Chap. VI : Matrices

Laurent Poinsot

18 janvier 2009

Plan

Plan

Plan

Plan

Plan

Plan

Plan

Plan

Plan

Plan

Pour manipuler des matrices et des vecteurs et résoudre des problèmes d'algèbre linéaire avec `Maple`, il est nécessaire de faire appel à des commandes particulières, contenues dans le paquetage `LinearAlgebra` ou le paquetage `linalg` (il s'agit d'une librairie plus ancienne). Dans le paquetage `LinearAlgebra`, les matrices et les vecteurs sont des objets d'un type plus général, le type `Array`. Toutes les commandes de ce paquetage commencent par une majuscule.

Pour manipuler des matrices et des vecteurs et résoudre des problèmes d'algèbre linéaire avec `Maple`, il est nécessaire de faire appel à des commandes particulières, contenues dans le paquetage `LinearAlgebra` ou le paquetage `linalg` (il s'agit d'une librairie plus ancienne). Dans le paquetage `LinearAlgebra`, les matrices et les vecteurs sont des objets d'un type plus général, le type `Array`. Toutes les commandes de ce paquetage commencent par une majuscule.

Pour manipuler des matrices et des vecteurs et résoudre des problèmes d'algèbre linéaire avec `Maple`, il est nécessaire de faire appel à des commandes particulières, contenues dans le paquetage `LinearAlgebra` ou le paquetage `linalg` (il s'agit d'une librairie plus ancienne).

Dans le paquetage `LinearAlgebra`, les matrices et les vecteurs sont des objets d'un type plus général, le type `Array`. Toutes les commandes de ce paquetage commencent par une majuscule.

Pour manipuler des matrices et des vecteurs et résoudre des problèmes d'algèbre linéaire avec `Maple`, il est nécessaire de faire appel à des commandes particulières, contenues dans le paquetage `LinearAlgebra` ou le paquetage `linalg` (il s'agit d'une librairie plus ancienne). Dans le paquetage `LinearAlgebra`, les matrices et les vecteurs sont des objets d'un type plus général, le type `Array`. Toutes les commandes de ce paquetage commencent par une majuscule.

Pour manipuler des matrices et des vecteurs et résoudre des problèmes d'algèbre linéaire avec `Maple`, il est nécessaire de faire appel à des commandes particulières, contenues dans le paquetage `LinearAlgebra` ou le paquetage `linalg` (il s'agit d'une librairie plus ancienne). Dans le paquetage `LinearAlgebra`, les matrices et les vecteurs sont des objets d'un type plus général, le type `Array`. Toutes les commandes de ce paquetage commencent par une majuscule.

Pour manipuler des matrices et des vecteurs et résoudre des problèmes d'algèbre linéaire avec `Maple`, il est nécessaire de faire appel à des commandes particulières, contenues dans le paquetage `LinearAlgebra` ou le paquetage `linalg` (il s'agit d'une librairie plus ancienne). Dans le paquetage `LinearAlgebra`, les matrices et les vecteurs sont des objets d'un type plus général, le type `Array`. Toutes les commandes de ce paquetage commencent par une majuscule.

Pour charger le paquetage `LinearAlgebra`, on commence par :

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

Toute exécution ultérieure d'une ligne "`restart;`" effacera de la mémoire les commandes d'algèbre linéaire. Une nouvelle ligne "`with(LinearAlgebra) :`" sera alors nécessaire.

Pour charger le paquetage `LinearAlgebra`, on commence par :

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

Toute exécution ultérieure d'une ligne "`restart;`" effacera de la mémoire les commandes d'algèbre linéaire. Une nouvelle ligne "`with(LinearAlgebra) :`" sera alors nécessaire.

Pour charger le paquetage `LinearAlgebra`, on commence par :

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

Toute exécution ultérieure d'une ligne "`restart;`" effacera de la mémoire les commandes d'algèbre linéaire. Une nouvelle ligne "`with(LinearAlgebra) :`" sera alors nécessaire.

Pour charger le paquetage `LinearAlgebra`, on commence par :

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

Toute exécution ultérieure d'une ligne "`restart;`" effacera de la mémoire les commandes d'algèbre linéaire. Une nouvelle ligne "`with(LinearAlgebra) :`" sera alors nécessaire.

Pour charger le paquetage `LinearAlgebra`, on commence par :

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

Toute exécution ultérieure d'une ligne "`restart;`" effacera de la mémoire les commandes d'algèbre linéaire. Une nouvelle ligne "`with(LinearAlgebra) :`" sera alors nécessaire.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :

”|” pour séparer les colonnes ;

”,” pour séparer les lignes.

En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”<” et ”>”.

Enfin, une matrice ou un tableau à deux dimensions, composé d'une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”<” et ”>”.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :

"|" pour séparer les colonnes ;

"," pour séparer les lignes.

En outre, chaque ligne ou colonne est entourée par des crochets anguleux "$<$" et "$>$".

Enfin, une matrice ou un tableau à deux dimensions, composé d'une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux "$<$" et "$>$".

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :
”|” pour séparer les colonnes ;
”,” pour séparer les lignes.
En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”<” et ”>”.
Enfin, une matrice ou un tableau à deux dimensions, composé d'une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”<” et ”>”.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :
”|” pour séparer les colonnes ;
”,” pour séparer les lignes.
En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”<” et ”>”.
Enfin, une matrice ou un tableau à deux dimensions, composé d’une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”<” et ”>”.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :
”|” pour séparer les colonnes ;
”,” pour séparer les lignes.
En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”$<$” et ”$>$”.
Enfin, une matrice ou un tableau à deux dimensions, composé d'une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”$<$” et ”$>$”.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :
”$|$” pour séparer les colonnes ;
”,” pour séparer les lignes.
En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”$<$” et ”$>$”.
Enfin, une matrice ou un tableau à deux dimensions, composé d’une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”$<$” et ”$>$”.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :
”|” pour séparer les colonnes ;
”,” pour séparer les lignes.
En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”$<$” et ”$>$”.
Enfin, une matrice ou un tableau à deux dimensions, composé d’une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”$<$” et ”$>$”.

Pour construire une matrice ou un vecteur, ou plus généralement un tableau de type `Array`, on utilise deux séparateurs :
”|” pour séparer les colonnes ;
”,” pour séparer les lignes.
En outre, chaque ligne ou colonne est entourée par des crochets anguleux ”$<$” et ”$>$”.
Enfin, une matrice ou un tableau à deux dimensions, composé d’une succession de lignes ou de colonnes, est lui-même entouré par un second niveau de crochets anguleux ”$<$” et ”$>$”.

Saisie d'une matrice par colonnes

```
> A :=< <1,2>|<3,4>|<5,6> >;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

Saisie d'un vecteur colonne :

```
> v :=<0,1>;
```

$$v := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Saisie d’une matrice par colonnes

```
> A :=< <1,2>|<3,4>|<5,6> >;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

Saisie d’un vecteur colonne :

```
> v :=<0,1>;
```

$$v := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Saisie d'une matrice par colonnes

```
> A :=< <1,2>|<3,4>|<5,6> >;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

Saisie d'un vecteur colonne :

```
> v :=<0,1>;
```

$$v := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Saisie d’une matrice par colonnes


```
> A :=< <1,2>|<3,4>|<5,6> >;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

Saisie d’un vecteur colonne :

```
> v :=<0,1>;
```

$$v := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Saisie d’une matrice par colonnes


```
> A :=< <1,2>|<3,4>|<5,6> >;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

Saisie d’un vecteur colonne :

```
> v :=<0,1>;
```

$$v := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Saisie d'une matrice par lignes


```
> B :=< <1|2|3>,<4|5|6> >;
```

$$B := \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Saisie d'un vecteur ligne :

```
> w :=<0|1|2>;
```

$$w := [0, 1, 2]$$

Saisie d'une matrice par lignes

```
> B :=< <1|2|3>,<4|5|6> >;
```

$$B := \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Saisie d'un vecteur ligne :

```
> w :=<0|1|2>;
```

$$w := [0, 1, 2]$$

Saisie d'une matrice par lignes


```
> B :=< <1|2|3>,<4|5|6> >;
```

$$B := \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Saisie d'un vecteur ligne :

```
> w :=<0|1|2>;
```

$$w := [0, 1, 2]$$

Saisie d'une matrice par lignes


```
> B :=< <1|2|3>,<4|5|6> >;
```

$$B := \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Saisie d'un vecteur ligne :

```
> w :=<0|1|2>;
```

$$w := [0, 1, 2]$$

Saisie d'une matrice par lignes

```
> B :=< <1|2|3>,<4|5|6> >;
```

$$B := \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Saisie d'un vecteur ligne :

```
> w :=<0|1|2>;
```

$$w := [0, 1, 2]$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs "| " et ",". Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs ”| ” et ”,”. Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs "|" et ",". Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs "| " et ",". Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs ”| ” et ”,”. Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs "| " et ",". Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Concaténation de matrices et de vecteurs

On peut facilement rajouter une ligne ou une colonne à une matrice avec les séparateurs "| " et ",". Cette méthode porte le nom de **concaténation**.

```
> <A|v>;
```

$$\begin{pmatrix} 1 & 3 & 5 & 0 \\ 2 & 4 & 6 & 1 \end{pmatrix}$$

```
> <B,<7|8|9> >;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Construction par une fonction défininssante

Si l'élément (i, j) de la matrice est donné par la valeur $f(i, j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i, j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction défininssante

Si l'élément (i, j) de la matrice est donné par la valeur $f(i, j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i, j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction définissante

Si l'élément (i, j) de la matrice est donné par la valeur $f(i, j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i, j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction défininssante

Si l'élément (i,j) de la matrice est donné par la valeur $f(i,j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i,j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction défininssante

Si l'élément (i,j) de la matrice est donné par la valeur $f(i,j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i,j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction définissante

Si l'élément (i,j) de la matrice est donné par la valeur $f(i,j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i,j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction définissante

Si l'élément (i, j) de la matrice est donné par la valeur $f(i, j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i, j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction défininssante

Si l'élément (i,j) de la matrice est donné par la valeur $f(i,j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i,j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction définissante

Si l'élément (i,j) de la matrice est donné par la valeur $f(i,j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i,j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction défininssante

Si l'élément (i, j) de la matrice est donné par la valeur $f(i, j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i, j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

Construction par une fonction défininssante

Si l'élément (i, j) de la matrice est donné par la valeur $f(i, j)$ d'une fonction f, on utilise la construction suivante :

```
> f :=(i,j)->2*(i+j)-4;
```

$$f := (i, j) \to 2i + 2j - 4$$

```
> C :=Matrix(3,3,f);
```

$$C := \begin{pmatrix} 0 & 1 & 4 \\ 2 & 4 & 6 \\ 4 & 6 & 8 \end{pmatrix}$$

La commande `Matrix` nécessite donc trois arguments : le nombre de lignes, le nombre de colonnes et la fonction définissante.

On affiche une matrice ou un vecteur en tapant tout simplement son nom, comme pour une variable quelconque :

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> v;
```

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

On affiche une matrice ou un vecteur en tapant tout simplement son nom, comme pour une variable quelconque :

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> v;
```

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

On affiche une matrice ou un vecteur en tapant tout simplement son nom, comme pour une variable quelconque :

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> v;
```

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

On affiche une matrice ou un vecteur en tapant tout simplement son nom, comme pour une variable quelconque :

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> v;
```

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

On affiche une matrice ou un vecteur en tapant tout simplement son nom, comme pour une variable quelconque :

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> v;
```

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

On affiche une matrice ou un vecteur en tapant tout simplement son nom, comme pour une variable quelconque :

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> v;
```

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Pour afficher (et manipuler) un élément d'une matrice, on utilise les crochets "[" et "]" :

```
> A[2,3];
```

$$6$$

```
> v[1];
```

$$0$$

Comme en mathématiques, on commence par indiquer le numéro de la ligne, puis le numéro de la colonne.

Pour afficher (et manipuler) un élément d'une matrice, on utilise les crochets "[" et "]" :

```
> A[2,3];
```

$$6$$

```
> v[1];
```

$$0$$

Comme en mathématiques, on commence par indiquer le numéro de la ligne, puis le numéro de la colonne.

Pour afficher (et manipuler) un élément d'une matrice, on utilise les crochets "[" et "]" :

```
> A[2,3];
```

$$6$$

```
> v[1];
```

$$0$$

Comme en mathématiques, on commence par indiquer le numéro de la ligne, puis le numéro de la colonne.

Pour afficher (et manipuler) un élément d'une matrice, on utilise les crochets "[" et "]" :

```
> A[2,3];
```

$$6$$

```
> v[1];
```

$$0$$

Comme en mathématiques, on commence par indiquer le numéro de la ligne, puis le numéro de la colonne.

Pour afficher (et manipuler) un élément d'une matrice, on utilise les crochets "[" et "]" :

```
> A[2,3];
```

$$6$$

```
> v[1];
```

$$0$$

Comme en mathématiques, on commence par indiquer le numéro de la ligne, puis le numéro de la colonne.

Pour afficher (et manipuler) un élément d'une matrice, on utilise les crochets "[" et "]" :

```
> A[2,3];
```

$$6$$

```
> v[1];
```

$$0$$

Comme en mathématiques, on commence par indiquer le numéro de la ligne, puis le numéro de la colonne.

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour modifier un élement de la matrice ou du vecteur, on utilise encore les crochets "[" et "]" :

```
> A[2,3] :=-10;
```

$$A_{2,3} := -10$$

```
> A;
```

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & -10 \end{pmatrix}$$

```
> v[1] :=-8;
```

$$v_1 := -8$$

```
> v;
```

$$\begin{pmatrix} -8 \\ 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une `matrice constante`, on donne en premier argument la valeur commune des éléments de la matrice, en deuxième argument le nombre de lignes et en troisième argument le nombre de colonnes :

```
> ConstantMatrix(6,2,4);
```

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 \end{pmatrix}$$

Pour définir la matrice *identité*, on donne sa taille :

```
> IdentityMatrix(3);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Pour définir une matrice *diagonale*, on donne une liste contenant les éléments diagonaux :

```
> DiagonalMatrix([1,2,7]);
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 7 \end{pmatrix}$$

On peut définir une matrice *aléatoire*, en donnant les nombres de lignes et de colonnes. L'option `generator=a..b` indique l'intervalle dans lequel seront tirés les nombres entiers aléatoires :

```
> R :=RandomMatrix(4,2,generator=0..20);
```

$$R := \begin{pmatrix} 11 & 20 \\ 4 & 20 \\ 19 & 19 \\ 17 & 3 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Commençons par définir quelques matrices et vecteurs :

```
> E := < <1,2,3>|<4,5,6> >;
```

$$E := \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
> F := < <7,8,9>|<10,11,12> >;
```

$$F := \begin{pmatrix} 7 & 10 \\ 8 & 11 \\ 9 & 12 \end{pmatrix}$$

```
> G := < <0,1,1>|<3,-1,0>|<-1,2,1> >;
```

$$G := \begin{pmatrix} 0 & 3 & -1 \\ 1 & -1 & 2 \\ 1 & 0 & 1 \end{pmatrix}$$

Puis :

```
> v :=<1,2>;
```

$$v := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

```
> w :=<3|4|5>;
```

$$w := [3, 4, 5]$$

Puis :

```
> v :=<1,2>;
```

$$v := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

```
> w :=<3|4|5>;
```

$$w := [3, 4, 5]$$

Puis :

```
> v :=<1,2>;
```

$$v := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

```
> w :=<3|4|5>;
```

$$w := [3, 4, 5]$$

Puis :

```
> v :=<1,2>;
```

$$v := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

```
> w :=<3|4|5>;
```

$$w := [3, 4, 5]$$

Puis :

```
> v :=<1,2>;
```

$$v := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

```
> w :=<3|4|5>;
```

$$w := [3, 4, 5]$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

Les opérations algébriques s'effectuent alors très simplement. L'*addition* s'écrit :

```
> E+F;
```

$$\begin{pmatrix} 8 & 14 \\ 10 & 16 \\ 12 & 18 \end{pmatrix}$$

La *multiplication par un scalaire* utilise le symbole "$*$" :

```
> 2*E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

ou bien par le symbole "." du produit matriciel :

```
> 2.E;
```

$$\begin{pmatrix} 2 & 8 \\ 4 & 10 \\ 6 & 12 \end{pmatrix}$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *multiplication matricielle* s'écrit donc avec le symbole "." :

```
> G.E;
```

$$\begin{pmatrix} 3 & 9 \\ 5 & 11 \\ 4 & 10 \end{pmatrix}$$

On peut de même multiplier une matrice par un vecteur, par la droite ou par la gauche, du moment que les dimensions concordent :

```
> E.v;
```

$$\begin{pmatrix} 9 \\ 12 \\ 15 \end{pmatrix}$$

```
> w.F;
```

$$([98, 134])$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1);
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1) ;
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1) ;
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1);
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1) ;
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1) ;
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1) ;
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1);
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

La *puissance* se note :

```
> G ^ 3;
```

$$\begin{pmatrix} 2 & 9 & -3 \\ 3 & -1 & 6 \\ 3 & 0 & 5 \end{pmatrix}$$

En particulier, l'*inverse* - lorsqu'il existe - s'obtient par :

```
> G ^(-1) ;
```

$$\begin{pmatrix} \frac{-1}{2} & \frac{-3}{2} & \frac{5}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{3}{2} & \frac{-3}{2} \end{pmatrix}$$

Finalement, on peut écrire une expression algébrique entière grâce à ces symboles :

```
> (1/3).G ^(-1).(E+F).v;
```

$$\begin{pmatrix} 13 \\ 5 \\ 3 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Commençons par définir quelques vecteurs :

```
> u := <1,1,0>;
```

$$u := \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

```
> v :=<1,2,3>;
```

$$v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
> w :=<9,8,7>;
```

$$w := \begin{pmatrix} 9 \\ 8 \\ 7 \end{pmatrix}$$

Le *produit scalaire* est :

```
> DotProduct(v,w);
```

$$46$$

Le *produit vectoriel* s'écrit :

```
> CrossProduct(v,w);
```

$$\begin{pmatrix} -10 \\ 20 \\ -10 \end{pmatrix}$$

Le *produit scalaire* est :

```
> DotProduct(v,w);
```

$$46$$

Le *produit vectoriel* s'écrit :

```
> CrossProduct(v,w);
```

$$\begin{pmatrix} -10 \\ 20 \\ -10 \end{pmatrix}$$

Le *produit scalaire* est :

```
> DotProduct(v,w);
```

$$46$$

Le *produit vectoriel* s'écrit :

```
> CrossProduct(v,w);
```

$$\begin{pmatrix} -10 \\ 20 \\ -10 \end{pmatrix}$$

Le *produit scalaire* est :

```
> DotProduct(v,w);
```

$$46$$

Le *produit vectoriel* s'écrit :

```
> CrossProduct(v,w);
```

$$\begin{pmatrix} -10 \\ 20 \\ -10 \end{pmatrix}$$

Le *produit scalaire* est :

```
> DotProduct(v,w);
```

$$46$$

Le *produit vectoriel* s'écrit :

```
> CrossProduct(v,w);
```

$$\begin{pmatrix} -10 \\ 20 \\ -10 \end{pmatrix}$$

Le *produit scalaire* est :

```
> DotProduct(v,w);
```

$$46$$

Le *produit vectoriel* s'écrit :

```
> CrossProduct(v,w);
```

$$\begin{pmatrix} -10 \\ 20 \\ -10 \end{pmatrix}$$

Quant au produit "mixte" ou *déterminant* d'une matrice carrée :

```
> M :=<u|v|w>;
```

$$M := \begin{pmatrix} 1 & 1 & 9 \\ 1 & 2 & 8 \\ 0 & 3 & 7 \end{pmatrix}$$

On calcule alors le déterminant de M :

```
> Determinant(M);
```

$$10$$

Quant au produit ”mixte” ou *déterminant* d’une matrice carrée :

```
> M :=<u|v|w>;
```

$$M := \begin{pmatrix} 1 & 1 & 9 \\ 1 & 2 & 8 \\ 0 & 3 & 7 \end{pmatrix}$$

On calcule alors le déterminant de M :

```
> Determinant(M);
```

$$10$$

Quant au produit "mixte" ou *déterminant* d'une matrice carrée :

```
> M :=<u|v|w>;
```

$$M := \begin{pmatrix} 1 & 1 & 9 \\ 1 & 2 & 8 \\ 0 & 3 & 7 \end{pmatrix}$$

On calcule alors le déterminant de M :

```
> Determinant(M);
```

$$10$$

Quant au produit "mixte" ou *déterminant* d'une matrice carrée :

```
> M :=<u|v|w>;
```

$$M := \begin{pmatrix} 1 & 1 & 9 \\ 1 & 2 & 8 \\ 0 & 3 & 7 \end{pmatrix}$$

On calcule alors le déterminant de M :

```
> Determinant(M);
```

$$10$$

Quant au produit ”mixte” ou *déterminant* d’une matrice carrée :

```
> M :=<u|v|w>;
```

$$M := \begin{pmatrix} 1 & 1 & 9 \\ 1 & 2 & 8 \\ 0 & 3 & 7 \end{pmatrix}$$

On calcule alors le déterminant de M :

```
> Determinant(M);
```

$$10$$

Quant au produit ”mixte” ou *déterminant* d'une matrice carrée :

```
> M :=<u|v|w>;
```

$$M := \begin{pmatrix} 1 & 1 & 9 \\ 1 & 2 & 8 \\ 0 & 3 & 7 \end{pmatrix}$$

On calcule alors le déterminant de M :

```
> Determinant(M);
```

$$10$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1) ;
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2) ;
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity) ;
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1);
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2);
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity);
```

$$3$$

La *norme p* d'un vecteur v, $\|v\|_p$, se note (pour $p = 1, 2$ ou ∞) :

```
> Norm(v,1);
```

$$6$$

La norme euclidienne est donc donnée par :

```
> Norm(v,2);
```

$$\sqrt{14}$$

Et la norme infinie :

```
> Norm(v,infinity);
```

$$3$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x^2;
```

$$carre := x \rightarrow x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x^2;
```

$$carre := x \rightarrow x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x ^2;
```

$$carre := x \rightarrow x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x^2;
```

$$carre := x \rightarrow x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x^2;
```

$$carre := x \rightarrow x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x^2;
```

$$carre := x \to x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x^2;
```

$$carre := x \to x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

Quand on souhaite appliquer une même fonction à chaque élément d'une matrice, on utilise la commande `map` :

```
> P :=< <1,2>|<3,4> >;
```

$$P := \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

```
> carre := x->x ^2;
```

$$carre := x \rightarrow x^2$$

```
> map(carre,P);
```

$$\begin{pmatrix} 1 & 9 \\ 4 & 16 \end{pmatrix}$$

On peut effectuer des calculs plus sophistiqués sur les matrices, par exemple :

```
> M :=< <18,3,-9>|<42,-9,-21>|<12,6,-6> >;
```

$$M := \begin{pmatrix} 18 & 42 & 12 \\ 3 & -9 & 6 \\ -9 & -21 & -6 \end{pmatrix}$$

La *transposée* est donnée par

```
> Transpose(M);
```

$$\begin{pmatrix} 18 & 3 & -9 \\ 42 & -9 & -21 \\ 12 & 6 & -6 \end{pmatrix}$$

On peut effectuer des calculs plus sophistiqués sur les matrices, par exemple :

```
> M :=< <18,3,-9>|<42,-9,-21>|<12,6,-6> >;
```

$$M := \begin{pmatrix} 18 & 42 & 12 \\ 3 & -9 & 6 \\ -9 & -21 & -6 \end{pmatrix}$$

La *transposée* est donnée par

```
> Transpose(M);
```

$$\begin{pmatrix} 18 & 3 & -9 \\ 42 & -9 & -21 \\ 12 & 6 & -6 \end{pmatrix}$$

On peut effectuer des calculs plus sophistiqués sur les matrices, par exemple :

```
> M :=< <18,3,-9>|<42,-9,-21>|<12,6,-6> >;
```

$$M := \begin{pmatrix} 18 & 42 & 12 \\ 3 & -9 & 6 \\ -9 & -21 & -6 \end{pmatrix}$$

La *transposée* est donnée par

```
> Transpose(M);
```

$$\begin{pmatrix} 18 & 3 & -9 \\ 42 & -9 & -21 \\ 12 & 6 & -6 \end{pmatrix}$$

On peut effectuer des calculs plus sophistiqués sur les matrices, par exemple :

```
> M :=< <18,3,-9>|<42,-9,-21>|<12,6,-6> >;
```

$$M := \begin{pmatrix} 18 & 42 & 12 \\ 3 & -9 & 6 \\ -9 & -21 & -6 \end{pmatrix}$$

La *transposée* est donnée par

```
> Transpose(M);
```

$$\begin{pmatrix} 18 & 3 & -9 \\ 42 & -9 & -21 \\ 12 & 6 & -6 \end{pmatrix}$$

On peut effectuer des calculs plus sophistiqués sur les matrices, par exemple :

```
> M :=< <18,3,-9>|<42,-9,-21>|<12,6,-6> >;
```

$$M := \begin{pmatrix} 18 & 42 & 12 \\ 3 & -9 & 6 \\ -9 & -21 & -6 \end{pmatrix}$$

La *transposée* est donnée par

```
> Transpose(M);
```

$$\begin{pmatrix} 18 & 3 & -9 \\ 42 & -9 & -21 \\ 12 & 6 & -6 \end{pmatrix}$$

On peut effectuer des calculs plus sophistiqués sur les matrices, par exemple :

```
> M :=< <18,3,-9>|<42,-9,-21>|<12,6,-6> >;
```

$$M := \begin{pmatrix} 18 & 42 & 12 \\ 3 & -9 & 6 \\ -9 & -21 & -6 \end{pmatrix}$$

La *transposée* est donnée par

```
> Transpose(M);
```

$$\begin{pmatrix} 18 & 3 & -9 \\ 42 & -9 & -21 \\ 12 & 6 & -6 \end{pmatrix}$$

La *trace* :

```
> Trace(M);
```

$$3$$

Le *rang* :

```
> Rank(M);
```

$$2$$

La *trace* :

```
> Trace(M) ;
```

$$3$$

Le *rang* :

```
> Rank(M) ;
```

$$2$$

La *trace* :

```
> Trace(M) ;
```

$$3$$

Le *rang* :

```
> Rank(M) ;
```

$$2$$

La *trace* :

```
> Trace(M) ;
```

$$3$$

Le *rang* :

```
> Rank(M) ;
```

$$2$$

La *trace* :

```
> Trace(M);
```

3

Le *rang* :

```
> Rank(M);
```

2

La *trace* :

```
> Trace(M);
```

$$3$$

Le *rang* :

```
> Rank(M);
```

$$2$$

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.
Nous allons voir comment résoudre un tel système avec `LinearSolve`.
Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.

Nous allons voir comment résoudre un tel système avec `LinearSolve`.

Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.

Nous allons voir comment résoudre un tel système avec `LinearSolve`.

Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.
Nous allons voir comment résoudre un tel système avec `LinearSolve`.
Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.
Nous allons voir comment résoudre un tel système avec `LinearSolve`.
Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.
Nous allons voir comment résoudre un tel système avec `LinearSolve`.
Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit le système linéaire écrit sous forme matricielle

$$Ax = b$$

où A est la matrice des coefficients, x est le vecteur colonne des inconnues et b est le vecteur colonne du second membre.
Nous allons voir comment résoudre un tel système avec `LinearSolve`.
Il est conseillé de toujours rajouter l'option `free='lambda'`. Dans le cas où il y aurait une infinité de solutions, les noms des paramètres seront alors λ_1, λ_2, *etc.*, et non pas des noms compliqués attribués automatiquement par `Maple`.

Soit la matrice suivante :

```
> A :=<
<1,1,1,4>|<1,1,-2,1>|<3,1,1,8>|<-1,1,-1,-1>
>;
```

$$A := \begin{pmatrix} 1 & 1 & 3 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & -1 \\ 4 & 1 & 8 & -1 \end{pmatrix}$$

```
> b :=<0,1,1,0>;
```

$$b := \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Soit la matrice suivante :

```
> A :=<
<1,1,1,4>|<1,1,-2,1>|<3,1,1,8>|<-1,1,-1,-1>
>;
```

$$A := \begin{pmatrix} 1 & 1 & 3 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & -1 \\ 4 & 1 & 8 & -1 \end{pmatrix}$$

```
> b :=<0,1,1,0>;
```

$$b := \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Soit la matrice suivante :

```
> A :=<
<1,1,1,4>|<1,1,-2,1>|<3,1,1,8>|<-1,1,-1,-1>
>;
```

$$A := \begin{pmatrix} 1 & 1 & 3 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & -1 \\ 4 & 1 & 8 & -1 \end{pmatrix}$$

```
> b :=<0,1,1,0>;
```

$$b := \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Soit la matrice suivante :

```
> A :=<
<1,1,1,4>|<1,1,-2,1>|<3,1,1,8>|<-1,1,-1,-1>
>;
```

$$A := \begin{pmatrix} 1 & 1 & 3 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & -1 \\ 4 & 1 & 8 & -1 \end{pmatrix}$$

```
> b :=<0,1,1,0>;
```

$$b := \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Soit la matrice suivante :

```
> A :=<
<1,1,1,4>|<1,1,-2,1>|<3,1,1,8>|<-1,1,-1,-1>
>;
```

$$A := \begin{pmatrix} 1 & 1 & 3 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & -1 \\ 4 & 1 & 8 & -1 \end{pmatrix}$$

```
> b :=<0,1,1,0>;
```

$$b := \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

On peut ensuite résoudre le système $Ax = b$:

```
> x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} \frac{25}{6} \\ \frac{4}{3} \\ \frac{-5}{2} \\ -2 \end{pmatrix}$$

On peut ensuite résoudre le système $Ax = b$:

```
> x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} \frac{25}{6} \\ \frac{4}{3} \\ \frac{-5}{2} \\ -2 \end{pmatrix}$$

On peut ensuite résoudre le système $Ax = b$:

```
> x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} \frac{25}{6} \\ \frac{4}{3} \\ \frac{-5}{2} \\ -2 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1 - \lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1 - \lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1-\lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1-\lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1 - \lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1 - \lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$

On peut voir un autre exemple avec une infinité de solutions :

```
> A :=< <1,0,0>|<2,1,0>|<1,0,0>|<-1,-1,-3>
>;
```

$$A := \begin{pmatrix} 1 & 2 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

```
> b :=<2,-1,-9>;
```

$$b := \begin{pmatrix} 2 \\ -1 \\ -9 \end{pmatrix}$$

```
x :=LinearSolve(A,b,free='lambda');
```

$$x := \begin{pmatrix} 1 - \lambda_3 \\ 2 \\ \lambda_3 \\ 3 \end{pmatrix}$$