4 – Courbes en PsTricks

On peut, en LaTeX, tracer des courbes au moyen d'extensions qu'il suffit de charger dans le préambule du document.
J'utilise depuis mes débuts l'extension `PsTricks` qui permet de faire de belles choses ; on entre donc dans le préambule :

```
\usepackage{pstricks-add}
```

1 Quelques préliminaires

L'extension `pstricks` qui maintenant s'appelle `pstricks-add` (on lui a rajouté des commandes), permet de définir un environnement `pspicture` qui délimite une zone dans laquelle on va pouvoir :

- tracer des axes avec `\psaxes` ;
- tracer un quadrillage avec `\psgrid` ;
- placer des points avec `\psdots` et écrire leurs noms avec `\uput` ou `\rput` ;
- tracer des segments avec `\psline` ou des vecteurs en rajoutant une option à cette commande ;
- tracer des cercles avec `\pscircle` ou des arcs de cercle avec `\psarc` ;
- tracer des courbes avec `\psplot` ou avec `\pscurve` ;
- etc.

La liste est longue.

Mais on n'est pas obligé d'être dans un environnement `pspicture` pour utiliser des commandes `PsTricks`.
Si, par exemple, on veut écrire « Tourner la feuille » et dessiner une belle flèche en bas de la première page d'un devoir, comme ici :

Tourner la feuille ⟶

il suffit d'entrer :

```
\begin{flushright}
Tourner la feuille \hspace{2cm}\psline[arrowsize=3pt 3]{->}(-2,0.1)(0,0.1)
\end{flushright}
```

L'environnement `flushright` sert à aligner sur la droite la suite de caractères « Tourner la page » suivie d'un espacement horizontal de 2 cm, défini par `\hspace{2cm}`.

`\psline` est la commande qui sert à tracer un trait, et on adjoint `{->}` pour tracer un vecteur ; ce vecteur partira du point de coordonnées $(-2\,,0.1)$ et s'arrêtera au point de coordonnées $(0\,,0.1)$.

Le plupart du temps, les options facultatives sont entrées entre crochets : `[arrowsize=3pt 3]` permet d'agrandir un peu la taille des flèches. Si on voulait changer la couleur de la flèche pour la dessiner en rouge, on entrerait `linecolor=red` entre les crochets, à la suite de `arrowsize` ; et il faudrait mettre une virgule comme séparateur : `[arrowsize=3pt 3, linecolor=red]`.

Quand il n'y a pas de repère défini par l'environnement `pspicture`, l'emplacement où l'on écrit la commande est de coordonnées $(0\,,0)$.

L'unité par défaut étant le cm, cette flèche mesurera 2 cm, ce qui explique le `\hspace{2cm}` qui réserve la place nécessaire à droite du texte pour tracer la flèche.

Quant à l'ordonnée 0.1, c'est juste pour remonter un peu la flèche qui serait trop basse sinon (et ce serait moins joli !).

Enfin si on veut être sûr que le « Tourner la feuille » soit le plus bas possible dans la page, on fera précéder les instructions par un `\vfill` qui poussera la phrase le plus bas possible, et on les fera suivre d'un `\newpage` pour passer à la page suivante :

```
\vfill
\begin{flushright}
Tourner la feuille \hspace{2cm}\psline[arrowsize=3pt 3]{->}(-2,0.1)(0,0.1)
\end{flushright}
\newpage
```

2 Création du repère

Que faut-il pour tracer une courbe ? D'abord un repère, c'est-à-dire :

- une origine O ;
- deux droites sécantes en O, qui seront perpendiculaires dans le cas d'un repère orthogonal. Ces axes sont dessinés avec des flèches au bout dans le cas d'un repère (O, I, J), sans flèche dans le cas de repère $(O\,;\vec{\imath},\vec{\jmath})$ ou $(O\,;\vec{u},\vec{v})$;
- des points I et J, ou des vecteurs $\vec{\imath}$ et $\vec{\jmath}$, ou $\vec{u}$ et $\vec{v}$, pour les unités. En plus du dessin des points ou des vecteurs, il faut écrire leurs noms ;
- éventuellement un quadrillage, voire du papier millimétré.

On détaille les différentes instructions nécessaires.

`\psset{xunit=1cm, yunit=1cm, runit=1cm}`
définit les unités du repère que l'on va créer : c'est assez explicite pour `xunit` et `yunit` ; quant à `runit`, c'est l'unité pour tracer des cercles.
Si toutes les unités sont égales à 1 cm, on peut écrire `unit=1cm`.
Il faut définir les unités avant de définir la zone graphique.

`\begin{pspicture}(-3,-5)(4,6)`
définit une zone graphique rectangulaire qui va du point de coordonnées `(-3,-5)` au point de coordonnées `(4,6)` ; ça correspond à des abscisses entre -3 et 4, et des ordonnées entre -5 et 6.
On terminera cet environnement en entrant `\end{pspicture}`.

`\psgrid[subgriddiv=2, gridlabels=0, gridcolor=gray, subgridcolor=orange]`
trace un quadrillage dont la couleur sera grise (`gridcolor=gray`) ; ce quadrillage sera divisé en deux (`subgriddiv=2`). Pour du papier millimétré, on prendra des unités à 1 cm et on entrera `subgriddiv=10`. La couleur de cette subdivision est gérée par `subgridcolor`, orange ici.
Enfin pour ne pas que les nombres de -3 à 4 sur l'axe des abscisses, et de -5 à 6 sur l'axe des ordonnées, s'affichent, on entre `gridlabels=0`.
Quand je trace une grille, je la trace toujours avant les axes pour que le dessin des axes se fasse par dessus celui de la grille (et non le contraire).

`\psaxes[arrowsize=3pt 3, ticksize=-2pt 2pt, labels=none]{->}(0,0)(-3,-5)(4,6)`
`\psaxes[ticksize=-2pt 2pt, labels=none](0,0)(-3,-5)(4,6)`
permettent de tracer des axes, avec ou sans flèches (présence de `{->}` ou non).
Le `(0,0)` indique à quel endroit les axes doivent se couper, et on remet les coordonnées du rectangle défini par `pspicture`. Il m'est arrivé, pour des raisons esthétiques, de « tricher » un peu et de mettre des axes un peu plus longs que le rectangle définie au départ. À voir au cas par cas.
L'égalité `ticksize=-2pt 2pt` définit la dimension du petit trait qui marque les unités entières : 2 points en dessous de la ligne, 2 points au dessus.
Et `labels=none` empêche la numérotation sur les axes ; on le retirera si on veut que cette numérotation soit présente.

À l'intersection des axes, se trouve l'origine O. Il suffit donc d'écrire le nom du point là où il faut. Pour écrire dans un environnement `pspicture`, j'utilise `\uput` qui nécessite trois paramètres : entre crochets la position par rapport au point, entre parenthèses les coordonnées du point d'ancrage, entre accolades ce que l'on veut écrire. Pour le point O, on écrira `\uput[dl](0,0){$O$}`.
Le `dl` veut dire `down left` ou en français, `en bas à gauche`. Les positions de base sont `u` pour `up`, `d` pour `down`, `l` pour `left` et `r` pour `right`, que l'on peut combiner en `dl`, `ur`, etc.
Pour un réglage plus fin, on peut entrer la valeur d'un angle en degrés, comme sur un cercle trigonométrique : le `ur` correspond à `45°` et le `u` correspond à `90°` ; si on veut placer le nom d'un point entre les deux, on peut donc entrer `[60]` comme paramètre de position.

Pour terminer la création du repère, on écrit les noms des points I et J, des vecteurs $\vec{\imath}$ et $\vec{\jmath}$, ou $\vec{u}$ et $\vec{v}$ selon le cas :

```
\uput[d](1,0){$I$}                  \uput[l](0,1){$J$}
\uput[d](0.5,0){$\vec{\imath}$}     \uput[l](0,0.5){$\vec{\jmath}$}
\uput[d](0.5,0){$\vec{u}$}          \uput[l](0,0.5){$\vec{v}$}
```

Quand on met une flèche sur une lettre `i`, c'est plus joli de ne pas mettre le point sur le `i` ; on utilise donc `\imath` et `\jmath` pour écrire $\vec{\imath}$ et $\vec{\jmath}$.

Naturellement, si on a des repères $(O\,;\vec{\imath},\vec{\jmath})$ ou $(O\,;\vec{u},\vec{v})$, il faut tracer les vecteurs ; on le fera avec l'instruction `\psaxes` de la façon suivante : `\psaxes[linewidth=1.8pt]{->}(0,0)(1,1)`
L'instruction `linewidth=1.8pt` va donner une épaisseur de 1,8 point aux dessins des deux vecteurs.

Une remarque à propos de l'écriture des noms des points ; j'ai choisi de les écrire en majuscule (c'est normal) et en italique (ça l'est peut-être moins). C'est un choix ! On peut les écrire en romain...

Voici donc ce qu'il faut entrer pour un repère $(O\,;\vec{\imath},\vec{\jmath})$, où $x \in [-3\,,4]$ et $y \in [-5\,,6]$, et avec quadrillage au demi-centimètre :

```
\psset{xunit=1cm, yunit=1cm, runit=1cm}
\begin{pspicture}(-3,-5)(4,6)
\psgrid[subgriddiv=2, gridlabels=0, gridcolor=gray, subgridcolor=orange]
\psaxes[ticksize=-2pt 2pt, labels=none](0,0)(-3,-5)(4,6)
\uput[dl](0,0){$O$}
\psaxes[linewidth=1.8pt]{->}(0,0)(1,1)
\uput[d](0.5,0){$\vec{\imath}$}
\uput[l](0,0.5){$\vec{\jmath}$}
\end{pspicture}
```

3 Petite amélioration

Si on modifie le rectangle de départ défini par `pspicture`, il faudra également le modifier dans `\psaxes`, et même plus tard modifier les valeurs de x dans le tracé de la fonction par `\psplot`.
On peut donc dès la création du repère entrer les valeurs extrêmes de x et de y dans des variables :

```
\psset{xunit=1cm, yunit=1cm, runit=1cm}
\def\xmin {-3}    \def\xmax {4}
\def\ymin {-5}    \def\ymax {6}
\begin{pspicture}(\xmin,\ymin)(\xmax,\ymax)
\psgrid[subgriddiv=2, gridlabels=0, gridcolor=gray, subgridcolor=orange]
\psaxes[ticksize=-2pt 2pt, labels=none](0,0)(\xmin,\ymin)(\xmax,\ymax)
\uput[dl](0,0){$O$}
\psaxes[linewidth=1.8pt]{->}(0,0)(1,1)
\uput[d](0.5,0){$\vec{\imath}$}
\uput[l](0,0.5){$\vec{\jmath}$}
\end{pspicture}
```

4 Le tracé de courbes

Quand tout ceci est mis en place (une bonne fois pour toutes), on va représenter des fonctions.
Par exemple on va tracer la représentation graphique de la fonction $f : x \longmapsto 2x^2 - 5x - 1$, de la droite d d'équation $y = -2x + 1$ et on va placer leurs points d'intersection A et B.
L'écriture $f : x \longmapsto 2x^2 - 5x - 1$ s'obtient en entrant `$f: x \longmapsto 2x^2-5x-1$`.
Ceux qui préfèrent $x \overset{f}{\longmapsto} 2x^2 - 5x - 1$ entreront `$x \stackrel{f}{\longmapsto} 2x^2-5x-1$`.

L'instruction qui permet de tracer la représentation graphique d'une fonction est `\psplot` qui nécessite trois paramètres : la plus petite valeur de x (donc `\xmin`), la plus grande valeur de x (`\xmax`) et l'expression de la fonction.
Par défaut, il faut entrer l'expression de la fonction en mode postfixé, comme on faisait dans le temps sur les calculatrices HP. On écrit donc `-2 x mul 1 add` pour $-2x+1$, et si on écrit $2x^2-5x-1$ sous la forme $2x\,(x - 2,5) - 1$, on entrera `2 x x 2.5 sub mul mul 1 sub`.
Il suffit de connaître `mul`, `div`, `add` et `sub` respectivement pour la multiplication, la division, l'addition et la soustraction ; on rajoute `exp` pour la puissance et `sqrt` pour la racine carrée et on peut ainsi définir la plupart des fonctions dont on a besoin en lycée.
Ceux que cette méthode effraie peuvent entrer les fonctions sous forme algébrique en entrant `algebraic=true` comme paramètre dans `\psset` (voir page 18).
Si on veut lisser un peu plus la courbe, on peut augmenter le nombre de points à tracer par l'option `[plotpoints=1000]`.

Pour tracer la parabole, on entrera donc :
`\psplot[plotpoints=1000]{\xmin}{\xmax}{2 x x 2.5 sub mul mul 1 sub}`
et pour la droite :
`\psplot[plotpoints=1000]{\xmin}{\xmax}{-2 x mul 1 add}`
ce qui donne finalement comme code :

```
\psset{xunit=1cm, yunit=1cm, runit=1cm}
\def\xmin {-3}   \def\xmax {4}
\def\ymin {-5}   \def\ymax {6}
\begin{pspicture}(\xmin,\ymin)(\xmax,\ymax)
\psgrid[subgriddiv=2, gridlabels=0, gridcolor=gray, subgridcolor=orange]
\psaxes[ticksize=-2pt 2pt, labels=none](0,0)(\xmin,\ymin)(\xmax,\ymax)
\uput[dl](0,0){$O$}
\psaxes[linewidth=1.8pt]{->}(0,0)(1,1)
\uput[d](0.5,0){$\vec{\imath}$} \uput[l](0,0.5){$\vec{\jmath}$}
\psplot[plotpoints=1000]{\xmin}{\xmax}{2 x x 2.5 sub mul mul 1 sub}
\psplot[plotpoints=1000]{\xmin}{\xmax}{-2 x mul 1 add}
\end{pspicture}
```

Mais si on entre exactement ce code, c'est... la catastrophe !

En effet, comme $f\,(x) = 2x^2 - 5x - 1$, $f\,(\texttt{xmin}) = f\,(-3) = 32$ dépasse largement le `ymax` c'est-à-dire 6. Donc la parabole sort du rectangle défini par `pspicture`, et la droite aussi !
Une solution ? Bien sûr ! Il suffit d'utiliser l'environnement `pspicture*` qui va limiter tous les tracés au rectangle `(\xmin,\ymin)(\xmax,\ymax)`.

Enfin, comme on a défini par des variables les valeurs extrêmes de x et de y, on peut donner des noms aux fonctions que l'on utilise, surtout si on les utilise plusieurs fois, par exemple dans le cas d'un tracé d'hyperbole (page 18) ou pour tracer une aire sous une courbe.

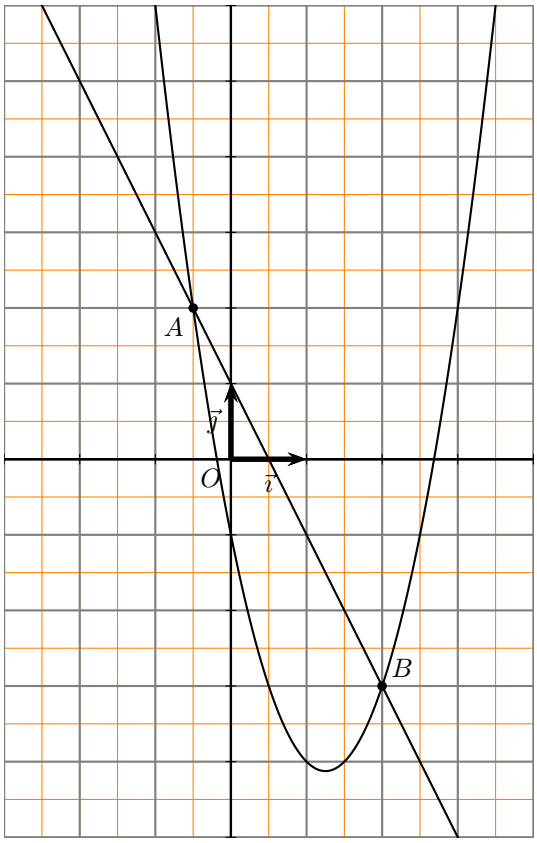
On définit donc la fonction f par `\def\f{2 x x 2.5 sub mul mul 1 sub}`,
la fonction affine par `\def\g{-2 x mul 1 add}`.

Il ne reste que les points d'intersection $A\,(-0,5\,;2)$ et $B\,(2\,;-3)$ à placer en utilisant l'instruction `\psdots` suivie des coordonnées des points : `\psdots(-0.5,2)(2,-3)`.
On écrit leurs noms avec `\uput[dl](-0.5,2){$A$}` pour A et `\uput[ur](2,-3){$B$}` pour B.

On entre donc le code suivant :

```
\psset{xunit=1cm, yunit=1cm, runit=1cm}
\def\xmin {-3}
\def\xmax {4}
\def\ymin {-5}
\def\ymax {6}
\begin{pspicture*}(\xmin,\ymin)(\xmax,\ymax)
\psgrid[subgriddiv=2, gridlabels=0, gridcolor=gray, subgridcolor=orange]
\psaxes[ticksize=-2pt 2pt, labels=none](0,0)(\xmin,\ymin)(\xmax,\ymax)
\uput[dl](0,0){$O$}
\psaxes[linewidth=1.8pt]{->}(0,0)(1,1)
\uput[d](0.5,0){$\vec{\imath}$} \uput[l](0,0.5){$\vec{\jmath}$}
\def\f{2 x x 2.5 sub mul mul 1 sub}
\def\g{ -2 x mul 1 add}
\psplot[plotpoints=1000]{\xmin}{\xmax}{\f}
\psplot[plotpoints=1000]{\xmin}{\xmax}{\g}
\psdots(-0.5,2)(2,-3)
\uput[dl](-0.5,2){$A$}
\uput[ur](2,-3){$B$}
\end{pspicture*}
```

Ce qui donne :

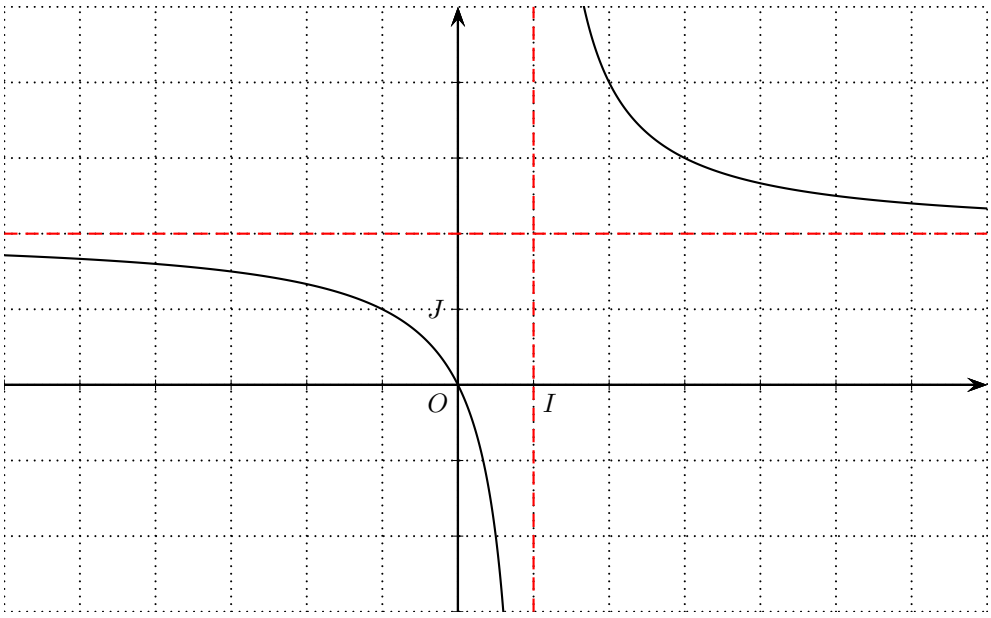


5 Autre exemple

Si on représente une fonction rationnelle, on peut avoir un problème d'ensemble de définition. Voici un exemple de tracé d'une hyperbole représentant la fonction f définie sur $\mathbf{R} \setminus \{1\}$ par $f(x) = \dfrac{2x}{x-1}$, et de ses deux asymptotes.

```
\psset{xunit=1cm, yunit=1cm, runit=1cm, algebraic=true}
\def\xmin {-6}   \def\xmax {7}
\def\ymin {-3}   \def\ymax {5}
\begin{pspicture*}(\xmin,\ymin)(\xmax,\ymax)
\psgrid[subgriddiv=1, griddots=10, gridlabels=0, gridcolor=black]
\psaxes[arrowsize=3pt 3, ticksize=-2pt 2pt, labels=none]{->}(0,0)
  (\xmin,\ymin)(\xmax,\ymax) % à écrire à la suite de la ligne précédente
\uput[dl](0,0){$O$}
%\psaxes[ linewidth=1.8pt]{->}(0,0)(1,1)
%\uput[d](0.5,0){$\vec{\imath}$} \uput[l](0,0.5){$\vec{\jmath}$}
\uput[dr](1,0){$I$} \uput[l](0,1){$J$}
\def\f{2*x/(x-1)}                              % définition de la fonction
\psplot[plotpoints=1000]{\xmin}{0.99}{\f}   % une branche de l'hyperbole
\psplot[plotpoints=1000]{1.01}{\xmax}{\f}   % l'autre branche
\psline[linestyle=dashed, linecolor=red](\xmin,2)(\xmax,2)    % y=2
\psline[linestyle=dashed, linecolor=red](1,\ymin)(1,\ymax)    % x=1
\end{pspicture*}
```

Au passage, on peut voir les tracés des deux branches de l'hyperbole, le `griddots=10` qui trace la grille en pointillés avec 10 points par division (donc par centimètre), et le `linestyle=dashed` qui trace les asymptotes en mode « tirets » ; pour des pointillés, il faudrait écrire `linestyle=dotted`. L'expression `algebraic=true` dans `\psset` permet de ne pas utiliser la notation postfixée pour entrer la fonction.
Enfin tout ce qui se trouve après un signe `%` est un commentaire qui n'est pas pris en compte.



À vous de jouer !