Algèbre linéaire

Appeler le package : >with(**LinearAlgebra**) ;

1. **Définir une matrice :**
   - >*a* := ***Matrix(2,3)*** ;(resp.>*a* := ***Matrix(2,3,symbol=b)*** ;)
     crée une matrice à 2 lignes et 3 colonnes de terme général 0 (resp.$b[i,j]$).
   - En donnant la liste des lignes :
     >*a* := ***Matrix(2,3,[[1,0,-2],[5,74,0]]) ;*** ou >*a* := ***Matrix(2,3,[1,0,-2,5,74,0]) ;***
   - En donnant la liste de ses vecteurs-colonnes :
     >*V1* := ***Vector([1,5])*** ; *V2* := ***Vector([0,74])*** ; *V3* := ***Vector([-2,0])***
     >*a* := ***Matrix([V1,V2,V3]) ;***
   - En définissant le terme général par une fonction :
     >*f :=(i,j)->3*i-sin(j) ;*
     >*a* := ***Matrix(2,3,f) ;***
     En particulier, avec une fonction constante :
     >*a :=Matrix(2,3,5) ;* donne la matrice à 2 lignes 3 colonnes de terme général 5.
   - cf. aussi " ***DiagonalMatrix***" pour définir une matrice diagonale, " ***Matrix(3,shape=identity)***" pour définir la matrice I_3.
2. **Travailler avec un vecteur, une matrice :**
   - Récupérer un élément/une colonne/une ligne :
     > ***a[1,3],Column[a,2], Row[a,1] ;***
   - Modifier un élément :
     >*a[1,3] :=27 ;*
   - Donner une valeur à un paramètre :
     >*a1 :=**subs**(t=2,a) ;*
   - Faire des opérations :
     > ***c :=a+b*** ;
     > ***aa :=a*** $\wedge(-1)$;
     > ***d :=a.b ;***
     a et b sont des matrices ; pas de "$\star$" pour le produit matriciel, mais un point.
     > ***y :=a.x ;***
     a est une matrice, x et y des vecteurs : un point pour le produit.
     > ***Transpose(a)***
   - Utiliser " map" :
     >*a1* := ***map****(sin,a) ;*
3. **Résoudre des équations :**
   - Résolution d'un système linéaire $AX = B$
     Utiliser " ***LinearSolve***"
   - Résolution d'une équation matricielle se ramenant à $A = O_{n,p}$:
     >*syst* := ***convert****=(a,* ***set****) ;*
     >*sol :=solve(syst) ;*
     >*newa :=subs(sol,a) ;*
4. **Calcul vectoriel :**
   - ***Norm(..,2)*** *(norme euclidienne)*
     ... ou d'autres normes (cf. help)
   - ***Dotproduct*** (produit scalaire hermitien)
   - ***Crossproduct*** (produit vectoriel)

5. **Fonctions usuelles :**

   - ***Determinant(a)***
   - ***CharacteristicPolynomial(a,x)*** :
     det(xI-a) (le polynôme caractéristique anglo-saxon)
   - ***Eigenvalues(a)*** (resp.Eigenvalues(a,output=list)) :
     $\begin{bmatrix}\lambda_1\\ \vdots \\ \lambda_p\end{bmatrix}$ (resp. idem en ligne)
   - ***r :=Eigenvectors(a)*** :
     La sequence r[1],r[2] où r[1]=$\begin{bmatrix}\lambda_1\\ \vdots \\ \lambda_p\end{bmatrix}$ et r[2]=$\begin{bmatrix}V_1 & \ldots & V_p\end{bmatrix}$, $[V1,...,Vp]$ étant une base de vecteurs propres si a est diagonalisable ; sinon, c'est une famille libre maximale de vecteurs propres complétée par des 0_p.
   - ***JordanForm(a)*** (resp. JordanForm(a,output='Q'))
     une réduite de Jordan (resp. la matrice de passage)
   - ***Nullspace(a)*** une base du noyau
   - Orthogonaliser/orthonormaliser :
     $>$ ***GramSchmidt([V1,V2,V3]) ;***
     $[V1, V2, V3]$ est une liste de vecteurs : on obtient une famille orthoGONALE
     $>$ ***GramSchmidt([V1,V2,V3],normalized) ;***
     On obtient une famille orthoNORMALE.
     En particulier, on peut transformer une matrice de diagonalisation en matrice orthogonale.