Correction de la séance du 30 octobre

Exercice 1. *Les inversibles de $\mathbb{Z}/n\mathbb{Z}$*

Écrire une procédure `Inv( , )` *ayant comme arguments deux entiers non nuls k et $n > 1$ et affichant l'inverse de k dans $\mathbb{Z}/n\mathbb{Z}$ si $k \in \mathcal{I}_n$*

Correction.

Pour déterminer l'inverse, on pouvait écrire l'algorithme étendu d'Euclide donnant les coefficients de Bezout. L'algorithme peut se retrouver en utilisant le tableau suivant :

$\boldsymbol{r}$	$\boldsymbol{u}$	$\boldsymbol{v}$
a	*1*	*0*
b	*0*	*1*
$\dots$	$\dots$	$\dots$
r_{i-1}	u_{i-1}	v_{i-1}
r_i	u_i	v_i
r_{i+1}	u_{i+1}	v_{i+1}
$\dots$	$\dots$	$\dots$
r_n	u	v
0		

A chaque ligne, on doit avoir $r_i = au_i + bv_i$, ce qui permet d'écrire les deux premières lignes. En appelant q_i le quotient entier de $\dfrac{r_{i-1}}{r_i}$ on a les formules de récurrence suivantes :

$$\begin{cases} r_{i+1} = r_{i-1} - q_i r_i \\ u_{i+1} = u_{i-1} - q_i u_i \\ v_{i+1} = v_{i-1} - q_i v_i \end{cases}$$

La récurrence d'ordre 2 nécessite d'utiliser 3 *indices pour effectuer les calculs. D'où l'écriture de l'algorithme avec des listes tel que celui-ci proposé par Xcas dans* `Aide\Exemples\arit\bezout.xws`

```
bezout(a,b):={
//renvoie la liste [u,v,d] telle que a*u+b*v=d=pgcd(a,b) (fct iterative)
  local la,lb,lr,q,lb2;
  la:=[1,0,eval(a)];
  lb:=[0,1,eval(b)];
  lb2:=eval(b);
  while (lb2 !=0){
    q:=iquo(la[2],lb2);
    lr:=la+(-q)*lb;
    la:=lb;
    lb:=lr;
    lb2:=lb[2];
  }
  return(la);
}:;
```

Exercice 2.

Résoudre les systèmes

$$\begin{cases} x+2y+3z-2t &= 6 \\ 2x-y-2z-3t &= 8 \\ 3x+2y-z+2t &= 4 \\ 2x-3y+2z+t &= -8 \end{cases}$$

$$\begin{cases} x+2y+3z-2t &= -4 \\ 2x-y-2z-3t &= 1 \\ 3x+2y-z+2t &= 6 \\ 2x-3y+2z+t &= 1 \end{cases}$$

Correction.

Il y a plusieurs méthodes de résolution. On écrit le système sous la forme $MX = Bi$ avec des notations évidentes.

1. *On calcule $X := M^{-1}Bi$ Il se peut que la matrice ne soit pas inversible...*
2. *On utilise la fonction* `simult(M|B1|B2)` *où $M|B1|B2$ est la matrice M bordée avec $B1$ puis $B2$ construite avec* `concat`. *On a en un calcul les solutions des deux systèmes (ou plus) mais le système doit être carré.*
3. *On utilise la fonction* `rref(M|B1)`. *Le résultat est une matrice diagonale identité plus une dernière colonne qui donne les solutions.* `rref` *utilisant l'algorithme de Gauss permet de connaître les conditions de compatibilité de systèmes non carrés.*
4. *On calcule le noyau de $M|B1$. S'il y a une solution au système alors la dernière colonne de $M|B1$ est combinaison linéaire des colonnes de M. Dans $\ker(M|B1)$, il doit donc y avoir un -1 dans la dernière colonne d'une des lignes. Cette ligne donne la solution du système (quel que soit la taille du système).*
5. *On utilise la fonction* `linsolve([eq1,eq2..,eqn],[x1,x2,..,xp])` *Cette fonction spécialement dédiée aux système linéaires donne les solutions quelle que soit la taille. Il faut par contre écrire une liste d'équation et une liste d'inconnue.*

Exercice 3.

Diagonaliser dans une base orthonormale la matrice :

$$\begin{pmatrix} 1 & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & 1 & 0 \\ -\frac{1}{\sqrt{2}} & 0 & 1 \end{pmatrix}$$

Correction.

Les fonctions `det pcar pmin eigenvalues eigenvectors normalize` *permettent de répondre plus ou moins rapidement à la question. On peut aussi utiliser* `jordan(M)` *qui renvoie la matrice semblable à M "la plus simple" (avec la matrice de passage) c'est à dire :*

- *soit une matrice diagonale si M est diagonalisable*
- *soit une matrice dont la diagonale* par bloc *est constituée d'homothétie ou de matrice de Jordan $J(\lambda)$*

Ici `jordan(M)` *renvoie les matrices* $P := \begin{pmatrix} 0 & 1 & 1 \\ -1 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ 1 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix}$ *et* $M := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix}$

La matrice de départ étant symétrique, les espaces propres sont orthogonaux. Il suffit ensuite d'utiliser la fonction `normalize` *sur les vecteurs obtenus à partir de* ${}^tP$. *Par exemple, avec* `P:=jordan(M)[0]`, `seq(simplify(normalize(tran(P)[k])),k=0..2)` *renvoie les 3 vecteurs propres normalisés sous une forme lisible.*