GRAPHES EN DIMENSION DEUX

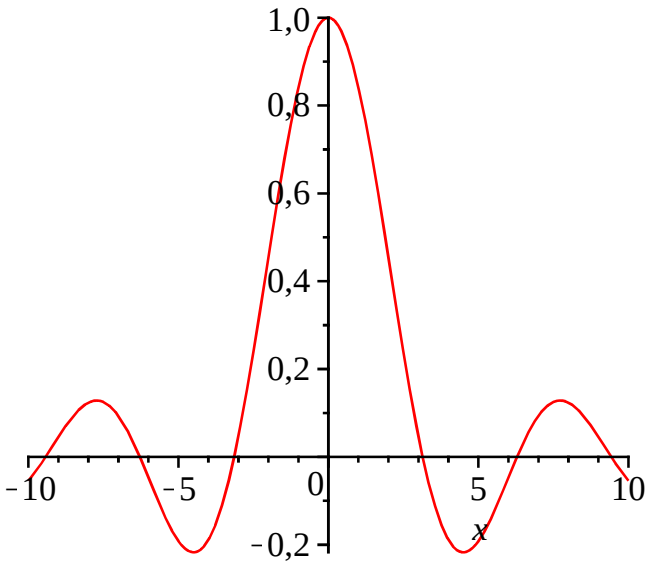
Bernard Dupont

Bernard.Dupont@univ-lille1.fr

La possibilité qu'offrent les logiciels de calcul formel de visualiser en un temps extrêmement court une courbe est un de leurs grands atouts. On sait en effet qu'un graphique est souvent très parlant dans les sciences appliquées et permet, bien mieux que ne le feraient une expression algébrique ou un commentaire littéraire, de s'approprier les propriétés qualitatives du phénomène étudié. Maple est parfaitement profilé pour remplir cette tâche. De fait, une multitude de commandes et d'options, disponibles dans la bibliothèque proncipale ou dans des paquetages dédiés, permet de construire des graphiques (presque) parfaits.

Pour représenter une fonction d'une variable réelle, Maple propose la commande directe **plot** disponible dans le noyau de base, par exemple :

```
> plot(sin(x)/x,x=-10..10);
```



Mais cette commande ne donne pas toute la mesure des capacités graphiques de Maple. On accède à celles-ci en chargeant le module (package) **plots** :

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

[*animate, animate3d, animatecurve, arrow, changecoords, complexplot, complexplot3d, conformal, conformal3d, contourplot, contourplot3d, coordplot, coordplot3d, densityplot, display, dualaxisplot, fieldplot, fieldplot3d, gradplot, gradplot3d, graphplot3d, implicitplot, implicitplot3d, inequal, interactive, interactiveparams, intersectplot, listcontplot, listcontplot3d, listdensityplot, listplot, listplot3d, loglogplot, logplot, matrixplot, multiple, odeplot, pareto, plotcompare, pointplot, pointplot3d, polarplot, polygonplot, polygonplot3d, polyhedra_supported, polyhedraplot, rootlocus, semilogplot, setcolors, setoptions, setoptions3d, spacecurve, sparsematrixplot, surfdata, textplot, textplot3d, tubeplot*] **(1)**

Parmi les fonctionnalités ajoutées, on s'intéressera tout particulièrement dans ce chapitre à animate , arrow, display, implicitplot, interactive, pointplot, textplot.

En outre, les versions successives du logiciel ont ajouté des outils supplémentaires ayant pour but de faciliter la vie des utilisateurs, en particulier dans le domaine graphique via les outils interactifs

smartplot (graphes dynamiques) et Interactive Plot Builder (construction interactive de graphe). Il devient inutile de programmer, ce qui n'est pas forcément une bonne chose ...

Les trois premières sections de ce chapitre présentent les ressources de la commande **plot** en allant du simple (tracés rudimentaires d'expressions, de fonctions, de points et de segments de droite) au complexe (options élaborées de toute nature) via les options élémentaires (spécification de l'intervalle d'arrivée, du repère, traitement des discontinuités, graduation des axes, titrage, etc). La possibilité de construire des courbes paramétrées est exposée dans la section 5.
La troisième section introduit au maniement des commande **display**, **textplot** et **arrows** du paquetage **plots**. Les instructions spécifiques relatives aux courbes implicites et aux animations sont traitées dans les sections 4 et 6.
La section 7 est consacrée aux outils interactifs.

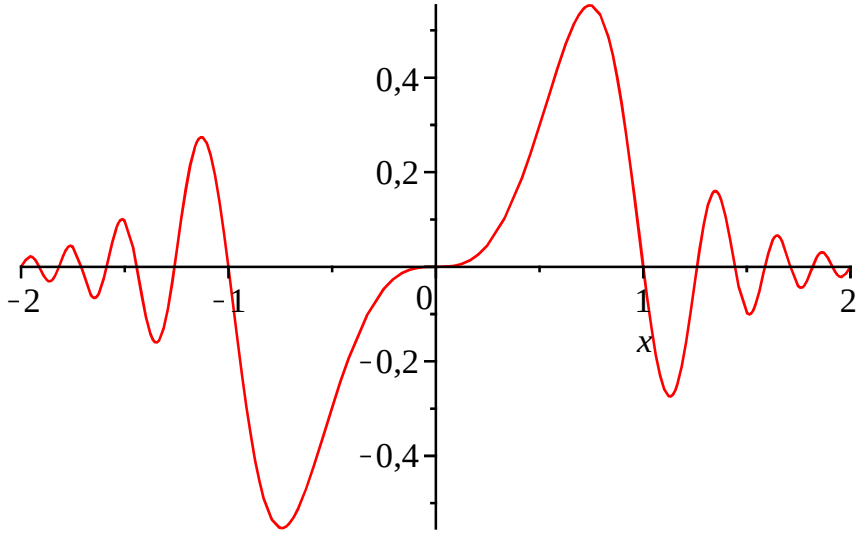
Tracés rudimentaires

Fondamentalement, il faut utiliser la commande universelle **plot** avec au minimum les deux arguments suivants : en premier lieu et en première place, l'expression ou la fonction à représenter; en second lieu, le domaine décrit par la variable. Ce n'est pas tout à fait la même chose de donner une fonction-procédure ou une expression.

Tracé d'une expression

Si la fonction à tracer est explicitement une expression dépendant d'une variable nominative, par exemple *x*, il faut indiquer que c'est cette variable qui prend toutes les valeurs sur un intervalle. La syntaxe est donc : **plot(expression de x,x=a..b)** où **a** < **b** avec **a** éventuellement égal à **-** ∞ et **b** éventuellement égal à **+** ∞.

```
> plot(exp(-x^2)*sin(Pi*x^3),x=-2..2);
```



Sur l'axe des abscisses figure (à un endroit peu usuel) la variable *x*. Un clic gauche sur le graphique fait apparaître le cadre dans lequel il s'insère. On en modifie la taille en tirant sur les poignées. Le pointeur se transforme en viseur et si on clique son centre sur un point quelconque de la courbe, celle-ci se réaffiche à l'intérieur d'un corridor bleu.

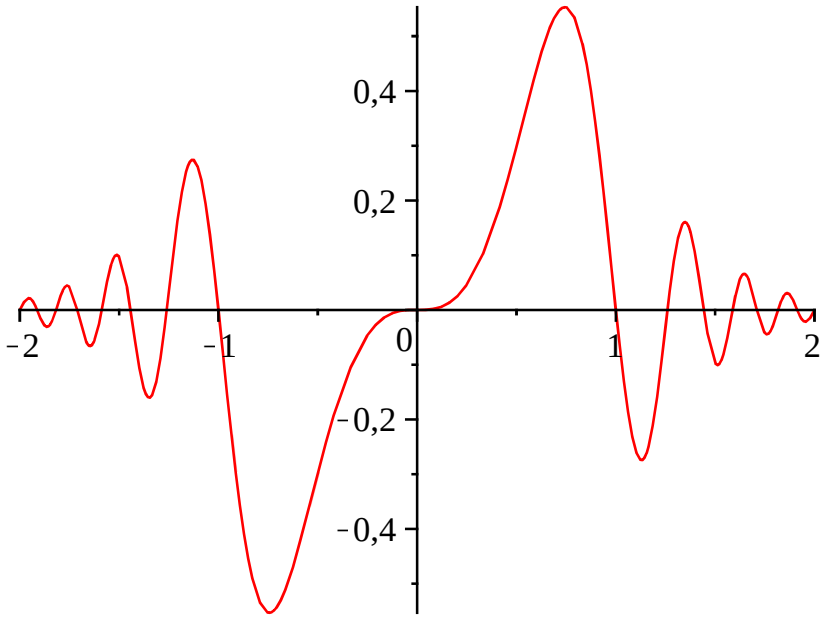
Tracé d'une fonction

Quand une fonction est décrite par une correspondance entre les éléments d'un ensemble de départ et les éléments d'un ensemble d'arrivée, il est inutile de spécifier en option de **plot** le nom de la variable décrivant l'ensemble de départ. La syntaxe est alors **plot(fonction,**

a..b) où **a** < **b** avec **a** éventuellement égal à **-** ∞ et **b** éventuellement égal à **+** ∞. Dans le graphique, le nom de l'argument ne figure pas en abscisse comme c'est le cas pour une expression.

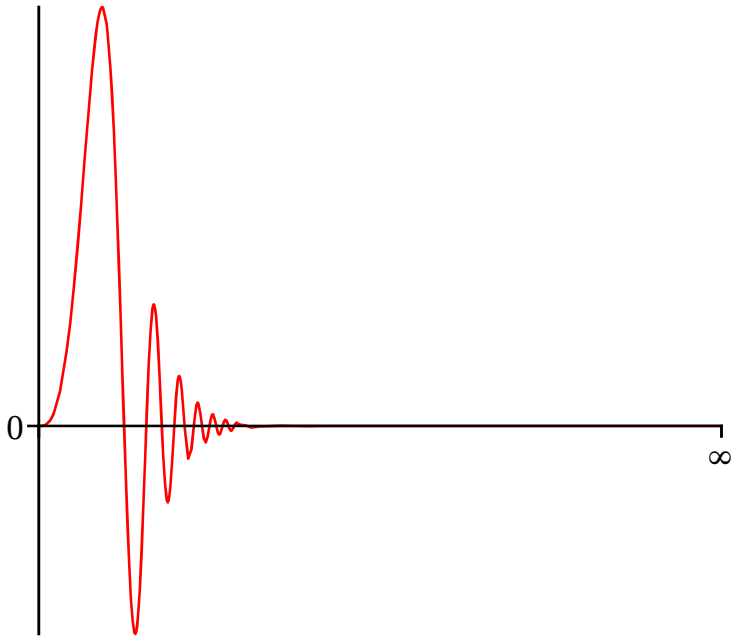
```
> f:=h->exp(-h^2)*sin(Pi*h^3);
  plot(f,-2..2);
```

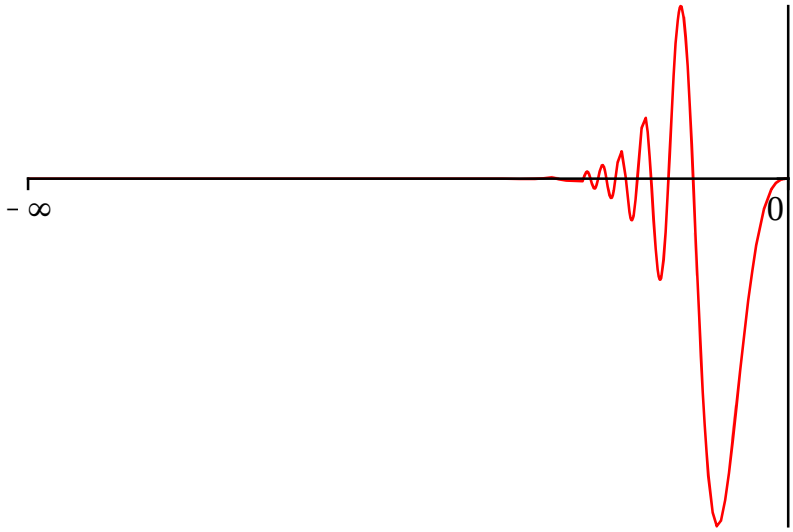
$$f := h \to e^{-h^2} \sin(\pi h^3)$$



Les deux graphiques suivants donnent l'allure de la fonction "à l'infini".

```
> plot(f,0..infinity);
  plot(f,-infinity..0);
```





Comme dans le paragraphe précédent, on a la possibilité de modifier la taille du graphique en cliquant gauche sur celui-ci puis en jouant sur les poignées. De même, les propriétés du viseur sont identiques.

Tracé de points et de segments

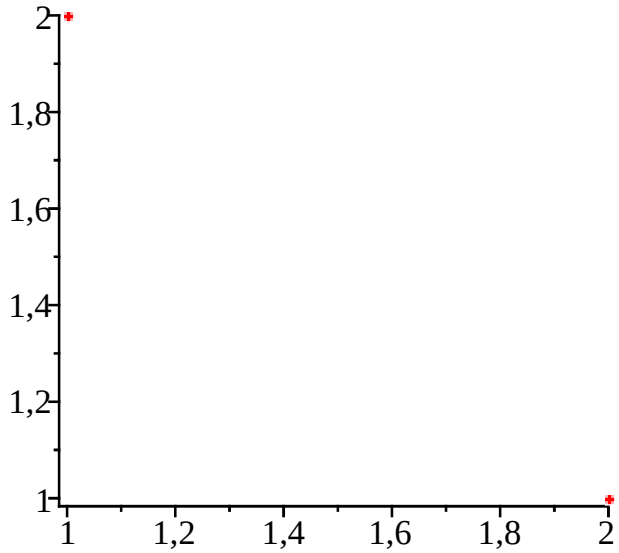
L'instruction **plot** n'est pas réservée aux seules fonctions. Elle permet aussi de visualiser des points et des segments dans le plan.

Pour tracer *n* points de coordonnées (x_1, y_1), ..., (x_n, y_n) dans un graphique, la commande **plot** a la syntaxe minimale suivante :

```
plot([[x[1],y[1]],...,[x[n],y[n]]],style=point)
```

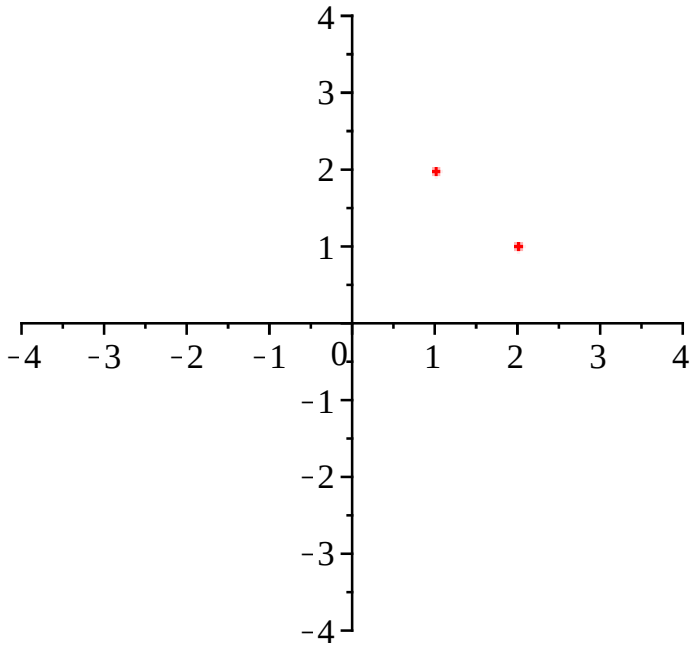
Le premier argument est une **liste** contenant une ou plusieurs **listes** de deux coordonnées. Le second fixe l'option style à point

```
> plot([[1,2],[2,1]],style=point);
```



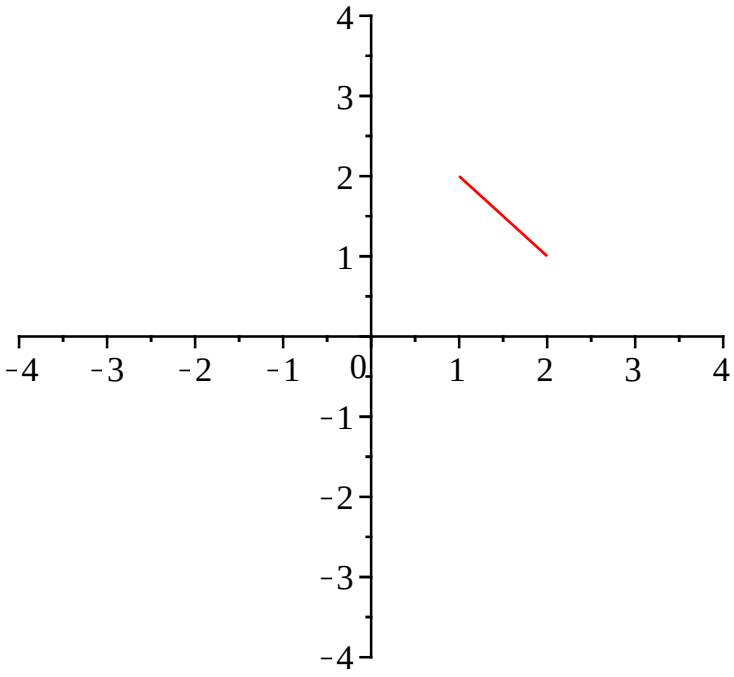
On peut préciser en option l'intervalle de valeurs pour les abscisses et les ordonnées, à condition de les préciser **avant** l'option **style=point** :

```
> plot([[1,2],[2,1]],-4..4,-4..4,style=point);
```



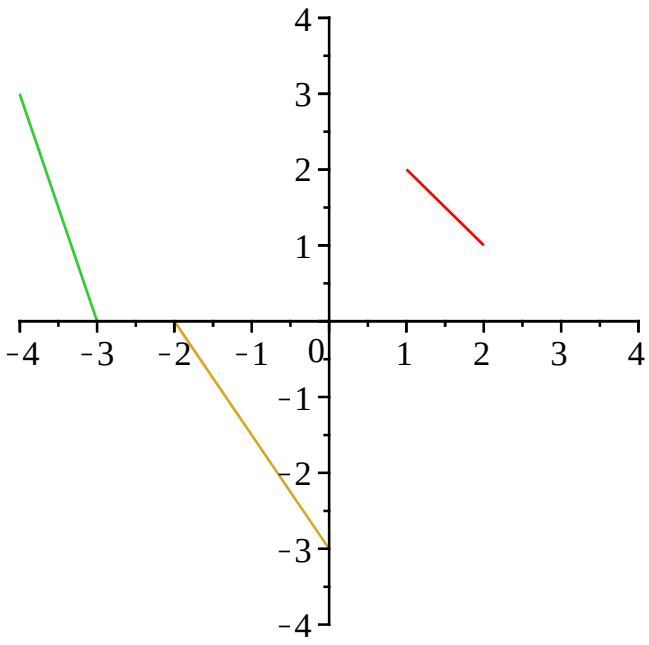
Pour tracer le segment de droite reliant les points du plan de coordonnées (x_1, y_1) et (x_2, y_2), l'option **`style`** doit être fixée à **`line`**.

```
> plot([[1,2],[2,1]],-4..4,-4..4,style=line);
```



Pour obtenir plusieurs segments de droite dans un même graphique, il suffit de respecter la règle de cohérence des crochets pour que la liste des points à tracer soit bien composée de listes de deux listes :

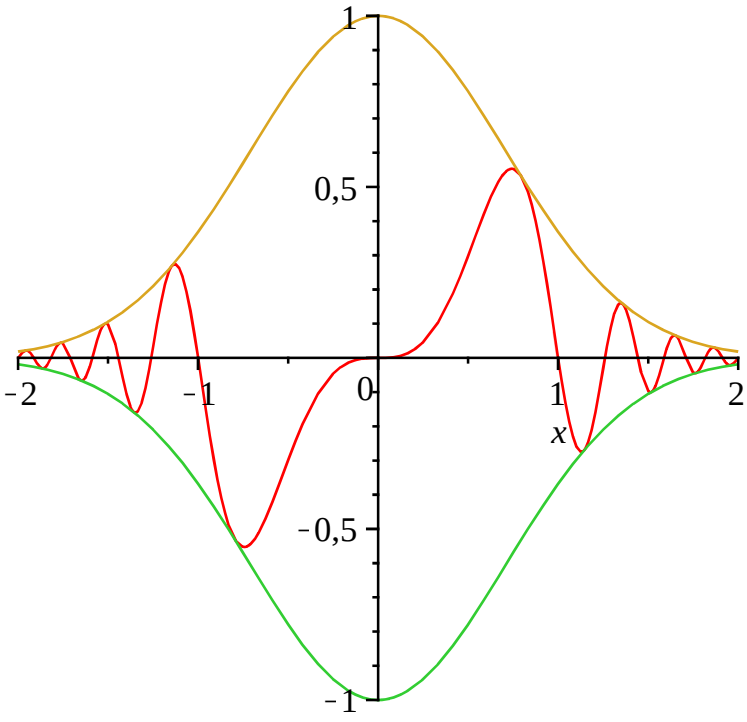
```
> plot([[[1,2],[2,1]],[[-4,3],[-3,0]],[[-2,0],[0,-3]]],-4.
  .4,-4..4,style=line);
```

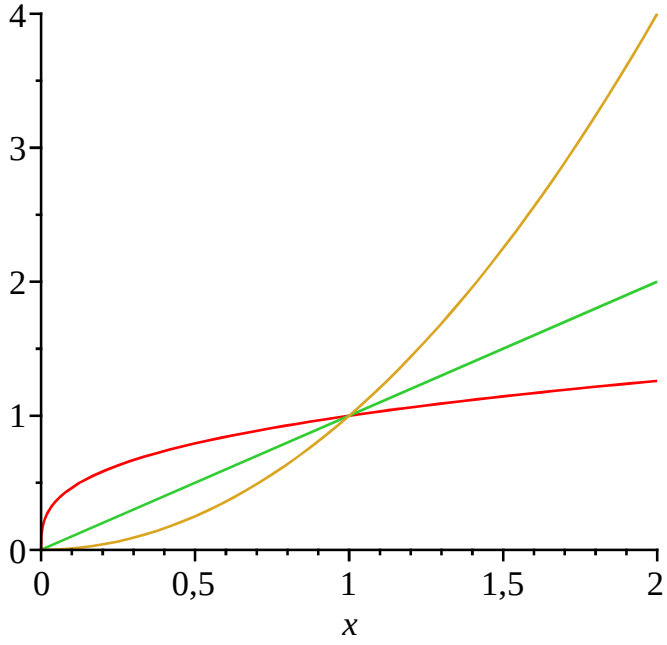


Plusieurs tracés dans un même graphique

Avec **plot**, on peut représenter plusieurs expressions ou fonctions sur le même intervalle de l'ensemble de départ à l'aide d'une syntaxe adaptée. Pour des expressions, on utilise : **plot({expression1,expression2,..,expressionN},x=a..b)** et **plot({fonction1,fonction2,..,fonctionN},a..b)**.
et pour des fonctions :
plot([expression1,expression2,..,expressionN],x=a..b) et **plot([fonction1,fonction2,..,fonctionN],a..b).**
Le premier argument de **plot**est un ensemble (set) ou une liste (list) dont les éléments sont des expressions ou des fonctions, ce qu'on exprime par la mise entre parenthèses des éléments si leur ordre n'est pas important (cas d'un ensemble/set) ou la mise entre crochets si leur ordre est important (cas d'une liste/list).
Maple attribue une couleur différente à chaque représentation graphique, mais on ignore les critères de choix de couleurs de sorte qu'on ne peut pas identifier les fonctions tracées. On reviendra plus loin sur ce problème.

```
> restart;
  plot({exp(-x^2)*sin(Pi*x^3),exp(-x^2),-exp(-x^2)},x=-2.
  .2);#cas d'un ensemble de trois expressions
  plot([x^(1/3),x,x^2],x=0..2);#cas d'une liste de trois
  expressions
```





Technique d'assignation d'un graphique

Dans une session longue qui enchaîne les figures, on a besoin de référencer les graphiques. A priori, il suffit d'attribuer un identificateur à chacun d'eux, mais cette opération simple d'assignation du type **`graphe:=plot( )`** réserve quelques surprises. Dans l'exemple suivant, on commence par assigner un graphique en espérant le labelliser **et** le visualiser.

```
> graphe1:=plot(x^2-6*x,x=-5..5);
```

graphe1 := *PLOT*(...)

Or, le graphique n'apparaît pas. Que contient **`graphe1`**?

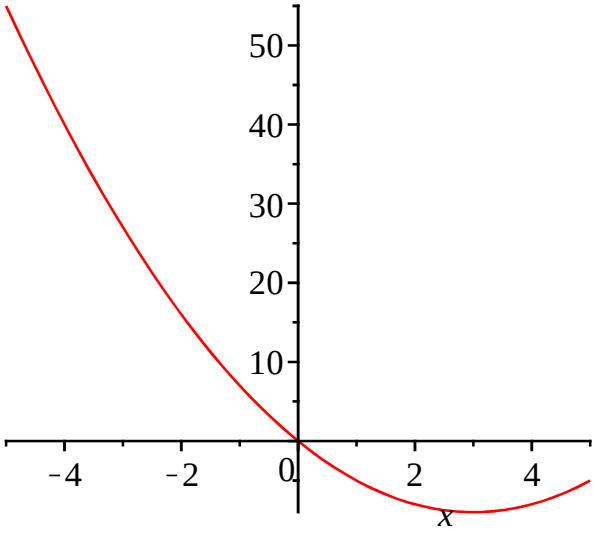
```
> about(graphe1);
PLOT(CURVES([[-5., 55.], [-4.78202845833333346,
51.5599669263098832], [-4.59237258541666636,
```

```
48.6441214757865552], [-4.37908472083333322,
45.4508913172359500], [-4.16438304583333352,
42.3283844274241119], [-3.95070181041666668,
39.3122556573295299], [-3.75259195208333330,
36.5974980713406026], [-3.54746066875000032,
33.8692412088281998], [-3.33531360208333316,
31.1361984367420988], [-3.12384689374999969,
28.5015007780915220], [-2.90632739166666632,
25.8847032575519656], [-2.71473609583333308,
23.6582086450204088], [-2.49904967499999974,
21.2395473281176024], [-2.28247762499999984,
18.9045698586256386], [-2.07377032499999992,
16.7431453108506042], [-1.88424410208333314,
14.8558404487358260], [-1.65887909166666648,
12.7051543907688238], [-1.46796536666666634,
10.9627145177327990], [-1.24588731874999992,
9.02755912352206380], [-1.04931061666666680,
7.39691647024938170], [-.833634818749999340,
5.69675592353233994], [-.628260456249999598,
4.16427393838745540], [-.413973854166665767,
2.65521747693359834], [-.217191860416666494,
1.35032346673125180], [-.4935895833333425 75e-2,
.29639738067678073 8e-1], [.215538397916667180,
-1.24677358652351944], [.407461918750000152,
-2.27874629726856926], [.614744579166667472,
-3.31055657738520193], [.828887850000000092,
-4.28627203212237796], [1.038385262500000029,
-5.15206762162280718], [1.241084781250000055,
-5.90621725324964153], [1.466148037500000000,
-6.64729815713489814], [1.668377383333333262,
-7.22678120678181735], [1.884300375000000046,
-7.75521434677486000], [2.079958697916666620,
-8.15352400246080400], [2.293864925000001116,
-8.50137325585474635], [2.495134014583332944,
-8.74511033676925820], [2.705518206250000072,
-8.91328047314978100], [2.911207208333332911,
-8.99211584014803832], [3.126544531249999544,
-8.98398648161071734], [3.333939966666667584,
-8.88848409866266564], [3.546031754166667944,
-8.70184932344167095], [3.756367314583334504,
-8.42790848542999527], [3.949643850000001064,
-8.09817655815717606], [4.171160570833334804,
-7.62838291732533591], [4.369287766666667844,
-7.12505101205701053], [4.580532681250000244,
-6.50191644350068643], [4.782721018750001104,
```

```
-5.82190576930695869], [5., -5.]],COLOUR(RGB,1.00000000,
0.,0.)),AXESLABELS(x,""),VIEW(-5. .. 5.,DEFAULT)):
  nothing known about this object
```

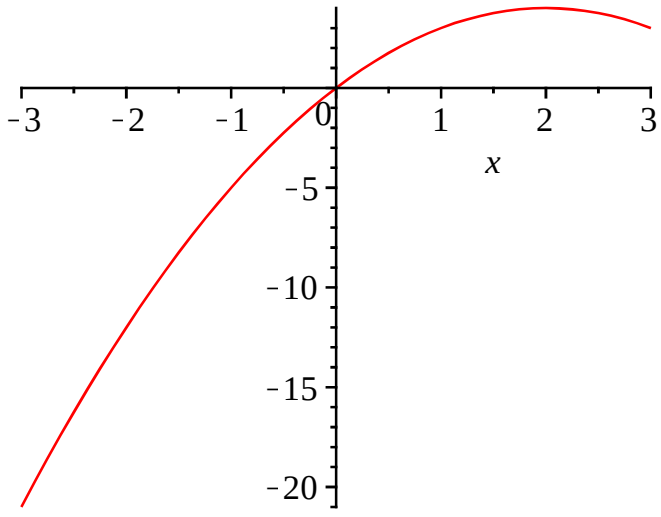
Mais si on demande maintenant :

```
> graphe1;
```



On obtient le résultat recherché. Que s'est-il passé? Maple a commencé par stocker dans l'identificateur la structure du graphique, à savoir des segments de courbes rouges reliant des points successifs connus par leur abscisse et leur ordonnée dans un repère orthogonal - mais non orthonormé. En appelant **graphe1**, Maple exécute les commandes et édite le graphe. Moralité : quand on désire labelliser un graphe et le visualiser aussitôt, il faut procéder en deux temps. La première commande sera close par un "deux points" (ce qui bloque la sortie du résultat Maple) et la seconde appellera le graphique par son identificateur/label suivi de "point-virgule".

```
> graphe2:=plot(-x^2+4*x,x=-3..3):graphe2;
```



Paramétrage de Maple

Si on veut faire apparaître les graphiques d'une session dans une nouvelle fenêtre au lieu de les voir s'afficher directement dans la feuille de travail, il faut paramétrer Maple comme suit : dans Tools, ouvrir Options puis l'onglet Display et cocher sur Window dans la rubrique Plot

Display, enfin valider la requête pour la session en cours ou pour toutes les sessions.

Options graphiques élémentaires

Les options graphiques sont nombreuses dans Maple. On peut en avoir une idée en appelant l'aide par la commande :

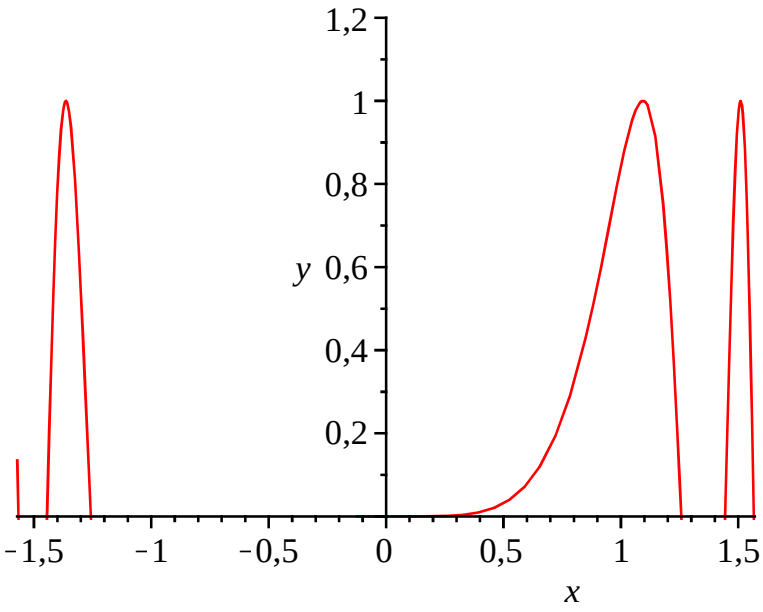
```
> ?plot,options
```

La page dédiée s'ouvre dans une fenêtre indépendante. 32 options sont proposées. Dans cette section, on passe en revue les options basiques, autrement dit les plus importantes.

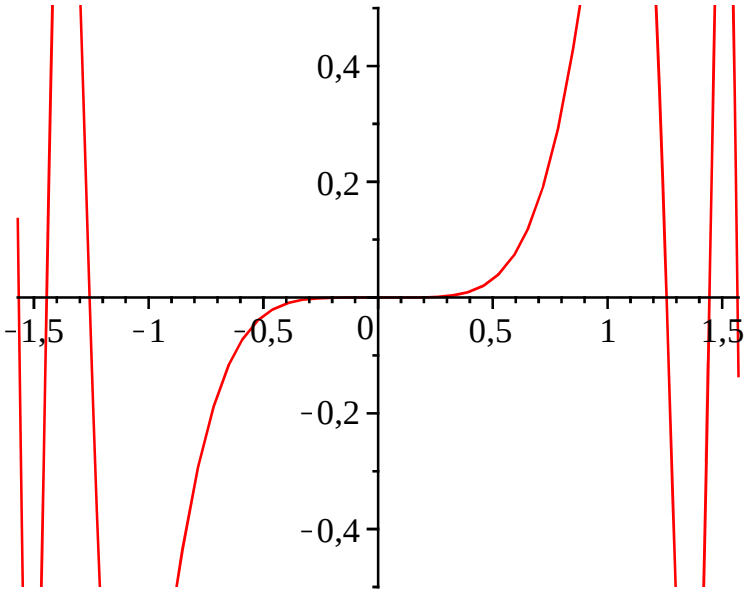
Spécification de l'intervalle d'arrivée (vertical range)

La spécification de l'ensemble d'arrivée est passée en troisième argument de **plot**.
Dans le cas d'une expression représentant $y = f(x)$, on utilise la syntaxe **y=c..d**, et le graphique inscrit x en abscisse et y en ordonnée.
Dans le cas d'une fonction, on utilise la syntaxe **c..d**, et le graphique ne labellise pas l'ordonnée.

```
> restart;
  plot(sin(x^5),x=-Pi/2..Pi/2,y=0..1.2);#cas d'une
  expression
  f:=t->sin(t^5);plot(f,-Pi/2..Pi/2,-0.5..0.5);#cas d'une
  fonction
```



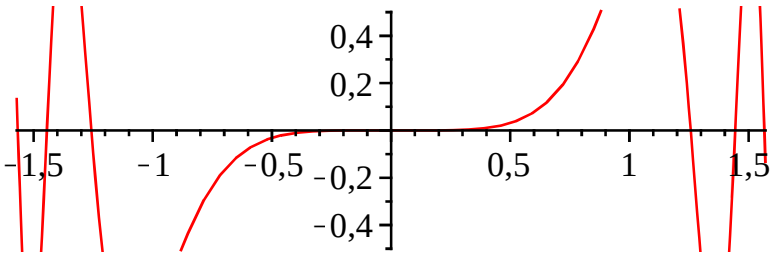
$$f := t \to \sin(t^5)$$



Spécification du repère orthogonal (axes)

Par défaut, Maple choisit un repère qui permet de remplir une feuille de format A4 (largeur de 21 cm et hauteur de 29,7 cm), ce qui amène évidemment des distorsions dans la représentation graphique. Lorsqu'on veut un repère orthonormé, il faut le spécifier en option par la commande **`scaling=constrained`**.
Reprenons l'exemple précédent afin de comparer les sorties :

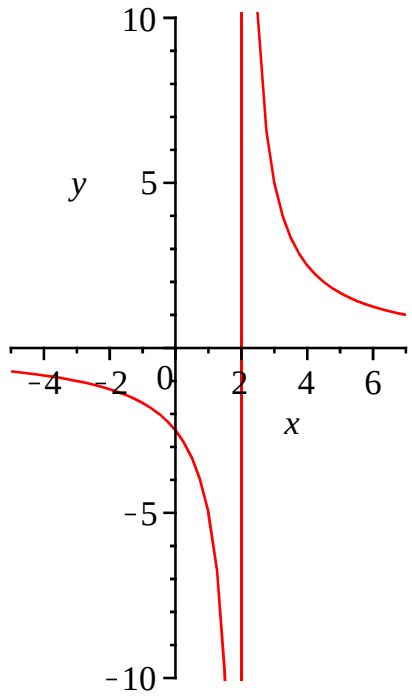
```
> plot(f,-Pi/2..Pi/2,-0.5..0.5,scaling=constrained);
```



Le problème des discontinuités

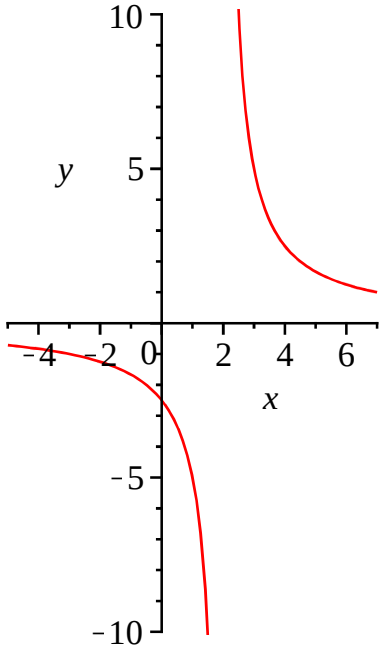
Maple trace un graphique en calculant d'abord les valeurs prises par l'expression ou la fonction puis relie **tous** les points deux à deux avec une droite. Bien évidemment, quand on est en présence d'une asymptote verticale, une droite verticale apparaît dans le graphique alors qu'elle n'appartient pas à la représentation graphique :

```
> restart;plot(5/(x-2),x=-5..7,y=-10..10,scaling=
  constrained);
```



On supprime le tracé de l'asymptote verticale en ajoutant l'option **discont=true**.

```
> plot(5/(x-2),x=-5..7,y=-10..10,scaling=constrained,
  discont=true);
```

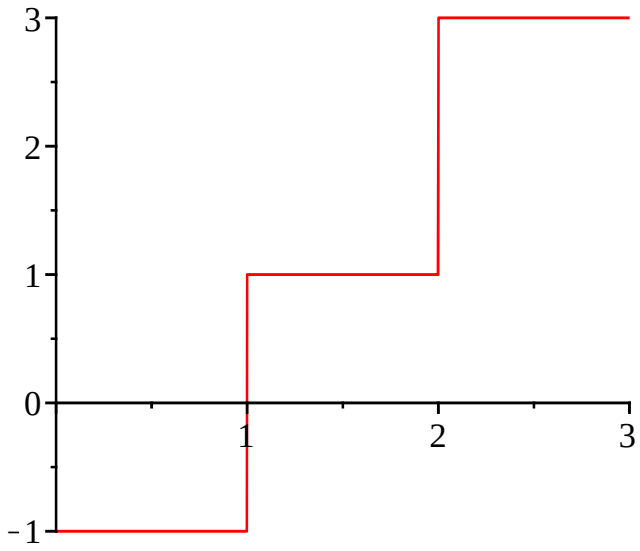


Cette option est également disponible pour une fonction.

Examinons à présent le cas des fonctions définies par morceaux, en particulier les fonctions en escalier qui présentent "naturellement" des discontinuités.

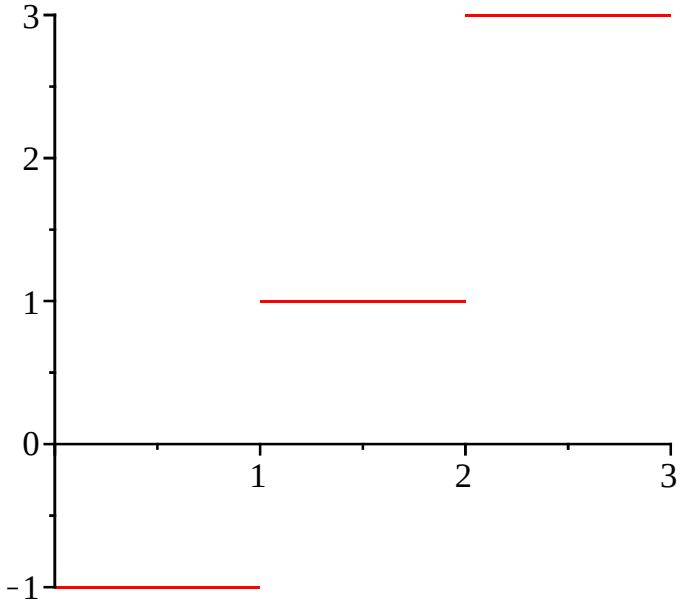
```
> restart:
  f:=x->piecewise(x<1,-1,x<2,1,3);
  plot(f,0..3);
```

$$f := x \rightarrow piecewise(x < 1, -1, x < 2, 1, 3)$$



L'option **discont** permet de faire apparaître nettement les discontinuités :

```
> plot(f,0..3,discont=true);
```



Titre

Un graphique sans titre est comme un tee-shirt sans marque. L'option **title** permet, comme son nom l'indique, d'afficher un titre au dessus du graphique et l'option **titlefont** en précise les caractéristiques typographiques.
Alternativement, l'option **caption** édite un "titre-bandeau".
Toutes deux sont rudimentaires et les résultats ne sont guère satisfaisants. On verra plus loin qu'un titrage plus convaincant est possible avec la commande **textplot** du paquetage **plots**.

Title

Comme un titre est une chaîne de caractères, l'option **title** a pour syntaxe **title= "rédaction du titre"**où les guillemets signalent à Maple qu'il doit traiter une chaîne de caractères. L'option supplémentaire **titlefont** comprend 3 critères : la police de caractères (family), le style (style) et la taille (size). Sa syntaxe est **titlefont=**

[police,style,taille] - notez la présence des crochets pour créer une liste. Toutes les tailles de caractères sont possibles à partir du moment où on choisit un entier naturel raisonnable. Les combinaisons possibles de police et de style sont restreintes :

1- COURIER, BOLD (gras) ou BOLDOBLIQUE (gras penché) ou DEFAULT (maigre) ou OBLIQUE (maigre oblique)

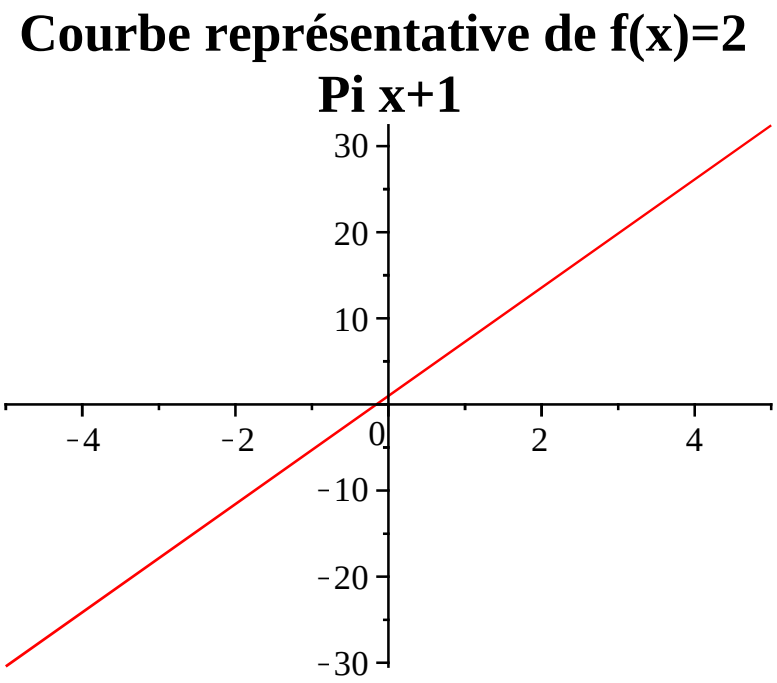
2- HELVETICA, BOLD (gras) ou BOLDOBLIQUE (gras penché) ou DEFAULT (maigre) ou OBLIQUE (maigre oblique)

3- TIMES, ROMAN (roman) ou BOLD (gras) ou ITALIC(italique) ou BOLDITALIC (gras italique)

4- SYMBOL, pas de style.

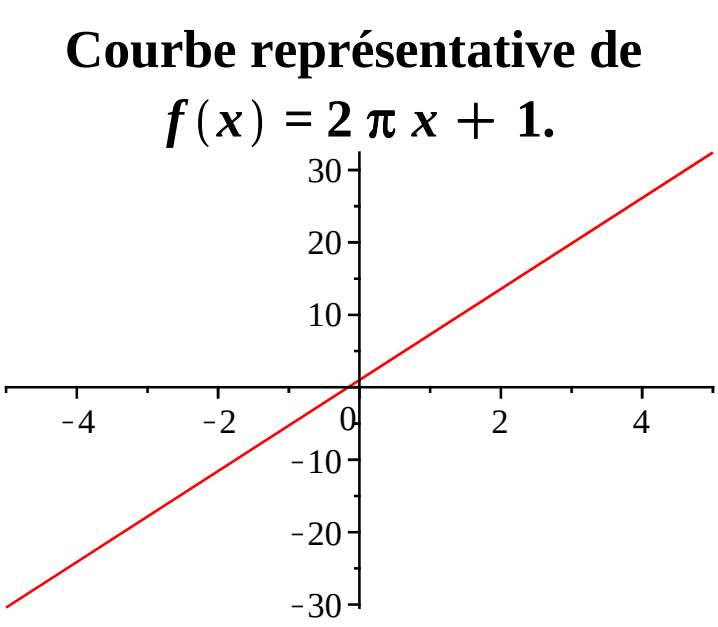
A l'usage, elles s'avèrent très insuffisantes, d'autant plus qu'il est impossible d'écrire proprement une expression mathématique et d'insérer des lettres grecques.

```
> restart;f:=x->2*Pi*x+1:plot(f,-5..5,title="Courbe
  représentative de f(x)=2 Pi x+1",titlefont=[TIMES,BOLD,
  16]);
```



Il existe heureusement une parade en invoquant les ressources de **typeset( )** qui exécute une composition typographique différenciant les chaînes de caractères et les expressions mathématiques. Les blocs de chaînes de caractères sont placés entre guillemets; les blocs d'expressions formelles sont écrits comme des inputs (Maple Input), éventuellement en puisant dans les palettes, et ne sont pas encadrées par des guillemets; des virgules séparent les blocs.

```
> plot(f,-5..5,title=typeset("Courbe représentative de ",
  'f'(x)=2*Pi*x+1,"."),titlefont=[TIMES,BOLD,16]);
```



Caption

L'option **caption** édite un titre sous le graphe. Les règles typographiques sont les mêmes que pour **title** à ceci près qu'on ne peut pas utiliser **titlefont**. Le recours à l'option **font**, qui a les mêmes propriétés que **titlefont**, est peu judicieux puisqu'il modifie non seulement l'allure du titre mais aussi celle des axes du graphique.

```
> plot(f,-5..5,caption=typeset("Figure 1. Courbe
  représentative de ",'f'(x)=2*Pi*x+1,"."));
```

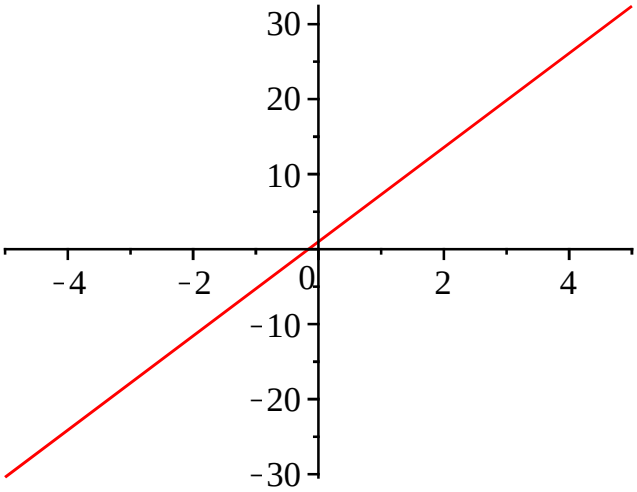


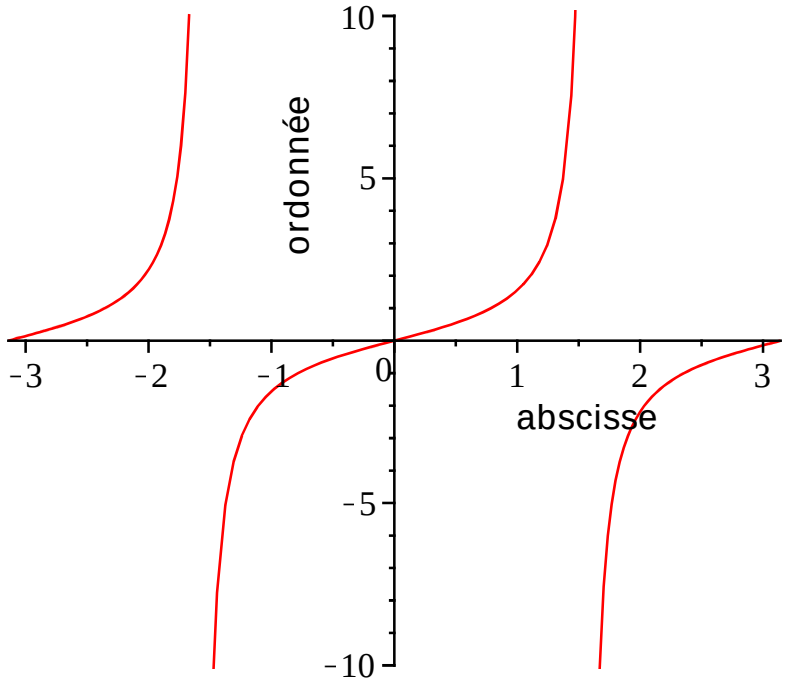
Figure 1. Courbe représentative de
$f(x) = 2\,\pi\,x + 1.$

Labellisation des axes (label)

L'annotation des axes est automatique quand **plot** porte sur une expression : l'axe des abscisses est labellisé par la variable entrant dans l'expression, l'axe des ordonnées n'est pas labellisé si son intervalle n'est pas spécifié et labellisé dans le cas contraire. Il n'est pas possible de modifier l'emplacement - peu commun en France - des labels.
Dans le cas d'une fonction, on dispose de 3 options : **labels**, **labeldirections** et **labelfont**.
L'option **labels** s'égalise à une liste composée de deux chaînes de caractères (avec

utilisation éventuelle de **typeset**), soit **["label de x","label de y"]**. Si on ne veut pas de label, la chaîne de caractères doit être vide et on écrit **labels=["",""]**.
L'option **labeldirections** s'égalise à une liste composée de deux valeurs **[valeur de x,valeur de y]**, une valeur étant **horizontal** ou **vertical**. Par défaut, les deux valeurs sont **horizontal.**
L'option **labelfont** est une liste du type **font**, soit **[police de caractères, style,taille]**. Les choix autorisés sont ceux de **titlefont**.

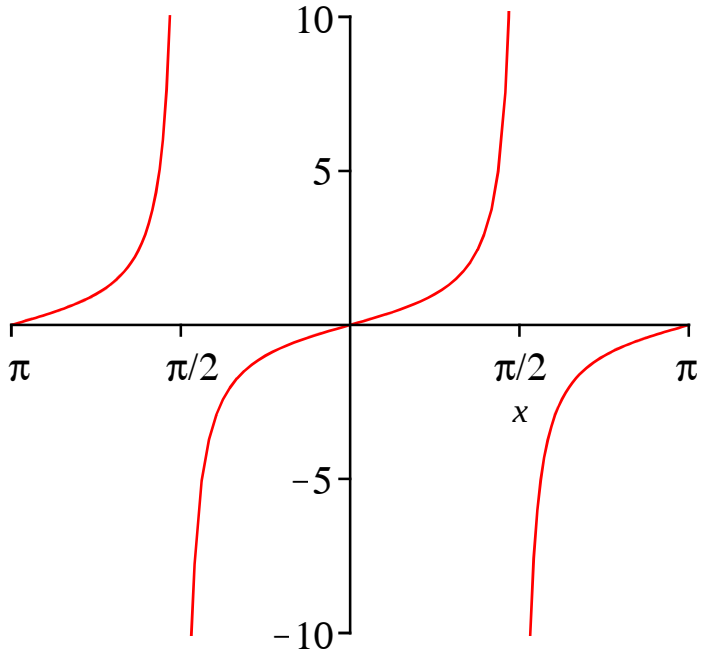
```
> restart;plot(tan(x),x=-Pi..Pi,-10..10,discont=true,
  labels=["abscisse","ordonnée"],labelfont=[HELVETICA,
  DEFAULT,10],labeldirections=[horizontal,vertical]);
```



Graduation des axes (ticks)

On personnalise la graduation de l'axe des abscisses et de l'axe des ordonnées par les options respectives **xtickmarks** et **ytickmarks**.
L'option **tickmarks** peut être utilisée pour indiquer explicitement la localisation des graduations au moyen d'un vecteur-ligne ou pour labelliser et situer des graduations particulières ou encore pour fixer le nombre de graduations.
La personnalisation typographique des labels éventuels se fait par l'option **axesfont** qui a les mêmes arguments que **titlefont** (voir ci-dessus).

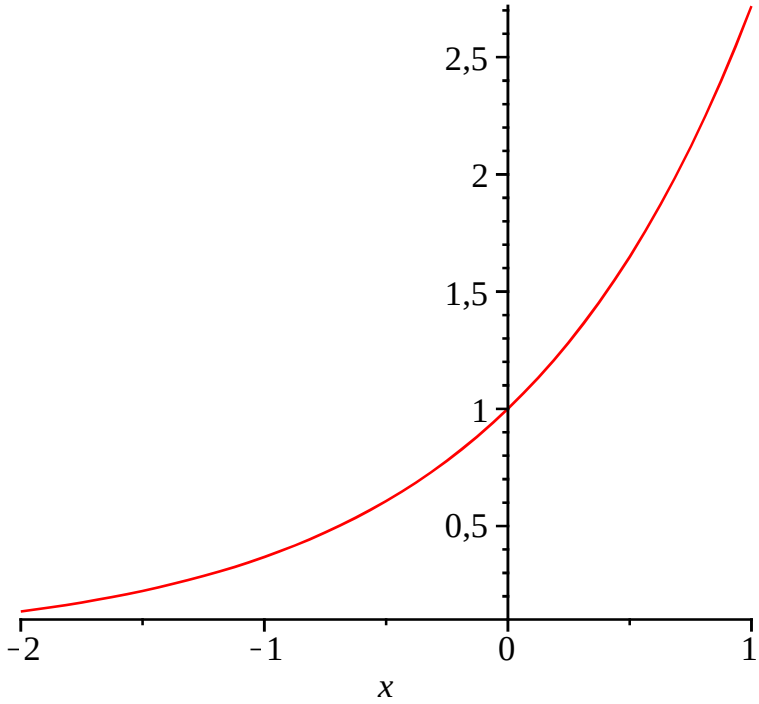
```
> restart;plot(tan(x),x=-Pi..Pi,-10..10,discont=true,
  xtickmarks=[-3.14="-p",-1.57="-p/2",1.57="p/2",3.14="p"],
  axesfont=[SYMBOL,12],ytickmarks=[-10,-5,0,5,10]);#Dans la
  police symbol, la lettre p devient π.
```



Barre de menu graphique et menu contextuel

Soit le tracé fruste :

```
> restart;plot(exp(x),x=-2..1);
```



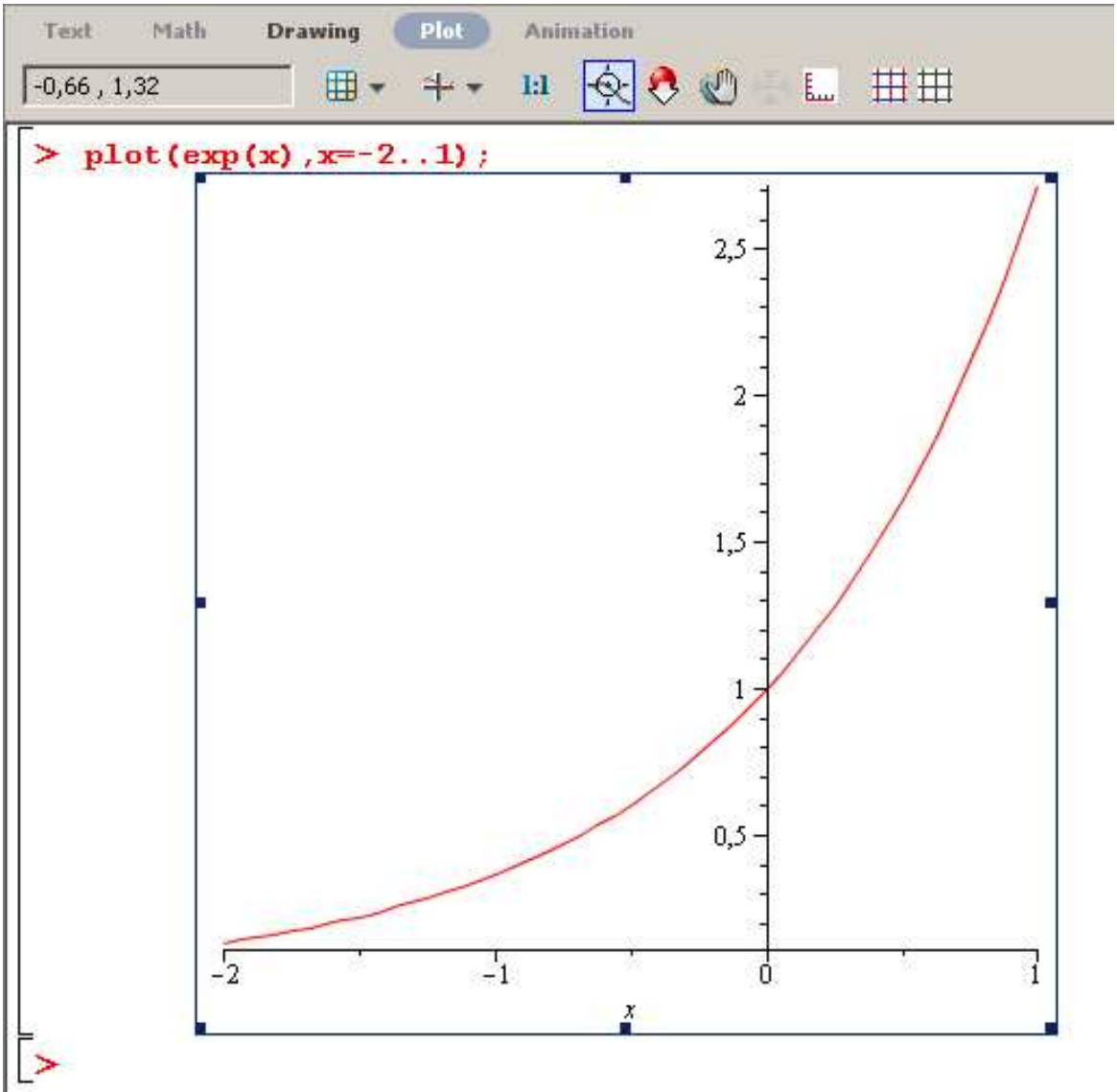
Un clic gauche sur le graphique fait apparaître une barre de menu contextuel spécifique. Dans la fenêtre de gauche, on obtient les coordonnées d'un point quelconque de la portion de plan dans laquelle se trouve le graphique (clic gauche, puis lecture des coordonnées du point situé au bout de la flèche qui se transforme en viseur dans la version 9.5). A sa droite, un certain nombre de transformations, assez primaires, sont proposées : courbes en points et non en ligne, graphique inscrit dans une boîte ou dans un système d'axes dont l'origine a pour coordonnées les valeurs minimales des intervalles considérés, suppression des axes, retour à une présentation standard.
Dans Maple 9.5, un clic droit sur le graphique fait apparaître un menu contextuel. On laisse le

lecteur explorer les possibilités offertes : type de tracés, mise au point d'une légende, sauvegarde du graphique dans plusieurs formats (EPS, Bitmap, JPG, GIF, Windows Metafile) . Cette dernière option peut être extrêmement utile quand on veut insérer un graphique Maple dans un texte écrit à partir d'un autre logiciel (Word, Scientific Word ou LaTeX).
Par ailleurs, un gadget sans grand intérêt permet d'épaissir automatiquement le tracé de la courbe lorsqu'un de ses points se trouve au centre du viseur.

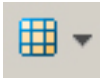
On peut l'améliorer en utilisant les options mentionnées plus haut ou, alternativement, en puisant dans les ressources de la barre contextuelle et du menu contextuel.

Barre de menu graphique

Un clic gauche sur le graphique fait apparaître une barre de menu contextuel spécifique tandis que la cartouche Plot est surlignée.



Dans la fenêtre de gauche, on obtient les coordonnées d'un point quelconque de la portion de plan dans laquelle se trouve le graphique (lecture de l'abscisse et de l'ordonnées du point situé au centre du viseur). A sa droite, un certain nombre de transformations sont proposées.

Le logo  a un menu déroulant permettant de modifier le style de la courbe : points, ligne, polygone ou contours de polygone (option par défaut).

Le logo  a également un menu déroulant qui vient fixer le style des axes : insertion

de la courbe représentative dans une boîte ou dans un système d'axes dont l'origine a pour coordonnées les valeurs minimales des intervalles considérés; suppression des axes; retour à une présentation standard.

En cliquant sur le logo  , il y a bascule du repère orthogonal standard vers un repère orthonormé. Un second clic fait revenir vers le repère standard.

La sélection du logo  autorise des effets de zoom sur des portions de la courbe ou, au contraire, de grand angle. L'utilisation conjointe de la main  permet de mettre en valeur une portion précise du graphe.

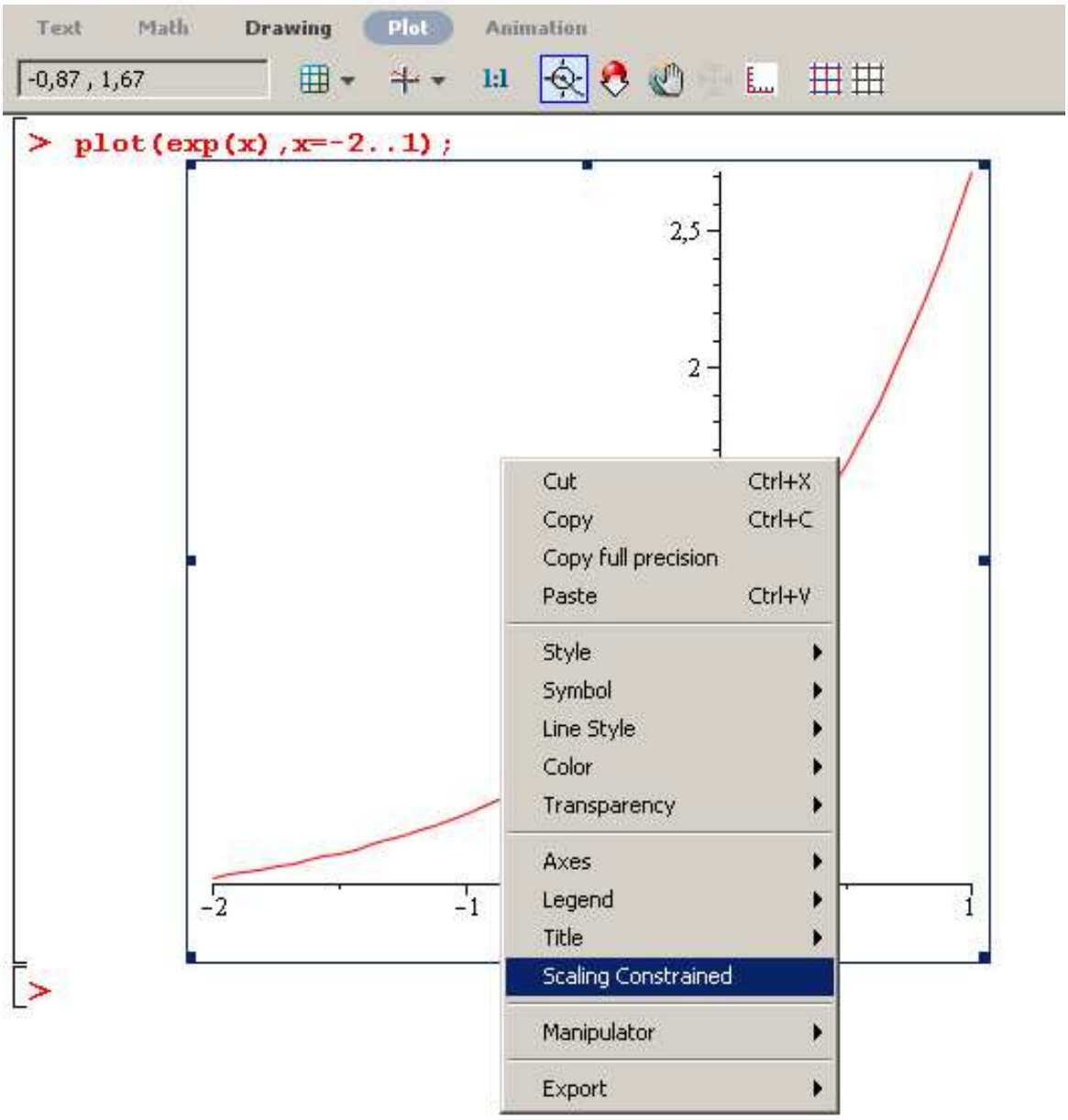
Le logo  affiche ou fait disparaître une grille dans le graphique. Ses caractéristiques sont définies par défaut mais on peut les modifier selon ses besoins à l'aide du menu jumeau (situé à gauche)  qui renvoie à une fenêtre-boîte de dialogue très complète pour paramétrer l'axe horizontal et l'axe vertical.

Enfin, avec  , on accède à une fenêtre-boîte de dialogue qui individualise la paramétrisation des axes de coordonnées : passage en graphique semi-logarithmique ou log-log; bornes inférieures et supérieures de chaque axe; choix des graduations (ticks) et des sous-graduations (subticks); et même couleur de chaque axe!

Une propriété intéressante de Maple est que les modifications apportées à un graphique par le menu graphique sont conservées en mémoire après la fermeture de la session de travail. A l'ouverture de la feuille, le graphe apparaît tel qu'il a été mis au point lors de la dernière session. Par contre, si on valide la ligne de programmation, c'est le graphique initial qui est rendu, et non le graphique corrigé. Par conséquent, un graphique amélioré par le menu graphique et qu'on veut conserver doit être impérativement sauvegardé en tant que fichier propre au moyen de Plot→Export, puis en choisissant un format d'image : Bitmap, GIF (Graphics Interchange Format), JPEG, EPS (Encapsuled PostScript), WMF (Windows MetaFile). Cette dernière option est très pratique quand on veut insérer un graphique Maple dans un document écrit avec un autre logiciel (Word, Scientific Word ou LaTeX).

Menu contextuel

Un clic droit sur un graphique fait apparaître un menu contextuel.



Les fonctionnalités proposées sont peu ou prou celles du menu graphique :

Style définit le type de représentation de la fonction : ligne ou points.

Si la fonction est représentée par une succession de points, ceux-ci peuvent être calibrés par Symbol.

Si la fonction est représentée par une ligne, celle-ci se paramètre avec Line Style (Solid= ligne pleine; Dot = ligne en pointillés; Dash = ligne en tirets; Dash Dot = ligne formée d'une succession de tirets et de points; etc...).

Comme son nom le suggère, Transparency règle la luminosité de la courbe entre 0% (luminosité maximale) et 100% (transparence totale).

Le réglage fin des axes se fait par le sous-menu de Axes, dans lequel on retrouve les paramètrages de la barre de menu graphique, y compris celles de la fenêtre Axes Properties.

Scaling Constrained renvoie le graphe dans un repère orthonormé.

Manipulator transforme la souris en viseur (Point Probe), en main/outil de déplacement (Pan) ou en flèche de zoom/dézoomage (Scale).

Export sert à sauvegarder le graphique dans un fichier séparé.

Par rapport à la barre de menu graphique, deux nouveautés - fort utiles - viennent s'ajouter : Legend et Title. Legend insère une ... légende dans le graphique à une place précisée par

l'utilisateur (en bas, en haut, à gauche ou à droite). L'activation de la sous-option Edit Legend crée un rectangle dans lequel on tape le commentaire **en mode texte ou en mode mathématique, avec possibilité d'utiliser les ressources des palettes**. Title propose deux sortes de titrage non exclusifs : titre au dessus du graphique (Title) et titre sous le graphique (Caption). Chaque sorte est matérialisée par une cartouche à l'intérieur de laquelle on tape **en mode texte ou en mode mathématique, avec possibilité de recourir aux palettes**. Après validation du titre, la cartouche disparaît.

courbes en points et non en ligne, graphique inscrit dans une boîte ou dans un système d'axes dont l'origine a pour coordonnées les valeurs minimales des intervalles considérés, suppression des axes, retour à une présentation standard.

Options élaborées

Cette section est consacrée aux raffinements tant il est vrai qu'un bon graphique exige beaucoup de travail, de rigueur et de patience. Certaines options sont disponibles avec **`plot`**, d'autres ne sont accessibles qu'en chargeant le paquetage **`plots`**.

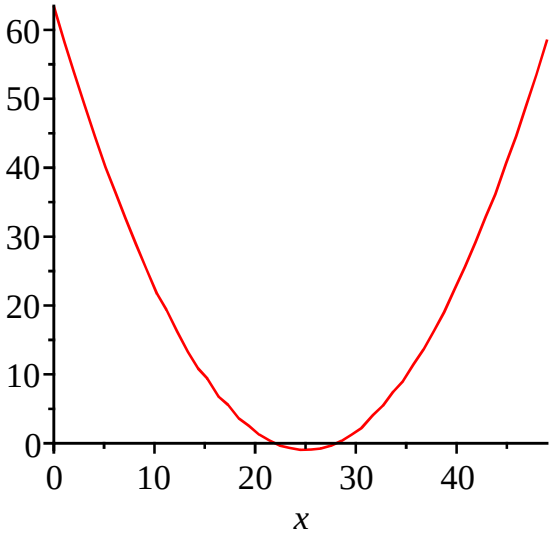
Amélioration de la précision du tracé

L'algorithme utilisé par Maple pour tracer une courbe est basé sur l'idée suivante : l'intervalle des valeurs de la variable indépendante que définit l'utilisateur est divisé en un minimum de 49 segments non nécessairement égaux, ce qui signifie qu'au moins 50 points non nécessairement équidistants sont choisis dans l'intervalle. Les valeurs prises par l'expression ou la fonction sont alors calculés et au moins 50 points du plan et au plus 200 sont ainsi repérés, puis tracés et reliés deux à deux par des segments de droite. Lorsque Maple flaire la présence d'une singularité du type point de rebroussement ou point d'arrêt, il concentre les subdivisions autour de ce point de manière à rendre la courbe la plus lisse possible.

Il peut arriver que l'échantillonnage sur 50 points s'avère insuffisant pour tracer correctement la fonction. On peut alors forcer Maple à accroître le nombre de points par l'option **`numplot=N`**, où N est un entier supérieur à 50. On peut aussi demander une meilleure résolution, c'est à dire accroître le nombre de calculs autour des points singuliers et ceux-là seulement, à l'aide de **`resolution=N`** où N est un entier supérieur à 200 (qui est la résolution par défaut).

L'exemple suivant montre l'intérêt de cette option. La fonction étudiée est une parabole augmentée d'un cosinus.

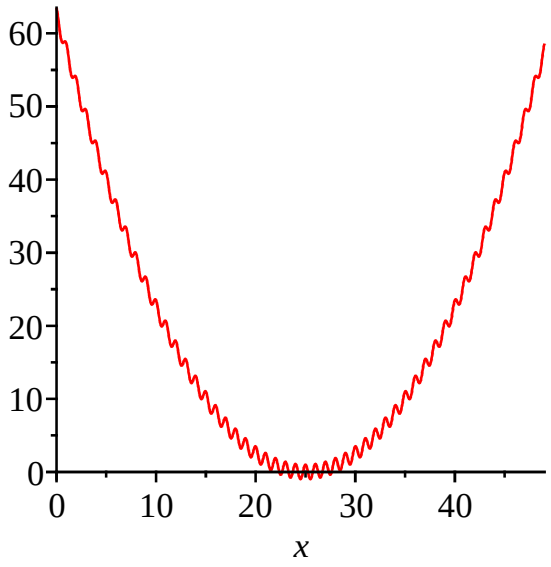
```
> plot((x-25)^2/10+cos(2*Pi*x),x=0..49);
```



Cette représentation est douteuse puisqu'on ne voit pas apparaître clairement l'effet induit par la présence d'une fonction trigonométrique. C'est pourquoi on augmente le nombre de points

à évaluer.

```
> plot((x-25)^2/10+cos(2*Pi*x),x=0..49,numpoints=2000);
```

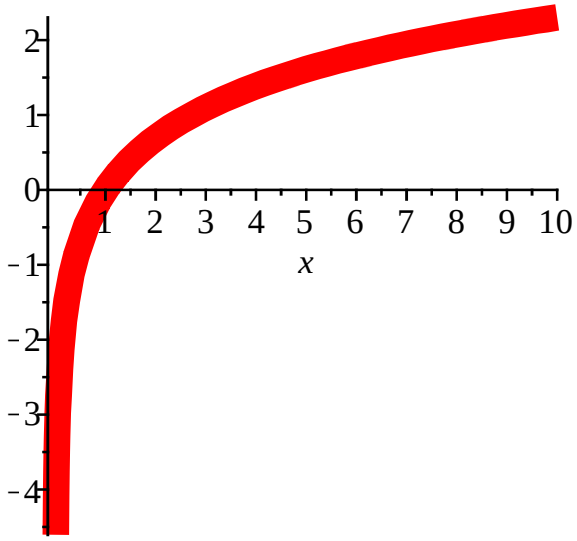


Tout est rentré dans l'ordre.

Epaisseur d'une courbe

L'épaisseur d'une courbe est définie par l'option **thickness** dont la syntaxe est **thickness=t** avec **t** nombre entier naturel, dont la valeur par défaut est 0.

```
> plot(log(x),x=0.01..10,thickness=10);
```



Styles de tracé

Le style d'une courbe est défini par l'option **linestyle** dont la syntaxe est **linestyle=t** avec **t** nombre réel compris entre 1 et 7.
L'option **linestyle=1** peut être remplacée par **linestyle=SOLID** (ligne pleine).
L'option **linestyle=2** peut être remplacée par **linestyle=DOT** (ligne en pointillés).
L'option **linestyle=3** peut être remplacée par **linestyle=DASH** (ligne en tirets).
L'option **linestyle=4** peut être remplacée par **linestyle=DASHDOT** (succession de tirets et de pointillés).

L'option **linestyle=5** peut être remplacée par **linestyle=LONGDASH** (succession de tirets longs).
L'option **linestyle=6** peut être remplacée par **linestyle=SPACEDASH** (succession de tirets et d'espaces).
L'option **linestyle=7** peut être remplacée par **linestyle=SPACEDOT** (succession de points et d'espaces).

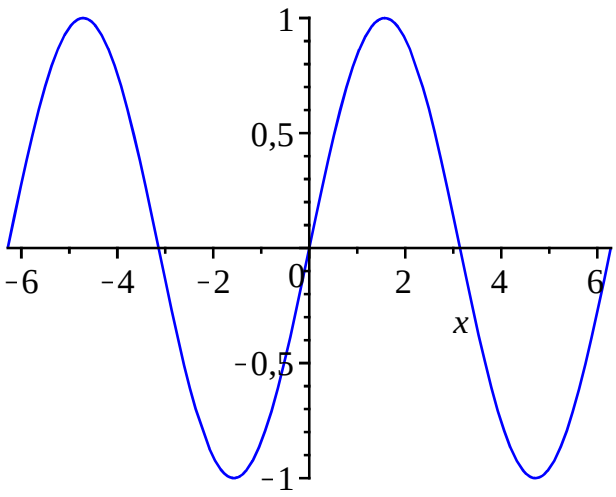
Les possibilités sont plus riches que dans les options du menu graphique ou le menu contextuel. De plus, le style, une fois programmé, est appliqué durablement.

Colorisation d'une courbe

Il existe quatre manières de coloriser une courbe. Soit on écrit directement une couleur par son nom anglais, soit on retient le mode RGB (Red Green Blue, soit en français RVB pour Rouge Vert Bleu), soit les modèles plus élaborés HUE (en français : Teinte) et HSV ("Hue, Saturation, Value", appelé en français TSL pour "Teinte, Saturation, Luminosité"). Dans tous les cas, l'option est annoncée par **colour** (pour les fans de la Grande Bretagne) ou **color** (pour les amoureux des Etats-Unis).

Le plus simple est de spécifier une couleur, ici bleu (blue) :

```
> restart; plot(sin(x),x=-2*Pi..2*Pi,color=blue);
```

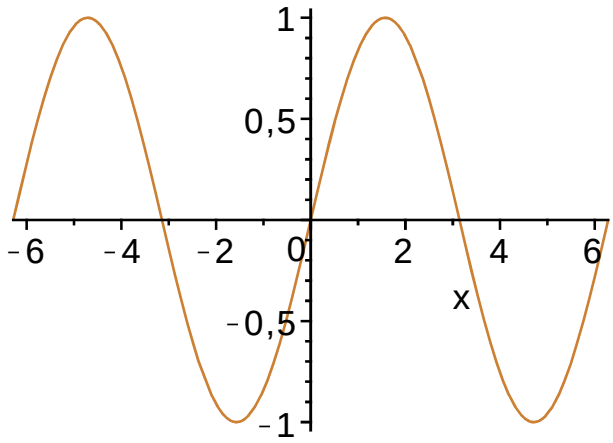


Encore faut-il connaître l'anglais! En fait, il faut choisir son bonheur dans la liste impressionnante donnée dans la page d'aide plot,colornames. En pratique, les couleurs suivantes suffiront largement :

```
aquamarine black  blue      navy   coral cyan
brown      gold   green     gray   grey  khaki
magenta    maroon orange    pink   plum  red
sienna     tan    turquoise violet wheat white
yellow
```

Plus technique : l'option **COLOR(RGB,r,g,b)** qui permet de mélanger du rouge en quantité **r**, du vert en quantité **g** et du bleu en quantité **b** pour obtenir sa couleur préférée.Chaque dose de couleur primaire doit être un nombre réel compris entre 0 (absence de couleur) et 1 (maximum d'intensité).

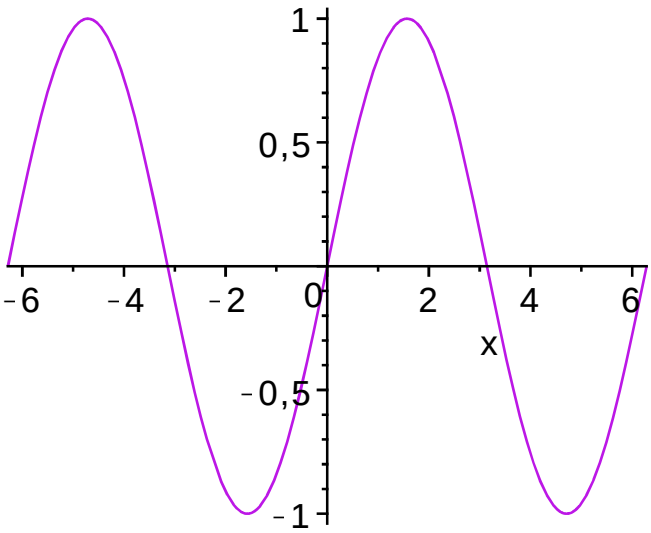
```
> plot(sin(x),x=-2*Pi..2*Pi,color=COLOR(RGB,0.8,0.5,0.2));
```



Encore plus technique, parce que méconnu : l'option **COLOR(HUE,h)** permet de fixer une teinte par un nombre **h** compris entre 0 et 1. Il s'agit en fait de fixer une couleur par sa position relative sur le cercle chromatique partant du rouge - valeur 0 - et retournant au rouge - valeur 1 - en passant par le jaune, vert, cyan, bleu et magenta.

Enfin, le fin du fin : l'option **COLOR(HSV,h,s,v)** qui permet de fixer la couleur dans le standard HSV en donnant un nombre **h** compris entre 0 et 1 (qui est une teinte sur le cercle chromatique), sa saturation **s** par un nombre dompris entre 0 et 1 (1 correspond à une couleur hyper saturée; 0 à une couleur complètement désaturée, grise) et sa luminosité **v** par un nombre compris entre 0 et 1 (0 donne du noir car il y a absence de lumière; 1 attribue la luminosité maximale que la couleur retenue peut accepter sans devenir du blanc). A noter que la commande **HUE(h)** remplace **COLOR(HSV,h,0.9,1.0)**.

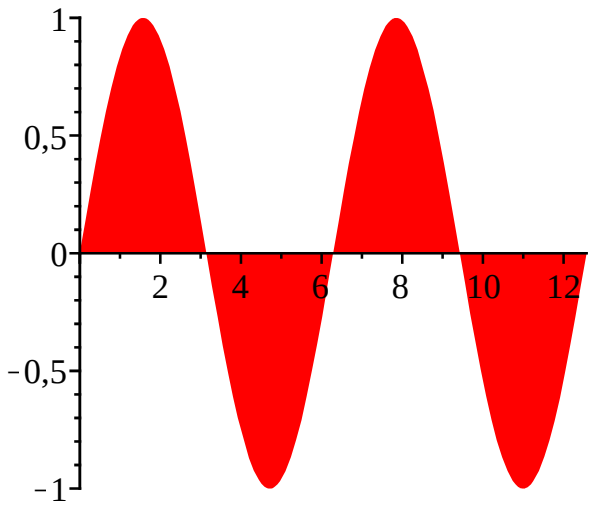
```
> plot(sin(x),x=-2*Pi..2*Pi,color=COLOR(HSV,0.8,0.9,0.9));
```



Colorisation de surfaces

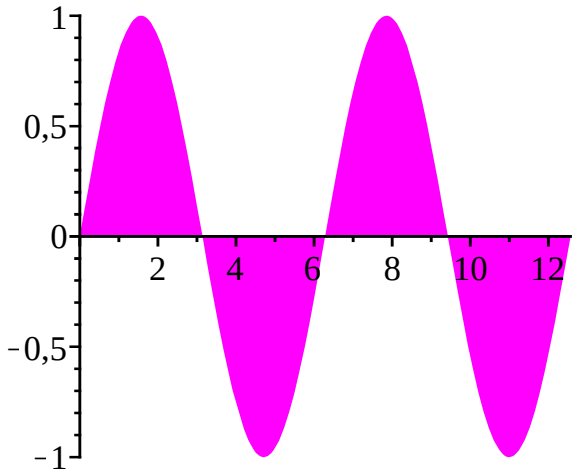
Colorer une surface se fait avec l'option **filled** disponible dans la commande universelle **plot**. La syntaxe **filled=true** met en valeur la surface comprise entre la courbe représentative de la fonction et l'axe des abscisses.

```
> restart;
  sinus:=plot(sin,0..4*Pi,filled=true):sinus;
```



La couleur de la surface est définie par l'option **color=nom de couleur**. Bien entendu, rien n'interdit d'utiliser un modèle de couleur plus élaboré.

```
> restart;
  sinus:=plot(sin,0..4*Pi,filled=true,color=magenta):sinus;
```

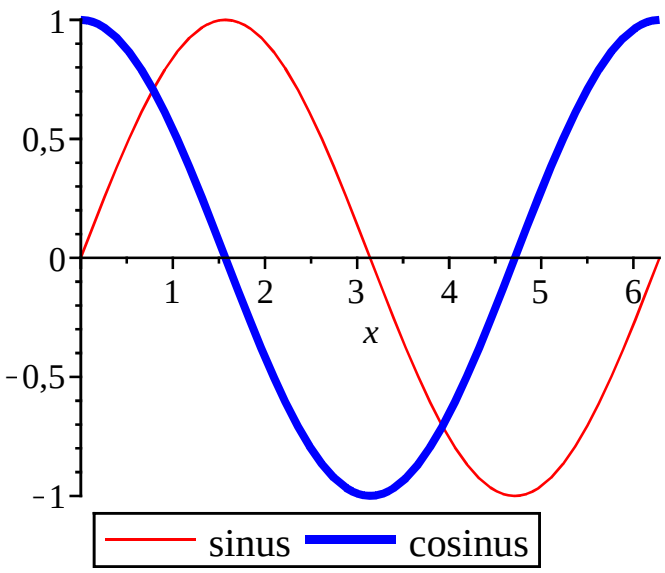


De fait, toutes les options de plot sont mobilisables, en particulier **transparency** qui atténue la luminosité de la couleur.

Plusieurs tracés dans le même graphique (bis) et légende

Pour tracer plusieurs courbes représentatives dans le même graphique et les distinguer, on donne en premier argument de **plot** une liste d'expressions ou de fonctions et les options (de couleur, d'épaisseur du trait, légende) s'égalisent à des listes dont les composantes particularisent les tracés de chaque expression ou fonction **suivant l'ordre où elles apparaissent dans le premier argument**. On peut alors facilement lever la contrainte de choix automatique des couleurs par Maple quand il doit tracer plusieurs fonctions dans un même graphe. Il suffit que l'utilisateur choisisse ses propres couleurs de lignes avec **color** (ou **colour**) et, s'il le désire, ajoute une légende grâce à l'option **legend** qui accepte les chaînes de caractères écrites entre guillemets ou typographiées au moyen de **typeset**.

```
> restart;plot([sin(x),cos(x)],x=0..2*Pi,color=[red,blue],
  thickness=[1,3],legend=["sinus","cosinus"]);
```

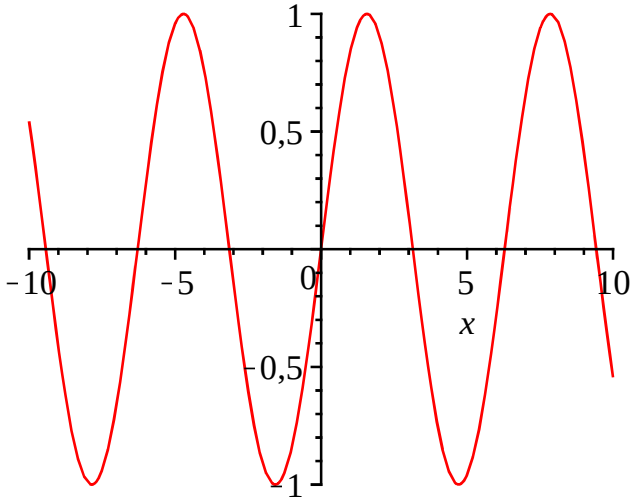


Plusieurs tracés dans le même graphique (ter)

La commande **display** est disponible dans le paquetage (package) **plots**. On la charge soit par **plots[display]** soit, si on dispose d'une bonne mémoire vive, par **with(plots)**.

Fondamentalement, elle permet d'afficher un graphique construit avec **plot** dont on a au préalable attribué un label. Il convient donc de "sauvegarder" un graphe en lui assignant un nom de guerre - ne commençant pas par un chiffre et ne comprenant pas l'esperluète (&)) et en terminant la commande par "deux points" et non "point virgule" faute de quoi Maple affiche des lignes de coordonnées. La seconde appelle le module **plots** si cela n'a pas été déjà fait et utilise **display** comme suit : **display(label du graphique, éventuelles options)**.

```
> graf1:=plot(sin(x),x=-10..10):#construction d'un
  graphique avec plot
  with(plots):#chargement du paquetage plots
  display(graf1);#utilisation de display
```

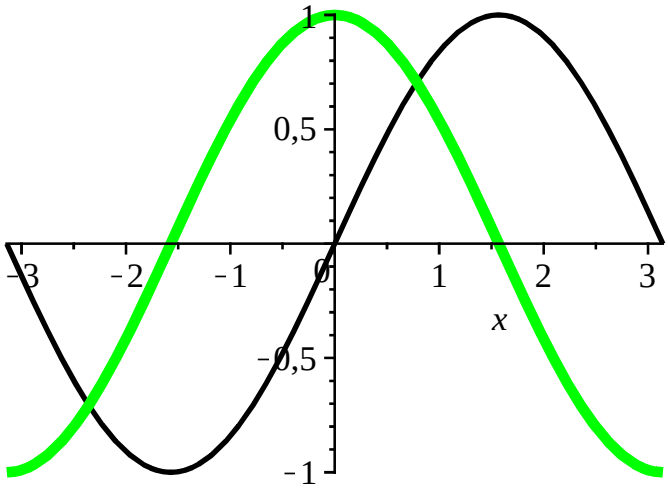


Ce sont les options de **display** qui sont intéressantes car elles permettent d'améliorer sensiblement la présentation d'un graphique comme on le verra plus loin. En particulier, on

pourra insérer des commentaires à l'intérieur du graphe.

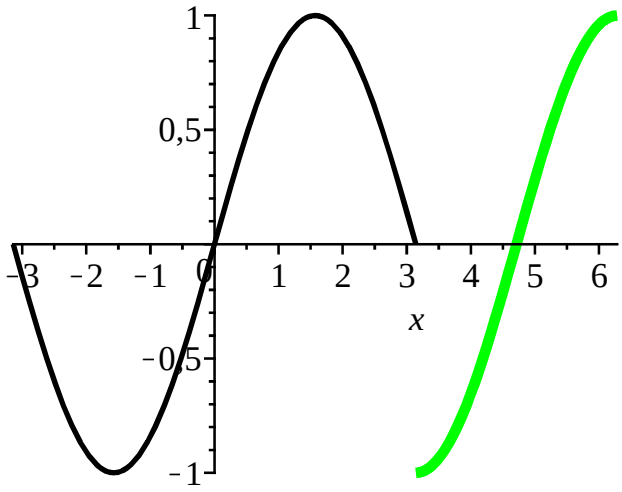
Mais en pratique l'utilisation la plus fréquente de **display** se fait pour réunir/superposer plusieurs tracés dans la même figure. La syntaxe est alors **display([graphe1, graphe2,..,grapheN])**.Le résultat est une superposition des graphes élémentaires dont on a géré séparément les caractéristiques, par exemple les couleurs et l'épaisseur.

```
> graf1:=plot(sin(x),x=-Pi..Pi,color=black,thickness=2):
  graf2:=plot(cos(x),x=-Pi..Pi,color=green,thickness=4):
  display([graf1,graf2]);
```



La commande **display** gère automatiquement les ensembles de départ et d'arrivée, sauf dans le cas où la borne inférieure est moins l'infini; et la borne supérieure plus l'infini .

```
> graf3:=plot(cos(x),x=Pi..2*Pi,color=green,thickness=4)
  :display([graf1,graf3]);
```

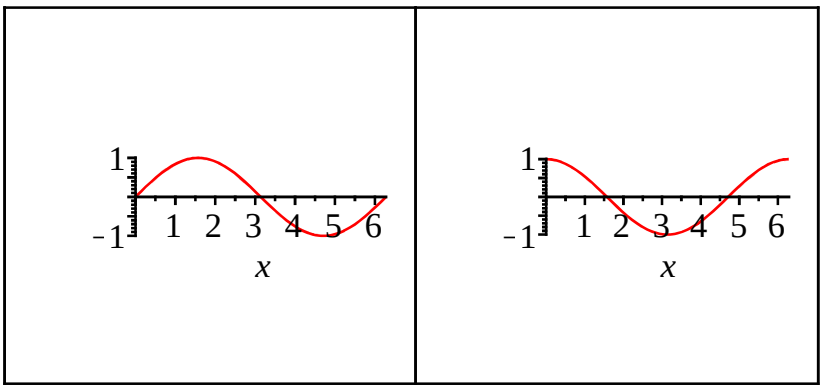


Plusieurs graphiques dans un même cadre

La commande **display** du module **plots** permet de présenter plusieurs graphiques dans un même cadre grâce à l'option **Array** dont la syntaxe basique est **Array([graphe1, graphe2,...,grapheN])**. Par défaut, Maple place les graphiques côte à côte en ligne :

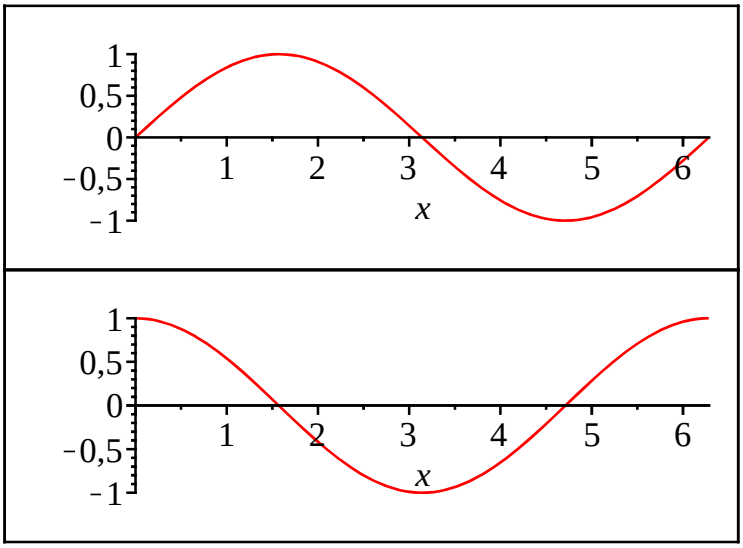
```
> restart;
  with(plots):
  sinus:=plot(sin(x),x=0..2*Pi):
```

```
cosinus:=plot(cos(x),x=0..2*Pi):
display(Array([sinus,cosinus]),scaling=constrained);
```



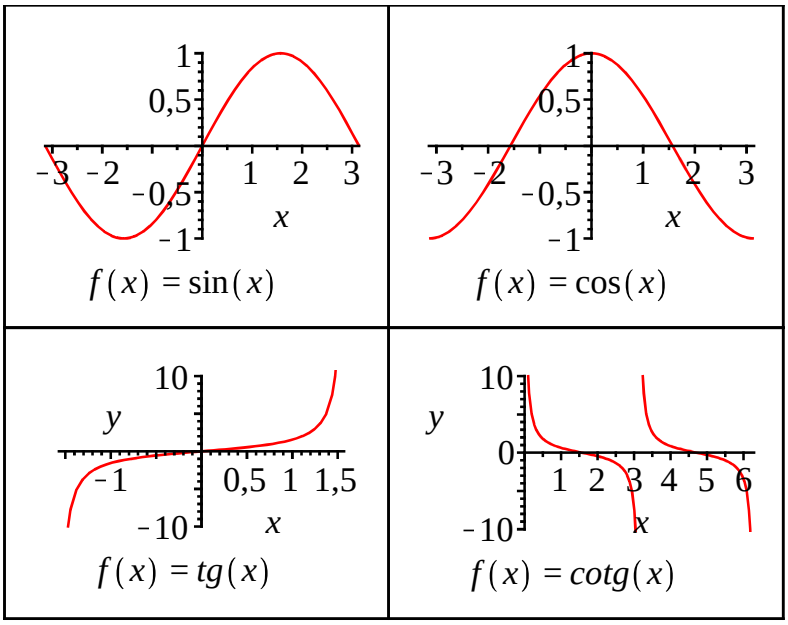
En règle générale, il est préférable de déclarer le nombre de lignes et de colonnes. Par exemple, pour 2 lignes et une colonne :

```
> display(Array(1..2,1..1,[[sinus],[cosinus]]));
```



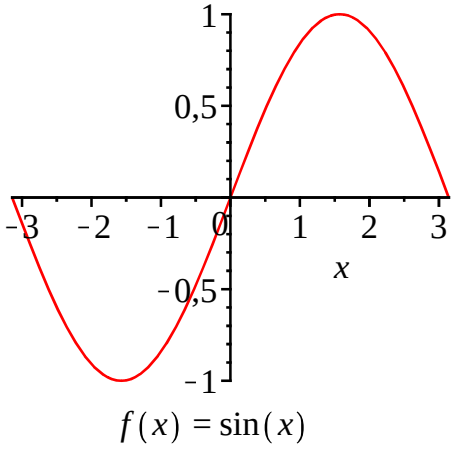
Et pour deux lignes et deux colonnes :

```
> restart;
  with(plots):
  sinus1:=plot(sin(x),x=-Pi..Pi,caption=typeset(f(x)=sin(x)
  )):
  cosinus1:=plot(cos(x),x=-Pi..Pi,caption=typeset(f(x)=cos
  (x))):
  tangente1:=plot(tan(x),x=-Pi/2..Pi/2,y=-10..10,caption=
  typeset(f(x)=tg(x))):
  cotangente1:=plot(cot(x),x=0..2*Pi,y=-10..10,discont=
  true,caption=typeset(f(x)=cotg(x))):
  display(Array(1..2,1..2,[[sinus1,cosinus1],[tangente1,
  cotangente1]]));
```



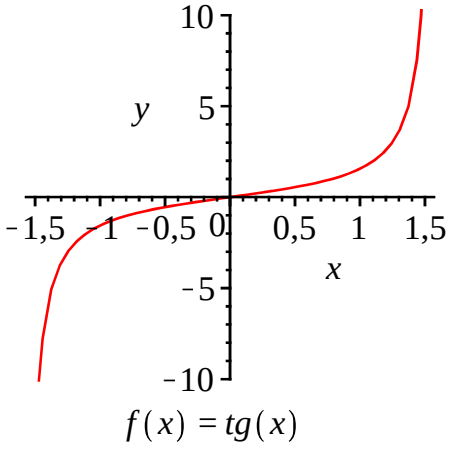
Si on désire insérer du code Maple à l'intérieur du ou des cadres, on commence par créer un tableau par Insert→Table en précisant le nombre de lignes (rows) et de colonnes (columns) puis on frappe la commande dans chaque sous-cadre en mode Input Maple (au moyen de Insert→Maple Input ou du raccourci clavier Ctrl+m). L'output est affiché dans le sous-cadre après validation.

```
> plot(sin(x),x=-Pi..Pi,
  caption=typeset(f(x)=sin
  (x)));
```




```
>
```

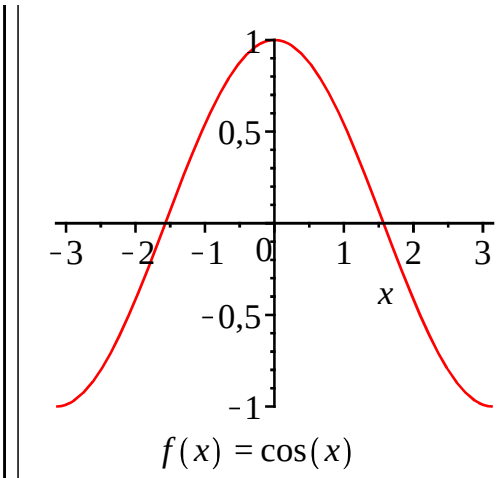
```
> plot(tan(x),x=-Pi/2..
  Pi/2,y=-10..10,caption=
  typeset(f(x)=tg(x)));
```




```
>
```

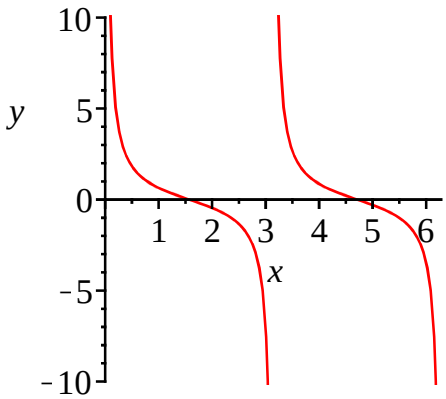
```
> plot(cos(x),x=-Pi..Pi,
  caption=typeset(f(x)=cos
  (x)));
```

```
> plot(cot(x),x=0..2*Pi,y=
  -10..10,discont=true,
  caption=typeset(f(x)=
  cotg(x)));
```



$f(x) = \cos(x)$

>



$f(x) = cotg(x)$

>

Insertion de texte dans le graphique

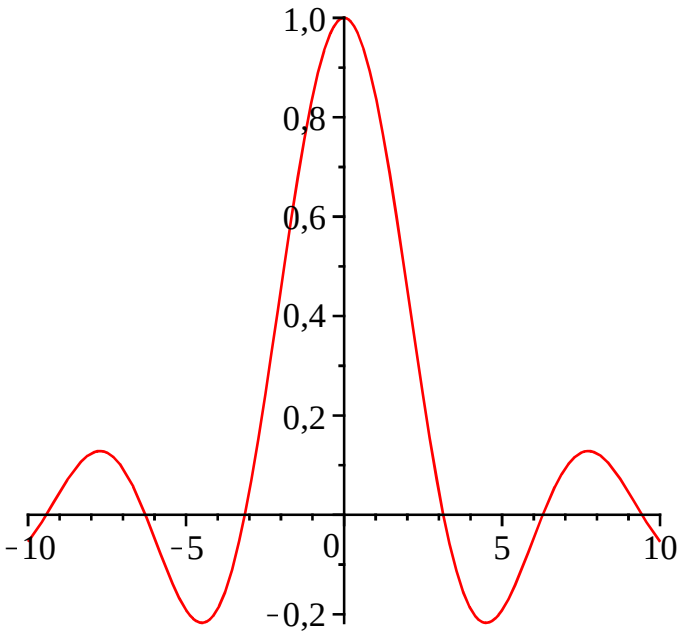
Les options de **display** sont grosso modo les mêmes que celles de **plot**. Elles sont en fait très précieuses car elles permettent d'améliorer sensiblement la présentation d'un graphique composé voire de réaliser des présentations impossibles sans elles. En particulier, on peut insérer des annotations à l'intérieur d'un graphique en regroupant le tracé de la courbe proprement dite et les annotations à insérer dans le graphique final. Comme de telles annotations sont généralement des chaînes de caractères, il faut employer la commande **textplot** du paquetage **plots**, dont la syntaxe est (le paquetage plots étant préalablement chargé) :

textplot({[abscisse, ordonnée,"texte1"],..,[abscisse,ordonnée, "texteN"]},options éventuelles]

les options étant celles de **plot** à quoi il faut ajouter **align={argument1, argument2}**, **argument1** étant **ABOVE** (au dessus du point de référence du texte) ou **BELOW** (en dessous) et **argument2** étant **RIGHT**(à droite du point de référence du texte) ou **LEFT** (à gauche).

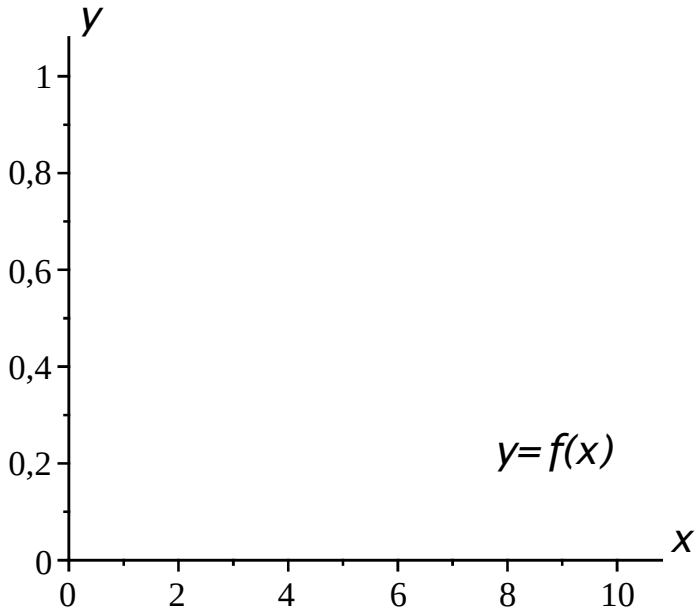
Dans l'exemple suivant, on insère l'annotation $y = f(x)$ au dessus de la courbe, l'annotation x au bout de l'axe des abscisses et l'annotation y en haut de l'axe des abscisses. On commence par tracer la courbe de la fonction, ici $y = \frac{\sin(x)}{x}$.

```
> restart;
  with(plots):
  graf1:=plot(sin(x)/x,x=-10..10,labels=["",""]):graf1;
```



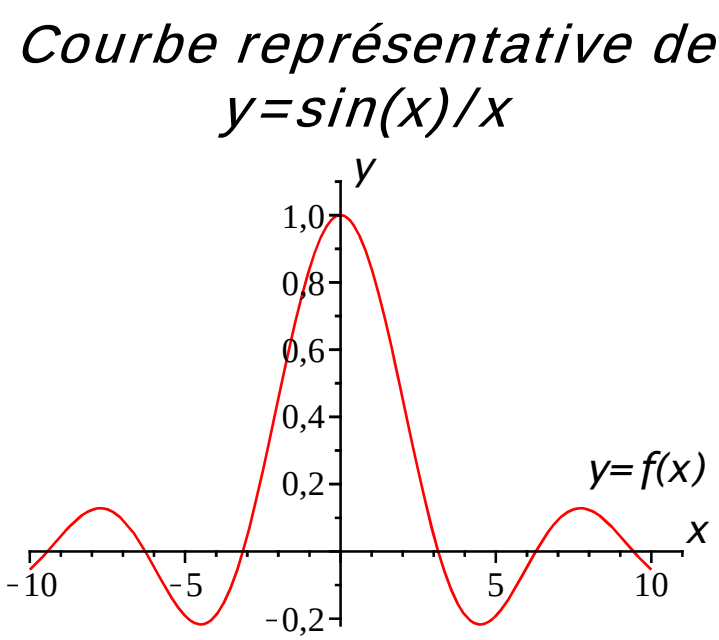
A l'aide du viseur, on repère les endroits où se situeront les annotations et on recopie leurs coordonnées dans les listes de **textplot**. Le texte est ici une simple chaîne de caractères en police Helvetica oblique de taille 12 points, mais on aurait pu utiliser les ressources de **typeset**.

```
> graf2:=textplot({[7.5625,0.1817,"y=f(x)"],[10.8125,0,
  "x"],[0,1.08,"y"]},align={ABOVE,RIGHT},font=[HELVETICA,
  OBLIQUE,12]):graf2;
```

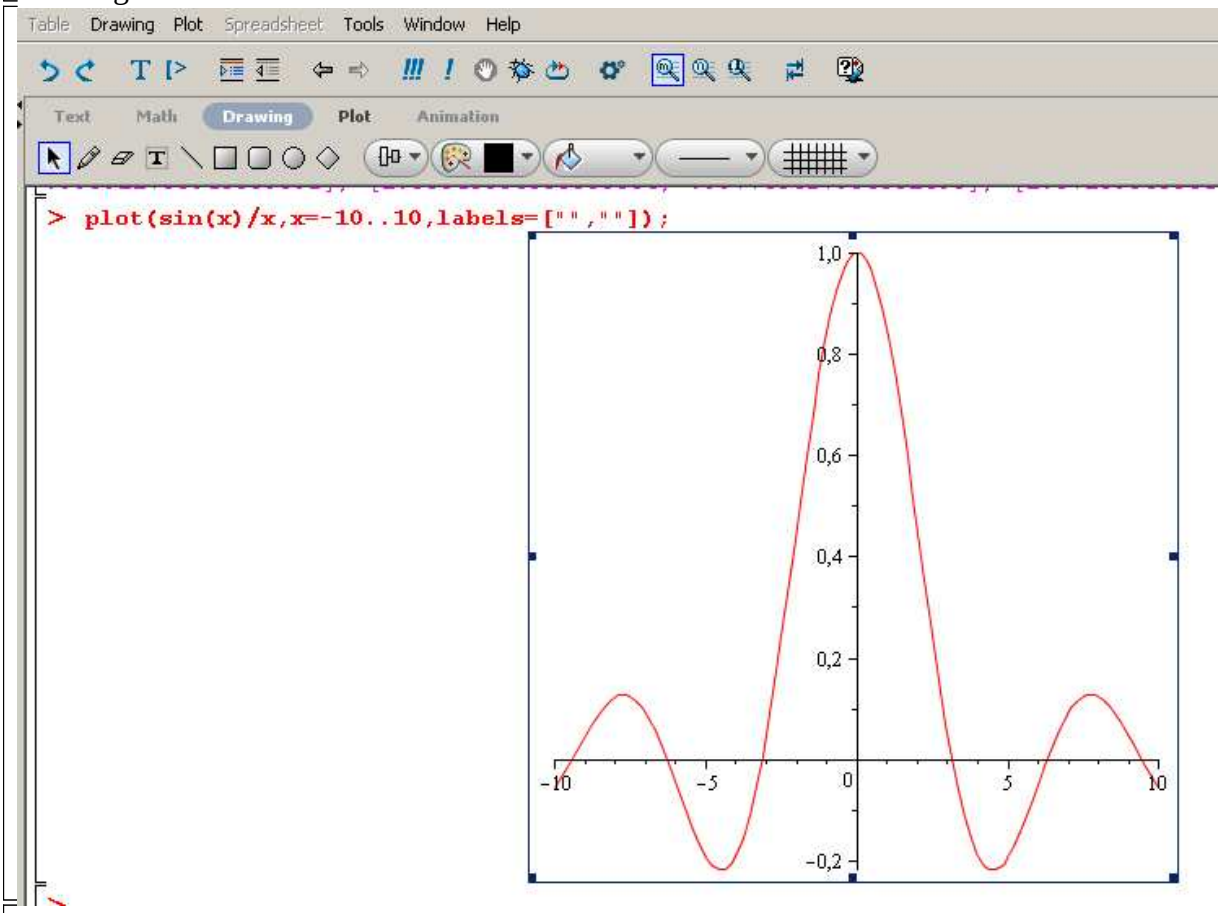


Il ne reste plus qu'à superposer les deux graphes et améliorer la présentation par un titre.

```
> display([graf1,graf2],view=[-10..11,-0.22..1.1],title=
  "Courbe représentative de y=sin(x)/x",titlefont=
  [HELVETICA,BOLDOBLIQUE,16]);
```



Avec les dernières versions de Maple, il existe une méthode bien plus simple pour annoter un graphique. Elle consiste à exploiter les fonctionnalités de la barre de menu graphique Drawing à laquelle on accède en sélectionnant un graphique puis en cliquant sur la cartouche Drawing.



L'outil texte est accessible parle bouton  . En cliquant la barre d'insertion dans le graphique, un rectangle apparaît et on tape son annotation en mode texte ou mode math avec possibilité d'insérer des signes de palettes. Pour valider l'annotation, il faut cliquer à l'extérieur du rectangle. En cas de remord, un nouveau clic gauche sur l'annotation fait réapparaître le rectangle et son contenu. Les poignées permettent de réajuster les dimensions du rectangle. Le viseur multidirectionnel permet de le déplacer n'importe où dans le cadre du

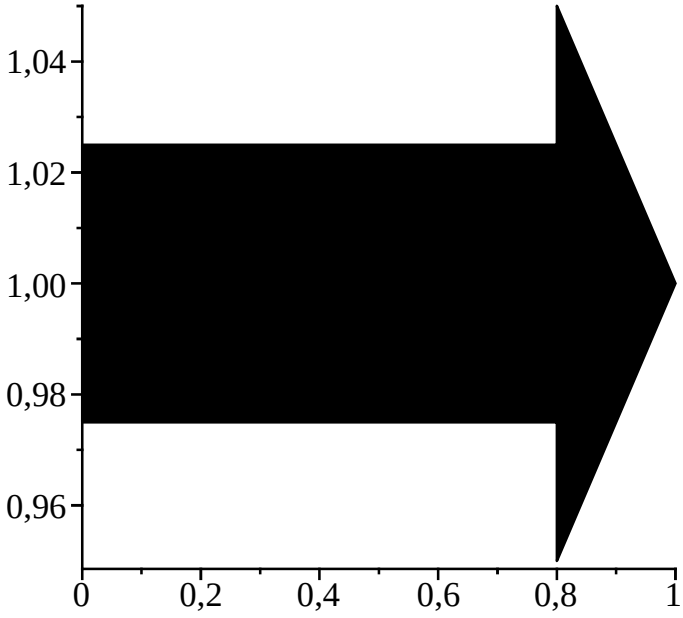
graphique. Le texte lui-même peut être modifié. Toutes les modifications sont validées par un simple clic extérieur. Le résultat est enregistré définitivement en mémoire même si les annotations n'ont pas été explicitement programmées.

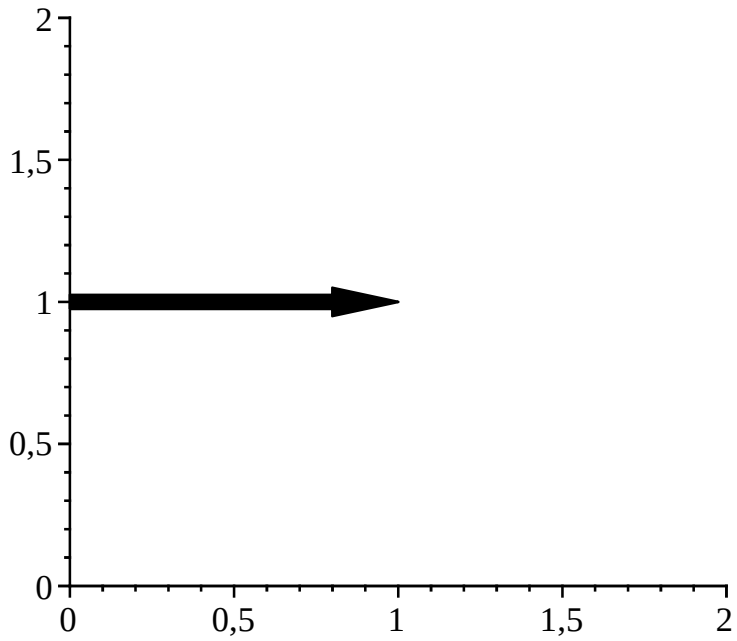
En examinant la barre contextuelle du menu Drawing, on constate que d'autres outils que l'insertion de texte sont disponibles. Il ne faut pas hésiter à les explorer et les exploiter.

Flèches

L'insertion de flèches (en anglais : arrows) dans un graphique est d'usage courant en économie afin de souligner le passage d'un équilibre à un autre ou de préciser le déplacement de courbes dans le plan lors d'une modification d'une variable exogène. Le paquetage **plots** propose la commande **arrow** dont la syntaxe basique est **arrow([x1,y1],[x2,y2])** où la liste **[x1,y1]** fournit les coordonnées dans le plan de la base de la flèche et la liste **[x2,y2]** fournit les coordonnées du sommet.

```
> restart;
  with(plots):
  arrow([0,1],[1,0]);#tracé d'une flèche avec les
  paramètres par défaut
  display(%,view=[0..2,0..2]);#même flèche dans une autre
  fenêtre graphique
```

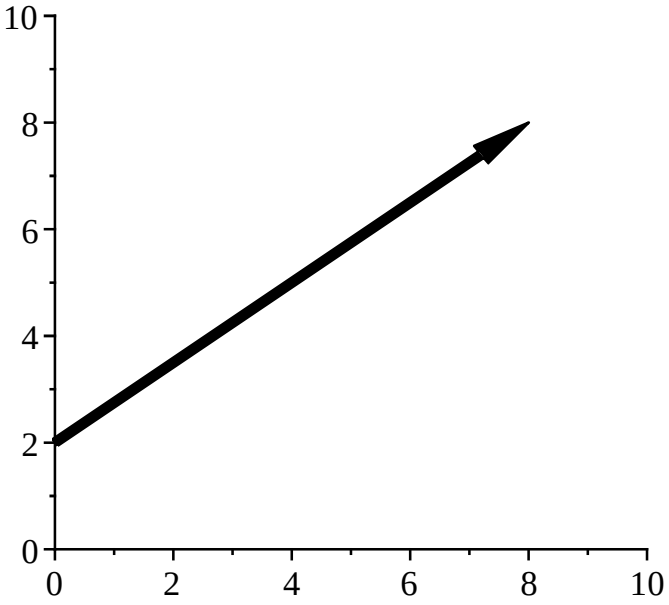




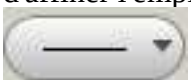
De nombreuses options permettent d'améliorer la présentation d'une flèche; les principales sont :
1- sa forme est précisée par **shape= harpoon** (le corps et la tête de la flèche sont formés d'un seul segment), **arrow** (le corps est un segment et la tête est formée de deux segments symétriques), **double_arrow** (option par défaut, comme dans l'exemple ci-dessus).
2- la longueur du corps de la flèche est précisée par **lenght=nombre**
3- l'épaisseur du corps de la flèche est précisée par **width=nombre** (par défaut, ce nombre est fixé au 1/20° de la longueur)
4- la longueur de la tête est précisée par **head_lenght=nombre** (par défaut, ce nombre est égal au 1/5° de la longueur)

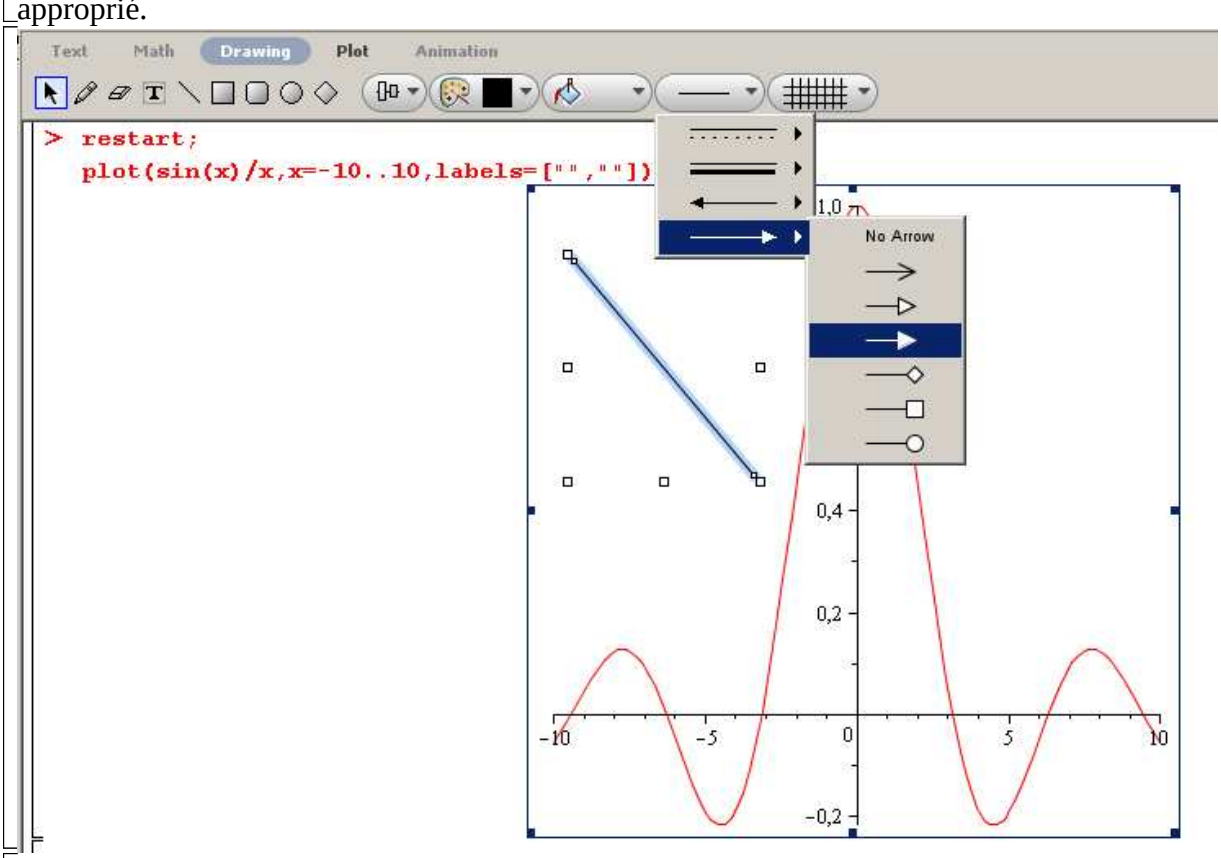
Une seconde possibilité de tracer des flèches est d'appeler la commande **arrow** du paquetage **plottools**. Sa syntaxe est **arrow(b,s,ec,et,tt,sh)** où **b** est la liste des deux coordonnées de la base de la flèche, **s** est la liste des coordonnées de son sommet, **ec** désigne l'épaisseur du corps de la flèche, **et** l'épaisseur de sa tête et **tt** la longueur relative de la tête par rapport au corps de la flèche. Ces 5 arguments sont obligatoires. En revanche, l'argument **sh**, abréviation de shape, est optionnel. Il peut prendre les valeurs **harpoon**, **arrow** ou **double_arrow**. En sus, la commande accepte les options usuelles de **plot** (par exemple, **color**). La visualisation de la flèche se fait par **display**.

```
> with(plottools):
  flech:=arrow([0, 2],[8, 8],.15,.4,.1,color = black):
  display(flech,view=[0..10,0..10]);
```



Comme on le constate à l'usage, programmer une flèche ne va pas de soi. C'est pourquoi on préférera sans doute recourir aux outils dédiés du menu contextuel Drawing. Pour tracer une flèche, il faut préalablement sélectionner l'outil "ligne droite" en cliquant sur le bouton 

puis tracer un segment du point de départ au point d'arrivée. En double cliquant sur le point d'arrivée, le segment est affiché dans un rectangle suggéré par des poignées (qui permettent d'affiner l'emplacement). Le rectangle restant sélectionné, on appuie sur le bouton  pour accéder à un sous-menu gérant les styles de ligne. Le sous-menu des flèches développe lui-même un sous-sous-menu où on sélectionne le type de flèche approprié.



Après validation, le segment devient une flèche que l'on peut d'ailleurs continuer à améliorer. Tout comme l'insertion de texte, l'insertion de flèche par le menu Drawing est définitive tant

qu'on ne l'a pas volontairement supprimée (par sélection du rectangle puis utilisation de la touche Supp du clavier).

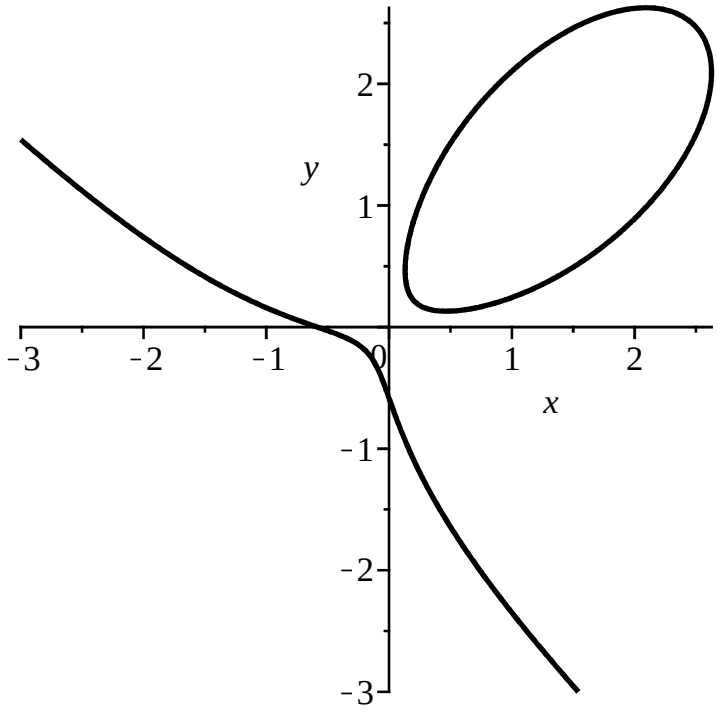
Courbes implicites

L'équation cartésienne $f(x, y) = 0$ définit implicitement y comme une fonction de x si $\frac{\partial}{\partial x} f(x, y) \neq 0$ ou x comme une fonction de y si $\frac{\partial}{\partial y} f(x, y) \neq 0$. Quand les deux dérivées partielles s'annulent, on est en présence d'un point singulier. Dans les autres cas, y est fonction implicite de x : $y = \varphi(x)$ et on appelle sa courbe représentative une courbe implicite.

Les courbes implicites sont fréquentes en microéconomie. Par exemple, les courbes d'iso-utilité (isoquants) et les courbes d'isoproduction sont des courbes implicites obtenus respectivement à partir de la fonction d'utilité et de la fonction de production pour un niveau d'utilité et un niveau d'output fixé.

Le module **plots** charge le sous-module **implicitplot** qui permet de tracer correctement les courbes implicites à condition de ne pas être en présence d'un point singulier. La syntaxe est la suivante : **implicitplot(équation f(x,y)=0,x=a..b,y=c..d,options)** où $f(x, y) = 0$ peut être remplacé par $f(x, y)$ car l'égalisation à 0 est sous-entendue. Maple accepte également $f(x, y) = p$, p étant un paramètre réel. Toutes les options de **plot** sont acceptées, la plus utile pour la finesse du tracé étant **grid[M,N]** . Cette dernière affine la grille d'analyse des changements de signe de la fonction au sens où cette grille possède M points horizontaux et N points verticaux. Par défaut M et N valent 25 chacun.

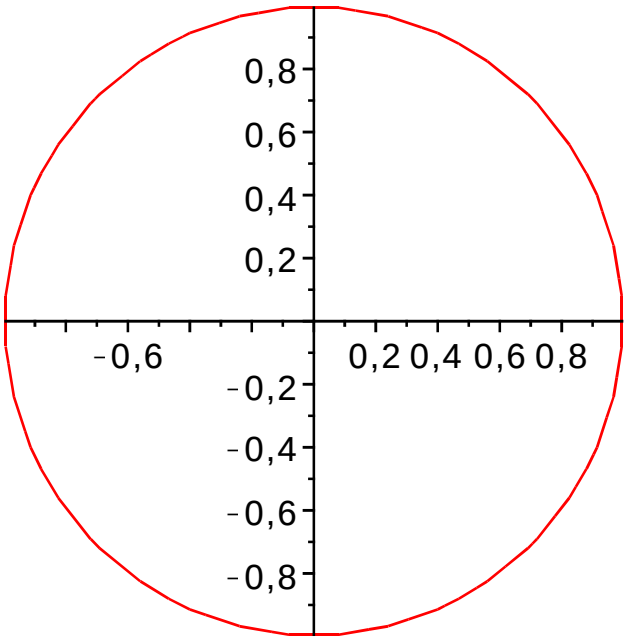
```
> restart;
  with(plots):#chargement du paquetage plots
  implicitplot(x^3+y^3-5*x*y+1/5=0,x=-3..3,y=-3..3,grid=[200,
  200],color=black,thickness=2);
```



Il faut noter que le premier argument peut être une fonction, auquel cas le second et le troisième arguments ne précisent pas de nom de variable. La syntaxe est donc :
implicitplot(f,a..b,c..d,options) où il est sous-entendu que la fonction s'égalise à 0.

```
> f:=(x,y)->x^2+y^2-1:
```

```
implicitplot(f,-2..2,-2..2);
```



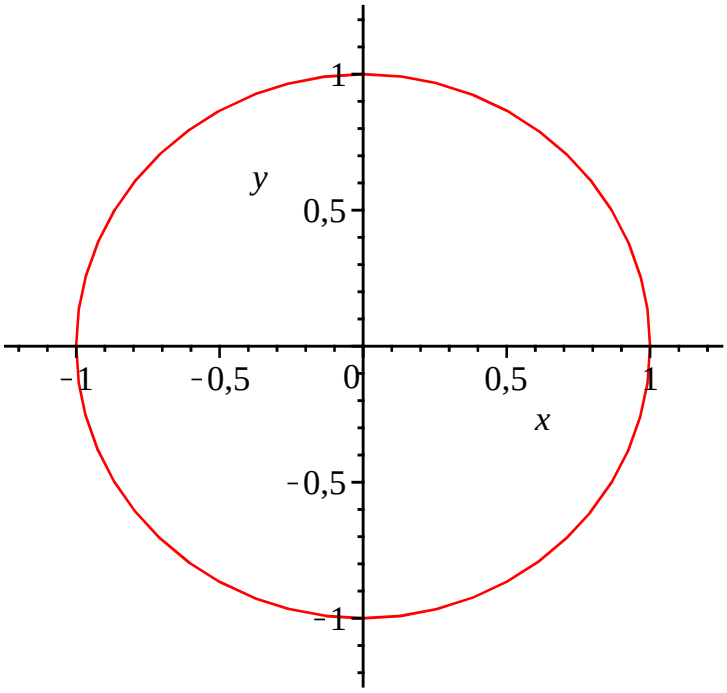
Courbes paramétrées en coordonnées cartésiennes

Lorsque la procédure basée sur **implicitplot** donne de mauvais résultats, il est judicieux de recourir à des courbes paramétrées en coordonnées cartésiennes ou polaires. Cette section concerne les courbes paramétrées en coordonnées cartésiennes.

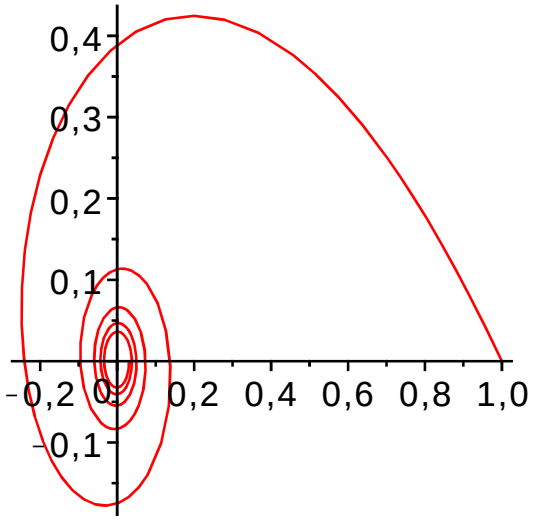
De deux choses l'une, soit on connaît le paramétrage associé à une courbe implicite, soit on n'en a pas la moindre idée.

Quand on connaît le paramétrage - par exemple à l'équation $x^2 + y^2 = 1$ est associé l'ensemble des points de coordonnées $\cos(t)$, $\sin(t)$ pour t variant de 0 à $2\,\pi$ - alors on utilise **plot** avec la syntaxe **plot([f(t),g(t),t=a..b],options]**, dans laquelle **t** est le paramètre compris entre les valeurs **a** et **b**, **f(t)** correspond à l'abscisse d'un point de la courbe paramétrée et **g(t)** son ordonnée. Toutes les options de **plot** sont tolérées, mais la plus utile ici est l'opération de calibrage placée en second et troisième argument et qui consiste à limiter le tracé de la courbe dans un rectangle dont la base est un intervalle des abscisses et la hauteur un intervalle des ordonnées : **x=b1..b2,y=h1..h2**.

```
> restart;
  plot([cos(t),sin(t),t=0..2*Pi],x=-1.25..1.25,y=-1.25..1.25)
  ;#cas du cercle
```



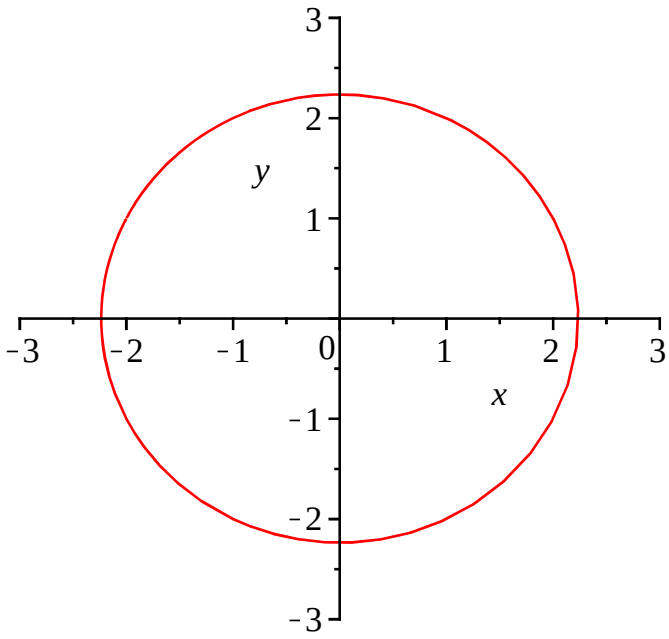

```
> restart;
  plot([cos(t)/(t+1),sin(t)/(t+1),t=0..10*Pi]);#cas d'une
  spirale
```



Et si on ignore la manière de paramétrer l'équation, on peut se faire aider par Maple qui propose dans le paquetage **algcurves** la commande **parametrization** qui retourne toujours une **paramétrisation polynômiale** dans une liste à deux éléments. Après transformation de cette liste en séquence par op, on trace la courbe avec plot en respectant la syntaxe décrite ci-dessus.

```
> with(algcurves):
  v:=parametrization(x^2+y^2-5,x,y,t);
  plot([op(v),t=-infinity..infinity],x=-3..3,y=-3..3);
```

$$v := \left[\frac{-2t^2+2+2t}{1+t^2}, \frac{-1+4t+t^2}{1+t^2} \right]$$



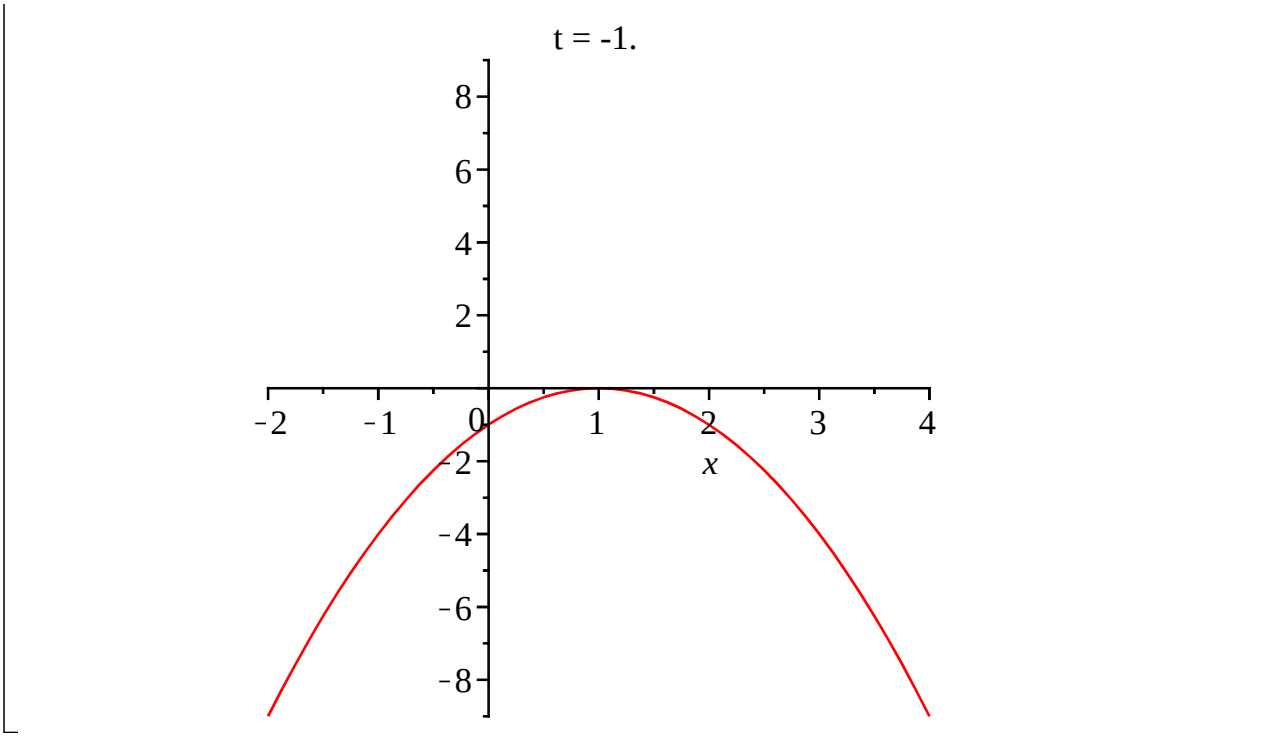
Animation

S'il y a bien une possibilité offerte par un logiciel de calcul formel qui n'existe pas avec du papier et un crayon et qu'on ne trouvera jamais dans un ouvrage, c'est celle de réaliser une animation graphique. Dans le paquetage **plots**, Maple propose la commande **animate** avec laquelle on réalise des petits "clips" par défilement rapide de vignettes donnant l'illusion du mouvement. On génère par défaut 25 graphiques légèrement différents en faisant balayer un paramètre sur un intervalle découpé par défaut en 24 sous-intervalles de même longueur dont il prend les valeurs des bornes.

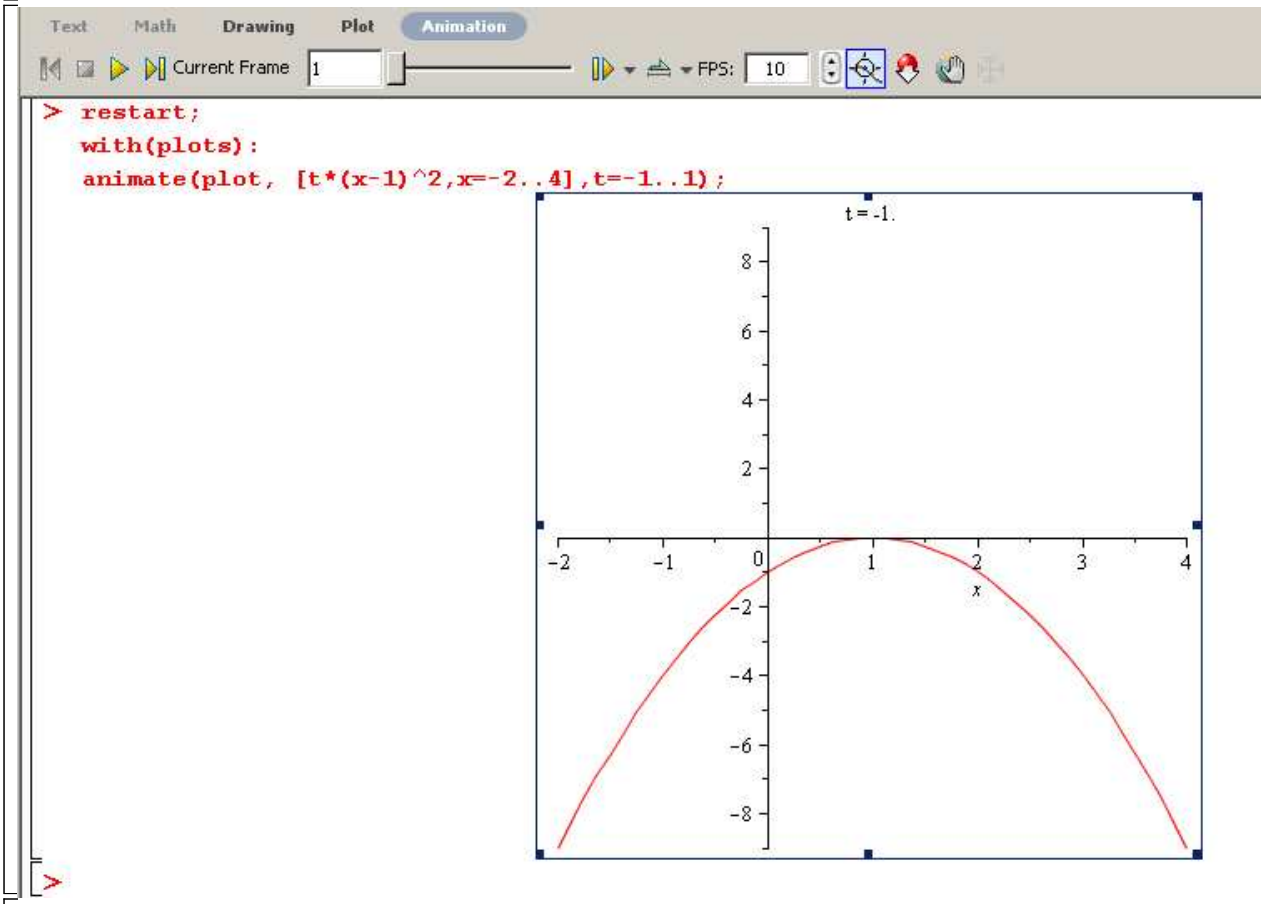
La syntaxe générale de la commande est **animate(plot,[Xp(x;t),x=a..b, options],t=c..d,options)** où **Xp(x;t)** est une fonction-expression de la variable réelle **x** et dépendant du paramètre **t**; **x** prend ses valeurs dans l'intervalle $[a, b]$ et **t** prend par défaut 25 valeurs dans l'intervalle $[c, d]$. Quelques options sont explicitées dans l'exemple suivant d'animation d'une fonction parabolique.

Ce premier tracé n'utilise aucune option. Plus exactement, les options conservent leur valeur par défaut.

```
> restart;
  with(plots):
  animate(plot, [t*(x-1)^2,x=-2..4],t=-1..1);
```



Un clic gauche dans le graphique fait basculer automatiquement dans le menu contextuel Animation



L'animation démarre avec le bouton  . On l'arrête à tout moment avec le carré placé immédiatement à gauche.

Plusieurs types de réglage sont proposés. En premier lieu, le défilement peut se faire suivant les valeurs croissantes (mode forward), décroissantes (mode backward) ou dans les deux sens au moyen du bouton suivant :

En second lieu, on passe du mode animation unique au mode animation en boucle par le bouton :

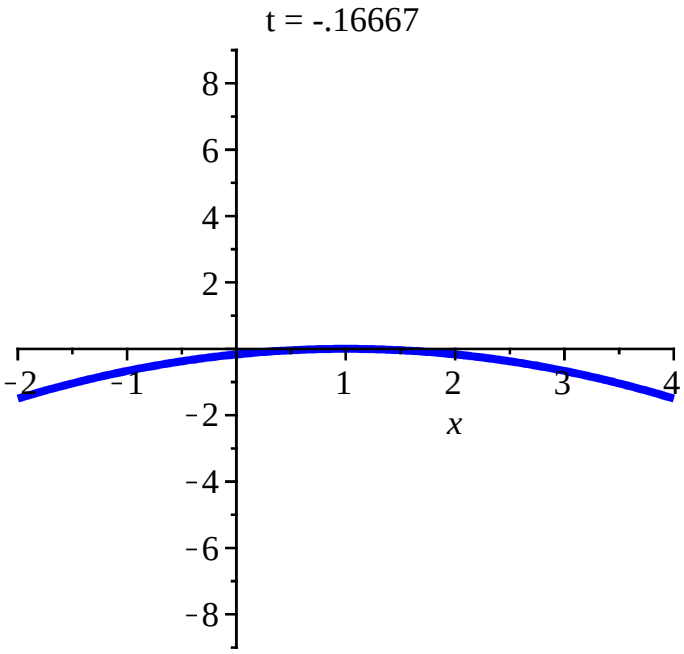
En troisième lieu, la vitesse du défilement est paramétrable avec :

Enfin, un défilement vignette par vignette, contrôlable par l'utilisateur, se fait avec le curseur :

Le numéro d'ordre de la vignette s'affiche à gauche de la barre. Une autre option est de se servir des boutons forward  et backward  .

La commande **animate** accepte toutes les options de **plot** dans la liste définissant le tracé à réaliser, de sorte qu'on peut améliorer le tracé de la courbe représentative. Par exemple, reprenons la parabole précédente afin d'accroître la précision du tracé et modifier l'épaisseur ainsi que la couleur de la courbe :

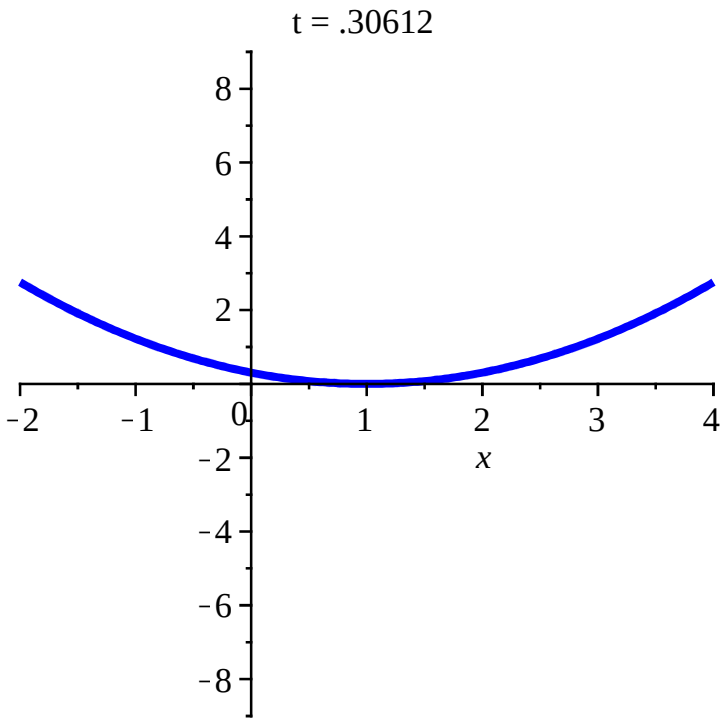
```
> animate(plot, [t*(x-1)^2,x=-2..4,numpoints=200,thickness=3,
  color=blue] ,t=-1..1);
```



Les options spécifiques de l'animation sont, quant à elles, inscrites en fin de commande, après la définition de l'intervalle de valeurs du paramètre *t*. La plus importante porte sur le nombre de vignettes (frames) composant le "clip".

```
> animate(plot, [t*(x-1)^2,x=-2..4,numpoints=200,thickness=3,
```

```
color=blue] ,t=-1..1,frames=50);
```



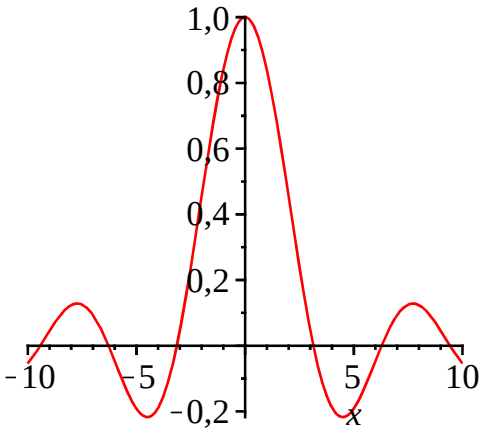
Graphisme interactif

Maple est un logiciel qui a connu plusieurs versions successives visant, entre autres, à améliorer le confort de l'utilisateur. Dans le domaine graphique, un gros effort a été fait dans le domaine de l'interactivité. Des boîtes de dialogue et des astuces réduisent sensiblement le temps passé à écrire des lignes de commande dans une syntaxe quelquefois contraignante. On présente dans cette section la commande **`smatplot`** puis l'outil "Interactive Plot Builder".

Graphe dynamique

Reprenons l'exemple introductif de la fonction $f(x) = \frac{\sin(x)}{x}$, mais en le traitant maintenant avec la commande à seul argument **`smartplot.`**

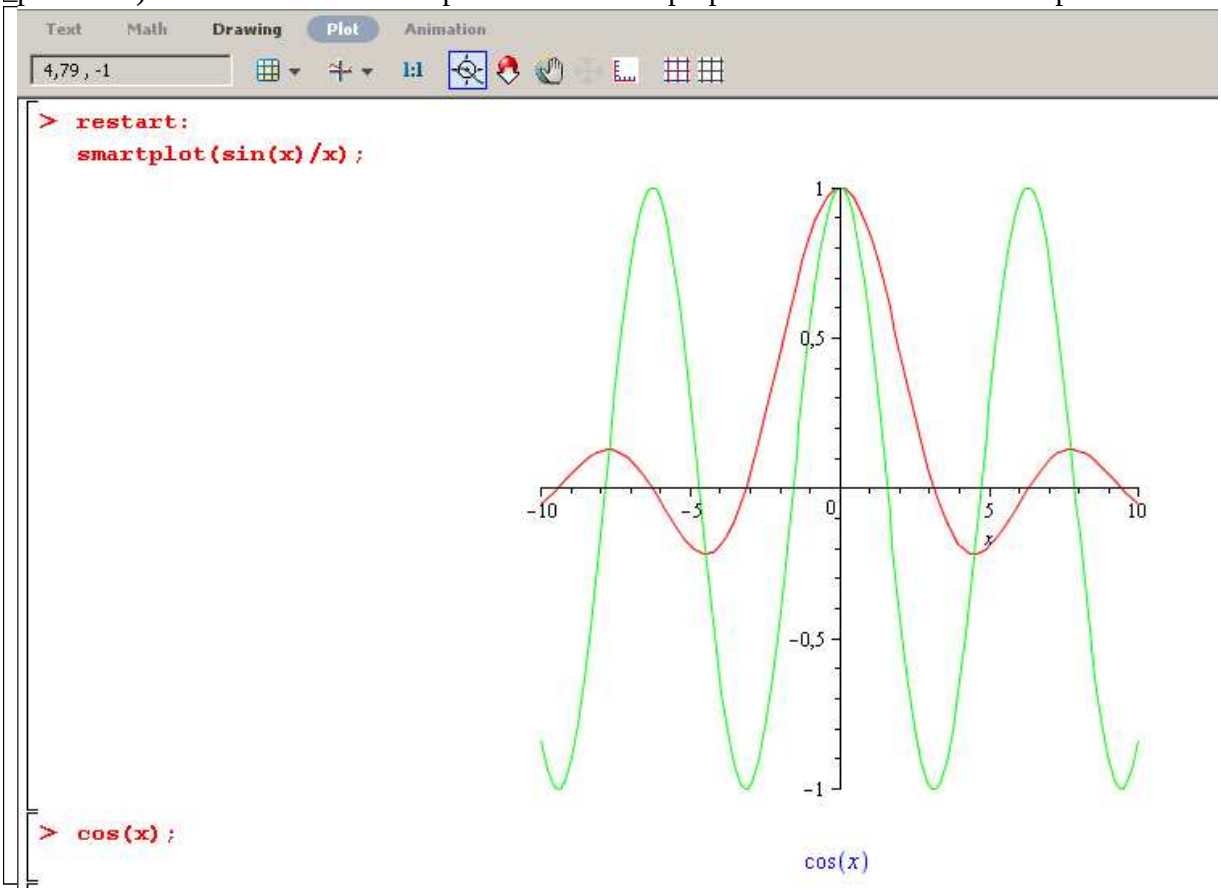
```
> restart:
  smartplot(sin(x)/x);
```



On constate que Maple a géré lui-même les axes. Un clic droit sur le graphique fait apparaître le menu contextuel décrit plus haut dans la section "Options graphiques élémentaires" qui

permet d'affiner le graphe suivant les besoins (intervalle des abscisses, intervalle des ordonnées, légende, etc..).

Plus fort. Supposons qu'on veuille superposer le graphe de $g(x) = \cos(x)$ sur celui de $f(x)$. Il est inutile de construire un second graphique puis d'invoquer le paquetage **plots**. Il suffit d'écrire la nouvelle fonction à tracer, de valider la requête, de sélectionner l'output bleu (en le surlignant en bleu) puis, en maintenant la souris enfoncée, de migrer l'output vers le graphique de $f(x)$ (technique du "drag and drop; le pointeur est indicié par un rectangle en pointillés). La nouvelle courbe représentative se superpose instantanément sur la première.

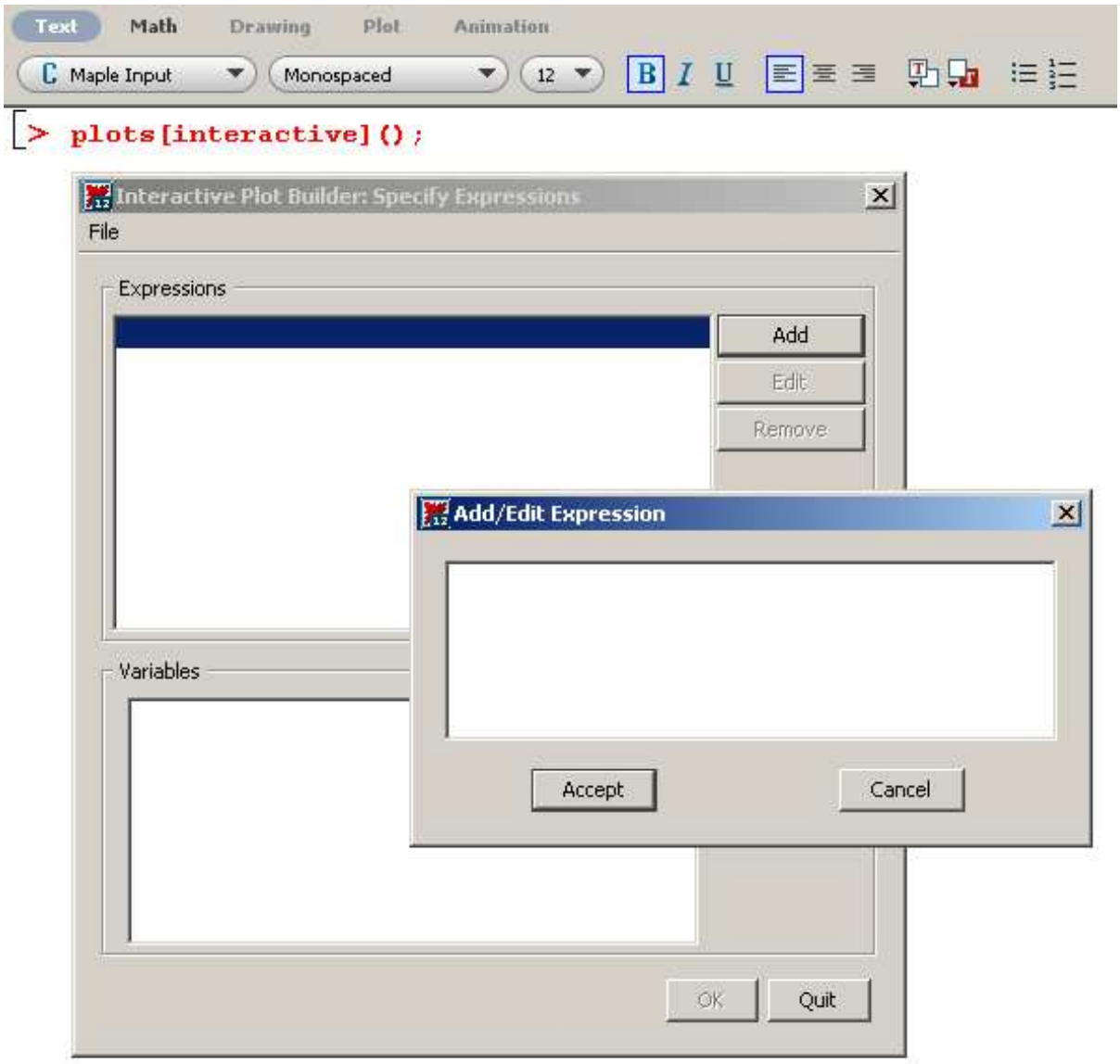


Outil graphique interactif

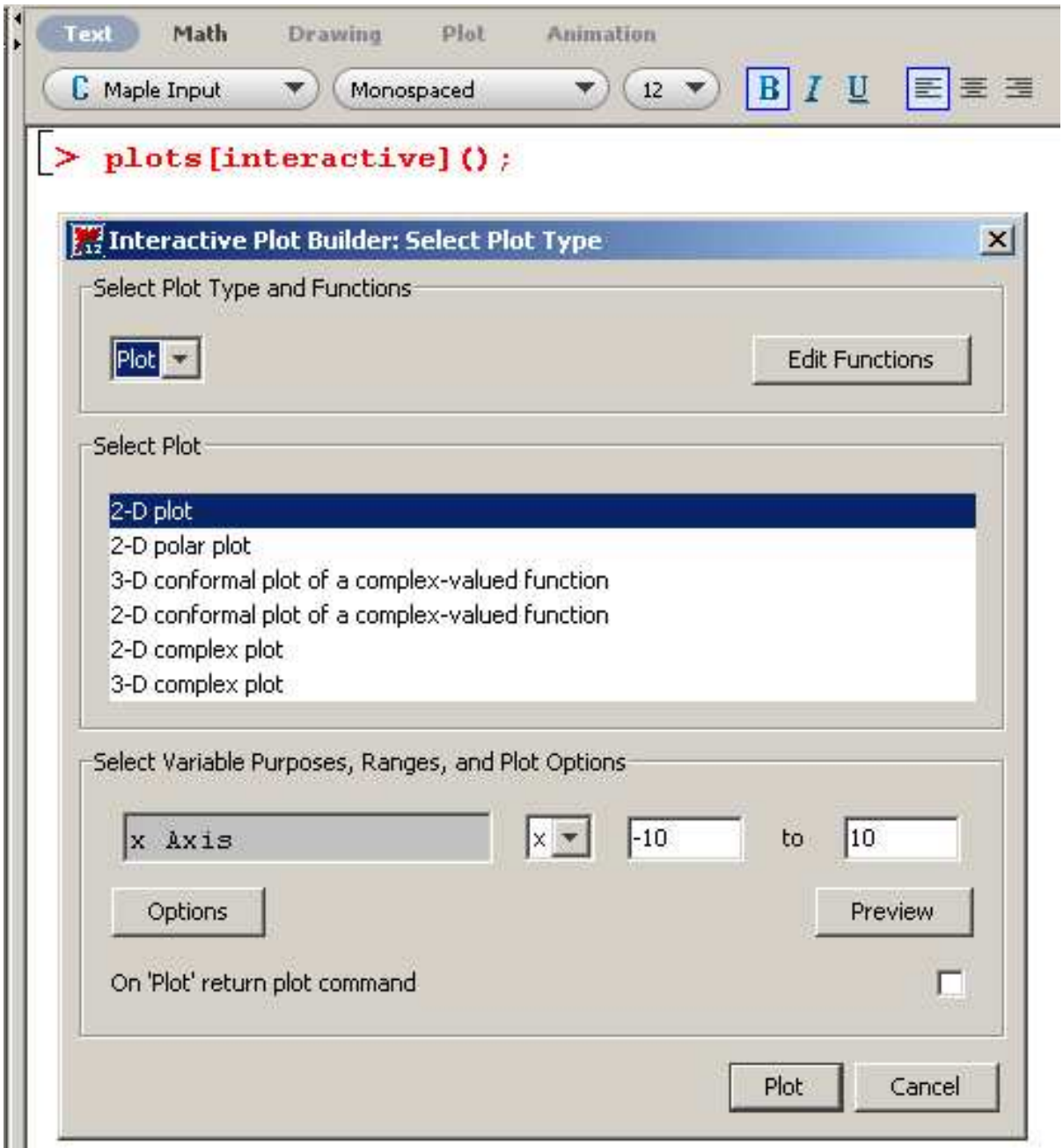
Les programmeurs de Maple ont mis au point un outil de construction graphique dit interactif au sens où il se plie à (presque) tous les caprices de l'utilisateur pour obtenir une courbe élaborée sans qu'il ait besoin de les programmer lui-même. Cet Interactive Plot Builder permet d'obtenir un résultat honorable en un temps record.

On accède à cet outil de trois manières :
1. Après avoir chargé le paquetage plots, il faut taper la commande à une variable **interactive(Xp)**, où **Xp** est une expression. La validation de l'input ouvre la boîte de dialogue "Interactive Plot Builder : Specify Expressions". On peut aussi valider l'instruction **plots[interactive]();**
2. Dans la barre de menu, valider la séquence Tools→Assistants→Plot Builder.
3. Utilisation du raccourci astucieux suivant : taper une expression; la valider; faire un clic droit sur l'output; choisir dans le menu Plots le sous-menu PlotBuilder. Maple ajoute immédiatement un nouvel input et affiche la boîte de dialogue "Interactive Plot Builder : Select Plot Type".

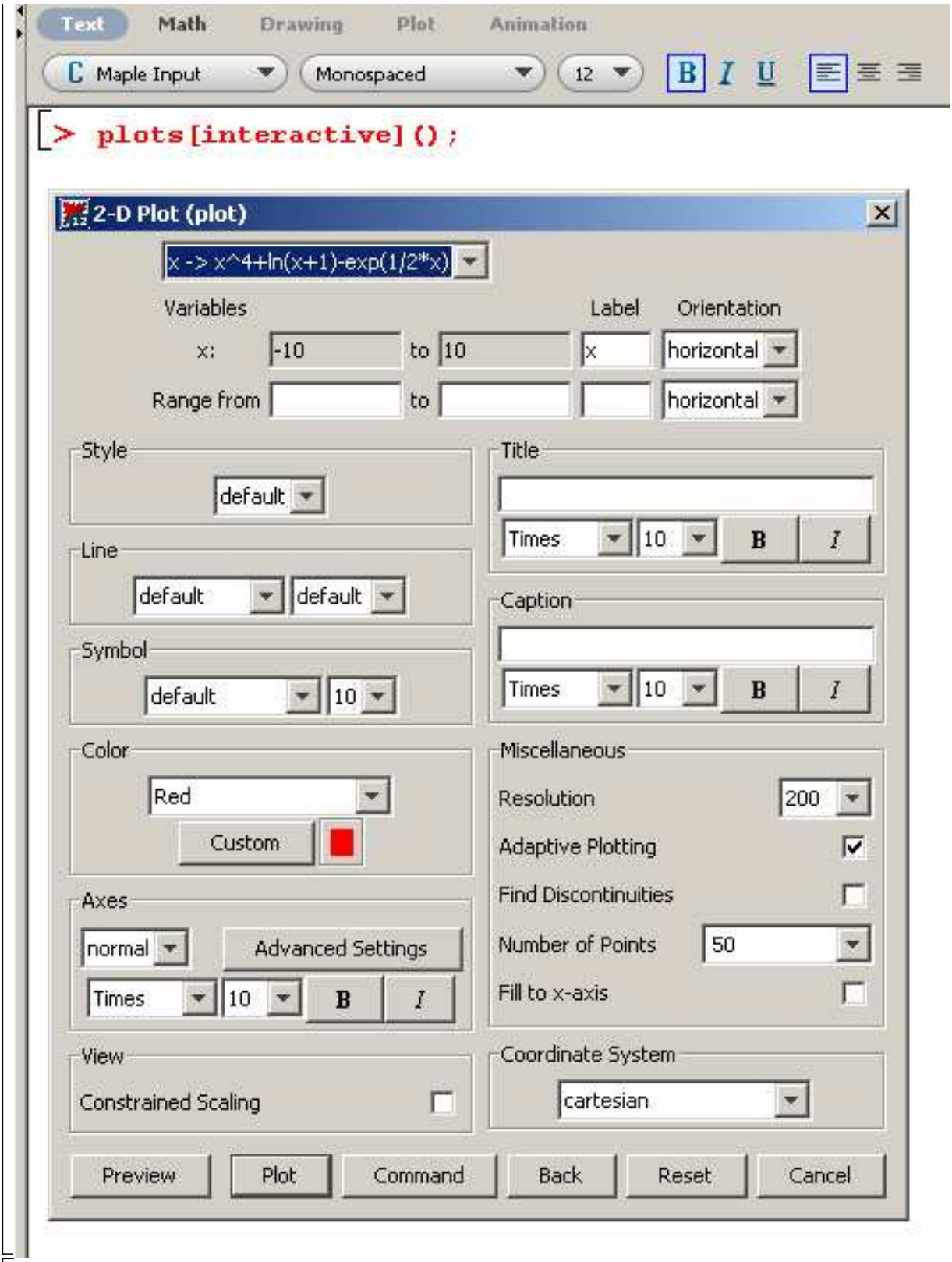
Dans les cas 1 et 2, l'outil graphique interactif doit connaître la ou les expressions à tracer. D'où la phase de remplissage de la fiche Specify Expressions.



Le premier cadre sert à entrer une ou plusieurs expressions. En cliquant sur Add, on accède à une fenêtre dans laquelle on frappe une expression comme on le ferait dans un Input Maple. On remplit autant de fiches qu'il y a de courbes à représenter dans le même graphe. Le second cadre définit la variable-argument de la ou des fonctions à étudier. Toutes les données sont validées par OK, qui ouvre une seconde fenêtre intitulée "Interactive Plot Builder : Select Plot Type".



La fiche propose d'abord de choisir un type de représentation graphique. Le thème de ce chapitre nous contraint à retenir 2-D Plot.
On paramètre l'intervalle des abscisses en donnant sa borne inférieure et sa borne supérieure.
Le bouton Preview affiche la courbe dans l'intervalle donné par l'utilisateur, qui peut donc corriger par tâtonnements ses indications.
Un clic sur le bouton Options ouvre une nouvelle fenêtre nommée 2-D Plot (plot) qui recèle une mine d'options d'amélioration du graphique.



On retrouve bien entendu la totalité des paramétrages exposés dans ce chapitre. Si besoin est, on peut visualiser les choix avec Preview et éventuellement les modifier. Plot renvoie le graphique en output inséré dans la feuille de travail. Command affiche dans la feuille de travail un output décrivant le programme de plot avec l'expression, l'intervalle des abscisses mais aussi toutes les options retenues par l'utilisateur. Back renvoie à la fiche Select Plot Type, afin de modifier la fonction ou l'intervalle des abscisses. Reset remet toutes les valeurs par défaut. Cancel annule le travail en cours.

Conclusion

Deux philosophies de la construction graphique coexistent dans Maple. La première repose sur la programmation et exige de connaître intimement - au prix d'un effort soutenu - tout l'éventail des ressources et des options de **plot** et du paquetage **plots**.La seconde permet de faire l'économie d'un apprentissage et s'en tient au but recherché : obtenir rapidement un bon graphique qui va s'insérer naturellement dans la feuille de travail pour illustrer une propriété économique. Pour l'économiste convaincu que l'art de la programmation doit rester en dehors de ses préoccupations professionnelles, la seconde solution est l'idéal. Mais le revers de ce choix est l'impossibilité de progresser vers des problèmes plus intéressants et souvent plus difficiles qui requièrent la mise au point de procédures, programmes d'automatisation de tâches qui ne peuvent par nature inclure des outils interactifs.

Exercices

Exercice M1

Joindre par des segments de doite les points du plan de coordonnées $(1; 4)$, $(-2; -3)$, $(4; -5)$ et $(-6; 5)$. Les points sont des petits cercles de couleur noire et les segments de couleur bleue.

Exercice M2

L'option **filled=true** de la commande plot permet de colorer la surface comprise entre la courbe représentative d'une fonction et l'axe des abscisses. A l'aide de la commande **display** du paquetage **plots**, construire un graphique colorant en rouge la surface comprise entre la courbe représentative de $y = \cos(x)$ et celle de $y = \sin(x)$ sur l'intervalle $[0; 4\,\pi]$.

Exercice M3

Soit la fonction f définie par $f(x) = \sqrt[3]{x^3 - 3\,x + 2}$.

1. Donner une représentation graphique pertinente.
2. Déterminer ses limites en $+\infty$ et $-\infty$.
3. Etudier la dérivabilité de f en $x = -2$ et $x = 1$.
4. Montrez que f admet un maximum local en $x = -1$.

Exercice E1

Soit la fonction de production Cobb-Douglas $Q = L^{\frac{2}{3}} K^{\frac{1}{3}}$. Tracer dans le même plan (quantité de travail, quantité de capital) = (L, K) les isoquants de production $Q = 10$, $Q = 20$, $Q = 30$.

Exercice E2

Cet exercice porte sur le modèle IS-LM simple. L'équation d'équilibre sur le marché des biens et services est donnée par $Y = 850 - 2500\,i$ et l'équation d'équilibre sur le marché de la monnaie par $Y = -500 + 5\,m + 1000\,i$, où Y est le PIB, i est le taux d'intérêt et m l'offre réelle de monnaie ($m = \frac{M}{P}$, M étant la masse monétaire et P l'indice général des prix.

1. On pose $m = 600$. Représentez graphiquement l'équilibre macroéconomique dans le plan iOY. Les droites seront annotées : $IS(1)$ pour l'équilibre sur le marché des biens et services;

$LM(1)$ pour le marché de la monnaie. Le point d'équilibre sera marqué par un petit cercle noir et noté $E(1)$. On reportera sur les axes les labels $Y(1)$ et $i(1)$ correspondant aux valeurs d'équilibre du PIB et du taux d'intérêt. Des lignes en pointillés joindront ces valeurs et le point d'équilibre $E(1)$. Le graphe aura un titre et une légende.

2. L'offre de monnaie passe à $m = 800$. Représentez graphiquement la situation sur le même modèle que la question 1 en notant $LM(2)$ la nouvelle droite d'équilibre sur le marché de la monnaie, $E(2)$ le nouvel équilibre, $Y(2)$ la nouvelle valeur d'équilibre du PIB et $i(2)$ le nouveau taux d'intérêt d'équilibre.

3. En superposant les deux précédents graphiques, montrez comment on est passé de $E(1)$ à $E(2)$ par un fléchage suggestif.

4. On admet que la masse monétaire a augmenté régulièrement de $m = 600$ à $m = 800$. Créez une animation montrant simultanément l'évolution de la droite LM et du point d'équilibre.