItere := proc(f,g,n)` qui renvoie une primitive de $\frac{f}{g^n}$ sous la forme $[\frac{a}{b}, h]$ d'une somme

$$\int \frac{f}{g^n} = \frac{a}{b} + \int \frac{h}{g}$$

Supposons que g est sans facteur carré, que $\deg h < \deg g$ et écrivons sa décomposition $g(X) = \prod_{\ell=1}^{k} (X - \alpha_\ell)$ où les $(\alpha_\ell)_{1 \leqslant \ell \leqslant k}$ sont tous distincts. Alors, en connaissant une décomposition en éléments simples

$$\frac{h}{g} = \sum_{\ell=1}^{k} \frac{c_\ell}{X - \alpha_\ell}$$

on peut en déduire

$$\int \frac{h}{g} = \sum_{\ell=1}^{k} c_\ell \ln(X - \alpha_\ell) \tag{4}$$

Trouver les racines d'un polynôme est une opération difficile. Le lemme suivant montre qu'il n'est pas nécessaire de calculer toutes les racines (α_ℓ), mais que l'on peut grouper les termes qui possèdent la même valeur c_ℓ en un terme $c_i \ln v_i = c_i \ln\Big(\prod_{\substack{\ell \ tq \\ c_\ell = c_i}} (X - \alpha_\ell)\Big)$ directement calculable.

Lemme 5 *Avec les même notations, soit $R(Y)$ le resultant $R(Y) = Res_x(h - Yg', g) \in \mathbb{Q}[Y]$. Soient $(\gamma_i)_{1 \leqslant i \leqslant m}$ les zéros de $R(Y)$ et $v_i = pgcd(h - \gamma_i g', g)$, alors*

$$\int \frac{h}{g} = \sum_{i=1}^{m} \gamma_i \ln(v_i) \tag{5}$$

On a vu au TP 11 que le résultant $R(\gamma)$ s'annule pour une certaine valeur γ si et seulement si $(h - \gamma g')$ et g ont une racine commune, autrement dit, si et seulement s'il existe un indice ℓ tel que $h(\lambda_\ell) - \gamma g'(\alpha_\ell) = 0$. Ainsi $R(\gamma) = 0$ équivaut à l'existence d'un indice ℓ tel que $\gamma = \frac{h(\alpha_\ell)}{g'(\alpha_\ell)}$.

Soit γ_i une racine de R et $v_i = pgcd(h - \gamma_i g', g)$. Montrons que v_i a exactement pour facteurs les $(X - \alpha_\ell)$ pour ℓ tel que $\gamma_i = \frac{h(\alpha_\ell)}{g'(\alpha_e ll)}$. Si $\gamma_i = \frac{h(\alpha_\ell)}{g'(\alpha_e ll)}$, alors α_ℓ est racine de $h - \gamma_i g'$ et de g, donc $(X - \alpha_i)$ divise leur pgcd. Réciproquement, lorsqu'un facteur $(X - \omega)$ divise R, autrement dit lorsque $R(\omega)$ est nul, nous savons déjà que ω est l'une des racines α_ℓ telle que $\gamma_i = \frac{h(\alpha_\ell)}{g'(\alpha_e ll)}$, ce qui achève la preuve.

Exercice B.2

1. A l'aide du lemme précédant, écrivez une procédure `RothsteinTrager := proc(a,b)` qui essaye de calculer la primitive $\int \frac{a}{b}$ lorsque b est sans facteurs carré.
2. Ecrivez une procédure `Primitive := proc(f,g,n)` qui recherche une primitive de $\frac{f}{g^n}$.
3. Calculez une primitive de $\dfrac{1}{x^2 - 2}$, de $\dfrac{6x^3 + 13x^2 + 28 - 72x}{x^4 - 48 - 8x^2 + x^3 + 4x}$ et de $\dfrac{1}{x^6 - 12x^5 + 57x^4 - 136x^3 + 171x^2 - 108x + 27}$.
4. Comparez le nombre de racines à calculer par l'égalité (4) et (5) lorsqu'on cherche $\int \frac{1}{x^3 + x}$. Concluez.

C Méthodes de multiplication rapide

MOTS CLÉS : Multiplication de Strassen, de Karatsuba, FFT

Travaux de préparation :

1. Comptez le nombre d'additions et de multiplication de scalaires nécessaires au calcul du produit de deux matrices carrés A et B de taille n par la méthode usuelle et vérifiez que l'algorithme de Strassen est meilleur.
2. Combien de multiplications de scalaires doit-on effectuer pour calculer le produit de deux polynômes de degré n par une méthode naïve ? Comparez avec les méthode de Karatsuba et de transformée de Fourier rapide.

* *
*

L'algorithme de multiplication rapide de Strassen permet de calculer le produit de deux matrices carrées A et B de taille n en effectuant moins d'opérations élémentaires (additions ou multiplications de scalaires) que par la méthode usuelle.

On divise les matrices en blocs de taille $n/2$.

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{pmatrix} \qquad B = \begin{pmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{pmatrix}$$

On calcule les sommes

$$\begin{array}{ll} S_1 = A_{2,1} + A_{2,2} & T_1 = B_{1,2} - B_{1,1} \\ S_2 = S_1 - A_{1,1} & T_2 = B_{2,2} - T_1 \\ S_3 = A_{1,1} - A_{2,1} & T_3 = B_{2,2} - B_{1,2} \\ S_4 = A_{1,2} - S_2 & T_4 = T_2 - B_{2,1} \end{array}$$

puis (comme il s'agit de produit, on utilise récursivement l'algorithme de Strassen),

$$\begin{array}{ll} P_1 = A_{1,1}B_{1,1} & P_5 = S_1T_1 \\ P_2 = A_{1,2}B_{2,1} & P_6 = S_2T_2 \\ P_3 = S_4B_{2,2} & P_7 = S_3T_3 \\ P_4 = A_{2,2}T_4 & \end{array}$$

et enfin

$$\begin{array}{ll} U_1 = P_1 + P_2 & U_5 = U_4 + P_3 \\ U_2 = P_1 + P_6 & U_6 = U_3 - P_4 \\ U_3 = U_2 + P_7 & U_7 = U_3 + P_5 \\ U_4 = U_2 + P_5 & \end{array}$$

et retourner la matrice

$$AB = \begin{pmatrix} U_1 & U_5 \\ U_6 & U_7 \end{pmatrix}$$

On peut démontrer que cet algorithme n'utilise qu'au plus $O(n^{\log_2 7})$ opérations.

Exercice C.1 Implémentez l'algorithme de Strassen.

L'algorithme de Karatsuba permet de calculer le produit de deux polynômes plus rapidement que par une multiplication classique. Soient f et g deux polynômes de degré n (où n est une puissance de 2). Notons F_1, F_0, G_1 et G_0 les polynômes de degré $\frac{n}{2}$ tels que

$$f(x) = \underbrace{f_n x^n + \cdots + f_{n/2} x^{n/2}}_{F_1(x)x^{n/2}} + \underbrace{f_{n/2-1} x^{n/2-1} + \cdots + f_0}_{F_0(x)}$$

$$g(x) = \underbrace{g_n x^n + \cdots + g_{n/2} x^{n/2}}_{G_1(x)x^{n/2}} + \underbrace{g_{n/2-1} x^{n/2-1} + \cdots + g_0}_{G_0(x)},$$

alors en écrivant le produit sous la forme ingénieuse suivante

$$f \cdot g = F_1 G_1 x^n + ((F_0 + F_1)(G_0 + G_1) - F_0 G_0 - F_1 G_1) x^{n/2} + F_0 G_0,$$

il suffit de calculer trois produits de polynômes, à savoir $F_0 \cdot G_0$, $F_1 \cdot G_1$ et $(F_0 + F_1) \cdot (G_0 + G_1)$. En utilisant récursivement l'algorithme pour calculer ces trois produits, on peut montrer que le coût de multiplication de deux polynômes de degré n est en $O(n^{\log_2 3})$ opérations élémentaires (additions et multiplications de scalaires).

Le même algorithme sert encore à multiplier des entiers : écrivons à cette fin des entiers a et b en base 2

$$a = a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_0$$

$$b = b_n 2^n + b_{n-1} 2^{n-1} + \cdots + b_0$$

et identifions a et b aux polynômes $\sum_{i=0}^{n} a_i x^i$ et $\sum_{i=0}^{n} b_i x^i$ évalués en $x = 2$.

Exercice C.2 Programmez une version de l'algorithme de Karatsuba adaptée au calcul du produit de deux entiers.

Si les polynômes f et g s'écrivent

$$f(x) = f_0 + f_1 x + \cdots + f_{n-1} x^{n-1}$$

$$g(x) = g_0 + g_1 x + \cdots + g_{n-1} x^{n-1},$$

leur transformée de Fourier discrète vérifie

$$DFT(f \times g) = DFT(f) \cdot DFT(g)$$

où le symbole $\cdot$ du membre de droite désigne la multiplication coordonnée par coordonnée.

Exercice C.3

1. Donnez la décomposition des entiers suivants en base 2 (on pourra utiliser `dec2bin`). Ecrivez à la main dans chaque cas le vecteur de zéros et de uns correspondant.
2. En utilisant la transformée de Fourier discrète (on pourra utiliser les commandes `fft` et `ifft`), calculez le produit de deux entiers a et b dans les cas suivants :

$a_1 = 1$	$b_1 = 2$
$a_2 = 2$	$b_2 = 3$
$a_3 = 45$	$b_3 = 2$.

D Etude qualitative d'un système d'équations différentielles : le modèle proie prédateur de Volterra–Lotka

MOTS CLÉS : Modèle proie prédateur, portraits de phase, intégrale première

Pour cette séance, on chargera les bibliothèques `LinearAlgebra`, `DEtools` et `plots`. Tout au long du T.P., on veillera à choisir suffisamment de points pour que les graphiques soient lisses.

En 1926, Volterra proposa un modèle simple de système *proie–prédateur* (par exemple le couple requin–sardine ou renards–lapins) afin d'expliquer les oscillations dans les campagnes de pêche dans l'Adriatique. Il s'écrit

$$\begin{cases} \dot{S}(t) &= S(t)\ (a - bR(t)) \\ \dot{R}(t) &= R(t)\ (-c + dS(t)) \end{cases}$$

où t représente le temps, $S(t)$ et $R(t)$ comptent respectivement le nombre de proies et de prédateurs au cours du temps et a, b, c et d sont des paramètres positifs.

La première équation se comprend comme suit : en l'absence de prédateurs, les proies se reproduisent naturellement avec un taux d'accroissement $\frac{\dot{S}(t)}{S(t)} = a$ (on suppose que le plancton est en quantité illimitée dans ce modèle), mais qu'elle sont mangées proportionnellement à la présence de prédateurs (terme $-bR(t)$). De même, en l'absence de proie, les prédateurs meurent avec un taux c, mais leur population s'accroît proportionnellement à la présence de proie (terme $dS(t)$).

Après renormalisation, c'est-à-dire en posant, $u(\tau) = \frac{d}{c}S(t)$, $v(\tau) = \frac{b}{a}R(t)$, $\tau = at$ et $\alpha = \frac{c}{a}$, on obtient le système suivant

$$\begin{cases} \dot{u}(\tau) &= u(\tau)\ (1 - v(\tau)) \\ \dot{v}(\tau) &= \alpha\ v(\tau)\ (u(\tau) - 1) \end{cases}. \tag{6}$$

que nous nous proposons d'étudier. Nous noterons encore Φ l'application

$$\Phi : \begin{array}{ccc} \mathbb{R}^2 & \to & \mathbb{R}^2 \\ (u,v) & \mapsto & (u(1-v), v(u-1)) \end{array}.$$

Exercice D.1 1. Après avoir consulté l'aide de `dsolve/numeric`, recherchez une solution numérique du système (6). Sur un même graphique, représentez les courbes de u et v en fonction de τ à l'aide de la commande `odeplot`.
On prendra les différentes valeurs initiales suivantes $u(0) = 0,4$ ou $u(0) = 2$, et $v(0) = 0,4$ ou $v(0) = 2$. Dans les quatre cas, on commentera l'évolution.
On prendra encore les conditions initiales $u(0) = 1$ et $v(0) = 1$ et on commentera le résultat.

2. En utilisant l'instruction `phaseportrait`, représentez le portrait de phase du système (6), c'est à dire les courbes paramétrées $x(\tau) = u(\tau)$ et $y(\tau) =$

$v(\tau)$ pour $\tau \in [-5, 5]$ et les conditions initiales suivantes $u(0) = 1$ et $v(0) = 1.5$, 2, 2.5, 3 et 3.5. Le modèle rend-il compte des observations faites par les pêcheurs ?

En combinant les deux équations de (6), on peut « deviner »que

$$\frac{\dot{u}}{\dot{v}} = \alpha \frac{v(u-1)}{u(1-v)}$$

qui s'exprime encore si l'on sépare les variables

$$\frac{1-v}{v}\,\mathrm{d}v = \alpha \frac{u-1}{u}\,\mathrm{d}u$$

Par intégration, on « devine »que $H(u, v) = \alpha u + v - \alpha \ln(u) - \ln(v) = H_0$, où H_0 est une constante. Des théorèmes mathématiques permettent de vérifier que en effet, pour toute solution $u(\tau)$, $v(\tau)$ donnée, l'expression $H(u(\tau), v(\tau))$ est constante. On dit que $H(u, v)$ est l'intégrale première du système.

Exercice D.2 Avec `implicitplot`, tracez les lignes de niveau de $H(x, y)$. Comparez avec les trajectoires du portrait de phase.

On appelle point d'équilibre les points u et v tels que $\Phi(u, v) = 0$. Pour étudier le comportement des solutions au voisinage de points critiques, on approxime l'équation $(\dot{u}, \dot{v}) = \Phi(u, v)$ par

$$(\dot{u}, \dot{v}) = D\Phi\ (u, v)$$

où $D\Phi = \begin{pmatrix} \frac{\partial \Phi_u}{\partial u} & \frac{\partial \Phi_u}{\partial v} \\ \frac{\partial \Phi_v}{\partial u} & \frac{\partial \Phi_v}{\partial v} \end{pmatrix}$ est la différentielle de Φ. Cela revient à linéariser le problème. On caractérise les situations selon les critères suivants

Définition 6 *Soient λ_1 et λ_2 les valeurs propres de la matrice $D\Phi$.*

1. *Si λ_1 et λ_2 sont réels et de même signe, le point d'équilibre est un point* noeud.
2. *Si λ_1 et λ_2 sont réels et de signe opposé, le point d'équilibre est un point* selle.
3. *Si λ_1 et λ_2 sont deux complexes conjugés de partie réelle nulle, le point d'équilibre est un point* foyer.
4. *Si λ_1 et λ_2 sont deux complexes conjugés de partie réelle non-nulle, le point d'équilibre est un point* centre.

Exercice D.3

1. Cherchez les points d'équilibre de Φ et calculer numériquement $D\Phi$ pour chacun d'entre eux.
2. Avec la commande `eigenvalue`, déterminez pour chaque point d'équilibre les valeurs propres de la différentielle. Donnez la nature de chaque point d'équilibre.
3. Tracez les portraits de phase du système linéarisé autour des points d'équilibres.

Le modèle étudié jusqu'ici reste toutefois très sommaire. Introduisons un second modèle un peu plus fin. On suppose désormais que la croissance du nombre de proie en l'absence de prédateur suit l'équation logistique.

$$\frac{\dot{S}(t)}{S(t)} = a(1 - \frac{S(t)}{K}) \tag{7}$$

où K représente la capacité maximale du milieu à accueillir des proies.

Exercice D.4 Résolvez par une méthode symbolique l'équation (7) en prenant $a = 1$, $K = 1$ et la condition initiale $u(0) = \frac{1}{2}$. Représentez la solution.

L'effet de la prédation est marqué par un terme $\frac{kS(t)R(t)}{S(t)+e}$ qui a un comportement proportionnel à la quantité de prédateurs (comme précédemment) mais qui est presque proportionnel au nombre de proies disponible lorsque celui-ci est faible et qui « sature »lorsqu'il y a beaucoup de proies. L'évolution de la population de prédateur est également soumis à une loi logistique, dont la capacité du milieu est de l'ordre de la quantité de proie.

Mises sous forme normalisée, les équations deviennent

$$\begin{cases} \dot{u} &= u\,(1-u) - \dfrac{auv}{u+d} \\ \dot{v} &= bv\left(1 - \dfrac{v}{u}\right) \end{cases} \tag{8}$$

Exercice D.5 Montrez que le point $(u^\star, v^\star)$ défini par

$$u^\star = v^\star = \frac{(1-a-d)^2 + \left((1-a-d)^2 + 4d\right)^{1/2}}{2}$$

est un point d'équilibre du système (8) . Etudiez sa nature pour les valeurs suivantes

$$a = 1, \quad b = 0,05, \quad d = 0,2$$

$$a = 1, \quad b = 1, \quad d = 0,2$$

Tracez les portraits de phase correspondant.

E Valeur propre et vecteur propre dominants : méthode de la puissance itérée

MOTS CLÉS : Matrice diagonalisable, méthode de la puissance itérée, valeur propre dominante, vecteur propre dominant

La méthode de la puissance itérée est une méthode itérative de calcul de la valeur propre dominante d'une matrice (*i.e.* celle de plus grand module) et du vecteur propre associé.

Soit A une matrice réelle de taille $n \times n$ diagonalisable. On note ses valeurs propres $(\lambda_i)_{1 \leqslant i \leqslant n}$ et ses vecteurs propres associés $(u_i)_{1 \leqslant i \leqslant n}$. On suppose que les valeurs propres sont distinctes et ordonnées comme suit :

$$|\lambda_1| > |\lambda_2| > ... > |\lambda_n|$$

Considérons, pour un vecteur $w_0 \in \mathbb{C}^n$ quelconque, la suite $(w_k)_{k \in \mathbb{N}} \subset \mathbb{C}^n$ définie pour tout entier k par $w_k = A^k w_0$. Soient $(\alpha_i)_{1 \leqslant i \leqslant n} \in \mathbb{C}^n$ les coordonnées de w_0 dans la base $(u_i)_{1 \leqslant i \leqslant n}$. Il vient alors

$$\begin{aligned} \forall k \in \mathbb{N}, \quad w_k &= A^k w_0 = \sum_{i=1}^{n} \alpha_i \lambda_i^k u_i \\ &= \lambda_1^k \alpha_1 u_1 + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^k u_i \end{aligned}$$

Renormalisons le vecteur w_k en construisant la suite $(v_k)_{k \in \mathbb{N}}$ définie par $v_0 = w_0$ et la relation

$$\forall k \in \mathbb{N}, \quad v_{k+1} = \frac{A v_k}{\|A v_k\|_\infty}$$

Alors la suite $(v_k)_{k \in \mathbb{N}}$ converge vers le vecteur propre dominant u_1 et la suite $\left(\|A v_k\|_\infty\right)_{k \in \mathbb{N}}$ converge vers $|\lambda_1|$.

Exercice E.1 Programmation de la méthode de la puissance itérée

1. Ecrivez une fonction `[x,l] = puissanceiteree2(A,w0,k)` qui, avec les notations précédentes, renvoie v_k et $\|A v_k\|_\infty$. On pourra utiliser la commande `norm`.
2. Appliquez cette fonction à la recherche du vecteur propre et de la valeur propre dominants de la matrice

$$M = \begin{bmatrix} -4 & 14 & 0 \\ -5 & 13 & 0 \\ -1 & 0 & 2 \end{bmatrix}$$

   avec le vecteur initial $x_0 = (1, 1, 1)^t$.
3. Validez votre résultat en utilisant la fonction Matlab `eig`.

Exercice E.2 Influence de la condition initiale
On considère la matrice

$$N = \begin{bmatrix} 0.5172 & 0.5473 & -1.224 & 0.8012 \\ 0.5473 & 1.388 & 1.353 & -1.112 \\ -1.224 & 1.353 & 0.03642 & 2.893 \\ 0.8012 & -1.112 & 2.893 & 0.05827 \end{bmatrix}$$

1. Pour les conditions initiales suivantes :

$$x_0 = (1,0,0,0)^t,\ y_0 = (1,1,1,1)^t \text{ et } z_0 = (1,1,0,0)^t,$$

   représentez graphiquement la suite $\left(\|v_k - u_1\|_2\right)_{k\in\mathbb{N}}$.
2. Même question, en utilisant la norme $\|\cdot\|_\infty$.

Exposés

Critères d'évaluation des exposés :

Implication dans le projet et degré d'approfondissement des questions, initiatives par rapport au sujet

Prise de risque par rapport à la difficulté du sujet et des points abordés dans la présentation

Maîtrise des mathématiques mises en jeu dans le projet, précision et exactitude des justifications théoriques, justesse et qualité des programmes.

Qualité et clarté (écrit) du document maple ou matlab présenté, précision des commentaires des programmes

Qualité et clarté (oral) des exposés, cohérence de la présentation

Réactivité pertinence des réponses aux questions, défense des choix opérés par le groupe.

Impression d'ensemble

Sujet des exposés : Dans un exposé de mathématique illustré par des exemples et des programmes de votre choix, vous traiterez par groupe de deux l'un des sujets suivants.

1. Techniques de résolution exacte ou approchée d'un système d'équations linéaires.
2. Outils de l'arithmétique des entiers et applications
3. Méthodes de recherche approchée de zéro de fonction
4. Equations polynomiales et systèmes d'équations polynomiales
5. Interpolation polynomiale
6. Transformée de Fourier discrète et application
7. Processus d'accélération de convergence de séries
8. Schémas numériques pour les équations différentielles
9. Méthode de quadrature et d'intégration numérique.
10. Problèmes de précision et de stabilité numérique.

Vous disposerez de dix minutes de présentation (cinq minutes par orateur) et de cinq minutes pour répondre aux questions du jury.

Références

[1] Notes des journées X UPS 97, http ://www.math.polytechnique.fr/xups/programme97.html

[2] Françoise Chatelin, *Valeurs propres de matrices*, Paris Milan Barcelone Masson (1988).

[3] Philippe G. Ciarlet, *Introduction à l'analyse numérique matricielle et à l'optimisation*, Dunod (1998).

[4] Henri Cohen, *A course in Computational Algebraic Number Theory*, Springer GTM fourth printing (2000).

[5] Michel Crouzeix & Alain L. Mignot, *Analyse numérique des équations différentielles*, Masson 2ème édition(1992).

[6] Jean-Pierre Demailly, *Analyse numérique et équations différentielles*, Presses universitaires de Grenoble (1996).

[7] Michel Demazure, *Cours d'algèbre*, Cassini, Paris (1997).

[8] Jean Dieudonné, *Calcul infinitésimal*, Hermann, Paris, nouveau tirage (1997).

[9] Joachim von zur Gathen & Jürgen Gerhard, *Modern Computer Algebra*, Cambridge University Press 2nd edition (2003).

[10] Xavier Gourdon et Pascal Sebah, *Numbers, constants and computations*, http ://numbers.computation.free.fr/Constants/constants.html

[11] John Hubbard & Beverly West, *Equations différentielles et systèmes dynamiques*, Cassini (1999).

[12] Donald Knuth, *The art of computer programming.*

[13] James D. Murray, *Mathematical Biology*, Springer (1989)

[14] Bernadette Perrin-Riou, *Algèbre, arithmétique et maple*, Cassini, Paris (2000).

[15] Laurent Schwartz, *Méthodes mathématiques pour les sciences physiques*, Hermann (1998).

[16] Denis Serre, *Matrices, Theory and Applications*, Graduate Texts in Mathematics, Springer (2000)

[17] Grégory Vial, Corpus de documents pour le calcul scientifique http ://www.bretagne.ens-cachan.fr/math/people/gregory.vial /enseignement.php ?menu=ens

[18] Gilles Zémor, *Cours de cryptographie*, Paris, Cassini (2000).