Considérons alors le système

```
> E1:=a1*x+a2*y+a3*z+a4=0;
> E2:=b1*x+b2*y+b3*z+b4=0;
> E3:=c1*x+c2*y+c3*z+c4=0;
```

Les fonctions équivalentes à **`plot()`** et **`implicitplot()`** sont **`plot3d()`** et **`implicitplot3d()`**. On dispose ici de très nombreux paramètres d'affichage. Testez-les en vous référant à l'aide en ligne.

Pour vérifier si les plans ne sont pas parallèles, il faut calculer les 'dérivées' de fonctions de deux variables.

```
> solve(E1,z); # pour déterminer les valeurs solution de l'inconnue y en
fonction de l'inconnue x
> f:=unapply(%,(x,y)); # pour obtenir une vraie fonction, à partir d'une
expression
> d11:=diff(f(x,y),x); # on calcule en fait la pente de la droite obtenue
par intersection du plan solution de E1 et d'un plan perpendiculaire à
l'axe des y (plan sur lequel y est constant)
> d12:=diff(f(x,y),y); # idem, avec x constant
```

En calculant de la même manière **`d21`**,**`d22`**,**`d31`**,**`d32`** , on peut vérifier que 2 au moins des plans sont parallèles ssi **`a1*(b2*c3-c2*b3)+b1*(a2*c3-a2*b3)+c1*(a2*c3-c2*a3)=0`**.

Exercice (plutôt mathématique…)

+>Déterminer les liens entre les dij et les ak,bk,ck ; retrouvez alors la condition d'égalité ci-dessus.

La semaine prochaine, nous repartirons des systèmes d'équations linéaires, pour développer l'aspect 'matrices et espaces vectoriels'.

2 - Equations linéaires / Interprétation graphique.

Résoudre une équation consiste à déterminer *toutes* les valeurs possibles des variables satisfaisant l'équation (c'est-à-dire telles que celle-ci soit vraie) ; pour un système d'équations, on cherche les valeurs des variables satisfaisant à la fois *chacune* des équations.
Donc si **E1…En** sont des équations, **S1…Sn** les ensembles des solutions de chacune de celles-ci, une solution du système d'équations doit être solution de toutes les équations à la fois, donc dans **E1∩E2∩…∩En** ; et réciproquement, une solution dans **E1∩E2∩…∩En** est à la fois dans **E1,E2…En**, et est donc à la fois solution de chacune des équations, donc du système.
Graphiquement, les solutions d'une équation linéaire à 2 inconnues forment une droite dans le plan (espace vectoriel engendré par toutes les valeurs possibles des deux inconnues). Donc, si on a deux équations linéaires à deux inconnues, correspondant à deux droites, la solution (unique…) du système de ces deux équations est le point situé à la fois dans chacune des droites, c'est-à-dire leur intersection. De même, avec 3 inconnues dans une équation linéaire, on a un plan dans un espace vectoriel de dimension 3, et l'intersection de 3 plans donne encore un point unique, sous des conditions de non-parallélisme. Maple permet de visualiser cela graphiquement.

D'abord en deux dimensions.

Le 'pack' de fonctions graphiques doit être préalablement 'chargé' par

```
> with(plots);
```

Considérons le système

```
> E1:=a1*x+a2*y+a3=0;
> E2:=b1*x+b2*y+b3=0;
```

La fonction **plot()** permet de tracer les graphes de fonctions, dans le plan. Cependant, nous n'avons ici que des équations, et pas de fonctions.
On utilise alors **implicitplot()**, qui trace en fait le graphe de la fonction implicitement définie par l'équation fournie en argument ; on obtient l'ensemble des points solution de l'équation indiquée.

Fixez les valeurs que vous voulez à a1,a2,a3,b1,b2,b3.

```
> implicitplot(E1,x=-10..10,y=-10..10).
```

Pour tracer à la fois les 2 ensembles de solutions, correspondant aux 2 équations, il suffit de remplacer **E1** par la liste **{E1,E2}**.
Le problème est de savoir, au-delà du coup d'œil, si les droites sont ou non parallèles. Elles le sont si elles ont la même 'pente'. Comment déterminez-vous les pentes des droites définies par **E1** et **E2** ?

Vous pouvez bien sûr le faire avec Maple, mais c'est plutôt long et inutile… Voyez néanmoins :

```
> solve(E1,y); # pour déterminer les valeurs solution de l'inconnue y en
fonction de l'inconnue x
> unapply(%,x); # pour obtenir une vraie fonction, à partir d'une
expression
> diff(%(x),x); # pour la dériver
```

Mais par le calcul (à la main ou dans la tête..), on voit que les droites sont parallèles ssi **a1*b2-a2/b1=0**, c'est-à-dire ssi le déterminant du système est nul. Nous reverrons cela dans la partie matricielle, plus tard.

Avec 3 dimensions.

Les raisonnements sont identiques, mais les droites sont alors des plans, qui ne doivent là aussi pas être parallèles pour obtenir une véritable solution ; les noms des fonctions changent aussi, bien sûr.

RESOLUTION D'EQUATIONS ET SYSTEMES D'EQUATIONS

1 - Résolution directe : solve etc.
2 - Equations linéaires / Interprétation graphique

1 - Résolution directe : solve etc.

La fonction **`solve()`** de Maple, très souple et efficace, permet de résoudre de très nombreux types d'équations, qu'il y en ait une seule, ou plusieurs en un système. La syntaxe est plutôt simple et claire :

`solve(`*`équation(s)`*`,`*`variable(s)`*`);`

Détails :
-équation(s) doit être une (liste d') équation(s) ou inéquation(s), l'égalité =0 étant prise par défaut si besoin est.
-variable(s) doit être la (liste de) variable(s) que vous voulez déterminer. Si vous ne précisez rien, toutes les variables libres des équations mentionnées seront prises en compte.

En général, Maple transmet en réponse la liste des solutions. Si celles-ci sont longues à calculer ou encombrantes (polynômes de degré 4), il utilise **`RootOf()`**, ou simplifie en notant **`%1`** un terme qui apparaît plusieurs fois.
Mais pour certaines équations, il n'existe pas de méthode connue pour obtenir la solution exacte. On peut néanmoins obtenir des solutions numériques approchées à l'aide de **`fsolve()`**.

Quelques exemples :

```
> solve(2*a*x^2=b,x);
> solve(2*a*x^2-b,x);
> solve({3*x+7*y-2,4*x-9*y+1,x-y},{x,y});
> solve({3*x+7*y-2,4*x-9*y+1,x-y});
> solve(cos(x)=x);
> solve({y>=3*x,x=y^2-10},{x,y});
> fsolve(cos(x)=x);
> E4:=a*x^4+b*x^3+c*x^2+d*x+e=0;
> solve(E4,x);
> R :=%;
> allvalues(R);
> nops({%}) ;
> a:=1, b:=2, c:=3, d:=4, e:=5;
> allvalues(R);
> evalf(%);
```

Exercices :

> Résolvez un système de 4 équations linéaires à 4 inconnues.
> Retrouvez avec **`solve()`** les expressions connues des solution d'un polynôme quelconque de degré1 ou 2 ; pour le degré 2, trouvez, toujours avec **`solve()`** les valeurs du coefficient de degré 1 telles que de discriminant soit nul, en fonction des autres coefficients ; ces derniers étant fixés, combien y a t'il de valeurs du coefficient de degré 1 possibles telles que le polynôme a une racine unique ?
> Pour n variant entre 1 et un N que vous fixez librement, déterminez les racines de Z^n-1. Comparez avec celles de Z^(-n)-1 ; expliquez.

+> Déterminez les arguments des racines de l'unité.
+> Essayez de déterminer les racines d'un polynôme de degré 4, sans avoir affecté de valeurs aux coefficients. Une méthode existe, mais le résultat est 'encombrant', et Maple répond **`RootOf()`**. Essayez **`allvalues(RootOf…)`**, pour admirer l'ampleur des dégâts. Affectez des valeurs aux coefficients, recommencez, et terminez avec **`evalf()`**. C'est mieux, non ? Remarquez qu'un polynôme de degré 4 a 4 racines.