Benjamin COLLAS
benjamin.collas@telaxo.com

Mi-101 *Mathematica* 2003-2004

Partiel 2003 - durée 1 heure

Listes et fonctions

■ Listes

Etablissez les commandes qui permettent d'obtenir les résultats suivants

```
Sqrt[Range[10]]
```

$\{1, \sqrt{2}, \sqrt{3}, 2, \sqrt{5}, \sqrt{6}, \sqrt{7}, 2\sqrt{2}, 3, \sqrt{10}\}$

```
Prime[Range[1, 12, 2]]
```

{2, 5, 11, 17, 23, 31}

```
Pi + Range[5] + Range[5] * x
```

$\{1+\pi+x, 2+\pi+2x, 3+\pi+3x, 4+\pi+4x, 5+\pi+5x\}$

```
Table[(k + 1) * x^k, {k, 0, 5}]
```

$\{1, 2x, 3x^2, 4x^3, 5x^4, 6x^5\}$

```
Table[{i, j}, {i, 1, 5}, {j, 1, 3}]
```

```
{{{1, 1}, {1, 2}, {1, 3}}, {{2, 1}, {2, 2}, {2, 3}},
 {{3, 1}, {3, 2}, {3, 3}}, {{4, 1}, {4, 2}, {4, 3}}, {{5, 1}, {5, 2}, {5, 3}}}
```

■ Fonctions

Créer les fonctions suivantes

f1 :
Qui à une liste *lis1*, une liste *lis2* et un élément *elem* associe la liste dont les éléments sont, dans l'ordre, les éléments de *lis1*, *elem*, puis les éléments de *lis2.*

```
f1[lis1_, elem_, lis2_] := Join[lis1, {a}, lis2]
```

f2:
Qui à une liste associe la liste formée des longeurs de ses éléments.

```
f2[lis_] := Map[Length, lis]
```


f3:
Qui à une liste associe la somme de ses troisième et quatrièmes éléments divisée par *2*.

```
f3[lis_] := (lis[[3]] + lis[[4]]) / 2
```

f4:
Qui à une liste associe cette même liste dont on a jeté tous les éléments de position impaire.

```
f4[lis_] := Drop[lis, {1, Length[lis], 2}]
```

f5:
Qui à un entier associe *True* si c'est un premier jumeau et *False* sinon.
NB : On dit qu'un entier est jumeau s'il est premier et si augmenté de *2* ou diminué de *2,* il est encore premier.

```
f5[n_] := And[PrimeQ[n], Or[PrimeQ[n - 2], PrimeQ[n + 2]]]
```

Exercice d'algèbre linéaire

Dans cet exercice, on se propose de trouver l'ensemble solution au système d'équations de paramètre *a* suivant. La discussion portera sur la valeur du paramètre ainsi que sur le vecteur *(x0,y0,z0).*

$$\begin{array}{rrrcl} x & +5y & -z & = & x0 \\ -x & -2y & +az & = & y0 \\ & y & +az & = & z0 \end{array}$$

On considère l'application linéaire suivante, de paramètre *a*.

$$\begin{array}{rcl} f : \mathbb{R}^3 & \longrightarrow & \mathbb{R}^3 \\ (x, y, z) & \longmapsto & (x + 5y - z, -x - 2y + a*z, y + a*z) \end{array}$$

1) Questions préliminaires
a) Ecrire en *Mathematica* la fonction *f*

```
f[x_, y_, z_] := {x + 5 * y - z, -x - 2 * y + a * z, y + a * z}
```

b) Ecrire en *Mathematica* la matrice *M* associée à *f*

```
M = {{1, 5, -1}, {-1, -2, a}, {0, 1, a}};
M // MatrixForm
```

$$\begin{pmatrix} 1 & 5 & -1 \\ -1 & -2 & a \\ 0 & 1 & a \end{pmatrix}$$

2) Dans toute cette partie, on fixera *a=-1/2*

```
a = -1 / 2;
```

a) Etude du noyau
i) Trouver une base du noyau

```
NullSpace[M]
```

$\left\{\left\{-\frac{3}{2}, \frac{1}{2}, 1\right\}\right\}$

ii) Trouver un système d'équations minimal du noyau

```
Reduce[f[x, y, z] == 0, {x, y, z}]
```

$y == -\frac{x}{3} \;\&\&\; z == -\frac{2\,x}{3}$

b) Etude de l'image
i) Trouver un système d'équations minimal de l'image

```
Eliminate[f[x, y, z] == {x0, y0, z0}, {x, y, z}]
```

$-y0 + 3\,z0 == x0$

```
Reduce[f[x, y, z] == {x0, y0, z0}, {x, y, z}]
```

$x0 == -y0 + 3\,z0 \;\&\&\; y == -\frac{x}{3} - \frac{y0}{3} + \frac{z0}{3} \;\&\&\; z == -\frac{2\,x}{3} - \frac{2\,y0}{3} - \frac{4\,z0}{3}$

Une équation de l'image est x0+y0-3*z0==0

ii) f est-elle surjective ?

Non, f n'est pas surjective puisque l'on obtient des condition sur les coordonnées du vecteur image. On pouvait s'en douter, puisque d'après le théorème du rang, la dimension de l'image était 3-1=2 (le noyau est de dimension 1 d'après i))

3) Dans toute cette partie, *a* redeviendra quelconque.
On prendra soin de bien fournir des réponses exhaustives en fonction des différentes valeurs du paramètre.

NB : on pourra utiliser *Clear[a]* pour effacer le contenu de *a*.

```
Clear[a]
```

a) Utilisant *Reduce*, trouver un système d'équations minimales du noyau.

```
Reduce[f[x, y, z] == 0, {x, y, z}]
```

$a == -\frac{1}{2} \;\&\&\; y == -\frac{x}{3} \;\&\&\; z == -\frac{2\,x}{3} \;||\; 1 + 2\,a \neq 0 \;\&\&\; x == 0 \;\&\&\; y == 0 \;\&\&\; z == 0$

Si a=-1/2, l'application linéaire n'est pas injective, donc il n'y a pas unicité de la solution (on retrouve ce qui a été traité en 1))
Si a≠-1/2, la fonction est injective et il y a unicité de la solution lorsqu'elle existe.

b) Trouver un système d'équations minimales de l'image.
En déduire, en fonction de la valeur du paramètre *a*, l'existence de solutions au système pour un *(x0,y0,z0)* donné.


```
Reduce[f[x, y, z] == {x0, y0, z0}, {x, y, z}]
```

$$x0 == -y0 + 3\,z0 \;\&\&\; a == -\frac{1}{2} \;\&\&\; y == \frac{1}{3}\,(-x - y0 + z0) \;\&\&\; z == -\frac{2}{3}\,(x + y0 + 2\,z0) \;||$$
$$1 + 2\,a \neq 0 \;\&\&\; x == \frac{-3\,a\,x0 - y0 - 5\,a\,y0 - 2\,z0 + 5\,a\,z0}{1 + 2\,a} \;\&\&$$
$$y == \frac{1}{3}\,(-x - y0 + z0) \;\&\&\; z == \frac{1}{3}\,(-2\,x - 3\,x0 - 5\,y0 + 5\,z0)$$

Si a=-1/2, l'application linéaire n'est pas surjective. Il n'y a pas de solution au système pour tout vecteur (x0,y0,z0).
Si a≠-1/2, l'application linéaire est surjective et le système admet une solution en (x,y,z) pour tout vecteur (x0,y0,z0).

c) Utilisant les questions 3)a) et 3)b), donner en fonction de *a* l'ensemble solution du système. On prendra garde à inclure les éléments du noyau dans le cas où *f* n'est pas injective.

Si a=-1/2, l'ensemble solution est
$\{(x, \frac{1}{3}(-x - y0 + z0), -\frac{2}{3}(x + y0 + 2\,z0))\ x \in \mathbb{R}\}$ dès lors que x0==-y0+3 z0. C'est ce cas où *f* n'est pas injective et le noyau joue le rôle de paramètre,puisque c'est $\{(x*(1,-1/3,-2/3,)\ x \in \mathbb{R}\}$.
Si a≠-1/2, l'ensemble solution est
$\{(\frac{-3\,a\,x0-y0-5\,a\,y0-2\,z0+5\,a\,z0}{1+2\,a}, \frac{1}{3}(-\frac{-3\,a\,x0-y0-5\,a\,y0-2\,z0+5\,a\,z0}{1+2\,a} - y0 + z0), \frac{1}{3}(-2\,\frac{-3\,a\,x0-y0-5\,a\,y0-2\,z0+5\,a\,z0}{1+2\,a} - 3\,x0 - 5\,y0 + 5\,z0)\}$

Exercices sur les booléens

▪ Tables de vérités

1) Grace à l'aide expliquer en une phrase le sens de la fonction *Implies*

```
? Implies
```

```
Implies[p, q] represents the logical implication p -> q.
```

2) En utilisant *Outer,* fabriquer la table de vérité de *Implies.*

```
Outer[Implies, {True, False}, {True, False}] // TableForm
```

```
True    False
True    True
```

3) On considère maintenant les fonctions *fBool* et *fContra.*

```
fBool[x_, y_] := Or[Not[x], y]
```

```
fContra[x_, y_] := Implies[Not[y], Not[x]]
```

a) Expliquer pourquoi ce sont des fonctions booléennes

b) Donner les tables de vérité de ces deux fonctions. Que peut-on en conclure ?

```
Outer[fBool, {True, False}, {True, False}] // TableForm
```

```
True    False
True    True
```

```
Outer[fContra, {True, False}, {True, False}] // TableForm
```

```
True      False
True      True
```

Comme ces deux fonctions booléennes ont des tables de vérité identiques, elles sont égales.

■ Methode de Monte-Carlo

On se propose dans cet exercice d'approximer $\pi/4$ par une méthode probabiliste.
On considère pour cela la région, notée *A,* comprise entre le cercle de rayon *1* et de centre *O*, l'axe *(Ox)* et l'axe *(Oy)*.

1) Créer une fonction qui, aux coordonnées d'un point du plan, associe *True* ou *False* selon que le point appartienne à la région *A* ou non.

```
regA[{x_, y_}] := And[x^2 + y^2 ≤ 1, x > 0, y > 0]
```

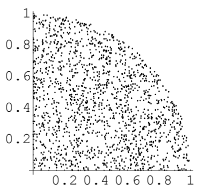
2) Générer une liste *pts* de *2000* points aléatoires pris dans le carré *C=[0,1]x[0,1]*.

```
pts = Table[Random[Real, {0, 1}], {i, 1, 2000}, {j, 1, 2}];
```

3) Sélectionner parmi ces points ceux qui sont dans *A*. On nommera *ptsInA* cette nouvelle liste de coordonnées. Afficher graphiquement cet ensemble de points.

```
ptsInA = Select[pts, regA];
```

```
ListPlot[ptsInA, AspectRatio → Automatic];
```



4) Utilisant *pts* et *ptsInA*, calculer la probabilité *prob* qu'un des points de *pts* tiré soit dans la région *A*. On en donnera une approximation.

```
prob = N[Length[ptsInA] / Length[pts]]
```

```
0.7885
```

5) En utilisant les aires des régions *A* et *C,* calculer de manière théorique la probabilité *prob.*
En déduire une valeur approchée de $\pi/4$.

```
prob = Pi / 4
```

```
N[Pi / 4]
```

```
0.785398
```