1. Quelques usages de plot

Vous pouvez télécharger le fichier Pplot.mw. Seules les utilisations basiques sont présentées, vous devez savoir chercher dans l'aide les innombrables options dont vous pouvez avoir besoin dans des cas spécifiques. Les options suivantes sont souvent utiles

- `axes = none` : pas de tracé des axes
- `scaling = constrained` : les unités sont les mêmes sur les deux axes. Sans cette option des affinités orthogonales sont effectuées par défaut.

1. Graphes de fonctions

Les fonctions à valeurs réelles sont définies sur le même intervalle $[a, b]$. On peut utiliser le nom des fonctions ou les valeurs.

```
plot([fonc1,fonc2,fonc3],a..b);
plot([fonc1(x),fonc2(x),fonc3(x)],x=a..b);
```

2. Polygones

Un polygone est une suite de points dans un plan. Chaque point est représenté par une liste de deux réels (float) et la suite de points par une liste de points.

```
A1 := [0,0]: A2 := [1,0] : A3 := [1,1]:
A4 := [2,3]: A5 := [1,7] : A6 := [-1,1]:
plot([A1,A2,A3,A4,A5,A6]);
```

3. Courbes paramétrées

Il s'agit de dessiner le support (trajectoire) de la courbe paramétrée. Une courbe paramétrées est représentée par une liste de trois objets : les deux coordonnées et le domaine du paramètre. On peut donner les fonctions ou les valeurs. On peut dessiner plusieurs trajectoires simultanément en les plaçant dans une liste.

```
Cp1 := [cos, sin, 0..2*Pi];
Cp2 := [3*cos(t), sin(t), t=0..2*Pi];
plot(Cp1);
plot(Cp2);
plot([Cp1,Cp2 ]);
plot([Cp1,Cp2 ], scaling = constrained);
```

2. Mélanger des dessins

Le principe est de calculer les dessins et de les placer dans une liste puis d'utiliser `display` qui se trouve dans la bibliothèque `plots` qui n'est pas chargée automatiquement. Il faut la charger avec `with(plots)`.

1. Simultanément

Si `D1`, `D2`, `D3` sont des dessins (renvoyés par `plot` par exemple). On appelle

```
display([D1,D2,D3]);
```

2. Successivement

Un animation consiste en un affichage successif de plusieurs dessins. La variation d'un dessin à l'autre doit être petite pour qu'on ait une impression de mouvement en la regardant.
On doit utiliser `display` avec l'option `insequence = true`.

```
with(plots):  # display est dans la bibliothèque plots
#listedess désigne une liste de dessins
display(listedess , insequence = true);
```

Exemple

```
with(plots):
opts := scaling = constrained:
tmax :=2;
n:=25;
pas := tmax/n;

x := t -> t+t^3; xx := D(x);
y := t -> t+t^2-t^4; yy := D(y);
M := t -> [x(t),y(t)];

seqDessins := plot([M(0)]):
for i from 1 to n do
  seqDessins := seqDessins , plot([seq(M(j*pas),j=0..i)],opts);
od:
listDessins := [seqDessins]:
display(listDessins,insequence = true);
```

```
# mouvement uniforme
l := t -> evalf(int(sqrt(xx(tt)^2+yy(tt)^2),tt=0..t)):
phi := x -> fsolve(l(t)=x,t):
#tabulation de la bijection réciproque
T:= array(0..n):
# l(tmax) longueur totale
for i from 0 to n do
  T[i] := phi(l(tmax)*i/n);
od:

seqDessins := plot([M(T[0])]):
for i from 1 to n do
  seqDessins := seqDessins , plot([seq(M(T[j]),j=0..i)],opts);
od:
listDessins := [seqDessins]:
display(listDessins,insequence = true);
```