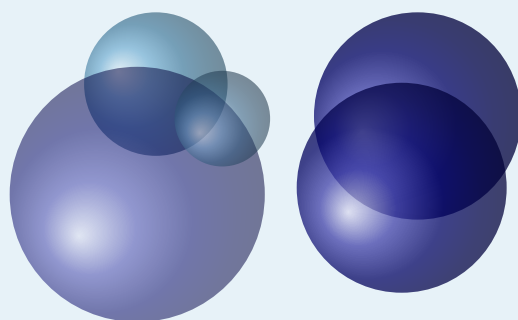


AlterMundus



Alain Matthes

11 avril 2020 Documentation V.2.3c

<http://altermundus.fr>

tkz-fct

AlterMundus

Alain Matthes

tkz-fct.sty est un package pour créer à l'aide de TikZ, des représentations graphiques de fonctions en 2D le plus simplement possible. Il est dépendant de TikZ et fera partie d'une série de modules ayant comme point commun, la création de dessins utiles dans l'enseignement des mathématiques.

☞ Je souhaite remercier **Till Tantau** pour avoir créé le merveilleux outil **TikZ**, ainsi que **Michel Bovani** pour **fourier**, dont l'association avec **utopia** est excellente.

☞ Je souhaite remercier aussi **David Arnold** qui a corrigé un grand nombre d'erreurs et qui a testé de nombreux exemples, **Wolfgang Büchel** qui a corrigé également des erreurs et a construit de superbes scripts pour obtenir les fichiers d'exemples, **John Kitzmiller** et ses exemples, et enfin **Gaétan Marris** pour ses remarques.

☞ Vous trouverez de nombreux exemples sur mes sites : altermundus.fr

Vous pouvez envoyer vos remarques, et les rapports sur des erreurs que vous aurez constatées à l'adresse suivante : [Alain Matthes](mailto:Alain.Matthes@univ-lille.fr).

This work may be distributed and/or modified under the conditions of the LaTeX Project Public License, either version 1.3 of this license or (at your option) any later version.

Table des matières

1	Fonctionnement	6
2	Utilisation de Gnuplot	7
2.1	Mécanisme d'interaction entre TikZ et Gnuplot	7
2.2	Installation de Gnuplot	9
2.3	Test de l'installation de tkz-base	10
2.4	Test de l'installation de tkz-fct	10
3	Les différentes macros	12
3.1	Tracé d'une fonction avec gnuplot <code>\tkzFct</code>	12
3.2	option : <code>samples</code>	12
3.3	options : <code>xstep</code> , <code>ystep</code>	13
3.4	Modification de <code>xstep</code> et <code>ystep</code>	13
3.5	<code>ystep</code> et les fonctions constantes	14
3.6	Les fonctions affines ou linéaires	14
3.7	Sous-grille	15
3.8	Utilisation des macros de <code>tkz-base</code>	15
4	Placer un point sur une courbe	17
4.1	Exemple avec <code>\tkzGetPoint</code>	17
4.2	Exemple avec <code>\tkzGetPoint</code> et <code>tkzPointResult</code>	18
4.3	Options <code>draw</code> et <code>ref</code>	18
4.4	Placer des points sans courbe	19
4.5	Placer des points sans se soucier des coordonnées	20
4.6	Placer des points avec deux fonctions	21
5	Labels	22
5.1	Ajouter un label	22
6	Macros pour tracer des tangentes	23
6.1	Représentation d'une tangente <code>\tkzDrawTangentLine</code>	23
6.2	Tangente avec <code>xstep</code> et <code>ystep</code> différents de 1	23
6.3	Les options <code>kl</code> , <code>kr</code> et l'option <code>draw</code>	24
6.4	Tangente et l'option <code>with</code>	25
6.5	Quelques tangentes	26
6.6	Demi-tangentes	27
6.7	Demi-tangentes Courbe de Lorentz	28
6.8	Série de tangentes	29
6.9	Série de tangentes sans courbe	29
6.9.1	Utilisation de <code>\tkzFctLast</code>	30
6.10	Calcul de l'antécédent	31
6.10.1	Valeur numérique de l'antécédent	31
6.10.2	utilisation de la valeur numérique	31
7	Macros pour définir des surfaces	32
7.1	Représentation d'une surface <code>\tkzDrawArea</code> ou <code>\tkzArea</code>	32
7.2	Naissance de la fonction logarithme népérien	32
7.3	Surface simple	33
7.4	Surface et hachures	35
7.5	Surface comprise entre deux courbes <code>\tkzDrawAreaafg</code>	36
7.6	Surface comprise entre deux courbes en couleur	36
7.7	Surface comprise entre deux courbes avec des hachures	37

7.8	Surface comprise entre deux courbes avec l'option between	37
7.9	Surface comprise entre deux courbes : courbes de Lorentz	38
7.10	Mélange de style	39
7.11	Courbes de niveaux	40
8	Sommes de Riemann	41
8.1	Somme de Riemann	41
8.2	Somme de Riemann Inf	42
8.3	Somme de Riemann Inf et Sup	43
8.4	Somme de Riemann Mid	44
9	Droites particulières	45
9.1	Tracer une ligne verticale	45
9.2	Ligne verticale	45
9.3	Lignes verticales	46
9.4	Ligne verticale et valeur calculée par fp	46
9.5	Une ligne horizontale	47
9.6	Asymptote horizontale	47
9.7	Lignes horizontales	48
9.8	Asymptote horizontale et verticale	49
10	Courbes avec équations paramétrées	50
10.1	Courbe paramétrée exemple 1	50
10.2	Courbe paramétrée exemple 2	51
10.3	Courbe paramétrée exemple 3	52
10.4	Courbe paramétrée exemple 4	53
10.5	Courbe paramétrée exemple 5	54
10.6	Courbe paramétrée exemple 6	55
10.7	Courbe paramétrée exemple 7	56
11	Courbes en coordonnées polaires	57
11.1	Équation polaire exemple 1	57
11.2	Équation polaire exemple 2	58
11.3	Équation polaire Heart	59
11.4	Équation polaire exemple 4	59
11.5	Équation polaire Cannabis ou Marijuana Curve	61
11.6	Scarabaeus Curve	62
12	Symboles	63
13	Quelques exemples	64
13.1	Variante intermédiaire : TikZ + tkz-fct	64
13.2	Courbes de Lorentz	65
13.3	Courbe exponentielle	66
13.4	Axe logarithmique	67
13.5	Un peu de tout	68
13.6	Interpolation	69
13.6.1	Le code	69
13.6.2	la figure	69
13.7	Courbes de Van der Waals	71
13.7.1	Tableau de variations	71
13.7.2	Première courbe avec $b=1$	72
13.7.3	Deuxième courbe $b=1/3$	73
13.7.4	Troisième courbe $b=32/27$	74

13.8 Valeurs critiques	75
13.8.1 Courbes de Van der Walls	75
13.8.2 Courbes de Van der Walls (suite)	76
14 Exemples avec les packages alterqcm et tkz-tab	77
14.0.1 Première représentation	78
14.0.2 Seconde représentation	79
15 Utilisation pgfmath et de fp.sty	81
15.1 pgfmath	81
15.2 fp.sty	81
16 Quelques remarques	83
16.1 Fonctions de gnuplot	84
17 Liste de toutes les macros	85
17.1 Liste de toutes les macros fournies par ce package	85
17.2 Liste de toutes des macros essentielles de \tkz-base	85
Index	86

1 Fonctionnement

TikZ apporte différentes possibilités pour obtenir les représentations graphiques des fonctions. J'ai privilégié l'utilisation de `gnuplot`, car je trouve `pgfmath` trop lent et les résultats trop imprécis.

Avec TikZ et `gnuplot`, on obtient la représentation d'une fonction à l'aide de

```
\draw[options] plot function {gnuplot expression};
```

Dans cette nouvelle version de `tkz-fct`, la macro `\tkzFct` reprend le code précédent avec les mêmes options que celles de TikZ. Parmi les options, les plus importantes sont `domain` et `samples`.

La macro `\tkzFct` remplace `\draw plot function` mais exécute deux tâches supplémentaires, en plus du tracé. Tout d'abord, l'expression de la fonction est sauvegardée avec la syntaxe de `gnuplot` et également sauvegardée avec la syntaxe de `fp` pour une utilisation ultérieure. Cela permet, sans avoir à redonner l'expression, de placer par exemple, des points sur la courbe (les images sont calculées à l'aide de `fp`), ou bien encore, de tracer des tangentes.

Ensuite, et c'est le plus important, `\tkzFct` tient compte des unités utilisées pour l'axe des abscisses et celui des ordonnées. Ces unités sont définies en utilisant la macro `\tkzInit` du package `tkz-base` avec les options `xstep` et `ystep`.

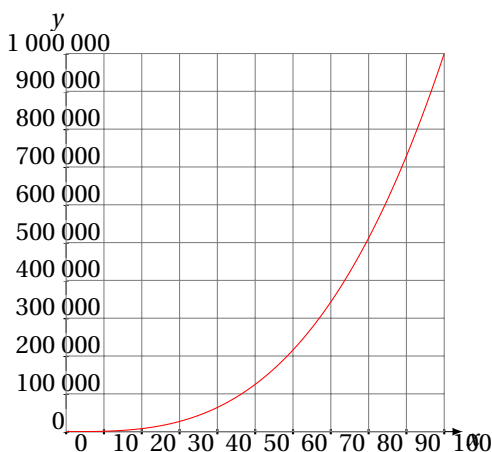
La macro `\tkzFct` intercepte les valeurs données à l'option `domain` et évidemment l'expression mathématique de la fonction; si `xstep` et `ystep` diffèrent de 1 alors il est tenu compte de ces valeurs pour le domaine, ainsi que pour les calculs d'images. Lorsque `xstep` diffère de 1 alors l'expression donnée, doit utiliser uniquement `\x` comme variable, c'est ainsi qu'il est possible d'ajuster les valeurs. Cela permet d'éviter des débordements dans les calculs.

Par exemple, soit à tracer le graphe de la fonction f définie par :

$$0 \leq x \leq 100 \text{ et } f(x) = x^3$$

Les valeurs de $f(x)$ sont comprises entre 0 et 1 000 000. En choisissant `xstep=10` et `ystep=100000`, les axes auront environ 10 cm de longueur (sans mise à l'échelle).

Les valeurs du domaine seront comprises entre 0 et 10, mais l'expression donnée à `gnuplot`, comportera des `\x` équivalents à $x \times 10$, enfin, la valeur finale sera divisée par `ystep=100000`. Les valeurs de $f(x)$ resteront ainsi comprises entre 0 et 10.

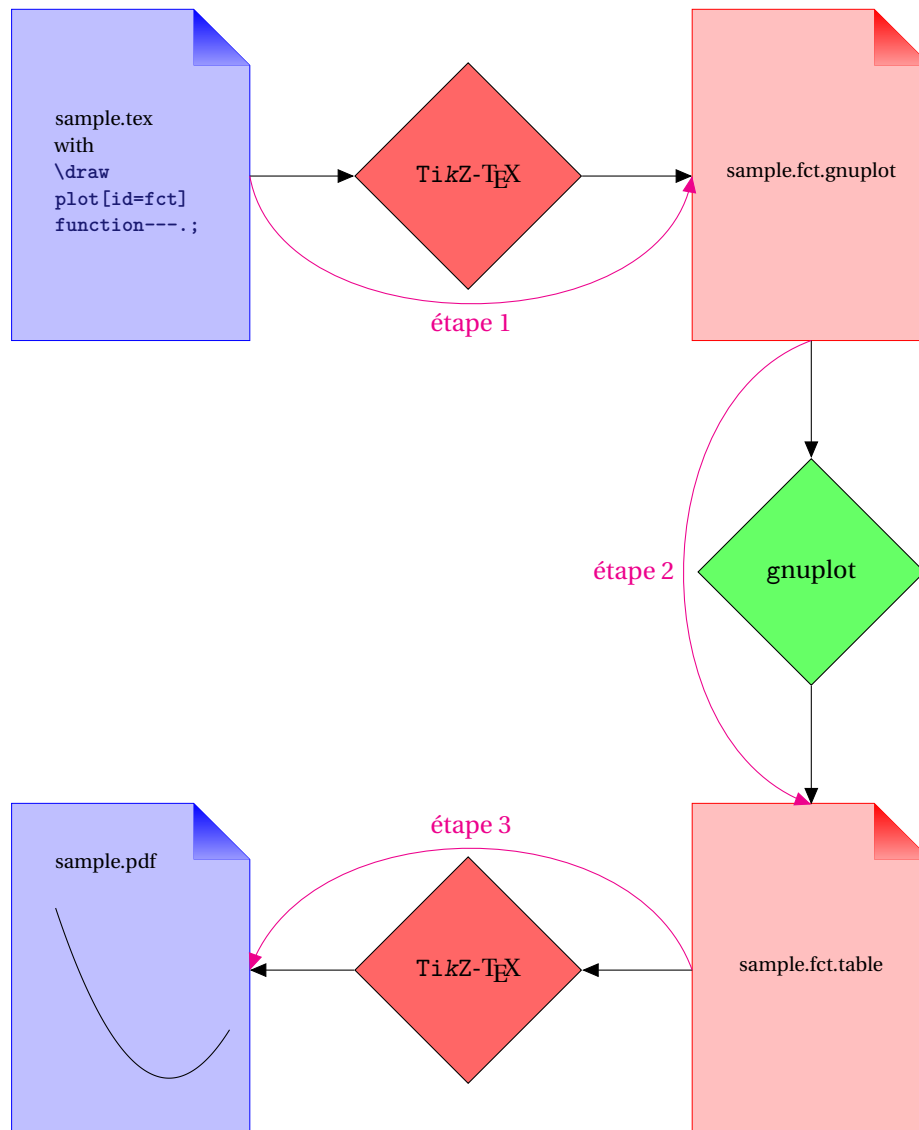


```
\begin{tikzpicture}[scale=.5]
  \tkzInit[xmax=100,xstep=10,
    ymax=1000000,
    ystep=100000]
  \tkzAxeX[right]
  \tkzAxeY[above]
  \tkzGrid
  \tkzFct[color=red,
    domain=0:100]{\x**3}
\end{tikzpicture}
```

2 Utilisation de Gnuplot

2.1 Mécanisme d'interaction entre TikZ et Gnuplot

T_EX est un système logiciel de composition de documents (text processing programm). Il permet bien sûr de calculer, mais avec des moyens limités. TikZ est ainsi limité par T_EX pour effectuer des calculs. Pour rappel ± 16383.99999 pt est l'intervalle dans lequel T_EX stocke ses valeurs. Sachant que 1 cm est égal à 28.45274 pt, on s'aperçoit que T_EX ne peut traiter que des dimensions inférieures à 5,75 mètres environ. Bien sûr, cela paraît suffisant, mais malheureusement, pendant un enchaînement de calculs, il est assez facile de dépasser ces limites.



Pour tracer des courbes en 2D en contournant ces problèmes, un moyen simple offert par TikZ, est d'utiliser `gnuplot`.

`tkz-fct.sty` s'appuie sur le programme `gnuplot` et le package `fp.sty`. Le premier est utilisé pour obtenir une liste de points, et le second pour évaluer ponctuellement des valeurs.

Vous devez donc installer `Gnuplot`, son installation dépend de votre système, puis il faudra que votre distribution trouve `Gnuplot`, et que `TEX` autorise `Gnuplot` à écrire un fichier.

— Étape 1

On part du fichier `sample.tex` suivant :

```
\documentclass{article}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}
\draw plot[id=f1,samples=200,domain=-2:2] function{x*x};
\end{tikzpicture}
\end{document}
```

La compilation de ce fichier créé avec TikZ, produit un fichier nommé `sample.f1.gnuplot`. Le nom du fichier est obtenu à partir de `\jobname` et de l'option `id`. Ainsi un même fichier peut créer plusieurs fichiers distincts. C'est un fichier texte ordinaire, affecté de l'extension `gnuplot`. Il contient un préambule indiquant à `gnuplot` qu'il doit créer une table contenant les coordonnées d'un certain nombre de points obtenu par la fonction définie par $x \rightarrow x^2$. Ce nombre de points est défini par l'option `samples`. Cette étape ne présente aucune difficulté particulière. Le fichier obtenu peut être traité manuellement avec `gnuplot`. Le résultat est le fichier suivant :

```
set table; set output "sample.f1.table"; set format "%.5f"
set samples 200; plot [x=-2:2] x*x
```

Une table sera créée et enregistrée dans un fichier texte nommé "sample.f1.table". Les nombres seront formatés pour ne contenir que 5 décimales. La table contiendra 201 couples de coordonnées.

— Étape 2

Elle est la plus délicate car le fichier `sample.f1.gnuplot` doit être ouvert par `gnuplot`. Cela implique d'une part, que `TEX` autorise l'ouverture¹ du fichier `sample.f1.gnuplot` par `gnuplot` et d'autre part, que `TEX` puisse trouver `gnuplot`².

Si `gnuplot` trouve `sample.f1.gnuplot` alors il produit un fichier texte `sample.f1.table`, évidemment s'il ne trouve d'erreur de syntaxe dans l'expression de la fonction.



Malheureusement, une incompréhension peut surgir entre TikZ et `gnuplot`. TikZ jusqu'à sa version 2.00 officielle, est conçu pour fonctionner avec `gnuplot` version 4.0 et malheureusement, `gnuplot` a changé de syntaxe. la documentation de `gnuplot` indique :

```
Features, changes and fixes in gnuplot version 4.2 (and >)
'set table "outfile"; ---.; unset table' replaces 'set term table'
```

La version 2.1 de TikZ a adopté `set table` et il n'y a plus d'incompatibilité entre TikZ et les versions récentes de `gnuplot` (v>4.2).

1. c'est ici que l'on parle des options `--shell-escape` et `--enable-write18`

2. c'est ici que l'on parle de `PATH`

— Étape 3

Le fichier `sample.f1.table` obtenu à l'étape précédente est utilisé par TikZ pour tracer la courbe.

```
# Curve 0 of 1, 201 points
# Curve title: "x*x"
# x y type
-2.00000 4.00000 i
-1.98000 3.92040 i
-1.96000 3.84160 i
---
1.98000 3.92040 i
2.00000 4.00000 i
```

1. Il faut remarquer qu'au cours d'une seconde compilation, si le fichier `sample.f1.gnuplot` ne change pas, alors `gnuplot` n'est pas lancé et le fichier présent `sample.f1.table` est utilisé.
2. On peut aussi remarquer que si vous êtes paranoïaque et que vous n'autorisez pas le lancement de `gnuplot`, alors une première compilation permettra de créer le fichier `sample.f1.table`, ensuite manuellement, vous pourrez lancer `gnuplot` et obtenir le fichier `sample.f1.table`.
3. Il est aussi possible de créer manuellement ou encore avec un quelconque programme, un fichier `data.table` que TikZ pourra lire avec

```
\draw plot[smooth] file {data.table};
```

2.2 Installation de Gnuplot

Gnuplot est proposé avec la plupart des distributions Linux, et existe pour OS X ainsi que pour Windows.

1. **UbuntuLinux** Ubuntu ou un autre système Linux : on l'installe en suivant la procédure classique d'installation d'un nouveau paquetage.
2. **WindowsWindows XP** Les utilisateurs de Windows doivent se méfier, après avoir téléchargé la bonne version et installé `gnuplot` alors il faudra renommer `wgnuplot` en `gnuplot`. Ensuite il faudra modifier le `path`. Si le chemin du programme est `C:\gnuplot` alors il faudra ajouter `C:\gnuplot\bin\` aux variables environnement (Aller à "Poste de Travail" puis faire "propriétés", dans l'onglet "Avancé", cliquer sur "Variables d'environnement"). Ensuite pour compiler sous latex, il faudra ajouter au script de compilation l'option `--enable-write18`.
3. **OS XOS X** C'est le système en version Snow Leopard qui pose le plus de problème, car il faut compiler les sources. Si vous n'utilisez `gnuplot` qu'en collaboration avec TikZ alors il vous suffit de compiler les sources ainsi :
 - a) Télécharger les sources de `gnuplot`, déposer les sources sur le bureau.
 - b) Ouvrir un terminal puis taper `cd` et glisser le dossier des sources après `cd` (en laissant un espace) Cela doit donner

```
$ cd /Users/ego/Desktop/gnuplot-4.4.2
```

- c) ensuite taper la ligne suivante et valider

```
$ ./configure --with-readline=builtin
```

- d) puis

```
$ make
```

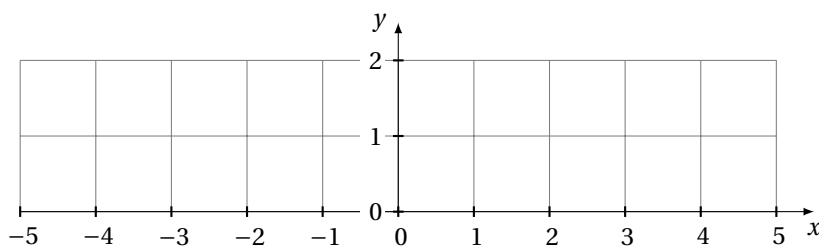
e) et enfin

```
$ sudo make install
```

2.3 Test de l'installation de tkz-base

Enregistrer le code suivant dans un fichier avec le nom test.tex, puis compiler avec pdflatex ou bien la chaîne dvi→ps→pdf. Vous devez obtenir cela :

```
\documentclass{scrartcl}
\usepackage[usenames,dvipsnames]{xcolor}
\usepackage{tkz-fct}
\begin{document}
  \begin{tikzpicture}
    \tkzInit[xmin=-5,xmax=5,ymax=2]
    \tkzGrid
    \tkzAxeXY
  \end{tikzpicture}
\end{document}
```

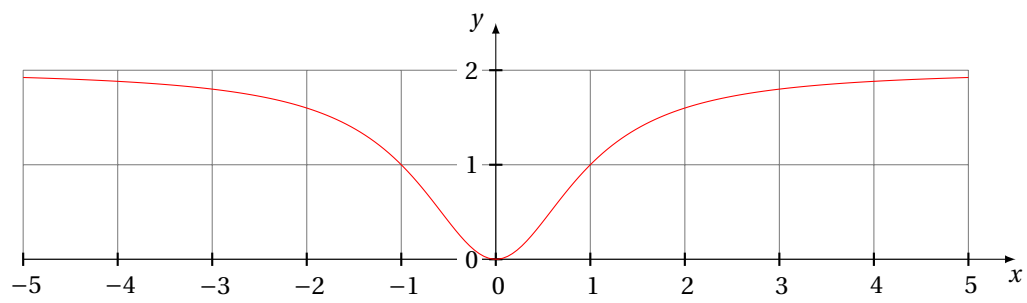


```
\begin{tikzpicture}
  \tkzInit[xmin=-5,xmax=5,ymax=2]
  \tkzGrid
  \tkzAxeXY
\end{tikzpicture}
```

2.4 Test de l'installation de tkz-fct

Il suffit d'ajouter une ligne pour tracer la représentation graphique d'une fonction.

```
\documentclass{scrartcl}
\usepackage[usenames,dvipsnames]{xcolor}
\usepackage{tkz-fct}
\begin{document}
  \begin{tikzpicture}[scale=1.25]
    \tkzInit[xmin=-5,xmax=5,ymax=2]
    \tkzGrid
    \tkzAxeXY
    \tkzFct[color=red]{2*x**2/(x**2+1)}
  \end{tikzpicture}
\end{document}
```



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-5,xmax=5,ymax=2]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[color=red]{2*x**2/(x**2+1)}
\end{tikzpicture}
```

3 Les différentes macros

Gnuplot détermine les points nécessaires pour tracer la courbe. Le nombre de points est fixé par l'option **samples**; dans les premiers exemples la valeur du nombre de points est celle donnée par défaut. Ensuite **Tikz** va utiliser cette table pour tracer la courbe. C'est donc **Tikz** qui trace la courbe.

3.1 Tracé d'une fonction avec `gnuplot \tkzFct`

Cette première macro est la plus importante car elle permet de tracer la représentation graphique d'une fonction continue.

```
\tkzFct[⟨local options⟩]{⟨gnuplot expression⟩}
```

La fonction est donnée en utilisant la syntaxe de **gnuplot**. x est la variable sauf si **xstep** est différent de 1, dans ce cas la variable est $\backslash x$.

options	exemple	explication
gnuplot expression	$x**3$	** représente la puissance \wedge

L'expression est de la forme $2*x+1; 3*log(x); x*exp(x); x*x*x+x*x+x$.

Les options sont celles de **TikZ**.

options	défaut	définition
domain	xmin:xmax	domaine de la fonction
samples	200	nombre de points utilisés
id	tkzfct	permet d'identifier les noms des fichiers auxiliaires
color	black	couleur de la ligne
line width	1pt	épaisseur de la ligne
style	solid	style de la ligne



Lorsque **xstep** est différent de 1, il est nécessaire de remplacer x par $\backslash x$.



Il faut bien évidemment avoir initialisé l'environnement à l'aide `\tkzInit` avant d'appeler `\tkzFct`.



Attention à ne pas mettre d'espace entre les arguments.

3.2 option : **samples**

Il faut remarquer que pour tracer une droite seulement deux points sont nécessaires, ainsi le code :

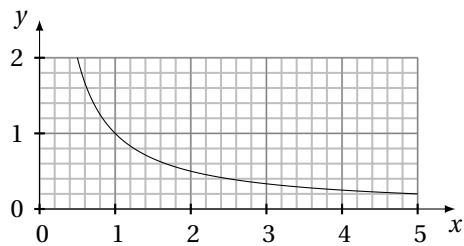
```
\tkzFct[{-(>,color=red,samples=2,domain =-1:2){(8-1.5*\x)/2}
```

donne un fichier xxx.table qui contient :

```
# Curve 0 of 1, 2 points
# Curve title: "(8-1.5*x)/2"
# x y type
-1.00000 4.75000 i
2.00000 2.50000 i
```

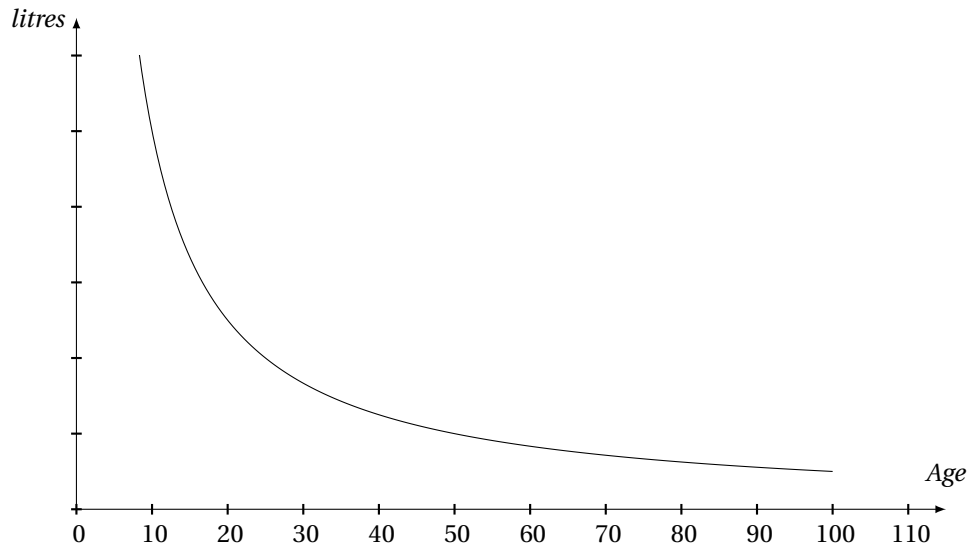
Ce qui est simplement suffisant. Plus simple est dans ce cas, de tracer un segment.

On demande 400 valeurs pour la table qui va permettre le tracé. Par défaut, la valeur choisie est 200.



```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=5,ymax=2]
  \tkzGrid[sub]
  \tkzAxeXY
  \tkzFct[samples=400,domain=.5:5]{1/x}
\end{tikzpicture}
```

3.3 options : xstep, ystep

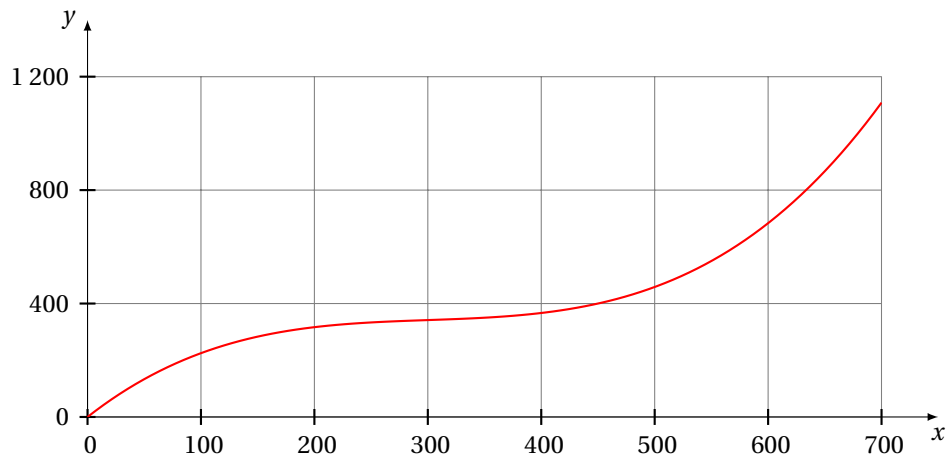


```
\begin{tikzpicture}
\tkzInit[xmax= 110,xstep=10,
        ymax=6,ystep=1]
\tkzDrawX[label={\textit{Age}},below= -18pt]
\tkzLabelX
\tkzDrawY[label={\textit{litres}}]
\tkzFct[domain = 0.1:100 ]{50/\x}
\end{tikzpicture}
```

3.4 Modification de xstep et ystep

Cette fois le domaine s'étend de 0 à 800, les valeurs prises par la fonction de 0 à 2 000. `xstep=100` donc il faut utiliser `\x` à la place de `x`. Une petite astuce au niveau de gnuplot, 1. et 113. permettent d'obtenir une division dans les décimaux sinon la division se fait dans les entiers.

Ensuite, j'utilise les macros pour placer des points



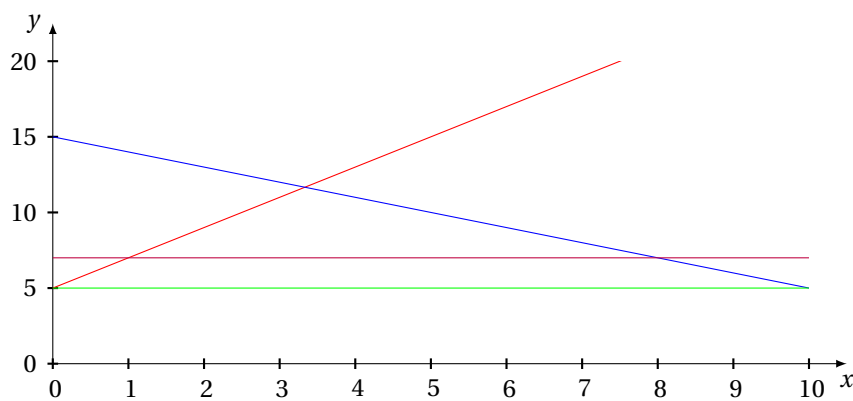
```
\begin{tikzpicture}[scale=1.5]
\tkzInit[xmax=700,xstep=100,ymax=1200,ystep=400]
\tkzGrid(0,0)(700,1200) \tkzAxeXY
\tkzFct[color=red,samples=100,line width=0.8pt,domain =0:700]%
    {(1./90000)*\x*\x*\x-(1./100)*\x*\x+(113./36)*\x}
\end{tikzpicture}
```

3.5 ystep et les fonctions constantes

Attention, ici `ystep=6` or `gnuplot` donne $80 \div 6 = 13$. il faut donc écrire 80.

3.6 Les fonctions affines ou linéaires

Pour obtenir des droites, on peut utiliser `gnuplot` même si l'outil est un peu lourd dans ce cas. Pour alléger les calculs, il est possible de ne demander que deux points!

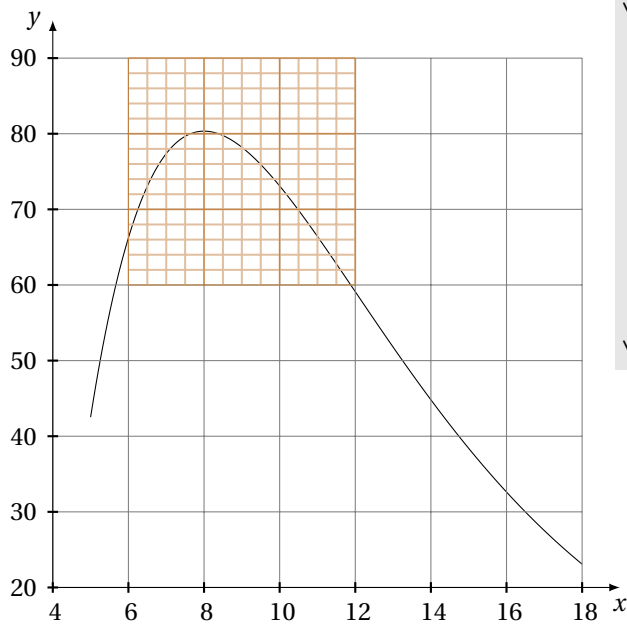


```
\begin{tikzpicture}[]
\tkzInit[ymax=20,ystep=5]
\tkzAxeXY
\tkzFct[color=red,domain=0:10,samples=2]{2*x+5}
\tkzFct[color=blue,domain=0:10,samples=2]{-x+15}
\tkzFct[color=green,domain=0:10,samples=2]{7} % 7/5=1
\tkzFct[color=purple,domain=0:10,samples=2]{7.}%7.0/5 =1.2
\end{tikzpicture}
```

3.7 Sous-grille

$$y = (x - 4)e^{-0.25x+5}$$

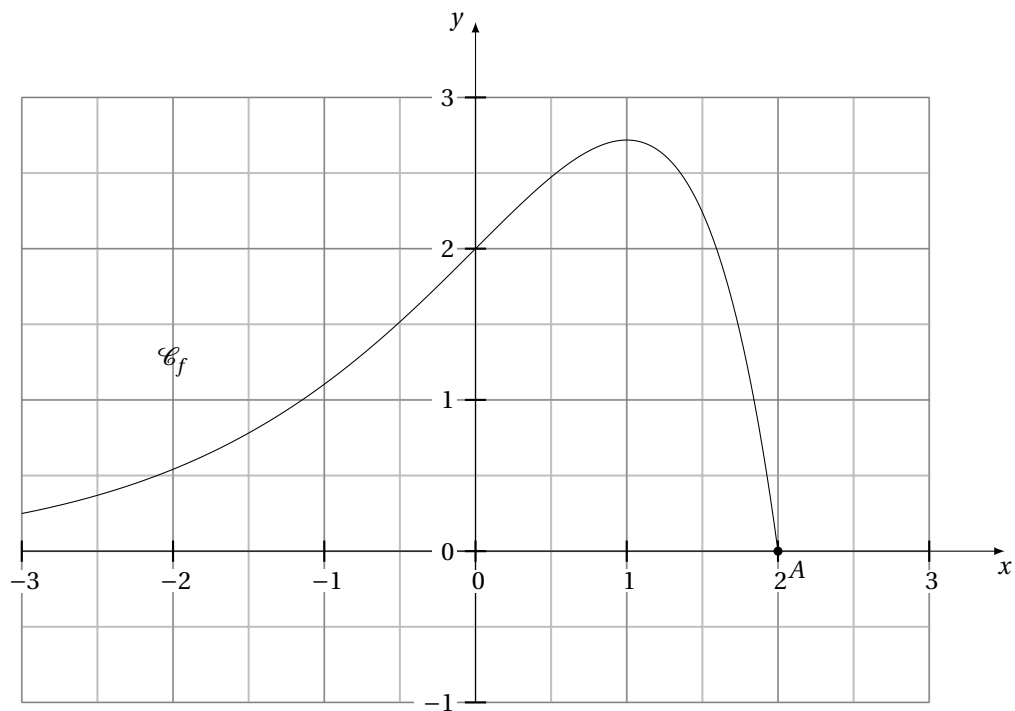
Il est possible de dessiner une autre grille.



```
\begin{tikzpicture}
\tkzInit[xmin=4,xmax=18,xstep=2,
        ymin=20,ymax=90,ystep=10]
\tkzFct[domain = 5:18]%
        {(\x-4)*exp(-0.25*\x+5)}
\tkzGrid(4,20)(18,90)
\tkzAxeXY
\tkzGrid[sub,
        subxstep=0.5,
        substep=2,
        color=brown](6,60)(12,90)
\end{tikzpicture}
```

3.8 Utilisation des macros de tkz-base

Toutes les macros de **tkz-base** sont bien sûr utilisables, en voici quelques exemples.



```
\begin{tikzpicture}[scale=2]
\tkzInit[xmin=-3,xmax=3, ymin=-1,ymax=3]
\tkzGrid[sub,subxstep=.5,subystep=.5]
\tkzAxeXY
\tkzFct[domain = -3:2]{(2-x)*exp(x)}
\tkzText(-2,1.25){$\mathcal{C}_f$}
\tkzDefPoint(2,0){A} \tkzDrawPoint(A) \tkzLabelPoints(A)
\end{tikzpicture}
```


4 Placer un point sur une courbe

`\tkzDefPointByFct(<decimalnumber>)`

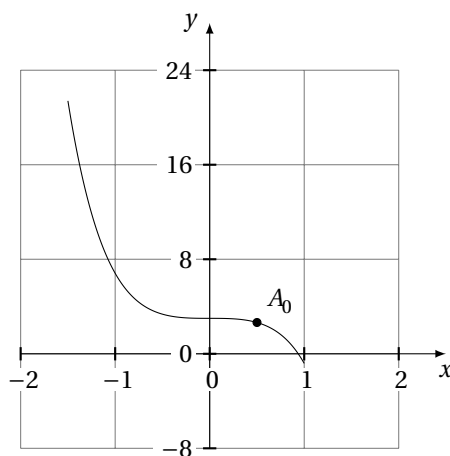
Cette macro permet de calculer l'image par la fonction définie précédemment, d'un nombre décimal.

argument	exemple	explication
decimal number	<code>\tkzDefPointByFct(0)</code>	définit un point d'abscisse 0
option	default	explication
draw	false	permet de tracer le point avec le style courant
with	a	permet de choisir la fonction
ref	empty	permet de donner une référence au point

C'est donc la dernière fonction définie qui est utilisée. Si une autre fonction, est utilisée alors il faut utiliser l'ancienne macro `\tkzFctPt`. Le point est défini sous un nom générique `tkzPointResult` mais non tracé. Afin de le tracer il faut utiliser la macro `\tkzDrawPoint`.

4.1 Exemple avec `\tkzGetPoint`

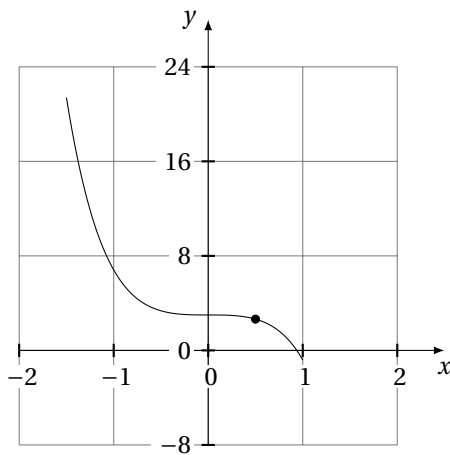
Cela permet de référencer le point créé par `\tkzDefPointByFct`.



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-2,xmax=2,xstep=1,
    ymin=-8,ymax=24,ystep=8]
  \tkzGrid \tkzAxeXY
  \tkzFct[domain=-1.5:1]{3.0-1.3125*x**5-2.5*x**3}
  \tkzDefPointByFct(.5) \tkzGetPoint{A}\tkzDrawPoint(A)
  \tkzLabelPoint[above right](A){$A_0$}
\end{tikzpicture}
```

4.2 Exemple avec `\tkzGetPoint` et `tkzPointResult`

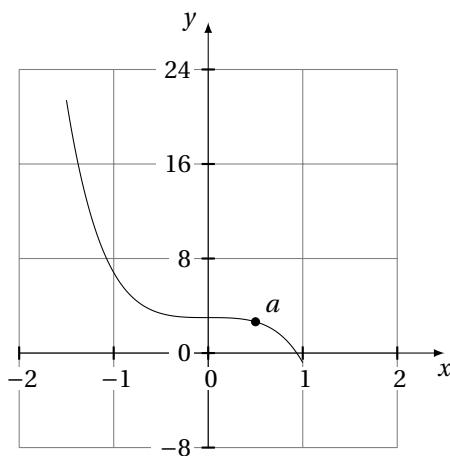
Il est possible de ne pas référencer le point et d'utiliser la référence générique.



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-2,xmax=2,xstep=1,
    ymin=-8,ymax=24,ystep=8]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain =-1.5:1]{3.0-1.3125*x**5-2.5*x**3}
  \tkzDefPointByFct(.5)
  \tkzDrawPoint(tkzPointResult)
  % ou bien \tkzDefPointByFct[draw](.5)
\end{tikzpicture}
```

4.3 Options `draw` et `ref`

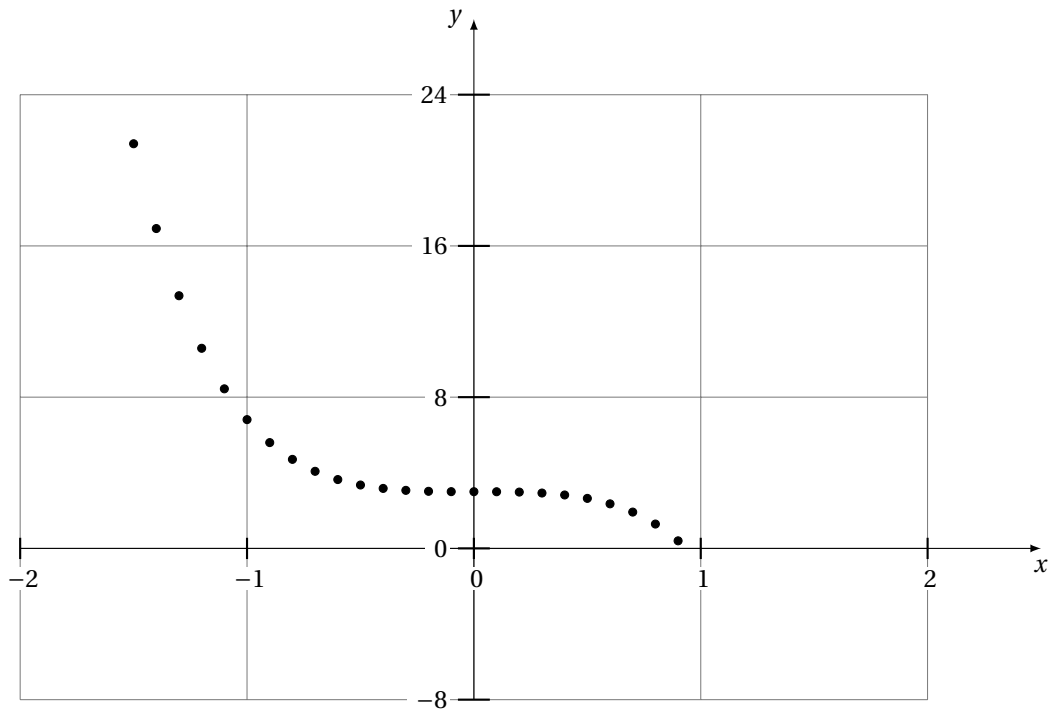
Cela permet de tracer un point directement avec les options usuelles donc sans possibilités de personnaliser et d'attribuer une référence à ce point.



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-2,xmax=2,xstep=1,
    ymin=-8,ymax=24,ystep=8]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain =-1.5:1]{3.0-1.3125*x**5-2.5*x**3}
  \tkzDefPointByFct[draw,ref=A](.5)
  \tkzLabelPoint[above right](A){$a$}
\end{tikzpicture}
```

4.4 Placer des points sans courbe

Attention, ceci est délicat. Il suffit de définir la macro `\tkzFctLast` qui est la dernière expression traduite avec la syntaxe de `fp.sty`. Les points sont donc déterminés avec `fp.sty`.

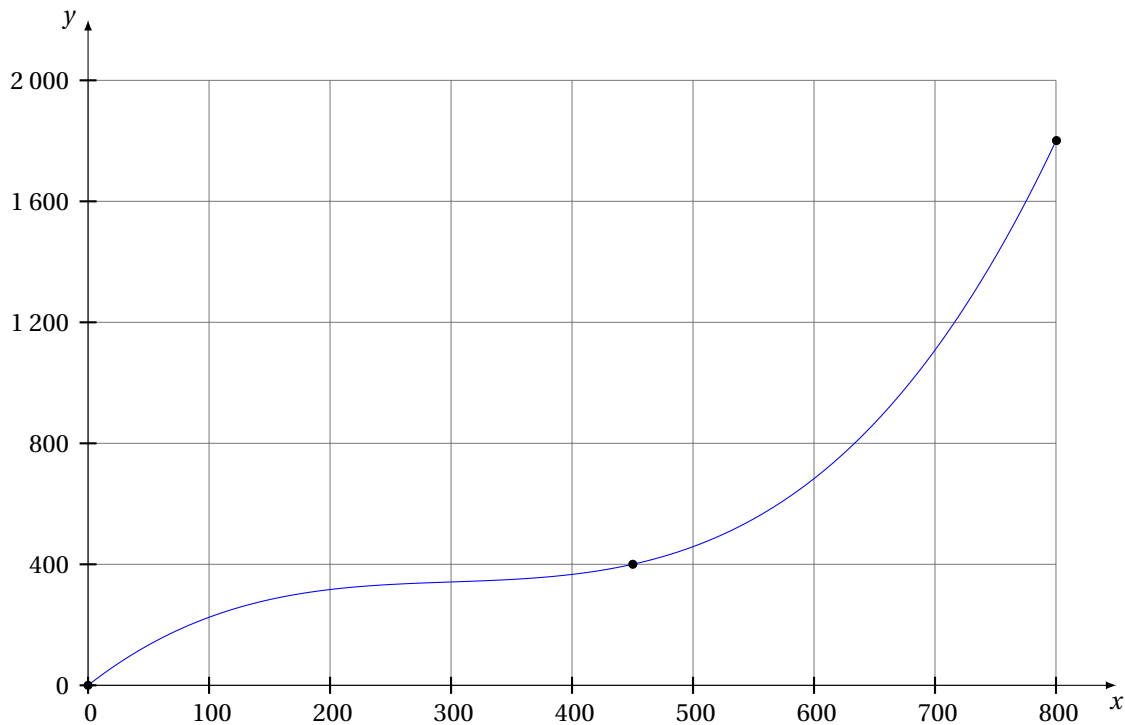


```
\begin{tikzpicture}[xscale=3,yscale=2]
  \tkzInit[xmin=-2,xmax=2,xstep=1,
    ymin=-8,ymax=24,ystep=8]
  \tkzGrid
  \tkzAxeXY
  \global\edef\tkzFctLast{3.0-1.3125*x^5-2.5*x^3}
  \foreach \va in {-1.5,-1.4,...,1}{%
    \tkzDefPointByFct[draw](\va)}
\end{tikzpicture}
```

4.5 Placer des points sans se soucier des coordonnées

Cette fois le domaine s'étend de 0 à 800, les valeurs prises par la fonction de 0 à 2 000. `xstep=100` donc il faut utiliser `\x` à la place de `x`. Une petite astuce au niveau de `gnuplot`, 1. et 113. permettent d'obtenir une division dans les décimaux sinon la division se fait dans les entiers.

Ensuite, j'utilise les macros pour placer des points

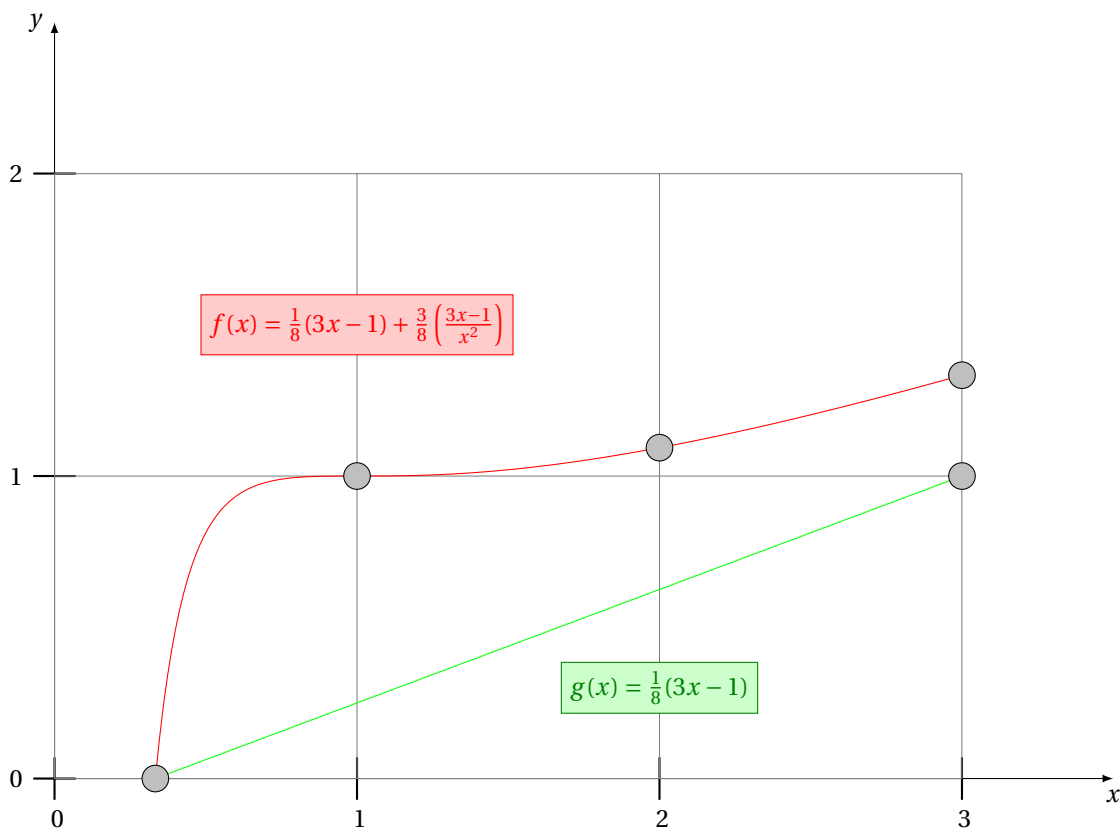


```
\begin{tikzpicture}[scale=1.6]
  \tkzInit[xmin = 0, xmax = 800,
    ymin = 0, ymax = 2000,
    xstep = 100, ystep = 400]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[color = blue,
    domain = 0:800]%
    {(1./90000)*\x*\x*\x-(1./100)*\x*\x+(113./36)*\x}
  \foreach \va in {0,450,800}{%
    \tkzDefPointByFct[draw](\va)}
\end{tikzpicture}
```

4.6 Placer des points avec deux fonctions

Revoir `\tkzSetUpPoint` et `\tkzText` du module `tkz-base.sty`

```
\begin{tikzpicture}[scale=4]
  \tkzInit[xmax=3,ymax=2]
  \tkzAxeX
  \tkzAxeY
  \tkzGrid(0,0)(3,2)
  \tkzFct[color = red,domain = 1./3:3]{0.125*(3*x-1)+0.375*(3*x-1)/(x*x)}
  \tkzFct[color = green,domain = 1./3:3]{0.125*(3*x-1)}
  \tkzSetUpPoint[shape=circle, size = 10, color=black, fill=lightgray]
  \tkzDefPointByFct[draw,with = a](1)
  \tkzDefPointByFct[draw,with = a](2)
  \tkzDefPointByFct[draw,with = a](3)
  \tkzDefPointByFct[draw,with = b](3)
  \tkzDefPointByFct[draw,with = b](1/3)
  \tkzText[draw,color= red,fill=red!20](1,1.5) %
    {\$f(x)=\frac{1}{8}(3x-1)+\frac{3}{8}\left(\frac{3x-1}{x^2}\right)%
    \left(\frac{3x-1}{x^2}\right)\$}
  \tkzText[draw,color= green!50!black,fill=green!20]%
    (2,0.3){\$g(x)=\frac{1}{8}(3x-1)\$}
\end{tikzpicture}
```



5 Labels

Ce qui est souhaitable, c'est de pouvoir nommer les courbes. Prenons comme exemple, la fonction f définie par :

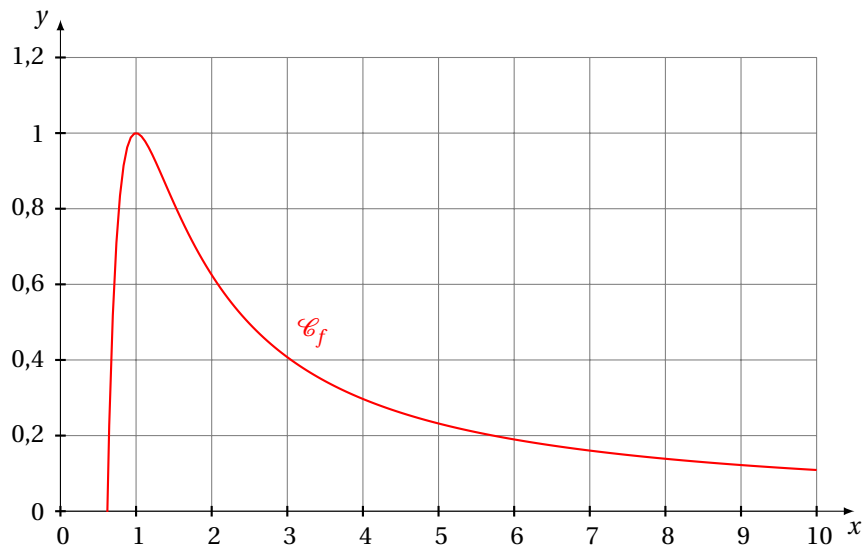
$$x > 0 \text{ et } f(x) = \frac{x^2 + 1}{x^3}$$

Il est assez aisé de mettre un titre en utilisant la macro `\tkzText` du package `tkz-base`. Les coordonnées utilisées font référence aux unités des axes du repère. Pour placer un texte le long de la courbe, le plus simple est choisir un point de la courbe, puis d'utiliser celui-ci pour afficher le texte.

```
1 \tkzDefPointByFct(3)
2 \tkzText[above right](tkzPointResult){$\mathcal{C}_f$}
```

La première ligne détermine un point de la courbe. Ce point est rangé dans `tkzPointResult`. Il suffit d'utiliser `\tkzText` avec ce point comme argument comme le montre la seconde ligne. Les options de TikZ permettent d'affiner le résultat.

5.1 Ajouter un label



```
\begin{tikzpicture}
  \tkzInit[xmin=0,xmax=10,
    ymin=0,ymax=1.2,ystep=0.2]
  \tkzGrid
  \tkzAxeXY
  \tkzClip
  \tkzFct[thick,color=red,domain=0.55:10]{(\x*\x+\x-1)/(\x**3)}
  \tkzText(3,-0.3){\textbf{Courbe de } $\mathbf{f}$}
  \tkzDefPointByFct(3)
  \tkzText[above right,text=red](tkzPointResult){$\mathcal{C}_f$}
\end{tikzpicture}
```

6 Macros pour tracer des tangentes

Si une seule fonction est utilisée, elle est stockée avec comme nom `\tkzFcta`, si une deuxième fonction est utilisée, elle sera stockée avec comme nom `\tkzFctb`, et ainsi de suite... Si plusieurs fonctions sont présentes dans un même environnement alors l'option `with` permet de choisir celle qui sera mise à contribution.

Il faut bien évidemment, avoir initialisé l'environnement à l'aide `\tkzInit`, avant d'appeler `\tkzFct` et `\tkzDrawTangentLine`. Pour la longueur des vecteurs représentant les demi-tangentes, il faut attribuer une valeur aux coefficients `kl` et `kr`. $kl = 0$ ou $kr = 0$ annule le dessin de la demi-tangente correspondante ($l=left$) et ($r=right$). Si `xstep=1` et `ystep=1` alors si la pente est égale à 1, la demi-tangente a pour mesure $\sqrt{2}$. Dans les autres cas si AT est la longueur de la demi-tangente et si p est la pente alors \vec{AT} a pour coordonnées $(kl, kl \cdot p)$.

6.1 Représentation d'une tangente `\tkzDrawTangentLine`

`\tkzDrawTangentLine[⟨local options⟩](⟨a⟩)`

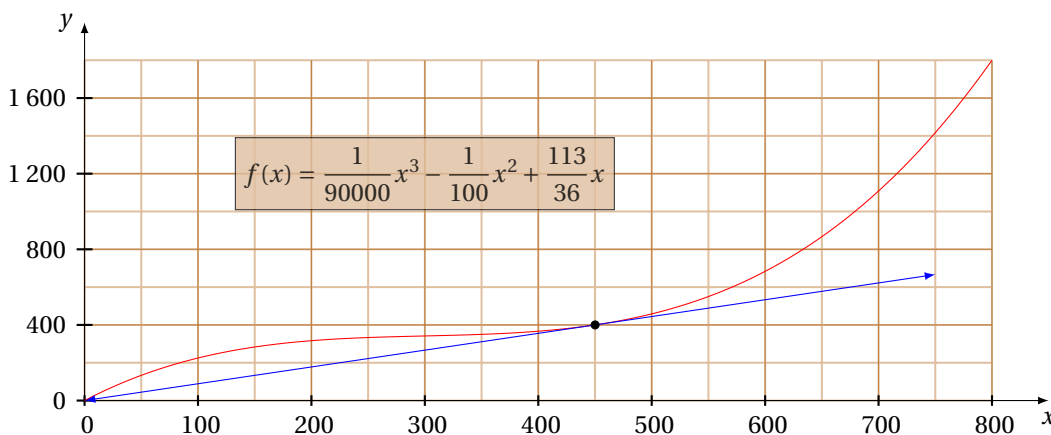
On l'emploie soit juste après l'utilisation de `\tkzFct`, sinon il faut donner la référence de la fonction à l'aide de l'option `with`.

options	exemple	explication
a	<code>\tkzDrawTangentLine(0)</code>	tangente au point d'abscisse 0

Les options sont celles de TikZ comme `color` ou `style` plus les options suivantes

options	défaut	définition
draw	false	booléen si true alors le point de contact est tracé
with	a	permet de choisir une fonction
kr	1	coefficient pour la longueur de la demi-tangente à droite
kl	1	coefficient pour la longueur de la demi-tangente à gauche

6.2 Tangente avec `xstep` et `ystep` différents de 1



Il faut remarquer qu'il n'est point nécessaire de faire des calculs. Il suffit d'utiliser les valeurs qui correspondent aux graduations.

On peut changer le style des tangentes avec, par exemple,

`\tikzset{tan style/.style={-}}` par défaut on a :

```

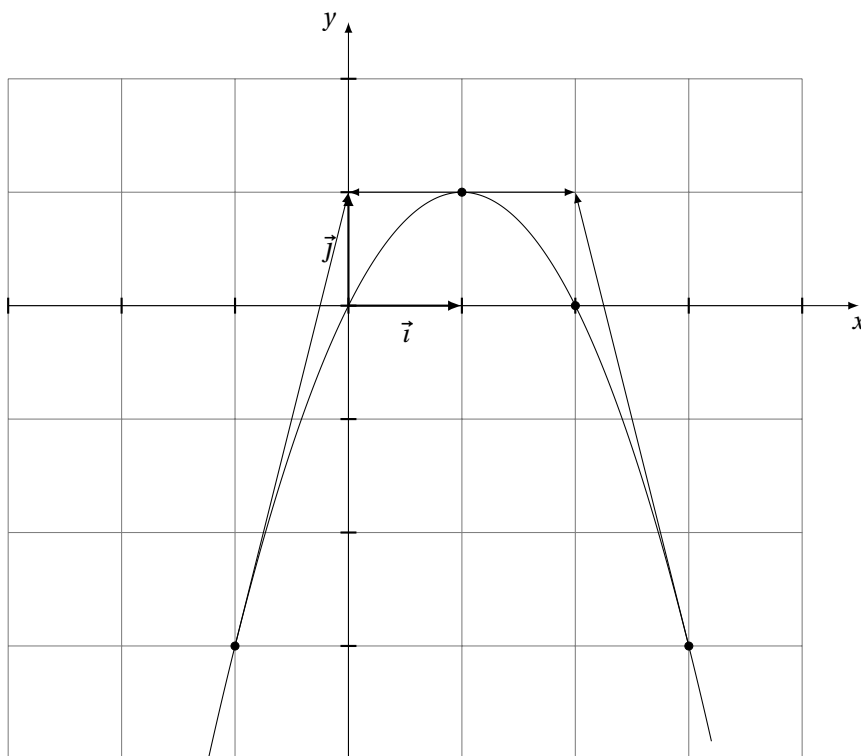
\tikzset{tan style/.style={->,>=latex}}

\begin{tikzpicture}[xscale=1.5]
\tikzset{tan style/.style={-}}
\tkzInit[xmin=0,xmax=800,xstep=100,
        ymin=0,ymax=1800,ystep=400]
\tkzGrid[color=brown,sub,subxstep=50,subystep=200](0,0)(800,1800)
\tkzAxeXY
\tkzFct[color=red,samples=100,domain = 0:800]%
    {(1./900000)*\x*\x*\x-(1./100)*\x*\x+(113./36)*\x}
\tkzDrawTangentLine[color=blue,kr=300,kl=450,coord](450)
\tkzText[draw,color=black,%
        fill=brown!50,opacity=0.8](300,1200)%
    {$f(x)=\dfrac{1}{900000}x^3-\dfrac{1}{100}x^2+\dfrac{113}{36}x$}
\end{tikzpicture}

```

6.3 Les options kl, kr et l'option draw

Si l'un des deux nombres kl ou kr est nul alors seulement une demi-tangente est tracée sinon ces nombres représentent un pourcentage de la longueur initiale de la tangente. L'option draw permet de tracer le point de contact.



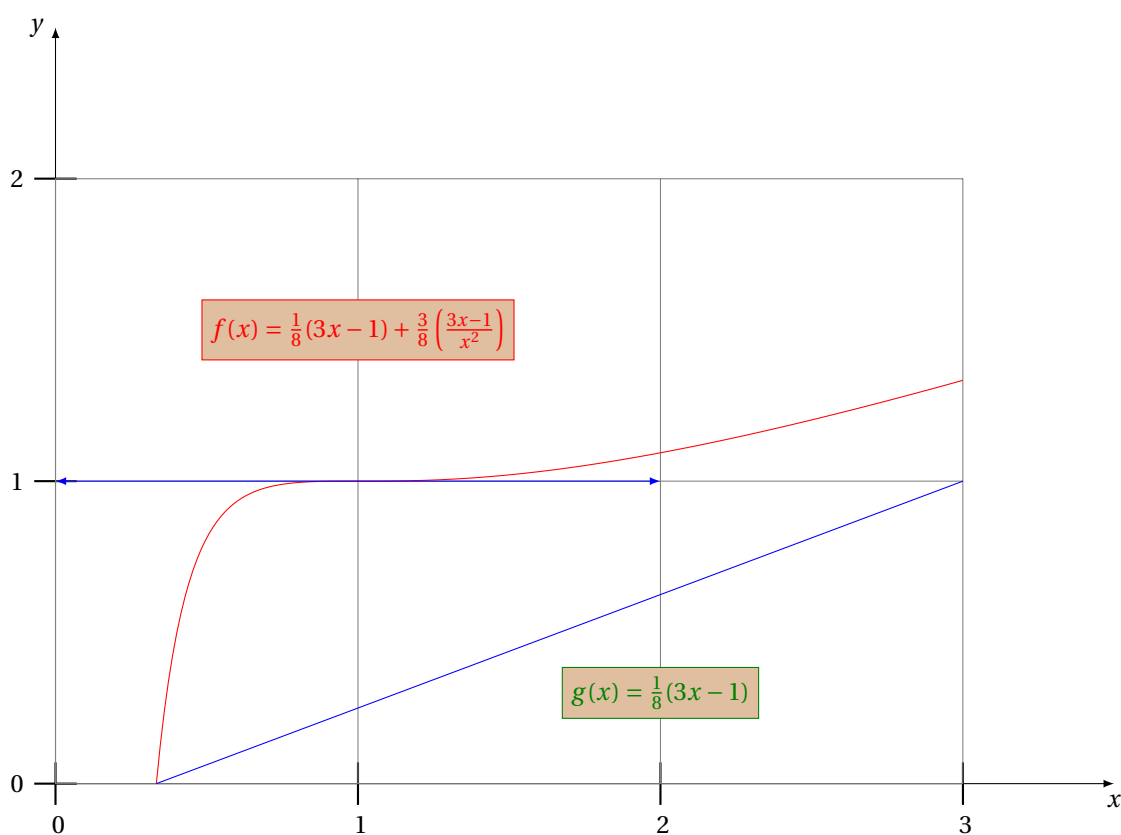

```

\begin{tikzpicture}[scale=1.5]
\tkzInit[xmin=-3,xmax=4,ymin=-4,ymax=2]
\tkzGrid \tkzDrawXY \tkzClip
\tkzFct[domain = -2.15:3.2]{(-x*x)+2*x}
\tkzDefPointByFct[draw](2)
\tkzDrawTangentLine[kl=0,draw](-1)
\tkzDrawTangentLine[draw](1)
\tkzDrawTangentLine[kr=0,draw](3)
\tkzRep
\end{tikzpicture}

```

6.4 Tangente et l'option with

Soit on place la macro `\tkzDrawTangentLine` après la ligne qui définit la première fonction (*a*), soit on trace une autre fonction avant, et dans ce cas, il est nécessaire de préciser quelle fonction sera utilisée. pour se faire, on utilise l'option `with`.

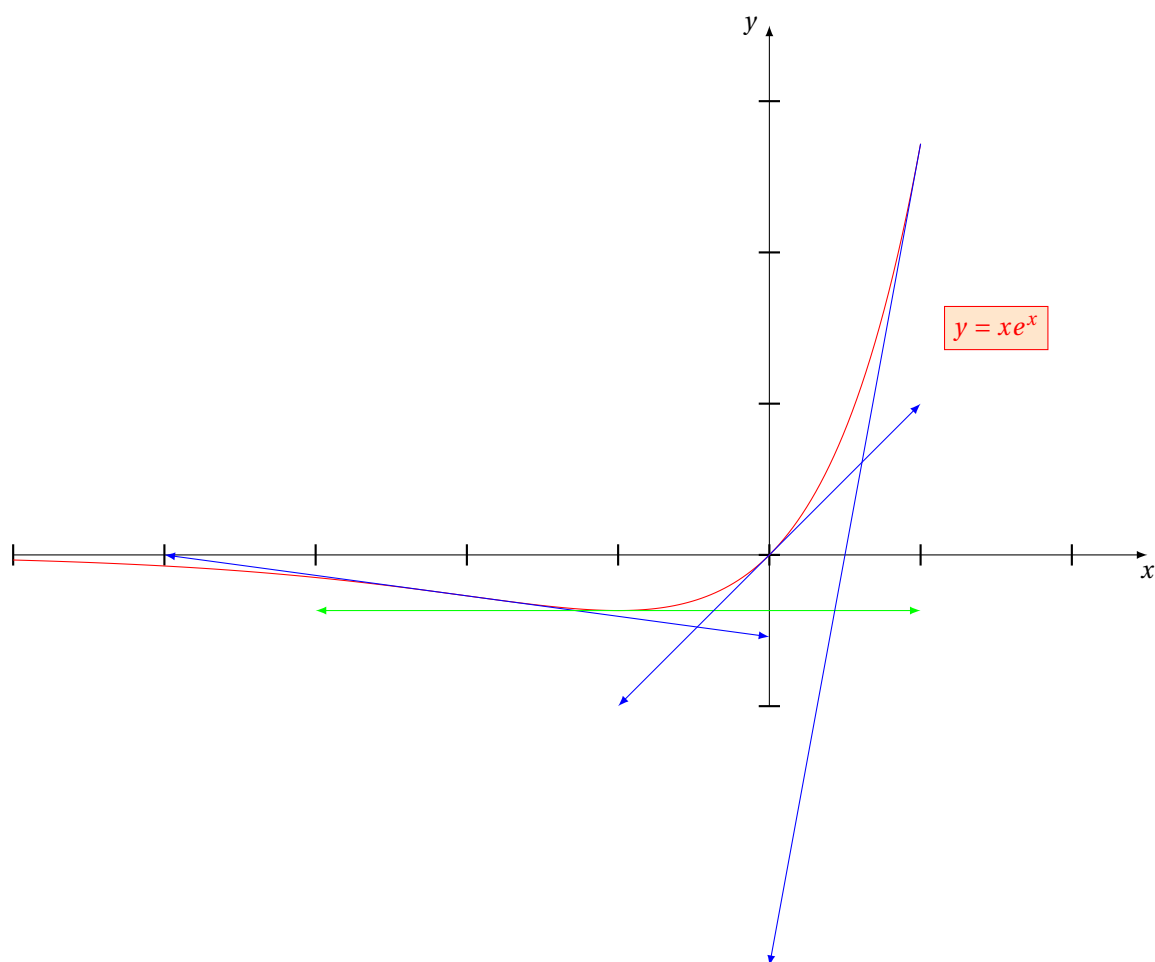


```

\begin{tikzpicture}[scale=4]
  \tkzInit[xmax=3,ymax=2]
  \tkzAxeXY
  \tkzGrid(0,0)(3,2)
  \tkzFct[color = red, domain = 1/3:3]{0.125*(3*x-1)+0.375*(3*x-1)/(x*x)}
  \tkzFct[color = blue, domain = 1/3:3]{0.125*(3*x-1)}
  \tkzDrawTangentLine[with=a,
                     color=blue](1)
  \tkzText[draw,
           color= red,
           fill=brown!50](1,1.5)%
    {$f(x)=\frac{1}{8}(3x-1)+\frac{3}{8}\left(\frac{3x-1}{x^2}\right)$}
  \tkzText[draw,
           color= green!50!black,
           fill=brown!50](2,0.3)%
    {$g(x)=\frac{1}{8}(3x-1)$}
\end{tikzpicture}

```

6.5 Quelques tangentes



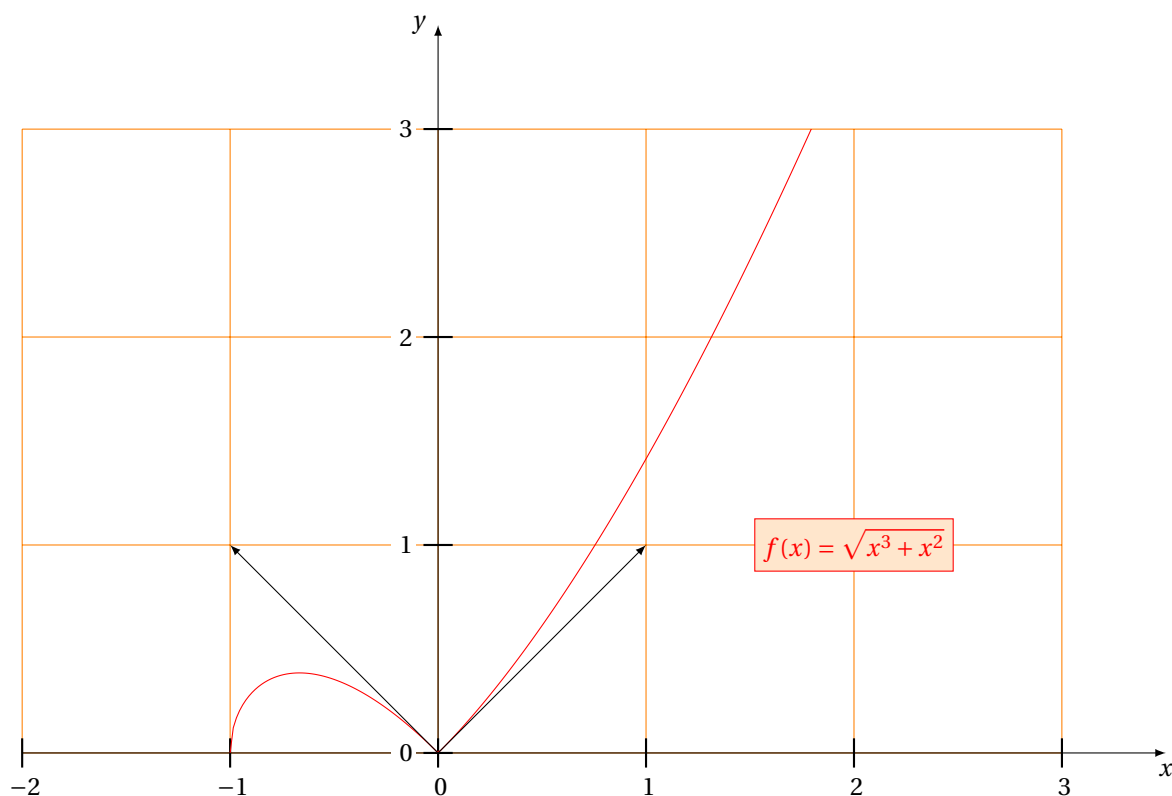
```
\begin{tikzpicture}[scale=2]
  \tkzInit[xmin=-5,xmax=2,ymin=-1,ymax=3]
  \tkzDrawX
  \tkzDrawY
  \tkzText[draw,color = red,fill = orange!20](1.5,1.5){$y = xe^x$}
  \tkzFct[color = red, domain = -5:1]{x*exp(x)}%
  \tkzDrawTangentLine[color=blue,kr=2,kl=2](-2)
  \tkzDrawTangentLine[color=green,kr=2,kl=2](-1)
  \tkzDrawTangentLine[color=blue](0)
  \tkzDrawTangentLine[color=blue,kr=0](1)
\end{tikzpicture}
```

6.6 Demi-tangentes

Il faut remarquer que les tangentes sont en réalité deux demi-tangentes ce qui permet d'obtenir simplement le résultat ci-dessous.

Possible sont les écritures $((x+1)*x)*x)^{0.5}$, $(x^3+x^2)^{0.5}$ et $(x*x*x+x*x)^{0.5}$.

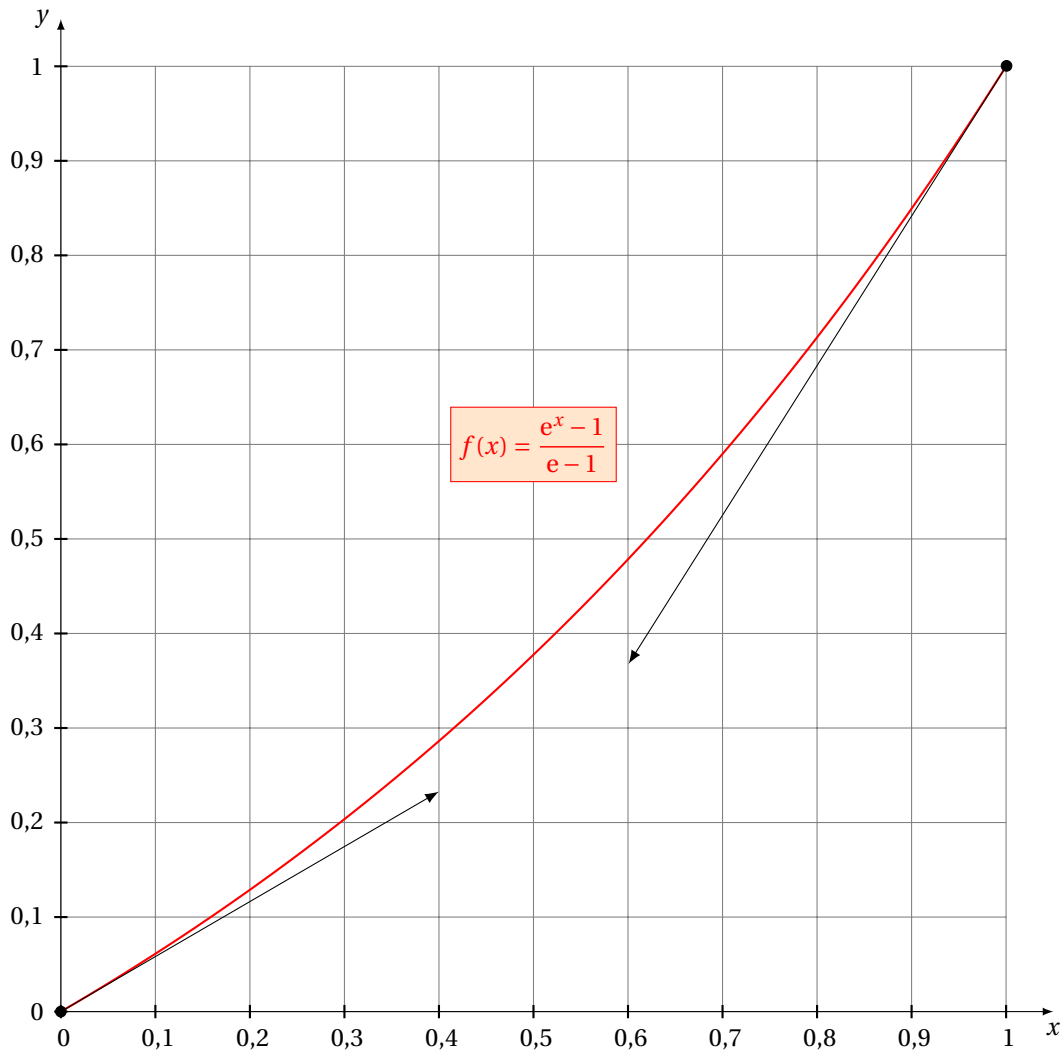
Dans cet exemple, les deux demi-tangentes sont obtenues automatiquement :



```
\begin{tikzpicture}[scale=2.75]
  \tkzInit[xmin=-2,xmax=3,ymax=3]
  \tkzGrid[color=orange](-2,0)(3,3)
  \tkzAxeX
  \tkzAxeY
  \tkzFct[color = red ,domain = -1:2]{(((x+1)*x)*x)**0.5}
  \tkzDrawTangentLine(0)
  \tkzText[draw,color = red,fill = orange!20](2,1){$f(x)=\sqrt{x^3+x^2}$}
\end{tikzpicture}
```

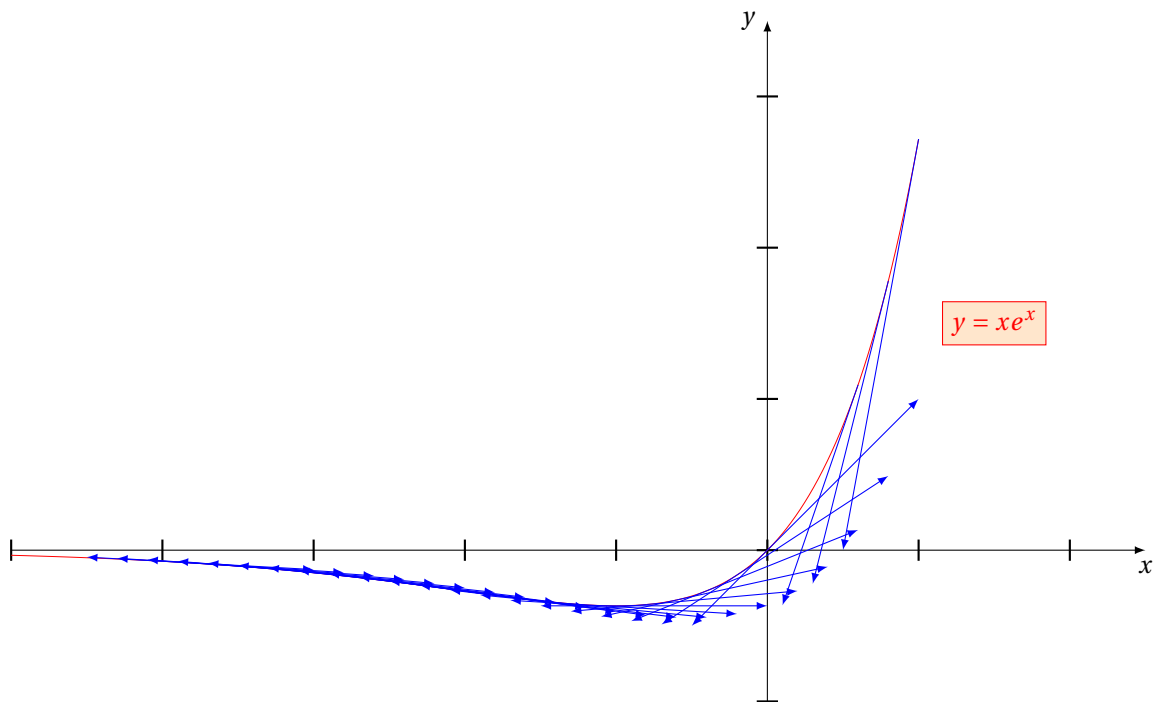
6.7 Demi-tangentes Courbe de Lorentz

Ici, on ne veut que les demi-tangentes comprises entre 0 et 1, pour cela il suffit dans un cas de donner la valeur 0 à `kr` et dans l'autre à `kl`.



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmax=1,ymax=1,xstep=0.1,ystep=0.1]
  \tkzGrid(0,0)(1,1)
  \tkzAxeXY
  \tkzFct[color = red,thick, domain =0:1]{(exp(\x)-1)/(exp(1)-1)}
  \tkzSetUpPoint[size=4]
  \tkzDrawTangentLine[draw, kl = 0, kr = 0.4](0)
  \tkzDrawTangentLine[draw, kl = 0.4,kr = 0 ](1)
  \tkzText[draw,color = red,fill = orange!20](0.5,0.6)%
    {\$f(x)=\dfrac{\text{e}^x-1}{\text{e}-1}\$}
\end{tikzpicture}
```

6.8 Série de tangentes



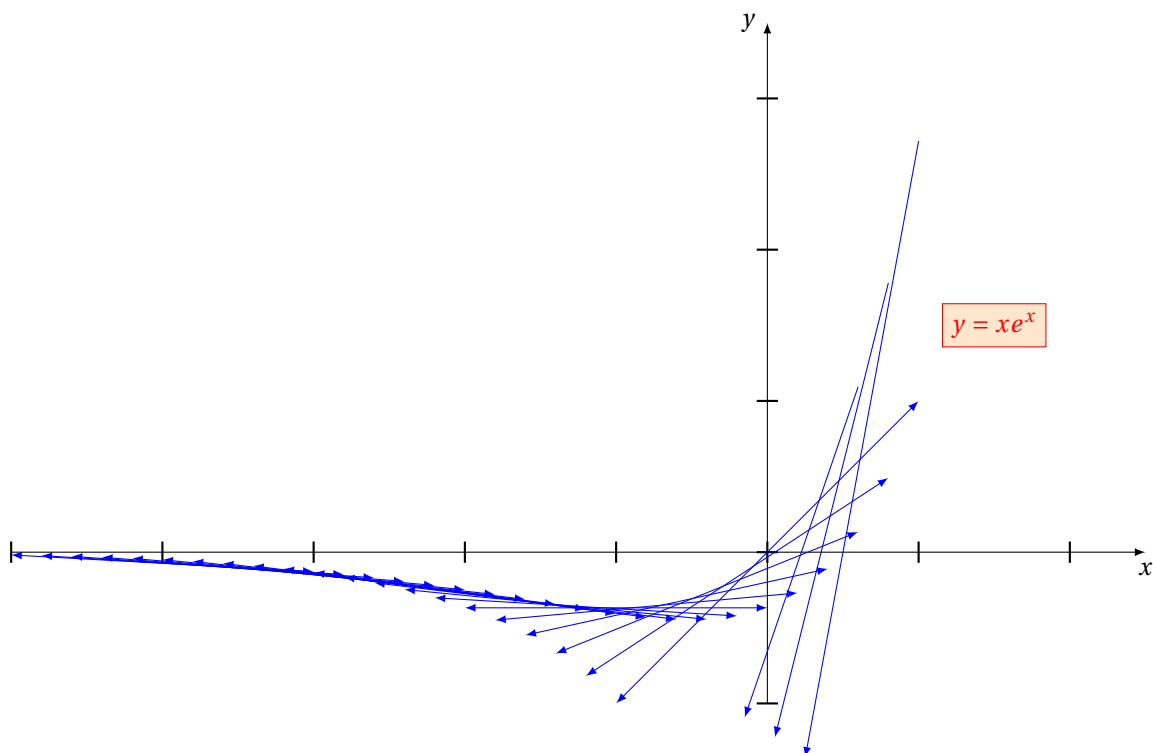
```
\begin{tikzpicture}[scale=2]
  \tikzstyle{tan style}=[-]
  \tkzInit[xmin=-5,xmax=2,ymin=-1,ymax=3]
  \tkzDrawXY
  \tkzText[draw,color = red, fill = orange!20](1.5,1.5){$y = xe^x$}
  \tkzFct[line width = 0.01 pt,color = red, domain = -5:1]{x*exp(x)}
  \foreach \x in {-4,-3.8,...,0}{%
    \tkzDrawTangentLine[color=blue,line width=.4pt,kr=1,kl=0.5](\x)}
  \foreach \x in {0.6,0.8,1}{%
    \tkzDrawTangentLine[color=blue,line width=.4pt, kr=0,kl=0.5](\x)}
\end{tikzpicture}
```

6.9 Série de tangentes sans courbe

Pour cela, il faut définir la dernière expression avec la syntaxe de `fp.sty`.

Définition de `\tkzFctLast`

```
\global\edef\tkzFctLast{x*exp(x)}
```

6.9.1 Utilisation de `\tkzFctLast`

```

\begin{tikzpicture}[scale=2]
  \tikzstyle{tan style}=[-]
  \tkzInit[xmin=-5,xmax=2,ymin=-1,ymax=3]
  \tkzDrawXY
  \tkzText[draw,color = red, fill = orange!20](1.5,1.5){$y = xe^x$}
  \global\edef\tkzFctLast{x*exp(x)}% c'est la ligne importante
  \foreach \v in {-4,-3.8,...,0}{%
    \tkzDrawTangentLine[color=blue,line width=.4pt,kl=1](\v)}
  \foreach \v in {0.6,0.8,1}{%
    \tkzDrawTangentLine[color=blue,line width=.4pt,kr=0,kl=.75](\v)}
\end{tikzpicture}

```

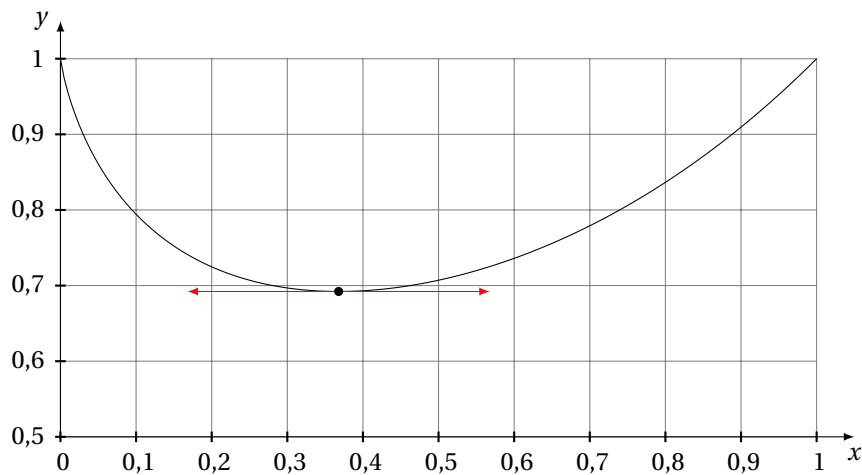
6.10 Calcul de l'antécédent

Un problème surgit si on emploie une expression contenant des parenthèses dans l'argument, ainsi $\{1/\exp(1)\}$ est correct mais $(1/\exp(1))$ donne une erreur. Il est aussi possible d'évaluer l'antécédent postérieurement comme cela :

6.10.1 Valeur numérique de l'antécédent

```
\FPeval\vx{1/exp(1)}
```

6.10.2 utilisation de la valeur numérique



```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=1,xstep=0.1,ymin=0.5,ymax=1,ystep=0.1]
  \tkzGrid      \tkzAxeXY
  \tkzFct[domain = 0.00001:1]{(\x**\x)}
  \tkzDrawTangentLine[draw,color = red, kr = 0.2,kl = 0.2](1/exp(1))
\end{tikzpicture}
```

7 Macros pour définir des surfaces

Il s'agit par exemple de représenter la partie du plan comprise entre la courbe représentative d'une fonction, l'axe des abscisses et les droites d'équation $x = a$ et $x = b$.

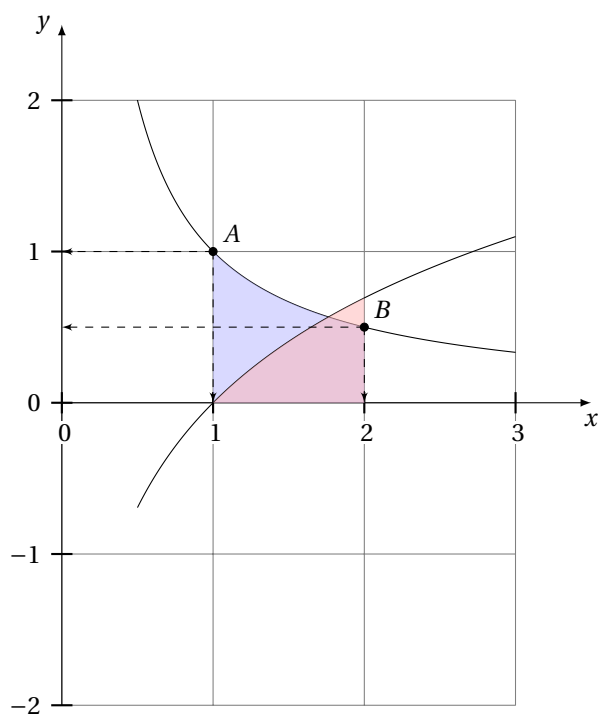
7.1 Représentation d'une surface `\tkzDrawArea` ou `\tkzArea`

`\tkzDrawArea[⟨local options⟩]`

Les options sont celles de TikZ.

options	défaut	définition
domain	-5:5	domaine de la fonction
with	a	référence de la fonction
color	200	nombre de points utilisés
opacity	no default	transparence
style	black	couleur de la ligne

7.2 Naissance de la fonction logarithme népérien

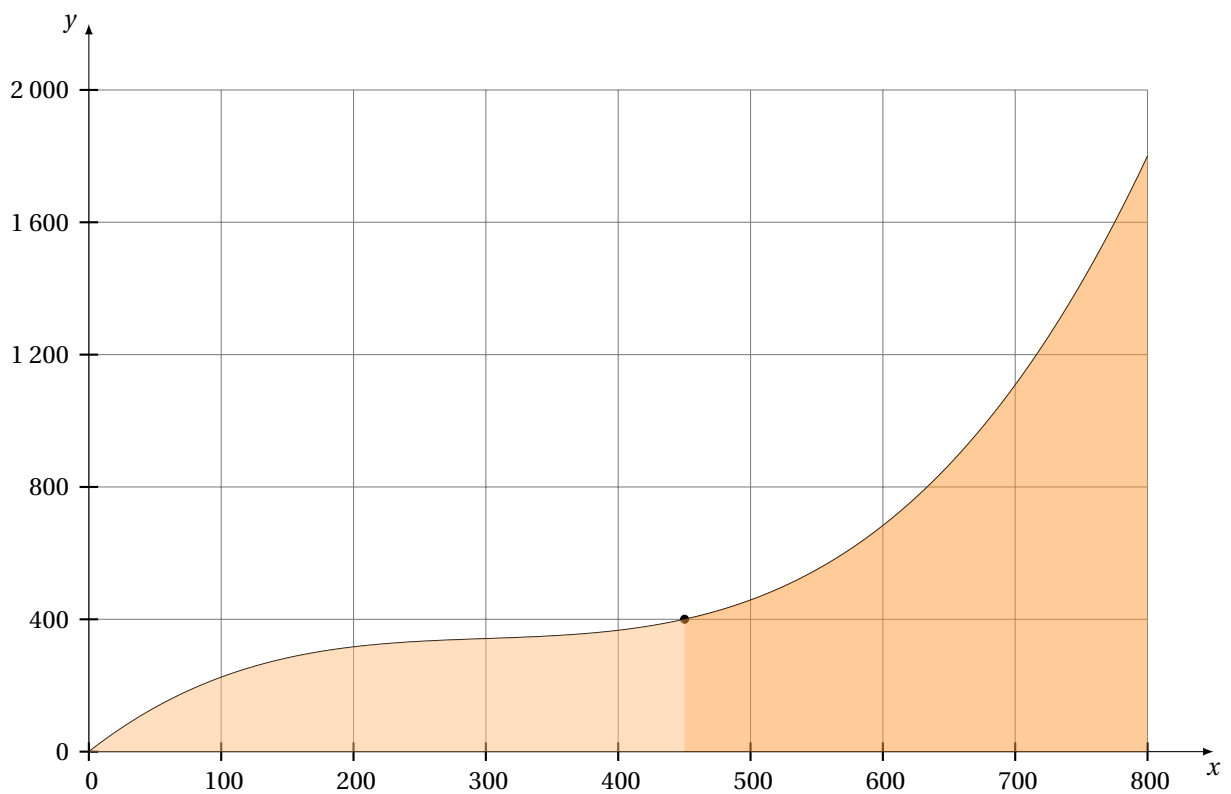



```

\begin{tikzpicture}[scale=2]
  \tkzInit[xmin=0,xmax=3,xstep=1,
           ymin=-2,ymax=2,ystep=1]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain= 0.4:3]{1./x}
  \tkzDefPointByFct(1)
  \tkzGetPoint{A}
  \tkzDefPointByFct(2)
  \tkzGetPoint{B}
  \tkzLabelPoints[above right](A,B)
  \tkzDrawArea[color=blue!30,
               domain = 1:2]
  \tkzFct[domain = 0.5:3]{log(x)}
  \tkzDrawArea[color=red!30,
               domain = 1:2]
  \tkzPointShowCoord(A)
  \tkzPointShowCoord(B)
  \tkzDrawPoints(A,B)
\end{tikzpicture}

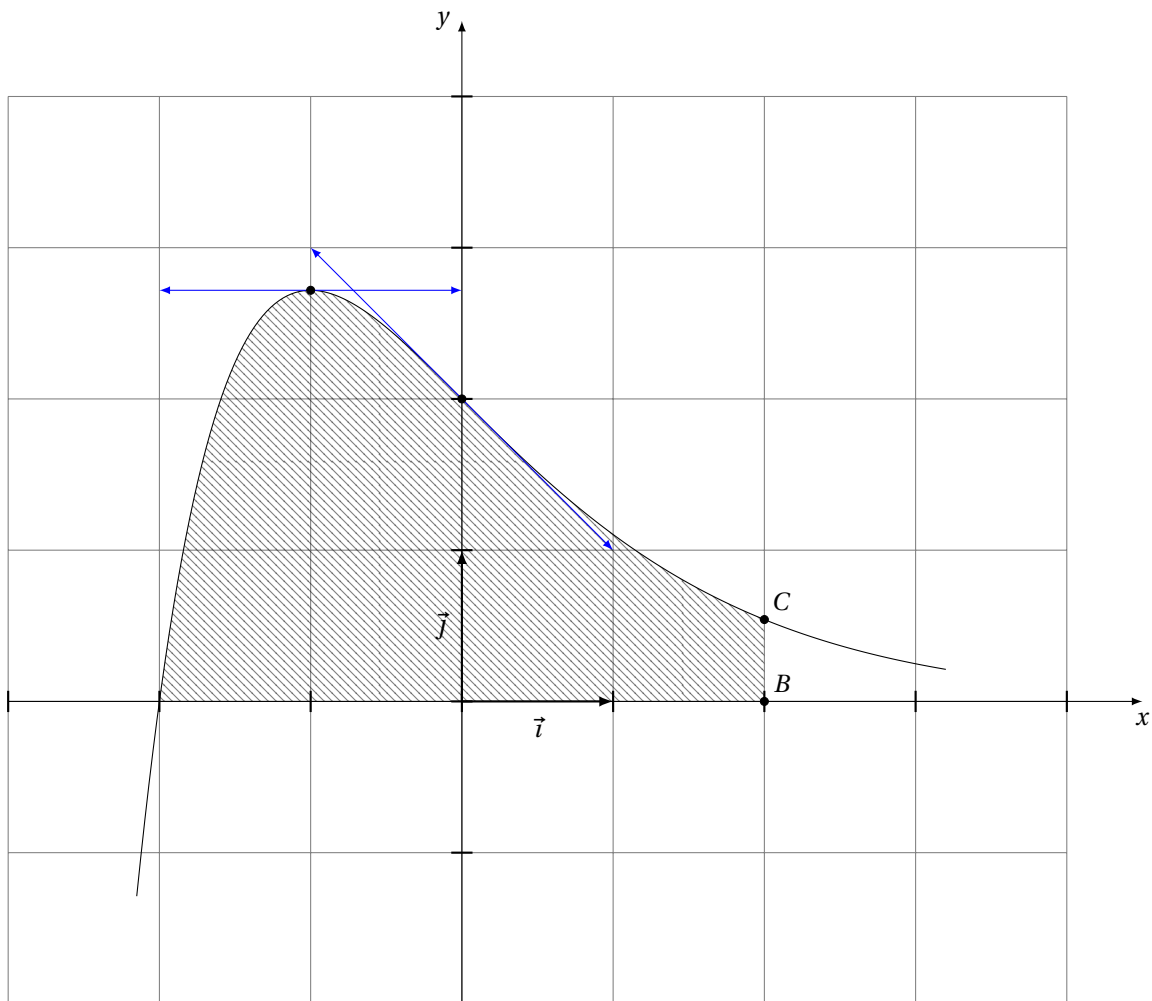
```

7.3 Surface simple



```
\begin{tikzpicture}[scale=1.75]
\tkzInit[xmin=0,xmax=800,xstep=100,
        ymin=0,ymax=2000,ystep=400]
\tkzGrid
\tkzAxeXY
\tkzFct[domain = 0:800]{(1./90000)*\x*\x*\x-(1./100)*\x*\x+(113./36)*\x}
\tkzDefPoint(450,400){a}
\tkzDrawPoint(a)
\tkzDrawArea[color=orange!50, domain =0:450]
\tkzDrawArea[color=orange!80, domain =450:800]
\end{tikzpicture}
```

7.4 Surface et hachures



```

\begin{tikzpicture}[scale=2]
  \tkzInit[xmin=-3,xmax=4,ymin=-2,ymax=4]
  \tkzGrid(-3,-2)(4,4)
  \tkzDrawXY
  \tkzFct[domain = -2.15:3.2]{(2+\x)*exp(-\x)}
  \tkzDrawArea[pattern=north west lines,domain =-2:2]
  \tkzDrawTangentLine[draw,color=blue](0)
  \tkzDrawTangentLine[draw,color=blue](-1)
  \tkzDefPointByFct(2) \tkzGetPoint{C}
  \tkzDefPoint(2,0){B}
  \tkzDrawPoints(B,C) \tkzLabelPoints[above right](B,C)
  \tkzRep
\end{tikzpicture}

```

7.5 Surface comprise entre deux courbes `\tkzDrawAreafg`

`\tkzDrawAreafg[(local options)]`

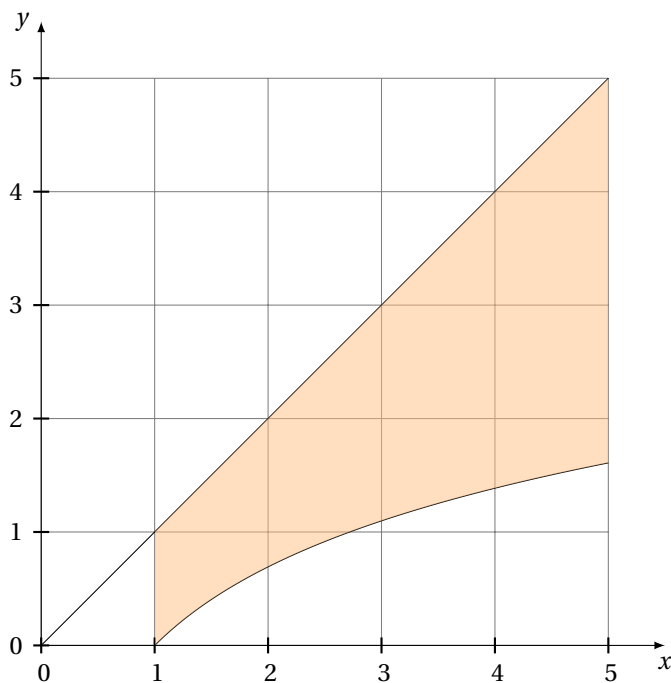
Cette macro permet de mettre en évidence une surface délimitée par les courbes représentatives de deux fonctions. La courbe (a) doit être au-dessus de la courbe (b).

options	défaut	explication
between	a and b	référence des deux courbes
domain= min:max	domain=-5:5	Les options sont celles de TikZ.
opacity	0.5	transparence

*L'option **pattern** de TikZ peut être utile!*

7.6 Surface comprise entre deux courbes en couleur

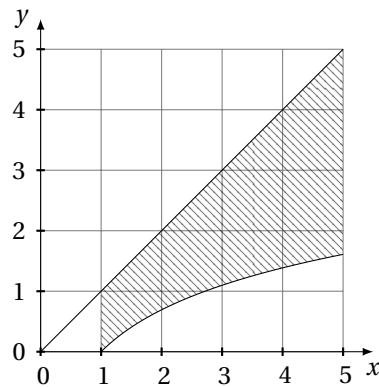
Par défaut, la surface définie est comprise entre les deux premières courbes.



```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmax=5,ymax=5]
  \tkzGrid \tkzAxeXY
  \tkzFct[domain = 0:5]{x}
  \tkzFct[domain = 1:5]{log(x)}
  \tkzDrawAreafg[color = orange!50,domain = 1:5]
\end{tikzpicture}
```

7.7 Surface comprise entre deux courbes avec des hachures

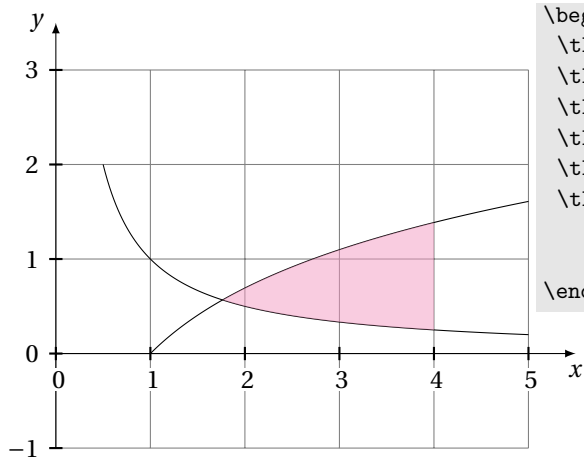
`\tkzDrawAreafg[between= a and b,pattern=north west lines,domain = 1:5]`



```
\begin{tikzpicture}[scale=.8]
  \tkzInit[xmax=5,ymax=5]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain = 0:5]{x}
  \tkzFct[domain = 1:5]{log(x)}
  \tkzDrawAreafg[between= a and b,pattern=north west lines,domain = 1:5]
\end{tikzpicture}
```

7.8 Surface comprise entre deux courbes avec l'option between

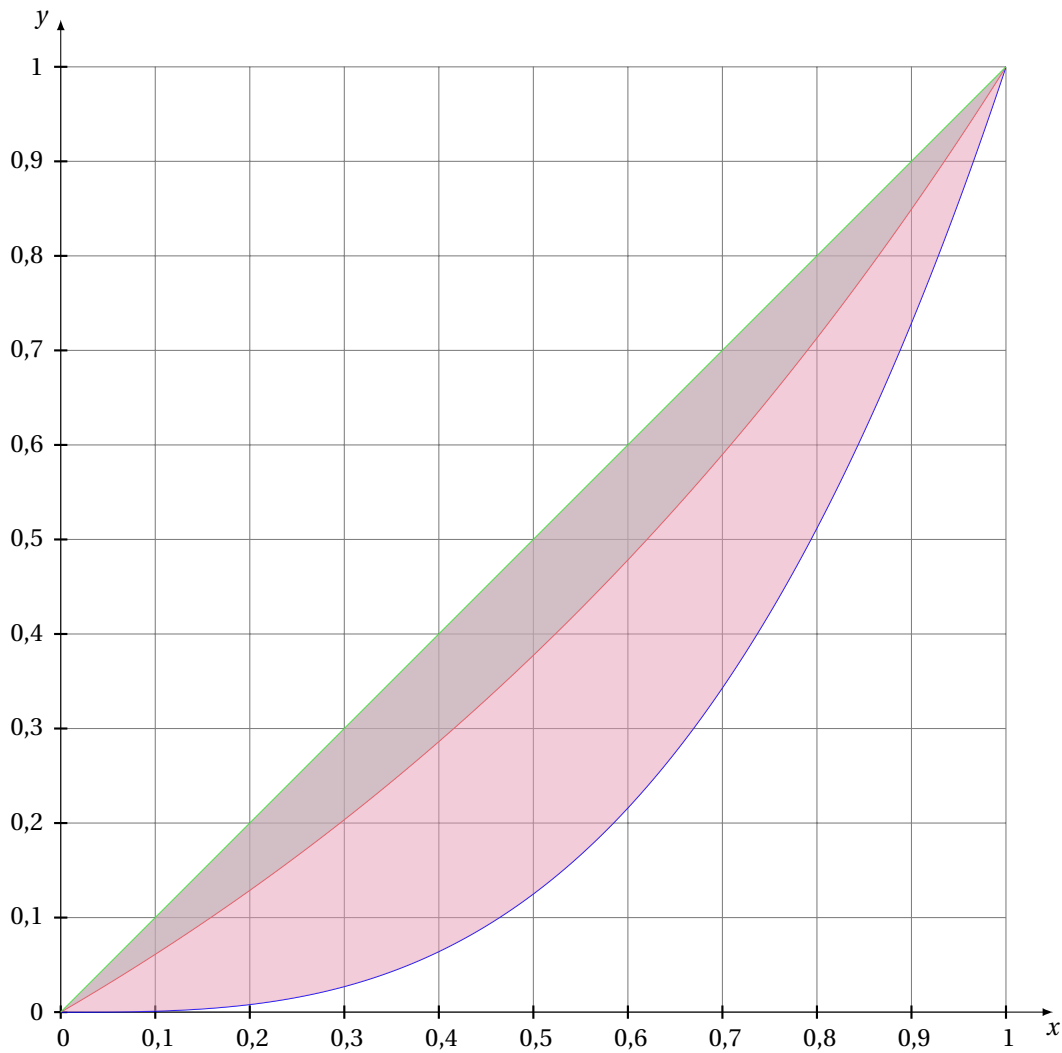
Attention à l'ordre des références dans l'option **between**. Seule la partie de la surface (b) est au-dessus de (a) est représentée.



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[ymin=-1,xmax=5,ymax=3]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain = 0.5:5]{1/x}% courbe a
  \tkzFct[domain = 1:5]{log(x)}% courbe b
  \tkzDrawAreafg[between=b and a,
    color=magenta!50,
    domain = 1:4]
\end{tikzpicture}
```

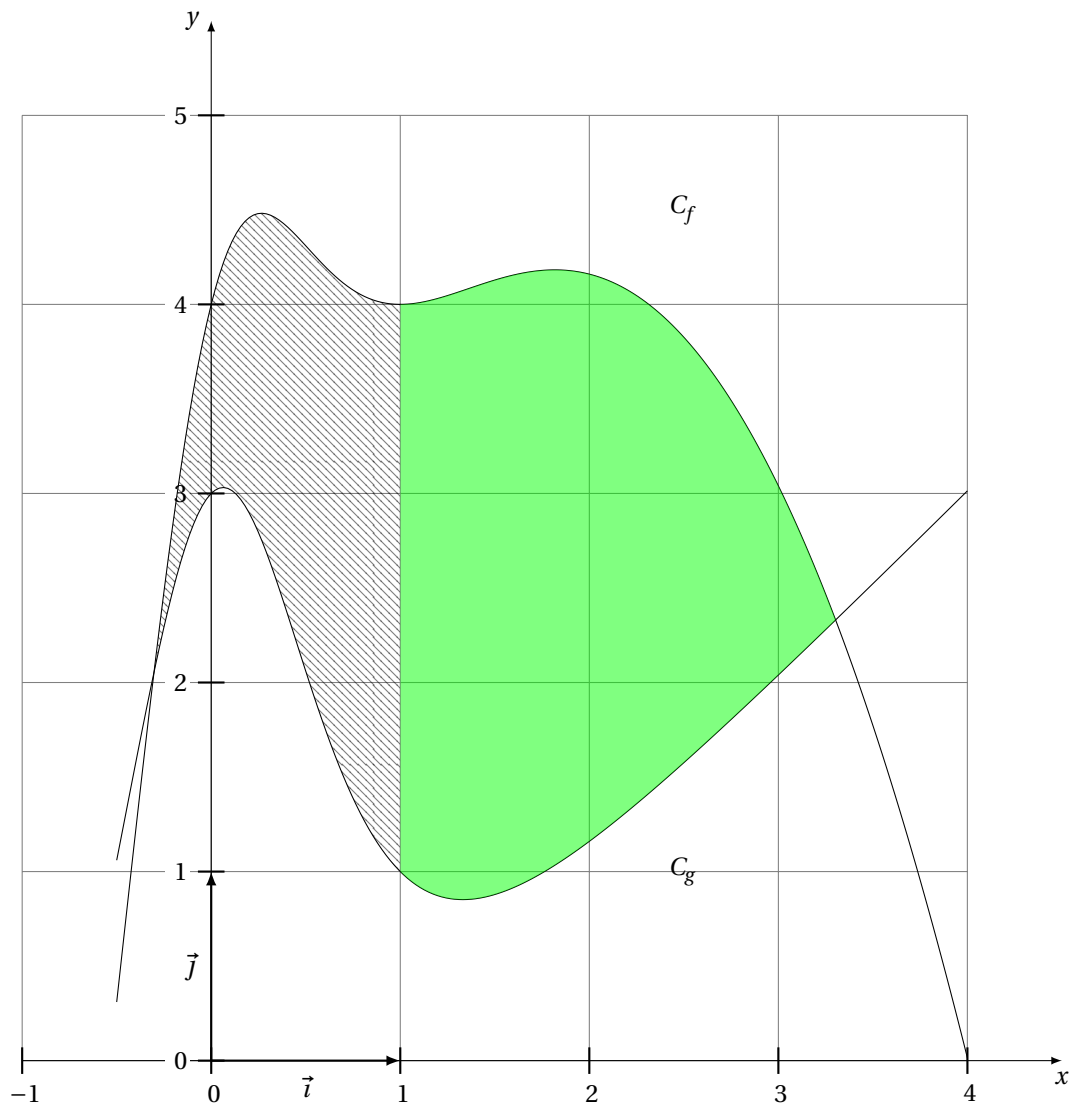
7.9 Surface comprise entre deux courbes : courbes de Lorentz

Ici aussi, attention à l'ordre des références dans l'option `between`.



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmax=1,ymax=1,xstep=0.1,ystep=0.1]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[color = red,domain = 0:1]{(exp(\x)-1)/(exp(1)-1)}
  \tkzFct[color = blue,domain = 0:1]{\x*\x*\x}
  \tkzFct[color = green,domain = 0:1]{\x}
  \tkzDrawAreaafg[between = c and b,color=purple!40,domain = 0:1]
  \tkzDrawAreaafg[between = c and a,color=gray!60,domain = 0:1]
\end{tikzpicture}
```

7.10 Mélange de style



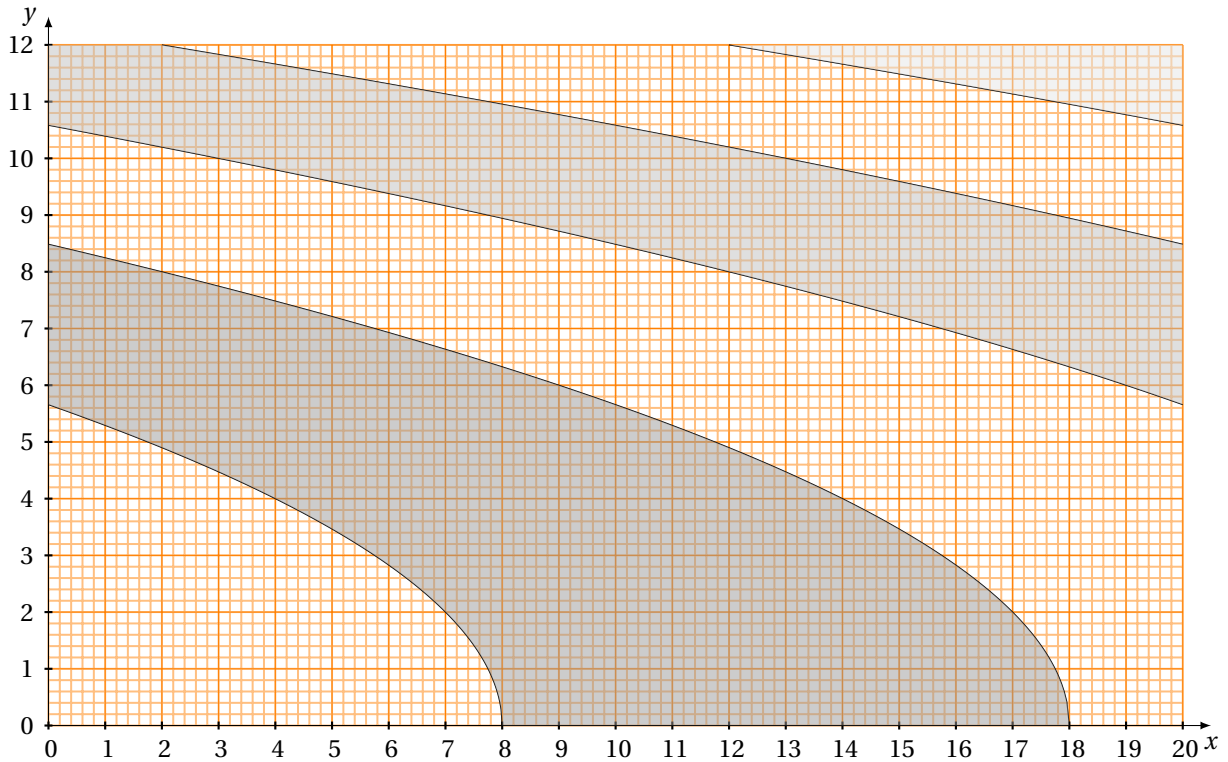
```

\begin{tikzpicture}[scale=2.5]
  \tkzInit[xmin=-1,xmax=4,ymin=0,ymax=5]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain = -.5:4]{ 4*x-x**2+4/(x**2+1)**2}
  \tkzFct[domain = -.5:4]{x-1+4/(x**2+1)**2}
  \tkzDrawAreaafg[color=green,domain = 1:4]
  \tkzDrawAreaafg[pattern=north west lines,domain = -.5:1]
  \tkzRep
  \tkzText(2.5,4.5){$C_f$}
  \tkzText(2.5,1){$C_g$}
\end{tikzpicture}%

```

7.11 Courbes de niveaux

Le code est intéressant pour la définition des fonctions constantes aux lignes 10 et 11.



```

1 \begin{tikzpicture}[scale=.75]
2   \tkzInit[xmax=20,ymax=12]
3   \tkzGrid[color=orange,sub](0,0)(20,12)
4   \tkzAxeXY
5   \tkzFct[samples=400,domain =0:8]{(32-4*x)**(0.5)} % a
6   \tkzFct[samples=400,domain =0:18]{(72-4*x)**(0.5)} % b
7   \tkzFct[samples=400,domain =0:20]{(112-4*x)**(0.5)} % c
8   \tkzFct[samples=400,domain =2:20]{(152-4*x)**(0.5)} % d
9   \tkzFct[samples=400,domain =12:20]{(192-4*x)**(0.5)} % e
10  \def\tkzFctgnuf{0} % f
11  \def\tkzFctgnug{12} % g
12  \tkzDrawAreafg[between= b and a,color=gray!80,domain = 0:8]
13  \tkzDrawAreafg[between= b and f,color=gray!80,domain = 8:18]
14  \tkzDrawAreafg[between= d and c,color=gray!50,domain = 2:20]
15  \tkzDrawAreafg[between= g and c,color=gray!50,domain = 0:2]
16  \tkzDrawAreafg[between= g and e,color=gray!20,domain =12:20]
17 \end{tikzpicture}%

```


8 Sommes de Riemann

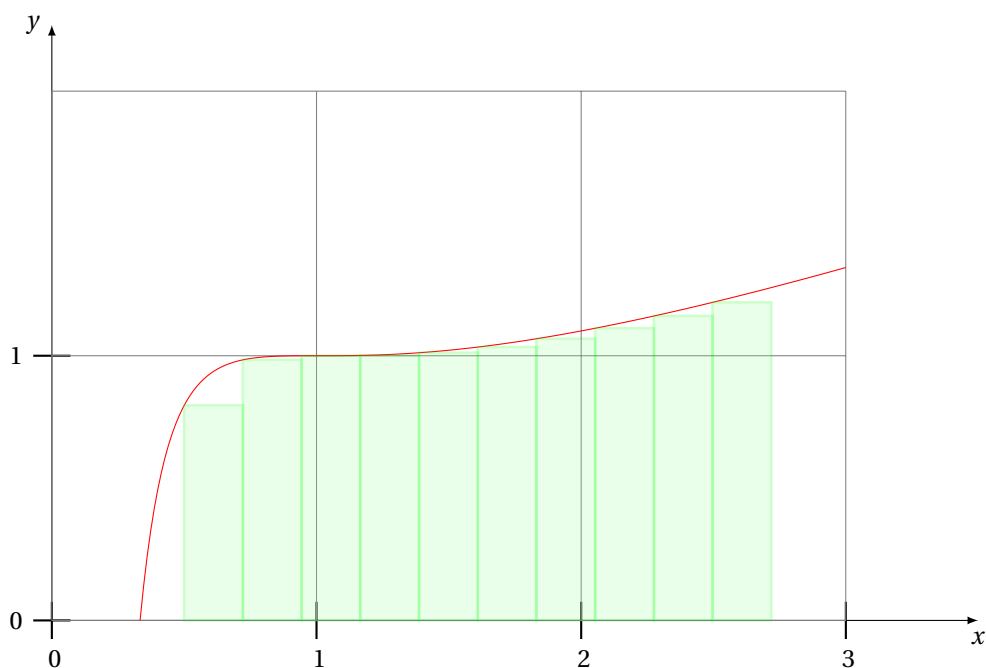
`\tkzDrawRiemannSum[⟨local options⟩]{⟨f(t)⟩}`

Cette macro permet de représenter les rectangles intervenant dans une somme de Riemann. Les options sont celles de TikZ, plus

options	défaut	définition
interval	1:2	l'intervalle sur lequel est appliqué la méthode
number	10	nombre de sous-intervalles utilisés

Possible est de réunir les quatres macros et de choisir la méthode avec une option.

8.1 Somme de Riemann

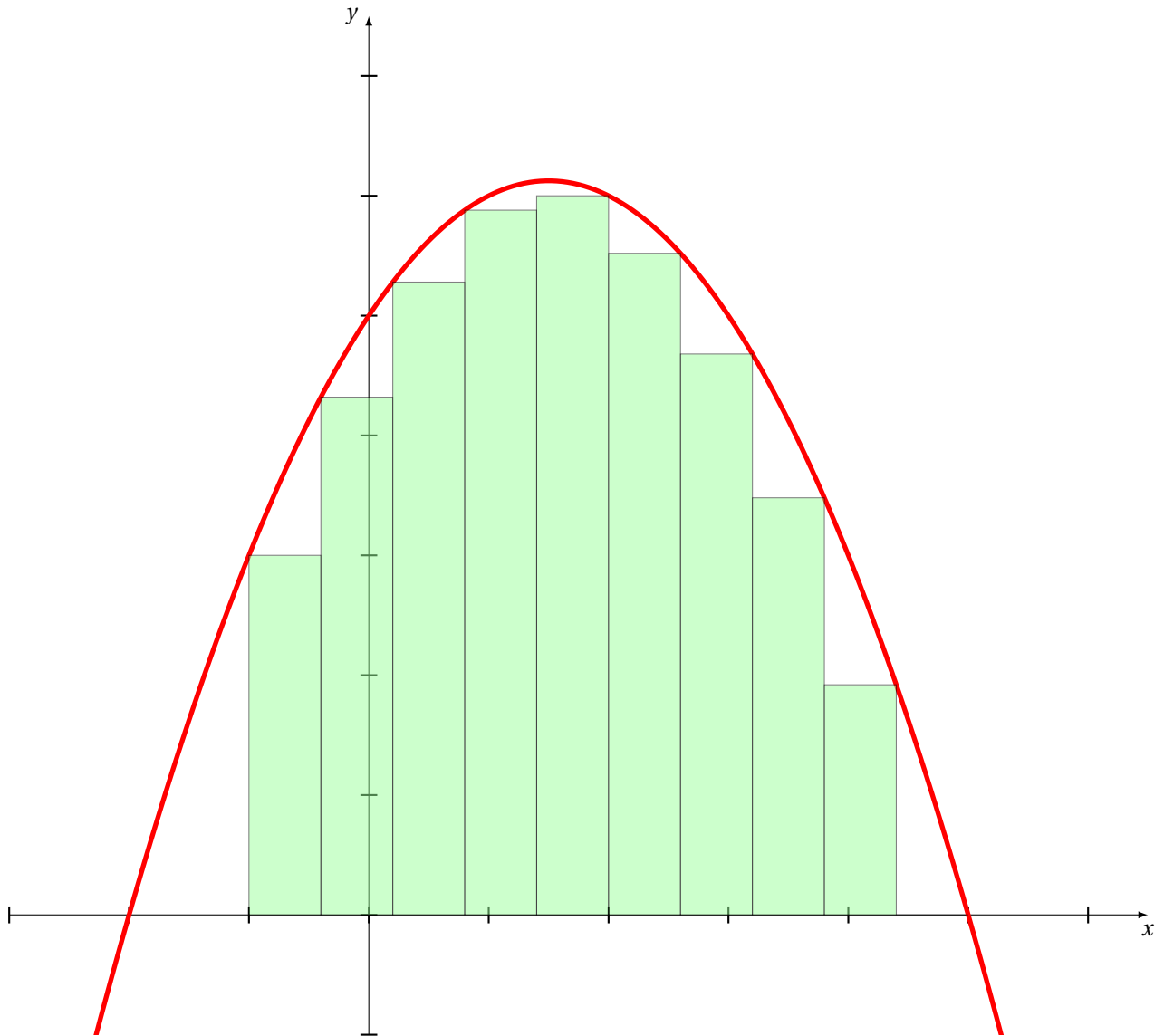


```
\begin{tikzpicture}[scale=3.5]
\tkzInit[xmax=3,ymax=1.75]
\tkzAxeXY
\tkzGrid(0,0)(3,2)
\tkzFct[color = red, domain =1/3:3]{0.125*(3*x-1)+0.375*(3*x-1)/(x*x)}
\tkzDrawRiemannSum[fill=green!40,opacity=.2,color=green,
                    line width=1pt,interval=1./2:exp(1),number=10]
\end{tikzpicture}
```

```
\tkzDrawRiemannSumInf[(local options)]
```

C'est une variante de la macro précédente mais les rectangles sont toujours sous la courbe.

8.2 Somme de Riemann Inf

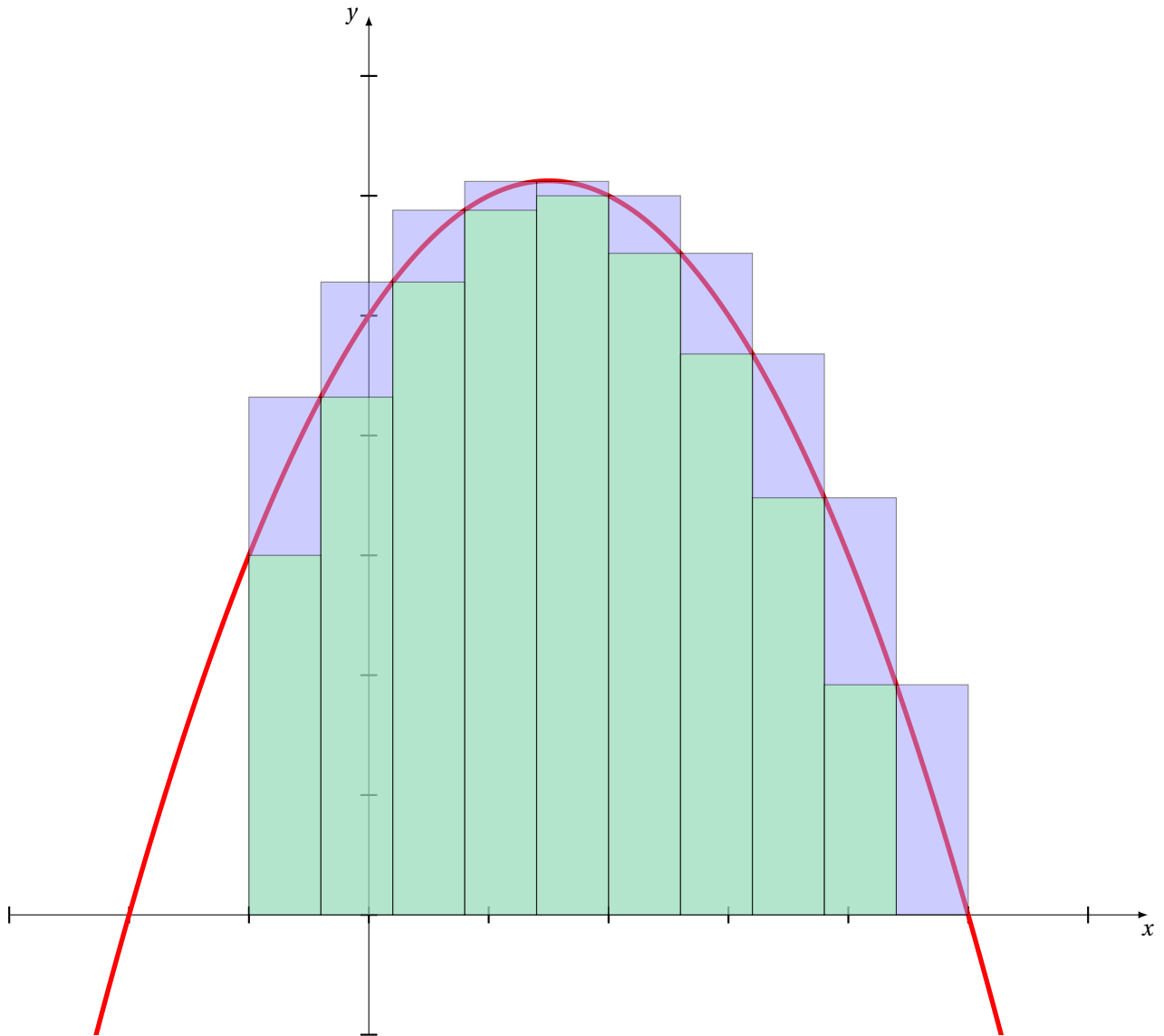


```
\begin{tikzpicture}[scale=1.75]
\tkzInit[xmin=-3,xmax=6,ymin=-2,ymax=14,ystep=2]
\tkzDrawX \tkzDrawY
\tkzFct[line width=2pt,color = red, domain =-3:6]{(-\x-2)*(\x-5)}
\tkzDrawRiemannSumInf[fill=green!40,opacity=.5,interval=-1:5,number=10]
\end{tikzpicture}
```

```
\tkzDrawRiemannSumSup[(local options)]
```

C'est une variante de la macro précédente mais les rectangles sont toujours au-dessus de la courbe.

8.3 Somme de Riemann Inf et Sup

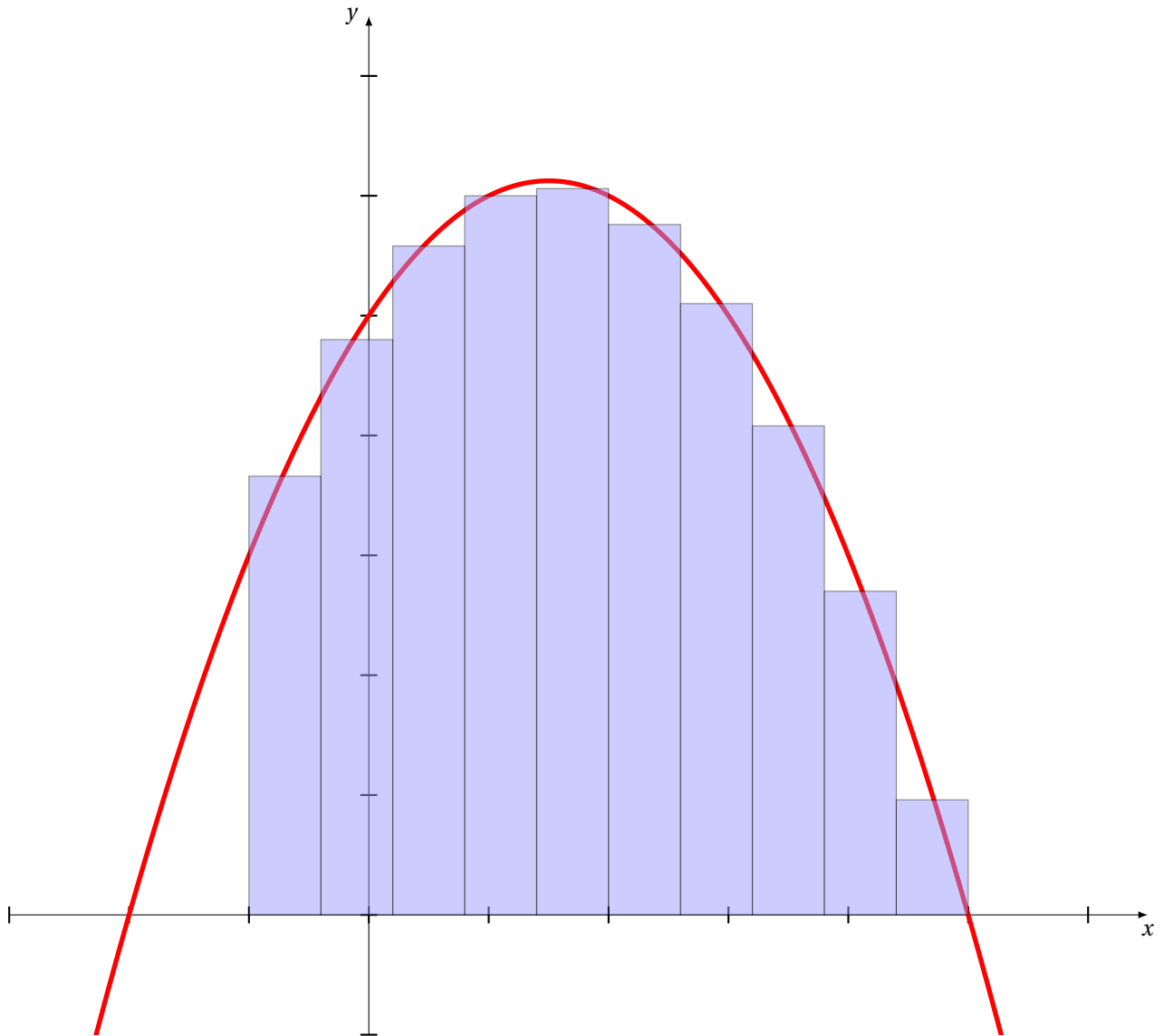


```
\begin{tikzpicture}[scale=1.75]
  \tkzInit[xmin=-3,xmax=6,ymin=-2,ymax=14,ystep=2]
  \tkzDrawX \tkzDrawY
  \tkzFct[line width=2pt,color = red, domain =-3:6]{(-\x-2)*(\x-5)}
  \tkzDrawRiemannSumSup[fill=blue!40,opacity=.5,interval=-1:5,number=10]
  \tkzDrawRiemannSumInf[fill=green!40,opacity=.5,interval=-1:5,number=10]
\end{tikzpicture}
```

```
\tkzDrawRiemannSumMid[(local options)]
```

C'est une variante de la macro précédente mais les rectangles sont à cheval sur la courbe.

8.4 Somme de Riemann Mid



```
\begin{tikzpicture}[scale=1.75]
\tkzInit[xmin=-3,xmax=6,ymin=-2,ymax=14,ystep=2]
\tkzDrawX \tkzDrawY
\tkzFct[line width=2pt,color = red, domain =-3:6]{(-\x-2)*(\x-5)}
\tkzDrawRiemannSumMid[fill=blue!40,opacity=.5,interval=-1:5,number=10]
\end{tikzpicture}
```

9 Droites particulières

9.1 Tracer une ligne verticale

```
\tkzVLine[⟨local options⟩]{⟨decimal number⟩}
```

Attention, la syntaxe est celle de **fp** car on n'utilise pas **gnuplot** pour tracer une droite.

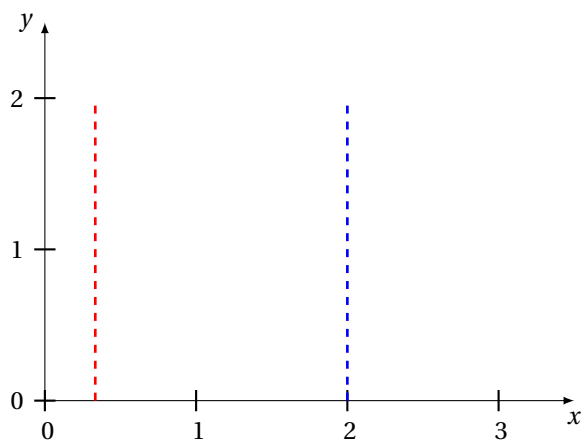
arguments	exemple	définition
decimal number	<code>\tkzVLine{1}</code>	Trace la droite $x = 1$

options	défaut	définition
color	black	couleur du trait
line width	0.6pt	épaisseur du point
style	solid	style du trait

voir les options les lignes dans TikZ

9.2 Ligne verticale

problème avec cette macro, en principe 1./3 devrait être acceptée.



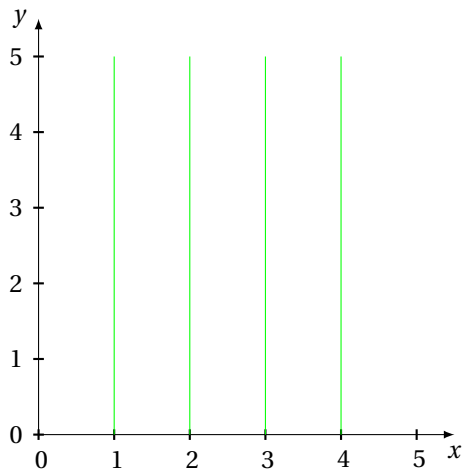
```
\begin{tikzpicture}[scale=2]
  \tkzInit[xmax=3,ymax=2]
  \tkzAxeXY
  \tkzVLine[color      = blue,
             style      = dashed,
             line width = 1pt]{2}
  \tkzVLine[color      = red,
             style      = dashed,
             line width = 1pt]{1./3}
\end{tikzpicture}
```

```
\tkzVLines[⟨local options⟩]{⟨list of values⟩}
```

Attention, la syntaxe est celle de **fp** car on n'utilise pas **gnuplot** pour tracer une droite.

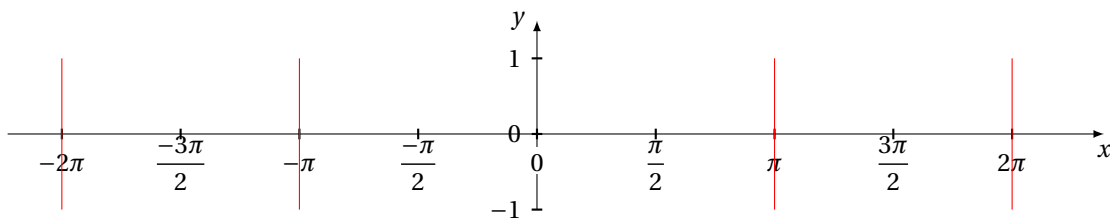
arguments	exemple	définition
list of values	<code>\tkzVLines{1,4}</code>	Trace les droites $x=1$ et $x=4$

9.3 Lignes verticales



```
\begin{tikzpicture}
\tkzInit[xmax=5,ymax=5]
\tkzAxeXY
\tkzVLines[color = green]{1,2,...,4}
\end{tikzpicture}
```

9.4 Ligne verticale et valeur calculée par fp

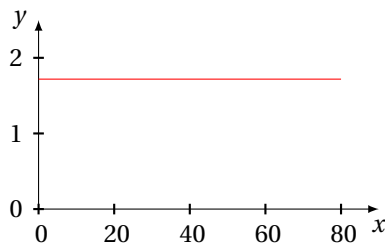


```
\begin{tikzpicture}
\tkzInit[xmin=-7,xmax=7,ymin=-1,ymax=1]
\tkzAxeY
\tkzAxeX[trig=2]
\foreach \v in {-2,-1,1,2}
{\tkzVLine[color=red]{\v*\FPpi}}
\end{tikzpicture}
```

9.5 Une ligne horizontale

```
\tkzHLine[⟨local options⟩]{⟨decimal number⟩}
```

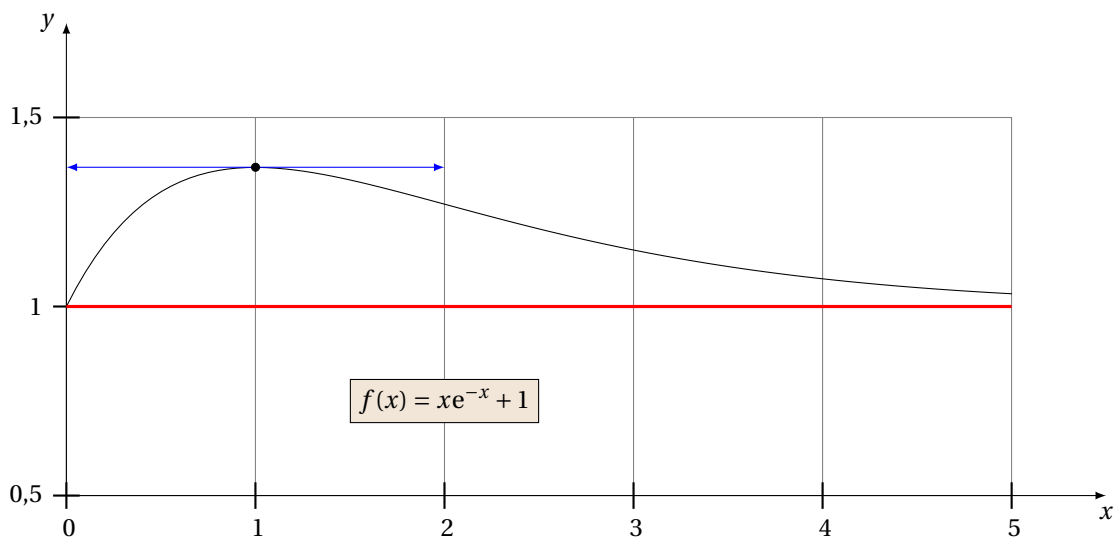
arguments	exemple	définition
decimal number	<code>\tkzVLine{1}</code>	Trace la droite $y = 1$



```
\begin{tikzpicture}
  \tkzInit[xmax=80,xstep=20,ymax=2]
  \tkzAxeXY
  \tkzHLine[color=red]{exp(1)-1}
\end{tikzpicture}
```

9.6 Asymptote horizontale

Attention, une autre méthode consiste à écrire `\tkzFctk` mais si `ystep= n` avec n entier naturel alors il est nécessaire d'écrire k comme un nombre réel, par exemple si `ystep= 3` alors il faut écrire $k = 5.0$.

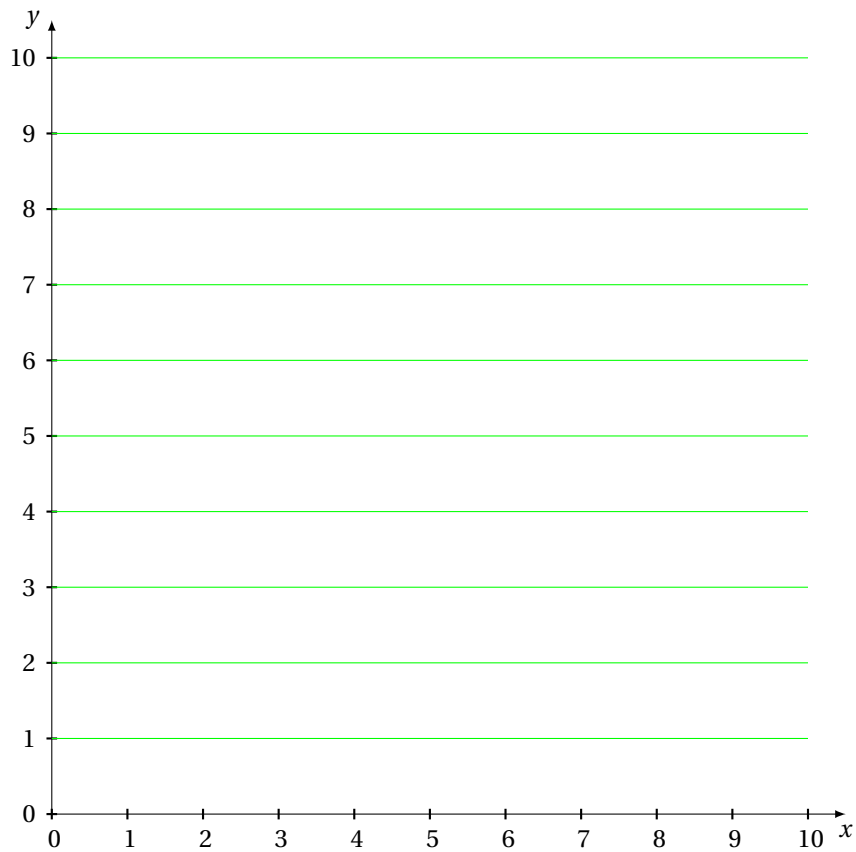


```
\begin{tikzpicture}[scale=2.5]
  \tkzInit[xmax=5,ymin=0.5,ymax=1.5,ystep=0.5]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain = 0:10]{x*exp(-x)+1}
  \tkzHLine[color=red,style=solid,line width=1.2pt]{1}
  \tkzDrawTangentLine[draw,color=blue](1)
  \tkzText[draw,fill = brown!20](2,0.75){$f(x)=x \text{ } \text{e}^{-x}+1$}
\end{tikzpicture}
```

9.7 Lignes horizontales

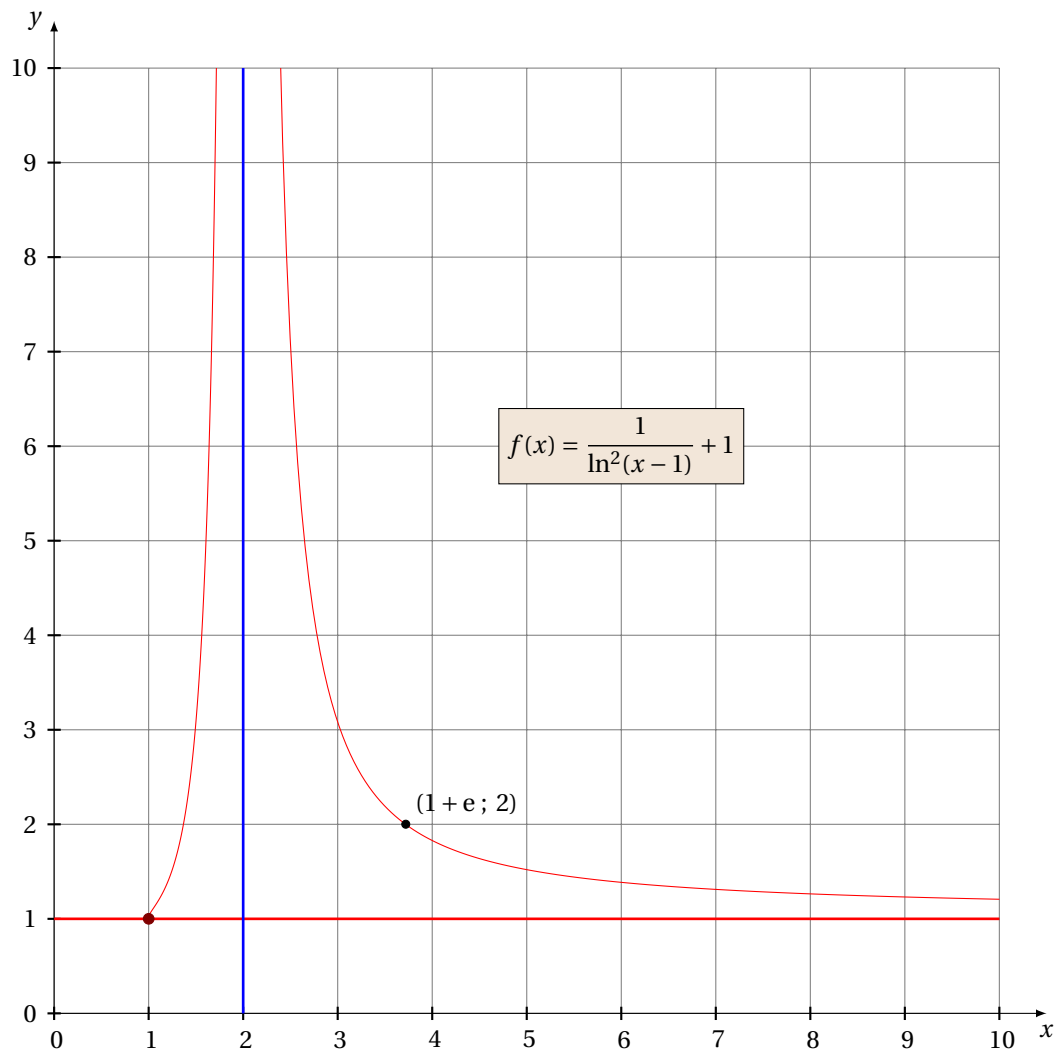
<code>\tkzHLines[⟨local options⟩]{⟨list of values⟩}</code>
--

arguments	exemple	définition
list of values	<code>\tkzHLines{1,4}</code>	Trace les droites $y = 1$ et $y = 4$



```
\begin{tikzpicture}
\tkzInit
\tkzAxeXY
\tkzHLines[color = green]{1,2,...,10}
\end{tikzpicture}
```


9.8 Asymptote horizontale et verticale



```
\begin{tikzpicture}[scale=1.25]
\tkzInit
\tkzGrid
\tkzAxeXY
\tkzFct[color=red,domain=1.001:1.9]{1+1/(log(x-1)**2)}
\tkzFct[color=red,domain = 2.1:10]{1+1/(log(x-1)**2)}
\tkzHLine[line width=1pt,color=red]{1}
\tkzVLine[line width=1pt,color=blue]{2}
\tkzDefPoint(1,1){A}
\tkzDrawPoint[fill=white,color=Maroon,size=4](A)
\tkzDefPointByFct[draw,with=b]({1+exp(1)})
\tkzLabelPoint[above right](tkzPointResult){$(1+\text{e};~2)$}
\tkzText[draw,color = black,fill = brown!20](6,6)%
    {$f(x)=\dfrac{1}{\ln^2(x-1)}+1$}
\end{tikzpicture}
```

10 Courbes avec équations paramétrées

```
\tkzFctPar[⟨local options⟩]{⟨x(t)⟩}{⟨y(t)⟩}
```

$x(t)$ et $y(t)$ sont des expressions utilisant la syntaxe de **gnuplot**. La variable est t .

options	exemple	explication
$x(t), y(t)$	<code>\tkzFctPar[0:1]{t**3}{t**2}</code>	$x(t) = t^3, y(t) = t^2$

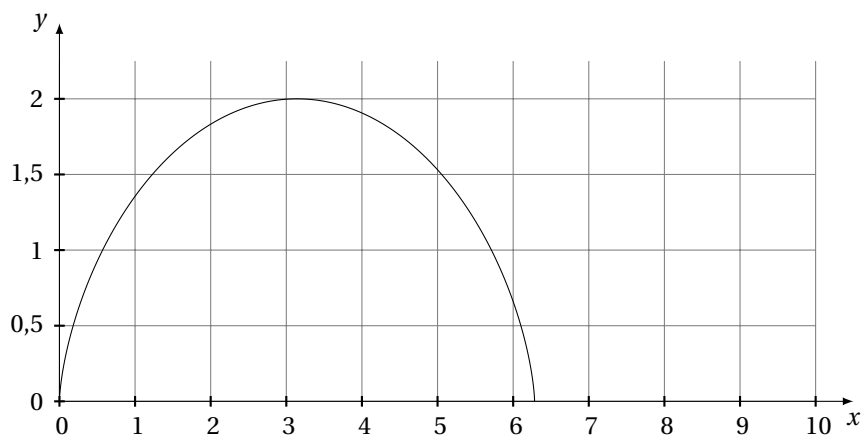
Les options sont celles de TikZ.

options	défaut	définition
domain	-5:5	domaine de la fonction
samples	200	nombre de points utilisés
id	tkzfnc	permet d'identifier les noms des fichiers auxiliaires
color	black	couleur de la ligne
line width	0.4pt	épaisseur de la ligne
style	solid	style de la ligne

10.1 Courbe paramétrée exemple 1

$$x(t) = t - \sin(t)$$

$$y(t) = 1 - \cos(t)$$

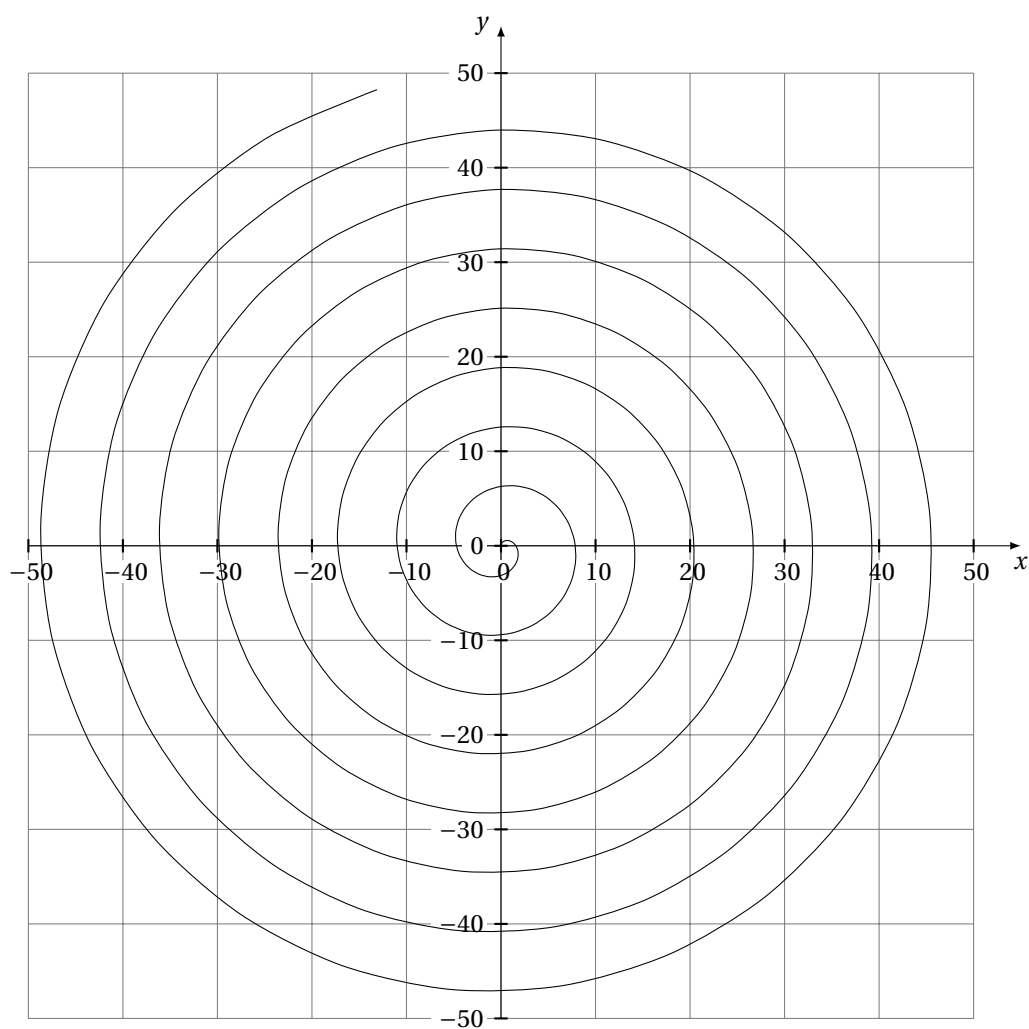


```
\begin{tikzpicture}
\tkzInit[ymax=2.25,ystep=.5] \tkzGrid
\tkzAxeXY
\tkzFctPar[samples=400,domain=0:2*pi]{(t-sin(t))}{(1-cos(t))}
\end{tikzpicture}
```

10.2 Courbe paramétrée exemple 2

$$x(t) = t \times \sin(t)$$

$$y(t) = t \times \cos(t)$$

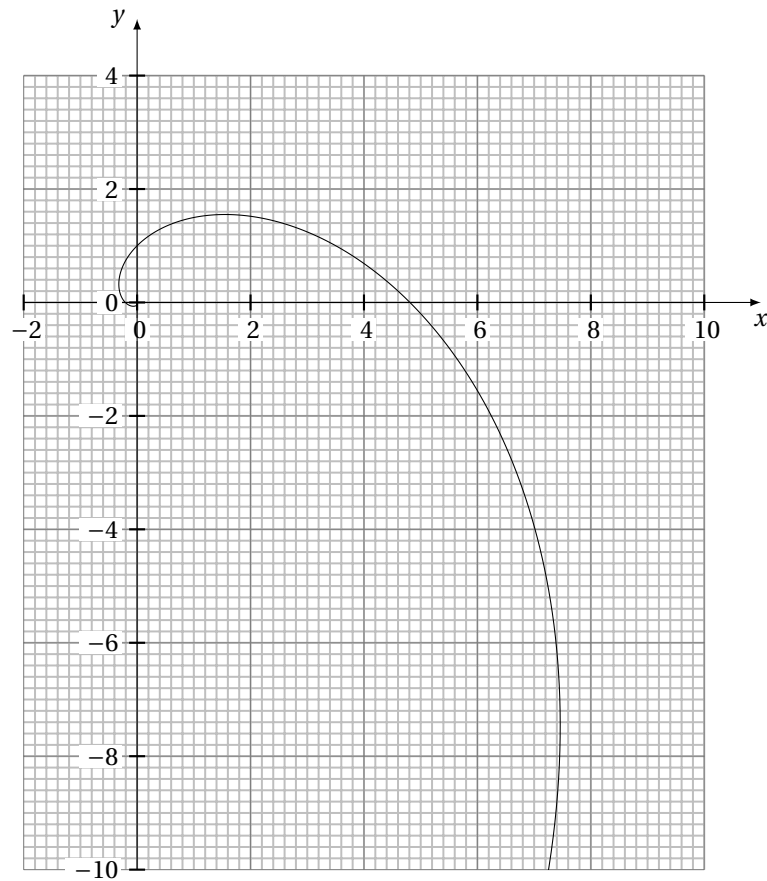


```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-50,xmax=50,xstep=10,
    ymin=-50,ymax=50,ystep=10]
  \tkzGrid
  \tkzAxeXY
  \tkzFctPar[smooth,samples=200,domain=0:50]{t*sin(t)}{t*cos(t)}
\end{tikzpicture}
```

10.3 Courbe paramétrée exemple 3

$$x(t) = \exp(t) \times \sin(t)$$

$$y(t) = \exp(t) \times \cos(t)$$

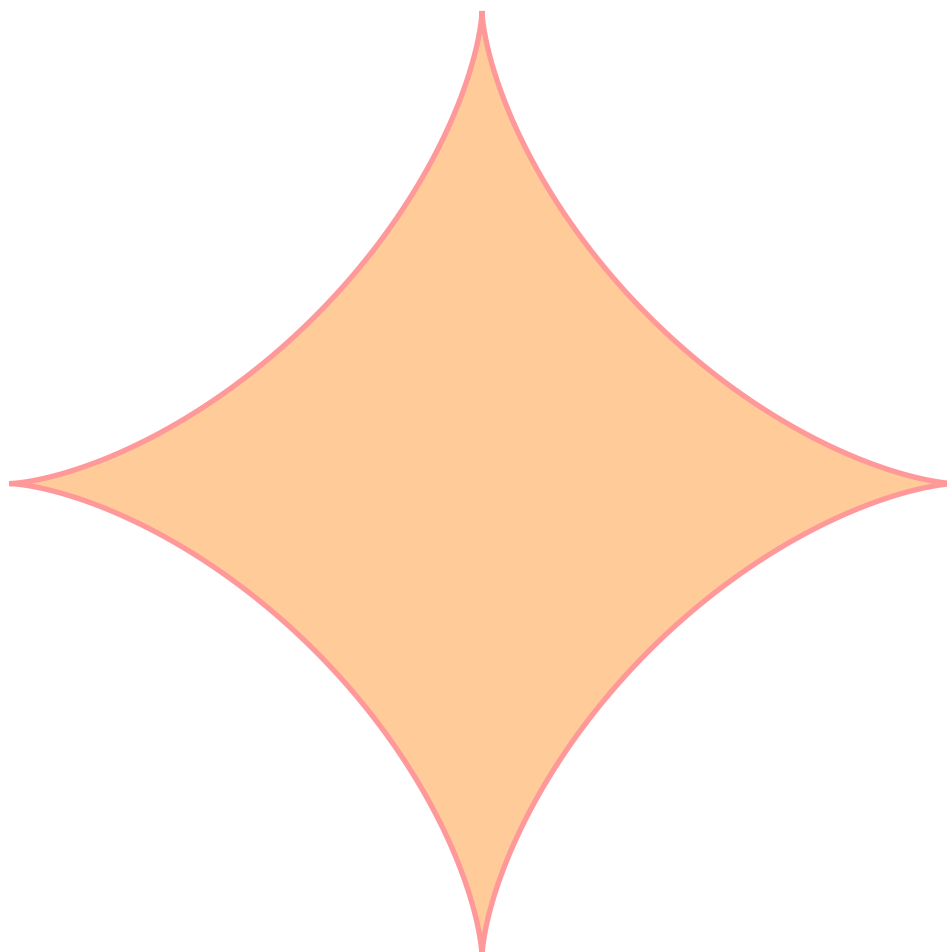


```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmin=-2,xmax=10,xstep=2,ymin=-10,ymax=4,ystep=2]
  \tkzGrid[sub]
  \tkzAxeX[step=2]
  \tkzAxeY[step=2]
  \tkzFctPar[samples=400,domain=-pi:pi]{exp(t)*sin(t)}{exp(t)*cos(t)}
\end{tikzpicture}
```

10.4 Courbe paramétrée exemple 4

$$x(t) = \cos^3(t)$$

$$y(t) = \sin^3(t)$$



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-1,xmax=1,xstep=.2,
           ymin=-1,ymax=1,ystep=.2]
  \tkzFctPar[color=red,
             line width=2pt,
             fill=orange,
             opacity=.4,
             samples=400,
             domain=0:2*pi]{(cos(t))**3}{(sin(t))**3}
\end{tikzpicture}
```

10.5 Courbe paramétrée exemple 5

Saint Valentin version 1

$$x(t) = \sin^3(t)$$
$$y(t) = \cos(t) - \cos^4(t)$$



```
\begin{tikzpicture}[scale=4]
  \tkzInit[xmin=-1,xmax=1,ymin=-2,ymax=1]
  \tkzClip
  \tkzFctPar[samples=500,smooth,domain=-pi:pi,
    ball color=red,shading=ball]%
    {(sin(t))**3}{cos(t)-(cos(t))**4}
\end{tikzpicture}
```

10.6 Courbe paramétrée exemple 6

Saint Valentin version 2 from <http://mathworld.wolfram.com/HeartCurve.html>

$$x(t) = \sin(t) \cos(t) \log(t)$$

$$y(t) = \sqrt{(t) \cos(t)}$$



```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmin=-.4,xmax=.4,xstep=.1,ymin=0,ymax=.7,ystep=.1]
  \tkzClip
  \tkzFctPar[samples=2000,smooth,domain=-1:1,
    ball color=red,shading=ball]%
    {\sin(t)*cos(t)*log(abs(t))}{sqrt(abs(t))*cos(t)}
\end{tikzpicture}
```

10.7 Courbe paramétrée exemple 7

Saint Valentin version 3 from [http://en.wikipedia.org/wiki/Heart_\(symbol\)](http://en.wikipedia.org/wiki/Heart_(symbol))

$$x(t) = 16 \sin^3(t)$$

$$y(t) = 13 \cos(t) - 5 \cos(2t) - 2 \cos(3t) - \cos(4t)$$



```
\begin{tikzpicture}[scale=1.75]
  \tkzInit[xmin=-20,xmax=20,xstep=5,ymin=-25,ymax=15,ystep=5]
  \tkzClip
  \tkzFctPar[samples=400,smooth,domain=0:6.28,
    ball color=red,shading=ball]%
    {16*(sin(t))**3}{13*cos(t)-5*cos(2*t)-2*cos(3*t)-cos(4*t)}
\end{tikzpicture}
```


11 Courbes en coordonnées polaires

```
\tkzFctPolar[⟨local options⟩]{⟨f(t)⟩}
```

$f(t)$ est une expression utilisant la syntaxe de `gnuplot`.

options	exemple	explication
$x(t), y(t)$	<code>\tkzFctPar[0:1]{\t**3}{\t**2}</code>	$x(t) = t^3, y(t) = t^2$

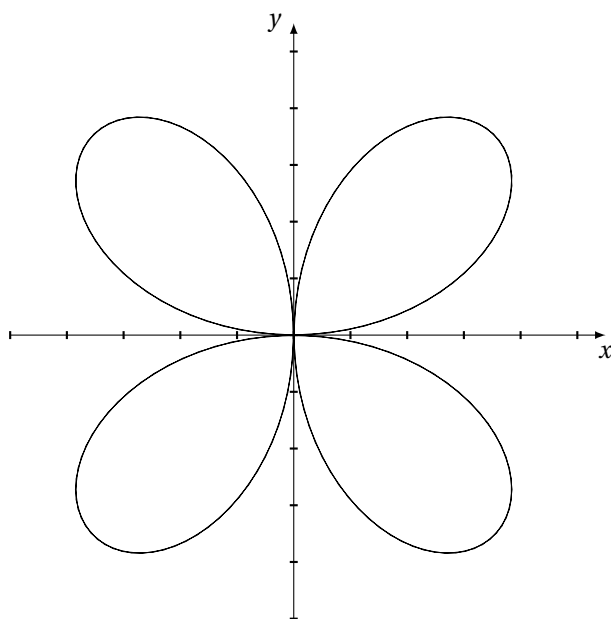
Les options sont celles de TikZ.

options	défaut	définition
domain	<code>0:2*pi</code>	domaine de la fonction
samples	<code>200</code>	nombre de points utilisés
id	<code>tkzfnc</code>	permet d'identifier les noms des fichiers auxiliaires
color	<code>black</code>	couleur de la ligne
line width	<code>0.4pt</code>	épaisseur de la ligne
style	<code>solid</code>	style de la ligne

`gnuplot` définit π avec `pi` et `fp.sty` avec `\FPpi`. Les valeurs qui déterminent le domaine sont évaluées par `fp.sty`. Il est possible d'utiliser soit `pi`, soit `\FPpi`.

11.1 Équation polaire exemple 1

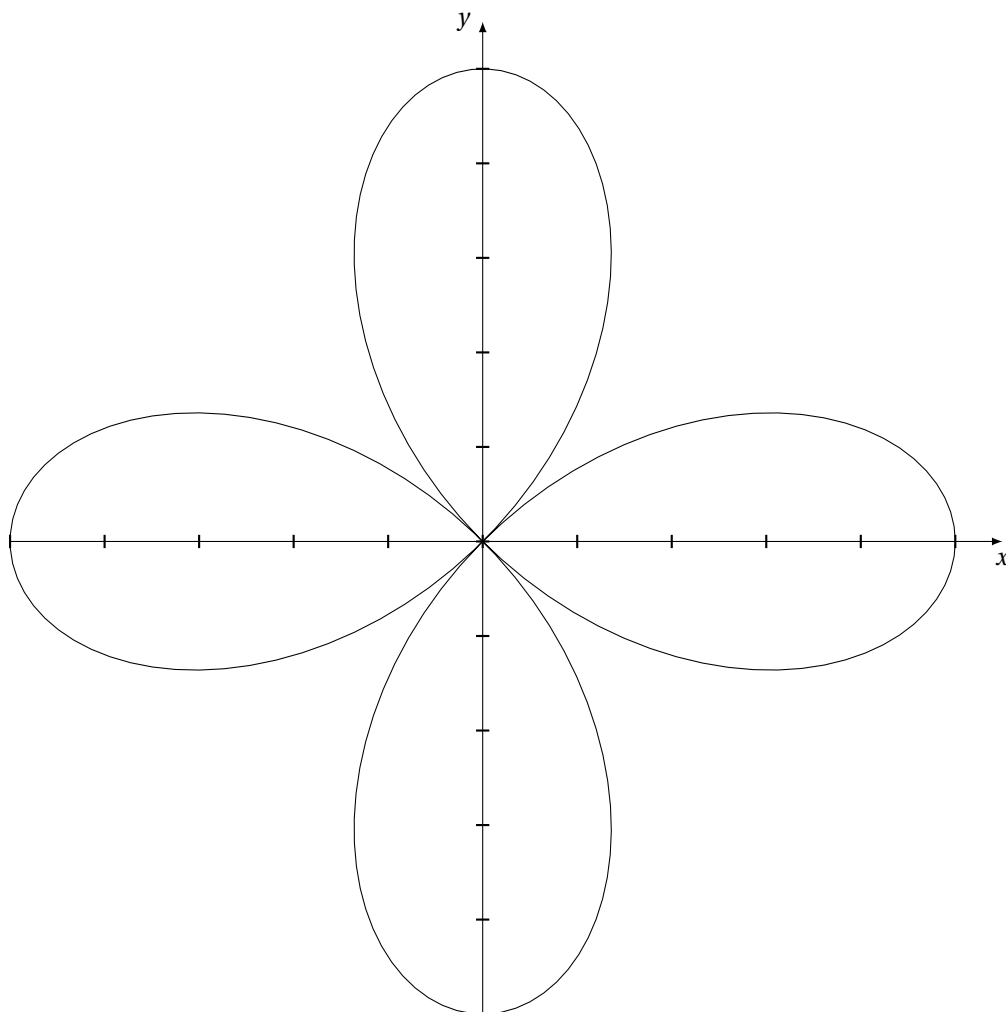
$$\rho(t) = \cos(t) * \sin(t)$$



```
\begin{tikzpicture}[scale=0.75]
\tkzInit [xmin=-0.5,xmax=0.5,
          ymin=-0.5,ymax=0.5,
          xstep=0.1,ystep=.1]
\tkzDrawX \tkzDrawY
\tkzFctPolar[domain=-2*pi:2*pi]{cos(t)*sin(t)}
\end{tikzpicture}
```

11.2 Équation polaire exemple 2

$$\rho(t) = \cos(2 * t)$$



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit [xmin=-1,xmax=1,
            ymin=-1,ymax=1,
            xstep=.2,ystep=.2]
  \tkzDrawX   \tkzDrawY
  \tkzFctPolar[domain=0:2*pi]{cos(2*t)}
\end{tikzpicture}
```

11.3 Équation polaire Heart

From Mathworld : <http://mathworld.wolfram.com/HeartCurve.html>

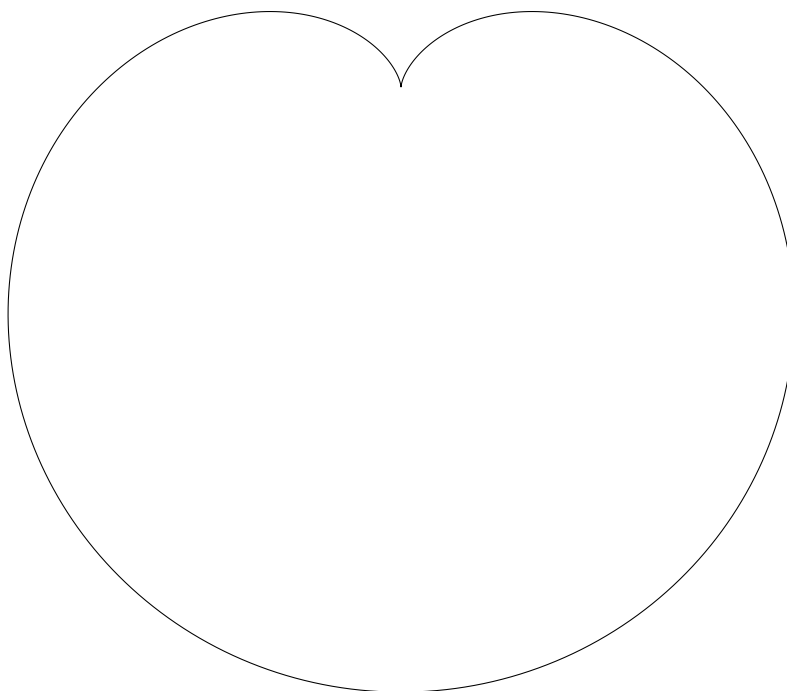
$$\rho(t) = 2 - 2 * \sin(t) + \sin(t) * \sqrt{(\cos(t)) / (\sin(t) + 1.4)}$$



```
\begin{tikzpicture}[scale=3]
\tkzInit[xmin=-5,xmax=5,ymin=-5,ymax=5]
\tkzFctPolar[domain      = -pi:pi,
              samples     = 800,
              ball color   = red,
              shading      = ball]%
  {2-2*sin(t)+sin(t)*sqrt(abs(cos(t)))/(sin(t)+1.4)}
\end{tikzpicture}
```

11.4 Équation polaire exemple 4

$$\rho(t) = 1 - \sin(t)$$

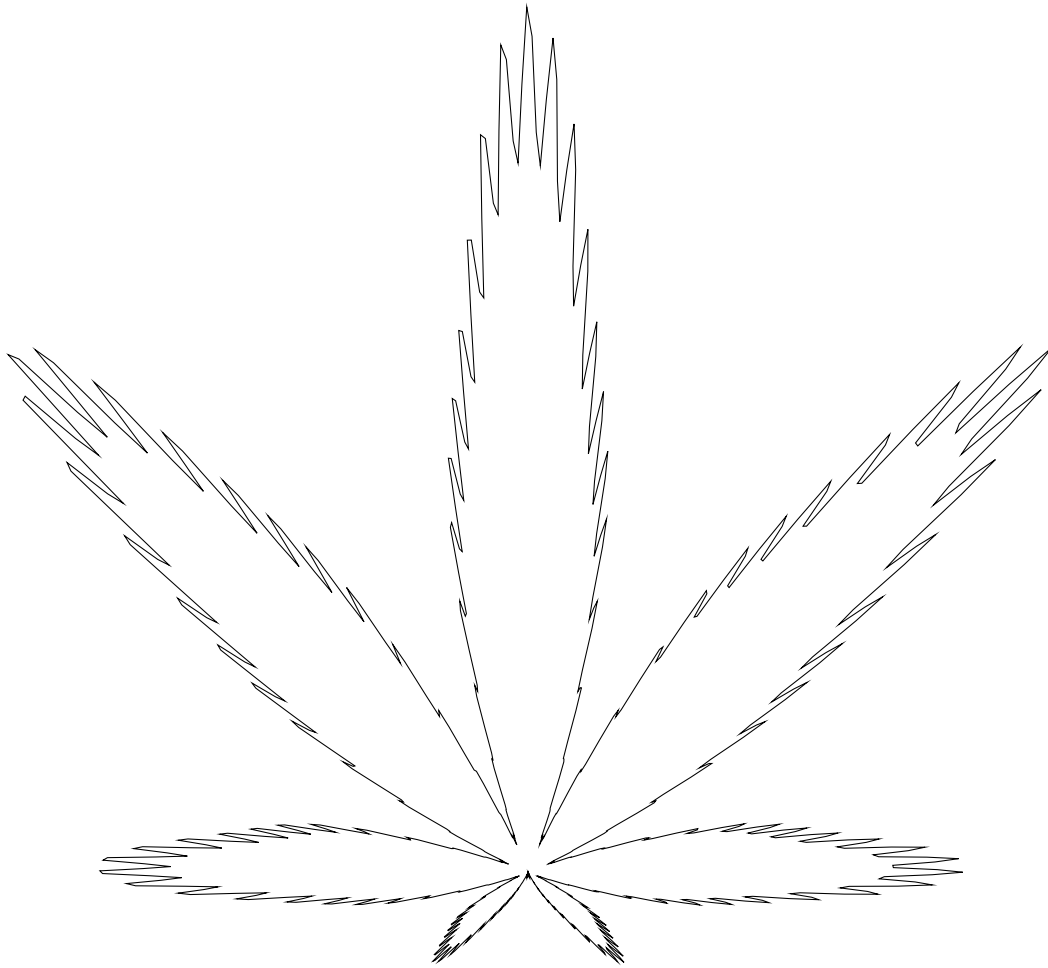


```
\begin{tikzpicture}[scale=4]
\tkzInit [xmin=-5,xmax=5,ymin=-5,ymax=5,xstep=1,ystep=1]
\tkzFctPolar[domain=0:2*pi,samples=400]{ 1-sin(t) }
\end{tikzpicture}
```

11.5 Équation polaire Cannabis ou Marijuana Curve

Cannabis curve from mathworld : <http://mathworld.wolfram.com/CannabisCurve.html>

$$\rho(t) = (1 + .9 * \cos(8 * t)) * (1 + .1 * \cos(24 * t)) * (1 + .1 * \cos(200 * t)) * (1 + \sin(t))$$

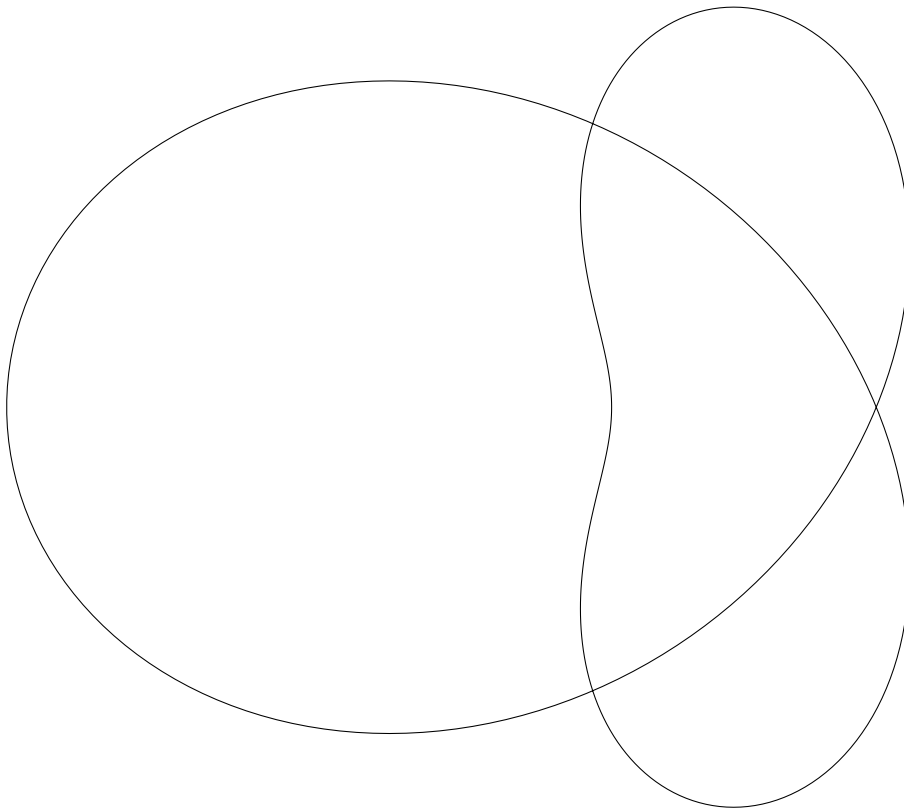


```
\begin{tikzpicture}[scale=2.5]
  \tkzInit [xmin=-5,xmax=5,ymin=-5,ymax=5,xstep=1,ystep=1]
  \tkzFctPolar[domain=0:2*pi,samples=1000]%
  { (1+.9*cos(8*t))*(1+.1*cos(24*t))*(1+.1*cos(200*t))*(1+sin(t)) }
\end{tikzpicture}
```

11.6 Scarabaeus Curve

From mathworld : <http://mathworld.wolfram.com/Scarabaeus.html>

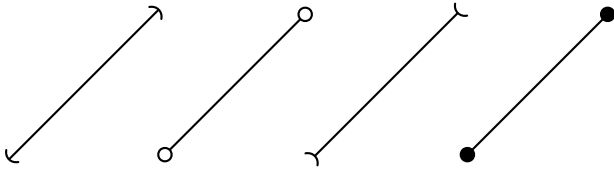
$$\rho(t) = 1.6 * \cos(2 * t) - 3 * \cos(t)$$



```
\begin{tikzpicture}[scale=2.5]
\tkzInit [xmin=-5,xmax=5,ymin=-5,ymax=5,xstep=1,ystep=1]
\tkzFctPolar[domain=0:2*pi,samples=400]{1.6*cos(2*t)-3*cos(t) }
\end{tikzpicture}
```

12 Symboles

Certains ajoutent aux courbes des symboles afin de donner des indications supplémentaires au lecteur. Voici quelques exemples possibles :



L'exemple suivant est de Simon Schläpfer :

On veut tracer

$$y = \begin{cases} 8 - 1.5x & , \text{if } x < 2 \\ 4 & , \text{if } 2 \leq x \leq 3 \\ 2x - 4 & , \text{if } x > 3 \end{cases}$$

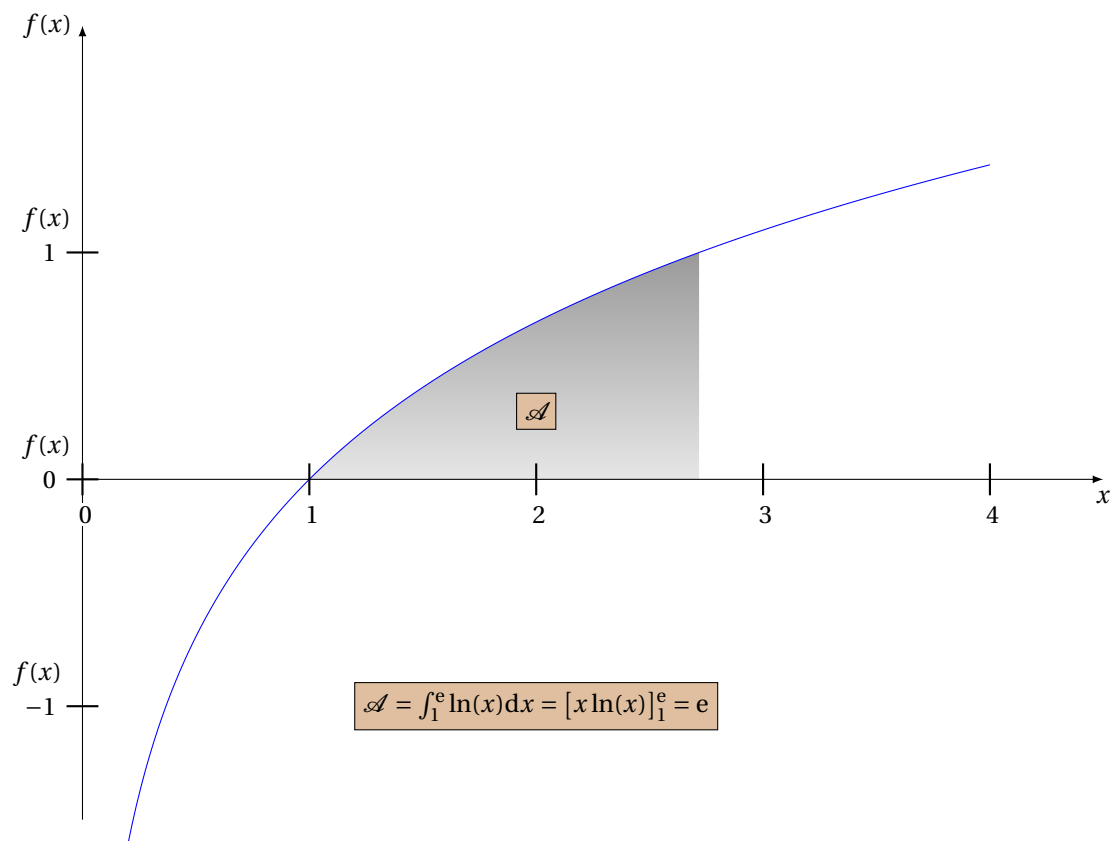


```
\begin{tikzpicture}
  \tkzInit[xmin=-1,xmax=6,ymin=0,ymax=10,xstep=1,ystep=1]
  \tkzGrid[color=gray]
  \tkzAxeXY
  \tkzFct[{-[]},color=red,domain =-1:2,samples=2]{8-1.5*\x}
  \tkzFct[{[-]},color=blue,domain =2:3,samples=2]{4}
  \tkzFct[{[]-},color=green!50!black,domain =3:6,samples=2]{2*\x-4}
\end{tikzpicture}
```

13 Quelques exemples

13.1 Variante intermédiaire : TikZ + tkz-fct

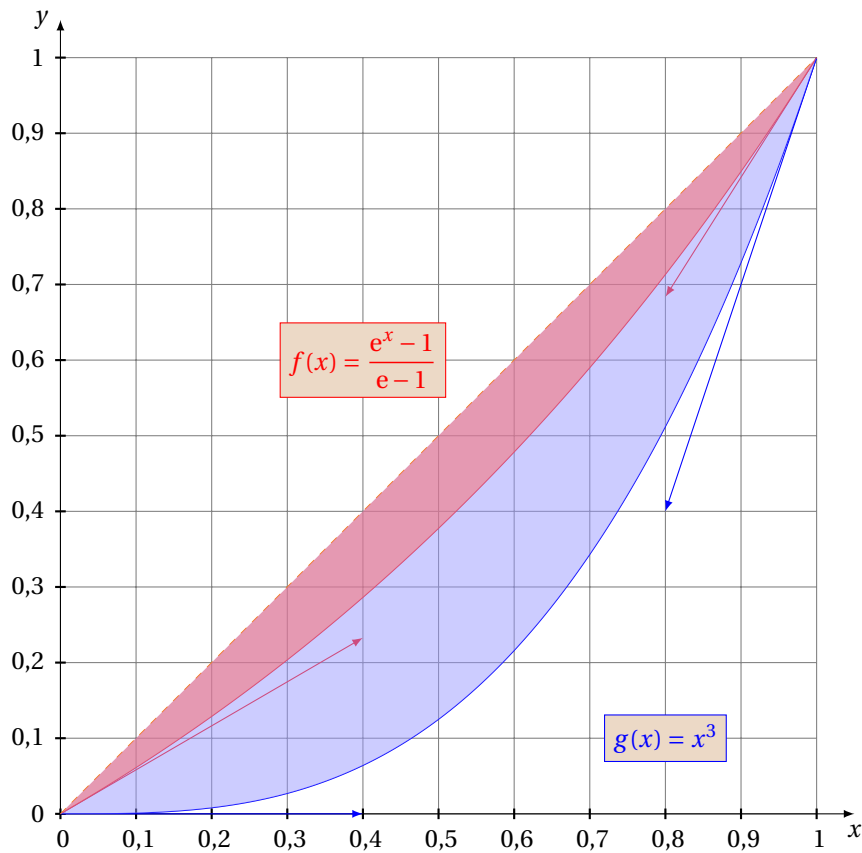
Les codes de TikZ et de `tkz-fct` peuvent se compléter. Ainsi les axes et les textes sont gérés par `tkz-fct` mais la courbe est laissée à TikZ et `gnuplot`.



```
\begin{tikzpicture}[scale=3]
\tkzInit[xmin=0,xmax=4,ymin=-1.5,ymax=1.5]
\tkzAxeY[label=$f(x)$]
\tkzDefPoint(1,0){x} \tkzDrawPoint[color=blue,size=0.6pt](x)
\shade[top color=gray!80,bottom color=gray!20] (1,0)%
    plot[id=ln,domain=1:2.718] function{log(x)} |- (1,0);
\draw[color=blue] plot[id=ln,domain=0.2:4,samples=200]function{log(x)};
\tkzAxeX
\tkzText[draw,color= black,fill=brown!50](2,-1)%
    {${\mathcal A} = \int_1^{\text{e}} \ln(x) \text{d}x = %
    \big[x \ln(x) \big]_1^{\text{e}} = \text{e} $}
\tkzText[draw,color= black,fill=brown!50](2,0.3){${\mathcal A}$}
\end{tikzpicture}
```


13.2 Courbes de Lorentz

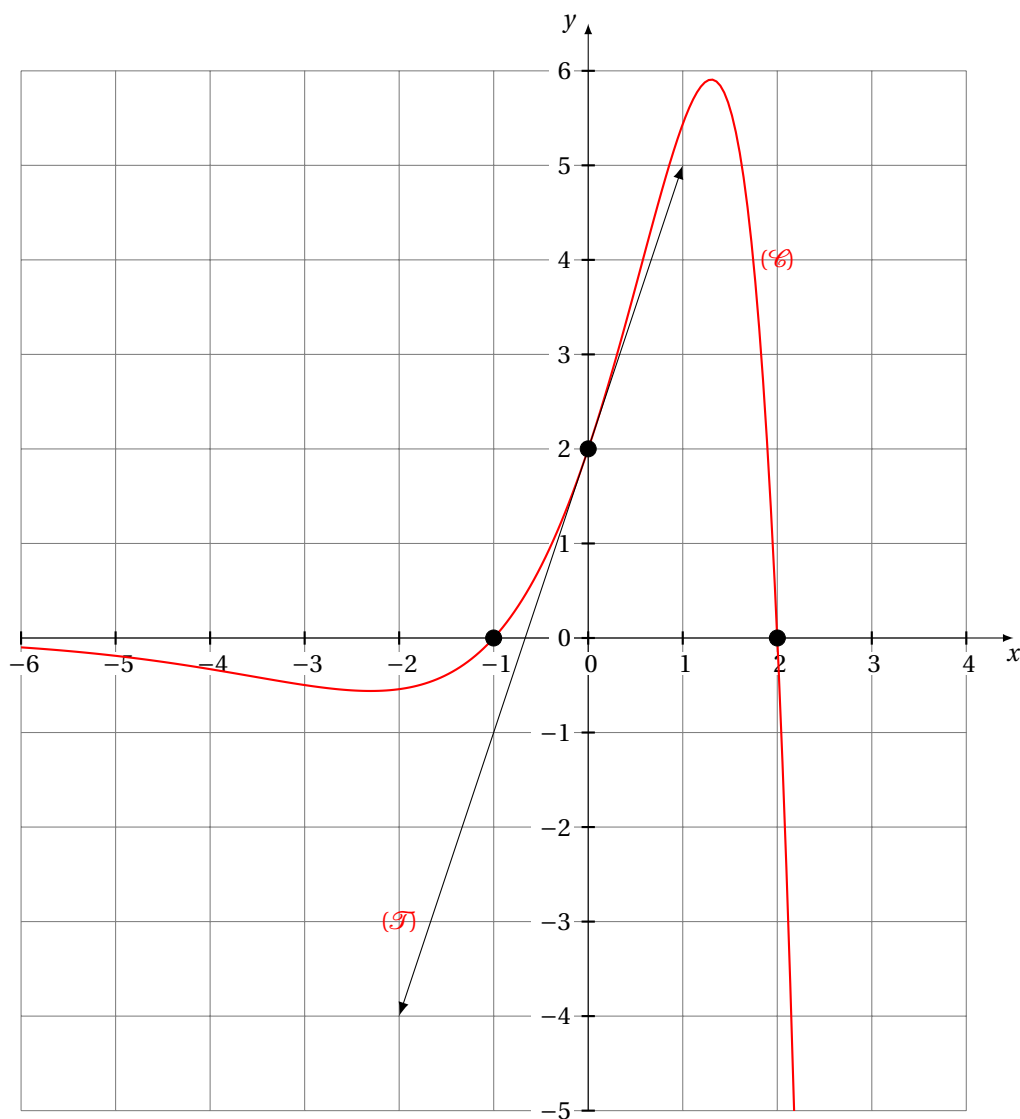
$$f(x) = \frac{e^x - 1}{e - 1} \text{ et } g(x) = x^3$$



```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=1,ymax=1,xstep=0.1,ystep=0.1]
  \tkzGrid(0,0)(1,1)
  \tkzAxeXY
  \tkzFct[color = red,domain = 0:1]{(exp(\x)-1)/(exp(1)-1)}
  \tkzDrawTangentLine[kl=0,kr=0.4,color=red](0)
  \tkzDrawTangentLine[kl=0.2,kr=0,color=red](1)
  \tkzText[draw,color = red,fill = brown!30](0.4,0.6)%
    {\$f(x)=\dfrac{\text{e}^x-1}{\text{e}-1}\$}
  \tkzFct[color = blue,domain = 0:1]{\x*\x*\x}
  \tkzDrawTangentLine[kl=0,kr=0.4,color=blue](0)
  \tkzDrawTangentLine[kl=0.2,kr=0,color=blue](1)
  \tkzText[draw,color = blue,fill = brown!30](0.8,0.1){\$g(x)=x^3\$}
  \tkzFct[color = orange,style = dashed,domain = 0:1]{\x}
  \tkzDrawAreaafg[between=c and b,color=blue!40,domain = 0:1]
  \tkzDrawAreaafg[between=c and a,color=red!60,domain = 0:1]
\end{tikzpicture}
```

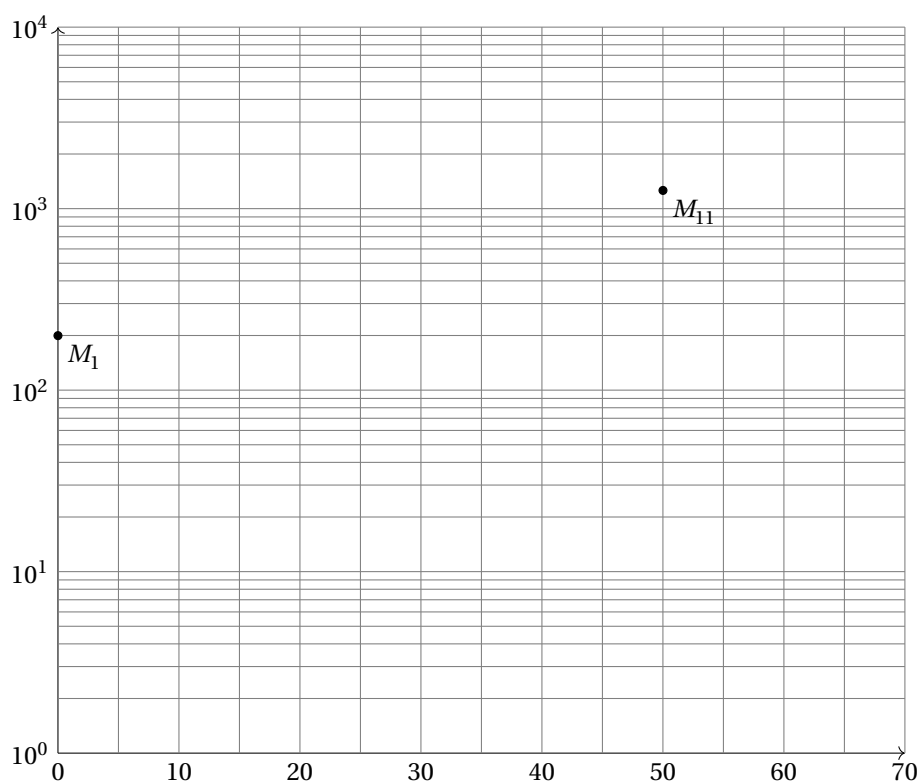
13.3 Courbe exponentielle

$$f(x) = (-x^2 + x + 2) \exp(x)$$



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-6,xmax=4,ymin=-5,ymax=6]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[color=red,thick,domain=-6:2.1785]{(-x*x+x+2)*exp(x)}
  \tkzSetUpPoint[size=6]
  \tkzDrawTangentLine[draw,kl=2](0)
  \tkzDefPoint(2,0){b} \tkzDrawPoint(b)
  \tkzDefPoint(-1,0){c} \tkzDrawPoint(c)
  \tkzText(2,4){(\mathcal{C})}
  \tkzText(-2,-3){(\mathcal{T})}
\end{tikzpicture}
```

13.4 Axe logarithmique

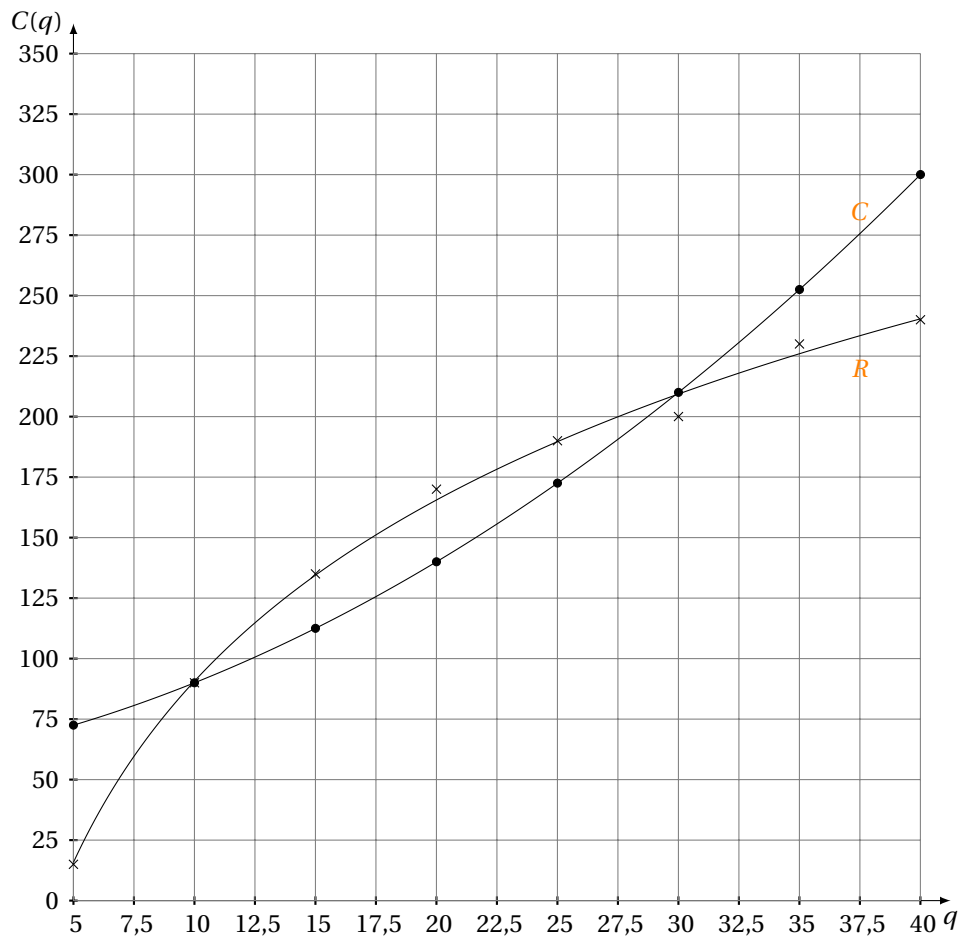


```

\begin{tikzpicture}[scale=0.8]
\tkzInit[xmax=14,ymax=12]
\draw[thin,->] (0,0) -- (14,0) node[below left] {};
\draw[thin,->] (0,0) -- (0,12) node[below left] {};
\foreach \x/\xtext in {0/0,2/10,4/20,6/30,8/40,10/50,12/60,14/70}%
  {\draw[shift={(0,\x)}] node[below] {\xtext$ };}
\foreach \y/\z in {0/0,3/1,6/2,9/3,12/4}%
  {\draw[shift={(0,\y)}] node[left] {$10^{\z}$};}
\foreach \x in {1,2,...,14}{\tkzVLine[gray,thin]{\x}}
\foreach \y in {3,6,...,12}{\tkzHLine[gray,thin]{\y}}
\foreach \y in {0,3,...,9}{
\foreach \z in {0.903,1.431,1.806,2.097,2.334,2.535,2.709,2.863}%
  {\tkzHLine[thin,gray,shift={(0,\y)}] {\z}}}
\tkzDefPoint(0,6.90){a}
\tkzDefPoint(10,9.30){b}
\tkzDrawPoints(a,b)
\tkzLabelPoint(a){$M_{1}$}
\tkzLabelPoint(b){$M_{11}$}
\end{tikzpicture}

```

13.5 Un peu de tout



```
\begin{tikzpicture}[scale=.8]
\tkzInit[xmin=5,xmax=40,ymin=0,ymax=350,xstep=2.5,ystep=25]
\tkzDrawX[label=$q$]
\tkzDrawY[label=$C(q)$]
\tkzLabelXY
\tkzGrid[orange]
\tkzFct[domain=5:40]{0.1*\x**2+2*\x+60}
\foreach \vv in {5,10,...,40}{%
\tkzDefPointByFct(\vv)
\tkzDrawPoint(tkzPointResult)}
\tkzFct[domain=5:40]{(108*log(\x)-158)}
\tkzText(37.5,285){$C$}
\tkzText(37.5,220){$R$}
\tkzDefSetOfPoints{%
5/15,10/90,15/135,20/170,25/190,30/210,35/230,40/240}
\tkzDrawSetOfPoints[mark = x,mark size=3pt]
\end{tikzpicture}
```

13.6 Interpolation

Il s'agit ici de trouver un polynôme d'interpolation sur l'intervalle $[-1 ; 1]$ de la fonction f définie par :

$$f(x) = \frac{1}{1+8x^2}$$

Le polynôme d'interpolation est celui obtenu par la méthode de **Lagrange** :

$$\begin{aligned} P(x) = & 1.000000000 - 0.0000000072x - 7.991424876x^2 + 0.000001079x^3 + 62.60245358x^4 \\ & - 0.00004253x^5 - 444.2347594x^6 + 0.0007118x^7 + 2516.046396x^8 - 0.005795x^9 \\ & - 10240.01777x^{10} + 0.025404x^{11} + 28118.29594x^{12} - 0.05934x^{13} - 49850.83249x^{14} \\ & + 0.08097x^{15} + 54061.87086x^{16} - 0.055620x^{17} - 32356.67279x^{18} + 0.015440x^{19} \\ & + 8140.046421x^{20} \end{aligned}$$

Ayant utilisé vingt et un points, le polynôme est de degré 20. Celui-ci est écrit en utilisant la méthode de **Horner**. Dans un premier temps, on demande à gnuplot de tracer la courbe de f en rouge, enfin on trace le polynôme d'interpolation en bleu. Les points utilisés sont en jaune.

13.6.1 Le code

```
\begin{tikzpicture}
\tkzInit[xmin=-1,xmax=1,ymin=-1.8,ymax=1.2,xstep=0.1,ystep=0.2]
\tkzGrid
\tkzAxeXY
\tkzFct[samples = 400, line width=4pt, color = red,opacity=.5](-1---1){1/(1+8*\x*\x)}
\tkzFct[smooth,samples = 400, line width=1pt, color = blue,domain =-1:1]%
{1.0+((((((((((((((((((((
      8140.04642)*\x
      +0.01544)*\x
      -32356.67279)*\x
      -0.05562)*\x
      +54061.87086)*\x
      +0.08097)*\x
      -49850.83249)*\x
      -0.05934)*\x
      +28118.29594)*\x
      +0.02540)*\x
      -10240.01777)*\x
      -0.00580)*\x
      +2516.04640)*\x
      +0.00071)*\x
      -444.23476)*\x
      -0.00004)*\x
      +62.60245)*\x
      +0.00000)*\x
      -7.99142)*\x
      -0.00000)*\x}
\tkzSetUpPoint[size=16,color=black,fill=yellow]
\foreach \v in {-1,-0.8,---,1}{\tkzDefPointByFct[draw](\v)}
\end{tikzpicture}
```

Le résultat est sur la page suivante où on peut constater le phénomène de **Runge**.

13.6.2 la figure

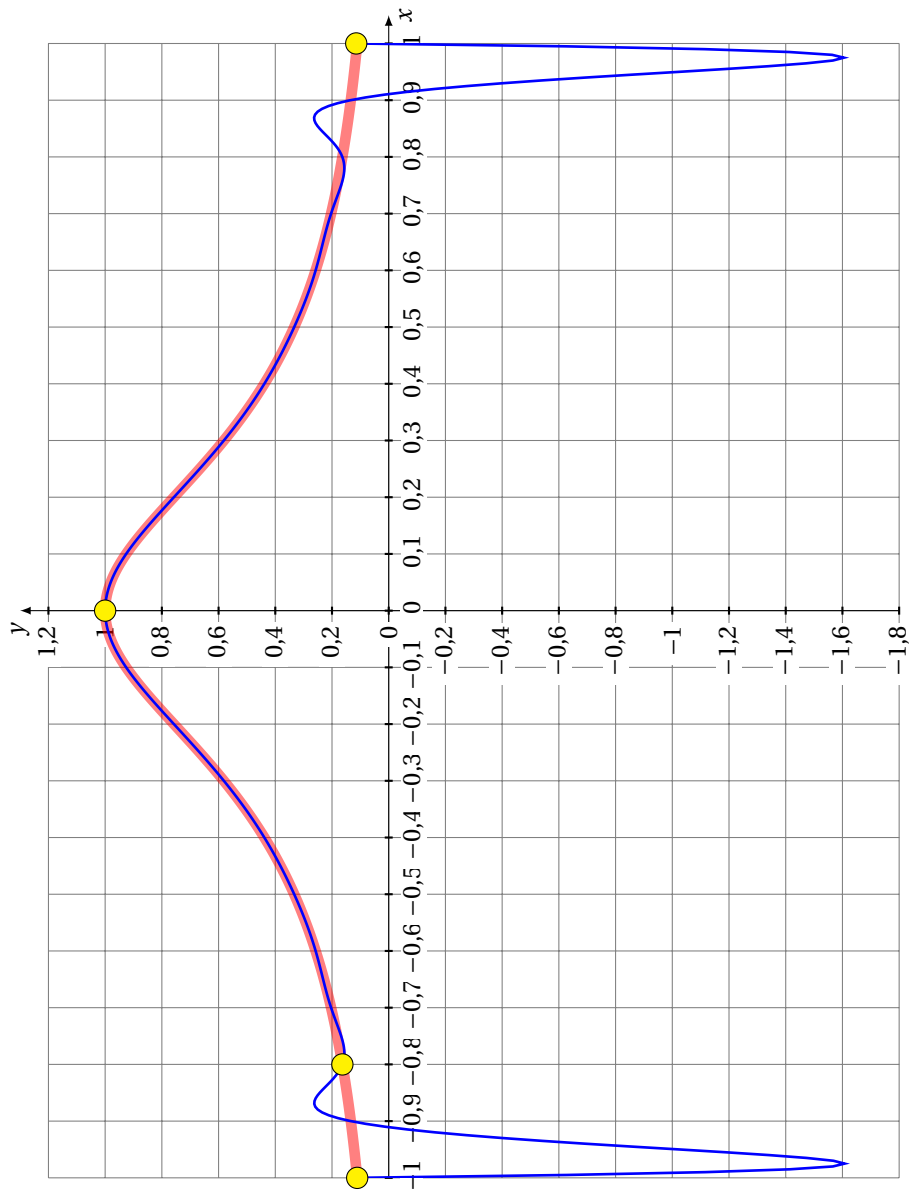


FIGURE 1 – Interpolation : $\frac{1}{1+8x^2}$

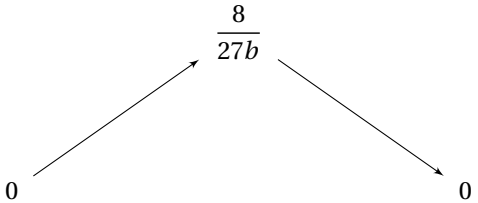
13.7 Courbes de Van der Waals

Soient v le volume d'une masse fluide et p sa pression. b et k sont deux nombres réels strictement positifs. On souhaite étudier une formule exprimant la dépendance de ces variables proposée par Van der Waals.

$$p(v) = \frac{-3}{v^2} + \frac{3k}{v-b}$$

définie sur l'intervalle $I =]b; +\infty]$

13.7.1 Tableau de variations

v	b		$3b$	$+\infty$
$g'(v)$	0	+	0	-
$g(v)$	0			

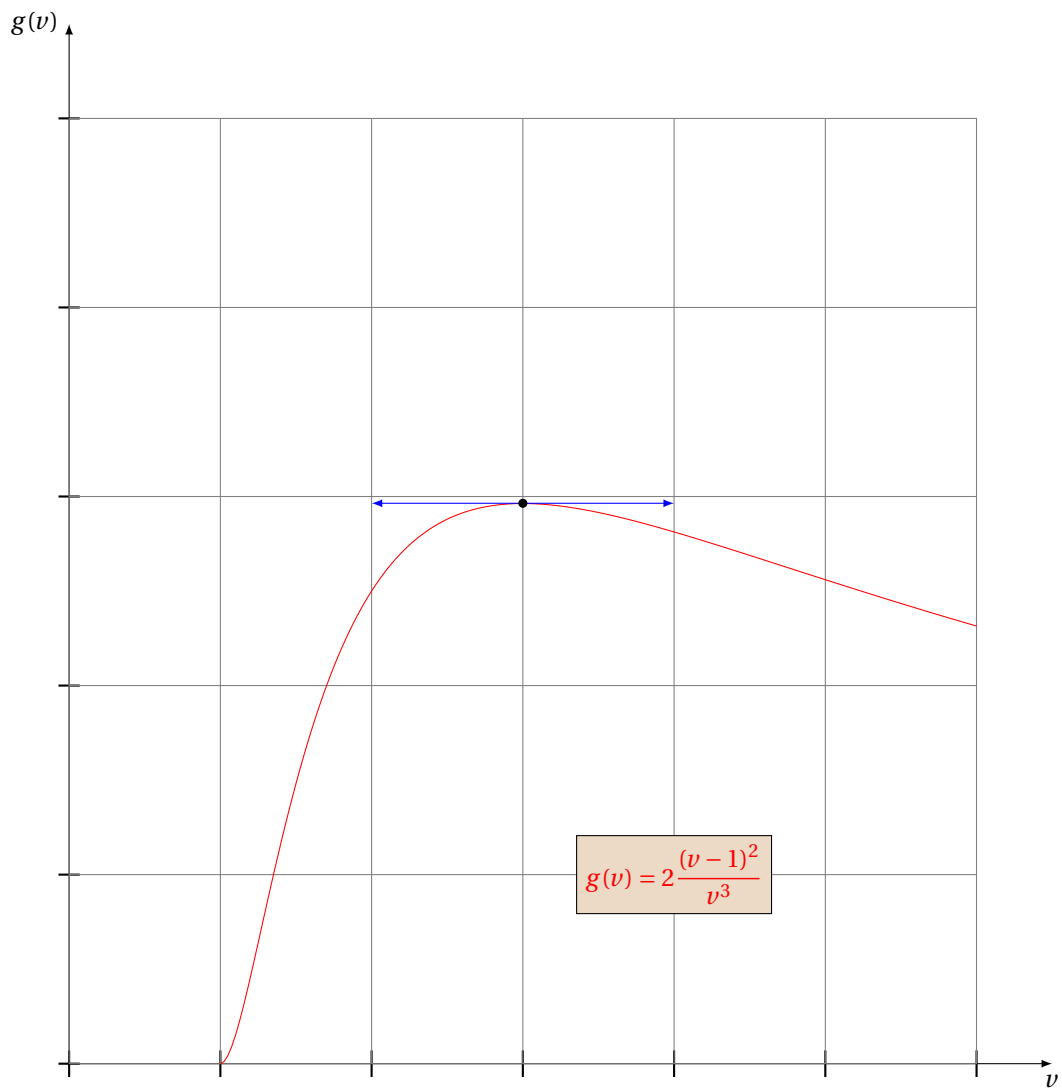
```

\begin{tikzpicture}
\tkzTab%
{ $v$ /1,%
  $g'(v)$ /1,%
  $g(v)$ /3%
}%
{ $b$ ,%
  $3b$ ,%
  $+\infty$%
}%
{0,$+,$,$0$,$-$,t}
{-/ $0$ /,%
+/$\dfrac{8}{27b}$ /,%
-/$0$ /}%
\end{tikzpicture}

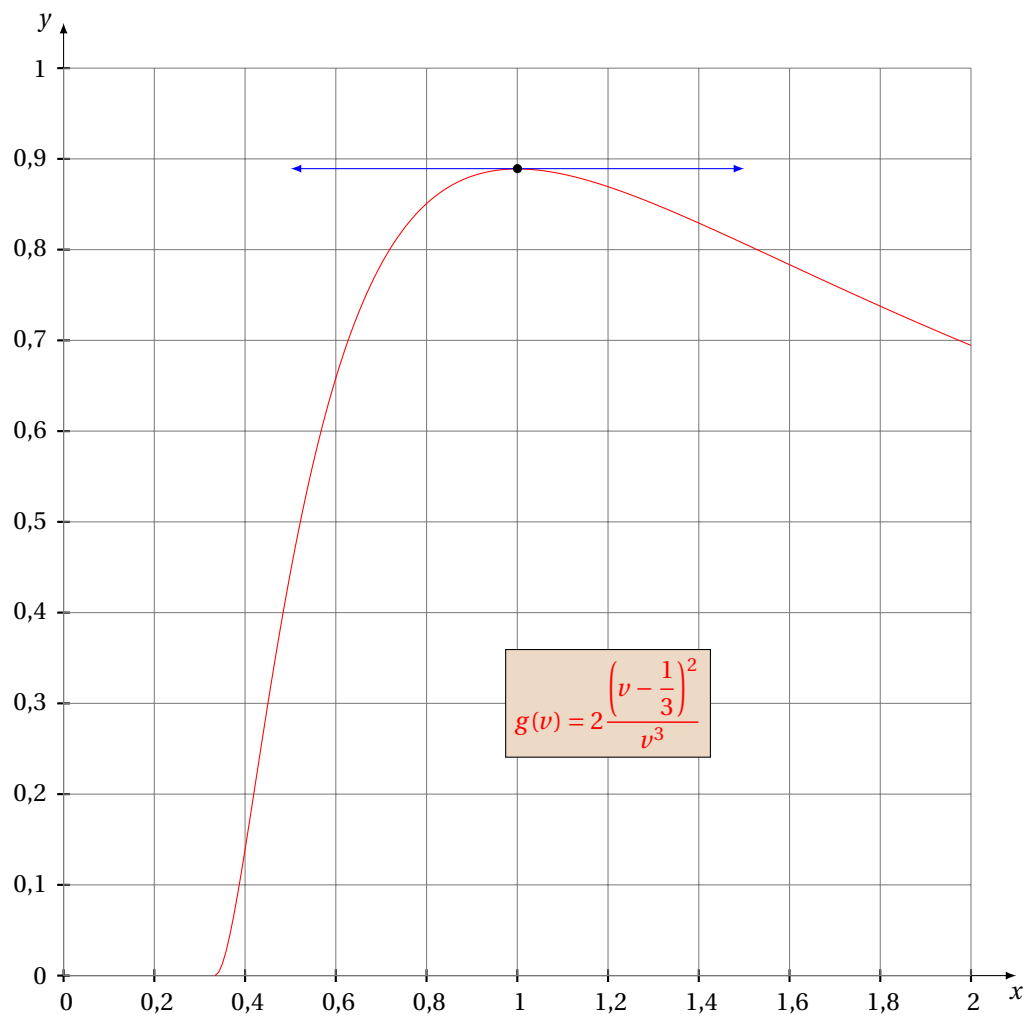
```

13.7.2 Première courbe avec $b=1$

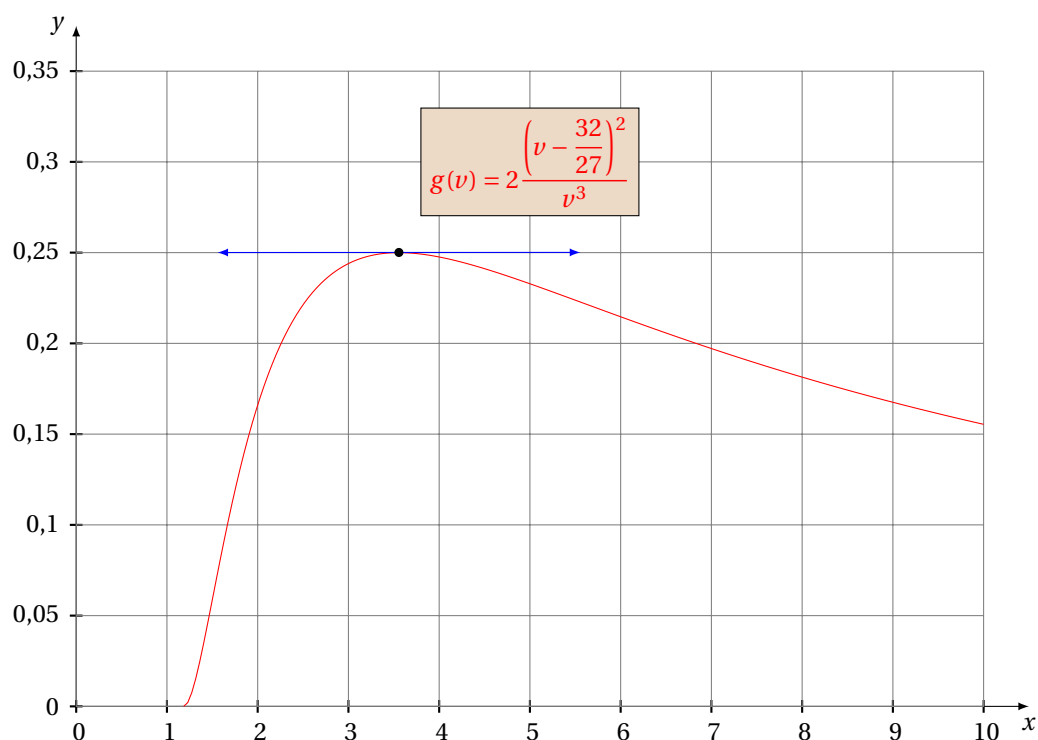
Quelques courbes pour $r \leq v \leq 6$



```
\begin{tikzpicture}[xscale=2,yscale=2.5]
  \tkzInit[xmin=0,xmax=6,ymax=0.5,ystep=0.1]
  \tkzDrawX[label=$v$]
  \tkzDrawY[label=$g(v)$]
  \tkzGrid(0,0)(6,0.5)
  \tkzFct[color = red,domain =1:6]{(2*(x-1)*(x-1))/(x*x*x)}
  \tkzDrawTangentLine[color=blue,draw](3)
  \tkzDefPointByFct(1)
  \tkzText[draw, fill = brown!30](4,0.1){$g(v)=2\dfrac{(v-1)^2}{v^3}$}
\end{tikzpicture}
```


13.7.3 Deuxième courbe $b=1/3$ 

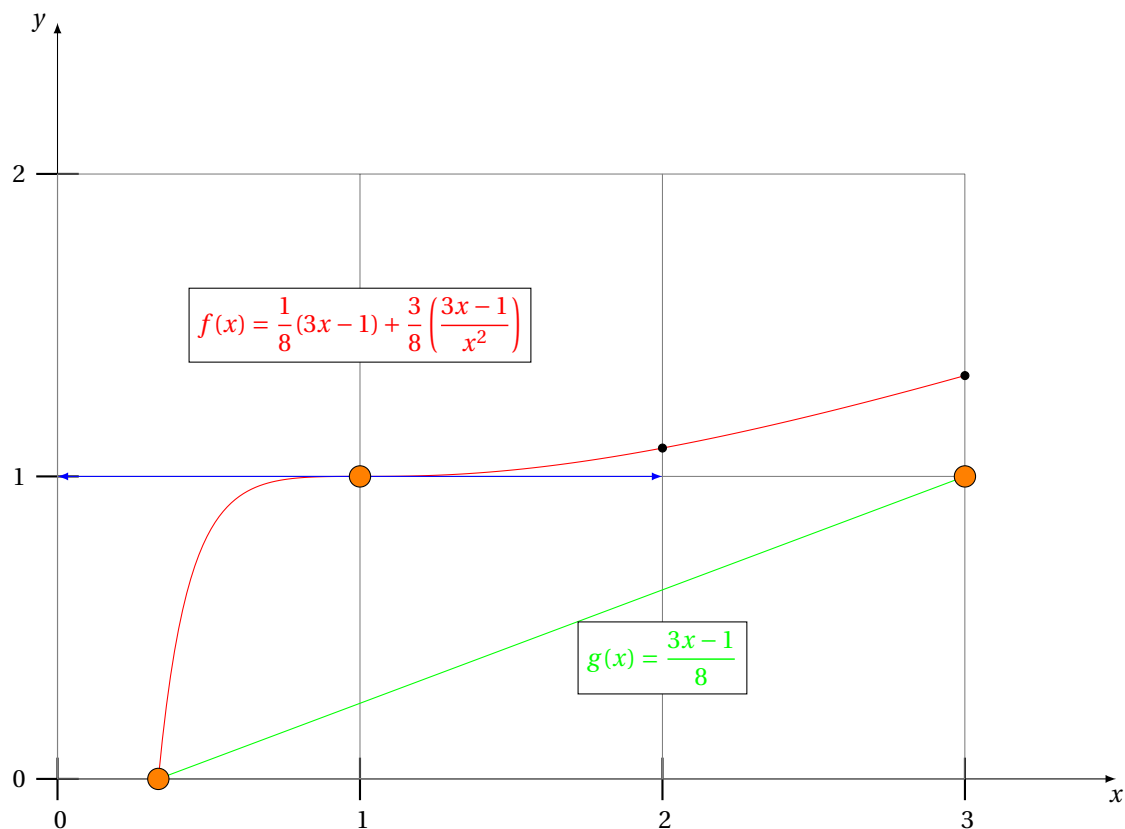
```
\begin{tikzpicture}[scale=1.2]
  \tkzInit[xmin=0,xmax=2,xstep=0.2,ymax=1,ystep=0.1]
  \tkzAxeXY
  \tkzGrid(0,0)(2,1)
  \tkzFct[color = red,domain =1/3:2]{(2*(\x-1./3)*(\x-1./3))/(\x*\x*\x)}
  \tkzDrawTangentLine[draw,color=blue,kr=.5,kl=.5](1)
  \tkzDefPointByFct(1)
  \tkzText[draw,fill = brown!30](1.2,0.3)%
    {\$g(v)=2\frac{\left(v-\dfrac{1}{3}\right)^2}{v^3}\$}
\end{tikzpicture}
```

13.7.4 Troisième courbe $b=32/27$ 

```
\begin{tikzpicture}[scale=1.2]
  \tkzInit[xmin=0,xmax=10,ymax=.35,ystep=.05];
  \tkzAxeXY
  \tkzGrid(0,0)(10,.35)
  \tkzFct[color = red,
    domain = 1.185:10]{(2*(\x-32./27)*(\x-32./27))/(\x*\x*\x)}
  \tkzDrawTangentLine[draw,color=blue,kr=2,kl=2](3.555)
  \tkzText[draw,fill = brown!30](5,0.3)%
    {\$g(v)=2\dfrac{\left(v-\dfrac{32}{27}\right)^2}{v^3}\$}
\end{tikzpicture}
```

13.8 Valeurs critiques

13.8.1 Courbes de Van der Walls

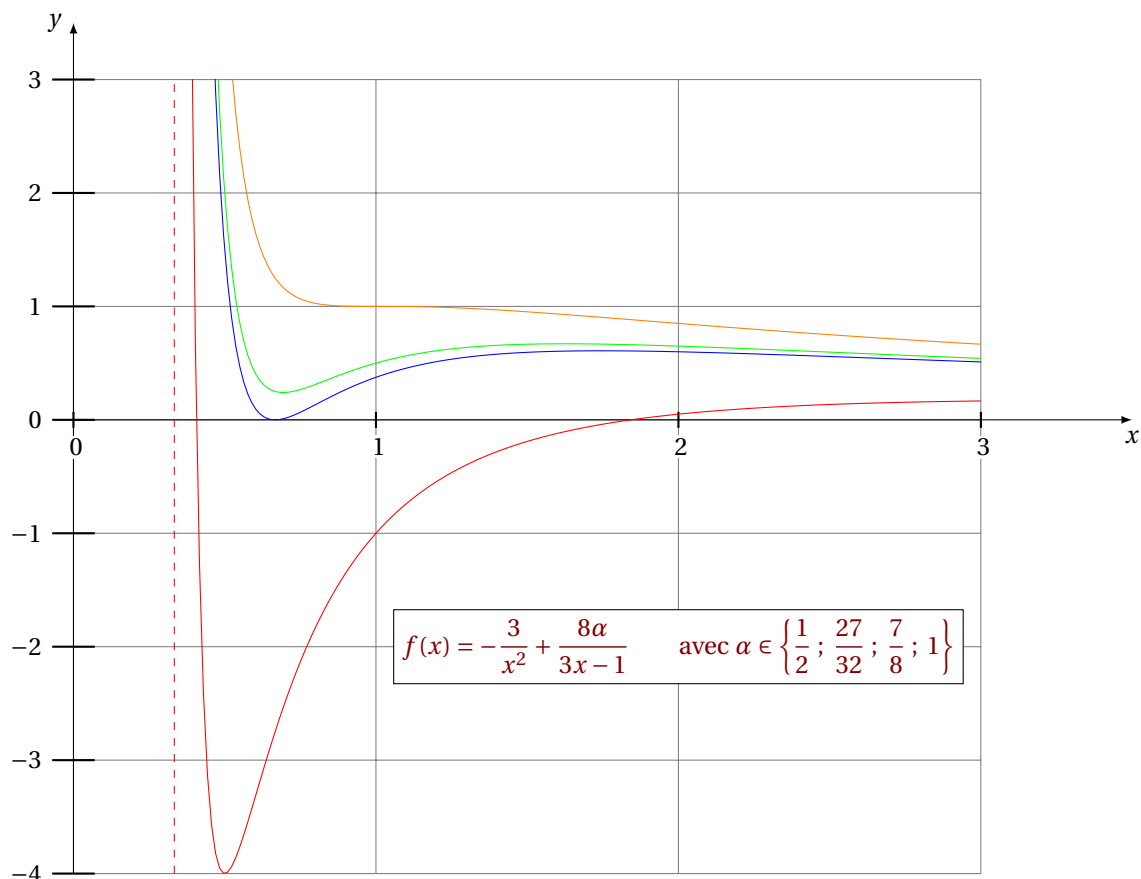


```

\begin{tikzpicture}[scale=4]
  \tkzInit[xmax=3,ymax=2];
  \tkzAxeXY
  \tkzGrid(0,0)(3,2)
  \tkzFct[color = red,domain =1/3:3]{0.125*(3*\x-1)+0.375*(3*\x-1)/(\x*\x)}
  \tkzDefPointByFct[draw](2)
  \tkzDefPointByFct[draw](3)
  \tkzDrawTangentLine[draw,color=blue](1)
  \tkzFct[color = green,domain =1/3:3]{0.125*(3*\x-1)}
  \tkzSetUpPoint[size=8,fill=orange]
  \tkzDefPointByFct[draw](3)
  \tkzDefPointByFct[draw](1/3)
  \tkzDefPoint(1,1){f}
  \tkzDrawPoint(f)
  \tkzText[draw,fill = white,text=red](1,1.5){
    {\$f(x)=\dfrac{1}{8}(3x-1)+\dfrac{3}{8}\left(\dfrac{3x-1}{x^2}\right)\$}
  }
  \tkzText[draw,fill = white,text=green](2,0.4){\$g(x) = \dfrac{3x-1}{8}\$}
\end{tikzpicture}

```

13.8.2 Courbes de Van der Walls (suite)

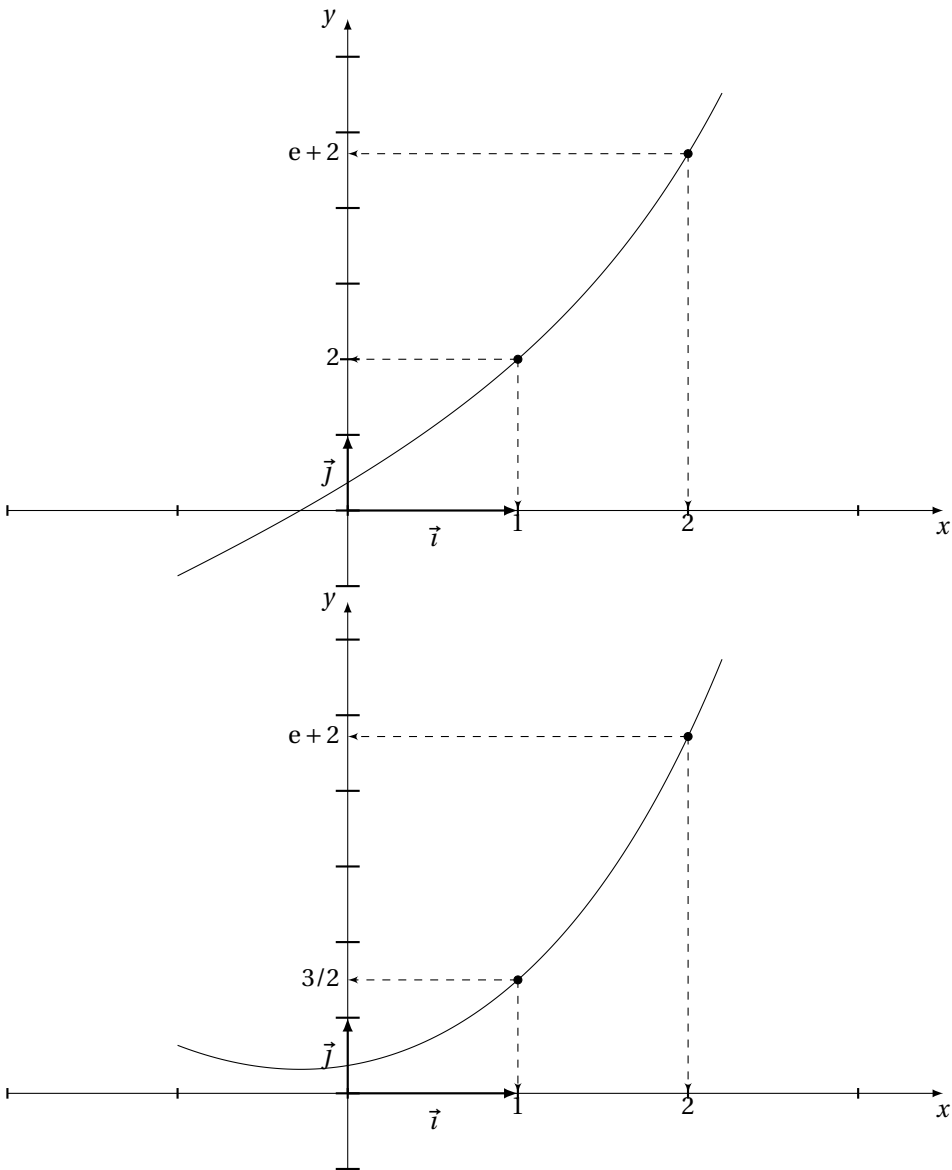


```

\begin{tikzpicture}[xscale=4,yscale=1.5]
  \tkzInit[xmin=0,xmax=3,ymax=3,ymin=-4]
  \tkzGrid(0,-4)(3,3)
  \tkzAxeXY
  \tkzClip
  \tkzVLine[color=red,style=dashed]{1/3}
  \tkzFct[color=red,domain = 0.35:3]{-3/(x*x) +4/(3*x-1)}
  \tkzFct[color=blue,domain = 0.35:3]{-3/(x*x) +27/(4*(3*x-1))}
  \tkzFct[color=orange,domain = 0.35:3]{-3/(x*x) +8/(3*x-1)}
  \tkzFct[color=green,domain = 0.35:3]{-3/(x*x) +7/(3*x-1)}
  \tkzText[draw,fill = white,text=Maroon](2,-2)%
    {$f(x)=-\dfrac{3}{x^2}+\dfrac{8\alpha}{3x-1}$ \hspace{.5cm}%
    avec $\alpha$ \in%
    \left\{\dfrac{1}{2};~\dfrac{27}{32};~\dfrac{7}{8};~1\right\}$}
\end{tikzpicture}

```

14 Exemples avec les packages alterqcm et tkz-tab

Questions	Réponses
<p>La figure 1. donne la représentation graphique d'une fonction f définie sur \mathbf{R}^+ et la figure 2 celle d'une primitive de f sur \mathbf{R}^+.</p> 	
<p>1. Quelle est l'aire, en unités d'aire, de la partie du plan limitée par la représentation graphique de la fonction f, l'axe des abscisses et les droites d'équation $x = 1$ et $x = 2$?</p>	<p> <input type="checkbox"/> $e + \frac{3}{4}$ <input type="checkbox"/> $e + \frac{1}{2}$ <input type="checkbox"/> 1 </p>

Questions		Réponses																														
La fonction k définie et strictement positive sur \mathbf{R}^+ est connue par son tableau de variations.																																
<table><tr><td>x</td><td>0</td><td>1</td><td>3</td><td>$+\infty$</td></tr><tr><td>$k(x)$</td><td></td><td>\nearrow</td><td>\searrow</td><td>\nearrow</td></tr></table>	x	0	1	3	$+\infty$	$k(x)$		\nearrow	\searrow	\nearrow																						
x	0	1	3	$+\infty$																												
$k(x)$		\nearrow	\searrow	\nearrow																												
2. Parmi les tableaux suivants, quel est le tableau de variations de la fonction g définie sur \mathbf{R}^+ par $g(x) = \frac{1}{k(x)}$?		<div><input type="checkbox"/> Tableau A</div> <div><input type="checkbox"/> Tableau B</div> <div><input type="checkbox"/> Tableau C</div>																														
<div>Tableau A</div> <table><tr><td>x</td><td>0</td><td>1</td><td>3</td><td>$+\infty$</td></tr><tr><td>$g(x)$</td><td></td><td>\nearrow</td><td>\searrow</td><td>\nearrow</td></tr></table> <div>Tableau B</div> <table><tr><td>x</td><td>0</td><td>1</td><td>3</td><td>$+\infty$</td></tr><tr><td>$g(x)$</td><td></td><td>\searrow</td><td>\nearrow</td><td>\searrow</td></tr></table> <div>Tableau C</div> <table><tr><td>x</td><td>0</td><td>1</td><td>3</td><td>$+\infty$</td></tr><tr><td>$g(x)$</td><td></td><td>\nearrow</td><td>\searrow</td><td>0</td></tr></table>			x	0	1	3	$+\infty$	$g(x)$		\nearrow	\searrow	\nearrow	x	0	1	3	$+\infty$	$g(x)$		\searrow	\nearrow	\searrow	x	0	1	3	$+\infty$	$g(x)$		\nearrow	\searrow	0
x	0	1	3	$+\infty$																												
$g(x)$		\nearrow	\searrow	\nearrow																												
x	0	1	3	$+\infty$																												
$g(x)$		\searrow	\nearrow	\searrow																												
x	0	1	3	$+\infty$																												
$g(x)$		\nearrow	\searrow	0																												
3. Soit h la fonction définie sur \mathbf{R} par $h(x) = e^x - x + 1$. On note \mathcal{C} la courbe représentative de h dans un repère orthonormal $O; \vec{i}; \vec{j}$.		<div><input type="checkbox"/> La droite d'équation $y = 1$ est asymptote à \mathcal{C}</div> <div><input type="checkbox"/> La droite d'équation $x = 0$ est asymptote à \mathcal{C}</div> <div><input type="checkbox"/> La droite d'équation $y = -x + 1$ est asymptote à \mathcal{C}</div>																														
4. En économie, le coût marginal est le coût occasionné par la production d'une unité supplémentaire, et on considère que le coût marginal est assimilé à la dérivée du coût total. Dans une entreprise, une étude a montré que le coût marginal $C_m(q)$ exprimé en milliers d'euro en fonction du nombre q d'articles fabriqués est donné par la relation : $C_m(q) = 3q^2 - 10q + \frac{2}{q} + 20.$		<div><input type="checkbox"/> $C_r(q) = q^3 - 5q^2 + 2\ln q + 20q + 9984$</div> <div><input type="checkbox"/> $C_r(q) = q^3 - 5q^2 + 2\ln q + 20q - 6$</div> <div><input type="checkbox"/> $C_r(q) = 6q - 10 - \frac{2}{q^2}$</div>																														

Voici le code des deux représentations de f et de sa primitive :

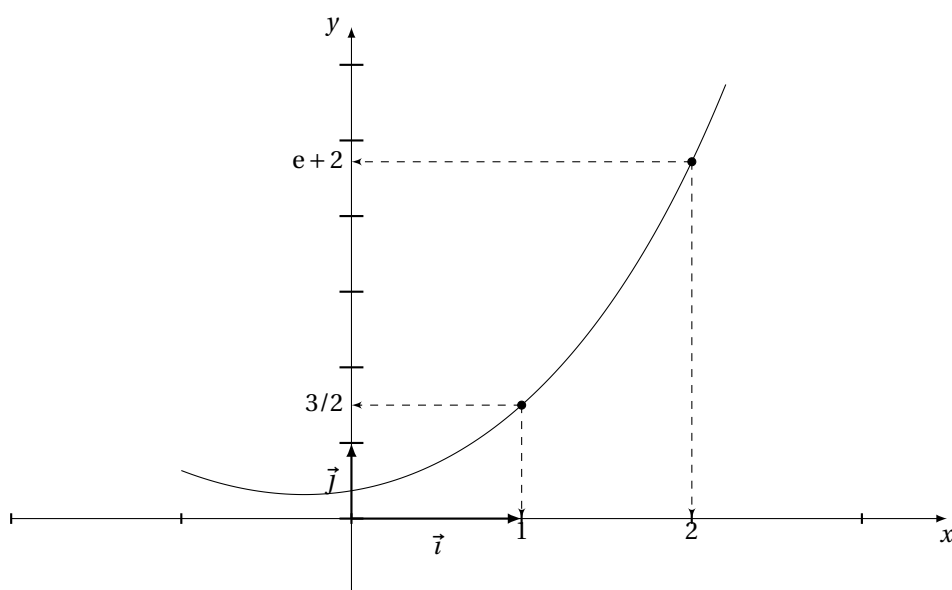
14.0.1 Première représentation

```

\begin{tikzpicture}[xscale=2.25,yscale=1]
  \tkzInit[xmin=-2,xmax=3,ymin=-1,ymax=6]
  \tkzDrawX
  \tkzDrawY
  \tkzFct[samples=100,domain = -1:2.2]{x+exp(x-1)}
  \tkzDefPoint(1,2){pt1}
  \tkzDrawPoint(pt1)
  \tkzPointShowCoord[xlabel=$1$,ylabel=$2$](pt1)
  \tkzDefPoint(2,4.71828){pt2}
  \tkzDrawPoint(pt2)
  \tkzPointShowCoord[xlabel=$2$,ylabel=$\text{e}+2$](pt2)
  \tkzRep
\end{tikzpicture}

```

14.0.2 Seconde représentation



```

\begin{tikzpicture}[xscale=2.25,yscale=1]
  \tkzInit[xmin=-2,xmax=3,ymin=-1,ymax=6]
  \tkzDrawX
  \tkzDrawY
  \tkzFct[samples=100,domain = -1:2.2]{x*x/2+exp(x-1)}
  \tkzDefPoint(1,1.5){pt1}
  \tkzDrawPoint(pt1)
  \tkzPointShowCoord[xlabel=$1$,ylabel=$3/2$](pt1)
  \tkzDefPoint(2,4.71828){pt2}
  \tkzDrawPoint(pt2)
  \tkzPointShowCoord[xlabel=$2$,ylabel=$\text{e}+2$](pt2)
  \tkzRep
\end{tikzpicture}

```

Code d'un tableau de variations

```

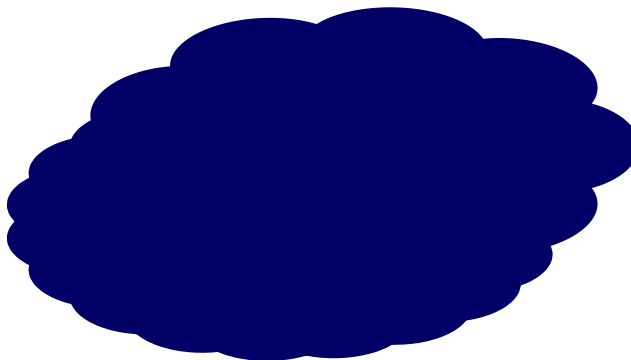
\begin{tikzpicture}
  \tkzTabInit[lgt=1,espcl=2]{ $x/0.5$ , $k(x)/1.5$ }
  { $0$ , $1$ , $3$ , $+\infty$ }
  \tkzTabVar{-/          /,%
             +/          /,%
             -/          /,%
             +/  $+\infty$  /}%
\end{tikzpicture}

```


15 Utilisation pgfmath et de fp.sty

15.1 pgfmath

On peut faire maintenant beaucoup de tracés sans Gnuplot, voici à titre d'exemple et d'après une idée d'Herbert Voss (le membre le plus actif de la communauté Pstricks) un exemple de courbes obtenues avec seulement Tikz.



```
\begin{tikzpicture}
\def\Asmall{0.7 } \def\Abig{3 } \def\B{20}%Herbert Voss
\path[fill=blue!40!black,domain=-pi:pi,samples=500,smooth,variable=\t]%
plot({\Abig*cos(\t r)+\Asmall*cos(\B*\t r)},%
{0.5*\Abig*sin(\t r)+0.5*\Asmall*sin(\B*\t r)});
\def\Asmall{0.7 } \def\Abig{3 } \def\B{10}
\path[shift={(1,1)},fill=blue!40!black,%
domain=-pi:pi,samples=500,smooth,variable=\t]%
plot({\Abig*cos(\t r)+\Asmall*cos(\B*\t r)},%
{0.5*\Abig*sin(\t r)+0.5*\Asmall*sin(\B*\t r)});
\end{tikzpicture}
```

15.2 fp.sty

Le principal problème de `fp.sty` se produit lors de l'évaluation par exemple de $(-4)^2$ ce qui peut se traduire avec `fp` par :

```
\begin{tikzpicture}
\FPeval\result{(-4)^2}
\end{tikzpicture}
```

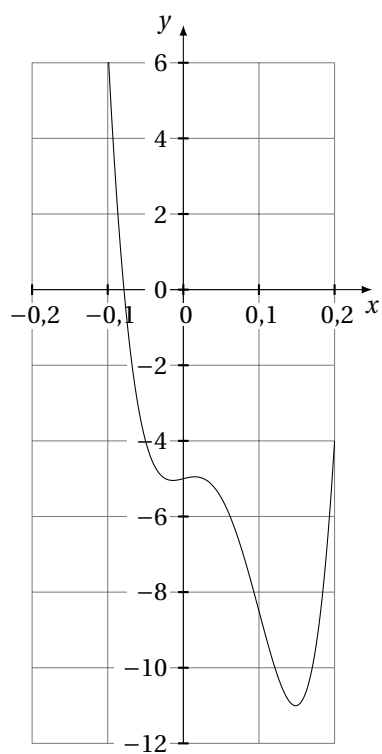
ce qui donne une erreur car `fp` utilise les logarithmes pour faire cette évaluation. `tkz-fct.sty` modifie la macro `\FP@pow` pour éviter cette erreur

Pour calculer les pentes des tangentes et pour placer des points sur les courbes, mon module traduit l'expression donnée pour Gnuplot et la stocke dans une commande `\tkzFcta`, pour être utilisée ensuite avec les macros `\tkzDefPointByFct` et `\tkzDrawTangentLine`.

mais si vous voulez placer un point de ce graphe ayant pour abscisse $x = 2$, il est alors préférable de choisir la première méthode.

Sinon pour une fonction polynomiale, il sera nécessaire pour utiliser les macros relatives aux images et aux tangentes de mettre le polynôme sous la forme d'Horner. Ainsi avec `\tkzFct`, l'argument $x^4 - 2x^3 + 4x - 5$ peut être écrit : `-5+x*(0.5+4*x*(x*(-2+x*1)))`.

Voici ce qu'il faut donc faire :



```
\begin{tikzpicture}
  \tkzInit[xmin=-0.2,xmax=0.2,xstep=.1,
    ymin=-12,ymax=6,ystep=2]
  \tkzGrid
  \tkzAxeXY
  \tkzFct[domain = -.1:.2]%
    {-5+x*(0.5+4*x*(x*(-2+x*1)))}
\end{tikzpicture}
```

16 Quelques remarques

1. Modification avec les anciennes versions :
 - `\tkzTan` est devenu `\tkzDrawTangentLine`
 - Désormais le domaine est donné comme avec TikZ et ce n'est plus $\langle x_a \dots x_b \rangle$
 - `\tkzFctPt` est devenu `\tkzDefPointByFct`
2. Quand `xstep` est différent de 1, la variable doit être `\x`.
3. Quand une fonction est passée en argument à la macro `\tkzFct`, elle est stockée avec la syntaxe de `gnuplot` dans la macro `\tkzFctgnu`. `tkzFctgnu` est un préfixe, « a » est la référence associée à la fonction, la fonction suivante dans le même environnement `tikzpicture` sera référencée « b » et ainsi de suite... Elle est aussi stockée avec la syntaxe de `fp.sty` dans la macro `\tkzFcta` avec le préfixe `tkzFcta`. La dernière macro utilisée est également sauvegardée sous les deux syntaxes avec `\tkzFctgnuLast` et `\tkzFctLast`.
4. Attention dans `gnuplot` un quotient doit être entré sous la forme `1./3`, car `1/3` donne le quotient d'une division euclidienne (ici 0).
5. Problème avec `gnuplot` :
 - Si le fichier `xxx.table` n'est pas créé, la cause probable est :
 - soit que \TeX ne trouve pas `gnuplot`, c'est en général un problème de « PATH »,
 - soit \TeX n'autorise pas le lancement de `gnuplot` alors c'est que l'option `shell-escape` n'est pas autorisé.
 Une autre possibilité est que le fichier `xxx.gnuplot` soit incorrect. Il suffit de l'ouvrir avec un éditeur pour lire les commandes passées à `gnuplot`. Il est à remarquer un changement de syntaxe de `gnuplot` autour de la version 4.2. La syntaxe pour créer une table avec des versions ultérieures (4.4 et bientôt 4.5), est désormais `set table`.
 - π est, avec `gnuplot`, défini par `pi`
 - π est, avec `fp.sty` défini par `\FPpi`.
 - `(set) samples = 2` est suffisant pour tracer une droite.
6. La puissance a^b est notée `a ^ b` avec `fp` et `pgfmath` mais `a * * b` avec `gnuplot`.
7. `tkz-fct` modifie `FP@pow` (code modifié de Christian Tellechea 2009) afin d'autoriser les puissances entières de nombres négatifs.
8. `(1/exp(1))` est correct mais `(1/exp(1))` donne une erreur

16.1 Fonctions de gnuplot

Gnuplot	fp	Description
+	+	addition
-	-	soustraction
*	*	multiplication
/	/	division
**	^	exponentiation
%	absente	modulo
pi	pi	constante 3.1415
abs(x)	abs	Valeur absolue
cos(x)	cos	Arc -cosinus
sin(x)	sin	Arc -cosinus
tan(x)	tan	Arc -cosinus
acos(x)	arccos	Arc -cosinus
asin(x)	arcsin	Arc-sinus
atan(x)	arctan	Arc-tangente
atan2(y,x)	absente	Arc-tangente
cosh(x)	absente	Cosinus hyperbolique
sinh(x)	absente	Sinus hyperbolique
acosh(x)	absente	Arc-cosinus hyperbolique
asinh(x)	absente	Arc-sinus hyperbolique
atanh(x)	absente	Arc-tangente hyperbolique
besj0(x)	absente	Bessel j0
besj1(x)	absente	Bessel j1
besy0(x)	absente	Bessel y0
besy1(x)	absente	Bessel y1
ceil(x)	absente	Le plus petit entier plus grand que
floor(x)	absente	Plus grand entier plus petit que
absente	trunc(x,n)	troncature n nombre de décimales
absente	round(x,n)	arrondi n nombre de décimales
exp(x)	exp	Exponentielle
log(x)	ln	Logarithme népérien (base e)
log10(x)	absente	Logarithme base 10
norm(x)	absente	Distribution normale
rand(x)	random	Générateur de nombre pseudo-aléatoire
sgn(x)	absente	Signe
sqrt(x)	absente	Racine carrée
tanh(x)	absente	Tangente hyperbolique

17 Liste de toutes les macros

17.1 Liste de toutes les macros fournies par ce package

- `\tkzFct[samples=200,domain=-5:5,color=black,id=tkzfct]{⟨gnuplot's expression⟩}`
- `\tkzDefPointByFct[draw=false](⟨point's name⟩) -> tkzPointResult`
- `\tkzDrawTangentLine[draw=false,color=black,kr=1,kl=1,style=solid,with=a](⟨point's name⟩)`
- `\tkzDrawArea[domain=-5:5,color=lightgray,opacity=.5]`
- `\tkzArea[domain=-5:5,color = lightgray,opacity=.5]`
- `\tkzDrawAreafg[domain=-5:5,between= a and b]`
- `\tkzAreafg[domain=-5:5,between= a and b]`
- `\tkzFctPar[samples=200,domain=-5:5, line width=1pt,id=tkzfctpar] $x(t)y(t)$`
- `\tkzFctPolar[samples=200,domain=0:2*pi, line width=1pt,id=tkzfctpolar] $\rho(t)$`
- `\tkzDrawRiemannSum[interval=1:2,number=10,fill=gray]`
- `\tkzDrawRiemannSumInf [interval=1:2,opacity=.5,fill=gray]`
- `\tkzDrawRiemannSumSup [interval=1:2,number=10,fill=gray]`
- `\tkzDrawRiemannSumMid[interval=1:2,opacity=1,fill=gray]`

17.2 Liste de toutes des macros essentielles de `\tkz-base`

- `\tkzInit[xmin=0,xmax=10,xstep=1,ymin=0,ymax=10,ystep=1]`
- `\tkzAxeX`
- `\tkzDrawX`
- `\tkzLabelX`
- `\tkzAxeY`
- `\tkzDrawY`
- `\tkzLabelY`
- `\tkzGrid`
- `\tkzClip`
- `\tkzDefPoint`
- `\tkzDrawPoint`
- `\tkzPointShowCoord`
- `\tkzLabelPoint`

Index

- `\draw plot function`, 6
- `\draw plot[id=fct] function---`., 7
- `\FPpi`, 57, 83
- `\jobname`, 8
- Operating System
 - Linux Ubuntu, 9
 - OS X, 9
 - Windows XP, 9
- `\t**2`, 50, 57
- `\t**3`, 50, 57
- `\tikzset{tan style/.style={->,>=latex}}`, 23
- `\tikzset{tan style/.style={-}}`, 23
- `\tkz-base`, 85
- `\tkzArea`, 32
- `\tkzAxeX`, 85
- `\tkzAxeY`, 85
- `\tkzClip`, 85
- `\tkzDefPoint`, 85
- `\tkzDefPointByFct(0)`, 17
- `\tkzDefPointByFct`, 17, 81, 83
- `\tkzDefPointByFct`: arguments
 - decimal number, 17
- `\tkzDefPointByFct`: options
 - draw, 17
 - ref, 17
 - with, 17
- `\tkzDefPointByFct(<decimalnumber>)`, 17
- `\tkzDrawArea`, 32
- `\tkzDrawArea`: options
 - color, 32
 - domain, 32
 - opacity, 32
 - style, 32
 - with, 32
- `\tkzDrawAreafg`, 36
- `\tkzDrawAreafg`: options
 - between, 36
 - domain= min:max, 36
 - opacity, 36
- `\tkzDrawAreafg[<local options>]`, 36
- `\tkzDrawArea[<local options>]`, 32
- `\tkzDrawPoint`, 17, 85
- `\tkzDrawRiemannSum`, 41
- `\tkzDrawRiemannSum`: options
 - interval, 41
 - number, 41
- `\tkzDrawRiemannSumInf`, 42
- `\tkzDrawRiemannSumInf[<local options>]`, 42

- \tkzDrawRiemannSumMid, 44
- \tkzDrawRiemannSumMid[[⟨local options⟩](#)], 44
- \tkzDrawRiemannSumSup, 43
- \tkzDrawRiemannSumSup[[⟨local options⟩](#)], 43
- \tkzDrawRiemannSum[[⟨local options⟩](#)]{[⟨f\(t\)⟩](#)}, 41
- \tkzDrawTangentLine(0), 23
- \tkzDrawTangentLine, 23, 25, 81, 83
- \tkzDrawTangentLine: arguments
 - a, 23
- \tkzDrawTangentLine: options
 - draw, 23
 - kl, 23
 - kr, 23
 - with, 23
- \tkzDrawTangentLine[[⟨local options⟩](#)]([⟨a⟩](#)), 23
- \tkzDrawX, 85
- \tkzDrawY, 85
- \tkzFct, 6, 12, 23, 81, 83
- \tkzFct: arguments
 - gnuplot expression, 12
- \tkzFct: options
 - color, 12
 - domain, 12
 - id, 12
 - line width, 12
 - samples, 12
 - style, 12
- \tkzFcta, 23, 81, 83
- \tkzFctb, 23
- \tkzFctgnua, 83
- \tkzFctgnuLast, 83
- \tkzFctLast, 19, 29, 30, 83
- \tkzFctPar[0:1], 50, 57
- \tkzFctPar, 50
- \tkzFctPar: arguments
 - $x(t), y(t)$, 50
- \tkzFctPar: options
 - color, 50
 - domain, 50
 - id, 50
 - line width, 50
 - samples, 50
 - style, 50
- \tkzFctPar[[⟨local options⟩](#)]{[⟨x\(t\)⟩](#)}{[⟨y\(t\)⟩](#)}, 50
- \tkzFctPolar, 57
- \tkzFctPolar: arguments
 - $x(t), y(t)$, 57
- \tkzFctPolar: options
 - color, 57
 - domain, 57
 - id, 57
 - line width, 57
 - samples, 57

- style, 57
- `\tkzFctPolar`[`<local options>`]{`<f(t)>`}, 57
- `\tkzFctPt`, 17, 83
- `\tkzFctk`, 47
- `\tkzFct`[`<local options>`]{`<gnuplot expression>`}, 12
- `\tkzGetPoint`, 17, 18
- `\tkzGrid`, 85
- `\tkzHLine`, 47
- `\tkzHLine`: arguments
 - decimal number, 47
- `\tkzHLines`{1,4}, 48
- `\tkzHLines`, 48
- `\tkzHLines`: arguments
 - list of values, 48
- `\tkzHLines`[`<local options>`]{`<list of values>`}, 48
- `\tkzHLine`[`<local options>`]{`<decimal number>`}, 47
- `\tkzInit`, 6, 12, 23, 85
- `\tkzLabelPoint`, 85
- `\tkzLabelX`, 85
- `\tkzLabelY`, 85
- `\tkzPointShowCoord`, 85
- `\tkzSetUpPoint`, 21
- `\tkzTan`, 83
- `\tkzText`, 21, 22
- `\tkzVLine`{1}, 45, 47
- `\tkzVLine`, 45
- `\tkzVLine`: arguments
 - decimal number, 45
- `\tkzVLine`: options
 - color , 45
 - line width, 45
 - style , 45
- `\tkzVLines`{1,4}, 46
- `\tkzVLines`, 46
- `\tkzVLines`: arguments
 - list of values, 46
- `\tkzVLines`[`<local options>`]{`<list of values>`}, 46
- `\tkzVLine`[`<local options>`]{`<decimal number>`}, 45
- `\x`, 6, 12, 83