Travaux pratiques
calcul formel
CPGE Première année
Stage d'enseignants de la faculté des sciences de l'Université Libanaise à Lyon
Journée du 12 septembre à l'INRP

Gilles Aldon

Septembre 2007

Table des matières

Chapitre 1

Introduction

1.1 Première approche

Maple, Maxima, Xcas, Mathematica sont des systèmes de calcul formel, c'est à dire des logiciels qui permettent de faire des mathématiques en manipulant des expressions symboliques. Contrairement à la plupart des langages classiques de programmation ils peuvent traiter non seulement des quantités numériques (entières, réelles, complexes) mais aussi des polynômes, des fonctions, des séries,... et d'effectuer des opérations courantes, dérivation, intégration, limites, simplifications,...

De plus ils permettent aussi de faire les calculs numériques classiques, solution d'équations ou de systèmes, résolution d'équations,...

Dans cette brochure, nous donnerons les exemples a priori en Maple et en Maxima lorsque des différences significatives apparaîtront ;

La première approche de ces logiciels de calcul formel se fera à travers l'étude et la représentation graphique d'une fonction de R dans R puis en effectuant quelques calculs numériques :

$$f : x \to \frac{e^x}{\sqrt{x+1}}$$

1.1.1 Définition de la fonction

Noter bien le point-virgule qui termine une ligne Maple.

```
> f:=x->exp(x)/sqrt(x+1);
```

$$f := x \to \frac{e^x}{\sqrt{x+1}}$$

```
> f(1);
```

$$\frac{1}{2}\, e\, \sqrt{2}$$

```
> f(0);
```

$$1$$

```
> f(-1);
```

```
Error, (in f) numeric exception: division by zero
```

Jusque là, rien que de très prévisible...

```
> f(-2);
```

$$-I\,e^{(-2)}$$

La syntaxe est ici complètement identique et les réponses données également avec Maxima.

Notez ici que Maple et Maxima travaillent a priori dans l'ensemble des complexes et que le travail de détermination du domaine de définition de la fonction dans R vous revient !

1.1.2 Calcul de la dérivée

Avec Maple, deux façons différentes permettent d'obtenir l'expression de la dérivée d'une fonction :

```
> der:=diff(f(x),x);
```

$$der := \frac{e^x}{\sqrt{x+1}} - \frac{1}{2}\,\frac{e^x}{(x+1)^{(3/2)}}$$

```
> simplify(der);
```

$$\frac{1}{2}\,\frac{e^x\,(2\,x+1)}{(x+1)^{(3/2)}}$$

```
> df:=x->D(f)(x);
```

$$df := x \to \frac{e^x}{\sqrt{x+1}} - \frac{1}{2}\,\frac{e^x}{\sqrt{x+1}^3}$$

```
> df(-1/2);
```

$$0$$

```
> solve(df(x)<0,x);
```

$$\mathrm{RealRange}(\mathrm{Open}(-1),\ \mathrm{Open}(\frac{-1}{2}))$$

```
> der(2);
```

$$\frac{(e^x)(2)}{\sqrt{\mathrm{x}(2)+1}} - \frac{1}{2}\,\frac{(e^x)(2)}{(\mathrm{x}(2)+1)^{(3/2)}}$$

```
> subs(x=-1/2,der);
```

$$0$$

Avec Maxima, on utilisera d'abord la commande *diff* puis on définira la fonction dérivée en recopiant le résultat obtenu :

```
(\%i18) diff(f(x),x);
```

(%o18) $$\frac{e^x}{\sqrt{x+1}} - \frac{e^x}{2\,(x+1)^{\frac{3}{2}}}$$

```
(%i19) df(x):=%e^x/sqrt(x+1)-%e^x/(2*(x+1)^(3/2));
```

(%o19) $$df\,(x) := \frac{e^x}{\sqrt{x+1}} - \frac{e^x}{2\,(x+1)^{\frac{3}{2}}}$$

```
(%i20) df(1);
```

(%o20) $$\frac{3\,e}{4\,\sqrt{2}}$$

1.1.3 Calcul des limites

```
> Limit(f(x),x=+infinity);
```

$$\lim_{x\to\infty} \frac{e^x}{\sqrt{x+1}}$$

```
> value(%);
```

$$\infty$$

```
> limit(f(x),x=infinity);
```

$$\infty$$

```
> limit(f(x),x=-1,right);
```

$$\infty$$

Pour Maxima, le calcul direct de la limite en -1 par valeur supérieur[1] ne rend pas de résultat, mais en demandant un traitement en utilisant Taylor la réponse est donnée :

```
(\%i21) limit(f(x),x,+inf);
```

(%o21) $$\infty$$

```
(%i22) limit(f(x), x, -1,plus);
```

(%o22) $$\lim_{x\to 0} \frac{e^{x-1}}{\sqrt{x}}$$

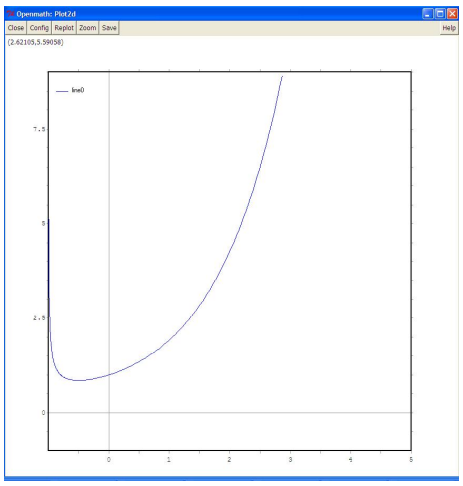
```
(%i23) tlimit(f(x), x, -1, plus);
```

(%o23) $$\infty$$

[1] on utilisera *plus* et *minus* pour les limites à droite et à gauche

1.1.4 Représentation graphique

```
> plot(f(x),x=-1.1..3,y=-0.1..10);
(\%i24) plot2d([f(x)], [x,-1,5],[y,-1,9],[plot_format, openmath]);
```



1.2 Calcul approché, calcul exact

Par défaut, Maple calcule en valeurs exactes. Il permet les calculs habituels sur les nombres :

```
> 2^100;
```

$$1267650600228229401496703205376$$

```
> 100!;
```

$$933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000$$

```
> ifactor(%); # factor pour Maxima
```

$$(2)^{97}\ (3)^{48}\ (5)^{24}\ (7)^{16}\ (11)^{9}\ (13)^{7}\ (17)^{5}\ (19)^{5}\ (23)^{4}\ (29)^{3}\ (31)^{3}\ (37)^{2}\ (41)^{2}\ (43)^{2}\ (47)^{2}\ (53)\ (59)\ (61)\ (67)\ (71)\ (73)\ (79)\ (83)\ (89)\ (97)$$

```
> isprime(1234567891); #primep(1234567891) pour Maxima
```

$$true$$

```
> nextprime(123456789); #next_prime(123456789) pour Maxima
```

$$123456791$$

```
> 120 mod 19; # mod(120,19) pour Maxima
```

6

Il est possible d'utiliser ces logiciels pour effectuer des calculs approchés :

```
> evalf(120/17); #float(120/17) ou bfloat(120/17) pour Maxima
```

7.058823529

```
> evalf(120/17,100);
```

7.05882352941176470588235294117647058823529411764705882352941176470588\
23529411764705882352941176470588235294117647059

```
> evalf[100](120/17);
```

7.05882352941176470588235294117647058823529411764705882352941176470588\
23529411764705882352941176470588235294117647059

1.3 Suites, Ensembles et Listes

1.3.1 Suites

La fonction "seq" est utilisée pour construire une suite de valeurs ; typiquement, avec Maple, on l'utilisera de la manière suivante :

$$seq(f(i), i = m..n)\,;$$

qui génèrera la suite de valeurs $f(m)$, $f(m+1)$, $\cdots$, $f(n)$.

L'appel

$$seq(f(i), i = x)\,;$$

génère une suite en appliquant f à une liste (ou un ensemble) x.

1.3.2 Ensembles et Listes

Un ensemble (set) est une suite non ordonnée d'expressions distinctes entourées d'accolades : c'est bien la notion usuelle d'ensemble mathématique.

e :={a,b,c,d} ; pour Maple et e :{a,b,c,d} pour Maxima

Une liste est une suite ordonnée d'expressions entourées de crochets :

L :=[a,b,c,d,a] ; pour Maple et L :[a,b,c,d,a] pour Maxima

L'ensemble vide sera représenté par {}. La liste vide par [].

Les éléments d'un ensemble ou d'une liste peuvent être extraits de l'ensemble ou de la liste : ainsi le i^{eme} élément de l'ensemble S s'obtiendra par :

S[i] ;

Avec Maple, les éléments compris entre le i^{eme} et le j^{eme} par :

S[i..j] ;

Pour rajouter un élément à une liste on utilisera la fonction op avec Maple et la fonction *append* avec Maxima :

l :=[op(l),x] ; l :append(l,[x])

Dans un ensemble, on utilisera l'opérateur "union" :

S :=S union {x} ; pour Maple et S :union(S,{e}) pour Maxima

Remplacer le i^{eme} élément d'une liste peut se faire avec Maple en utilisant la fonction "subsop" :

l :=subsop(i=x,l) ;
l :=subsop(i=NULL,l) ;

Supprimer un élément x d'un ensemble S se fera en utilisant l'opérateur "minus" avec Maple et "delete" avec Maxima

S :=S minus x ;delete(S,x)

1.4 Exercices

1. Faire l'étude complète de la fonction

$$f : x \to f(x) = \left(x + \frac{1}{x}\right)^x$$

2. Peut on trouver une fraction dont l'écriture décimale illimitée est :

$$0,\overline{123456789}$$

3. Les nombres de métal
   (a) Quelle est la solution positive de l'équation : $x^2 = x + 1$ [2]
   (b) Quelle est la solution positive de l'équation $x^3 = x^2 + x + 1$ [3]
   (c) Montrez que l'équation

$$x^n = \sum_{k=0}^{n-1} x^k$$

   a toujours une unique solution positive notée u_n.

[2] le nombre d'or
[3] le nombre d'argent ? !

(d) La suite u converge t'elle ? Si oui vers quel nombre ?

4. Soit la somme :

$$S_n = \sum_{i=0}^{n} \frac{1}{16^i} \left(\frac{4}{8i+1} - \frac{2}{8i+4} - \frac{1}{8i+5} - \frac{1}{8i+6} \right)$$

(a) Calculer une valeur approchée de S_n pour n allant de 1 à 20.

(b) Calculer la limite de S_n lorsque n tend vers l'infini.

1.5 Eléments de correction

Etude de fonction

```
> solve(x+1/x>0,x);
```

$$\text{RealRange}(\text{Open}(0),\ \infty)$$

```
> f:=x->(x+1/x)^x;
```

$$f := x \to (x + \frac{1}{x})^x$$

```
> df:=x->D(f)(x);
```

$$df := x \to \mathrm{D}(f)(x)$$

```
> dlog:=x->df(x)/f(x);
```

$$dlog := x \to \frac{\mathrm{df}(x)}{\mathrm{f}(x)}$$

```
> simplify(dlog(x));
```

$$\frac{\ln(\frac{x^2+1}{x})\, x^2 + \ln(\frac{x^2+1}{x}) + x^2 - 1}{x^2+1}$$

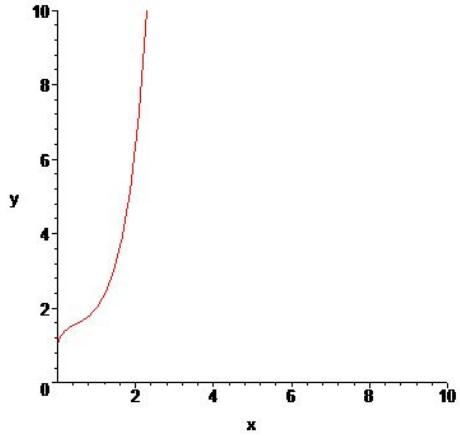
```
> limit(f(x),x=infinity);
```

$$\infty$$

```
> limit(f(x),x=0,right);
```

$$1$$

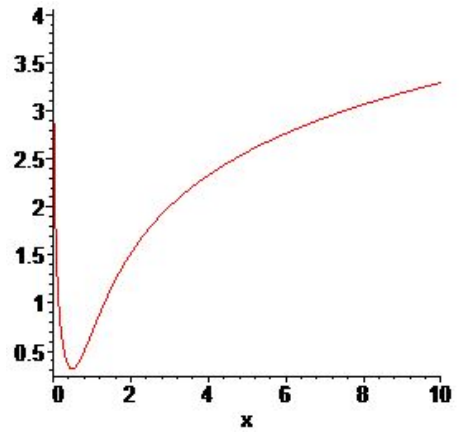
```
> plot(f(x),x=0..10,y=0..10);
```




```
> g:=x->ln((x^2+1)/x)+(x^2-1)/(x^2+1);
```

$$g := x \to \ln(\frac{x^2+1}{x}) + \frac{x^2-1}{x^2+1}$$

```
> plot(g(x),x=0..10);
```




```
> dg:=x->D(g)(x);
```

$$dg := x \to \mathrm{D}(g)(x)$$

```
> simplify(dg(x));
```

$$\frac{x^4 - 1 + 4\,x^2}{x\,(x^2+1)^2}$$

```
> solve(dg(x)>0);
```

$$\mathrm{RealRange}(\mathrm{Open}(-\sqrt{-2+\sqrt{5}}),\ \mathrm{Open}(0)),\ \mathrm{RealRange}(\mathrm{Open}(\sqrt{-2+\sqrt{5}}),\ \infty)$$

```
> g(sqrt(sqrt(5)-2));
```

$$\ln(\frac{\sqrt{5}-1}{\sqrt{-2+\sqrt{5}}})+\frac{\sqrt{5}-3}{\sqrt{5}-1}$$

```
> evalf(%);
```

$$0.3157191045$$

Avec Maxima :

```
(\%i1) f(x):=(x+1/x)^x;
```

(%o1) $$f\left(x\right):=\left(x+\frac{1}{x}\right)^{x}$$

```
(%i2) diff(f(x),x);
```

(%o2) $$\left(x+\frac{1}{x}\right)^{x}\left(log\left(x+\frac{1}{x}\right)+\frac{\left(1-\frac{1}{x^2}\right)\,x}{x+\frac{1}{x}}\right)$$

```
(%i3) df(x):=(x+1/x)^x*(log(x+1/x)+((1-1/x^2)*x)/(x+1/x));
```

(%o3) $$df\left(x\right):=\left(x+\frac{1}{x}\right)^{x}\left(log\left(x+\frac{1}{x}\right)+\frac{\left(1-\frac{1}{x^2}\right)\,x}{x+\frac{1}{x}}\right)$$

```
(%i4) dlog(x):=df(x)/f(x);
```

(%o4) $$dlog\left(x\right):=\frac{df\left(x\right)}{f\left(x\right)}$$

```
(%i5) dlog(x);
```

(%o5) $$log\left(x+\frac{1}{x}\right)+\frac{\left(1-\frac{1}{x^2}\right)\,x}{x+\frac{1}{x}}$$

```
(%i6) limit(f(x),x,inf);
```

(%o6) $$\infty$$

```
(%i7) limit(f(x),x,0,plus);
```

(%o7) $$1$$

```
(%i8) dg(x):=(x*(2-(x^2+1)/x^2))/(x^2+1)+(2*x)/(x^2+1)+(-2*x*(x^2-1))/(x^2+1)^2;
```

(%o8) $$dg\left(x\right):=\frac{x\,\left(2-\frac{x^2+1}{x^2}\right)}{x^2+1}+\frac{2\,x}{x^2+1}+\frac{\left(-2\right)\,x\,\left(x^2-1\right)}{\left(x^2+1\right)^2}$$

```
(%i9) ratsimp(%);
```

(%o9) $$dg\left(x\right):=\frac{x^4+4\,x^2-1}{x^5+2\,x^3+x}$$

```
(\%i10) g(sqrt(sqrt(5)-2));
```

(%o10) $$log\left(\frac{\sqrt{5}-1}{\sqrt{\sqrt{5}-2}}\right)+\frac{\sqrt{5}-3}{\sqrt{5}-1}$$

```
(%i11) ratsimp(%);
```

(%o11) $$\frac{\left(\sqrt{5}-1\right)\,log\left(\frac{\sqrt{5}-1}{\sqrt{\sqrt{5}-2}}\right)+\sqrt{5}-3}{\sqrt{5}-1}$$

```
(%i12) bfloat(%);
```

(%o12) $$3.157191043398522b-1$$

Les nombres de métal

```
> u2:=solve(x^2=x+1,x);
```

$$u2:=\frac{1}{2}\,\sqrt{5}+\frac{1}{2},\,-\frac{1}{2}\,\sqrt{5}+\frac{1}{2}$$

```
> u3:=solve(x^3=x^2+x+1,x);
```

$$u3:=\frac{1}{3}\,\%1+\frac{4}{3}\,\frac{1}{\left(19+3\,\sqrt{33}\right)^{(1/3)}}+\frac{1}{3},$$
$$-\frac{1}{6}\,\%1-\frac{2}{3}\,\frac{1}{\left(19+3\,\sqrt{33}\right)^{(1/3)}}+\frac{1}{3}+\frac{1}{2}\,I\,\sqrt{3}\,(\frac{1}{3}\,\%1-\frac{4}{3}\,\frac{1}{\left(19+3\,\sqrt{33}\right)^{(1/3)}}),$$
$$-\frac{1}{6}\,\%1-\frac{2}{3}\,\frac{1}{\left(19+3\,\sqrt{33}\right)^{(1/3)}}+\frac{1}{3}-\frac{1}{2}\,I\,\sqrt{3}\,(\frac{1}{3}\,\%1-\frac{4}{3}\,\frac{1}{\left(19+3\,\sqrt{33}\right)^{(1/3)}})$$
$$\%1:=\left(19+3\,\sqrt{33}\right)^{(1/3)}$$

```
> sol3:=select(t->is(t,real),[u3]);
```

$$sol3 := [\frac{1}{3}\,(19+3\,\sqrt{33})^{(1/3)} + \frac{4}{3}\,\frac{1}{(19+3\,\sqrt{33})^{(1/3)}} + \frac{1}{3}]$$

```
> evalf(sol3);
```

$$[1.839286755]$$

```
> u4:=solve(x^4=x^3+x^2+x+1,x);
```

$$u4 := \mathrm{RootOf}(\%1,\ index = 1),\ \mathrm{RootOf}(\%1,\ index = 2),\ \mathrm{RootOf}(\%1,\ index = 3),\ \mathrm{RootOf}(\%1,\ index = 4)$$

$$\%1 := \_Z^4 - \_Z^3 - \_Z^2 - \_Z - 1$$

```
> u4:=fsolve(x^4=x^3+x^2+x+1,x);
```

$$u4 := -0.7748041132,\ 1.927561975$$

```
> simplify(x^n-sum(x^k,k=0..n-1));
```

$$-\frac{-x^{(n+1)} + 2\,x^n - 1}{x-1}$$

```
> eq:=x^(n+1)-2*x^n-1=0;
```

$$eq := x^{(n+1)} - 2\,x^n - 1 = 0$$

```
> fe:=x->x^(n+1)-2*x^n-1;
```

$$fe := x \to x^{(n+1)} - 2\,x^n - 1$$

```
> dfe:=x->D(fe)(x);
```

$$dfe := x \to \mathrm{D}(fe)(x)$$

```
> solve(dfe(x)=0,x);
```

$$2\,\frac{n}{n+1}$$

```
(\%i1) u2:solve(x^2=x+1,x);
```

(%o1)
$$[x = -\frac{\sqrt{5}-1}{2}, x = \frac{\sqrt{5}+1}{2}]$$

```
(%i2) u3:solve(x^3=x^2+x+1,x);
```

(%o2)
$$[x = \frac{4\,\left(\frac{\sqrt{3}\,i}{2} - \frac{1}{2}\right)}{9\left(3^{-\frac{3}{2}}\,\sqrt{11} + \frac{19}{27}\right)^{\frac{1}{3}}} + \left(3^{-\frac{3}{2}}\,\sqrt{11} + \frac{19}{27}\right)^{\frac{1}{3}}\,\left(-\frac{\sqrt{3}\,i}{2} - \frac{1}{2}\right) + \frac{1}{3}, x = \left(3^{-\frac{3}{2}}\,\sqrt{11} + \frac{19}{27}\right)^{\frac{1}{3}}\,\left(\frac{\sqrt{3}\,i}{2} - \frac{1}{2}\right) + \frac{4\,\left(-\right.}{9\left(3^{-\frac{3}{2}}\right.}$$

```
(%i3) float(realroots(x^3=x^2+x+1));
```

(%o3)
$$[x = 1.839286774396896]$$

```
(%i4) u4:solve(x^4=x^3+x^2+x+1,x);
```

(%o4)

$$[x = -(2^{-\frac{3}{2}}\, sqrt(\left(18\left(\frac{3^{-\frac{3}{2}}\sqrt{563}}{2}-\frac{65}{54}\right)^{\frac{2}{3}}-33\left(\frac{3^{-\frac{3}{2}}\sqrt{563}}{2}-\frac{65}{54}\right)^{\frac{1}{3}}-28\right)\sqrt{36\left(\frac{3^{-\frac{3}{2}}\sqrt{563}}{2}-\frac{65}{54}\right)^{\frac{2}{3}}+33\left(\frac{3^{-\frac{3}{2}}}{2}\right.}$$

```
(%i5) float(realroots(x^4=x^3+x^2+x+1));
```

(%o5) $$[x = -0.77480408549309, x = 1.927561968564987]$$

```
(%i6) sum(x^k,k,0,n-1);
```

(%o6) $$\sum_{k=0}^{n-1} x^k$$

```
(%i8) fe(x):=x^(n+1)-2*x^n-1;
```

(%o8) $$fe\,(x) := x^{n+1} - 2\,x^n - 1$$

```
(%i9) diff(fe(x),x);
```

(%o9) $$(n+1)\;x^n - 2\,n\,x^{n-1}$$

```
(%i10) factor(%);
```

(%o10) $$x^{n-1}\;(n\,x + x - 2\,n)$$

```
(%i11) solve(n*x+x-2*n=0,x);
```

(%o11) $$[x = \frac{2\,n}{n+1}]$$

L'intervention de l'utilisateur est souvent nécessaire dans cette session de travail ; une leçon à retenir, le logiciel de calcul formel ne fournit une aide qu'à celui qui est capable d'interpréter les résultats donnés.

Série

```
> s:=n->sum(1/16^i*(4/(8*i+1)-2/(8*i+4)-1/(8*i+5)-1/(8*i+6)),i=0..n);
```

$$s := n \to \sum_{i=0}^{n} \frac{\frac{4}{8\,i+1} - \frac{2}{8\,i+4} - \frac{1}{8\,i+5} - \frac{1}{8\,i+6}}{16^i}$$

```
> suite:=seq(s(n),n=0..15):
> evalf(suite,20);
```

$$3.1333333333333333333,\ 3.1414224664224664225,\ 3.1415873903465815231,$$
$$3.1415924575674353818,\ 3.1415926454603363196,\ 3.1415926532280875347,$$
$$3.1415926535728808278,\ 3.1415926535889727049,\ 3.1415926535897522752,$$
$$3.1415926535897911464,\ 3.1415926535897931296,\ 3.1415926535897932327,$$
$$3.1415926535897932382,\ 3.1415926535897932384,\ 3.1415926535897932385,$$
$$3.1415926535897932385$$

```
> l:=limit(s(n),n=infinity):
> evalf(l-Pi,100);
```

$$0.$$

```
(\%i1) S(n):=sum(1/16^i*(4/(8*i+1)-2/(8*i+4)-1/(8*i+6)-1/(8*i+5)),i,0,n);
```

$$(\%o1) \qquad S(n) := sum\left(\frac{1}{16^i}\left(\frac{4}{8\,i+1} - \frac{2}{8\,i+4} + \frac{-1}{8\,i+6} + \frac{-1}{8\,i+5}\right), i, 0, n\right)$$

```
(%i2) for i from 0 thru 15 do display(float(S(i)));
```

$$float\left(\frac{47}{15}\right) = 3.133333333333333 float\left(\frac{102913}{32760}\right) = 3.141422466422466 float\left(\frac{615863723}{196035840}\right) = 3.141587390346$$

Chapitre 2

Géométrie analytique

2.1 Bibliothèque de géométrie analytique

Le but de ce chapitre est de définir les fonctions élémentaires de géométrie analytique et les utiliser pour démontrer le théorème d'Euler :

Dans un triangle l'orthocentre, le centre de gravité et le centre du cercle circonscrit sont alignés.

Les points du plan seront représentés par leurs affixes complexes.

D'une manière générale, un nombre complexe est représenté de la manière suivante :

$a_1 + I * a_2$ avec a_1 et a_2 des réels. On utilisera la fonction "assume" pour imposer à des variables l'appartenance à R :

>assume(a_1,real,a_2, real) ;$z := a_1 + I * a_2$

Les fonctions de base des nombres complexes sont :

conjugate(z) qui rend le complexe conjugué du complexe z, **Re(z)** qui rend sa partie réelle, **Im(z)** sa partie imaginaire, **evalc(expression)** qui évalue des expressions complexes pour les rendre sous la forme $x + iy$, **polar**(ρ, θ) qui rend le nombre complexe de module ρ et d'argument θ, **abs(z)** qui rend le module du nombre complexe z.

Définir les fonctions suivantes :

1. milieu qui à deux complexes a et b affixes de deux points A et B fait correspondre l'affixe du milieu de $[AB]$.
2. droite_a_b qui rend l'équation d'une droite passant par deux points.
3. droite_a_v qui rend l'équation d'une droite passant par un point de vecteur directeur donné.
4. inter qui rend le point d'intersection de deux droites sécantes.[1]

[1] inter :=proc(d1,d2)
local res ;
res :=solve(d1,d2,x,y) ;

5. aligné qui à trois points rend true si les points sont alignés, false sinon.
6. gravité qui rend le centre de gravité d'un triangle donné.
7. orthocentre qui rend l'orthocentre d'un triangle donné.
8. circonscrit qui rend le centre du cercle circonscrit à un triangle donné.

En utilisant ces fonctions, démontrer le théorème d'Euler pour un triangle quelconque.

2.2 Eléments de correction

```
> milieu:=(a,b)->(a+b)/2:
> droite_a_b:=(a,b)->expand((x-Re(a))*(Im(b)-Im(a))-(y-Im(a))*(Re(b)-Re
> (a))=0):
> droite_a_v:=(a,v)->expand((x-Re(a))*Im(v)-(y-Im(a))*Re(v)=0):
> isstraight:=(a,b,c)->evalb(argument((b-a)/(b-c)) = 0 or
> argument((b-a)/(b-c)) = Pi):
> mediatrice:=(a,b)->droite_a_v(milieu(a,b),I*(b-a)):
> inter:=proc(d1,d2)
> local res;
> res:=solve({d1,d2},{x,y});
> if op(1,op(1,res))=x then return op(2,op(1,res))+I*op(2,op(2,res));
> else if op(1,op(1,res))=y then return
> op(2,op(2,res))+I*op(2,op(1,res)); else return false fi;
> fi;
> end:
> gravite:=(a,b,c)->evalc((a+b+c)/3):
> orthocentre:=(a,b,c)->evalc(inter(droite_a_v(a,I*(b-c)),droite_a_v(b,
> I*(a-c)))):
> circonscrit:=(a,b,c)->evalc(inter(mediatrice(a,b),mediatrice(b,c))):
> isstraight(circonscrit(1,0,1+I),gravite(0,1,1+I),orthocentre(0,1,1+I)
> );
```

$$true$$

```
> droite_a_b(circonscrit(0,1,1+I),gravite(0,1,1+I));
```

$$-\frac{x}{6}+\frac{1}{6}-\frac{y}{6}=0$$

```
> assume(x1,real,x2,real,x3,real,y1,real,y2,real,y3,real):z1:=x1+I*y1:z
> 2:=x2+I*y2:z3:=x3+I*y3:
> g:=gravite(z1,z2,z3):
> h:=orthocentre(z1,z2,z3):
```

---

if op(1,op(1,res))=x then return op(2,op(1,res))+I*op(2,op(2,res)) ;
else if op(1,op(1,res))=y then return op(2,op(2,res))+I*op(2,op(1,res)) ; else return false fi ;
fi ;
end :

```
> o:=circonscrit(z1,z2,z3):
> isstraight(o,g,h);
```

$$true$$

```
> distance:=(a,b)->abs(b-a);
```

$$distance := (a,\, b) \to |b - a|$$

```
> d1:=distance(o,g):
> d2:=distance(g,h):
> normal(d2/d1);
```

$$2$$

```
(\%i1) droite_a_b(a,b):=expand((x-realpart(a))*(imagpart(b)-imagpart(a))-(y-imagpart(a))*(rea
  -realpart(a))=0)$(%i2) milieu(a,b):=(a+b)/2;
```

(%o2) $$milieu\,(a,b) := \frac{a+b}{2}$$

```
(%i3) droite_a_v(a,v):=expand((x-realpart(a))*imagpart(v)-(y-imagpart(a))*realpart(v)=0);
```

(%o3)
$$droite\_a\_v\,(a,v) := expand\,((x - realpart\,(a))\ imagpart\,(v) - (y - imagpart\,(a))\ realpart\,(v) = 0)$$

```
(%i4) isstraight(a,b,c):=(if second(solve(first(droite_a_b(a,b)),y)[1])-second(solve(first(dr
  then true else false);
```

(%o4)
$$isstraight\,(a,b,c) := if\, second\,((solve\,(first\,(droite\_a\_b\,(a,b))\,,y))_1) - second\,((solve\,(first\,(droite\_a\_b\,(b,c)$$

```
(%i5) mediatrice(a,b):=droite_a_v(milieu(a,b),%i*(b-a));
```

(%o5) $$mediatrice\,(a,b) := droite\_a\_v\,(milieu\,(a,b)\,,i\ (b-a))$$

```
(%i6) inter(d1,d2):=block(res,res:linsolve([d1,d2],[x,y]),
second(res[1])+%i*second(res[2]))$(%i7) orthocentre(a,b,c):=ratsimp(inter(droite_a_v(a,%i*(b-
```

(%o7) $$orthocentre\,(a,b,c) := ratsimp\,(inter\,(droite\_a\_v\,(a,i\ (b-c))\,,droite\_a\_v\,(b,i\ (a-c))))$$

```
(%i8) gravite(a,b,c):=(a+b+c)/3;
```

(%o8) $$gravite\,(a,b,c) := \frac{a+b+c}{3}$$

```
(%i9) circonscrit(a,b,c):=ratsimp(inter(mediatrice(a,b),mediatrice(b,c)));
```

(%o9) $circonscrit\left(a,b,c\right):=ratsimp\left(inter\left(mediatrice\left(a,b\right),mediatrice\left(b,c\right)\right)\right)$

```
(%i10) declare(x1,real,y1,real,x2,real,y2,real,x3,real,y3,real);
```

(%o10) *done*

```
(%i11) z1:x1+y1*%i;
```

(%o11) $i\,y1+x1$

```
(%i12) z2:x2+y2*%i;
```

(%o12) $i\,y2+x2$

```
(%i13) z3:x3+y3*%i;
```

(%o13) $i\,y3+x3$

```
(%i14) g:gravite(z1,z2,z3);
```

(%o14) $$\frac{i\,y3+i\,y2+i\,y1+x3+x2+x1}{3}$$

```
(%i15) h:orthocentre(z1,z2,z3);
```

(%o15)
$$((y2-y1)\ y3^2+\left(-y2^2+(i\,x2-i\,x3)\ y2+y1^2+(i\,x3-i\,x1)\ y1+(x2-x1)\ x3\right)\ y3+y1\,y2^2+\left(-y1^2+(i\,x1-$$

```
(%i16) o:circonscrit(z1,z2,z3);
```

(%o16)
$$-((y2-y1-i\,x2+i\,x1)\ y3^2+\left(-y2^2+y1^2-x2^2+x1^2\right)\ y3+(y1+i\,x3-i\,x1)\ y2^2+\left(-y1^2+x3^2-x1^2\right)\ y2+$$

```
(%i17) isstraight(o,g,h);
```

(%o17) *true*

Chapitre 3

Equations différentielles et logiciel de calcul formel

Le but de ce chapitre est d'une part d'explorer les capacités de Maple et de Maxima à résoudre des équations différentielles et d'autre part à programmer la méthode d'Euler pour représenter graphiquement une solution approchée d'une équation différentielle du premier ordre.

3.1 Maple, Maxima et les équations différentielles

3.1.1 Maple

La fonction dsolve résouds un bon nombre d'équations différentielles ordinaires (ODE en anglais).
La syntaxe de cette fonction est la suivante :
dsolve(ODE) : La syntaxe de base pour résoudre une équation différentielle ; par exemple :
> rep :=dsolve(diff(y(x),x,x)+diff(y(x),x)+y(x)=x) ;

$$y\left(x\right) = e^{-1/2\,x}\sin\left(1/2\,\sqrt{3}x\right)\_C2 + e^{-1/2\,x}\cos\left(1/2\,\sqrt{3}x\right)\_C1 - 1 + x$$

Notez bien la façon dont Maple écrit la constante.
On peut représenter des solutions de la manière suivante :
> s :={seq(seq(op(2,rep),_C2=-5..5),_C1=-5..5)} :plot(s,x=-5..5) ;

3.1.2 Maxima

La syntaxe n'est pas tout à fait identique et on utilisera la fonction *ode2* de la manière suivante :

```
(\%i1) ode2('diff(y,x)-y=k,y,x);
```

(%o1) $$y = (\%c - k\,e^{-x})\;e^{x}$$

Notez bien le ' devant diff empéchant le logiciel d'évaluer la dérivée...

Testez par vous même les résolutions des équations différentielles suivantes en n'oubliant pas de vérifier, à la main la réponse du logiciel !

* $y'' - y' - y = e^x$
* $y'' + 2y' + y = x^2$
* $2y' + 3y = x$

dsolve(ODE, ICs) :

Résolution de l'équation différentielle avec des conditions initiales (ICs) ; par exemple :

```
> rep :=dsolve({diff(y(x),x,x)+diff(y(x),x)+y(x)=x,y(0)=0,D(y)(0)=1}) ;
```

$$rep := y(x) = 1/3\, e^{-1/2\, x} \sin\left(1/2\, \sqrt{3} x\right) \sqrt{3} + e^{-1/2\, x} \cos\left(1/2\, \sqrt{3} x\right) - 1 + x$$

```
> plot(op(2,rep),x=-1..1) ;
```

Essayez vous avec les équations suivantes :

* $(1 + x)y' + y = 1$ et $y(0) = 0$
* $y'' + y' + y = 1$ et $y(0) = 0$ et $y'(0) = 0$
* $2y' + 3y = sin(x)$ et $y'(0) = 0$

Méthode d'Euler

On peut interpréter géométriquement une équation différentielle du premier ordre pour l'intégrer graphiquement.

Une équation du premier ordre peut s'écrire formellement $F(x, y, y') = 0$. Dans le cas général, on pourra écrire : $y' = f(x, y)$.

Par conséquent, pour chaque point M_0 de coordonnées (x_0, y_0) il passe une courbe intégrale et $y'_0 = y'(x_0)$ est la pente de la tangente en ce point de la courbe.

Sur cette tangente, déplaçons nous de h, ce qui donne un point M_1 de coordonnées x_1 et y_1 et en ce point : $y'_1 = f(x_1, y_1)$ etc.

Ecrire une procédure qui a x_0, y_0, f, h et $xmax$ fait correspondre la liste des coordonées des points successifs obtenus grace à la méthode d'Euler décrite ci-dessus.

Testez votre procédure sur les équations différentielles suivantes :

* $(1 + x)y' + y = 1$ et $y(0) = 0$
* $y' + y\cos(x) = 0$ et $y(0) = 2$
* $(x^2 + y^2) - xyy' = 0$ et $y(1) = 1$

Représentez graphiquement ces solutions.

Application : la loi de la chute des corps

hypothèse : la résistance de l'air est proportionnelle à la vitesse v

Appelons R la force retardatrice : $R = Kv$. Si P est le poids du corps, la formule fondamentale de la dynamique donne , en appelant γ l'accélération :

$$\sum F = P - Kv = m\gamma$$

$$mg - Kv = m\frac{dv}{dt}$$

Résoudre cette équation en supposant qu'à l'instant $t = 0$, $v = 0$.

Donner la loi du mouvement avec $x = 0$ pour $t = 0$.

Vers quoi tend le mouvement quand t devient grand.

La resistance de l'air est proportionnelle au carré de la vitesse

Poser le problème et le résoudre !

3.2 Eléments de correction

```
> restart:
> eq1:=diff(y(x),x,x)-diff(y(x),x)-y(x)=exp(x):
> dsolve(eq1);
```

$$\mathrm{y}(x) = e^{(\frac{(\sqrt{5}+1)\,x}{2})}\,\_C2 + e^{(-\frac{(\sqrt{5}-1)\,x}{2})}\,\_C1 - e^{x}$$

```
> eq2:=diff(y(x),x,x)+2*diff(y(x),x)+y(x)=x^2:
> dsolve(eq2);
```

$$\mathrm{y}(x) = e^{(-x)}\,\_C2 + e^{(-x)}\,x\,\_C1 + 6 - 4\,x + x^2$$

```
> eq3:=2*diff(y(x),x)+3*y(x)=x:
> dsolve(eq3);
```

$$\mathrm{y}(x) = \frac{x}{3} - \frac{2}{9} + e^{(-\frac{3\,x}{2})}\,\_C1$$

```
> eq4:=(1+x)*diff(y(x),x)+y(x)=1:
> dsolve({eq4,y(0)=0});
```

$$\mathrm{y}(x) = \frac{x}{1+x}$$

```
> eq5:=diff(y(x),x,x)+diff(y(x),x)+y(x)=1:
> dsolve({eq5,y(0)=0,D(y)(0)=0});
```

$$\mathrm{y}(x) = -\frac{1}{3}\,e^{(-\frac{x}{2})}\sin(\frac{\sqrt{3}\,x}{2})\,\sqrt{3} - e^{(-\frac{x}{2})}\cos(\frac{\sqrt{3}\,x}{2}) + 1$$

```
> eq6:=2*diff(y(x),x)+3*y(x)=sin(x):
> dsolve({eq6,D(y)(0)=0});
```

$$\mathrm{y}(x) = -\frac{2}{13}\cos(x) + \frac{3}{13}\sin(x) + \frac{2}{13}\,e^{(-\frac{3\,x}{2})}$$

```
> methodedeuler:=proc(x0,y0,f,pas,xmax)
> local l,i,x1,y1;
> l:=[[x0,y0]];x1:=x0;y1:=y0;
> for i from 1 to floor((xmax-x1)/pas) do
> y1:=f(x1,y1)*pas+y1;
> x1:=x1+pas;
> l:=[op(l),[x1,y1]];
> od;
> l;
> end:
> sol1:=methodedeuler(0,0,(x,y)->(1-y)/(1+x),0.01,2):sol2:=op(2,dsolve(
> {diff(y(x),x)=(1-y(x))/(1+x),y(0)=0})):plot({sol1,sol2},x=0..2):
> eqdyn:=m*g-K*v(t)=m*diff(v(t),t):
> sol:=dsolve({eqdyn,v(0)=0});
```

$$sol := \mathrm{v}(t) = \frac{g\,m}{K} - \frac{e^{(-\frac{K\,t}{m})}\,g\,m}{K}$$

```
> int(op(2,sol),t);
```

$$\frac{g\,m\,t}{K} + \frac{g\,m^2\,e^{(-\frac{K\,t}{m})}}{K^2}$$

```
> mvt:=t->g/K*m*t+g/K^2*m^2*exp(-K/m*t)-g*m^2/K^2;
```

$$mvt := t \to \frac{g\,m\,t}{K} + \frac{g\,m^2\,e^{(-\frac{K\,t}{m})}}{K^2} - \frac{g\,m^2}{K^2}$$

```
> simplify(mvt(t));
```

$$\frac{m\,g\,(t\,K + m\,e^{(-\frac{K\,t}{m})} - m)}{K^2}$$

```
> P/K*t-P*m/K^2*(1-exp(-K/m*t));
```

$$\frac{P\,t}{K} - \frac{P\,m\,(1 - e^{(-\frac{K\,t}{m})})}{K^2}$$

Le mouvement tend à devenir un mouvement uniforme de vitesse P/K.

```
> restart:eqdyn2:=m*g-K*(v(t))^2=m*diff(v(t),t):
> rep2:=dsolve({eqdyn2,v(0)=0});
```

$$rep2 := \mathrm{v}(t) = \frac{\tanh(\frac{\sqrt{m\,g\,K}\,t}{m})\,\sqrt{m\,g\,K}}{K}$$

```
> vv:=t->op(2,rep2):
> x2:=t->m/K*ln(cosh(sqrt(g*K/m)*t)):
```

```
> subs(m=1,g=9.81,K=1,x2(t));
```

$$\ln(\cosh(3.132091953\,t))$$

```
(\%i1) eq1:'diff(y,x,2)-'diff(y,x)-y=exp(x);
```

(%o1)
$$\frac{d^2}{d\,x^2}\,y-\frac{d}{d\,x}\,y-y=e^x$$

```
(%i2) ode2(eq1,y,x);
```

(%o2)
$$y=\%k1\,e^{\frac{(\sqrt{5}+1)\,x}{2}}+\%k2\,e^{\frac{(1-\sqrt{5})\,x}{2}}-e^x$$

```
(%i3) eq2:'diff(y,x,2)+2*'diff(y,x)+y=x^2;
```

(%o3)
$$\frac{d^2}{d\,x^2}\,y+2\,\left(\frac{d}{d\,x}\,y\right)+y=x^2$$

```
(%i4) ode2(eq2,y,x);
```

(%o4)
$$y=(\%k2\,x+\%k1)\,e^{-x}+x^2-4\,x+6$$

```
(%i5) eq3:2*'diff(y,x)+3*y=x;
```

(%o5)
$$2\,\left(\frac{d}{d\,x}\,y\right)+3\,y=x$$

```
(%i6) ode2(eq3,y,x);
```

(%o6)
$$y=e^{-\frac{3\,x}{2}}\,\left(\frac{(6\,x-4)\,e^{\frac{3\,x}{2}}}{18}+\%c\right)$$

```
(%i7) eq4:(1+x)*'diff(y,x)+y=1;
```

(%o7)
$$(x+1)\,\left(\frac{d}{d\,x}\,y\right)+y=1$$

```
(%i8) ode2(eq4,y,x);
```

(%o8)
$$y=\frac{x+\%c}{x+1}$$

```
(%i9) eq5:'diff(y,x,x)+diff(y,x)+y=1;
```

(%o9)
$$\frac{d^x}{d\,x^x}\,y+y=1$$

```
(%i10) eq5:'diff(y,x,2)+'diff(y,x)+y=1;
```

(%o10)
$$\frac{d^2}{d\,x^2}\,y+\frac{d}{d\,x}\,y+y=1$$

```
(%i11) ode2(eq5,y,x);
```

(%o11)
$$y=e^{-\frac{x}{2}}\,\left(\%k1\,sin\left(\frac{\sqrt{3}\,x}{2}\right)+\%k2\,cos\left(\frac{\sqrt{3}\,x}{2}\right)\right)+1$$

```
(%i13) ic2(%o11,x=0,y=0,diff(y,x)=0);
```

(%o13)
$$y=e^{-\frac{x}{2}}\,\left(-\frac{\sqrt{3}\,sin\left(\frac{\sqrt{3}\,x}{2}\right)}{3}-cos\left(\frac{\sqrt{3}\,x}{2}\right)\right)+1$$

```
(%i15) eq6:2*'diff(y,x)+3*y=sin(x);
```

(%o15)
$$2\,\left(\frac{d}{d\,x}\,y\right)+3\,y=sin\left(x\right)$$

```
(%i16) ode2(eq6,y,x);
```

(%o16)
$$y=e^{-\frac{3\,x}{2}}\,\left(\frac{2\,e^{\frac{3\,x}{2}}\,\left(\frac{3\,sin(x)}{2}-cos\left(x\right)\right)}{13}+\%c\right)$$

```
(%i18) ic1(%o16,x=0,diff(y,x)=0);
```

(%o18)
$$y=\frac{e^{-\frac{3\,x}{2}}\,\left(13\,y+3\,e^{\frac{3\,x}{2}}\,sin\left(x\right)-2\,e^{\frac{3\,x}{2}}\,cos\left(x\right)+2\right)}{13}$$

```
(%i19) expand(%);
```

(%o19)
$$y=e^{-\frac{3\,x}{2}}\,y+\frac{3\,sin\left(x\right)}{13}-\frac{2\,cos\left(x\right)}{13}+\frac{2\,e^{-\frac{3\,x}{2}}}{13}$$

```
(%i20) meth_euler(x0,y0,f,pas,xmax):=block([l,i,x1,y1],
l:[[x0,y0]],
x1:x0,
y1:y0,
for i from 1 thru floor((xmax-x1)/pas) do
(y1:f(x1,y1)*pas+y1,
x1:x1+pas,
l:append(l,[[x1,y1]])),

l
)$
```

La suite pourra être traduite de la même façon avec Maxima.

Chapitre 4

Courbes paramétrées

4.1 Syntaxe Maple

Pour représenter graphiquement une courbe plane définie paramétriquement, on écrira :
plot([x(t), y(t), t=tmin..tmax], h, v, options)
Description
La représentation graphique d'une courbe paramétrée sera spécifiée par une liste de trois éléments : les deux premiers sont des fonctions du paramètre et la troisième est l'intervalle d'appartenance du paramètre.

Plusieurs courbes paramétriques peuvent être représentées simultanément en indiquant un ensemble de listes ou une liste de listes.

Exemples :
> plot([sin(t), 2*cos(t), t=0..Pi]) ;
> plot($[(t^2-1)/(t^2+1), 2*t/(t^2+1)$, t=-infinity..infinity]) ;
> plot([[2*cos(s), sin(s), s=0..2*Pi], [cos(t), sin(t), t=0..2*Pi]], color=[blue,yellow]) ;

4.2 Syntaxe Maxima

plot2d([['parametric, x(t), y(t), [t, tmin, tmax], [nticks, n]]], [x,xmin,xmax], [plot_format, *openmath* ou *gnuplot*])

De la même façon, on représentera plusieurs courbes paramétrées simultanément en rajoutant une liste ; par exemple :

plot2d([['parametric, 2*sin(t), sin(t)*cos(t), [t, -6, 6], [nticks, 300]],['parametric,cos(t),sin(t),[t, -6,6],[nticks,300]]],[x,-5,5],[plot_format,openmath]) ;

4.3 Exercices

Trois exercices tirés de la banque PT

1. (a) Tracer la courbe :

$$\begin{cases} x = t^3 - 4t \\ y = 2t^2 - 3 \end{cases}$$

   (b) Calculer l'angle des tangentes au point double.

2. Déterminer les branches infinies et les points doubles de la courbe paramétrée définie par :

$$\begin{cases} x = \frac{t-1}{t^2-4} \\ y = \frac{t^2-3}{t+2} \end{cases}$$

3. Etude et représentation graphique de :

$$\begin{cases} x = \frac{u^3}{u^2-9} \\ y = \frac{u(u-2)}{u-3} \end{cases}$$

4.3.1 Un jeu pour terminer

Soit n un entier compris entre 1 et 1000. les opérations permises sont des divisions, des additions et des soustractions par un chiffre de 1 à 9. En combien d'étapes maximum peut-on atteindre 0 ?

Par exemple il suffit de 4 étapes pour atteindre 0 en partant de 159 :

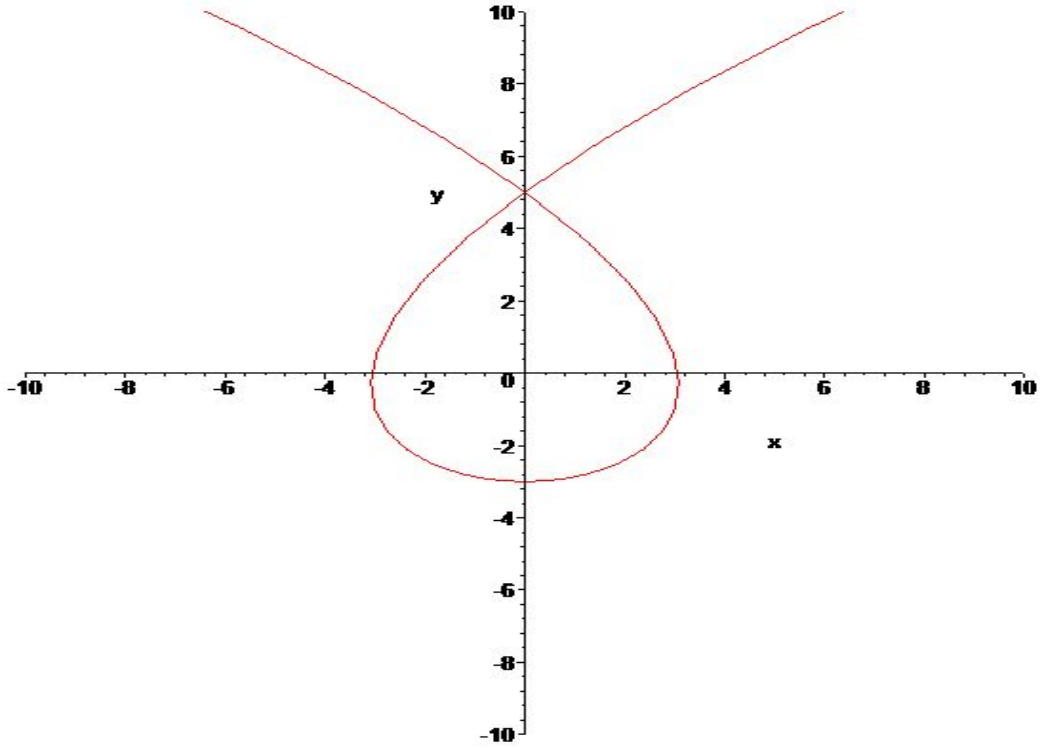
$$159 - 6 = 153 \tag{4.1}$$
$$153/9 = 17 \tag{4.2}$$
$$17 - 8 = 9 \tag{4.3}$$
$$9 - 9 = 0 \tag{4.4}$$

4.4 Eléments de correction

```
> x:=t->t^3-4*t;y:=t->2*t^2-3;
```

$$x := t \to t^3 - 4\,t$$
$$y := t \to 2\,t^2 - 3$$

```
> plot([x(t),y(t),t=-4..4],x=-10..10,y=-10..10);
```




```
> solve({x(u)=x(v),y(u)=y(v),u<>v},{u,v});
```

$$\{v = 2,\ u = -2\},\ \{v = -2,\ u = 2\}$$

```
> xp:=t->D(x)(t):yp:=t->D(y)(t):(xp(2)*xp(-2)+yp(2)*yp(-2))/(sqrt(xp(2)
> ^2+yp(2)^2)*sqrt(xp(-2)^2+yp(-2)^2));
```

$$0$$

```
> restart:
> x:=t->(t-1)/(t^2-4):y:=t->(t^2-3)/(t+2):
> limit(x(t),t=2,right);limit(y(t),t=2,right);
```

$$\infty$$

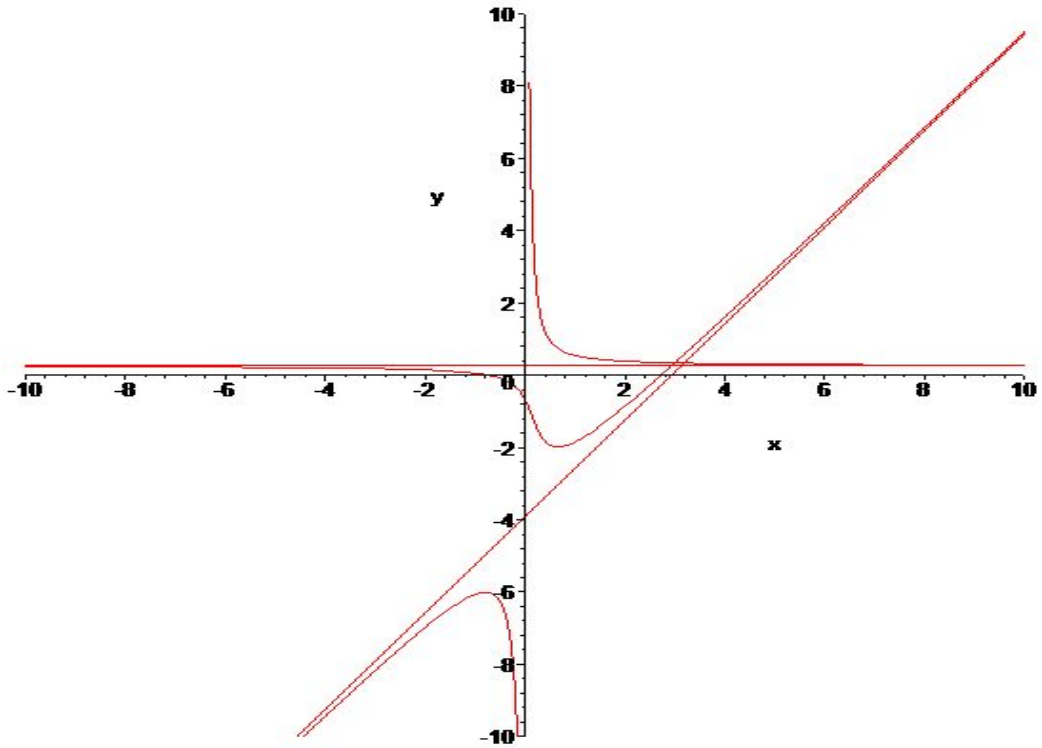
$$\frac{1}{4}$$

```
> limit(x(t),t=2,left);limit(y(t),t=2,left);
```

$$-\infty$$

$$\frac{1}{4}$$

```
> plot([x(t),y(t),t=-10..10],x=-10..10,y=-10..10);
```



La droite d'équation x=0 est asymptote verticale (pour t tendant vers l'infini) .

La droite d'équation y=1/4 est asymptote horizontale (pour t tendant vers 2).

On étudie l'asymptote oblique au voisinage de -2

```
> limit(y(t)/x(t),t=-2);
```

$$\frac{4}{3}$$

```
> limit(y(t)-4/3*x(t),t=-2);
```

$$\frac{-47}{12}$$

```
> s:=solve({x(u)=x(v),y(u)=y(v),u<>v},{u,v});assign(s);
```

$$s := \{u = \frac{1}{3} - \text{RootOf}(3\_Z^2 - \_Z - 11),\ v = \text{RootOf}(3\_Z^2 - \_Z - 11)\}$$

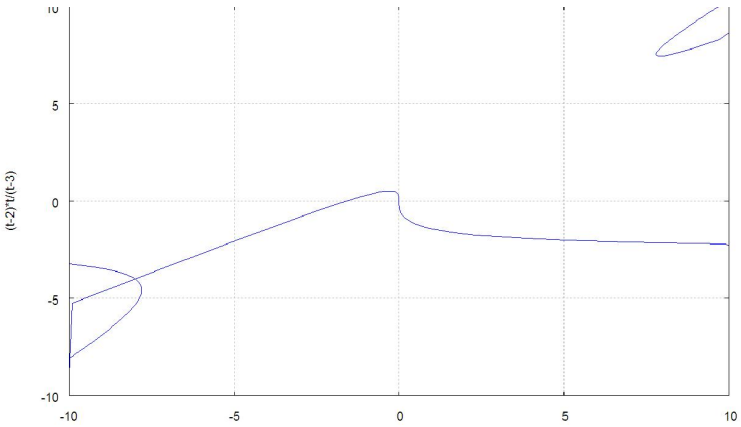
```
> simplify([x(u),y(u)]);
```

$$[3,\ \frac{1}{3}]$$

Donnons la dernière correction en utilisant Maxima :

```
(\%i1) (x(u):=u^3/(u^2-9),
y(u):=(u*(u-2))/(u-3))$

(%i2)plot2d([parametric,x(t),y(t),[t,-10,10]],[x,-10,10],[y,-10,10],[nticks,200],[gnuplot_pre
  grid;"]
  ); $
```



(%t2)

```
(%i3) limit(x(t),t,inf);
```

(%o3) ∞

```
(%i4) limit(y(t),t,inf);
```

(%o4) ∞

```
(%i5) limit(y(t)/x(t),t,inf);
```

(%o5) 1

```
(%i6) limit(y(t)-x(t),t,inf);
```

(%o6) 1

```
(%i7) limit(x(t),t,3,plus);
```

(%o7) ∞

```
(%i8) limit(y(t),t,3,plus);
```

(%o8) ∞

```
(%i9) limit(y(t)/x(t),t,3,plus);
```

(%o9) $\frac{2}{3}$

```
(%i11) limit(y(t)-2*x(t)/3,t,3,plus);
```

(%o11) $\frac{3}{2}$

```
(%i12) limit(x(t),t,-3,plus);
```

(%o12) ∞

```
(%i13) limit(y(t),t,-3,plus);
```

(%o13) $-\frac{5}{2}$

```
(%i14) solve([x(u)=x(v),y(u)=y(v)],[u,v]);
```

(%o14)
$[[u = \%r1, v = \%r1], [u = \frac{3\sqrt{13}+9}{\sqrt{13}+4}, v = -\sqrt{13}-1], [u = \frac{3\sqrt{13}-9}{\sqrt{13}-4}, v = \sqrt{13}-1], [u = 3-\sqrt{3}, v = \frac{3\sqrt{3}-3}{\sqrt{3}}], [u$

```
(%i16) radcan(%);
```

(%o16)
$[[u = \%r1, v = \%r1], [u = \frac{3\sqrt{13}+9}{\sqrt{13}+4}, v = -\sqrt{13}-1], [u = \frac{3\sqrt{13}-9}{\sqrt{13}-4}, v = \sqrt{13}-1], [u = 3-\sqrt{3}, v = \frac{3\sqrt{3}-3}{\sqrt{3}}], [u$

Eléments de solution du jeu en Maple

La procédure qui suit est écrite récursivement (elle s'appelle elle-même, voir prochains TP) mais on pourrait l'écrire dans une boucle...

```
> ramene:=proc(n,compt,l)
> option remember;
> local x,j,aux,k,aux2;
> x:=n;aux:=1;
> if x<=9 then return [compt+1,[op(l),x,"-",x,"=0"]]; fi;
> for j from 5 to 8 do
> if (x mod j=0) then
> aux:=j fi;
> od;
> if  aux>=2 and x mod 9<>0  then
> ramene(x/aux,compt+1,[op(l),x,"/",aux,"=",x/aux]) else
> if x mod 9=0 then ramene(x/9,compt+1,[op(l),x,"/9=",x/9]) else
> ramene(x-(x mod 9),compt+1,[op(l),x,"-",x mod 9,"=",x-(x mod
> 9)])
> fi;fi;end:
```

```
> ramene(990,0,[]);
```

[5, [990, "/9=", 110, 110, "/", 5, "=", 22, 22, "-", 4, "=", 18, 18, "/9=", 2, 2, "-", 2, "=0"]]

```
> ramene_plus_fin:=proc(n)
> local i,k,x;
> x:=n;
> if op(1,ramene(x,0,[]))>4 then
> for k from 1 to 9 do
> if op(1,ramene(x+k,0,[]))<5 then  return
> [op(1,ramene(x+k,0,[]))+1,x,"+",k,"=",x+k,op(2,ramene(x+k,0,[]))]
> fi;
> if op(1,ramene(x-k,0,[]))<5 then  return
> [op(1,ramene(x-k,0,[]))+1,x,"-",k,"=",x-k,op(2,ramene(x-k,0,[]))]
> fi;
> od;
> fi;
> return ramene(x,0,[]);
> end:
```

```
> ramene_plus_fin(829);
```

[5, 829, "+", 3, "=", 832,
[832, "/", 8, "=", 104, 104, "/", 8, "=", 13, 13, "-", 4, "=", 9, 9, "-", 9, "=0"]]

```
> ramene_plus_fin(851);
```

[6, [851, "-", 5, "=", 846, 846, "/9=", 94, 94, "-", 4, "=", 90, 90, "/9=", 10, 10, "/", 5, "=", 2, 2, "-", 2, "=0"]]

```
> ramene_plus_fin(990);
```

[5, [990, "/9=", 110, 110, "/", 5, "=", 22, 22, "-", 4, "=", 18, 18, "/9=", 2, 2, "-", 2, "=0"]]

```
> with(networks):
> Une solution plus générale utilisant la théorie des graphes : on
> construit un graphe de tous les chemins possibles entre n+9 et 0 et on
> cherche le chemin le plus court dans ce graphe.
```

```
> construis:=proc(n)
> local vset,eset,G,i,j;
> vset:={seq(i,i=0..n+9)};
> eset:={};
> G:=graph(vset,eset);
> for i from 0 to n+9 do
> for j from i+1 to n+9 do
> if  j-i<=9 or (i>0 and j mod i=0 and j/i<=9)  then addedge([j,i],G);
> addedge([i,j],G) fi;
> od;
> od;
> G;
> end:
> T:=shortpathtree(construis(829),0):
> path([0,829],T);
```

$$[0, 9, 13, 104, 832, 829]$$

```
> T:=shortpathtree(construis(851),0):path([0,851],T);
> C'est un des deux cas à 6 pas, l'autre étant 853 :
```

$$[0, 9, 15, 120, 840, 844, 851]$$

```
> T:=shortpathtree(construis(853),0):path([0,853],T);
```

$$[0, 9, 15, 120, 840, 844, 853]$$

Chapitre 5

Récursivité

5.1 Procédures récursives

Une procédure est récursive lorsqu'elle s'appelle elle-même. L'exemple le plus simple est le calcul de la factorielle d'un nombre. n étant un nombre naturel, $n!$ est défini comme le produit des entiers inférieurs ou égaux à n.

Mais on peut aussi définir $n!$ comme suit :

$0! = 1$ et $n! = (n-1)! \times n$

Deux programmations sont alors possibles :

Itérative :

Maple

```
factoriel_iteratif :=proc(n)
local i,aux ;
aux :=1 ;
If n=0 then return 1 else
for i from 1 to n do
aux :=aux*i ;
od ;
fi ;
aux ;
end ;
```

Maxima

```
facto(n) :=block(
[i,aux :1],
if n=0 then return(1) else
for i from 1 thru n do
aux :aux*i,
return(aux)
)
```

Récursive :

Maple

```
factoriel_recursif :=proc(n)
if n=0 then 1 else
n*factoriel_recursif(n-1) fi ;
end ;
```

Maxima

```
factorec(n) :=block(
if n=0 then return(1) else
factorec(n-1)*n )
```

Maple dispose d'une possibilité précieuse, c'est l'option remember. Lorsqu'une fonction est définie avec cette option, Maple lui associe une table de souvenance qui contient tous les appels de la fonction et les résultats trouvés. Si la fonction est de nouveau appelée avec les mêmes valeurs des paramètres, le

résultat est directement retourné à partir de la table de souvenance sans que le corps de la fonction soit réexécuté.

Cette option permet d'une part d'accélérer les calculs et d'autre part de permettre des calculs ayant un grand nombre d'appels récursifs.

Essayez, par exemple :

```
> fact1:=proc(n)
> if n=0 then 1 else fact1(n-1)*n fi;
> end:
> fact2:=proc(n)
> option remember;
> if n=0 then 1 else fact2(n-1)*n fi;
> end:
> a:=time():fact1(1800);evalf(time()-a,100);
> a:=time():fact2(1800);evalf(time()-a,100);
> a:=time():fact1(5000);evalf(time()-a,100);
> a:=time():fact2(5000);evalf(time()-a,100);
```

Notez la fonction "time()" qui retourne le temps en seconde utilisé par le microprocesseur durant la session courante ; il suffit donc de comparer la valeur de "time()" avant et après un calcul pour récupérer le temps de ce calcul.

5.2 Suites récurrentes et récursivité

Etude d'un exemple : la (célèbre) suite de Fibonacci définie par :

$$\begin{cases} u_0 &= 1 \\ u_1 &= 1 \\ u_{n+1} &= u_n + u_{n-1} \end{cases}$$

1. Ecrire une procédure itérative permettant de calculer u_n en fonction de n.
2. Ecrire une procédure récursive permettant de calculer u_n en fonction de n.
3. Soit $(v_n)_n$ la suite définie par

$$v_n = \frac{(1+\sqrt{5})^{n+1} - (1-\sqrt{5})^{n+1}}{2^{n+1}\sqrt{5}}$$

   Démontrer que $v = u$

   En déduire une procédure de calcul du n^{ieme} terme de la suite de Fibonacci
4. Comparer le temps de calcul des différentes procédures en utilisant la fonction "time()".

5. Donner une définition par récurrence de la suite

$$w_n = \frac{u_n}{u_{n-1}}$$

Démontrer que cette suite converge.
Ecrire une procédure calculant w_n.

6. Soit la fraction continue définie par :

$$a = 1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{\ddots}}}$$

Ecrire une procédure calculant une valeur approchée de a. Déterminer a.

5.3 Eléments de solution

On donne ici une solution utilisant Maxima

```
(\%i1) fibrec(n):=block(
if n=0 or n=1 then return(1) else fibrec(n-1)+fibrec(n-2)

)$(%i2) fibrec(10);

(%o2)                                    89

(%i3) for i from 1 thru 50 do display(fibrec(i));
```

$fibrec\,(1) = 1 fibrec\,(2) = 2 fibrec\,(3) = 3 fibrec\,(4) = 5 fibrec\,(5) = 8 fibrec\,(6) = 13 fibrec\,(7) = 21 fibrec\,(8) =$

Chapitre 6

Suites

6.1 Suites

Le but de ce TP est de programmer et de prévoir le comportement de différentes suites en utilisant, notamment, le concept de récursivité. On pourra représenter graphiquement ces suites.

6.1.1 Suites de Padovan et de Perrin

Elle est définie par :

$$\begin{cases} u_0 &= 3 \\ u_1 &= 3 \\ u_2 &= 3 \\ u_n &= u(n-2) + u(n-3) \end{cases}$$

1. Programmer cette suite
2. On définit la suite $v_n = \frac{u_n}{u_{n-1}}$; Quel est son comportement à l'infini ?

6.2 Suite de Perrin

C'est la même que la suite de Padovan, mais $u_0 = 3$, $u_1 = 0$ et $u_2 = 2$. On peut démontrer que pour tout nombre premier p, p divise u_p. Ce théorème est utilisé pour tester la non-primalité d'un nombre. par exemple : 18 ne divise pas u_{18} donc 18 n'est pas premier. Mais on ne sait pas si n est premier lorsqu'il divise u_n.

6.2.1 La suite de Syracuse

Soit n un nombre entier positif. S'il est pair, on le divise par 2 ; s'il est impair on le remplace par $3n + 1$. Que devient la suite ?

On appelle hauteur le nombre d'étapes nécessaires pour atteindre 1. Dessinez la hauteur des nombres naturels pour n allant de 2 à 1000.

Ce problème est un problème ouvert en mathématiques.

6.2.2 Suites chaotiques

Soit u_k la suite définie par :

$$\begin{cases} u_0 &= \frac{1}{10} \\ u_{n+1} &= k.u_n.(1-u_n) \end{cases}$$

On étudie le comportement de cette suite pour des valeurs de k comprises entre 0 et 4.

Représentez graphiquement le comportement de la suite en fonction de k en utilisant l'algorithme suivant :

Pour k allant de 0 à 4 faire
Calculer les 50 premiers termes de la suite.
Représenter les points de coordonnées (p, u_p) pour p allant de 51 à 100

Interprétez ce dessin !

6.3 Eléments de solution

```
> restart:with(plots):
```

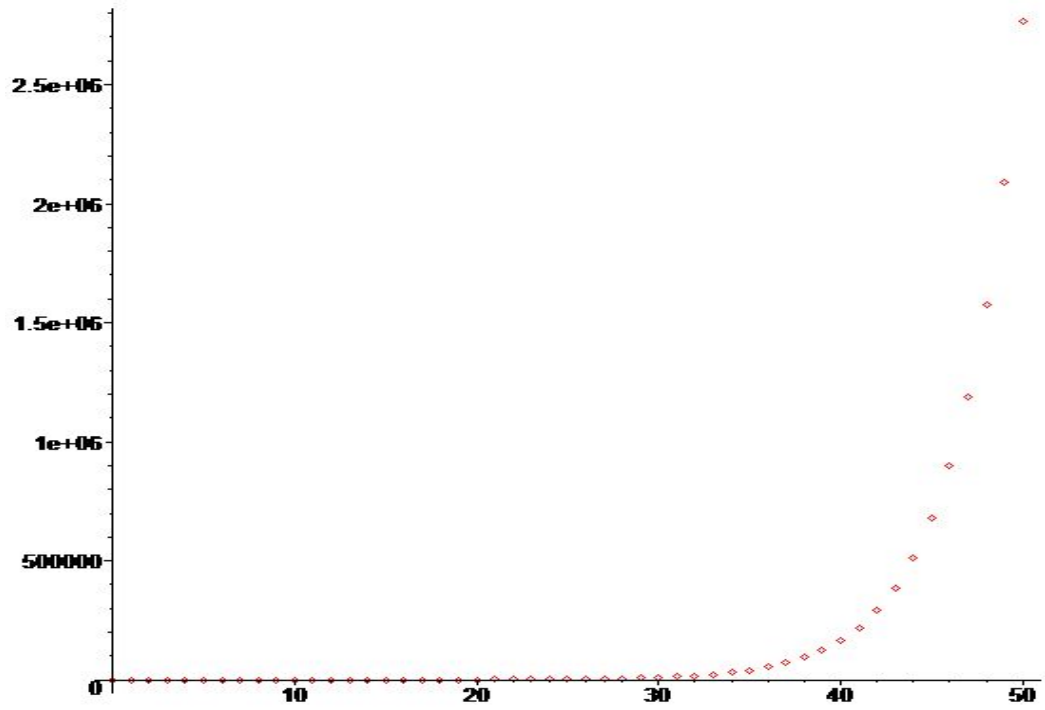
```
Warning, the name changecoords has been redefined
```

```
> padovian:=proc(n::integer)
> option remember;
> if n<3 then 3 else padovian(n-2)+padovian(n-3) fi;
> end:
```

```
> a:=[seq([i,padovian(i)],i=0..50)];
```

$a :=$ [[0, 3], [1, 3], [2, 3], [3, 6], [4, 6], [5, 9], [6, 12], [7, 15], [8, 21], [9, 27], [10, 36], [11, 48], [12, 63], [13, 84], [14, 111], [15, 147], [16, 195], [17, 258], [18, 342], [19, 453], [20, 600], [21, 795], [22, 1053], [23, 1395], [24, 1848], [25, 2448], [26, 3243], [27, 4296], [28, 5691], [29, 7539], [30, 9987], [31, 13230], [32, 17526], [33, 23217], [34, 30756], [35, 40743], [36, 53973], [37, 71499], [38, 94716], [39, 125472], [40, 166215], [41, 220188], [42, 291687], [43, 386403], [44, 511875], [45, 678090], [46, 898278], [47, 1189965], [48, 1576368], [49, 2088243], [50, 2766333]]

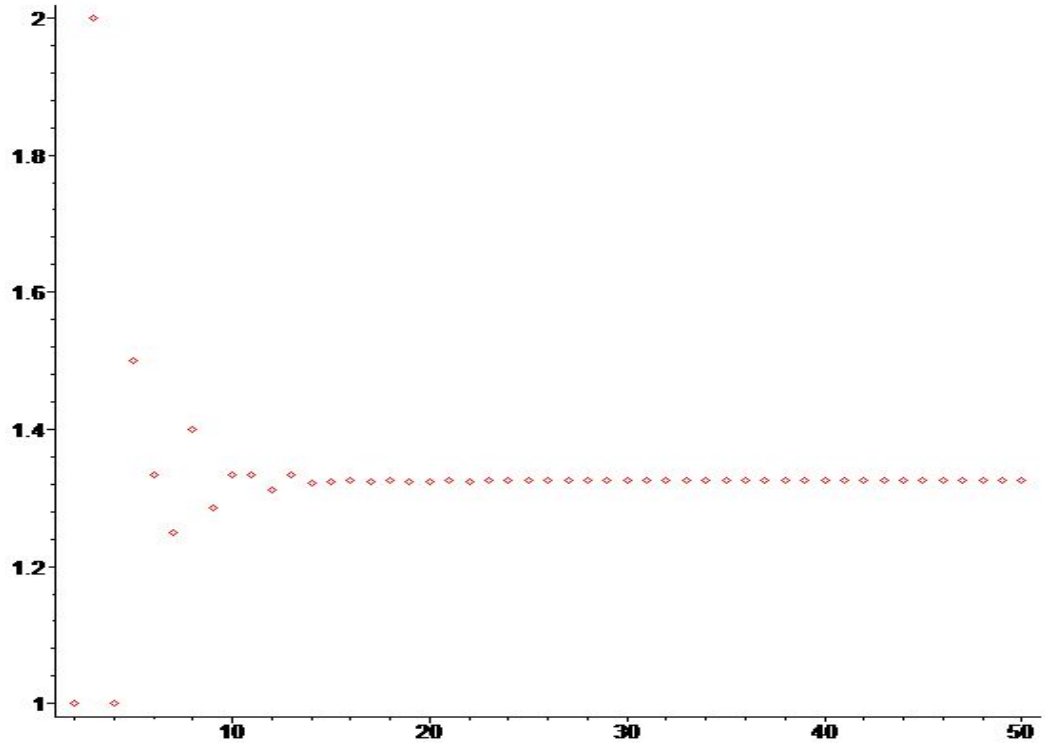
```
> plot(a,style=POINT);
```




```
> rap:=n->padovian(n)/padovian(n-1):
> a:=[seq([i,rap(i)],i=2..50)]:evalf(a);
```

[[2., 1.], [3., 2.], [4., 1.], [5., 1.500000000], [6., 1.333333333], [7., 1.250000000], [8., 1.400000000], [9., 1.285714286], [10., 1.333333333], [11., 1.333333333], [12., 1.312500000], [13., 1.333333333], [14., 1.321428571], [15., 1.324324324], [16., 1.326530612], [17., 1.323076923], [18., 1.325581395], [19., 1.324561404], [20., 1.324503311], [21., 1.325000000], [22., 1.324528302], [23., 1.324786325], [24., 1.324731183], [25., 1.324675325], [26., 1.324754902], [27., 1.324699352], [28., 1.324720670], [29., 1.324723247], [30., 1.324711500], [31., 1.324722139], [32., 1.324716553], [33., 1.324717562], [34., 1.324718956], [35., 1.324717128], [36., 1.324718357], [37., 1.324717915], [38., 1.324717828], [39., 1.324718105], [40., 1.324717865], [41., 1.324717986], [42., 1.324717968], [43., 1.324717934], [44., 1.324717976], [45., 1.324717949], [46., 1.324717958], [47., 1.324717960], [48., 1.324717954], [49., 1.324717959], [50., 1.324717957]]

```
> plot(a,style=POINT);
```




```
> evalf(solve(x^3=x+1,x));
```

$$1.324717958,\ -0.6623589786 + 0.5622795125\,I,\ -0.6623589786 - 0.5622795125\,I$$

```
> restart:
> f:=proc(n::integer)
> if n mod 2 =0 then n/2 else 3*n+1 fi;
> end:
> syracuse:=proc(l,n)
> option remember;
> if n=1 then l else syracuse([op(l),f(n)],f(n)) fi;
> end:

> syracuse([10],10);
```

$$[10,\ 5,\ 16,\ 8,\ 4,\ 2,\ 1]$$

```
> syr:=proc(n)
> syracuse([n],n);
> end:

> syr(10);
```

$$[10,\ 5,\ 16,\ 8,\ 4,\ 2,\ 1]$$

```
> hauteur:=proc(l)
> local i,max;
> max:=op(1,l);
> for i from 2 to nops(l) do
> if op(i,l)>max then max:=op(i,l) fi;
> od;
> max;
> end;
```

$$hauteur := \mathbf{proc}(l)$$
$$\mathbf{local}\, i,\ max;$$
$$max := \mathrm{op}(1,\ l)\,;$$
$$\mathbf{for}\, i\, \mathbf{from}\, 2\, \mathbf{to}\, \mathrm{nops}(l)\, \mathbf{do}\, \mathbf{if}\, max < \mathrm{op}(i,\ l)\, \mathbf{then}\, max := \mathrm{op}(i,\ l)\, \mathbf{end\ if}\, \mathbf{end\ do}\,;$$
$$max$$
$$\mathbf{end\ proc}$$

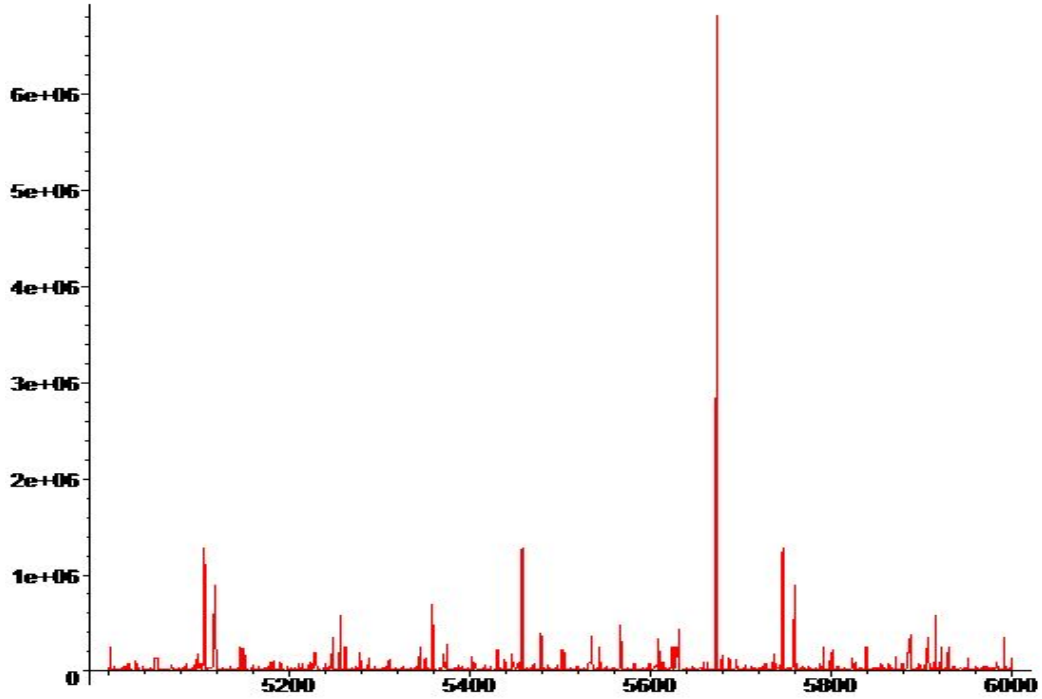
```
> hauteur(syr(5000));syr(5000);
```

$$5000$$

[5000, 2500, 1250, 625, 1876, 938, 469, 1408, 704, 352, 176, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]

```
> a:=[seq([i,hauteur(syr(i))],i=5000..6000)]:
```

```
> plot(a);
```




```
> ([seq([i,hauteur(syr(i))],i=5650..5690)]);
```

[[5650, 14308], [5651, 25432], [5652, 9232], [5653, 16960], [5654, 39364], [5655, 39364], [5656, 9232], [5657, 39364], [5658, 9232], [5659, 96712], [5660, 9556], [5661, 16984], [5662, 28672], [5663, 86020], [5664, 5664], [5665, 16996], [5666, 9232], [5667, 25504], [5668, 8080], [5669, 17008], [5670, 12760], [5671, 43072], [5672, 5672], [5673, 6810136], [5674, 8512], [5675, 25540], [5676, 9232], [5677, 17032], [5678, 19168], [5679, 163780], [5680, 5680], [5681, 17044], [5682, 12148], [5683, 25576], [5684, 5684], [5685, 17056], [5686, 138400], [5687, 138400], [5688, 5688], [5689, 19204], [5690, 8536]]

```
> ifactor(5673);
```

$$(3)\ (31)\ (61)$$

```
> hauteur(syr(4096));
```

$$4096$$

```
> base:=proc(n,l)
> option remember;
> if iquo(n,2)=0 then [n mod 2,op(l)] else base(iquo(n,2),[n mod
> 2,op(l)]) fi;
> end:
```

```
> base2:=proc(n)
> base(n,[]);
> end:
```

```
> base2(5673);
```

$$[1,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 1,\ 0,\ 1,\ 0,\ 0,\ 1]$$

```
> restart:with(plots):
```

```
> chaos:=proc(n,a)
> option remember;
> if n=0 then a else chaos(n-1,a)^2-a fi;
> end;
```

$$\begin{aligned} &chaos := \mathbf{proc}(n,\ a)\\ &\mathbf{option}\ remember;\\ &\quad \mathbf{if}\, n = 0\, \mathbf{then}\, a\ \ \mathbf{else}\, \mathrm{chaos}(n-1,\ a)^2 - a\, \mathbf{end\ if}\\ &\mathbf{end\ proc} \end{aligned}$$

```
> seq(chaos(i,0.9),i=0..20);
```

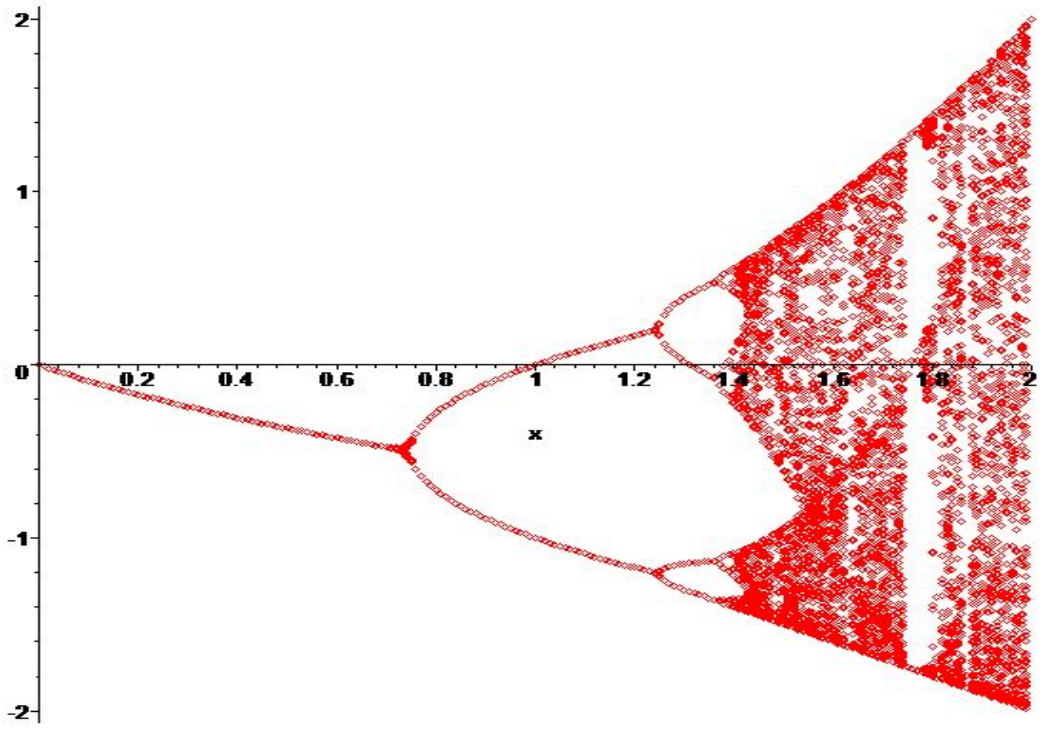
$$\begin{aligned} &0.9,\ -0.09,\ -0.8919,\ -0.10451439,\ -0.8890767423,\ -0.1095425463,\ -0.8880004306,\\ &-0.1114552353,\ -0.8875777305,\ -0.1122057723,\ -0.8874098647,\ -0.1125037320,\\ &-0.8873429103,\ -0.1126225595,\ -0.8873161591,\ -0.1126700338,\ -0.8873054635,\\ &-0.1126890144,\ -0.8873011860,\ -0.1126966053,\ -0.8872994752 \end{aligned}$$

```
> fegen:=proc()
> local a,i,u,des;
> des:=[];
> for a from 0 to 2 by 0.01 do
> u:=a;
> for i from 1 to 100 do
> u:=u^2-a:
> od;
> for i from 1 to 100 do
> u:=u^2-a;
> des:=[op(des),[a,u]];
> od;
> od;
> des;
> end;
```

```
fegen := proc()
local a, i, u, des;
    des := [] ;
    for a from 0 by 0.01 to 2 do
        u := a ;
        for i to 100 do u := u^2 - a end do ;
        for i to 100 do u := u^2 - a ; des := [op(des), [a, u]] end do
    end do;
    des
end proc
```

```
> dessin:=plot(fegen(),x=0..2,style=point):
> display(dessin);
```

```
Warning, the name changecoords has been redefined
```



Les traductions avec Maxima ne posent ici pas de problème particulier.

Chapitre 7

Arithmétique et cryptographie

7.1 Méthode RSA

Le codage RSA[1] est un codage à clef publique ; toutes les indications nécessaires au codage d'un message sont accessibles à tous. En revanche, seul le destinataire sera capable de décoder le message.

Cette curiosité repose sur le fait que la multiplication de deux nombres est une opération facile (au sens où le temps nécessaire pour l'effectuer est court) alors que la factorisation en produit de facteurs premiers est difficile, c'est à dire, qu'aucun algorithme rapide de factorisation n'est connu à condition, bien sûr, que les nombres soient grands.

Essayez, par exemple :

>t :=time() :p :=nextprime(123456789123456789123) ; q :=nextprime(987654321987654321987) ; n :=p*q ;time()-t ;

>t :=time() :ifactor(n) ;time()-t ;

nextprime(a) est la fonction qui à a fait correspondre le plus petit nombre premier plus grand que a.

7.1.1 Mode d'emploi :

- choisir deux nombres premiers p et q aussi grands que possible
- calculer $n = p \times q$
- calculer $\phi(n) = (p-1) \times (q-1)$
- choisir un nombre e inférieur à $\phi(n)$ et premier avec $\phi(n)$
- calculer d tel que $ed \equiv 1[\phi(n)]$

n et e sont publics p, q, $\phi(n)$ et d sont secrets.

Pour coder un message on calcule

$$C \equiv M^e[n]$$

[1] Du nom des mathématiciens qui sont à l'origine de ce codage : Rivest, Shamir et Adleman ; l'article des trois mathématiciens date de 1978

(M est un message numérique inférieur à n) ; ne sont donc nécessaires que les éléments publics du codage.

Pour décoder le message on calcule

$$C^d[n]$$

On a donc besoin de d et n secrets.

7.1.2 Justification du codage :

Soit M un nombre inférieur à n et $C = M^e$ dans Z/nZ

$$C^d = M^{ed} = M^{k\phi(n)+1} = \left(M^{\phi}(n)\right)^k M$$

Si M et $\phi(n)$ sont premiers entre eux, le théorème d'Euler permet de conclure.
Sinon
M peut s'écrire : $M = p^{\alpha}q^{\beta}m$ avec m premier avec n (α ou β pouvant être nuls.)
Comme $ed \equiv 1$ modulo $\phi(n)$, $q-1$ divise $ed-1$ et donc $\exists k$ tel que $ed-1 = k(q-1)$
Mais alors :

$$p^{ed} = pp^{ed-1} = p\left(p^{q-1}\right)^k \equiv p \text{ modulo } q$$

On en déduit que $p^{ed} = p$ modulo n
en effet : $p^{ed} - p \equiv 0$ modulo q donc $\exists k$ tel que $p^{ed} - p = kq$ donc p divise k (théorème de Gauss) donc $p^{ed} = p$ modulo n
De la même façon, on démontrerait que $q^{ed} = q$ modulo n, et finalement :

$$M^{ed} = \left(p^{ed}\right)^{\alpha}\left(q^{ed}\right)^{\beta} m^{ed} \equiv M \text{ modulo } n$$

TP

Pour programmer la méthode RSA, il est nécessaire de savoir construire des nombres premiers grands, de trouver les coefficients de Bezout de deux nombres premiers entre eux, et de calculer la puissance d'un nombre modulo n.

Il existe des commandes Maple et Maxima pour effectuer ces opérations :

Maple	Maxima	
nextprime(a)	next_prime(a)	rend le plus petit nombre premier supérieur à a
gcd(a,b)	gcd(a,b)	rend le pgcd de a et b
igcdex(a,b,'u','v')	Non disponible sur Maxima, à programmer	rend le pgcd de a et b et affecte à u et v des coeffici Bezout tels que au+bv=pgcd(a,b)
power(M,e) mod n	power_mode(M,e,n)	Calcule M à la puissance e modulo n

En utilisant ces commandes, programmez la méthode de codage RSA.

7.1.3 Pour aller plus loin

Trouver des grands nombres premiers

Le critère de Lehmer permet de savoir si un nombre n est premier lorsqu'on connait la décomposition en facteurs premiers de $n-1$.

Théorème : Soit n un entier impair supérieur à 2.

n est premier si et seulement si il existe un entier a vérifiant à la fois :

$$a^{\frac{n-1}{2}} \equiv -1[n]$$

$$a^{\frac{n-1}{q}} \not\equiv 1[n]$$

pour tout facteur premier q de $n-1$.

Ce critère suggére une façon de trouver des grands nombres premiers en partant de deux nombres premiers impairs$p$ et q. On applique le critère à $n = 2^x p^y q^z + 1$ pour savoir s'il est premier en essayant différentes valeurs de x, y et z et plusieurs valeurs de a.

Ecrire une procédure qui utilise ce critère pour déterminer des nombres premiers.

Algorithme d'Euclide étendu

L'algorithme d'Euclide calcule le pgcd de deux nombres mais cet algorithme ne fournit pas les coefficients de Bezout. L'algorithme d'Euclide étendu est une généralisation :

euclide_etendu(a,b,u1,v1,u2,v2)
  si (b=0) alors rendre a,u1 et u2
    sinon euclide_etendu(b, a mod b, v1, u1-iquo(a,b)*v1, v2, u2-iquo(a,b)*v2) fi
fin :

En initialisant avec les valeurs u1=1, u2=0, v1=0, v2=1.

7.2 Eléments de solution

```
> p:=nextprime(rand());
```

$$p := 427419669163$$

```
> q:=nextprime(rand());
```

$$q := 321110693273$$

```
> n:=p*q;
```

$$n := 137249026283447229640499$$

```
> phi:=(p-1)*(q-1);
```

$$\phi := 137249026282698699278064$$

```
> det_e:=proc(x)
> local de,i;
> de:=rand(1..x);
> for i from de() to x while gcd(i,x)<>1 do
> od;
> i;
> end:
> e:=det_e(phi);
```

$$e := 6913502113207685758 7437$$

```
> igcdex(e,phi,'u','v');
```

$$1$$

```
> u;
```

$$-42709855198900824968795$$

```
> v;
```

$$21513790091610782712444$$

```
> u*e mod phi;
```

$$1$$

```
> det_d:=proc(e,phi)
> local aux,ee,phiphi;
> ee:=e;phiphi:=phi;aux:=igcdex(ee,phi,'u');
> if u>0 then u else u+phiphi fi;
> end;
```

$$det\_d := \mathbf{proc}(e,\ \phi)$$
$$\mathbf{local}\ aux,\ ee,\ phiphi;$$
$$ee := e\,;$$
$$phiphi := \phi\,;$$
$$aux := \mathrm{igcdex}(ee,\ \phi,\ 'u')\,;$$
$$\mathbf{if}\ 0 < u\ \mathbf{then}\ u\ \ \mathbf{else}\ u + phiphi\ \mathbf{end\ if}$$
$$\mathbf{end\ proc}$$

```
> d:=det_d(e,phi);
```

$$d := 94539171083797874309269$$

```
> M:=123456789;
```

$$M := 123456789$$

```
> C:=power(M,e) mod n;
```

$$C := 83604444564558733821412$$

```
> DM:=power(C,d) mod n;
```

$$DM := 123456789$$

Avec Maxima, on peut retrouver les coefficients de Bezout en utilisant la fonction suivante, ce qui donne, de plus une correction du dernier exercice demandé :

```
(\%i1) euclide(a,b,u1,v1,u2,v2):=block(
if b=0 then [a,u1,u2] else euclide(b,mod(a,b),v1,u1-floor(a/b)*v1,v2,u2-floor(a/b)*v2)
)$(%i2) bezout(a,b):=euclide(a,b,1,0,0,1);
```

(%o2) $$bezout\,(a,b) := euclide\,(a,b,1,0,0,1)$$

```
(%i3) bezout(12345,45321);
```

(%o3) $$[3, 6744, -1837]$$

On utilisera la fonction "random(x)" avec x un entier qui rend un nombre pseudo aléatoire entre 1 et x-1.

Chapitre 8

Courbes et surfaces paramétrées : l'exemple de Bézier

8.1 Courbes de Bézier

Le modèle de Bézier (proposé par Pierre Bézier en 1962) a été développé dans les bureaux d'étude de Renault. Il s'agit d'un modèle permettant de créer des formes du plan et de l'espace à partir de points de contrôle.

D'une façon générale si t est une variable de l'intervalle $[0, 1]$ et $P_0, P_1, ...P_n$ n+1 points, on définit la famille de polynômes de la manière suivante :

$$B_n^i : t \quad \rightarrow \quad C_n^i t^i (1-t)^{n-i}$$

On définit alors le point M(t) dans un repère orthonormé par :

$$\overrightarrow{OM}(t) \quad = \quad \sum_{i=0}^{n} B_i^n(t) \overrightarrow{OP_i} \tag{8.1}$$

Pour chaque t, M apparaît comme le barycentre du système de points pondérés : $(P_i, B_i^n(t))$

Par ailleurs, la définition (1) permet de construire une courbe ayant les propriétés élémentaires suivantes :

1. Elle passe par P_0 et P_n
2. La tangente à la courbe en P_0 est la droite $(P_0 P_1)$

1) il suffit pour s'en convaincre de donner à t les valeurs 0 puis 1 !

2)

$$\frac{d\overrightarrow{OM}(t)}{dt} = -n(1-t)^{n-1}\overrightarrow{OP_0} + n\left((1-t)^{n-1} - (n-1)\,t\,(1-t)^{n-2}\right)\overrightarrow{OP_1} + t\,(\ldots)$$

$$\frac{d\overrightarrow{OM}(t)}{dt}(0) = n\overrightarrow{P_0P_1}$$

On démontrerait de même que :

$$\frac{d\overrightarrow{OM}(t)}{dt}(1) = n\overrightarrow{P_{n-1}P_n}$$

8.2 Questions

1. Définir une fonction B qui à (n, i, t) fait correspondre $B_n^i(t)$. Remarque : En Maple, C_n^p s'écrit binomial(n,p)
2. Définir une fonction M qui à une liste de points (une liste de listes à deux éléments) et à t fait correspondre le vecteur $\overrightarrow{OM}(t)$ donné par la liste de ses composantes.
3. Utiliser cette fonction pour tracer la courbe de Bézier ayant comme points de contrôle : $[0, 0], [1, 2], [3, 0]$ puis $[0, 0], [1, 3], [3, 0]$
4. Tracer une courbe dérivable en tout point passant par les points $[0, 0], [1, 2], [3, 0][1, -2]$ en rejoignant deux courbes de Bézier de degré 2 (3 points de contrôle), ou en utilisant une courbe de Bézier à 4 points de contrôle.
5. Dans une courbe de Bézier à 4 points de contrôle, M apparaît comme le barycentre du système de points pondérés

   $$(P_0, (1-t)^3), (P_1, 3t(1-t)^2), (P_2, 3t^2(1-t)), (P_3, t^3)$$

   Représenter graphiquement les poids respectifs en fonction de t sur un même graphique.

Exemples de surfaces de Bézier

Une extension toute naturelle des courbes de Bézier est la représentation de surfaces de Bézier. La surface est construite à partir d'une matrice $n \times m$ de points de l'espace qui forment une grille de points de contrôle :

$$\{P_{i,j} : 0 \leq i \leq n, 0 \leq j \leq m\}$$

La surface est alors définie paramétriquement à l'aide de deux variables et est donnée par la formule :

$$P(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u) B_j^m(v) P_{i,j}$$

Tracer par exemple la surface de Bézier correspondant aux points de contrôles :
[0,0,0],[0,1,0],[-1,2,0.5],[-1.5,2.5,1]
[1,0,1],[1,1,1],[-1,2,1.5],[-1.5,2.5,2]
[0.5,0,2],[1,1,2],[-1,2,2.5],[-2,2.5,3]
[0,0,3],[0,1,3],[-1,2,4],[-2,2.5,4]

8.3 Eléments de solution

```
> B:=(n,i,t)->binomial(n,i)*t^i*(1-t)^(n-i):
```

```
> l:=[[0,0],[1,2],[3,0]];l2:=[[0,0],[1,3],[3,0]]:
```

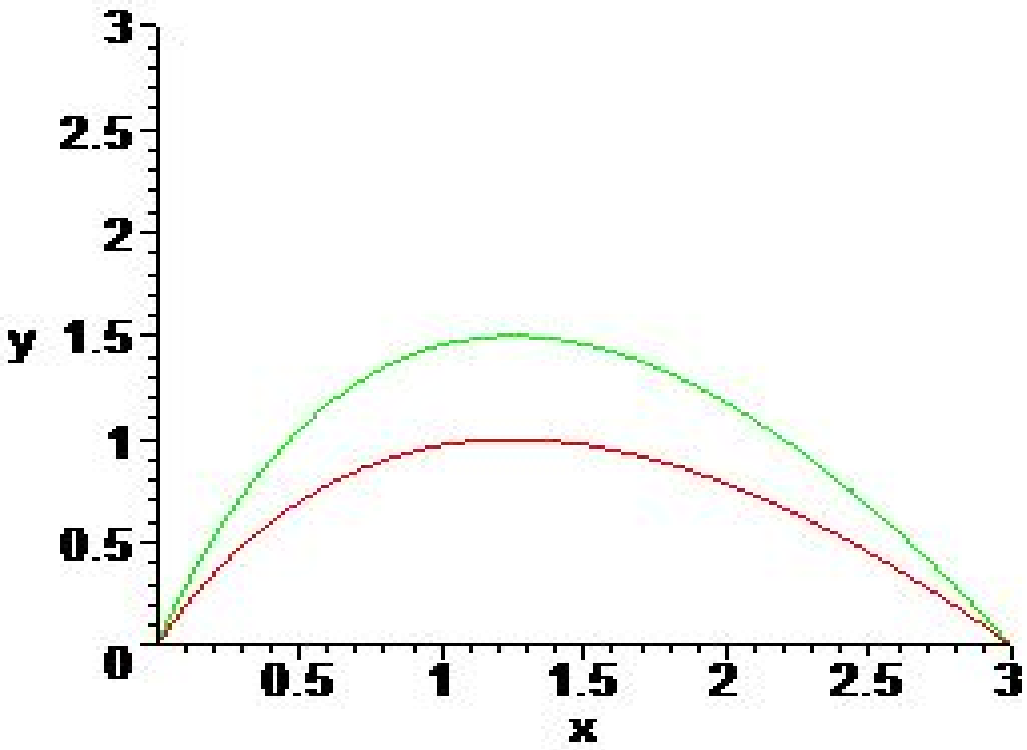
$$l := [[0,\ 0],\ [1,\ 2],\ [3,\ 0]]$$

```
> M:=(t,l)->[sum(B(nops(l)-1,i,t)*l[i+1][1],i=0..nops(l)-1),sum(B(nops(
> l)-1,i,t)*l[i+1][2],i=0..nops(l)-1)]:
```

```
> subs(t=1,M(t,l));
```

$$[3,\ 0]$$

```
> plot({[op(M(t,l)),t=0..1],[op(M(t,l2)),t=0..1]},x=0..3,y=0..3);
```



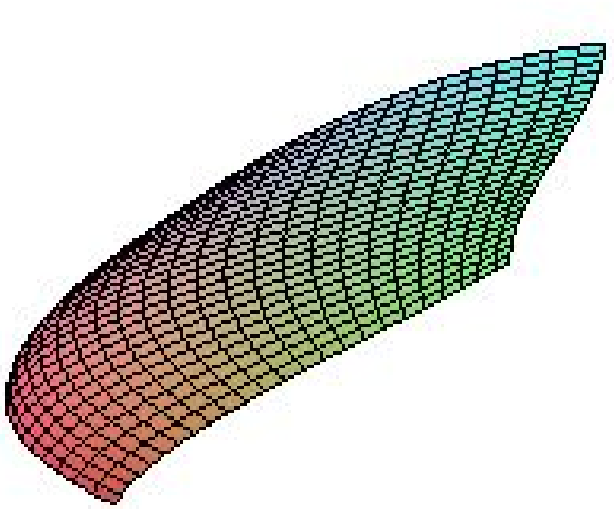

```
> carreau:=[[[0,0,0],[0,1,0],[-1,2,0.5],[-1.5,2.5,1]],[[1,0,1],[1,1,1],
> [-1,2,1.5],[-1.5,2.5,2]],[[0.5,0,2],[1,1,2],[-1,2,2.5],[-2,2.5,3]],[[0
> ,0,3],[0,1,3],[-1,2,4],[-2,2.5,4]]]:s:=[[[0,0,0],[2,0,0]],[[0,0,2],[0,
> 2,0]]]:
```

```
> Px:=(u,v,carreau)->sum(sum(carreau[i+1][j+1][1]*binomial(nops(carreau
> )-1,i)*binomial(nops(carreau[1])-1,j)*u^(i)*(1-u)^(nops(carreau)-1-i)*
> v^(j)*(1-v)^(nops(carreau[1])-1-j),i=0..nops(carreau)-1),j=0..nops(car
> reau[1])-1):
```

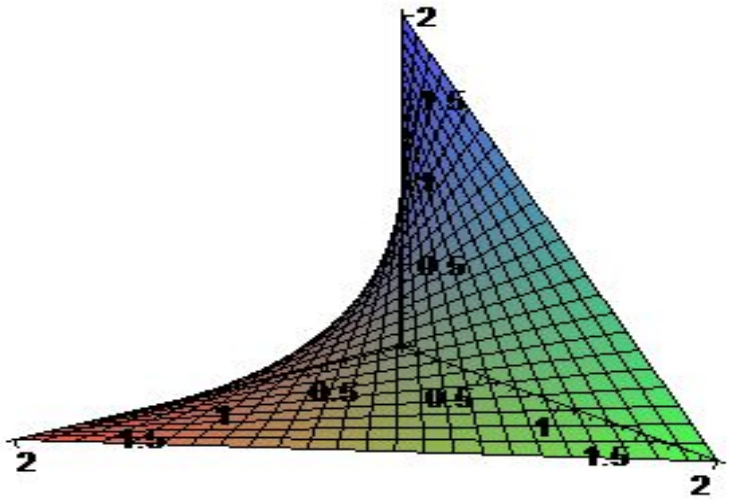
```
> Py:=(u,v,carreau)->sum(sum(carreau[i+1][j+1][2]*binomial(nops(carreau
> )-1,i)*binomial(nops(carreau[1])-1,j)*u^(i)*(1-u)^(nops(carreau)-1-i)*
> v^(j)*(1-v)^(nops(carreau[1])-1-j),i=0..nops(carreau)-1),j=0..nops(car
> reau[1])-1):


> Pz:=(u,v,carreau)->sum(sum(carreau[i+1][j+1][3]*binomial(nops(carreau
> )-1,i)*binomial(nops(carreau[1])-1,j)*u^(i)*(1-u)^(nops(carreau)-1-i)*
> v^(j)*(1-v)^(nops(carreau[1])-1-j),i=0..nops(carreau)-1),j=0..nops(car
> reau[1])-1):


> plot3d([Px(u,v,carreau),Py(u,v,carreau),Pz(u,v,carreau)],u=0..1,v=0..
> 1);
```



```
> plot3d([Px(u,v,s),Py(u,v,s),Pz(u,v,s)],u=0..1,v=0..1);
```



Avec Maxima, on peut également répondre aux questions posées :

8.3.1 Courbes avec Maxima

```
B(n,i,t):=binomial(n,i)*t^i*(1-t)^(n-i);
```

(%o1) $$B\left(n,i,t\right):=\binom{n}{i}\,t^{i}\left(1-t\right)^{n-i}$$

```
(%i2) l:[[0,0],[1,2],[3,0]];l2:[[0,0],[1,3],[3,0]];
```

(%o3) $$[[0,0],[1,2],[3,0]][[0,0],[1,3],[3,0]]$$

```
(%i5) f(t,l):=sum(B(length(l)-1,i,t)*l[i+1][1],i,0,length(l)-1);
```

(%o5) $$f\left(t,l\right):=sum\left(B\left(length\left(l\right)-1,i,t\right)\,\left(l_{i+1}\right)_{1},i,0,length\left(l\right)-1\right)$$

```
(%i6) g(t,l):=sum(B(length(l)-1,i,t)*l[i+1][2],i,0,length(l)-1);
```

(%o6) $g\left(t,l\right) := sum\left(B\left(length\left(l\right)-1,i,t\right)\ \left(l_{i+1}\right)_2, i, 0, length\left(l\right)-1\right)$

```
(%i7) plot2d([['parametric, f(t,l), g(t,l), [t, 0, 1], [nticks, 300]],['parametric, f(t,l2),
  g(t,l2), [t, 0, 1], [nticks, 300]]], [x,0,3], [y,0,3],
 [plot_format, gnuplot])$
```

8.3.2 Surfaces avec Maxima

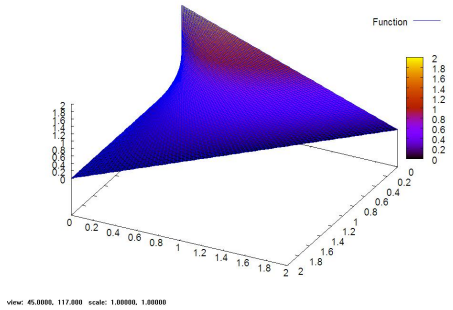
(%i7) Px(u,v,carreau) :=sum(sum(carreau[i+1][j+1][1]*binomial(length(carreau)-1,i)*binomial(length(carre
-1,j)*$u^(i)*(1-u)^(length(carreau)-1-i)*v^(j)*(1-v)^(length(carreau[1])-1-j)$,i,0,length(carreau)
-1),j,0,length(carreau[1])-1)

(%i8) Py(u,v,carreau) :=sum(sum(carreau[i+1][j+1][2]*binomial(length(carreau)-1,i)*binomial(length(carre
-1,j)*$u^(i)*(1-u)^(length(carreau)-1-i)*v^(j)*(1-v)^(length(carreau[1])-1-j)$,i,0,length(carreau)
-1),j,0,length(carreau[1])-1)

(%i9) carreau :[[[0,0,0],[0,1,0],[-1,2,0.5],[-1.5,2.5,1]],[[1,0,1],[1,1,1], [-1,2,1.5],[-1.5,2.5,2]],[[0.5,0,2],[1,1,2],[-
1,2,2.5],[-2,2.5,3]],[[0 ,0,3],[0,1,3],[-1,2,4],[-2,2.5,4]]]

(%i10) Pz(u,v,carreau) :=sum(sum(carreau[i+1][j+1][3]*binomial(length(carreau)-1,i)*binomial(length(carr
-1,j)*$u^(i)*(1-u)^(length(carreau)-1-i)*v^(j)*(1-v)^(length(carreau[1])-1-j)$,i,0,length(carreau)
-1),j,0,length(carreau[1])-1)

(%i11) plot3d([Px(x,y,carreau),Py(x,y,carreau),Pz(x,y,carreau)],[x,0,1],[y,0,1],['grid,40,40]) ;



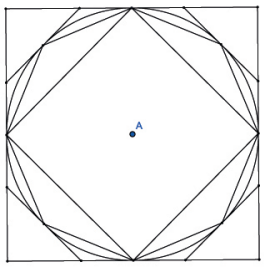
Chapitre 9

π, un voyage dans l'histoire des maths

Dans ce TP de détermination de valeurs approchées de π, nous ferons un petit voyage à travers l'histoire des mathématiques.

9.1 Première époque, le règne de la géométrie : la méthode d'Archimède

Le nombre π apparaît à l'origine comme la longueur du cercle de diamètre 1. La méthode d'Archimède de détermination de π consiste à encadrer le cercles par deux polygones, l'un inscrit et l'autre exinscrit dont les longueurs encadrent la longueur du cercle puis d'augmenter le nombre de côtés des polygones.
Sur ce dessin, on part des carrés dont les périmètres sont un premier encadrement de π :



$$2\sqrt{2} \leq \pi \leq 4$$

On double alors le nombre de côtés du polygone :

1. Déterminer la formule de récurrence donnant le périmètre du polygone inscrit à $2n$ côtés connaissant le périmètre du polygone inscrit à n côtés.
2. Même question pour le polygone exinscrit.
3. Programmer avec Maple cet encadrement de π.

9.2 Deuxième époque, le triomphe de l'algèbre : la méthode de Viète

La méthode de Viète est la première méthode débouchant sur un produit infini. Elle consiste à mesurer l'aire d'un polygone à 2^n côtés inscrit dans le cercle de rayon 1.

1. En utilisant des considérations géométriques élémentaires, démontrer que le rapport des aires des polygones à 2^{n+1} côtés et à 2^n côtés vaut $\frac{1}{\cos\left(\frac{\pi}{2^n}\right)}$
2. En utilisant la formule $\cos(a) = \frac{\sqrt{2\cos(2a)+2}}{2}$ en déduire que :
$$\pi \approx 2 \times \frac{2}{\sqrt{2}} \times \frac{2}{\sqrt{2+\sqrt{2}}} \times \frac{2}{\sqrt{2+\sqrt{2+\sqrt{2}}}} \times \cdots$$
3. Programmez cette méthode avec Maple.

9.3 Troisième époque, l'avènement de l'analyse :

9.3.1 la méthode de Wallis

Wallis publie en 1655 la formule de produit infini suivante :

$$\pi = 2 \times \frac{2 \times 2}{1 \times 3} \times \frac{4 \times 4}{3 \times 5} \times \frac{6 \times 6}{5 \times 7} \cdots$$

Preuve :
On considère l'intégrale

$$W_n = \int_0^{\frac{\pi}{2}} \cos^n(x)dx$$

que l'on intègre en utilisant une intégration par parties :

$$W_n = \int_0^{\frac{\pi}{2}} \cos^{n-1}(x)\cos(x)dx = \Big[sin(x)\cos^{n-1}(x)\Big]_0^{\frac{\pi}{2}} + \int_0^{\frac{\pi}{2}} (n-1)\sin^2(x)\cos^{n-2}(x)dx$$

et, en remplaçant $\sin^2 x$ par $1 - \cos^2 x$ on trouve :

$$nW_n = (n-1)W_{n-2}$$

En exprimant W_{2p} en fonction de W_0 et W_{2p+1} en fonction de W_1, on obtient :

$$W_{2p} = W_0 \frac{1 \times 3 \times 5 \times \cdots \times (2p-1)}{2 \times 4 \times 6 \times \cdots \times 2p} = \frac{\pi}{2} \times \frac{(2p)!}{2^{2p}(p!)^2}$$

$$W_{2p+1} = W_1 \frac{2 \times 4 \times 6 \times \cdots \times 2p}{1 \times 3 \times 5 \times \cdots \times (2p+1)} = \frac{\pi}{2} \times \frac{1}{(2p+1)W_{2p}}$$

Du fait que x appartient à l'intervalle $[0, \frac{\pi}{2}]$, $\cos^{n+1}(x) \leq \cos^n(x)$ et donc :

$$W_{n+2} \leq W_{n+1} \leq W_n$$

$$\frac{n+1}{n+2} = \frac{W_{n+2}}{W_n} \leq \frac{W_{n+1}}{W_n} \leq 1$$

On en déduit que $\frac{W_{n+1}}{W_n}$ tend vers 1 quand n tend vers $+\infty$, puis la formule de Wallis.

1. Programmez la formule de Wallis
2. Comparez la vitesse de convergence avec les formules précédentes.

9.3.2 La méthode de Newton

Il s'agit d'une généralisation de la formule du binôme en donnant à n des valeurs quelconques :

$$(1+x)^a = 1 + ax + \frac{a(a-1)}{2}x^2 + \frac{a(a-1)(a-2)}{2\times 3}x^3 + \cdots$$

En remarquant, par ailleurs que la dérivée de arcsin est $(1-x^2)^{\frac{-1}{2}}$, on en déduit un développement de arcsin en somme infinie :

$$\arcsin(x) = x + \frac{1}{2}\times\frac{x^3}{2} + \frac{1\times 3}{2\times 4}\times\frac{x^5}{5} + \cdots + \frac{1\times 3\times\cdots\times(2p-1)}{2\times 4\times\cdots\times 2p}\times\frac{x^{2p+1}}{2p+1} + \cdots$$

Et, en prenant $x = \frac{\pi}{2}$, il vient :

$$\pi = 6\left(\frac{1}{2} + \frac{1}{2}\times\frac{1}{3}\times\frac{1}{2^3} + \cdots + \frac{1\times 3\times\cdots\times(2p-1)}{2\times 4\times\cdots\times 2p}\times\frac{1}{2p+1}\times\frac{1}{2^{2p+1}} + \cdots\right)$$

1. Programmez la formule de Newton
2. Comparez la vitesse de convergence avec les formules précédentes

9.4 Quatrième époque, des statistiques aux probabilités

9.4.1 la méthode de Monte-Carlo

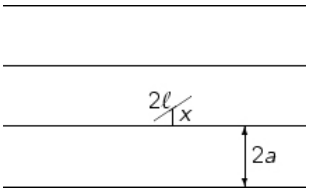
Algorithme : Choisir deux nombres au hasard entre 0 et 1.

Ces nombres sont les coordonnées d'un point du carré unité. La probabilité qu'il appartienne au quart de cercle est égal au rapport des aires du quart de disque et du carré unité soit $\frac{\pi}{4}$

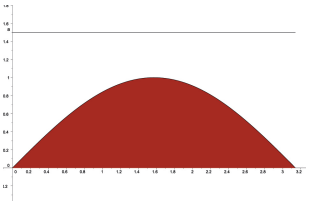
1. Ecrire une procédure Maple simulant cette expérience aléatoire.
2. En déduire une valeur approchée de π

9.4.2 L'aiguille de Buffon

Sur un parquet formé de planches de largeur $2a$, séparées par des rainures droites, parallèles et équidistantes, on jette une aiguille de longueur $2l$, avec $l < a$.
Quelle est la probabilité que l'aiguille coupe l'une des rainures (événement A) ?
Soit x la distance du milieu de l'aiguille à la rainure la plus proche. x prend une valeur aléatoire quelconque dans $[0, a]$.
Soit θ l'angle des droites formées par cette rainure et l'aiguille. θ prend une valeur aléatoire quelconque dans $[0, \pi]$.



Dans une représentation cartésienne du pavé $\Omega = [0, a] \times [0, \pi]$, l'événement A est représenté par la partie hachurée délimitée par la courbe d'équation $y = l \sin(\theta)$.
Dans cette représentation, on prend pour modèle probabiliste la loi uniforme sur le rectangle Ω. La probabilité de A est donc le rapport de l'aire sous la courbe et du rectangle de longueur π et de hauteur a.



On a donc

$$p(A) = \frac{\int_0^\pi l \cdot sin(\theta) d\theta}{\pi \cdot a} = \frac{2l}{\pi a}$$

1. Ecrire une procédure Maple pour simuler l'expérience de l'aiguille de Buffon.

9.5 Cinquième époque, la puissance des algorithmes au vingtième siècle

En 1973, Eugène Salamin et Richard Brent ont publié indépendamment un algorithmes fondé sur la moyenne arithmetico-géométrique dont la convergence est très rapide :

$$\begin{array}{ccc} a_0 = 1 & b_0 = \frac{\sqrt{2}}{2} & s_0 = \frac{1}{2} \\ a_k = \frac{a_{k-1}+b_{k-1}}{2} & b_k = \sqrt{a_{k-1}b_{k-1}} & c_k = a_k^2 - b_k^2 \\ s_k = s_{k-1} - 2^k c_k & p_k = 2\frac{a_k^2}{s_k} & \end{array}$$

p_k est une approximation de π.

1. Programmez cet algorithme
2. Comparez la vitesse de convergence avec les formules précédentes.

9.6 Eléments de solution

9.6.1 Session Maple

Méthode d'Archimède

```
> inscrit:=proc(n)
> option remember;
> if n=4 then sqrt(2)/2 else sqrt(1/2*(1-sqrt(1-inscrit(n/2)^2))) fi;
> end:
> peri_inscrit:=proc(n)
> n*inscrit(n);
> end:
> exinscrit:=proc(n)
> option remember;
> if n=4 then 1 else (-1+sqrt(1+exinscrit(n/2)^2))/exinscrit(n/2) fi;
> end:
> peri_exinscrit:=proc(n)
> n*exinscrit(n);
> end:
> seq([evalf(peri_inscrit(2^i),20)," < Pi <
> ",evalf(peri_exinscrit(2^i),20)],i=2..10);
```

$$
\begin{aligned}
&[2.8284271247461900976, \text{“ } < \text{Pi} < \text{ ”}, 4.],\\
&[3.0614674589207181738, \text{“ } < \text{Pi} < \text{ ”}, 3.3137084989847603 90],\\
&[3.1214451522580522851, \text{“ } < \text{Pi} < \text{ ”}, 3.1825978780745281106],\\
&[3.1365484905459392619, \text{“ } < \text{Pi} < \text{ ”}, 3.1517249074292561011],\\
&[3.1403311569547529106, \text{“ } < \text{Pi} < \text{ ”}, 3.1441183852459042449],\\
&[3.1412772509327728873, \text{“ } < \text{Pi} < \text{ ”}, 3.1422236299424568865],\\
&[3.1415138011443011585, \text{“ } < \text{Pi} < \text{ ”}, 3.1417503691689661097],\\
&[3.1415729403670918356, \text{“ } < \text{Pi} < \text{ ”}, 3.1416320807031806711],\\
&[3.1415877252771602840, \text{“ } < \text{Pi} < \text{ ”}, 3.1416025102568050317]
\end{aligned}
$$

Méthode de Viète

On remarquera la puissance de la programmation récursive !

```
> ccos:=proc(n)
> if n=4 then sqrt(2)/2 else sqrt(2*ccos(n/2)+2)/2 fi;
> end:
> aire:=proc(n)
> if n=4 then 2 else aire(n/2)/ccos(n/2) fi;
> end:
> seq(evalf(aire(2^i),20),i=2..10);
```

2., 2.8284271247461900976, 3.0614674589207181738, 3.1214451522580522854, 3.1365484905459392637, 3.1403311569547529122, 3.1412772509327728680, 3.1415138011443010764, 3.1415729403670913841

Méthode de Wallis

```
> wallis:=proc(n)
> product(2*i,i=1..n)^2/product(2*i+1,i=0..n-1)^2*2/(2*n+1);
> end:
> seq(evalf(wallis(k),20),k=990..1000);
```

3.1407998226740813754, 3.1408006222009590466, 3.1408014201169025515, 3.1408022164267756985, 3.1408030111354227355, 3.1408038042476684484, 3.1408045957683182586, 3.1408053857021583199, 3.1408061740539556154, 3.1408069608284580529, 3.1408077460303945605

Méthode de Newton

```
> newton:=proc(n)
> local i,s;
> s:=1/2;
> for i from 1 to n do
> s:=s+product(2*k-1,k=1..i)/product(2*k,k=1..i)*(1/((2*i+1)*2^(2*i+1)))
> ;
> od;
> 6*s;
> end:
> seq(evalf(newton(p),20),p=990..1000);
```

3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385, 3.1415926535897932385

L'algorithme de Salamin et Brent

```
> salbre:=proc(n)
> local a,b,c,s,p,i,l;
> l:=[1,sqrt(2)/2,1/2,1/2];
> for i from 1 to n do
> a:=(op(1,l)+op(2,l))/2;
> b:=sqrt(op(1,l)*op(2,l));
> c:=a^2-b^2;
> s:=op(4,l)-2^i*c;
> l:=[a,b,c,s];
> od;
> 2*a^2/s;
> end:
```

```
> seq(evalf(salbre(i),100),i=2..8);
```

3.14168029329765329391807042456000938279571943881540283264418946314\
95663001010255319388889427515264602, 3.14159265389544649600299\
14758818043486108879237261311589651101357684653079503086501\
774097586289863172, 3.14159265358979323846636060270663132175 7\
70241134242935648684601523841094860692775826806220073327629\
4, 3.14159265358979323846264338327950288419716994916472660583\
46961259487480060953290058518515759317103 14, 3.14159265358979\
32384626433832795028841971693993751058209749445923078164062\
862089986280468522286543 48, 3.14159265358979323846264338327 95\
02884197169399375105820974944592307816406286208998628034825\
342117232, 3.14159265358979323846264338327950288419716939937 5\
10582097494459230781640628620899862803482534211682 2

Méthode de Monte-Carlo

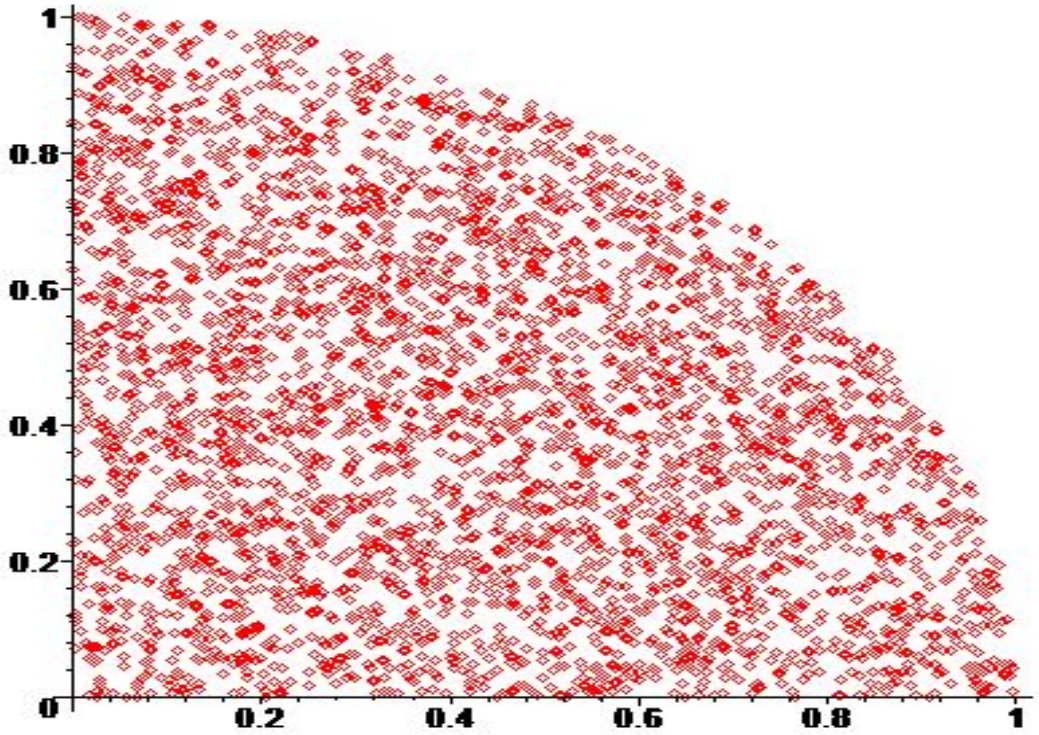
```
> montecarlo:=proc(n)
> local x,y,i,p;
> p:=0;
> for i from 1 to n do
> x:=rand()/1E12;y:=rand()/1E12;
> if evalf(x^2+y^2)<1 then p:=p+1; fi;
> od;
> evalf(4*p/n);
> end:

> seq(montecarlo(10000),i=1..10);
```

3.127600000, 3.136000000, 3.157200000, 3.150400000, 3.152800000, 3.139600000, 3.131600000, 3.147600000, 3.136800000, 3.140800000

```
> montecarlo2:=proc(n)
> local x,y,i,p;
> p:=[];
> for i from 1 to n do
> x:=rand()/1E12;y:=rand()/1E12;
> if evalf(x^2+y^2)<1 then p:=[op(p),[x,y]] fi;
> od;
> plot(p,style=point);
> end:

> montecarlo2(5000);
```



L'aiguille de Buffon

```
> buffon:=proc(n)
> local i,x,theta,compt,a,l;
> a:=1;l:=1;compt:=0;
> for i from 1 to n do
> x:=rand()/1E12;
> theta:=rand()/1E12 * Pi;
> if evalf(l*sin(theta))>x then compt:=compt+1 fi;
> od;
> 2*n/compt;
> end:
> for i from 1 to 10 do print(evalf(buffon(10000))) od;
```

3.141690229
3.085943527
3.133322889
3.125488358
3.160056881
3.171582620

3.123048095
3.168567807
3.129400720
3.089280198

A la suite

```
> f:=(x,n)->(x-x^2)^(4*n)/((x^2+1)*4^(n-1)):
> g:=(x,n)->convert(f(x,n)-(-1)^n*4/(x^2+1),parfrac,x);
```

$$g := (x,\, n) \to \mathrm{convert}(\mathrm{f}(x,\, n) - \frac{4\,(-1)^n}{x^2+1},\, parfrac,\, x)$$

```
> for i from 1 to 20 do print(evalf(abs(int(g(x,i),x=0..1)),50+2*i))
> od;
```

3.1428571428571428571

```
                3.14159265358979323846264338327950288419716939937486475316125263 11\
                81540645590331588
                3.14159265358979323846264338327950288419716939937510604944583461 31\
                2311227916146682502
                3.14159265358979323846264338327950288419716939937510582075804539 95\
                318646329107670512117
                3.14159265358979323846264338327950288419716939937510582097515081 74\
                8967023478080914612224 2
                3.14159265358979323846264338327950288419716939937510582097494439 59\
                6481234595582937495567002
> evalf(Pi,90);

                3.14159265358979323846264338327950288419716939937510582097494459 23\
                0781640628620899862803483
```

A suivre...