TP 4 : Espaces euclidiens

```
> restart;with(linalg):
Warning, the protected names norm and trace have been redefined and unprotected
```

1 Orthogonalisation

Dans R^3, avec la fonction "clé en main"

```
> v1:=vector([1,2,-3]):v2:=vector([-1,2,-1]):v3:=vector([1,2
,3]):
> GramSchmidt([v1,v2,v3]);
```

$$\left[[1, 2, -3], \left[\frac{-10}{7}, \frac{8}{7}, \frac{2}{7}\right], [2, 2, 2]\right]$$

Orthogonalisation générale

```
> orthogonalisation:=proc(e,scal)
  local f,i;
  f:=e:
  for i from 2 to nops(e) do
  f[i]:=f[i]-add(scal(f[i],f[k])/scal(f[k],f[k])*f[k],k=1..i
  -1) od:
  RETURN(map(evalm,f))
  end:
> orthogonalisation([v1,v2,v3],dotprod);
```

$$\left[[1, 2, -3], \left[\frac{-10}{7}, \frac{8}{7}, \frac{2}{7}\right], [2, 2, 2]\right]$$

Dans Rn[X]

```
> scal1:=(P,Q)->int(P*Q,X=-1..1):
```

On a privilégié le point de vue "expression" par rapport au point de vue "fonction". Attention, les polynomes en jeu doivent absolument avoir X comme indéterminée, et pas t ou x...

```
> scal1(X,X^3);
```

$$\frac{2}{5}$$

```
> base_can:=[seq(X^k,k=0..6)];
```

$$base\_can := [1, X, X^2, X^3, X^4, X^5, X^6]$$

```
> base_orth:=orthogonalisation(base_can,scal1);
```

$base\_orth :=$

$$\left[1, X, X^2 - \frac{1}{3}, X^3 - \frac{3}{5}X, X^4 + \frac{3}{35} - \frac{6}{7}X^2, X^5 + \frac{5}{21}X - \frac{10}{9}X^3, X^6 - \frac{5}{231} + \frac{5}{11}X^2 - \frac{15}{11}X^4\right]$$

Vous reconnaissez les 4 premiers ?

Lien avec les polynomes de Legendre

```
> with(orthopoly);
```

$$[G, H, L, P, T, U]$$

P(n, x) generates the nth Legendre polynomial.

```
> seq(P(n,X),n=0..6);
```

$$1, X, \frac{3}{2}X^2-\frac{1}{2}, \frac{5}{2}X^3-\frac{3}{2}X, \frac{35}{8}X^4-\frac{15}{4}X^2+\frac{3}{8}, \frac{63}{8}X^5-\frac{35}{4}X^3+\frac{15}{8}X,$$
$$\frac{231}{16}X^6-\frac{315}{16}X^4+\frac{105}{16}X^2-\frac{5}{16}$$

```
> seq(rem(base_orth[k],P(k-1,X),X),k=1..7);
```

$$0, 0, 0, 0, 0, 0, 0$$

The **rem** function returns the remainder of **a** divided by **b**. The **quo** function returns the quotient of **a** divided by **b**. The remainder **r** and quotient **q** satisfy: **a = b*q + r** where **degree(r,x) < degree(b,x)**

```
> restart;with(linalg):
Warning, the protected names norm and trace have been redefined and unprotected
```

2 Identification de réflexions/projections orthogonales

```
> A:=matrix([[1/3, 2/3, -2/3], [2/3, 1/3, 2/3], [-2/3, 2/3, 1/3]]):B:=matrix([[13/14, -1/7, -3/14], [-1/7, 5/7, -3/7], [-3/14, -3/7, 5/14]]):
```

Un premier exemple

```
> evalm(A^2),evalm(transpose(A)-A);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

```
> kernel(A-Matrix(3,3,shape=identity)),kernel(A+Matrix(3,3,shape=identity));
```

$$\{[-1, 0, 1], [1, 1, 0]\}, \{[1, -1, 1]\}$$

Autre possibilité, quand on a une réflexion, l'axe est l'orthogonal du plan. On peut donc trouver une base de l'axe par produit vectoriel des vecteurs d'une base du plan :

```
> plan:=kernel(A-Matrix(3,3,shape=identity)):crossprod(plan[1],plan[2]);
```

$$[1, -1, 1]$$

Systématisons

```
> evalb(transpose(A)=A);
```

$$\mathit{false}$$

Arf !

```
> nops(kernel(A-transpose(A)));
```

$$3$$

Léger...

```
> analyse_reflexion:=A->
  if nops(kernel(transpose(A)-A))<3
  or nops(kernel(A^2-Matrix(3,3,shape=identity)))<3
  or nops(kernel(A-Matrix(3,3,shape=identity)))<>2
  then false
  else op(kernel(A+Matrix(3,3,shape=identity))) fi:
> analyse_reflexion(A);analyse_reflexion(B);
```

$$[1, -1, 1]$$

$$\mathit{false}$$

Cas des projections orthogonales

```
> analyse_projection:=A->
  if nops(kernel(transpose(A)-A))<3 or nops(kernel(A^2-A))<3
  then false
  else kernel(A-Matrix(3,3,shape=identity)) fi:
```

Exemples

```
> C:=matrix([[25/146, 33/146, -22/73], [33/146, 137/146,
  6/73], [-22/73, 6/73, 65/73]]):Dd:=matrix([[12/13, -3/13,
  4/13], [-3/13, 4/13, 12/13], [4/13, 12/13,
  -3/13]]):E:=matrix([[-10/19, 6/19, 15/19], [6/19, -15/19,
  10/19], [15/19, 10/19, 6/19]]):
> analyse_reflexion(C),analyse_projection(C);analyse_reflexi
  on(Dd),analyse_projection(Dd);analyse_reflexion(E),analyse
  _projection(E);
```

$$\mathit{false}, \left\{\left[0, \frac{4}{3}, 1\right], \left[1, \frac{11}{3}, 0\right]\right\}$$

$$\left[\frac{1}{3}, 1, \frac{-4}{3}\right], \mathit{false}$$

$$\mathit{false}, \mathit{false}$$

```
> evalm(E-transpose(E)),evalm(E^2-Matrix(3,3,shape=identity)
  );
```

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

```
> kernel(E-Matrix(3,3,shape=identity)),kernel(E+Matrix(3,3,s
  hape=identity));
```

$$\left\{\left[\frac{3}{2}, 1, \frac{5}{2}\right]\right\}, \left\{\left[0, \frac{-5}{2}, 1\right], \left[1, \frac{-3}{2}, 0\right]\right\}$$

E est donc la matrice d’une symétrie orthogonale par rapport à une droite (on parle de retournement)

```
> restart;with(linalg):
```

3 Création de matrices de réflexion/projection orthogonale

Un premier exemple

```
> v0:=vector([1,-1,1]):v:=vector([x,y,z]):evalm(v-2*dotprod(
  v,v0)/3*v0);
```

$$\left[\frac{1}{3}x + \frac{2}{3}y - \frac{2}{3}z, \frac{1}{3}y + \frac{2}{3}x + \frac{2}{3}z, \frac{1}{3}z - \frac{2}{3}x + \frac{2}{3}y\right]$$

```
> genmatrix(convert(%,list),[x,y,z]);
```

$$\begin{bmatrix} \frac{1}{3} & \frac{2}{3} & \frac{-2}{3} \\ \frac{2}{3} & \frac{1}{3} & \frac{2}{3} \\ \frac{-2}{3} & \frac{2}{3} & \frac{1}{3} \end{bmatrix}$$

Systématisons

```
> generateur_reflexion:=proc(v0)
  local v,x,y,z;
  v:=vector([x,y,z]):
  RETURN(genmatrix(convert(evalm(v-2*dotprod(v,v0)/dotprod(v0,v0)*v0),list),[x,y,z]))
  end:
> generateur_reflexion(vector([1,-1,1]));
```

$$\begin{bmatrix} \frac{1}{3} & \frac{2}{3} & \frac{-2}{3} \\ \frac{2}{3} & \frac{1}{3} & \frac{2}{3} \\ \frac{-2}{3} & \frac{2}{3} & \frac{1}{3} \end{bmatrix}$$

Exemples

```
> generateur_reflexion(vector([1,2,3])),generateur_reflexion(vector([1,0,1]));
```

$$\begin{bmatrix} \frac{6}{7} & \frac{-2}{7} & \frac{-3}{7} \\ \frac{-2}{7} & \frac{3}{7} & \frac{-6}{7} \\ \frac{-3}{7} & \frac{-6}{7} & \frac{-2}{7} \end{bmatrix}, \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$

Projections

```
> generateur_projection:=proc(v0)
  local v,x,y,z;
  v:=vector([x,y,z]):
  RETURN(genmatrix(convert(evalm(v-dotprod(v,v0)/dotprod(v0,v0)*v0),list),[x,y,z]))
  end:
> generateur_projection(vector([1,-1,1]));
```

$$\begin{bmatrix} \frac{2}{3} & \frac{1}{3} & \frac{-1}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{-1}{3} & \frac{1}{3} & \frac{2}{3} \end{bmatrix}$$

```
> generateur_projection(vector([1,2,3])),generateur_projection(vector([1,0,1]));
```

$$\begin{bmatrix} \frac{13}{14} & \frac{-1}{7} & \frac{-3}{14} \\ \frac{-1}{7} & \frac{5}{7} & \frac{-3}{7} \\ \frac{-3}{14} & \frac{-3}{7} & \frac{5}{14} \end{bmatrix}, \begin{bmatrix} \frac{1}{2} & 0 & \frac{-1}{2} \\ 0 & 1 & 0 \\ \frac{-1}{2} & 0 & \frac{1}{2} \end{bmatrix}$$