Université Claude Bernard Lyon 1 - printemps 2010
Licence Sciences, Technologies, Santé - mention mathématiques

UE Math III Algèbre - MAT2002L

PLANCHE DE TRAVAUX APPLIQUÉS I
- PRISE EN MAIN DE L'ENVIRONNEMENT SAGE -

L'objectif des travaux appliqués dans cette UE est d'illustrer, d'expérimenter et d'appliquer certains résultats du cours sur la réduction des matrices et des endomorphismes. Nous utiliserons pour cela le système de calcul SAGE.

Le premier objectif de cette séance est de prendre en main le système de calcul, en particulier sa syntaxe, ses mécanismes d'affectation et d'évaluation, l'autre objectif est de découvrir les opérations élémentaires de Sage sur les vecteurs et matrices.

1. LE SYSTÈME DE CALCUL SAGE

Sage est un système de calcul formel et numérique dont le développement a commencé en 2005 à l'université de Washington[1]. Sage est construit au dessus de systèmes libres déjà existants tels que MAXIMA et SYMPY pour le calcul symbolique, GAP pour la théorie des groupes, PARI pour la théorie des nombres, SINGULAR pour l'algèbre commutative, SCYPY pour le calcul numérique, R pour les statistiques. Sage a pour objectif de fournir une alternative libre aux systèmes propriétaires tels que MAPLE, MAGMA, MATLAB.

Un point fort, outre la mise en commun des potentialités de tous ces systèmes, est l'utilisation, au lieu d'une multitude de langages spécifiques, d'un langage informatique universel Python[2] comme langage fédérateur. Ainsi les structures mathématiques sont implémentées dans un cadre catégorique et orienté objets avec des méthodes pour les objets structurés et des méthodes pour leurs éléments. Les classes ainsi définies sont regroupées dans des modules Python.

Il existe plusieurs façons d'utiliser Sage : en « ligne de commande » , en écrivant des programmes interprétés ou compilés en Sage, en écrivant des scrits Python qui font appels à la bibiothèque Sage. Nous utiliserons ici Sage par l'intermédiaire d'une « interface graphique » permettant d'éditer des « feuilles de travail » .

– Connexion au serveur Sage – La connection au serveur Sage passe par une identification via la page à l'adresse `http://sage-math.univ-lyon1.fr`. Saisir cette url dans votre navigateur web, puis saisir votre login référérencé par l'annuaire ldap de l'UCBL. Après cette première identification, vous êtes connecté à un serveur Sage qui demande une identification : saisir le même login et le mot de passe associé. Vous accédez ainsi à une interface de gestion de « feuilles de travail » .

– Première feuille de travail – Pour ouvir une nouvelle feuille de travail, cliquer sur le lien New Worksheet. La feuille présente différentes fonctionnalités, en particulier :

- une zone de menus : File, Action, Data, sage, permettant de gérer la feuille de travail, en particulier une entrée du menu File permet d'enregister la feuille de travail dans un fichier (le serveur Sage de l'UCBL étant en phase expérimentale, il est conseillé d'enregistrer ses feuilles de travail à l'issue de la séance),
- trois boutons Save, Save & quit, Discard & quit, permettant d'enregistrer la feuille de travail et de quitter la feuille de travail en l'enregistrant ou non.
- des cellules pour la saisie des commandes.

Dans un premier temps, nous n'utiliserons pas les autres fonctionnalités disponibles.

– Les cellules – Les cellules permettent de saisir les instructions. Pour évaluer l'ensemble des instructions saisies dans une cellule, on peut cliquer sur le lien evaluate au-dessous de la cellule ou bien utiliser le racourci clavier `<MAJ><ENTREE>`.

➢ Saisir l'expression suivante puis l'évaluer.

[1] `http://www.sagemath.org/`
[2] `http://www.python.org/`

```
sage: 2+2
```

Il est possible d'évaluer séquentiellement toutes les cellules de la feuille de travail avec l'entrée Evaluate All du menu Action.

Pour insérer une cellule (entre deux cellules), cliquer sur la ligne bleu au-dessus ou au-dessous de la cellule ou bien faire <CTRL><ENTREE> au clavier. Pour supprimer une cellule, vider la cellule de son contenu puis <DELETE>.

– Aide en ligne et complétion dynamique – Une aide en ligne est disponible pour les commandes et les méthodes. Cette aide en ligne permet aussi d'obtenir une description des classes. Pour obtenir de l'aide sur une méthode ou une classe on exécute l'instruction `nomMethode ?` ou `nomClasse ?`. Tester les instructions suivantes :

```
sage : cos?
```

```
sage : RationalField?
```

Pour écrire le nom d'une méthode ou d'une classe, on peut utiliser la complétion dynamique en utilisant la touche <TAB>.

2. VARIABLES ET EXPRESSIONS

– Affectation – Pour affecter à une *variable* une valeur, on utilise le symbole = :

```
sage: x = 2 + 2
```

```
sage: x
4
sage: x = cos(pi/12)
sage: x
1/12*(sqrt(3) + 3)*sqrt(6)
```

L'affectation d'une variable n'affiche pas la valeur de la variable sur la sortie standard. Tester :

```
sage: x = 1 ; x
1
sage: y = 1
...   print y
1
```

Pour un affichage plus visuel on peut utiliser la méthode `show()`.
Saisir l'instruction suivante :

```
sage: x.show()
```

Pour afficher la valeur de variable `x` avec une approximation avec 20 chiffres décimaux :

```
sage: x.n(digits=20)
```

Il est possible de saisir des instructions sur plusieurs lignes, pour passer à la ligne, utiliser la touche <ENTRÉE> :

```
sage : x=2
...    y=x
...    print y
...    z=3 ; t = z+y ; z = z+t
...    print z
2
8
```

Pour réinitialiser les variables, on peut utiliser la méthode `reset()` :

```
sage : x=2
...    print x
...    reset()
...    print x
2
x
```

– Expressions symboliques – Sage permet de manipuler des expressions symboliques. La commande `var` permet de déclarer des variables symboliques.
➢ Saisir les instructions suivantes puis les évaluer.

```
sage : var('x,y')
...    z = cos(x)^2 + sin(y)^2
...    print z
...    z = z(x=y)
...    print z
...    z.simplify_trig()
```

La méthode `subs.expr()`, ou la syntaxe `expression(variable = valeur)`, permet de substituer des valeurs dans une expression symbolique :

```
sage : var('x,y')
...    z = cos(x)^2 + sin(y)^2
...    print z.subs_expr(x==pi/2)
...    print z(y=pi/2)
```

– Fonctions – On peut définir des fonctions de la façon suivante :

```
sage : reset()
...    var('x,y')
...    f(x,y) = x/sin(x) + sqrt(y)
...    f
```

Dans Sage, toute expression mathématique est un objet. Un objet possède des attributs définissant sa structure et des méthodes permettant de modifier sa structure ou de communiquer avec lui. En particulier, les fonctions sont des objets :

```
sage : f.parent()
```

qui possèdent des méthodes d'accès, par exemple :

```
sage : f.show()
...    f.limit(x=0).show()
...    f.diff(x).show()
```

3. MANIPULATION DE MATRICES ET DE VECTEURS

Sage fournit un ensemble de classes et méthodes d'algèbre linéaire. Dans cette partie, nous allons découvrir les principales méthodes permettant de manipuler matrices et vecteurs.

– Construction de vecteurs et matrices – Pour construire le vecteur $\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$ on utilise la méthode `vector()` avec pour argument la liste $[1,2,3]$, l'affichage des vecteurs est horizontal :

```
sage: v = vector([1,2,3])
...   print v
(1,2,3)
```

On accède aux composantes d'un vecteur comme aux éléments d'une liste :

```
sage: v[0] ; v[1] ; v[2]
1
2
3
```

pour la liste des composantes d'un vecteur

```
sage: list(v)
[1, 2, 3]
```

Pour construire la matrice $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ on utilise le constructeur `Matrix()` :

```
sage: A = Matrix([[1,2],[3,4]])
...   A
[1 2]
[3 4]
```

On peut aussi spécifier la taille de la matrice et la liste de ses coefficients :

```
sage: B = Matrix(2,3,[1,2,3,4,5,6])
...   B
[1 2 3]
[4 5 6]
```

```
sage: vars = range(9)
...   C = Matrix(3,3,vars)
...   C
```

On accède aux coefficients d'une matrice comme dans un tableau :

```
sage: B[1,2] ; B[0,0] ; B.list()
6
1
[1, 2, 3, 4, 5, 6]
```

Des méthodes permettent de construire des matrices identités (`identity_matrix`), des matrices nulles (`zero_matrix`), des matrices diagonales (`diagonal_matrix`), voir l'aide en ligne pour l'utilisation de ces méthodes.

➢ Écrire les instructions permettant de définir les matrices suivantes.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}, \qquad \mathbf{V} = \begin{bmatrix} 1 & a & a^2 & a^3 & a^4 \\ 1 & b & b^2 & b^3 & b^4 \\ 1 & c & c^2 & c^3 & c^4 \\ 1 & d & d^2 & d^3 & d^4 \\ 1 & e & e^2 & e^3 & e^4 \end{bmatrix},$$

$$\mathbf{S} = \begin{bmatrix} 1 & a & b \\ a & 2 & c \\ b & c & 3 \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix}.$$

– Opérations sur les matrices et les vecteurs – Sur les vecteurs on a les opérations vectorielles :

```
sage : v = vector([1,2,3]) ; u = vector([3,2,1])
...    2*u - 3*v
(3, -2, -7)
```

Principales opérations sur les matrices :

- `A*B`, `A*v`, `v*A` : produit matriciel, produits matrice par vecteur,
- `A.transpose()` : transposée de la matrice,
- `A^n` : puissance n-ième d'une matrice,
- `inverse()` (ou `A^-1`) : inverse d'une matrice inversible,
- `trace()`, `det()` : trace et déterminant .

➢ Calculer $\mathbf{A}^{50}$.

➢ Montrer que la matrice **B** est *orthogonale*, i.e., qu'elle satisfait la relation

$$\mathbf{B}\,{}^t\mathbf{B} = {}^t\mathbf{B}\mathbf{B} = \mathbf{1}_2.$$

➢ Calculer la trace et le rang de la matrice de Vandermonde **V**, ainsi que son déterminant sous une forme factorisée. Calculer l'inverse de **V**.

➢ Les matrices **S** et **T** sont-elles inversibles ? Si oui, calculer leur inverse.

➢ Pour quelle valeur de a, la matrice suivante

$$\begin{bmatrix} 1 & 1 & a \\ 1 & a & 1 \\ a & 1 & 1 \end{bmatrix}$$

est-elle inversible ?

– Noyaux – La méthode `right_kernel()` noyau permet de calculer le noyau d'une matrice. Attention, la méthode `kernel()` calcule le *noyau gauche*, i.e., le noyau gauche de **A** est ensemble des vecteurs **x** tels que $\mathbf{xA} = \mathbf{0}$. Par exemple, les instructions suivantes permettent de calculer les noyaux gauche et droit de la matrice **B**. C'est ce dernier noyau qui nous interesse.

$$\mathbf{B} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
sage: B = Matrix([[0,1,0,0],[0,0,0,0],[0,0,1,1],[0,0,0,1]])
...   print B
...   print B.kernel()
...   print B.right_kernel()
Free module of degree 4 and rank 1 over Integer Ring
Echelon basis matrix:
[0 1 0 0]
Free module of degree 4 and rank 1 over Integer Ring
Echelon basis matrix:
[1 0 0 0]
```

Le noyau (droit) de **B** est le sous-espace vectoriel engendré par le vecteur $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$.

4. QUELQUES EXERCICES DE DIAGONALISATION

– Généralités – Sage propose plusieurs méthodes sur les matrices permettant de calculer par exemple le polynôme caractéristique d'une matrice `charpoly()`, son polynôme minimal `minpoly()`, ses valeurs propres `eigenvalues()`, une liste de vecteurs propres `eigenvalues_right()`. On pourra regarder l'aide en ligne de ces méthodes qui fournit quantité d'exemples d'utilisation.

➢ Calculer les valeurs propres de la matrice suivante :

$$\begin{bmatrix} a & b & b & b \\ b & a & b & b \\ b & b & a & b \\ b & b & b & a \end{bmatrix}$$

– Matrices diagonalisables –

➢ Calculer le polynôme caractéristique et le polynôme minimal de la matrice suivante :

$$\mathbf{M} = \begin{bmatrix} -1 & 1 & 3 \\ -2 & 2 & 2 \\ -2 & 1 & 4 \end{bmatrix}$$

➢ La matrice **M** est-elle diagonalisable ?

➢ Déterminer le spectre de **M** ainsi qu'une base pour chaque sous-espace propre.

➢ Les matrices suivantes sont-elles diagonalisables sur $\mathbb{Q}$, sur $\mathbb{R}$, sur $\mathbb{C}$?

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 2 \\ 1 & 1 & -1 & 1 \\ 1 & 2 & 0 & -1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 1 & 2 & 3 \\ 1 & 2 & 3 & 2 \\ 2 & 3 & 3 & 4 \\ 3 & 2 & 4 & 4 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 4 & 2 & 4 \\ 0 & 0 & -1 \\ -3 & -2 & -3 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 0 & -2 & 0 \\ 1 & 0 & -1 \\ 0 & 2 & 0 \end{bmatrix}.$$

Pour avoir la liste des valeurs propres dans $\mathbb{C}$ d'une matrice, on pourra calculer son polynôme caractéristique `p(x)`, puis les racines de ce polynôme. Attention aux erreurs d'approximation numérique. Pour afficher ces racines avec une approximation numérique avec 3 chiffres décimaux, on pourra utiliser le code suivant (cf. manuel en ligne de `solve`) :

```
valspres = solve([p(x) == 0], x, solution_dict = True)
for vp in valspres:
    print vp[x].n(digits=3)
```

On considère la matrice réelle suivante

$$\mathbf{A} = \begin{bmatrix} 1 & a^2 & a^2 & a \\ 1 & 1 & a & 1 \\ 1 & a & 1 & 1 \\ a & a^2 & a^2 & 1 \end{bmatrix}.$$

➢ Calculer le polynôme caractéristique de **A**.

➢ Pour quelles valeurs de a, la matrice **A** est-elle diagonalisable ?

➢ Diagonaliser **A** en donnant une matrice de passage.

– Valeurs propres et vecteurs propres – On considère les trois matrices réelles suivantes

$$\mathbf{A} = \begin{bmatrix} a & 1 & 1 & 1 \\ 1 & b & 1 & 1 \\ 1 & 1 & c & 1 \\ d & 1 & 1 & d \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} -1 & a & d \\ b & e & -1 \\ c & 1 & f \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} a & 2 & 1 \\ 2 & b & -1 \\ -1 & 1 & c \end{bmatrix}.$$

➢ Déterminer les quadruplets $(a, b, c, d) \in \mathbb{R}^4$, pour lesquels la matrice $\mathbf{A}$ admet $\begin{bmatrix} -1 \\ 2 \\ 2 \\ -1 \end{bmatrix}$ pour vecteur propre.

On pourra utiliser la fonction `solve`.

➢ Déterminer les réels a, b, c, d, e, f, pour lesquels la matrice $\mathbf{B}$ admet $\begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$ pour vecteurs propres.

➢ Déterminer les triplets $(a, b, c) \in \mathbb{R}^3$, pour lesquels la matrice $\mathbf{C}$ admet 1, a et b comme valeurs propres.

5. CALCUL DES PROJECTEURS SPECTRAUX

Soit $\mathbf{A}$ une matrice de $\mathcal{M}_n(\mathbb{K})$. D'après le thèorème de décomposition en noyaux, si P_1 et P_2 sont deux polynômes premiers entre eux et tels que leur produit P_1P_2 soit annulateur de $\mathbf{A}$, alors on a une décomposition de $\mathbb{K}^n$ en somme directe :

$$\mathbb{K}^n = \text{Ker } P_1(\mathbf{A}) \oplus \text{Ker } P_2(\mathbf{A}).$$

L'objectif de cet exercice est de calculer les projections de $\mathbb{K}^n$ sur les sous-espaces Ker $P_1(\mathbf{A})$ et Ker $P_2(\mathbf{A})$. On va pour cela exprimer les matrices de ces projections en fonction de la matrice $\mathbf{A}$.

D'après le théorème de Bézout, si P_1 et P_2 sont premiers entre eux, il existe des polynômes U et V tels que

$$P_1U + P_2V = 1.$$

Pour construire des polynômes à coefficients rationnels en une indéterminée :

```
sage: R = PolynomialRing(QQ,'x')
...   R
Univariate Polynomial Ring in x over Rational Field
```

```
sage: p = x^3 + 2*x^2 + x + 1
...   print p in R
...   A = Matrix([[1,1,0],[0,1,0],[0,0,2]])
...   print A.charpoly() in R
True
True
```

La méthode `gcd()` retourne le pgcd de deux polynômes :

```
sage : x = PolynomialRing(QQ, 'x').gen()
...    f = (x-1)*(x+2)
...    g = x+1
...    f.gcd(g)
1
```

La méthode `xgcd` permet de calculer les coefficients de Bézout pour un couple de polynômes :

```
sage: d,u,v = xgcd(f,g)
...   print d,u,v
...   print f*u+g*v == d
-2 1 -x
True
```

➢ Exprimer en fonction de P_1, P_2, U et V la matrice Π_1 de la projection de la $\mathbb{K}^n$ sur le noyau Ker $P_1(\mathbf{A})$ parallèlement à Ker $P_2(\mathbf{A})$) et la matrice Π_2 de la projection de la $\mathbb{K}^n$ sur le noyau Ker $P_2(\mathbf{A})$ parallèlement à Ker $P_1(\mathbf{A})$).
➢ On considère la matrice suivante vue en travaux dirigés

$$\mathbf{B} = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}.$$

➢ Calculer le polynôme caractéristique et le polynôme minimal de **B**.
➢ Notons λ_1 et λ_2 les deux valeurs propres de **B**. Déterminer les sous-espaces propres E_{λ_1} et E_{λ_2}.
➢ Calculer les matrices de la projections de $\mathbb{R}^3$ sur le sous-espace E_{λ_1} parallèlement à E_{λ_2} et de la projection de $\mathbb{R}^3$ sur le sous-espace E_{λ_2} parallèlement à E_{λ_1}. Ces projections sont appelées les *projecteurs spectraux* de **B**.
➢ Calculer les projecteurs spectraux des matrices suivantes rencontrées aussi cours des travaux dirigés

$$\mathbf{A} = \begin{bmatrix} 5 & 1 & 3 \\ 4 & 3 & 4 \\ -1 & -1 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} -1 & 1 & 3 \\ -2 & 2 & 2 \\ -2 & 1 & 4 \end{bmatrix}.$$