

# Documentation de `repere.mp`

Olivier PÉAULT\*

17 décembre 2013

## Table des matières

<b>1</b>	<b>Utilisation du fichier</b>	<b>1</b>	<b>6</b>	<b>Surfaces</b>	<b>13</b>
			6.1	Calcul intégral . . . . .	13
<b>2</b>	<b>Repère utilisateur</b>	<b>2</b>	6.2	Demi-plans . . . . .	14
2.1	Numérotation des figures . . . . .	2	6.3	Remplissage . . . . .	15
2.2	Définition du repère . . . . .	2	<b>7</b>	<b>Projections sur les axes</b>	<b>16</b>
2.3	Axes . . . . .	2	7.1	Projetés . . . . .	16
2.4	Quadrillages . . . . .	6	7.2	Intervalles . . . . .	17
2.5	Base . . . . .	7	<b>8</b>	<b>Statistiques</b>	<b>17</b>
<b>3</b>	<b>Points, vecteurs</b>	<b>8</b>	8.1	Boite à moustache . . . . .	17
3.1	Points . . . . .	8	<b>9</b>	<b>Divers</b>	<b>18</b>
3.2	Vecteurs . . . . .	9	9.1	Composition des étiquettes . . . . .	18
<b>4</b>	<b>Droites, courbes...</b>	<b>9</b>	9.2	Couleurs . . . . .	18
4.1	Droites . . . . .	9	9.3	Figures pour une présentation . . . . .	19
4.2	Courbes et fonctions . . . . .	10	9.4	Code embarqué dans un document L <sup>A</sup> T <sub>E</sub> X . . . . .	20
4.3	Dérivée et tangentes . . . . .	11	<b>10</b>	<b>Exemples d'utilisation simultanée de <code>repere</code> et <code>geometriesyr</code></b>	<b>20</b>
<b>5</b>	<b>Suites</b>	<b>12</b>			

## 1 Utilisation du fichier

Les macros du fichier `repere.mp` ont pour but de simplifier la création de figures dans un repère du plan avec METAPOST. L'idée de départ est de coller le plus possibles aux besoins de l'enseignement secondaire de mathématiques.

Il est possible d'utiliser `repere` et `geometriesyr` (les macros de Christophe POULAIN pour la géométrie disponibles à l'adresse <http://melusine.eu.org/syracuse/poulecl/macros/>) dans une même figure comme le montrent les exemples page 20.

Le fichier `repere.mp` doit être placé dans un répertoire accessible à METAPOST (Par ex. le répertoire `metapost` du `texmf` local). De plus, la ligne `input repere;` doit apparaître dans le document contenant les figures.

Certaines des étiquettes (noms de points, de courbes, de vecteurs...) sont composées automatiquement au format L<sup>A</sup>T<sub>E</sub>X mais la compilation peut devenir assez longue pour un nombre important de figures car un fichier est créé et compilé automatiquement par METAPOST pour chaque étiquette. Il est alors possible d'utiliser le fichier `latexmp.mp` à condition de placer l'instruction `input latexmp;` avant l'instruction `input repere;`. Attention, utiliser `latexmp.mp` oblige à compiler deux fois les documents.

---

\*E-mail : [o.peault@gmail.com](mailto:o.peault@gmail.com)

## 2 Repère utilisateur

### 2.1 Numérotation des figures

Chaque figure devra débuter par une instruction `repere()` et se terminer par `fin` (voir ci-dessous). Si ces instructions se trouvent en dehors d'un environnement `beginfig()-endfig` la numérotation est automatique :

```
beginfig(2);  
repere(...);  
<instructions de dessin>  
fin;  
endfig;
```

La figure porte le numéro 2

```
repere(...);  
<instructions de dessin>  
fin;
```

La numérotation est automatique. La figure porte le numéro qui suit la figure précédemment dessinée. S'il s'agit de la première, elle porte le numéro 1.

### 2.2 Définition du repère

`repere(xmin,xmax,ux,ymin,ymax,uy,theta)` débute une figure et définit le repère utilisateur : axe des abscisses de `xmin` à `xmax`, unité `ux`, axe des ordonnées de `ymin` à `ymax`, unité `uy` et `theta` est l'angle en degrés entre les axes. Le paramètre `theta` est optionnel. Il est égal à 90 par défaut.

`repere.larg(xmin,xmax,Lx,ymin,ymax,Ly,theta)` définit un repère tel que la largeur totale de la figure produite soit `Lx` et sa hauteur `Ly`.

`repere.orth(xmin,xmax,Lx,ymin,ymax)` définit un repère orthonormé de largeur totale `Lx`.

`interaxes(x,y)` définit les coordonnées du point d'intersection des axes. Par défaut ces coordonnées sont (0,0).

`cadre` chemin fermé qui fait le tour du repère.

`fin` termine la figure et la découpe pour ne garder que la partie limitée par le repère utilisateur.

### 2.3 Axes

#### 2.3.1 Généralités

`axex.pos(grad,val)` axe des abscisses gradué avec un pas de `grad` et étiqueté avec un pas de `val`.

Si `grad` est négatif ou nul, l'axe n'est pas gradué et si `val` est négatif ou nul, l'axe n'est pas étiqueté.

`pos` est un paramètre optionnel qui désigne la position (au sens de METAPOST : `rt`, `urt`, `top`, `ulft`, `lft`, `llft`, `bot` ou `lrt`) des étiquettes. `pos` peut être omis, la valeur par défaut est `bot`.

Les étiquettes qui ne sont pas entièrement à l'intérieur du cadre ne sont pas dessinées.

`axey.pos(grad,val)` axe des ordonnées. La valeur par défaut de `pos` est `lft`.

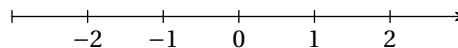
Au niveau de l'intersection des axes, les étiquettes sont tracées à la position `pos` si l'abscisse est différente de l'ordonnée ou si un seul axe est tracé. Dans le cas contraire, une seule étiquette est tracée pour les deux axes à une position « intermédiaire » (pour `axex.bot` et `axey.lft`, on obtient la position `llft`)

`axes.pos(grad,val)` figure formée par les deux axes gradués avec le même pas `grad` et étiquetés avec le même pas `val`. `pos` désigne la position de l'étiquette de l'intersection des axes, sa valeur par défaut est `llft`. La position des étiquettes des axes est définie à partir de `pos` (pour `urt` on obtient `top` pour l'axe des abscisses et `rt` pour l'axe des ordonnées).

Les axes sont dessinés, gradués et étiquetés par défaut sur toute la longueur du repère utilisateur.  
 Pour des valeurs différentes on peut utiliser les macros suivantes :

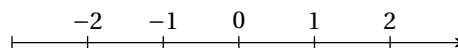
### Exemple 1

```
repere(-3,3,1cm,-1,1,1cm);
draw axex(1,1);
fin;
```



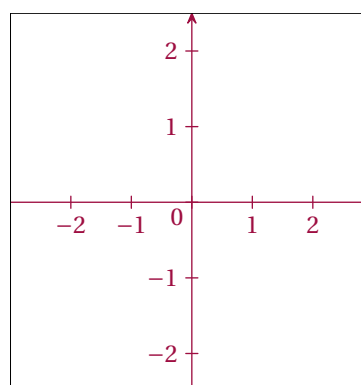
### Exemple 2

```
repere(-3,3,1cm,-1,1,1cm);
draw axex.top(1,1);
fin;
```



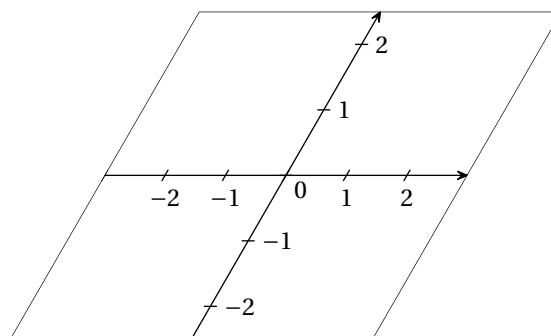
### Exemple 3

```
repere(-3,3,0.8cm,-2.5,2.5,1cm);
draw axes(1,1) withcolor pourpre;
draw cadre;
fin;
```



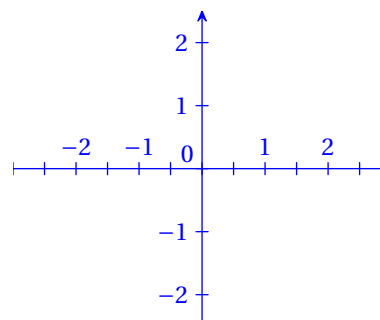
### Exemple 4

```
repere(-3,3,0.8cm,-2.5,2.5,1cm,60);
draw axes.lrt(1,1);
draw cadre;
fin;
```



### Exemple 5

```
repere.orth(-3,3,5cm,-2.5,2.5);
draw axex.top(0.5,1) withcolor bleu;
draw axey(1,1) withcolor bleu;
fin;
```



### 2.3.2 Réglages des axes

Les axes sont dessinés, gradués et étiquetés par défaut sur toute la longueur du repère utilisateur. Pour des valeurs différentes on peut utiliser les macros suivantes :

`setaxes(xmin,xmax,ymin,ymax)` définit les valeurs minimales et maximales pour les axes.

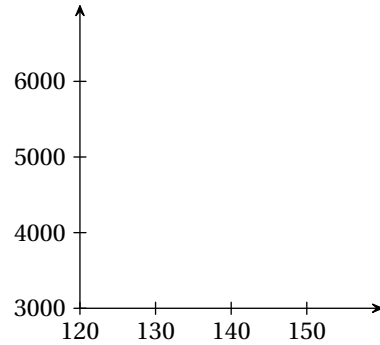
`setgrad(xmin,xmax,ymin,ymax)` définit les valeurs minimales et maximales pour les graduations.

`setval(xmin,xmax,ymin,ymax)` définit les valeurs minimales et maximales pour l'étiquetage.

`flecheaxe` booléen égal à `true` par défaut qui permet de dessiner, ou non, des flèches au bout des axes.

#### Exemple 6

```
repere.larg(110,160,5cm,2000,7000,5cm);
interaxes(120,3000);
setaxes(120,160,3000,7000);
setgrad(120,160,3000,7000);
draw axex(10,10);
draw axey(1000,1000);
fin;
```

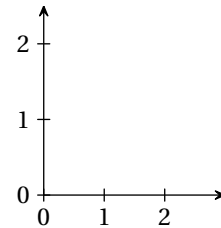


`axexo.pos(grad,val)`, `axeyo.pos(grad,val)`, `axeso.pos(grad,val)` macros identiques aux précédentes sauf pour l'étiquette correspondant à l'intersection des axes qui est toujours dessinée.

`axexn.pos(grad,val)`, `axeyn.pos(grad,val)`, `axesn.pos(grad,val)` macros identiques aux précédentes sauf pour l'étiquette correspondant à l'intersection des axes qui n'est jamais dessinée.

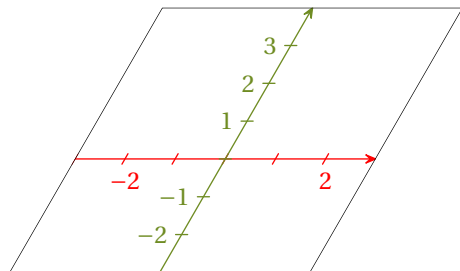
#### Exemple 7

```
repere(-0.5,3,0.8cm,-0.5,2.5,1cm);
setaxes(0,3,0,2.5);
draw axeso(1,1);
fin;
```



#### Exemple 8

```
repere.larg(-3,3,6cm,-3,4,3.5cm,60);
draw axexn(1,2) withcolor rouge;
draw axeyn(1,1) withcolor olive;
draw cadre;
fin;
```



### 2.3.3 Graduations multiples de $\pi$

`axexpi.pos(n,d)` axe des abscisses gradué et étiqueté avec un pas de  $\frac{n\pi}{d}$ . Les fractions sont composées en mode normal. Pour les obtenir en mode « `displaystyle` », la variable de type booléen `displayfrac` doit être égale à `true`.

`axeypi.pos(n,d)` axe des ordonnées gradué et étiqueté avec un pas de  $\frac{n\pi}{d}$ .

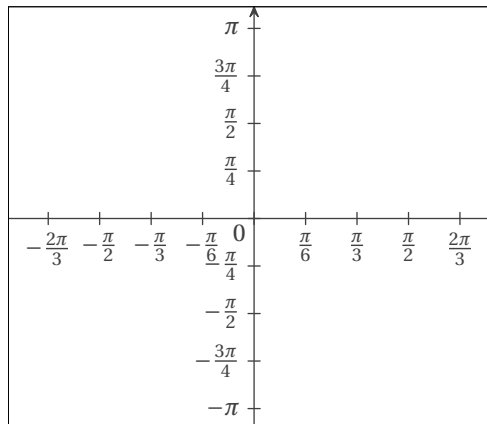
`axespi.pos(n,d)` les deux axes gradués et étiquetés avec un pas de  $\frac{n\pi}{d}$ .

`axexpio.pos(n,d)`, `axeypio.pos(n,d)`, `axespio.pos(n,d)` même chose que précédemment sauf pour l'étiquette correspondant à l'intersection des axes qui est toujours dessinée.

`axexpin.pos(n,d)`, `axeypin.pos(n,d)`, `axespin.pos(n,d)` même chose que précédemment sauf pour l'étiquette correspondant à l'intersection des axes qui n'est jamais dessinée.

### Exemple 9

```
repere(-2.5,2.5,1.3cm,-3.5,3.5,0.8cm);
draw axexpi(1,6) withcolor grisfonce;
draw axeypi(1,4) withcolor grisfonce;
draw cadre;
fin;
```



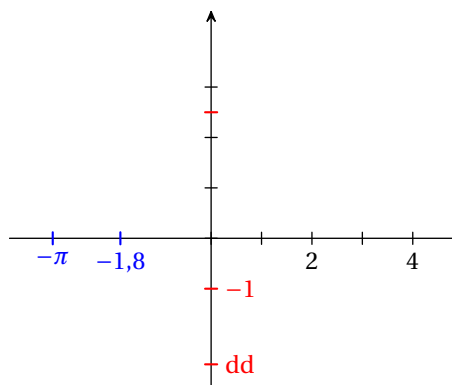
### 2.3.4 Graduations isolées

`axexpart.pos(x1,lab1,x2,lab2,...)` graduation et étiquetage partiels de l'axe des abscisses pour les valeurs  $x_1, x_2, \dots$  et les étiquettes  $lab_1, lab_2, \dots$  à la position `pos`. Si `pos` est omis, les étiquettes sont placées à la position `bot`. Les étiquettes peuvent être soit des chaînes de caractères ("aa", "bonjour"), soit des expressions du type `bTeX  $\pi$  eTeX` (ou `LaTeX("$\pi$")` voir page 18), soit d'autres figures. Si `labn` est omis, la valeur de  $x_n$  sera utilisée comme étiquette. Pour obtenir une graduation sans étiquette, on peut utiliser la chaîne vide "".

`axeypart.pos(y1,lab1,y2,lab2,...)` même chose sur l'axe des ordonnées. Si `pos` est omis, les étiquettes sont placées à la position `lft`.

### Exemple 10

```
repere.orth(-4,5,6cm,-3,4.5);
setgrad(0,5,0,3);
setval(0,5,0,3);
draw axex(1,2);
draw axey(1,0);
draw axexpart(-1.8,-pi,LaTeX("$-\pi$"))
                                withcolor bleu;
draw axeypart.rt(-2.5,"dd",-1,2.5,"")
                                withcolor rouge;
fin;
```



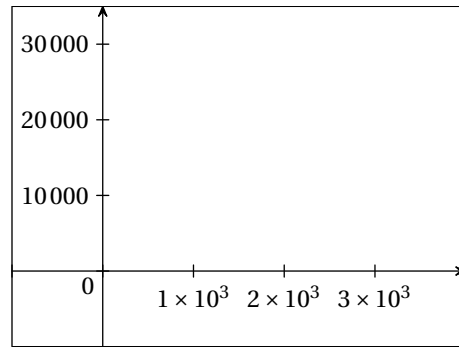
### 2.3.5 Ajout de texte sur les graduations

`extranumx` chaîne de caractères qui sera ajoutée après les valeurs des graduations sur l'axe des abscisses avant d'être composée avec la commande `\num` de `siunitx`.

`extranumy` même chose sur l'axe des ordonnées.

## Exemple 11

```
repere(-1,4,1.2cm,-1,3.5,1cm);
extranumx:="e3";
extranumy:="0000";
draw axes(1,1);
draw cadre;
fin;
```



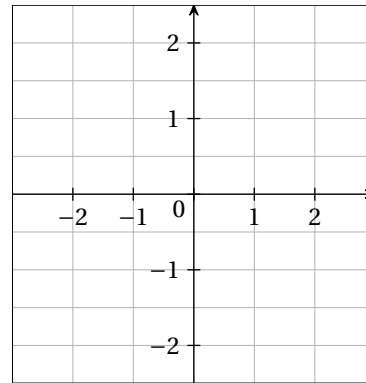
## 2.4 Quadrillages

`setquad(xmin,xmax,ymin,ymax)` définit les valeurs minimales et maximales pour le tracé des quadrillages.<sup>1</sup>

`quadrillage(x,y)` quadrillage avec un pas de  $x$  sur l'axe des abscisses et de  $y$  sur l'axe des ordonnées. L'épaisseur des traits par défaut est `0.3bp` et la couleur par défaut est `0.7white`.

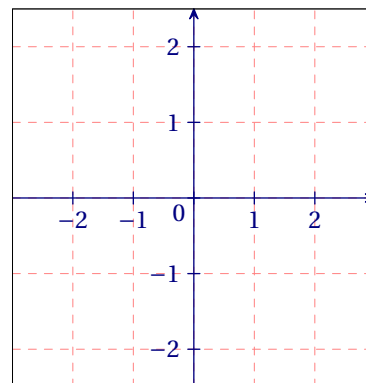
## Exemple 12

```
repere(-3,3,0.8cm,-2.5,2.5,1cm);
draw quadrillage(1,0.5);
draw axes(1,1);
draw cadre;
fin;
```



## Exemple 13

```
repere(-3,3,0.8cm,-2.5,2.5,1cm);
draw quadrillage(1,1) dashed evenly
                    withcolor (1,0.5,0.5);
draw axes(1,1) withcolor marine;
draw cadre;
fin;
```

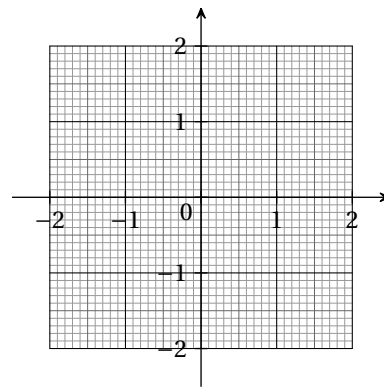


`papiermillimetre` comme son nom l'indique... Les couleurs et les épaisseurs des traits sont stockées dans `pm_coula`, `pm_coulb`, `pm_coulc` et `pm_epa`, `pm_epb`, `pm_epc`.

1. Il existe une macro `settout` qui appelle successivement `setaxes`, `setgrad`, `setval` et `setquad`.

### Exemple 14

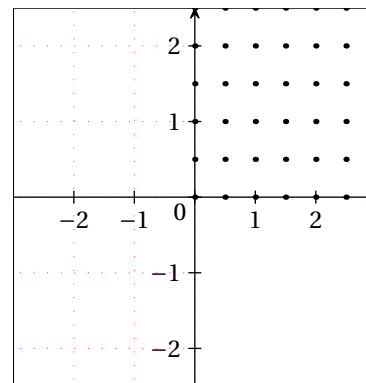
```
repere(-2.5,2.5,1cm,-2.5,2.5,1cm);
setquad(-2,2,-2,2);
draw papiermillimetre;
draw axes(1,1);
fin;
```



**papierpointe(x,y)** quadrillage formé de points avec un pas de  $x$  sur l'axe des abscisses et de  $y$  sur l'axe des ordonnées. La taille des points par défaut est 2bp.

### Exemple 15

```
repere(-3,3,0.8cm,-2.5,2.5,1cm);
setquad(0,3,0,2.5);
draw papierpointe(0.5,0.5);
setquad(-3,0,-2.5,2.5);
draw quadrillage(1,1) dashed
      withdots withcolor magenta;
draw axes(1,1);
draw cadre;
fin;
```



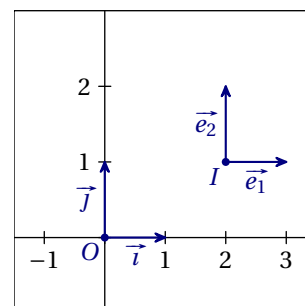
## 2.5 Base

**base(0,i,j)** figure formée par le point d'intersection des axes et son nom (0), ainsi que des deux vecteurs de la base et leurs noms ( $i$  et  $j$  surmontés d'une flèche). Si les noms sont de la forme « lettre + nombre », le nombre est affiché en indice.

**basep(0,I,J)** figure formée par le point d'intersection des axes et son nom (0), ainsi que des deux points qui définissent le repère.

### Exemple 16

```
repere(-1.5,3.5,0.8cm,-1,3,1cm);
flecheaxe:=false;
draw axesn(1,1);
drawoptions(withcolor marine);
draw base(0,i,j);
interaxes(2,1);
draw base(I,e1,e2);
drawoptions();
draw cadre;
fin;
```

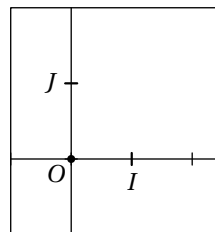


### Exemple 17

```

repere(-1,2.5,0.8cm,-1,2,1cm);
flecheaxe:=false;
draw axes(1,0);
draw basep(0,I,J);
draw cadre;
fin;

```



## 3 Points, vecteurs

### 3.1 Points

`_c(x,y)` désigne le point (ou le vecteur) de coordonnées cartésiennes  $x$  et  $y$  dans le repère utilisateur.

`pol(r,t)` désigne le point (ou le vecteur) de coordonnées  $(r \cos t; r \sin t)$  dans le repère utilisateur.

`pol(r,t)` même chose avec l'angle donné en degrés.

Les macros suivantes sont directement inspirées des macros similaires de `geometriesyr16.mp`.

`MarquePoint(A)` marque le point A. Le style de marque est contrôlé par le paramètre `marque_p` qui peut prendre les valeurs "plein" (valeur par défaut), "creux" ou "croix". Une autre valeur que celles-ci ne produira aucune marque.

`pointe(A,B,C...)` permet de marquer plusieurs points.

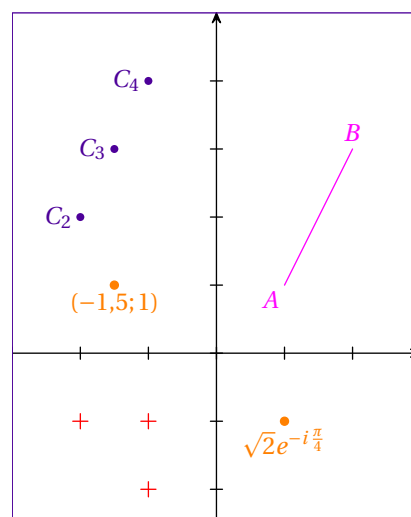
`nomme.pos(A,nom)` marque le point et affiche son nom à la position `pos`. `nom` peut être soit une chaîne de caractères, soit une expression du type `btex ... etex`, soit une autre figure. Si `nom` est omis, le nom A est affiché. S'il s'agit d'un élément d'un tableau de points (A1, A2...), le nombre est affiché en indice.

### Exemple 18

```

repere(-3,3,0.9cm,-2.5,5,0.9cm);
pair A,B,C[],D,E,F;
A=_c(1,1);B=_c(2,3);
D=_c(-2,-1);E=_c(-1,-1);F=_c(-1,-2);
draw axes(1,0);
marque_p="";drawoptions(withcolor magenta);
nomme.llft(A);nomme.top(B);draw A--B;
marque_p="croix";drawoptions(withcolor rouge);
pointe(D,E,F);
marque_p="creux";drawoptions(withcolor orange);
nomme.bot(pol(sqrt(2),-pi/4),
    LaTeX("$\sqrt{2}e^{-i\frac{\pi}{4}}$"));
nomme.bot(_c(-1.5,1),LaTeX("$(-1,5;1)$"));
marque_p="plein";drawoptions(withcolor violet);
for i=2 upto 4:
    C[i]=_c(-3+i/2,i);nomme.lft(C[i]);
endfor
draw cadre;
fin;

```





## 3.2 Vecteurs

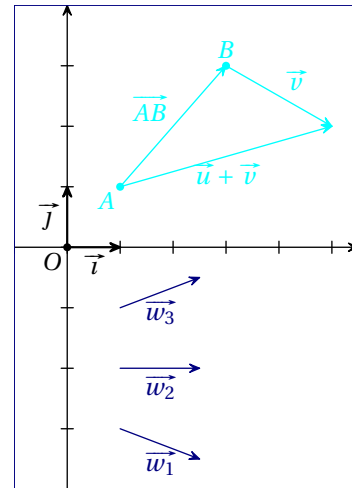
`vecteur.pos(A,u,nom)` figure formée du représentant du vecteur  $u$  d'origine  $A$  ainsi que de  $nom$  placé à la position `pos` par rapport au milieu de la flèche. Si `nom` est une chaîne de caractère, il sera affiché avec une flèche. Si `nom` est omis,  $u$  surmonté d'une flèche est utilisé. S'il s'agit d'un élément d'un tableau de points ( $u_1, u_2\dots$ ), le nombre est affiché en indice.

### Exemple 19

```

repere(-1,5.5,0.7cm,-4,4,0.8cm);
pair A,B,C[],u,v,w[];
u=_c(2,2);v=_c(2,-1);
A=_c(1,1);B=A+u;
draw axes(1,0);
draw base(0,i,j);
drawoptions(withcolor cyan);
nomme.llft(A);nomme.top(B);
draw vecteur.ulft(A,u,"AB");
draw vecteur.urft(B,v);
draw vecteur.bot(A,u+v,%
    LaTeX("\vect{u}+\vect{v}"));
drawoptions(withcolor marine);
for i=1 upto 3:
  C[i]=_c(1,i-4);w[i]=_c(1.5,0.5*i-1);
draw vecteur.bot(C[i],w[i]);
endfor
draw cadre;
fin;

```



## 4 Droites, courbes...

### 4.1 Droites

`droite(A,B)` droite  $(AB)$ .

`droite(a,b,c)` droite d'équation  $ax + by + c = 0$  dans le repère utilisateur.

`droite(a,b)` droite d'équation  $y = ax + b$  dans le repère utilisateur.

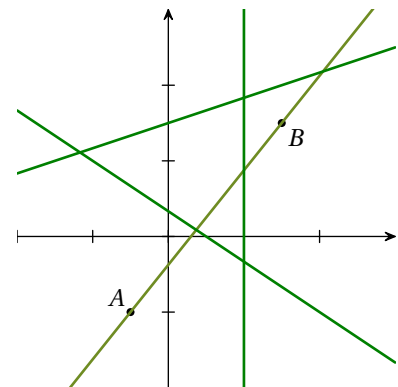
`droite(c)` droite d'équation  $x = c$  dans le repère utilisateur.

### Exemple 20

```

repere(-2,3,1cm,-2,3,1cm);
pair A,B;
A=_c(-0.5,-1);B=_c(1.5,1.5);
draw axes(1,0);
nomme.ulft(A);
nomme.lrt(B);
drawoptions(withpen pencircle scaled 1);
draw droite(A,B) withcolor olive;
draw droite(1) withcolor vertfonce; %x=1
draw droite(1/3,3/2) withcolor vertfonce; %y=(1/3)x+3/2
draw droite(2,3,-1) withcolor vertfonce; %2x+3y-1=0
fin;

```



## 4.2 Courbes et fonctions

METAPOST permet de définir simplement des fonctions (en utilisant par exemple la syntaxe suivante : `vardef f(expr x)=2x+1 enddef;`) et de définir des courbes passant par des points donnés (A..B..C). Ces possibilités sont utilisées dans les macros qui suivent.

`courbefonc(f,xmin,xmax,n)` ou `courbefonc(f)(xmin,xmax,n)` courbe représentant la fonction `f` entre `xmin` et `xmax` en utilisant `n` points d'interpolation.

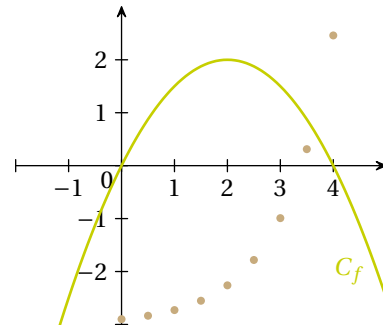
`courbepoints(f,xmin,xmax,n)` ne trace que les `n` points sans les relier. Les points sont dessinés en fonction de la valeur de `marque_p` (voir 3.1).

`fonccourbe.p(x)` image de `x` par la fonction dont la courbe représentative est le chemin `p`. La macro renvoie 0 si la fonction n'est pas définie.

`nomme.pos(p,x,nom)` affiche `nom` au point d'abscisse `x` de la courbe `p` à la position `pos`. `nom` peut être soit une chaîne de caractères, soit une expression du type `btex ... etex`, soit une autre figure. Si `nom` est omis, le nom `p` est affiché. S'il s'agit d'un élément d'un tableau de points (`p1, p2...`), le nombre est affiché en indice.

### Exemple 21

```
repere(-2,5,0.7cm,-3,3,0.7cm);
vardef f(expr x)=-0.5(x**2)+2*x enddef;
vardef g(expr x)=exp(x)/10-3 enddef;
path C_f;
draw axes(1,1);
drawoptions(withcolor moutarde);
C_f=courbefonc(f)(-2,5,50);
draw C_f withpen pencircle scaled 1;
nomme.llft(C_f,4.7);
drawoptions(withcolor beige);
draw courbepoints(g,0,4,9);
fin;
```



`intercourbes(P,p,q)` stocke dans le tableau de points `P` les points d'intersection des chemins `p` et `q`. `P1` est un des points d'intersection, `P2` un autre etc. Il faut, avant d'utiliser cette macro, déclarer le tableau `P` de la façon suivante : `pair P[];`

`ptantecedents(P,y,p)` stocke dans le tableau de points `P` les points du chemin `p` d'ordonnée `y`. De même que précédemment, le tableau `P` doit être déclaré avant d'utiliser cette macro.

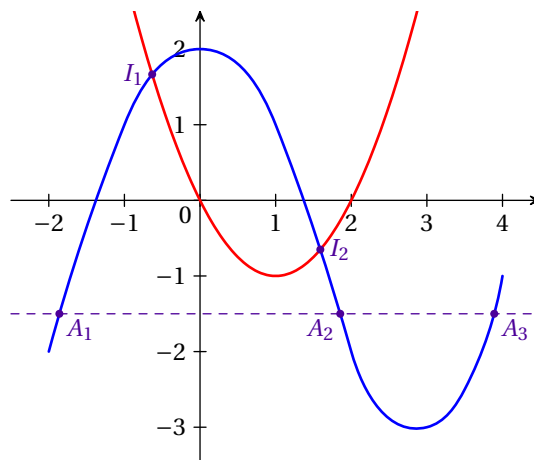
`antecedents(X,y,p)` stocke dans le tableau de nombres `X` les antécédents de `y` par la fonction dont la courbe représentative est le chemin `p`. De même que précédemment, le tableau `X` doit être déclaré avant d'utiliser cette macro.

### Exemple 22

```

repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path p,C_f;
pair I[],A[];
vardef f(expr x)= x**2-2x enddef;
p=_c(-2,-2).._c(-1,1).._c(0,2).._c(1,1)
  .._c(2,-2).._c(3,-3).._c(3.5,-2.5)
  .._c(4,-1);
C_f=courbefonc(f,-2,4,40);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw p withcolor bleu;
draw C_f withcolor rouge;
intercourbes(I,C_f,p);
drawoptions(withcolor violet);
nomme.lft(I1);nomme.rft(I2);
draw droite(0,-1.5) dashed evenly;
ptantecedents(A,-1.5,p);
nomme.lrt(A1);nomme.llft(A2);
nomme.lrt(A3);
fin;

```



**marquepointcourbe(p,x1,x2,...)** marque les points de la courbe p d'abscisses x1, x2... La marque dépend de la valeur de `marque_p`.

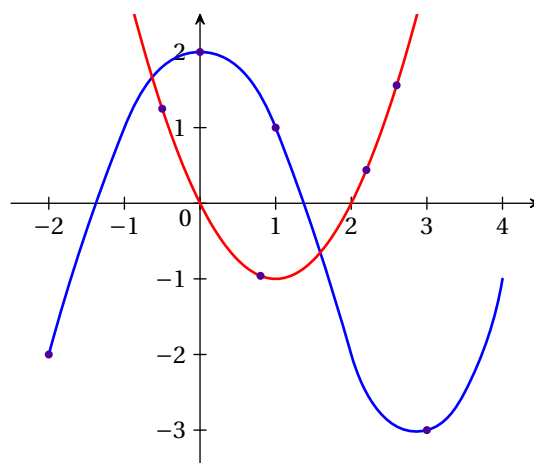
**marquepointchemin(p,n1,n2,...)** dans le cas d'un chemin défini par A..B..C..., marque le n1ème point, le n2ième point... La marque dépend de la valeur de `marque_p`. Attention, le premier point est numéroté 0.

### Exemple 23

```

repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path p,C_f;
pair I[],A[];
vardef f(expr x)= x**2-2x enddef;
p=_c(-2,-2).._c(-1,1).._c(0,2)
  .._c(1,1).._c(2,-2).._c(3,-3)
  .._c(3.5,-2.5).._c(4,-1);
C_f=courbefonc(f,-2,4,40);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw p withcolor bleu;
draw C_f withcolor rouge;
drawoptions(withcolor violet);
marquepointcourbe(C_f,-0.5,0.8,2.2,2.6);
marquepointchemin(p,0,2,3,5);
fin;

```



## 4.3 Dérivée et tangentes

**der.p(x)** image de x par la dérivée de la fonction dont la courbe représentative est p.

**tangente(p,x)** tangente à la courbe p au point d'abscisse x.

**tangente.gauche(p,x,long)** flèche de longueur `long` représentant la demi-tangente gauche à la courbe `p` au point d'abscisse `x`. Le paramètre `long` est optionnel. Sa valeur par défaut est 20bp.

**tangente.droite(p,x,long)** idem à droite.

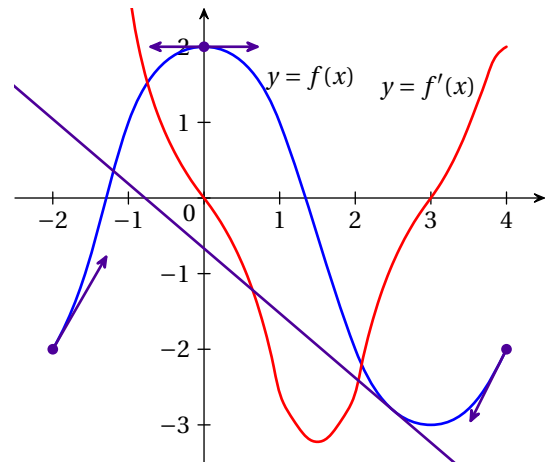
**tangente.double(p,x,long)** idem des deux côtés.

### Exemple 24

```

repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path p,q;
p=_c(-2,-2){dir 60}.._c(-1,1)
  .._c(0,2){right}.._c(1,1).._c(2,-2)
  .._c(3,-3){right}.._c(4,-2){(1,2)};
q=courbefonc(der.p,-1,4,40);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw p withcolor bleu;
nomme.rt(p,0.7,LaTeX("$y=f(x)$"));
draw q withcolor rouge;
nomme.lft(q,3.7,LaTeX("$y=f'(x)$"));
drawoptions(withpen pencircle scaled 1
  withcolor violet);
draw tangente.double(p,0);
draw tangente.droite(p,-2,40);
draw tangente.gauche(p,4,30);
draw tangente(p,2.5);
fin;

```



## 5 Suites

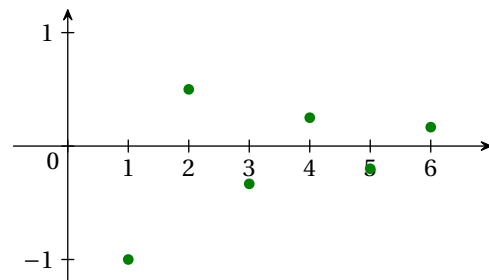
**suite(u,deb,fin)** figure formée des points  $(i; u_i)$  pour  $i$  variant entre `deb` et `fin`.

### Exemple 25

```

repere(-0.9,7,0.8cm,-1.2,1.2,1.5cm);
vardef u(expr n)=(-1)**n/n enddef;
taillepoint:=4;
draw axes(1,1);
draw suite(u,1,6) withcolor vertfonce;
fin;

```



**suiterec(f,deb,fin,init)** ligne brisée (« escalier » ou « escargot ») permettant de visualiser les termes de la suite définie par  $u_{n+1} = f(u_n)$  de premier terme  $u_{deb} = init$  et de dernier terme  $u_{fin}$ .

**suiterecprojx.pos(lab,min,max)** figure formée des segments joignant les points  $(u_n; u_n)$  et  $(u_n; 0)$  pour  $n$  compris entre `min` et `max`. La suite  $u$  et sa valeur initiale sont définies par le dernier appel de la macro `suiterec`. `lab` désigne l'étiquette au niveau de l'axe des abscisse placée à la position `pos`. Si `lab` est la chaîne vide "", rien n'est écrit; si `lab` est une autre chaîne de caractère (par ex. "u"), elle est utilisée comme nom de la suite (on obtiendra  $u_0, u_1, \dots$ ); si `lab` est un nombre, les valeurs de la suites seront affichées et arrondies à `lab` décimales. Les valeurs `min` et `max` sont facultatives et égales par défaut aux valeurs `deb` et `fin` passées à la macro `suiterec`.

`suiterecproj.pos(lab,min,max)` même chose sur l'axe des ordonnées.

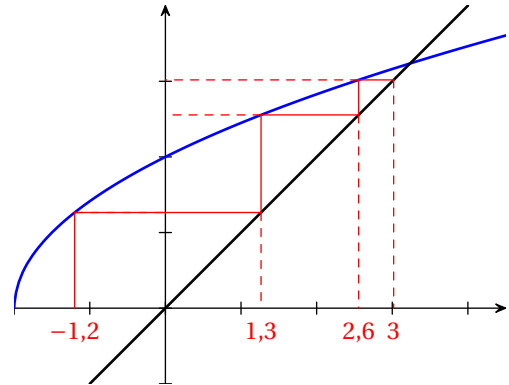
`suiterecproj(lab,min,max)` même chose sur les deux axes. Les positions sont `bot` sur l'axe des abscisses et `lft` sur l'axe des ordonnées.

### Exemple 26

```

repere(-2,4.5,1cm,-1,4,1cm);
vardef f(expr x)=sqrt(2*x+4) enddef;
path C_f,sr;
C_f=courbefonc(f,-2,5,50);
sr=suiterec(f,0,3,-1.2);
draw axes(1,0);
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor bleu;
draw droite(1,0);
drawoptions(withcolor rouge);
draw suiterecprojx.bot(1) dashed evenly;
draw suiterecprojy.lft("") dashed evenly;
draw sr withcolor rouge;
fin;

```

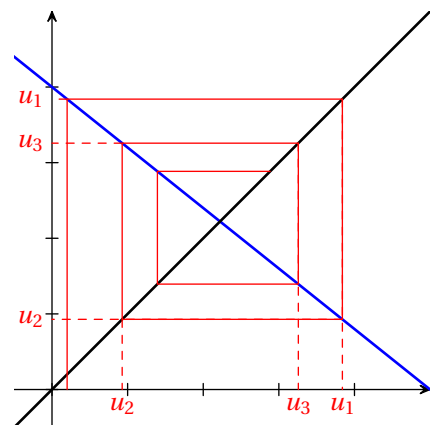


### Exemple 27

```

repere(-0.5,5,1cm,-0.5,5,1cm);
vardef f(expr x)=4-0.8*x enddef;
path C_f,sr;
C_f=courbefonc(f,-1,5,10);
sr=suiterec(f,0,5,0.2);
draw axes(1,0);
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor bleu;
draw droite(1,0);
drawoptions(withcolor rouge);
draw suiterecproj("u",1,3) dashed evenly;
draw sr withcolor rouge;
fin;

```



## 6 Surfaces

### 6.1 Calcul intégral

`entrecourbes(p,q,xmin,xmax)` chemin fermé délimitant la zone comprise entre les courbes `p` et `q` et les droites d'équations  $y = x_{\min}$  et  $y = x_{\max}$ . Il peut donc être dessiné, rempli...

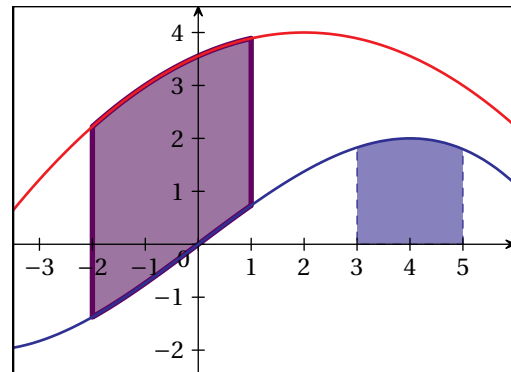
`souscourbe(p,xmin,xmax)` chemin fermé délimitant la zone comprise entre la courbe `p`, l'axe des abscisses et les droites d'équations  $y = x_{\min}$  et  $y = x_{\max}$ .

### Exemple 28

```

repere(-3.5,6,0.7cm,-2.5,4.5,0.7cm);
vardef f(expr x)= -(x/4)**3+0.75x enddef;
vardef g(expr x)= -((x-2)**2)/9+4 enddef;
path C_f,C_g,p,q;
C_f:=courbefonc(f,-3.5,6,40);
C_g:=courbefonc(g,-3.5,6,40);
p:=entrecourbes(C_f,C_g,-2,1);
q:=souscourbe(C_f,3,5);
fill p withcolor 0.5Violet;
draw p withpen pencircle scaled 2
      withcolor Violet;
fill q withcolor 0.5Bleu;
draw q dashed evenly withcolor Bleu;
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor Bleu;
draw C_g withcolor Rouge;
draw cadre;
fin;

```



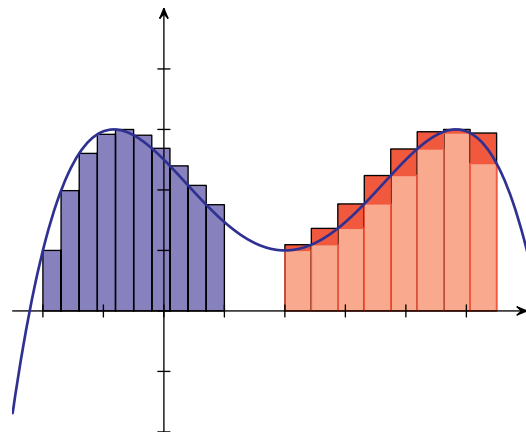
`rectangles.type(p,a,b,n)` figure formée de  $n$  rectangles s'appuyant sur la courbe  $p$  entre les abscisses  $a$  et  $b$ . `type` peut être `min`, `max`, `droite` ou `gauche`.

### Exemple 29

```

repere(-2.5,6,0.8cm,-2,5,0.8cm);
vardef f(expr x)=
  -((x-2)**4)/32+((x-2)**2)/2+1
enddef;
path Cf,r[];
Cf=courbefonc(f,-2.5,6.5,50);
r1=rectangles.max(Cf,2,5.5,8);
r2=rectangles.min(Cf,2,5.5,8);
r3=rectangles.droite(Cf,-2,1,10);
fill r1 withcolor 0.8Rouge;
fill r2 withcolor 0.4Rouge;
fill r3 withcolor 0.5Bleu;
draw r1;draw r2 withcolor 0.8Rouge;
draw r3;
draw axes(1,0);
draw Cf withcolor Bleu
      withpen pencircle scaled 1;
fin;

```



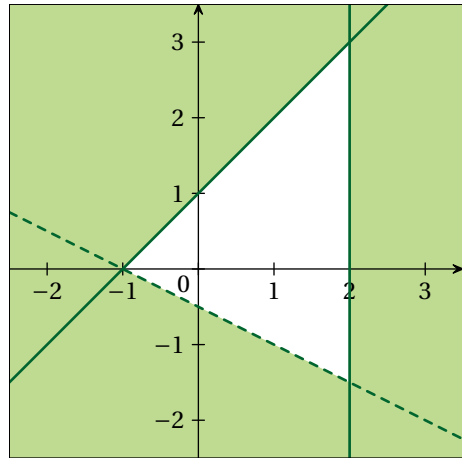
## 6.2 Demi-plans

`demiplaninf(d)` chemin fermé délimité par la droite  $d$  et par la partie inférieure de `cadre` (ou la partie gauche si  $d$  est parallèle à l'axe des ordonnées).

`demiplansup(d)` chemin fermé délimité par la droite  $d$  et par la partie supérieure de `cadre` (ou la partie droite si  $d$  est parallèle à l'axe des ordonnées).

### Exemple 30

```
repere(-2.5,3.5,1cm,-2.5,3.5,1cm);
numeric qqw;qqw=6;
path d[],dp[];
d1=droite(2);
d2=droite(1,1);
d3=droite(-0.5,-0.5);
dp1=demiplansup(d1);
dp2=demiplansup(d2);
dp3=demiplaninf(d3);
for i=1 upto 3:
fill dp[i] withcolor 0.7Lime;
endfor
draw axes(1,1);
drawoptions(withpen pencircle scaled 1
            withcolor Vertfonce);
draw d1;draw d2;
draw d3 dashed evenly;
drawoptions();
draw cadre;
fin;
```



### 6.3 Remplissage

Pour remplir des chemins fermés avec autre chose que de la couleur, **repere** permet l'utilisation de la syntaxe `fill p` avec `motif` où `motif` est un des motifs décrits ci-dessous. Cette instruction peut être complétée par des options de dessin (`withpen`, `withcolor...`).

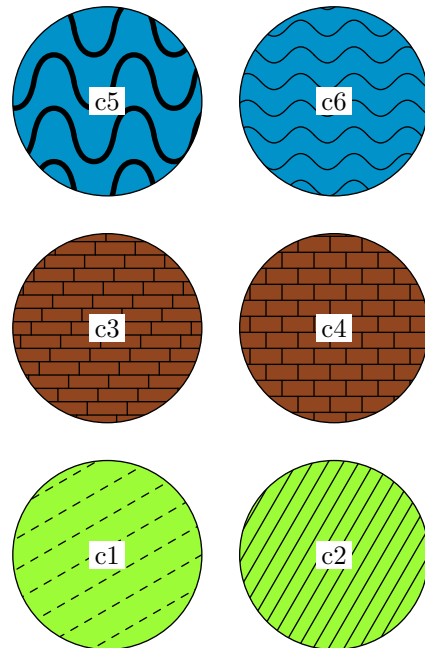
**hachures(pas,angle)** hachures espacées de `pas` et formant un angle en degrés de `angle` avec l'horizontale. Si les valeurs sont omises, `pas` vaut 5 et `angle` vaut 60.

**briques(larg,haut,dec)** briques de largeur `larg`, de hauteur `haut` et décalées d'une ligne à l'autre de `dec`. Si les valeurs sont omises, `larg` vaut 12, `haut` vaut 6 et `dec` vaut 6.

**vagues(per,amp,dec)** (d'après le manuel de l'utilisateur) « vagues » de période `per`, d'amplitude `amp` et décalées d'une ligne à l'autre de `dec`. Si les valeurs sont omises, `per` vaut 20, `amp` vaut 3 et `dec` vaut 10.

## Exemple 31

```
repere(-1.5,4.5,1cm,-1.5,7.5,1cm);
path c[];picture lab;numeric u;u:=3cm;
c1=fullcircle scaled 2.5cm;
for k=1 upto 6:
  i:=(k-1) mod 2;j:=(k-1) div 2;
  c[k]:=c1 shifted (u*i,u*j);
endfor;
fill c1 withcolor lime;
fill c1 avec hachures(10,30) dashed evenly;
fill c2 withcolor lime;
fill c2 avec hachures();
fill c3 withcolor (0,0.65,0.8,0.48);
fill c3 avec briques(15,5,4);
fill c4 withcolor (0,0.65,0.8,0.48);
fill c4 avec briques();
fill c5 withcolor (1,0,0,0.2);
fill c5 avec vagues(30,10,20)
      withpen pencircle scaled 2;
fill c6 withcolor (1,0,0,0.2);
fill c6 avec vagues();
for k=1 upto 6:
  i:=(k-1) mod 2;j:=(k-1) div 2;
  draw c[k];
  lab:=thelabel("c"&decimal(k),u*(i,j));
  unfill bbox lab;draw lab;
endfor;
fin;
```



## 7 Projections sur les axes

### 7.1 Projetés

**projetex(A)** projeté de A sur l'axe des abscisses parallèlement à l'axe des ordonnées.

**projetey(A)** projeté de A sur l'axe des ordonnées parallèlement à l'axe des abscisses.

**projectionx.pos(A,lab,dec)** figure constituée du segment joignant A à son projeté sur l'axe des abscisses ainsi que de l'étiquette **lab** placée à la position **pos** par rapport à ce projeté. La valeur **dec** indique un décalage par rapport à l'axe des abscisses. L'étiquette et le décalage sont optionnels.

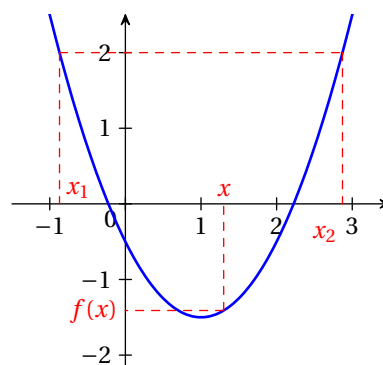
**projectiony.pos(A,lab,dec)** même chose sur l'axe des ordonnées.

**projectionaxes(A,labx,laby,dec)** figure constituée des segments joignant A à ses projetés sur les axes ainsi que des étiquettes **labx** et **laby** positionnées automatiquement avec un décalage **dec** par rapport aux axes. Les étiquettes et le décalage sont optionnels.



## Exemple 32

```
repere(-1.5,3.5,1cm,-2.2,2.5,1cm);
path Cf; pair A[];
vardef f(expr x)= x**2-2x-0.5 enddef;
Cf=courbefonc(f,-2,4,40);
ptantecedents(A,2,Cf);
draw axes(1,1);
draw Cf withpen pencircle scaled 1 withcolor bleu;
drawoptions(dashed evenly withcolor rouge);
draw projectionaxes(_c(1.3,f(1.3)),
    LaTeX("$x$"),LaTeX("$f(x)$"));
draw projectionx.urt(A1,LaTeX("$x_1$"));
draw projectionx.llft(A2,LaTeX("$x_2$"),-6);
draw A1--A2;
fin;
```



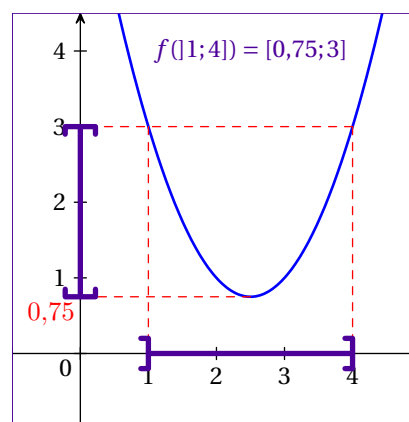
## 7.2 Intervalles

`intervallex.bornes(a,b)` intervalle dessiné sur l'axe des abscisses entre  $a$  et  $b$  avec une épaisseur par défaut de 1.5bp. `bornes` peut être OO (ouvert à gauche, ouvert à droite), OF, FO ou FF.

`intervalley.bornes(a,b)` même chose sur l'axe des ordonnées.

## Exemple 33

```
repere(-1,5,0.9cm,-1,4.5,1cm);
vardef f(expr x)=x**2-5x+7 enddef;
draw axes(1,1);
draw courbefonc(f,-1,5,50)
    withpen pencircle scaled 1 withcolor bleu;
drawoptions(dashed evenly withcolor rouge);
draw projectionx.bot(_c(1,f(1)));
draw projectiony.llft(_c(2.5,f(2.5)),"0,75");
draw projectionaxes(_c(4,f(4)));
drawoptions(withcolor violet);
draw intervallex.OF(1,4);
draw intervalley.FF(0.75,3);
label(LaTeX("$f([1;4])=[0,75;3]$"),_c(2.5,4));
draw cadre;
fin;
```



## 8 Statistiques

### 8.1 Boite à moustache

`boitemoustache(min,Q1,Me,Q3,max,dec,larg)` « Boite à moustache » correspondant aux données en argument. Elle est située à une distance `dec` de l'axe des abscisses et le rectangle a une largeur de `larg`. Ces deux dernières valeurs sont optionnelles et valent par défaut 1,5 cm et 1 cm.

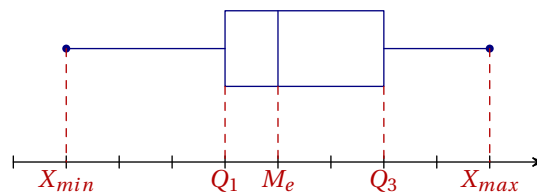
`projboitemoustache(t)` Figure formée des lignes joignant les cinq valeurs du dernier diagramme en boite dessiné à son projeté sur l'axe des abscisses ainsi que de certaines étiquettes : Si `t` est vide, les textes  $X_{min}$ ,  $Q_1$ ,  $M_e$ ,  $Q_3$  et  $X_{max}$  sont affichés ; si `t` est un entier, les valeurs arrondies à  $10^{-t}$  sont affichées ; si `t` est une liste de cinq textes (ou valeurs), ceux-ci sont affichés.

### Exemple 34

```

repere(-0.5,10,0.7cm,-1,5,0.7cm);
setaxes(0,10,0,1);
draw axex(1,0);
draw boitemoustache(1,4,5,7,9)
                    withcolor marine;
draw projboitemoustache.bot()
                    withcolor 0.7rouge dashed evenly;
fin;

```

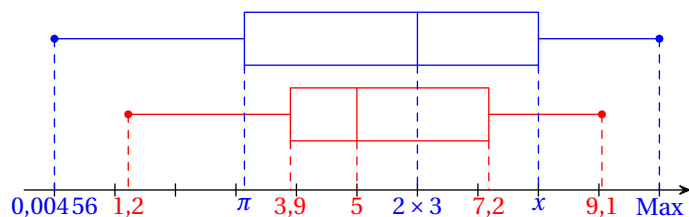


### Exemple 35

```

repere(-1,11,0.8cm,-1,5,0.7cm);
settout(-0.5,10.5,0,1);
draw axex(1,0);
drawoptions(withcolor rouge);
draw boitemoustache(1.22,3.9,5,7.18,9.05,1cm,0.7cm);
draw projboitemoustache.bot(1) dashed evenly;
drawoptions(withcolor blue);
draw boitemoustache(0,3.14,6,8,10,2cm,0.7cm);
draw projboitemoustache.bot(0.00456,"$\pi$","$\num{2x3}$","$x$","Max")
                    dashed evenly;
fin;

```



## 9 Divers

### 9.1 Composition des étiquettes

Tous les textes et étiquettes peuvent être composés en utilisant la macro ci-dessous. Cette macro utilise `latexmp` si ce fichier est chargé ou l'écriture dans un fichier externe (adaptation de la macro `TEX`) dans le cas contraire.

**LaTeX(ch)** Figure formée de la chaîne `ch` composée avec `LATEX`. Il est donc possible d'utiliser la macro `decimal()` de `METAPOST`.

### Exemple 36

```






















repere(-1,7,1cm,-1,1,1cm);
for i=2 upto 6:
label(LaTeX("$\frac{1}{&decimal(i)}$"),(i*cm,0));
endfor
fin;

```

$\frac{1}{2}$     $\frac{1}{3}$     $\frac{1}{4}$     $\frac{1}{5}$     $\frac{1}{6}$

### 9.2 Couleurs

Certaines couleurs sont définies par leur nom et peuvent être utilisées directement :

	rouge		vert		bleu		cyan		magenta		jaune		noir
	marron		lime		orange		rose		pourpre		olive		violet
	beige		marine		moutarde		grisclair		gris		grisfoncé		vertfoncé

Toutes ces couleurs sont définies selon le modèle « rgb ». Pour les obtenir selon le modèle « cmyk », remplacer la première lettre par une majuscule.

### 9.3 Figures pour une présentation

`figureinter` exporte la figure telle qu'elle est au moment où cette commande apparaît. La numérotation est incrémentée et la figure peut continuer.

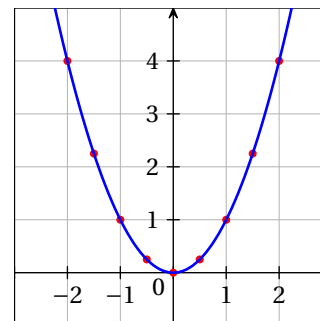
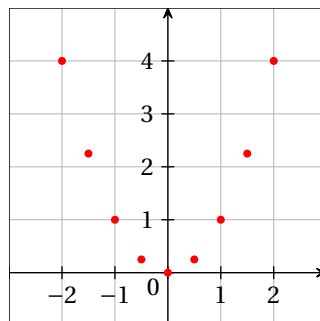
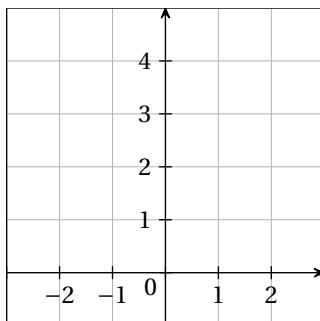
L'exemple ci-dessous crée trois figures :

#### Exemple 37

```

repere(-3,3,0.7cm,-1,5,0.7cm);
path C_f;
vardef f(expr x)=x**2 enddef;
C_f=courbefonc(f,-3,3,50);
draw quadrillage(1,1);
draw axes(1,1);
draw cadre;
figureinter;
draw courbepoints(f,-2,2,9) withcolor rouge;
figureinter;
draw C_f withcolor bleu withpen pencircle scaled 1;
fin;

```



Si ces trois figures s'appellent `mfigure.1`, `mfigure.2` et `mfigure.3`, elles peuvent être incluses dans un document de la classe `beamer` avec le code ci-dessous :

```

\documentclass{beamer}
\ifpdf % Pour utiliser pdflatex
\DeclareGraphicsRule{*}{mps}{*}{}
\fi
\begin{document}
\begin{frame}
\includegraphics<+>{mfigure.1}%
\includegraphics<+>{mfigure.2}%
\includegraphics<+>{mfigure.3}%
\end{frame}
\end{document}

```

## 9.4 Code embarqué dans un document L<sup>A</sup>T<sub>E</sub>X

Certains packages permettent d'écrire du code METAPOST directement dans un document L<sup>A</sup>T<sub>E</sub>X. `repere` est compatible avec, entre autres, `emp` et `mpgraphics`.

### Utilisation du package `emp`

```
pdflatex monfichier.tex
mpost monfichier.mp
mpost monfichier.mp
pdflatex monfichier.tex
```

```
\documentclass{article}
\usepackage{emp}
\usepackage{ifpdf}
\ifpdf % Pour utiliser pdflatex
\DeclareGraphicsRule{*}{mps}{*}{}
\fi
\begin{document}
\begin{empfile}
\begin{empcmds}
input latexmp;
input repere;
\end{empcmds}
\begin{emp}(0,0)
repere(-3,3,0.8cm,-2.5,2.5,1cm);
draw axes(1,1);
fin;
\end{emp}
\end{empfile}
\end{document}
```

### Utilisation du package `mpgraphics`

```
pdflatex -shell-escape monfichier.tex
```

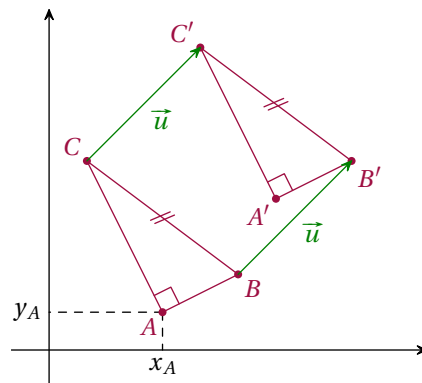
```
\documentclass{article}
\usepackage[runs=2]{mpgraphics}
\begin{document}
\begin{mpdefs}
input latexmp;
input repere;
\end{mpdefs}
\begin{mpdisplay}
repere(-3,3,0.8cm,-2.5,2.5,1cm);
draw axes(1,1);
fin;
\end{mpdisplay}
\end{document}
```

## 10 Exemples d'utilisation simultanée de `repere` et `geometriesyr`

Il est possible, dans une figure créée avec `repere`, d'utiliser certaines macros de `geometriesyr`. Il faut alors charger `geometriesyr` *avant* `repere`.

### Exemple 38

```
repere(-1,10,0.5cm,-1,9,0.5cm);
pair A,B,C,A',B',C',u;
A=_c(3,1);B=_c(5,2);C=_c(1,5);u=_c(3,3);
A'-A=B'-B=C'-C=u;
draw projectionaxes(A,LaTeX("$x_A$"),
LaTeX("$y_A$")) dashed evenly;
draw axes(0,0);
drawoptions(withcolor pourpre);
draw triangle(A,B,C);draw triangle(A',B',C');
draw codeperp(B,A,C,7);draw codeperp(B',A',C',7);
draw codesegments(B,C,B',C',2);
nomme.llft(A);nomme.lrt(B);nomme.ulft(C);
nomme.llft(A');nomme.lrt(B');nomme.ulft(C');
drawoptions(withcolor vertfonce);
draw vecteur.lrt(B,u);draw vecteur.lrt(C,u);
fin;
```



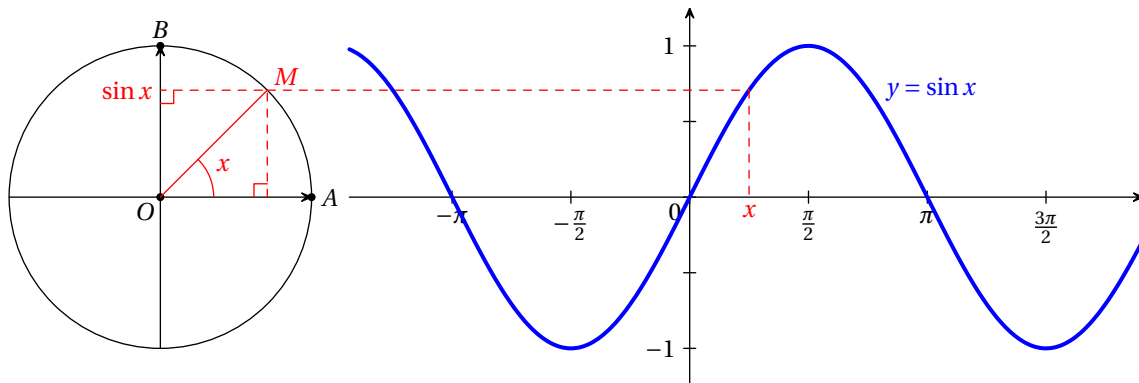
### Exemple 39

```

repere(-9.5,6,1cm,-1.25,1.25,2cm);
%cercle
pair O,A,A',B,B',M,S,C;
numeric x;x=pi/4;
O=_c(-7,0);A=_c(-5,0);
B=_c(-7,1);A+A'=B+B'=20;
M=rotation(A,O,180*x/pi);
C=projection(M,A,A');
S=projection(M,B,B');
draw cerclepoint(O,A);
drawarrow A'--A;drawarrow B'--B;
nomme.rt(A);nomme.top(B);nomme.llft(O);
drawoptions(withcolor rouge);
draw O--M;draw C--M--S dashed evenly;
draw codeperp(M,C,O,5);
draw codeperp(O,S,M,5);
draw codeangle.urt(A,O,M,0,
                    LaTeX("$x$"));
marque_p:="";

nomme.urt(M);
label.lft(LaTeX("$\sin x$"),S);
%repere
settout(-4.5,6,-1.25,1.25);
path C_f;
C_f=courbefonc(sin,-4.5,6,80);
pair m;m=_c(x,sin(x));
drawoptions();
draw axexpi.bot(1,2);
draw axey(0.5,1);
drawoptions(withcolor bleu);
draw C_f withpen pencircle scaled 1.5;
nomme.urt(C_f,2.5,LaTeX("$y=\sin x$"));
drawoptions(withcolor rouge);
draw projectionx.bot(m,LaTeX("$x$"))
                    dashed evenly;
%
draw M--m dashed evenly;
fin;

```



### Exemple 40

```

repere(-0.5,5,1cm,-0.5,5,1cm);
coulpoint:=blue;coullabel:=blue;
pair A,B,C,D;
A=_c(0.5,0.5);B=_c(4,1);C=_c(3,4);
typetrace:="mainlevee";
draw axes(1,1);
drawoptions(withcolor violet);
draw triangle(A,B,C);
nomme.llft(A);nomme.lrt(B);
nomme.top(C);
draw marqueangle(B,A,C,0);
drawoptions(withcolor vertfonce);
draw cercles(CentreCercleC(A,B,C),A);
fin;

```

