Documentation de `repere.mp`

Olivier PÉAULT*

2 mai 2023

Table des matières

*E-mail : o.peault@posteo.net

Première partie

Repères et géométrie

1 Utilisation du fichier

1.1 Généralités

Les macros du fichier `repere.mp` ont pour but de simplifier la création de figures dans un repère du plan avec METAPOST notamment :

— Figures dans un repère, représentations graphiques de fonctions ;

— Figures usuelles de géométrie plane ;

— En complément, possibilité de tracer des grilles et tableaux..

L'objectif principal est de répondre aux besoins de l'enseignement secondaire français de mathématiques.

1.2 Documentation

Les commandes et macros proposées sont de plusieurs types :

— Des `déclarations`, qui définissent la figure, modifient certaines variables ou paramètres.

— Des `objets`, au sens de METAPOST qui peuvent être des points (pair), des chemins (path), des figures (picture)...

— Des `« labels »` qui fonctionnent sur le même principe que la commande `label` de METAPOST et affichent une étiquette qui peut être définie séparément.

— Des `paramètres` qui peuvent être modifiés pour changer l'apparence de la figure.

Les objets qui sont des « figures complètes » (dessins et étiquettes) ont des noms qui commencent par une majuscule.

1.3 Utilisation de `repere`

L'utilisation de LuaLaTeX avec le package `luamplib` est la manière la plus simple d'utiliser `repere`. Il suffit juste de charger les packages `siunitx` et `esvect` utilisés par `repere`.

Figure isolée

`lualatex mafigure.tex`

```
\documentclass{standalone}
\usepackage{fontspec}
\usepackage{siunitx,esvect}
\usepackage{luamplib}
\mplibnumbersystem{decimal} %Si besoin
\everymplib{input repere;}
\begin{document}
\begin{mplibcode}
  repere();
    draw axes(1,1);
  fin;
\end{mplibcode}
\end{document}
```

Code embarqué

`lualatex monfichier.tex`

```
\documentclass{article}
\usepackage{fontspec}
\usepackage{siunitx,esvect}
\usepackage{luamplib}
\mplibnumbersystem{decimal} %Si besoin
\everymplib{verbatimtex
              \leavevmode
            etex;
            input repere;}
\begin{document}
Mon texte, mon texte, mon texte

\begin{mplibcode}
  repere();
    draw axes(1,1);
  fin;
\end{mplibcode}

Mon texte, mon texte, mon texte
\end{document}
```

Il est aussi possible d'utiliser une compilation `mpost` « traditionnelle ». Voir section 12.1.

2 Repère utilisateur

2.1 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 `Xmin` 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`.

`repere()  repere.larg()  repere.orth()`

Définit un repère en utilisant les valeurs des paramètres `Xmin`, `Xmax`...

`fin`

Termine la figure et la découpe pour ne garder que la partie limitée par le repère utilisateur.

Paramètres

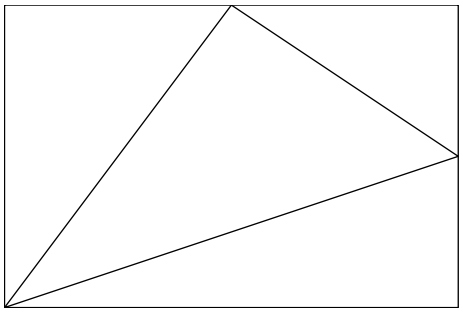
Nom	Type	Défaut	
Ux	numeric	1cm	Unité sur l'axe des abscisses
Uy	numeric	1cm	Unité sur l'axe des ordonnées
Uxy	numeric		Si positif, unité sur les deux axes
Xmin	numeric	-10	Valeur minimale sur l'axe des abscisses
Xmax	numeric	10	Valeur maximale sur l'axe des abscisses
Ymin	numeric	-10	Valeur minimale sur l'axe des ordonnées
Ymax	numeric	10	Valeur maximale sur l'axe des ordonnées
theta	numeric	90	Angle entre les axes du repère

`cadre` `path`

Chemin fermé qui fait le tour du repère.

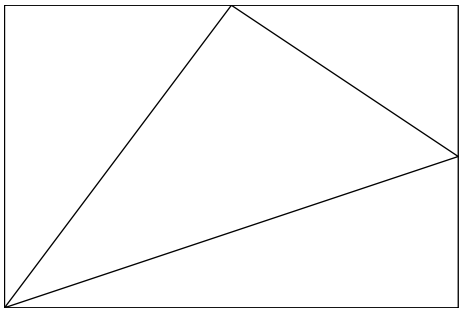
Exemple 1

```
repere(-3,3,1cm,-2,2,1cm);
 draw (-3,-2)--(3,0)--(0,2)--cycle;
 draw cadre;
fin;
```



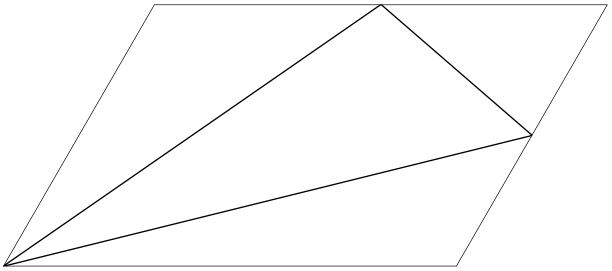
Exemple 2

```
Uxy:=0.5cm;
Xmin:=-6;Xmax:=6;
Ymin:=-4;Ymax:=4;
repere();
 draw (-6,-4)--(6,0)--(0,4)--cycle;
 draw cadre;
fin;
```



Exemple 3

```
repere(-3,3,1cm,-2,2,1cm,60);
 draw (-3,-2)--(3,0)--(0,2)--cycle;
 draw cadre;
fin;
```



2.2 Axes

2.2.1 Généralités

`axex.<pos>(<pas_grad>,<pas_val>)` `picture`

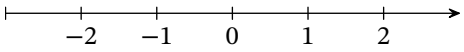
Axe des abscisses gradué avec un pas de `pas_grad` et étiqueté avec un pas de `pas_val`.
Si `pas_grad` est négatif ou nul, l'axe n'est pas gradué et si `pas_val` est négatif ou nul, l'axe n'est pas étiqueté.
`pos` est un paramètre optionnel qui désigne la position (au sens de METAPOST : `rt`, `urt`, `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>(<pas_grad>,<pas_val>)` `picture`

Axe des ordonnées. La valeur par défaut de `pos` est `lft`.

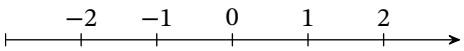
Exemple 4

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



Exemple 5

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



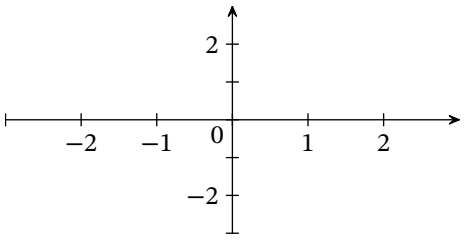
Exemple 6

```
repere(-3,3,1cm,-1,1,1cm);
 draw axex(1,0);
fin;
```


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`)

Exemple 7

```
repere(-3,3,1cm,-3,3,0.5cm);
 draw axex(1,1);
 draw axey(1,2);
fin
```

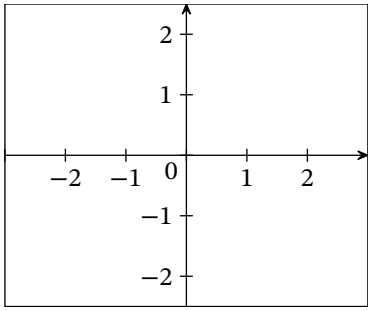


`axes.<pos>(<pas_grad>,<pas_val>)` `picture`

Figure formée par les deux axes gradués avec le même pas `pas_grad` et étiquetés avec le même pas `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.

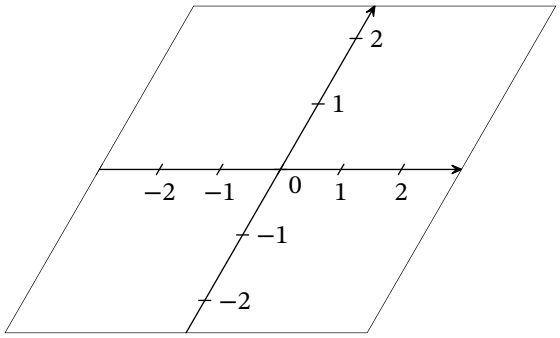
Exemple 8

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



Exemple 9

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



`axexo.<pos>(<pas_grad>,<pas_val>)` `picture`

Axe des abscisses. L'étiquette correspondant à l'intersection des axes est dessinée à la position `pos`.

`axeyo.<pos>(<pas_grad>,<pas_val>)` `picture`

Axe des ordonnées. L'étiquette correspondant à l'intersection des axes est dessinée à la position `pos`.

`axeso.<pos>(<pas_grad>,<pas_val>)` `picture`

Les deux axes. Les étiquettes correspondant à l'intersection des axes sont dessinées en fonction de la position `pos`.

`axexn.<pos>(<pas_grad>,<pas_val>)` `picture`

Axe des abscisses. L'étiquette correspondant à l'intersection des axes n'est pas dessinée.

`axeyn.<pos>(<pas_grad>,<pas_val>)` `picture`

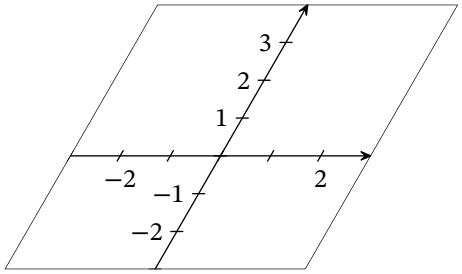
Axe des ordonnées. L'étiquette correspondant à l'intersection des axes n'est pas dessinée.

`axesn.<pos>(<pas_grad>,<pas_val>)` `picture`

Les deux axes. Les étiquettes correspondant à l'intersection des axes ne sont pas dessinées.

Exemple 10

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



2.2.2 Réglages des axes

`interaxes(<x>,<y>)`

Définit les coordonnées du point d'intersection des axes. Par défaut ces coordonnées sont (0,0).

`setaxes(<xmin>,<xmax>,<ymin>,<ymax>)`

Définit les valeurs minimales et maximales pour les axes.

`setgrad(<xmin>,<xmax>,<ymin>,<ymax>)`

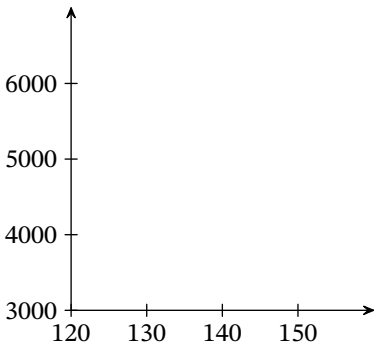
Définit les valeurs minimales et maximales pour les graduations.

`setval(<xmin>,<xmax>,<ymin>,<ymax>)`

Définit les valeurs minimales et maximales pour l'étiquetage.

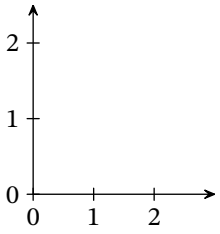
Exemple 11

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



Exemple 12

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

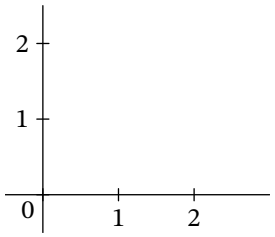


Paramètres

Nom	Type	Défaut	
`flecheaxe`	`boolean`	`true`	Trace les flèches (ou non) au bout des axes
`defaultscale`	`numeric`	`1`	Échelle à laquelle sont dessinées les étiquettes

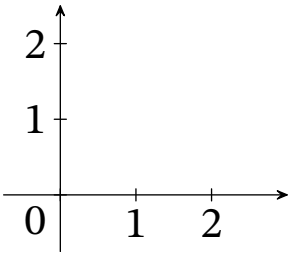
Exemple 13

```
repere(-0.5,3,1cm,-0.5,2.5,1cm);
 flecheaxe:=false;
 draw axes(1,1);
fin;
```



Exemple 14

```
repere(-0.75,3,1cm,-0.75,2.5,1cm);
 defaultscale:=1.5;
 draw axes(1,1);
fin;
```



2.2.3 Graduations multiples de π

`axexpi.<pos>(<n>,<d>)` — `picture`

Axe des abscisses gradué et étiqueté avec un pas de $\frac{n\pi}{d}$.

`axexpi.<pos>(<n>,<d>)` — `picture`

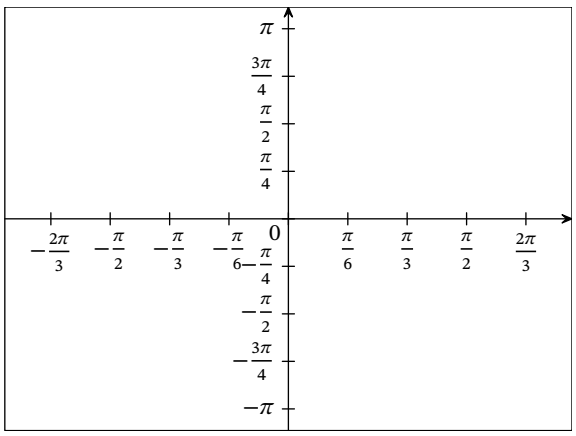
Axe des ordonnées gradué et étiqueté avec un pas de $\frac{n\pi}{d}$.

`axespi.<pos>(<n>,<d>)` — `picture`

Les deux axes gradués et étiquetés avec un pas de $\frac{n\pi}{d}$.

Exemple 15

```
repere(-2.5,2.5,1.5cm,-3.5,3.5,0.8cm);
 defaultscale:=0.8;
 draw axexpi(1,6);
 draw axeypi(1,4);
 draw cadre;
fin;
```



`axexpio.<pos>(<n>,<d>)` `axeypio.<pos>(<n>,<d>)` `axespio.<pos>(<n>,<d>)` `picture`

Axes gradués et étiquetés avec un pas de $\frac{n\pi}{d}$ sans l'étiquette correspondant à l'intersection des axes.

`axexpin.<pos>(<n>,<d>)` `axeypin.<pos>(<n>,<d>)` `axespin.<pos>(<n>,<d>)` `picture`

Axes gradués et étiquetés avec un pas de $\frac{n\pi}{d}$ avec l'étiquette correspondant à l'intersection dessinées sur chacun des axes.

Paramètres

Nom	Type	Défaut	
`displayfrac`	`boolean`	`false`	Permet d'obtenir les fractions en mode « display-style »

2.2.4 Graduations isolées

`axexpart.<pos>(<x1>,<lab1>,<x2>,<lab2>,...)` `picture`

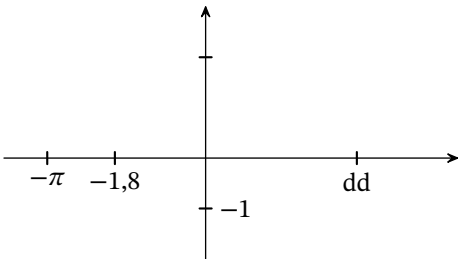
Graduations et étiquetages partiels de l'axe des abscisses pour les valeurs `x1`, `x2`... et les étiquettes `lab1`, `lab2`... à la position `pos`. Si `pos` est omis, les étiquettes sont placées à la position `bot`.
Les étiquettes peuvent être soit des chaînes de caractères (`string`), soit des figures (`picture`). Si `labn` est omis, la valeur de `xn` sera utilisée comme étiquette. Pour obtenir une graduation sans étiquette, on peut utiliser la chaîne vide `""`.

`axeypart.<pos>(<y1>,<lab1>,<y2>,<lab2>,...)` `picture`

Même chose que précédemment mais sur l'axe des ordonnées. Si `pos` est omis, les étiquettes sont placées à la position `lft`.

Exemple 16

```
repere.orth(-4,5,6cm,-2,3);
 draw axes(0,0);
 draw axexpart(-1.8,-pi,"$-\pi$",3,"dd");
 draw axeypart.rt(-1,2,"");
fin;
```

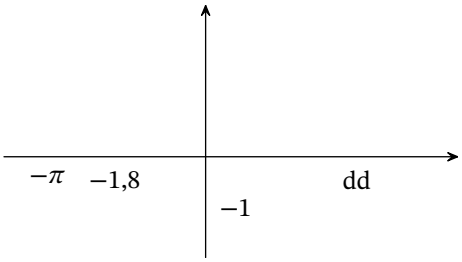


Paramètres

Nom	Type	Défaut	
gradxpart	boolean	true	Contrôle l'affichage des graduations isolées sur l'axe des abscisses.
gradypart	boolean	true	Contrôle l'affichage des graduations isolées sur l'axe des ordonnées.

Exemple 17

```
repere.orth(-4,5,6cm,-2,3);
 draw axes(0,0);
 gradxpart:=false;gradypart:=false;
 draw axexpart(-1.8,-pi,"$-\pi$",3,"dd");
 draw axeypart.rt(-1,2,"");
fin;
```



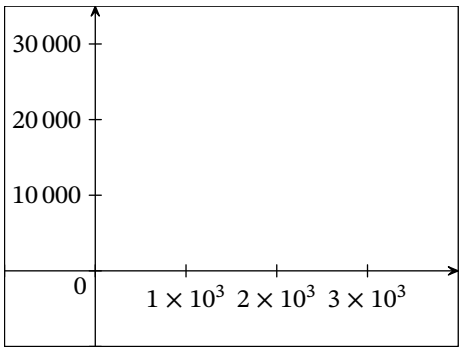
2.2.5 Ajout de texte sur les graduations

Paramètres

Nom	Type	Défaut	
extranumx	string	""	Chaine de caractères qui sera ajoutée après les valeurs des graduations sur l'axe des abscisses avant d'être composées avec la commande \num de siunitx.
extranumy	string	""	Même chose sur l'axe des ordonnées.

Exemple 18

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



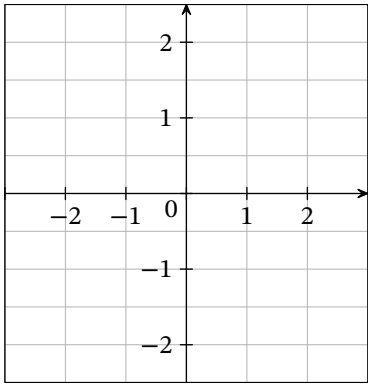
2.3 Quadrillages

quadrillage(<x>,<y>) picture

Quadrillage avec un pas de x sur l'axe des abscisses et de y sur l'axe des ordonnées.

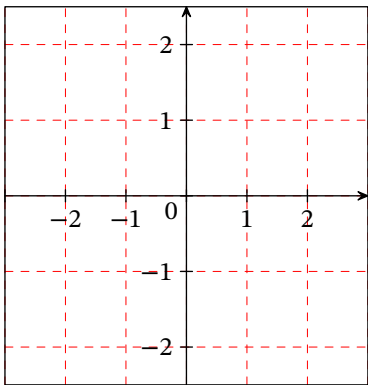
Exemple 19

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



Exemple 20

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

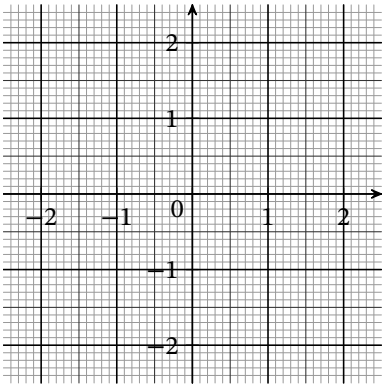


`papiermillimetre` `picture`

Quadrillage en cm et dixièmes de cm.

Exemple 21

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

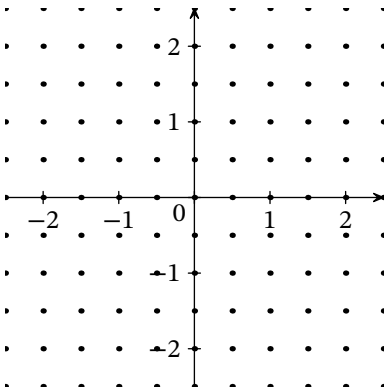


`papierpointe(<x>,<y>)` `picture`

quadrillage formé de points avec un pas de x sur l'axe des abscisses et de y sur l'axe des ordonnées.

Exemple 22

```
repere(-2.5,2.5,1cm,-2.5,2.5,1cm);
draw papierpointe(0.5,0.5);
draw axes(1,1);
fin;
```



Paramètres

Nom	Type	Défaut	
q_ep	numeric	0.3	Épaisseur des traits des quadrillages
q_coul	color	0.7white	Couleur des traits des quadrillages
pm_epa	numeric	0.2	Épaisseur des traits des dixièmes de cm du papier millimétré
pm_epb	numeric	0.4	Épaisseur des traits des demis cm du papier millimétré
pm_epc	numeric	0.6	Épaisseur des traits des cm du papier millimétré
pm_coula	color	0.6white	Couleur des traits des dixièmes de cm du papier millimétré
pm_coulb	color	0.2white	Couleur des traits des demis cm du papier millimétré
pm_coulc	color	black	Couleur des traits des cm du papier millimétré
pp_ep	numeric	2	Taille des points du papier pointé
pp_coul	color	black	Couleur des points du papier pointé

`setquad(<xmin>,<xmax>,<ymin>,<ymax>)`

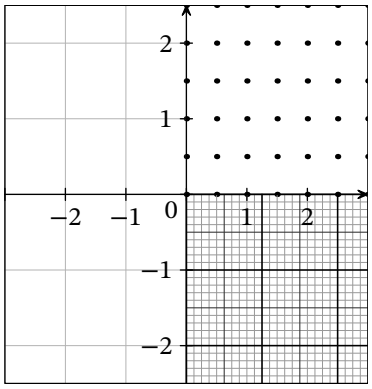
Définit les valeurs minimales et maximales pour les divers quadrillages.

`setrepere(<xmin>,<xmax>,<ymin>,<ymax>)`

Définit les valeurs minimales et maximales pour les axes, les graduations, les étiquettes et les quadrillages.

Exemple 23

```
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);
setquad(0,3,-2.5,0);
draw papiermillimetre;
draw axes(1,1);
draw cadre;
fin;
```



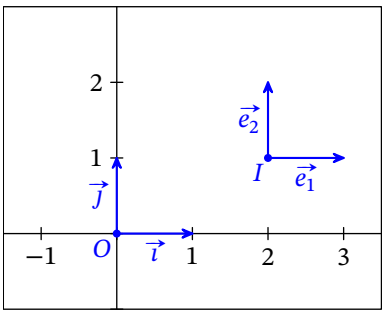
2.4 Base

`base(<O>,<i>,<j>)` `picture`

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

Exemple 24

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

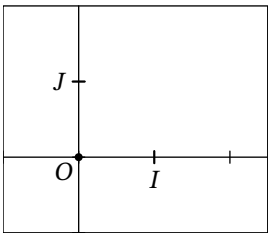


`basep(<O>,<I>,<J>)` `picture`

Figure formée par le point d'intersection des axes et son nom (O), ainsi que des deux points qui définissent le repère.

Exemple 25

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



3 Points, vecteurs

3.1 Points

`(<x>,<y>)` `pair`

Point (ou vecteur) de coordonnées cartésiennes `x` et `y` dans le repère utilisateur.

`pol(<r>,<t>)` `pair`

Point (ou vecteur) de coordonnées $(r\cos t; r\sin t)$ dans le repère utilisateur.

`pold(<r>,<t>)` `pair`

Même chose avec l'angle donné en degrés.

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

`marquepointFig(<A>)` `picture`

Dessin du point `A` selon le paramètre `marque_p`.

`marquepoint(<A>)`

Dessine le point `A` selon le paramètre `marque_p`. C'est en réalité `draw marquepointFig(A)`. C'est un peu l'équivalent de `drawdot`.

`marquepoints(<A>,<B>,<C>...)`

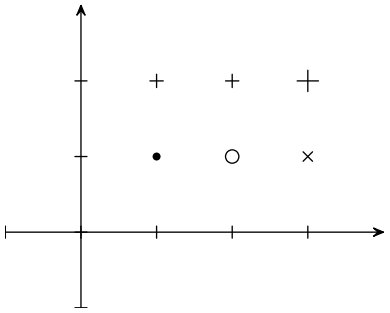
Permet de marquer plusieurs points.

Paramètres

Nom	Type	Défaut	
`marque_p`	`string`	`"*"`	Définit le dessin du point. Les valeurs possibles sont `"*"`, `"o"`, `"x"`, `"+"` et `""`.
`taillepoint`	`numeric`	`3`	Diamètre du cercle représentant le point dans les cas `"*"` et `"o"`.
`taillecroix`	`numeric`	`5`	Largeur de la croix représentant le point dans le cas `"+"` et `"x"`.

Exemple 26

```
repere(-1,4,1cm,-1,3,1cm);
pair A,B,C,D,E,F;
A=(1,1);B=(2,1);C=(3,1);
D=(1,2);E=(2,2);F=(3,2);
draw axes(1,0);
marquepoint(A);
marque_p:="o";taillepoint:=5;
marquepoint(B);
marque_p:="x";marquepoint(C);
marque_p:="+";marquepoints(D,E);
taillecroix:=8;marquepoint(F);
fin;
```

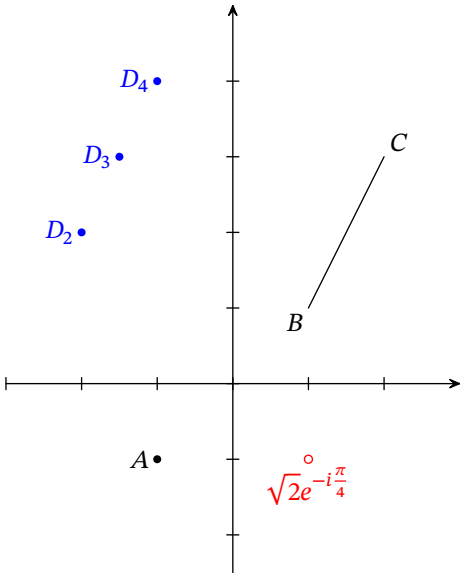


`nomme.<pos>(<A>,<nom>)`

Marque le point et affiche son nom à la position `pos` (qui peut être `rt`, `urt`, `top`, `ulft`...). `nom` peut être soit une chaîne de caractères, 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 27

```
repere(-3,3,1cm,-2.5,5,1cm);
pair A,B,C,D[];
A=(-1,-1);B=(1,1);C=(2,3);
draw axes(1,0);
nomme.lft(A);
marque_p:="";
nomme.llft(B);nomme.urt(C);
draw B--C;
marque_p:="o";
drawoptions(withcolor red);
nomme.bot(pol(sqrt(2),-pi/4),
        "$\sqrt{2}e^{-i\frac{\pi}{4}}$");
marque_p:="*";
for i=2 upto 4:
  D[i]=(-3+i/2,i);
  nomme.lft(D[i]) withcolor blue;
endfor
fin;
```

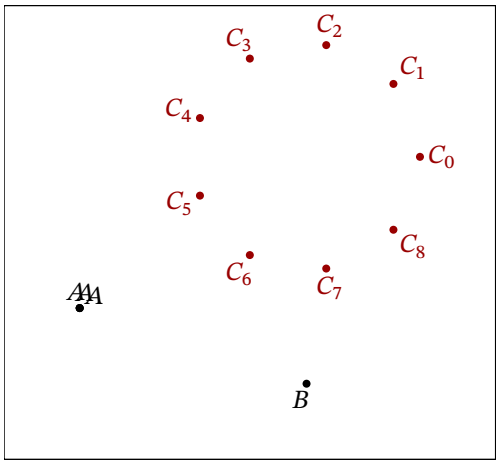


`nomme.[<a>](<A>,<nom>)`

Il est possible d'obtenir un placement plus fin des étiquettes en remplaçant la position au sens de METAPOST (`rt`, `urt`...) par un nombre `a` qui représente alors la position de l'étiquette par rapport au point en degrés.

Exemple 28

```
repere(-3,3.5,1cm,-3,3,1cm);
 pair A,B,C[];
 A=(-2,-1);B=(1,-2);
 nomme[40](A);nomme[70](A);nomme[100](A);
 nomme[-110](B);
 for i=0 upto 8:
   C[i]:= (1+1.5*cosd(40i),1+1.5*sind(40i));
   nomme[40*i](C[i]) withcolor 0.6red;
 endfor
draw cadre;
fin;
```



`nommerot.<pos>(<A>,<nom>)(<angle>)`

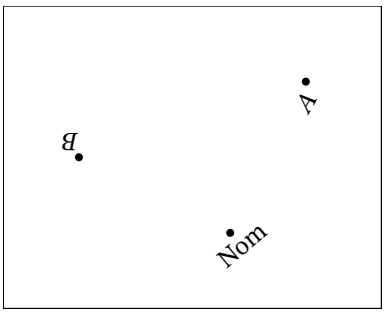
Même chose que `nomme.pos(A,nom)` mais l'étiquette est tournée d'un angle `angle`.

`nommerot.[<a>](<A>,<nom>)(<angle>)`

Même chose que `nomme[a](A,nom)` mais l'étiquette est tournée d'un angle `angle`.

Exemple 29

```
repere(-2,3,1cm,-1,3,1cm);
 pair A,B,C;
 A=(2,2);B=(-1,1);C=(1,0);
 nommerot.bot(A)(60);
 nommerot[120](B)(180);
 nommerot[-45](C,"Nom")(40);
 draw cadre;
fin;
```

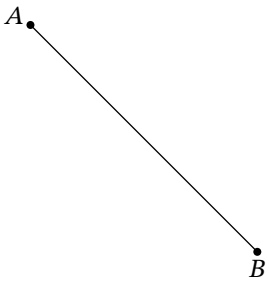


`nommedef.<pos>(<A>,<point>)`

Définit la variable `A`, lui donne la valeur `point` et affiche le point. C'est donc un raccourci pour : `pair A;A:=point;nomme.pos(A)`.

Exemple 30

```
repere();
nommedef.[150](A,(0,0));
nommedef.bot(B,(3,-3));
draw A--B;
fin;
```



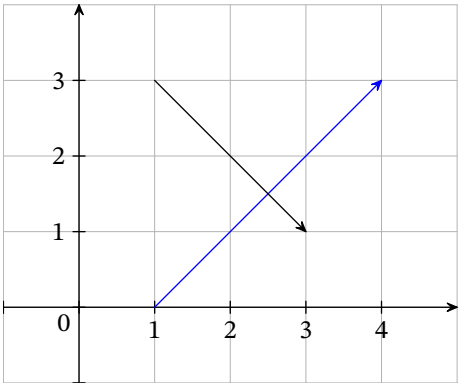
3.2 Vecteurs

`drawvecteur(A,u)`

Trace le représentant d'origine `A` du vecteur `u`. C'est l'équivalent de la commande `drawarrow A--A+u`

Exemple 31

```
repere(-1,5,1cm,-1,4,1cm);
 pair A,u;
 A=(1,0);u=(3,3);
 draw quadrillage(1,1);
 draw axes(1,1);
 drawvecteur(A,u) withcolor blue;
 drawvecteur((1,3),(2,-2));
fin;
```



4 Droites

4.1 Droites

`droite(<A>,<B>)` `path`

Droite (AB). Il s'agit en réalité d'un segment coupé au niveau des limites du repère.

`droite(<a>,<b>,<c>)` `path`

Droite d'équation $ax + by + c = 0$ dans le repère utilisateur.

`droite(<a>,<b>)` `path`

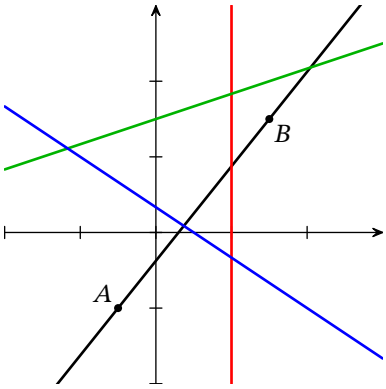
Droite d'équation $y = ax + b$ dans le repère utilisateur.

`droite(<c>)` `path`

Droite d'équation $x = c$ dans le repère utilisateur.

Exemple 32

```
repere(-2,3,1cm,-2,3,1cm);
pair A,B;
A=(-0.5,-1);B=(1.5,1.5);
draw axes(1,0);
nomme.ulft(A);
nomme.lrt(B);
drawoptions(withpen pencircle scaled 1);
draw droite(A,B);
draw droite(1) withcolor red;      %x=1
draw droite(1/3,3/2) withcolor 0.7green; %y=(1/3)x+3/2
draw droite(2,3,-1) withcolor blue; %2x+3y-1=0
fin;
```



4.2 Demi-droites

`demidroite(<A>,<B>)` — `path`

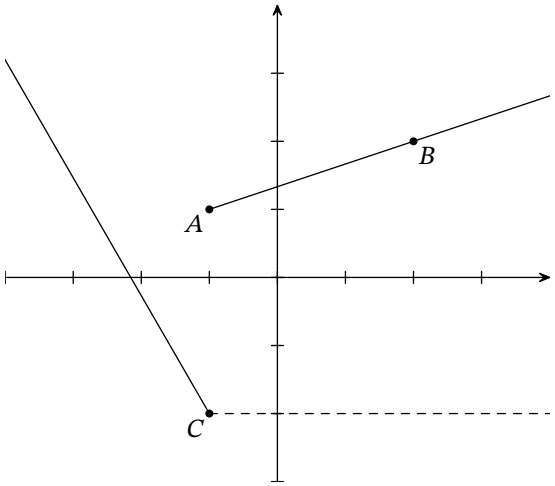
Demi-droite $[AB)$.

`demidroite(<A>,<a>)` — `path`

Demi-droite d'origine A qui fait un angle a avec la direction (Ox).

Exemple 33

```
repere(-4,4,0.9cm,-3,4,0.9cm);
pair A,B,C;
numeric a;
A=(-1,1);B=(2,2);C=(-1,-2);
draw axes(1,0);
draw demidroite(A,B);
nomme.llft(A);nomme.lrt(B);
draw demidroite(C,0) dashed evenly;
draw demidroite(C,120);
nomme.llft(C);
fin;
```



5 Courbes et fonctions

5.1 Courbes

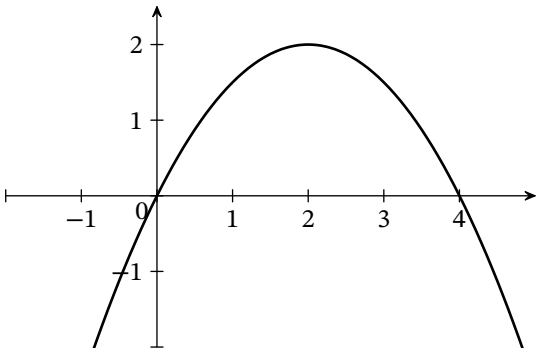
METAPOST permet de définir simplement des fonctions (en utilisant par exemple la syntaxe suivante : `vardef f(expr x)=2x+1 enddef;`) et de définir des courbes passant par des points donnés (`A..B..C`). Ces possibilités sont utilisées dans les macros qui suivent.

`courbefonc(<f>)()` — `path`

Courbe représentant la fonction f sur l'intervalle `[Xmin;Xmax]` définissant le repère.

Exemple 34

```
repere(-2,5,1cm,-2,2.5,1cm);
vardef f(expr x)=-0.5(x**2)+2*x enddef;
path C_f;
draw axes(1,1);
C_f=courbefonc(f)();
draw C_f withpen pencircle scaled 1;
fin;
```

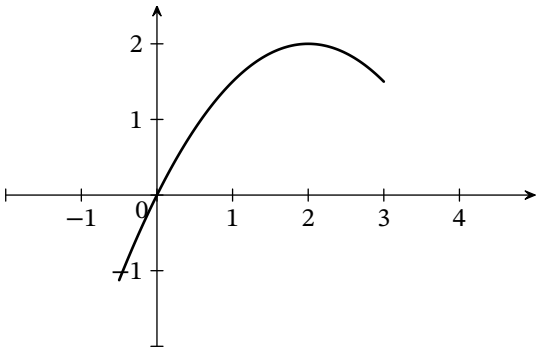


`courbefonc(<f>)(<xminf>,<xmaxf>)` — `path`

Courbe représentant la fonction f sur l'intervalle `[xminf;xmaxf]`. Ces valeurs peuvent aussi être données de manière globale.

Exemple 35

```
repere(-2,5,1cm,-2,2.5,1cm);
vardef f(expr x)=-0.5(x**2)+2*x enddef;
path C_f;
draw axes(1,1);
C_f=courbefonc(f)(-0.5,3);
draw C_f withpen pencircle scaled 1;
fin;
```



`courbefonc(<f>)(<xminf>,<xmaxf>,<nbf>)` — `path`

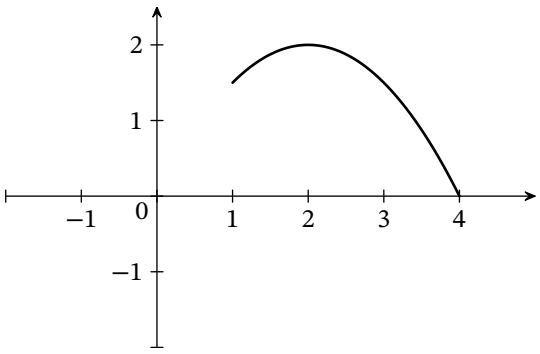
Courbe représentant la fonction f sur l'intervalle `[xminf;xmaxf]`. Le chemin est tracé avec `nbf` points d'interpolation. Ces valeurs peuvent aussi être données de manière globale.

Paramètres

Nom	Type	Défaut	
`xminf`	`numeric`	`Xmin`	Valeur minimale de l'abscisse pour le tracé des représentations graphiques de fonctions.
`xmaxf`	`numeric`	`Xmax`	Valeur maximale de l'abscisse pour le tracé des représentations graphiques de fonctions.
`nbf`	`numeric`	`50`	Nombre de points d'interpolation pour le tracé des représentations graphiques de fonctions

Exemple 36

```
repere(-2,5,1cm,-2,2.5,1cm);
vardef f(expr x)=-0.5(x**2)+2*x enddef;
path C_f;
draw axes(1,1);
xminf:=1;xmaxf:=4;
C_f=courbefonc(f)();
draw C_f withpen pencircle scaled 1;
fin;
```

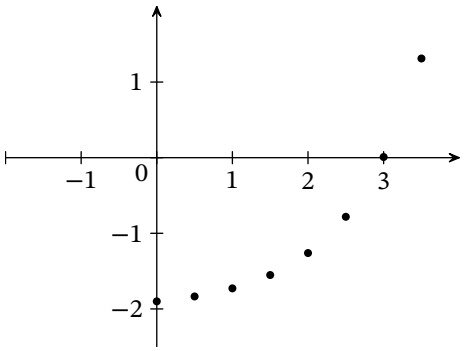


`courbepoint(<f>)(<xmin>,<xmax>,<n>)` — `picture`

Ne trace que les `n` points sans les relier. Les points sont dessinés en fonction de la valeur de `marque_p` (voir 3.1).

Exemple 37

```
repere(-2,4,1cm,-2.5,2,1cm);
vardef g(expr x)=exp(x)/10-2 enddef;
draw axes(1,1);
draw courbepoints(g)(0,4,9);
fin;
```

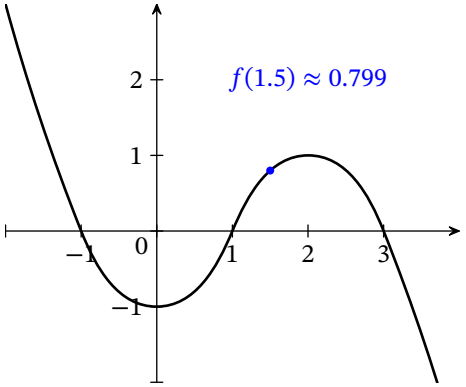


`fonccourbe(<p>)(<x>)` numeric

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.

Exemple 38

```
repere(-2,4,1cm,-2,3,1cm);
 path p; numeric a,b;
 p=(-2,3)..(-1,0)..(0,-1)..(1,0)
             ..(2,1)..(3,0)..(4,-3);
 draw axes(1,1);
 draw p withpen pencircle scaled 1;
 drawoptions(withcolor blue);
 a=1.5;b=fonccourbe(p)(a);
 marquepoint((a,b));
 b:=arrondi(1000,b);
 label("$f(1.5)\approx"& decimal(b) &"$",
        (2,2));
fin;
```

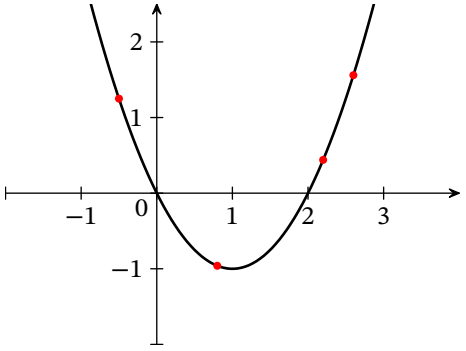


`marquepointcourbe(<p>,<x1>,<x2>,...)`

Marque les points de la courbe `p` d'abscisses `x1`, `x2`... La marque dépend de la valeur de `marque_p`.

Exemple 39

```
repere(-2,4,1cm,-2,2.5,1cm);
path C_f;
vardef f(expr x)= x**2-2x enddef;
C_f= courbefonc(f)(-2,3);
draw axes(1,1);
draw C_f withpen pencircle scaled 1;
drawoptions(withcolor red);
marquepointcourbe(C_f,-0.5,0.8,2.2,2.6);
fin;
```

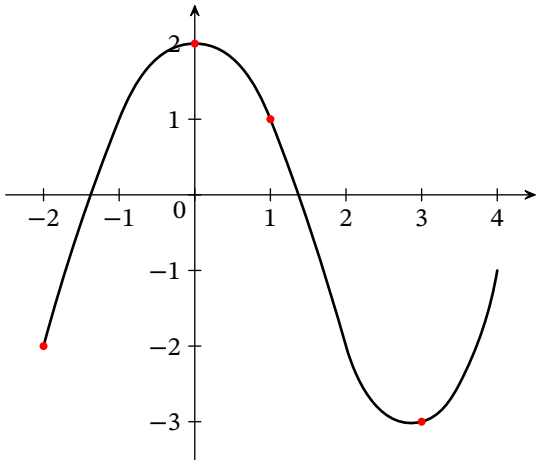


`marquepointchemin(<p>,<n1>,<n2>,...)`

Dans le cas d'un chemin défini par `A..B..C..`, marque le `n1`-ième point, le `n2`-ième point... La marque dépend de la valeur de `marque_p`. Attention, le premier point est numéroté 0.

Exemple 40

```
repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path p;
p=(-2,-2)..(-1,1)..(0,2)
  ..(1,1)..(2,-2)..(3,-3)
  ..(3.5,-2.5)..(4,-1);
draw axes(1,1);
draw p withpen pencircle scaled 1;
drawoptions(withcolor red);
marquepointchemin(p,0,2,3,5);
fin;
```



`antecedents(<X>,<y>,<p>)`

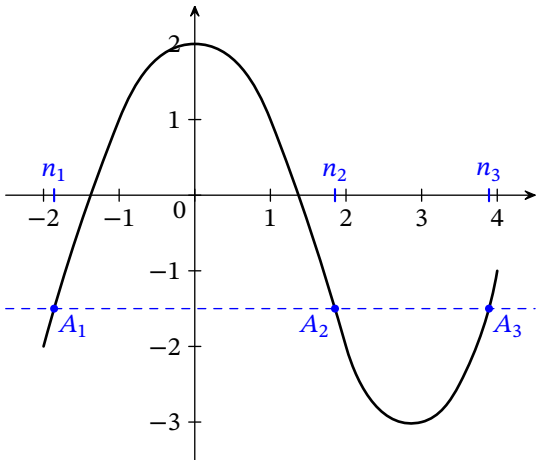
Stocke dans le tableau de nombres `X` (qu'il n'est pas nécessaire de déclarer) les antécédents de `y` par la fonction dont la courbe représentative est le chemin `p`.

`ptantecedents(<P>,<y>,<p>)`

Stocke dans le tableau de points `P` (qu'il n'est pas nécessaire de déclarer) les points du chemin `p` d'ordonnée `y`.

Exemple 41

```
repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path p;
p=(-2,-2)..(-1,1)..(0,2)..(1,1)
      ..(2,-2)..(3,-3)..(3.5,-2.5)
      ..(4,-1);
draw axes(1,1);
draw p withpen pencircle scaled 1;
drawoptions(withcolor blue);
draw droite(0,-1.5) dashed evenly;
antecedents(n,-1.5,p);
draw axexpart.top(n1,"$n_1$",n2,"$n_2$",
                  n3,"$n_3$");
ptantecedents(A,-1.5,p);
nomme.lrt(A1);nomme.llft(A2);
nomme.lrt(A3);
fin;
```

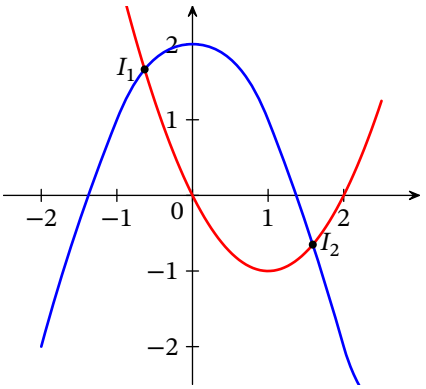


`intercourbes(<P>,<p>,<q>)`

Stocke dans le tableau de points P (qu'il n'est pas nécessaire de déclarer) les points d'intersection des chemins p et q.

Exemple 42

```
repere(-2.5,3,1cm,-2.5,2.5,1cm);
path p,C_f;
vardef f(expr x)= x**2-2x enddef;
p=(-2,-2)..(-1,1)..(0,2)..(1,1)
      ..(2,-2)..(3,-3)..(3.5,-2.5)
      ..(4,-1);
C_f= courbefonc(f)(-2,2.5);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw p withcolor blue;
draw C_f withcolor red;
intercourbes(I,C_f,p);
drawoptions(withcolor black);
nomme.lft(I1);nomme.rt(I2);
fin;
```



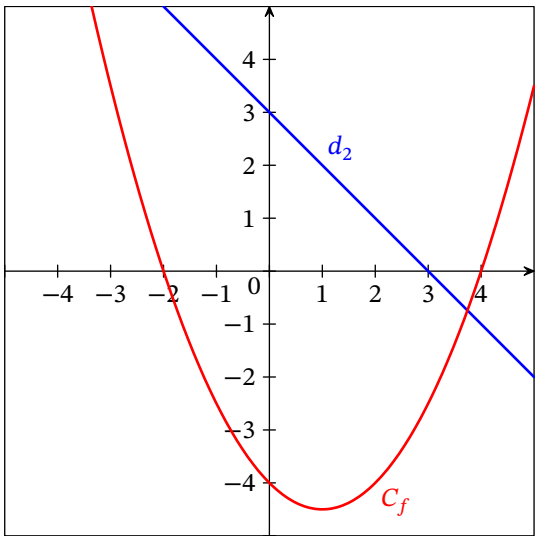
5.2 Nom des courbes

`nomme.<pos>(<p>,<x>,<nom>)`

Affiche nom au point d'abscisse x de la courbe p à la position pos. 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 43

```
repere(-5,5,0.7cm,-5,5,0.7cm);
path d,C_f;
d=droite(-1,3);
vardef f(expr x)=0.5(x**2)-x -4 enddef;
C_f=courbefonc(f)();
draw axes(1,1);
draw cadre;
drawoptions(withcolor blue);
draw d epaisseur 1;
nomme.urt(d,1,"$d_2$");
drawoptions(withcolor red);
draw C_f epaisseur 1;
nomme.lrt(C_f,2);
fin;
```



`nomme(<p>,<nom>)`

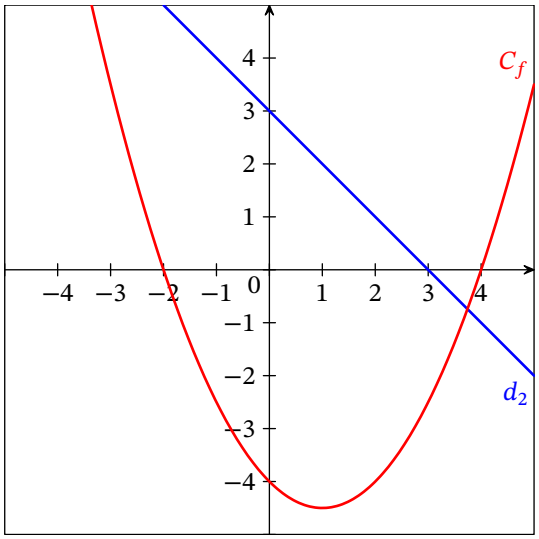
Affiche `nom` (qui peut être omis) au niveau d'un point d'intersection de `p` et du contour de la figure. La position est choisie automatiquement en fonction de la chaine `prefnomme` et de la place restante.

Paramètres

Nom	Type	Défaut	
`prefnomme`	`string`	`"right"`	indique le placement préféré du nom des courbes. Les valeurs possibles sont `"right"`, `"left"`, `"top"` ou `"bottom"`.

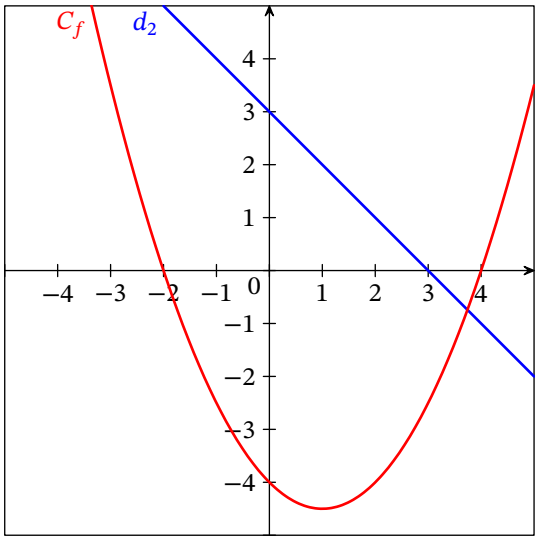
Exemple 44

```
repere(-5,5,0.7cm,-5,5,0.7cm);
path d,C_f;
d=droite(-1,3);
vardef f(expr x)=0.5(x**2)-x -4 enddef;
C_f=courbefonc(f)();
draw axes(1,1);
draw cadre;
drawoptions(withcolor blue);
draw d epaisseur 1;
nomme(d,"$d_2$");
drawoptions(withcolor red);
draw C_f epaisseur 1;
nomme(C_f);
fin;
```



Exemple 45

```
prefnomme:="top";
repere(-5,5,0.7cm,-5,5,0.7cm);
path d,C_f;
d=droite(-1,3);
vardef f(expr x)=0.5(x**2)-x -4 enddef;
C_f=courbefonc(f)();
draw axes(1,1);
draw cadre;
drawoptions(withcolor blue);
draw d epaisseur 1;
nomme(d,"$d_2$");
drawoptions(withcolor red);
draw C_f epaisseur 1;
nomme(C_f);
fin;
```



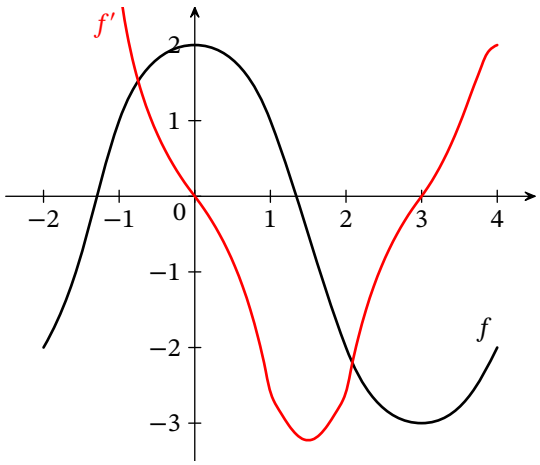
5.3 Dérivée et tangentes

`der(<p>)(<x>)` — `numeric`

Image de `x` par la dérivée de la fonction dont la courbe représentative est `p`.

Exemple 46

```
repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path f,f';
f=(-2,-2){dir 60}..(-1,1)
  ..(0,2){right}..(1,1)..(2,-2)
  ..(3,-3){right}..(4,-2){(1,2)};
vardef df(expr x)= der(f)(x) enddef;
f'= courbefonc(df)(-1,4);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw f;
nomme(f);
draw f' withcolor red;
nomme(f') withcolor red;
fin;
```



`tangente(<p>,<x>)` — `path`

Droite tangente à la courbe `p` au point d'abscisse `x`.

`tangente.gauche(<p>,<x>,<long>)` — `picture`

Flèche de longueur `long` représentant la demi-tangente gauche à la courbe `p` au point d'abscisse `x`. `long` est optionnel, la valeur par défaut est donnée par le paramètre `longtan`.

`tangente.droite(<p>,<x>,<long>)` — `picture`

Même chose à droite.

`tangente.gauche(<p>,<x>,<long>)` — `picture`

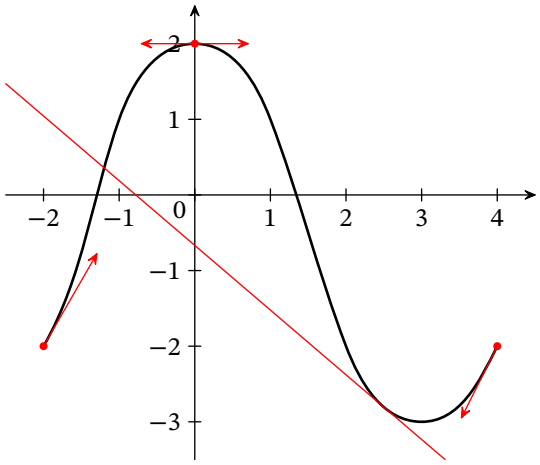
Même chose des deux côtés

Paramètres

Nom	Type	Défaut	
`longtan`	`numeric`	`20`	Longueur des « demi-flèches » dans les tracés de tangentes.

Exemple 47

```
repere(-2.5,4.5,1cm,-3.5,2.5,1cm);
path p;
p=(-2,-2){dir 60}..(-1,1)
  ..(0,2){right}..(1,1)..(2,-2)
  ..(3,-3){right}..(4,-2){(1,2)};
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw p;
drawoptions(withcolor red);
draw tangente.double(p,0);
draw tangente.droite(p,-2,40);
longtan:=30;
draw tangente.gauche(p,4);
draw tangente(p,2.5);
fin;
```



5.4 Interpolation

METAPOST propose les commandes suivantes (qui peuvent être combinées dans une même courbe) :

`A--B--C--` Ligne brisée passant par les points A, B, C...

`A..B..C..` Courbe de Bézier passant par les points A, B, C...

Interpolation polynomiale

Ce n'est bien sûr pas toujours la meilleure méthode d'approximation. Il est conseillé de compiler avec `numbersystem` réglé sur `decimal` pour gagner en précision.

`lagrange(<A>,<B>,<C>,...)()` `path`

Courbe passant par A, B, C... représentant le polynôme de degré maximal $n-1$ tel que $P(x_A)=y_A$, $P(x_B)=y_B$... sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`lagrange(<A>,<B>,<C>,...)(<xminf>,<xmaxf>)` `path`

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

`lagrange(<x1>,<y1>,<x2>,<y2>,<x3>,<y3>...)()` `path`

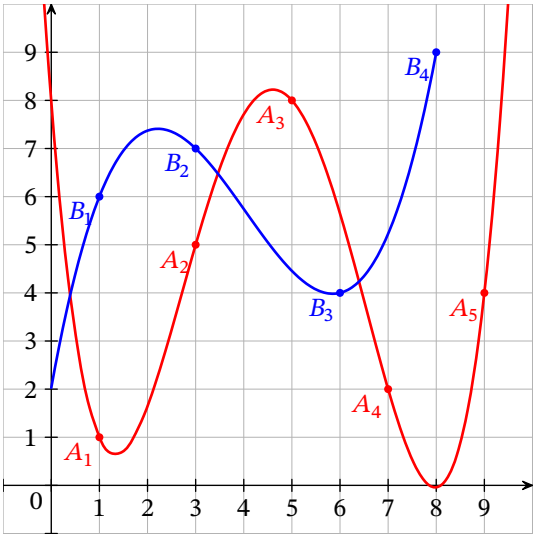
Courbe passant par les points $(x_1;y_1)$, $(x_2;y_2)$, $(x_3;y_3)$... représentant le polynôme de degré maximal $n-1$ tel que $P(x_i)=y_i$ sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`lagrange(<x1>,<y1>,<x2>,<y2>,<x3>,<y3>...)(<xminf>,<xmaxf>)` `path`

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

Exemple 48

```
repere.orth(-1,10,7cm,-1,10);
pair A[],B[];
A[1]=(1,1);A[2]=(3,5);A[3]=(5,8);
A[4]=(7,2);A[5]=(9,4);
B[1]=(1,6);B[2]=(3,7);B[3]=(6,4);B[4]=(8,9);
path L;L=lagrange(A[1],A[2],A[3],A[4],A[5])();
path C;C=lagrange(1,6,3,7,6,4,8,9)(0,8);
draw quadrillage(1,1);
draw axes(1,1);
drawoptions(withcolor red);
draw L epaisseur 1;
for i=1 upto 5: nomme.llft(A[i]); endfor
drawoptions(withcolor blue);
draw C epaisseur 1;
for i=1 upto 4: nomme.llft(B[i]); endfor
fin;
```



`hermite((<x1>,<y1>,<y'1>),(<x2>,<y2>,<y'2>)...)()` **path**

Courbe passant par les points $(x_1;y_1)$, $(x_2;y_2)$, $(x_3;y_3)$... représentant le polynôme de degré maximal $2n-1$ tel que $P(x_i)=y_i$ et $P'(x_i)=y'_i$ sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`hermite((<x1>,<y1>,<y'1>),(<x2>,<y2>,<y'2>)...)(<xminf>,<xmaxf>)` **path**

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

`hermite(<A>,<y'A>),(B,<y'B>)...)()` **path**

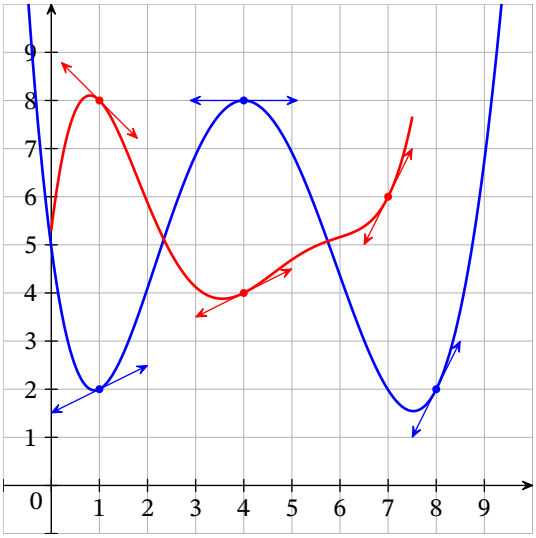
Courbe passant par les points A, B, C... représentant le polynôme de degré maximal $2n-1$ tel que $P(x_A)=y_A$ et $P'(x_A)=y'_A$... sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`hermite(<A>,<y'A>),(B,<y'B>)...)(<xminf>,<xmaxf>)` **path**

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

Exemple 49

```
repere.orth(-1,10,7cm,-1,10);
draw quadrillage(1,1);
draw axes(1,1);
path H;H=hermite((1,2,0.5),(4,8,0),(8,2,2))();
drawoptions(withcolor blue);
draw H withpen pencircle scaled 1;
draw tangente.double(H,1);
draw tangente.double(H,4);
draw tangente.double(H,8);
pair A,B,C; A:=(1,8);B:=(4,4);C:=(7,6);
path I;I=hermite(A,-1,B,0.5,C,2)(0,7.5);
drawoptions(withcolor red);
draw I withpen pencircle scaled 1;
draw tangente.double(I,1);
draw tangente.double(I,4);
draw tangente.double(I,7);
fin;
```



Interpolation à l'aide de splines cubiques

`spline(<A>,<B>,<C>...)()` **path**

Courbe passant par les points A, B, C représentant une fonction cubique par morceaux telle que $f(x_A) = y_A$, $f(x_B) = y_B$... sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`spline(<A>,<B>,<C>...)(<xminf>,<xmaxf>)` **path**

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

`spline(<xA>,<yA>,<xB>,<yB>,<xC>,<yC>...)()` **path**

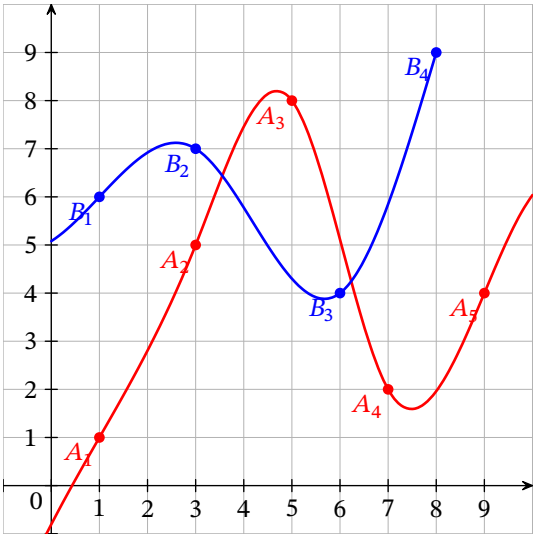
Même courbe que précédemment mais les valeurs sont données sous forme de liste.

`spline(<xA>,<yA>,<xB>,<yB>,<xC>,<yC>...)(<xminf>,<xmaxf>)` **path**

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

Exemple 50

```
repere.orth(-1,10,7cm,-1,10);
pair A[],B[];
A[1]=(1,1);A[2]=(3,5);A[3]=(5,8);
A[4]=(7,2);A[5]=(9,4);
B[1]=(1,6);B[2]=(3,7);B[3]=(6,4);B[4]=(8,9);
path L;L=spline(A[1],A[2],A[3],A[4],A[5])();
path C;C=spline(1,6,3,7,6,4,8,9)(0,8);
draw quadrillage(1,1);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw L withcolor red;
draw C withcolor blue;
for i=1 upto 5:
  nomme.llft(A[i]) withcolor red;
endfor
for i=1 upto 4:
  nomme.llft(B[i]) withcolor blue;
endfor
fin;
```



`spline(<A>,<y'A>,<B>,<y'B>,<C><y'C>,...)()` **path**

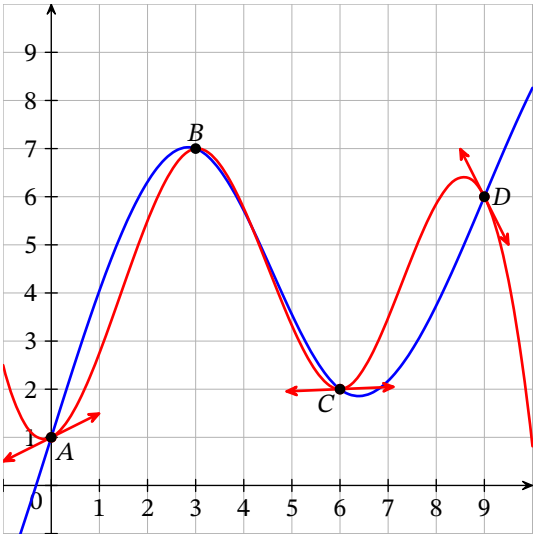
Courbe passant par les points A, B, C représentant une fonction cubique par morceaux telle que $f(x_A) = y_A$, $f'(x_A) = y'_A$, $f(x_B) = y_B$, $f'(x_B) = y'_B$... sur l'intervalle `[Xmin;Xmax]` définissant le repère. Les valeurs `y'A`, `y'B`... peuvent être omises

`spline(<A>,<y'A>,<B>,<y'B>,<C><y'C>,...)(<xminf>,<xmaxf>)` **path**

Même courbe que précédemment mais sur l'intervalle `[xminf;xmaxf]`.

Exemple 51

```
repere.orth(-1,10,7cm,-1,10);
pair A,B,C,D;
A=(0,1);B=(3,7);C=(6,2);D=(9,6);
path S,T;
S=spline(A,B,C,D)();
T=spline(A,0.5,B,C,0,D,-2)();
draw quadrillage(1,1);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw S withcolor blue;
draw T withcolor red;
draw tangente.double(T,0) withcolor red;
draw tangente.double(T,6) withcolor red;
draw tangente.double(T,9) withcolor red;
nomme.lrt(A);nomme.top(B);
nomme.llft(C);nomme.rt(D);
fin;
```



Attention : lorsque la courbure est importante, l'utilisation de `tangente` peut donner des résultats erronés...

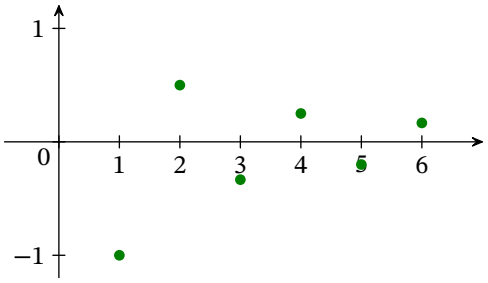
6 Suites

`suite(<u>,<deb>,<fin>)` `picture`

Figure formée des points $(i; u_i)$ pour i variant entre `deb` et `fin`.

Exemple 52

```
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 0.5green;
fin;
```

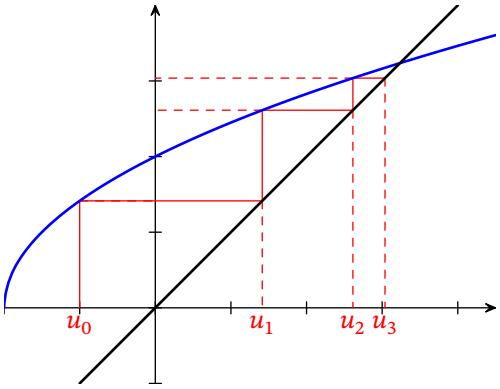


`Suiterec(f,deb,fin,init)` `picture`

Figure complète formée d'une 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}, des projections des différents termes sur les axes et des valeurs ou nom des termes. Des paramètres permettent de contrôler ce qui doit être affiché (voir plus bas).

Exemple 53

```
repere(-2,4.5,1cm,-1,4,1cm);
vardef f(expr x)=sqrt(2*x+4) enddef;
path C_f;
C_f= courbefonc(f)();
draw axes(1,0);
draw Suiterec(f,0,3,-1) withcolor red;
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor blue;
draw droite(1,0);
fin;
```

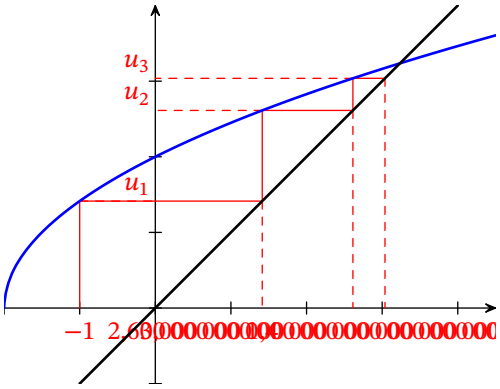


Paramètres

Nom	Type	Défaut	
suite_nom	string	"u"	Chaine indiquant le nom de la suite pour affichage
suite_affx	boolean	true	Booléen qui indique si les projections sur l'axe des abscisses doivent être tracées.
suite_affy	boolean	true	Booléen qui indique si les projections sur l'axe des ordonnées doivent être tracées.
suite_labx	string	"nom"	Chaine qui indique le type d'affichage des étiquettes sur l'axe des abscisses. Les valeurs sont "nom" (le nom du terme est affiché), "val" (la valeur du terme est affichée) ou "" (rien n'est affiché).
suite_laby	string	""	Chaine qui indique le type d'affichage des étiquettes sur l'axe des ordonnées. Les valeurs sont "nom" (le nom du terme est affiché), "val" (la valeur du terme est affichée) ou "" (rien n'est affiché).
suite_arrondi	numeric	1	Nombre qui indique, dans le cas où un des paramètres précédents vaut "val", la précision des valeurs à afficher.
suite_posx	string	"bot"	Chaine qui indique la position des étiquettes sur l'axe des abscisses.
suite_posy	string	"lft"	Chaine qui indique la position des étiquettes sur l'axe des ordonnées.
suite_styleproj	string	"dashed evenly"	Chaine qui indique le style des tracés des projections sur les axes.

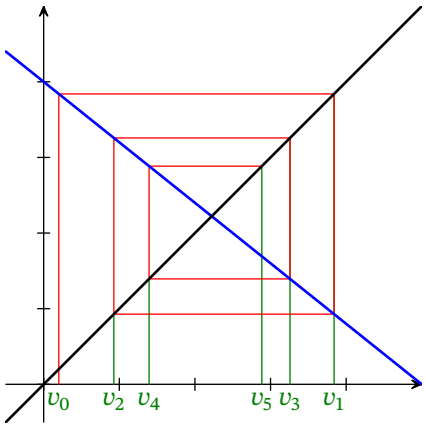
Exemple 54

```
repere(-2,4.5,1cm,-1,4,1cm);
vardef f(expr x)=sqrt(2*x+4) enddef;
path C_f;
C_f= courbefonc(f)();
draw axes(1,0);
suite_labx:="val";
suite_laby:="nom";
suite_posy:="ulft";
draw Suiterec(f,0,3,-1) withcolor red;
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor blue;
draw droite(1,0);
fin;
```



Exemple 55

```
repere(-0.5,5,1cm,-0.5,5,1cm);
vardef f(expr x)=4-0.8*x enddef;
path C_f;
C_f= courbefonc(f)();
draw axes(1,0);
suite_nom:="v";
suite_affy:=false;
suite_styleproj:="withcolor␣0.5green";
draw Suiterec(f,0,5,0.2) withcolor red;
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor blue;
draw droite(1,0);
fin;
```



7 Surfaces

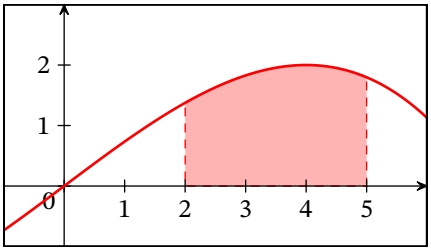
7.1 Calcul intégral

`souscourbe(<p>,<xmin>,<xmax>)` **(closed) path**

Chemin fermé délimitant la zone comprise entre la courbe p, l'axe des abscisses et les droites d'équations $y =$ xmin et $y =$ xmax.

Exemple 56

```
repere(-1,6,0.8cm,-1,3,0.8cm);
vardef f(expr x)= -(x/4)**3+0.75x enddef;
path C_f,q;
C_f:= courbefonc(f)();
q:=souscourbe(C_f,2,5);
fill q withcolor (1,0.7,0.7);
draw q dashed evenly withcolor red;
draw axes(1,1);
drawoptions( withpen pencircle scaled 1);
draw C_f withcolor red;
draw cadre;
fin;
```

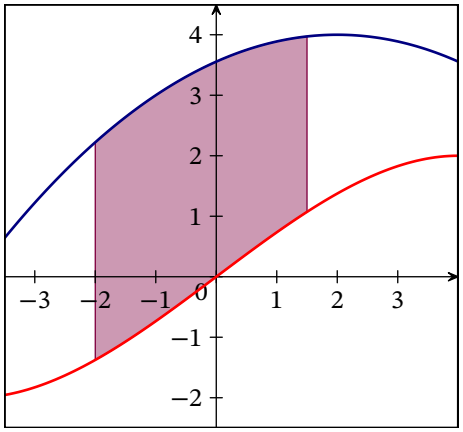


`entrecourbes(<p>,<q>,<xmin>,<xmax>)` (closed) path

Chemin fermé délimitant la zone comprise entre les courbes p et q et les droites d'équations $y =$ xmin et $y =$ xmax.

Exemple 57

```
repere(-3.5,4,0.8cm,-2.5,4.5,0.8cm);
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;
C_f:= courbefonc(f)();
C_g:= courbefonc(g)();
p:=entrecourbes(C_f,C_g,-2,1.5);
fill p withcolor (0.8,0.6,0.7);
draw p withcolor (0.5,0,0.25);
draw axes(1,1);
drawoptions(withpen pencircle scaled 1);
draw C_f withcolor red;
draw C_g withcolor 0.5blue;
draw cadre;
fin;
```

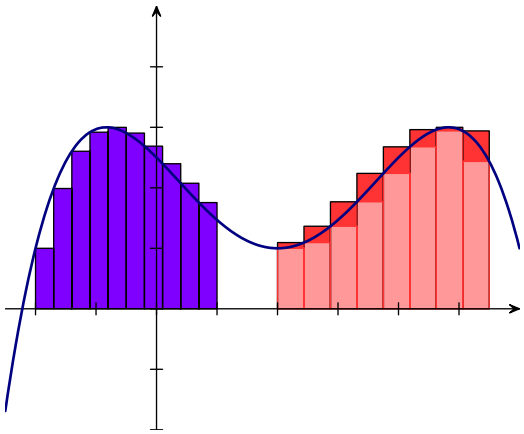


`rectangles.<type>(<p>,<a>,<b>,<n>)` (closed) path

Ligne brisée représentant la 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 58

```
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)();
r1=rectangles.droite(Cf,-2,1,10);
fill r1 withcolor 0.5red+blue;
draw r1;
r2=rectangles.max(Cf,2,5.5,8);
r3=rectangles.min(Cf,2,5.5,8);
fill r2 withcolor (1,0.2,0.2);
fill r3 withcolor (1,0.6,0.6);
draw r2;
draw r3 withcolor (1,0.2,0.2);
draw axes(1,0);
draw Cf withcolor 0.5blue
             withpen pencircle scaled 1;
fin;
```



7.2 Demi-plans

`demiplaninf(<d>)` `(closed) path`

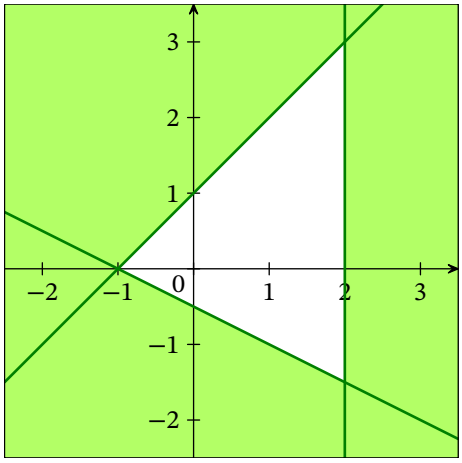
Chemin fermé délimité par la droite `d` et par la partie inférieure de `cadre` (ou la partie gauche si `d` est parallèle à l'axe des ordonnées).

`demiplansup(<d>)` `(closed) path`

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 59

```
repere(-2.5,3.5,1cm,-2.5,3.5,1cm);
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 (0.7,1,0.4);
endfor
draw axes(1,1);
drawoptions(withpen pencircle scaled 1
                  withcolor 0.5green);
draw d1;draw d2;draw d3;
drawoptions();
draw cadre;
fin;
```



8 Projections sur les axes

8.1 Projetés

`projetex(<A>)` `pair`

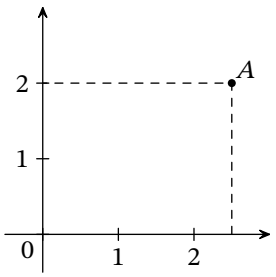
Projeté de `A` sur l'axe des abscisses parallèlement à l'axe des ordonnées.

`projetey(<A>)` `pair`

Projeté de `A` sur l'axe des ordonnées parallèlement à l'axe des abscisses.

Exemple 60

```
repere(-0.5,3,1cm,-0.5,3,1cm);
 pair A;
 A=(2.5,2);
 draw axes(1,1);
 nomme.urt(A);
 draw projetex(A)--A--projetey(A) dashed evenly;
fin;
```



`Projectionx.<pos>(<A>,<lab>)` `picture`

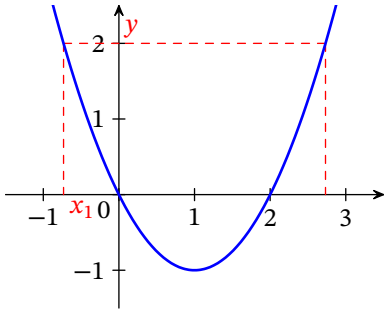
Figure constituée du segment joignant `A` à son projeté sur l'axe des abscisses ainsi que de l'étiquette `lab` placée à la position `pos` par rapport à ce projeté. L'étiquette est optionnelle, si elle est omise, rien n'est affiché.

`Projectiony.<pos>(<A>,<lab>)` **picture**

Figure constituée du segment joignant `A` à son projeté sur l'axe des ordonnées ainsi que de l'étiquette `lab` placée à la position `pos` par rapport à ce projeté. L'étiquette est optionnelle, si elle est omise, rien n'est affiché.

Exemple 61

```
repere(-1.5,3.5,1cm,-1.5,2.5,1cm);
path Cf; pair A[];
vardef f(expr x)= x**2-2x enddef;
Cf= courbefonc(f)();
ptantecedents(A,2,Cf);
draw axes(1,1);
drawoptions(dashed evenly withcolor red);
draw Projectionx.lrt(A1,"$x_1$");
draw Projectionx(A2);
draw Projectiony(A1);
draw Projectiony.urt(A2,"$y$");
drawoptions();
draw Cf withpen pencircle scaled 1 withcolor blue;
fin;
```

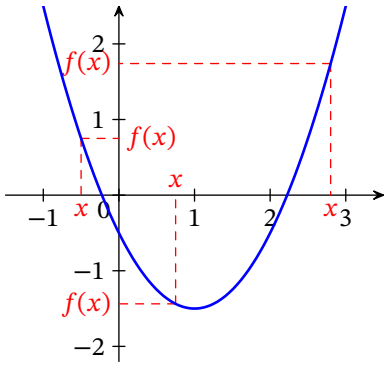


`Projectionaxes(<A>,<labx>,<laby>)` **picture**

Figure constituée des segments joignant `A` à ses projetés sur les axes ainsi que des étiquettes `labx` et `laby` positionnées automatiquement par rapport aux axes. Les étiquettes sont optionnelles.

Exemple 62

```
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)();
A1=(-0.5,f(-0.5));A2=(2.8,f(2.8));
A3=(0.75,f(0.75));
draw axes(1,1);
drawoptions(dashed evenly withcolor red);
draw Projectionaxes(A1,"$x$","$f(x)$");
draw Projectionaxes(A2,"$x$","$f(x)$");
draw Projectionaxes(A3,"$x$","$f(x)$");
drawoptions();
draw Cf withpen pencircle scaled 1 withcolor blue;
fin;
```



8.2 Intervalles

`intervallex.<bornes>(<a>,<b>)` `picture`

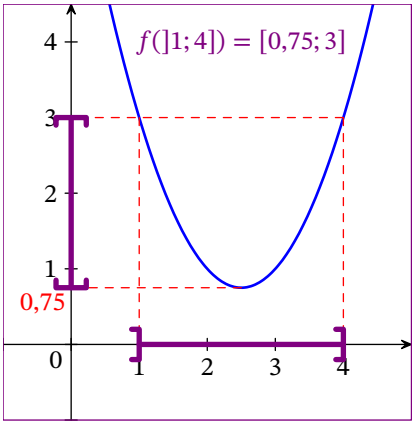
Intervalle dessiné sur l'axe des abscisses entre `a` et `b` avec une épaisseur par défaut de `1.5bp`. `bornes` peut être `OO` (ouvert à gauche, ouvert à droite), `OF`, `FO` ou `FF`.

`intervalley.<bornes>(<a>,<b>)` `picture`

Même chose sur l'axe des ordonnées.

Exemple 63

```
repere(-1,5,0.9cm,-1,4.5,1cm);
vardef f(expr x)=x**2-5x+7 enddef;
draw axes(1,1);
drawoptions(withcolor blue);
draw courbefonc(f)() withpen pencircle scaled 1;
drawoptions(dashed evenly withcolor red);
draw projectionx.bot((1,f(1)));
draw projectiony.llft((2.5,f(2.5)),"0,75");
draw projectionaxes((4,f(4)));
drawoptions(withcolor (0.5,0,0.5));
draw intervallex.OF(1,4);
draw intervalley.FF(0.75,3);
label("$f(]1;4])=[0,75;3]$",(2.5,4));
draw cadre;
fin;
```



Paramètres

Nom	Type	Défaut	
`int_ep`	`numeric`	`2bp`	Épaisseur du tracé des intervalles.

9 Statistiques et probabilités

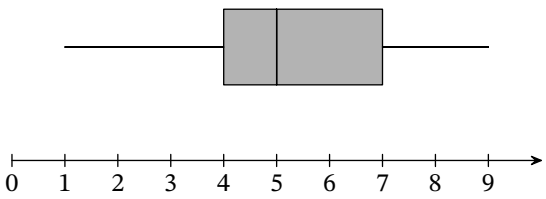
9.1 Boite à moustache

`diagrammeboite(<min>,<q1>,<med>,<q3>,<max>)` `(closed) path`

Diagramme en boite ou « boite à moustache » correspondant aux données en argument. Il s'agit d'un chemin fermé et peut donc être rempli. Les données sont optionnelles. En cas d'absence, les données du diagramme précédent sont utilisées.

Exemple 64

```
repere(-0.5,10,0.7cm,-1,5,1cm);
 setaxes(0,10,0,1);
 draw axex(1,1);
 fill diagrammeboite(1,4,5,7,9)
                      withcolor 0.7white;
 draw diagrammeboite();
fin;
```



Paramètres

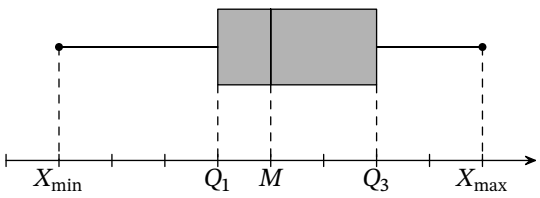
Nom	Type	Défaut	
boite_dec	numeric	1.5	Valeur qui indique le décalage de la boite par rapport à l'axe des abscisses (l'unité est celle de l'axe des ordonnées).
boite_larg	numeric	1	Valeur qui indique la largeur de la boite (l'unité est celle de l'axe des ordonnées).

`Diagrammeboite(<min>,<q1>,<med>,<q3>,<max>)` `picture`

Figure complète qui contient le dessin du diagramme ainsi que les projections des points importants avec éventuellement une étiquette. Des paramètres contrôlent ce qui doit être affiché. Les données sont optionnelles. En cas d'absence, les données du diagramme précédent sont utilisées.

Exemple 65

```
repere(-0.5,10,0.7cm,-1,5,1cm);
 setaxes(0,10,0,1);
 draw axex(1,0);
 fill diagrammeboite(1,4,5,7,9)
                      withcolor 0.7white;
 draw Diagrammeboite();
fin;
```



Paramètres

Nom	Type	Défaut	
boite_points	boolean	true	Booléen qui indique si les points aux extrémités du diagramme doivent être marqués.

Paramètres

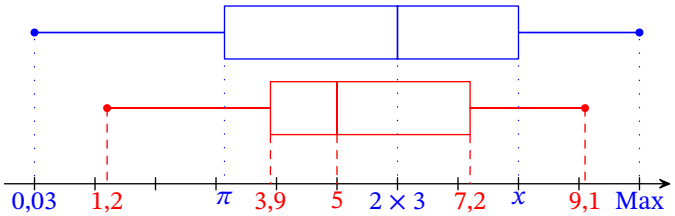
Nom	Type	Défaut	
boite_proj	boolean	true	Booléen qui indique si les projection doivent être tracées.
boite_lab	string	"nom"	Chaine qui indique le type d'affichage des étiquettes sur l'axe des abscisses. Les valeurs sont "nom" (le nom est affiché, voir ci-dessous), "val" (la valeur du terme est affichée) ou "" (rien n'est affiché).
boite_pos	string	"bot"	Chaine qui indique la position des étiquettes sur l'axe des abscisses.
boite_styleproj	string	"dashed evenly"	Chaine qui indique le style des tracés des projections sur les axes.

`def_boite_lab(<ch1>,<ch2>,<ch3>,<ch4>,<ch5>)`

Fonction qui permet de définir les 5 chaines qui seront affichées en abscisses si le paramètre `boite_lab` vaut "nom".
Cette fonction est appelée par défaut avec les paramètres "X_{min}", "Q_1", "M", "Q_3", "X_{max}".

Exemple 66

```
repere(-0.5,10.5,0.8cm,-1,5,1cm);
 draw axex(1,0);
 boite_larg:=0.7;
 drawoptions(withcolor red);
 boite_dec:=1;
 boite_lab:="val";
 draw Diagrammeboite(1.2,3.9,5,7.2,9.1);
 drawoptions(withcolor blue);
 boite_dec:=2;
 boite_lab:="nom";
 def_boite_lab(0.03,"$\pi$","$\numproduct{2x3}$","$x$","Max");
 boite_styleproj:="dashed␣withdots";
 draw Diagrammeboite(0,3.14,6,8,10);
fin;
```



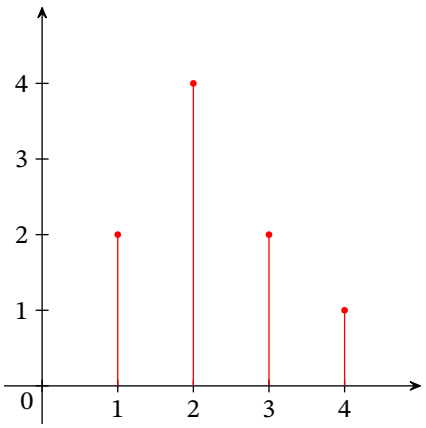
9.2 Diagrammes divers

`diagrammebatons((<v1>,<e1>),(<v2>,<e2>),...,(<vn>,<en>))` `picture`

Figure formée des n segments joignant les points `(v1,e1),(v2,e2),...(vn,en)` et leur projeté sur l'axe des abscisses. Les bâtons sont surmontés d'un point.

Exemple 67

```
repere(-0.5,5,1cm,-0.5,5,1cm);
 picture diag;
 draw axes(1,1);
 diag:=diagrammebatons((1,2),(2,4),(3,2),(4,1));
 draw diag withcolor red;
fin
```

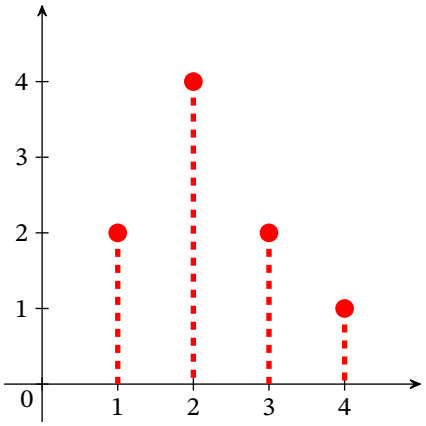


Paramètres

Nom	Type	Défaut	
`batons_diampoint`	`numeric`	`5`	Diamètre des points sur les bâtons.

Exemple 68

```
repere(-0.5,5,1cm,-0.5,5,1cm);
 picture diag;
 draw axes(1,1);
 batons_diampoints:=10;
 diag:=diagrammebatons((1,2),(2,4),(3,2),(4,1));
 draw diag withpen pencircle scaled 2
                  withcolor red dashed evenly;
fin
```

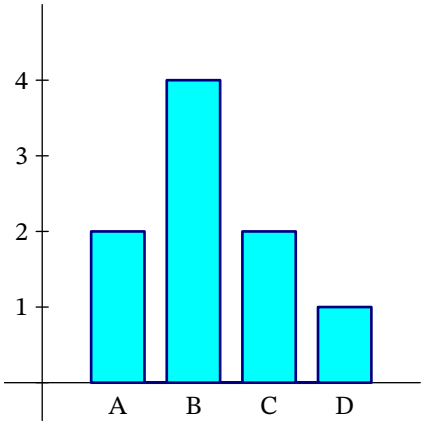


`diagrammebarres((<a1>,<h1>),(<a2>,<h2>),...(<an>,<hn>))` `(closed) path`

Chemin fermé qui forme n barres rectangulaires de hauteurs `h1` ...`hn` aux abscisses `a1` ...`an`.

Exemple 69

```
repere(-0.5,5,1cm,-0.5,5,1cm);
 gradxpart:=false;
 flecheaxe:=false;
 draw axey(1,1);
 draw axex(0,0);
 draw axexpart.bot(1,"A",2,"B",3,"C",4,"D");
 path diag;
 diag:=diagrammebarres((1,2),(2,4),(3,2),(4,1));
 fill diag withcolor (0,1,1);
 draw diag withpen pencircle scaled 1
                              withcolor 0.5blue;
fin
```

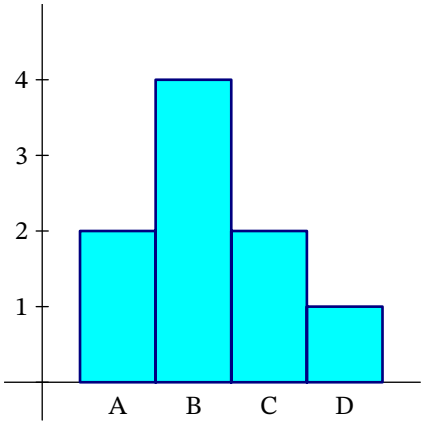


Paramètres

Nom	Type	Défaut	
barres_larg	numeric	20	Largeur des barres des diagrammes en barres.

Exemple 70

```
repere(-0.5,5,1cm,-0.5,5,1cm);
 gradxpart:=false;
 flecheaxe:=false;
 barres_larg:=1cm;
 draw axey(1,1);
 draw axex(0,0);
 draw axexpart.bot(1,"A",2,"B",3,"C",4,"D");
 path diag;
 diag:=diagrammebarres((1,2),(2,4),(3,2),(4,1));
 fill diag withcolor (0,1,1);
 draw diag withpen pencircle scaled 1
                              withcolor 0.5blue;
fin
```



9.3 Probabilités

Calculs

Quelques fonctions mathématiques sont proposées. Pour les grandes valeurs, on dépasse rapidement les capacités de METAPOST. Il est dans ce cas conseillé de compiler en utilisant la ligne de commande `mpost -numbersystem="decimal" <fichier>.mp` ou, avec LuaLaTeX et le package `luamplib`, d'utiliser `\mplibnumbersystem{decimal}`.

`factorielle(<n>)` `numeric`

Entier égal à $n!$.

`binom(<n>,<k>)` `numeric`

Entier égal à $\binom{n}{k}$.

`binomiale(<n>,<p>,<k>)` `numeric`

$P(X = k)$ pour X suivant la loi binomiale de paramètres n et p.

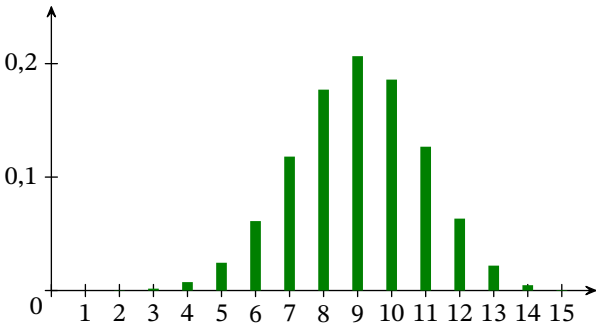
Diagrammes

`diagrammebinomiale(<n>,<p>)` `picture`

Diagramme en bâtons de la loi binomiale de paramètres n et p.

Exemple 71

```
repere(-2,16,0.45cm,-0.1,0.25,15cm);
setrepere(0,16,0,0.25);
draw axex(1,1);
draw axey(0.1,0.1);
picture diag;
batons_diampoints:=0;
diag:=diagrammebinomiale(15,0.6);
draw diag withcolor 0.5green
              withpen pencircle scaled 4;
fin;
```



`diagrammeuniforme(<n>,<m>)` `picture`

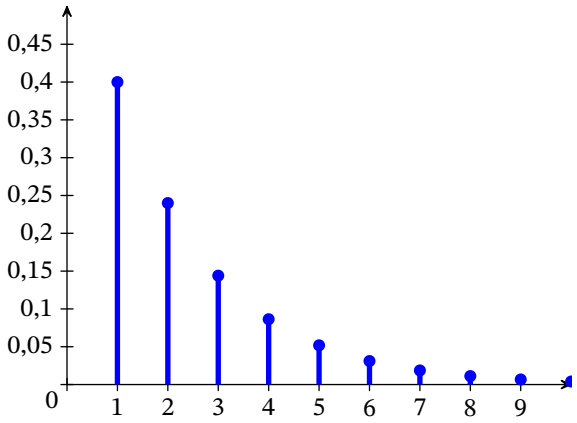
Diagramme en bâtons de la loi uniforme discrète sur les entiers consécutifs de `n` à `m`.

`diagrammegeometrique(<p>)` `picture`

Diagramme en bâtons de la loi géométrique de paramètre `p`.

Exemple 72

```
repere.larg(-2,10,8cm,-0.1,0.5,6cm);
 setrepere(0,10,0,0.5);
 draw axex(1,1);
 draw axey(0.05,0.05);
 draw diagrammegeometrique(0.4)
              withpen pencircle scaled 2
              withcolor blue;
fin;
```

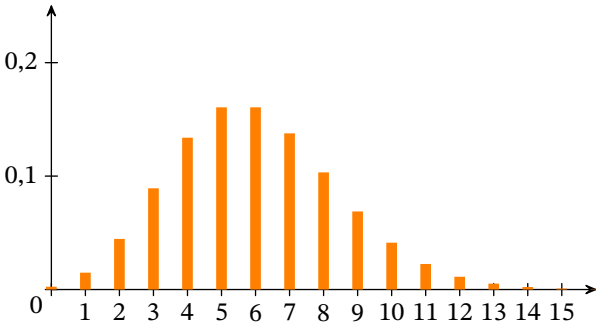


`diagrammepoisson(<lambda>)` `picture`

Diagramme en bâtons de la loi de Poisson de moyenne `lambda`.

Exemple 73

```
repere(-2,16,0.45cm,-0.1,0.25,15cm);
setrepere(0,16,0,0.25);
draw axex(1,1);
draw axey(0.1,0.1);
picture diag;
batons_diampoints:=0;
diag:=diagrammepoisson(6);
draw diag withcolor (1,0.5,0)
            withpen pencircle scaled 4;
fin;
```



`densitenormale(<mu>,<sigma>)()` `path`

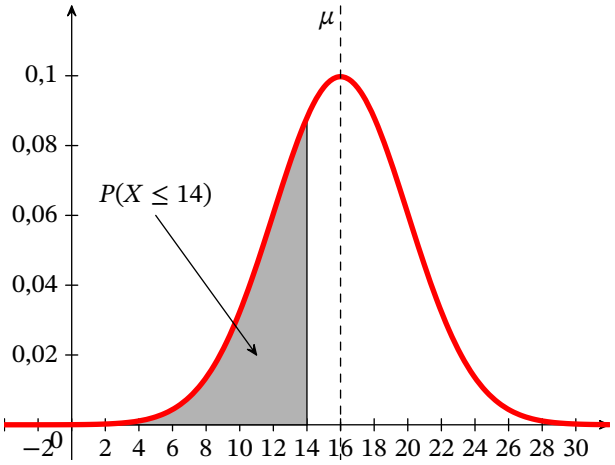
Courbe représentant la densité de la loi normale de moyenne `mu` et d'écart type `sigma` sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`densitenormale(<mu>,<sigma>)(<xminf>,<xmaxf>)` `path`

Courbe représentant la densité de la loi normale de moyenne `mu` et d'écart type `sigma` sur l'intervalle `[xminf;xmaxf]`.

Exemple 74

```
repere.larg(-4,32,8cm,-0.01,0.12,6cm);
draw axex(2,2);
draw axey(0.02,0.02);
path C,d;
C=densitenormale(16,4)();
fill souscourbe(C,0,14)
                    withcolor 0.7white;
draw souscourbe(C,0,14);
draw C withpen pencircle scaled 2
                         withcolor red;
d=droite(16);
draw d dashed evenly;
drawarrow (5,0.06)--(11,0.02);
label.top("$P(X\leq 14)$",(5,0.06));
nomme(d,"$\mu$");
fin;
```



`densiteexponentielle(lambda)()` **path**

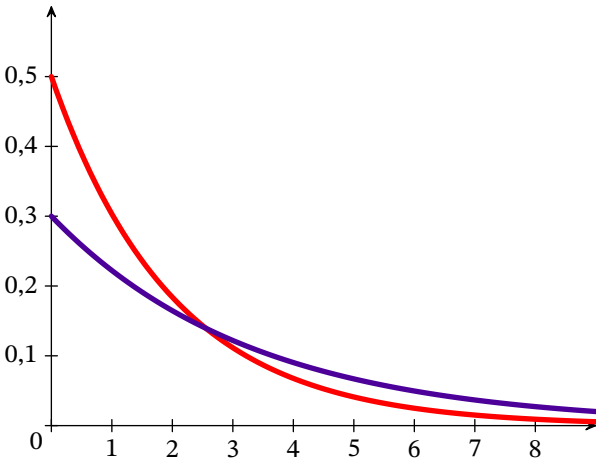
Courbe représentant la densité de la loi exponentielle de paramètre `lambda` sur l'intervalle `[Xmin;Xmax]` définissant le repère.

`densiteexponentielle(lambda)(<xminf>,<xmaxf>)` **path**

Courbe représentant la densité de la loi exponentielle de paramètre `lambda` sur l'intervalle `[xminf;xmaxf]`.

Exemple 75

```
repere.larg(-1,9,8cm,-0.05,0.6,6cm);
  setrepere(0,9,0,0.6);
  draw axex(1,1);
  draw axey(0.1,0.1);
  path C,D;
  C=densiteexponentielle(0.5)();
  D=densiteexponentielle(0.3)();
  drawoptions(withpen pencircle scaled 2);
  draw C withcolor red;
  draw D withcolor (0.3,0,0.6);
fin;
```



10 Géométrie

Certaines des macros suivantes sont inspirées des macros de `geometriesyr16.mp` de Christophe POULAIN disponible ici.

10.1 Codage des segments et des angles

Segments

`marquesegment(<A>,<B>)` `picture`

Figure formée d'une marque sur le segment $[AB]$. Des paramètres permettent de contrôler la forme et la taille de la marque.

`marquesegment(<p>)` `picture`

Figure formée d'une marque sur le chemin `p`. Des paramètres permettent de contrôler la forme et la taille de la marque.

`marquesegment(<A>,<B>,<C>,<D>...)` `picture`

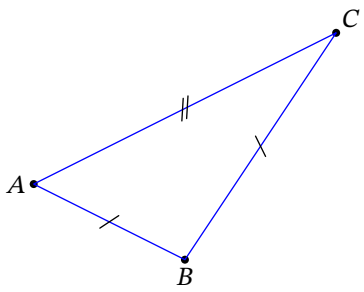
Plusieurs segments peuvent être marqués simultanément.

`marquesegment(<A>,<B>,<n>)` `picture`

Figure formée de `n` marques sur le segment $[AB]$. L'espace entre les marques est contrôlé par le paramètre `sep_marque_s`.

Exemple 76

```
repere();
pair A,B,C;
A=(1,2);B=(3,1);C=(5,4);
nomme.lft(A);nomme.urt(C);
nomme.bot(B);
draw triangle(A,B,C) withcolor blue;
draw marquesegment(A,B,B,C);
draw marquesegment(A,C,2);
fin;
```



`marquesegment(<A>,<B>,<formemarque>)` `picture`

Figure formée d'une marque sur le segment $[AB]$. Le paramètre `formemarque` indique la forme du dessin de la marque. Il peut aussi être donné de manière globale.

Paramètres

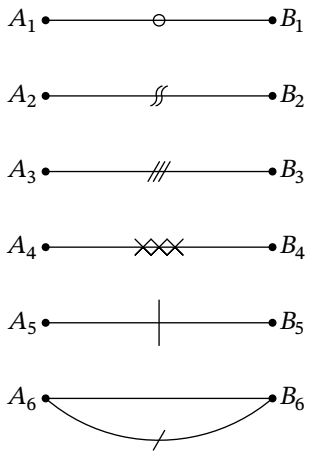
Nom	Type	Défaut	
`forme_marque_s`	`string`	`"/"`	Chaine de caractères permettant de choisir la forme et le nombre de marques. Les valeurs possibles sont : `"/"` (ou `"//"`, ou `"///"`...), `"x"` (ou `"xx"`...), `"o"`, `"s"`.

Paramètres

Nom	Type	Défaut	
echelle_marque_s	numeric	1	Facteur permettant de contrôler la taille des marques.
angle_marque_s	numeric	60	Angle de la marque par rapport au segment dans les cas "/" et "s".
sep_marque_s	numeric	2	Écart entre les marques dans le cas où il y en a plusieurs.

Exemple 77

```
repere();
pair A[],B[];
path p;
for i=1 upto 6:
  A[i]:=(0,-i);B[i]:=(3,-i);
  draw A[i]--B[i]; nomme.lft(A[i]);nomme.rt(B[i]);
endfor
p:=A[6]..(1,-6.5)..B[6];
draw marquesegment(A[1],B[1],"o");
draw marquesegment(A[2],B[2],"ss");
draw marquesegment(A[3],B[3],"/",3);
sep_marque_s:=6;
draw marquesegment(A[4],B[4],"xxx");
angle_marque_s:=90;echelle_marque_s:=2;
draw marquesegment(A[5],B[5],"/");
sep_marque_s:=2;
angle_marque_s:=60;echelle_marque_s:=1;
draw p;
draw marquesegment(p,"/");
fin;
```



Angles

`marqueangle(<A>,<O>,<B>,<n>)` **(closed) path**

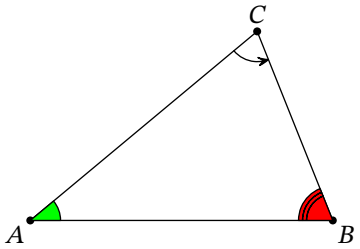
Chemin formé de `n` arcs de cercle de centre O permettant de marquer l'angle géométrique $\widehat{AOB}$. Il s'agit d'un chemin fermé qui peut donc être rempli.

`marqueangle(<A>,<O>,<B>)` **path**

Arc de cercle de centre `O` permettant de marquer l'angle $\widehat{AOB}$. On peut utiliser `drawarrow` pour marquer un angle orienté..

Exemple 78

```
repere();
pair A,B,C;
A=(0,0);B=(4,0);C=(3,2.5);
draw A--B--C--cycle;
nomme.llft(A);nomme.lrt(B);nomme.top(C);
fill marqueangle(C,B,A,3) withcolor red;
draw marqueangle(C,B,A,3);
drawarrow marqueangle(A,C,B);
fill marqueangle(B,A,C,1) withcolor green;
draw marqueangle(B,A,C,1);
fin;
```

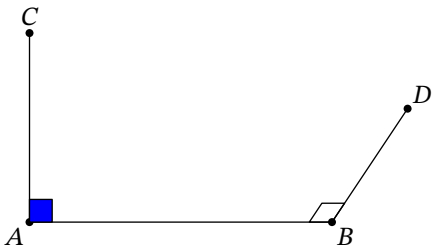


`marqueangledroit(<A>,<O>,<B>)` `(closed) path`

Chemin fermé permettant de marquer l'angle droit $\widehat{AOB}$ sous forme d'un losange (il s'agit donc d'un carré si l'angle est réellement droit).

Exemple 79

```
repere();
pair A,B,C,D;
A=(0,0);B=(4,0);C=(0,2.5);D=(5,1.5);
draw C--A--B--D;
nomme.llft(A);nomme.lrt(B);
nomme.top(C);nomme.urt(D);
fill marqueangledroit(B,A,C) withcolor blue;
draw marqueangledroit(B,A,C);
draw marqueangledroit(D,B,A);
fin;
```



Paramètres

Nom	Type	Défaut	
`echelle_marque_ad`	`numeric`	`1`	Facteur permettant de contrôler la taille des marques des angles droits.

`nomme.pos(<A>,<O>,<B>,<texte>)`

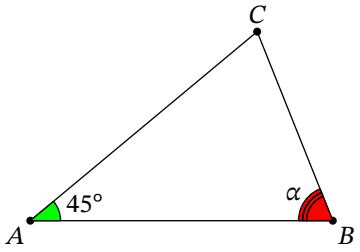
Place le texte à la position `pos` par rapport au point central de l'arc de cercle de centre O et de rayon `taille_marque_a`. `pos` peut être `rt`, `urt`, `top`, etc. ou un angle donné par rapport à la direction (Ox).

`nomme(<A>,<O>,<B>,<texte>)`

Même chose que précédemment mais la position est calculée automatiquement en fonction de l'angle.

Exemple 80

```
repere();
pair A,B,C;
A=(0,0);B=(4,0);C=(3,2.5);
draw A--B--C--cycle;
nomme.llft(A);nomme.lrt(B);nomme.top(C);
fill marqueangle(C,B,A,3) withcolor red;
draw marqueangle(C,B,A,3);
fill marqueangle(B,A,C,1) withcolor green;
draw marqueangle(B,A,C,1);
nomme[15](B,A,C,"\ang{45}");
nomme(C,B,A,"$\alpha$");
fin;
```



`nomme.pos(<A>,<O>,<B>,<texte>,<n>)`

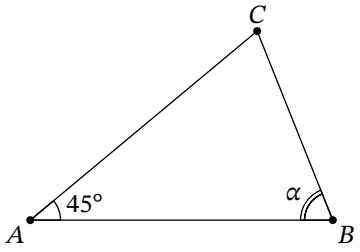
Même chose que précédemment avec en plus le tracé de `n` arcs de cercle pour marquer l'angle.

`nomme(<A>,<O>,<B>,<texte>,<n>)`

Même chose que précédemment mais la position est calculée automatiquement en fonction de l'angle.

Exemple 81

```
repere();
pair A,B,C;
A=(0,0);B=(4,0);C=(3,2.5);
draw A--B--C--cycle;
nomme.llft(A);nomme.lrt(B);nomme.top(C);
nomme[15](B,A,C,"\ang{45}",1);
nomme(C,B,A,"$\alpha$",2);
fin;
```



10.2 Cotes

`cote(<A>,<B>,<texte>)`

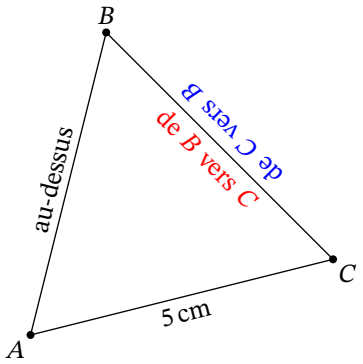
Figure formée du texte orienté dans la direction du segment $[AB]$, placé au niveau du milieu et situé « sous » le segment.

`cote.top(<A>,<B>,<texte>)`

Même chose mais le texte est placé au-dessus du segment.

Exemple 82

```
repere(0,6,1cm,0,6,1cm);
pair A,B,C;
A=(1,1);B=(2,5);C=(5,2);
nomme.llft(A);nomme.top(B);nomme.lrt(C);
draw triangle(A,B,C);
cote(A,C,"\SI{5}{cm}");
cote(B,C,"de␣$B$␣vers␣$C$") withcolor red;
cote(C,B,"de␣$C$␣vers␣$B$") withcolor blue;
cote.top(A,B,"au-dessus");
fin;
```



`cotefleche(<A>,<B>,<texte>)`

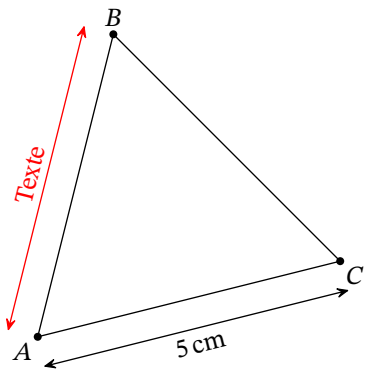
Figure formée d'une double flèche et du texte orienté dans la direction du segment $[AB]$, placés au niveau du milieu et situés « sous » le segment.

`cotefleche.top(<A>,<B>,<texte>)`

Même chose mais la double flèche et le texte sont placés au-dessus du segment.

Exemple 83

```
repere(0,6,1cm,0,6,1cm);
pair A,B,C;
A=(1,1);B=(2,5);C=(5,2);
nomme.llft(A);nomme.top(B);nomme.lrt(C);
draw triangle(A,B,C);
cotefleche(A,C,"\SI{5}{cm}");
cotefleche.top(A,B,"Texte") withcolor red;
fin;
```



Paramètres

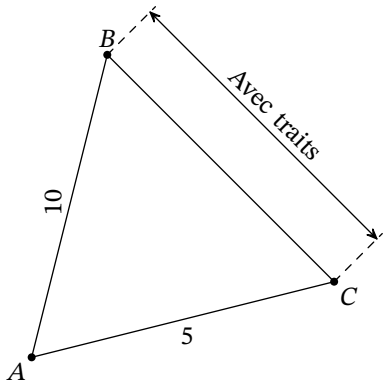
Nom	Type	Défaut	
`angle_cote`	`numeric`	`-1`	Variable numérique qui fixe l'angle de rotation de l'étiquette. La valeur par défaut de -1 indique que l'étiquette est tournée en fonction de l'angle du segment.

Paramètres

Nom	Type	Défaut	
dec_cote	numeric	4mm	Variable numérique qui indique le décalage entre le segment et la double flèche.
traits_cote	boolean	false	Variable booléenne qui indique si des traits délimitant la double flèche doivent être tracés.

Exemple 84

```
repere(0,6,1cm,0,6,1cm);
pair A,B,C;
A=(1,1);B=(2,5);C=(5,2);
nomme.llft(A);nomme.top(B);nomme.lrt(C);
draw triangle(A,B,C);
angle_cote:=90;
cote.top(A,B,"10");
angle_cote:=0;
cote(A,C,"5");
angle_cote:=-1;
traits_cote:=true;
dec_cote:=0.8cm;
cotefleche.top(B,C,"Avec␣traits");
fin;
```



10.3 Polygones

`polygone(<A>,<B>,<C>,...)` (closed) path

Chemin fermé représentant le polygone *ABC*...

`triangle(<A>,<B>,<C>)` (closed) path

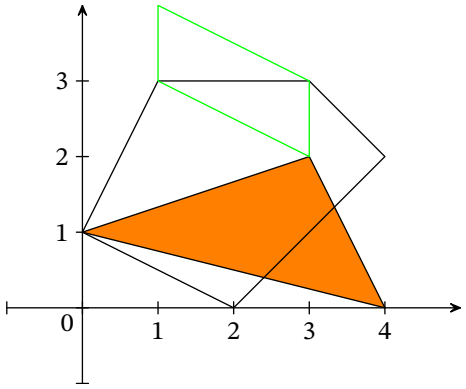
Cas particulier du précédent. Chemin fermé représentant le triangle *ABC*.

`parallelogramme(<A>,<B>,<C>)` (closed) path

Chemin fermé représentant *ABCD* où *D* est le quatrième point du parallélogramme.

Exemple 85

```
repere(-1,5,1cm,-1,4,1cm);
draw axes(1,1);
pair A,B,C,D,E,F,G;
A=(0,1);B=(2,0);C=(4,2);D=(3,3);E=(1,3);
F=(4,0);G=(3,2);
fill triangle(A,F,G) withcolor (1,0.5,0);
draw triangle(A,F,G);
draw polygone(A,B,C,D,E);
draw parallelogramme(D,G,E) withcolor green;
fin;
```



`polygoneregulier(<A>,<B>,<n>)` `(closed) path`

Chemin fermé représentant le polygone régulier de sens direct à n côtés dont un des côtés est $[AB]$.

`equilateral(<A>,<B>)` `(closed) path`

Cas particulier du précédent. Triangle équilatéral de sens direct de côté $[AB]$.

`carre(<A>,<B>)` `(closed) path`

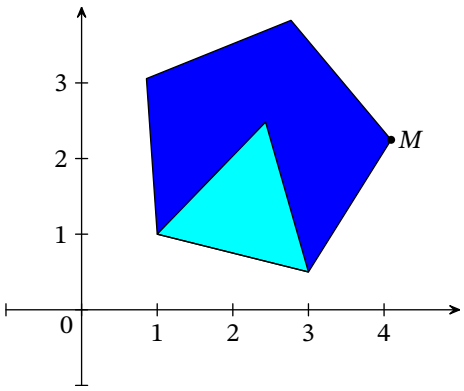
Autre cas particulier. Carré de sens direct de côté $[AB]$.

`sommetpolygoneregulier(<A>,<B>,<n>,<i>)` `pair`

Sommet numéro i du polygone régulier à n côtés dont un des côtés est $[AB]$. A est le sommet numéro 1 et B est le sommet numéro 2.

Exemple 86

```
repere(-1,5,1cm,-1,4,1cm);
draw axes(1,1);
pair A,B,M;
A=(1,1);B=(3,0.5);
fill polygoneregulier(A,B,5) withcolor blue;
fill equilateral(A,B) withcolor (0,1,1);
draw polygoneregulier(A,B,5);
M=sommetpolygoneregulier(A,B,5,3);
nomme.rt(M);
draw equilateral(A,B);
fin;
```



10.4 Cercles et arcs

`cercle(<A>,<B>,<C>)` `(closed) path`

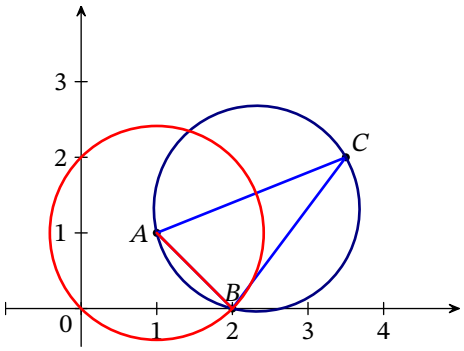
Cercle circonscrit au triangle ABC.

`cercle(<O>,<A>)` `(closed) path`

Cercle de centre O passant par A.

Exemple 87

```
repere(-1,5,1cm,-0.5,4,1cm);
draw axes(1,1);
pair A,B,C;
A=(1,1);B=(2,0);C=(3.5,2);
nomme.lft(A);nomme.urt(C);nomme.top(B);
drawoptions(withpen pencircle scaled 1);
draw triangle(A,B,C) withcolor blue;
draw cercle(A,B,C) withcolor 0.5blue;
draw A--B withcolor red;
draw cercle(A,B) withcolor red;
fin;
```



`cercle(<O>,<r>)` `(closed) path`

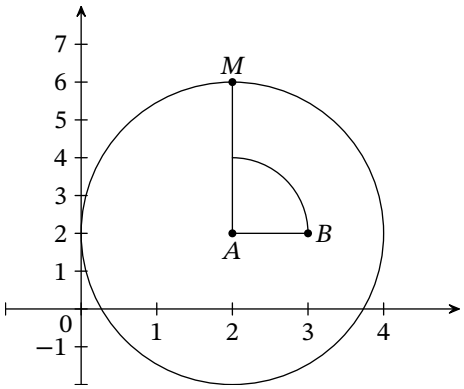
Cercle de centre O et de rayon r. L'unité de longueur est l'unité de l'axe des abscisses.

`arccercle(<A>,<O>,<B>)` `path`

Arc de cercle de sens direct de centre O, passant par A et s'appuyant sur la demi-droite $[OB)$.

Exemple 88

```
repere(-1,5,1cm,-2,8,0.5cm);
draw axes(1,1);
pair A,M,B;
A=(2,2);M=(2,6);B=(3,2);
nomme.bot(A);nomme.rt(B);nomme.top(M);
draw B--A--M;
draw cercle(A,2);
draw arccercle(B,A,M);
fin;
```



10.5 Figures complètes

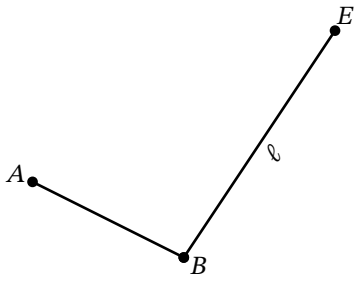
Les figures « complètes » sont des figures composées d'un chemin, du nom des points et éventuellement des cotes et angles. Pour les distinguer des chemins, leur nom commence par une capitale.

`Segment(<A>,<B>)(<points>)(<cote>)` `picture`

Figure formée du segment $[AB]$ et des noms des points, et éventuellement des cotes. Les étiquettes sont composées automatiquement mais peuvent être modifiées en passant des valeurs à `(points)` ou `(cote)`.

Exemple 89

```
repere();
pair A,B,C;
A=(0,0);B=(2,-1);C=(4,2);
drawoptions(withpen pencircle scaled 1);
draw Segment(A,B)()();
draw Segment(C,B)("$E$","")("$\ell$");
fin;
```

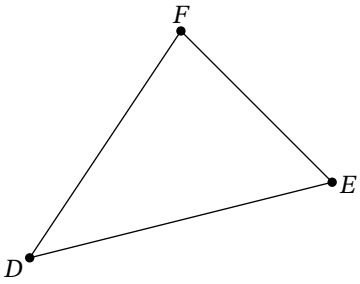


`Triangle(<A>,<B>,<C>)(<points>)(<cotes>)(<angles>)` picture

Figure formée du triangle *ABC* et des noms des points, et éventuellement des cotes et angles. Les étiquettes sont composées automatiquement mais peuvent être modifiées en passant des valeurs à `(points)`, `(cotes)` ou `(angles)`.

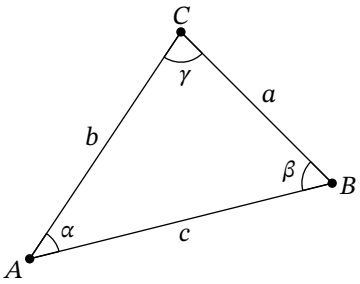
Exemple 90

```
repere();
pair D,E,F;
D:=(0,0);E:=(4,1);F:=(2,3);
draw Triangle(D,E,F)()()();
fin;
```



Exemple 91

```
repere();
pair D,E,F;
D:=(0,0);E:=(4,1);F:=(2,3);
angle_cote:=0;
draw Triangle(D,E,F)("$A$","$B$","$C$")
                    ("$a$","$b$","$c$")
     ("$\alpha$","$\beta$","$\gamma$");
fin;
```



Paramètres

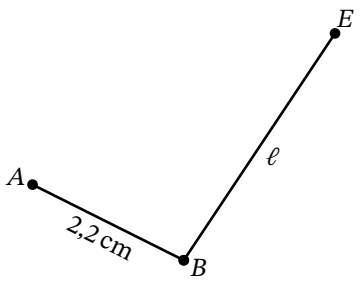
Nom	Type	Défaut	
`AffPoints`	`boolean`	`true`	Booléen qui indique si les noms des points doivent être affichés. Les noms sont affichés quelle que soit la valeur de `AffPoints` si le nom est donné explicitement.

Paramètres

Nom	Type	Défaut	
AffCotes	boolean	false	Booléen qui indique si la cote des segment doit être affichée. La cote est affichée quelle que soit la valeur de `AffCotes` si une cote est donnée explicitement.
UnitCotes	string	""	Chaine de caractère qui indique l'unité à rajouter à la cote lorsqu'elle est composée automatiquement.
ArrondiCotes	numeric	2	Valeur entière qui indique à quelle décimale la cote doit être arrondie lorsqu'elle est affichée automatiquement.
AffAngles	boolean	false	Booléen qui indique si les angles doivent être marqués et leur nom affiché. Les noms sont affichés quelle que soit la valeur de `AffAngles` si le nom est donné explicitement.
ArrondiAngles	numeric	1	Valeur entière qui indique à quelle décimale l'angle doit être arrondi lorsqu'il est affiché automatiquement.
EchelleAngles	numeric	0.8	Valeur numérique indiquant le facteur d'échelle de l'affichage des angles.
AutoAngleDroit	boolean	false	Booléen qui indique si les angles droits doivent être marqués comme tels.

Exemple 92

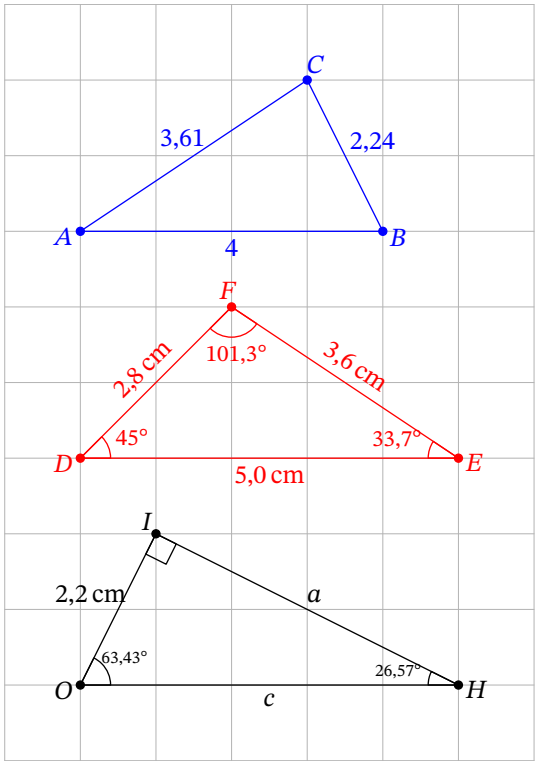
```
repere();
pair A,B,C;
A=(0,0);B=(2,-1);C=(4,2);
drawoptions(withpen pencircle scaled 1);
AffCotes:=true;
ArrondiCotes:=1;UnitCotes:="cm";
draw Segment(A,B)()();
angle_cote:=0;
draw Segment(C,B)("$E$","")("$\ell$");
fin;
```



Pour les points, cotes ou angles, la chaine `"~"` permet de laisser les valeurs calculées tout en passant d'autres valeurs à l'affichage.

Exemple 93

```
repere(-1,6,1cm,-4,6,1cm);
pair A,B,C,D,E,F,G,H,I;
A=(0,3);B=(4,3);C=(3,5);
D=(0,0);E=(5,0);F=(2,2);
G=(0,-3);H=(5,-3);I=(1,-1);
draw quadrillage(1,1);
angle_cote:=0;AffCotes:=true;
draw Triangle(A,B,C)()()() withcolor blue;
AffAngles:=true;angle_cote:=-1;
ArrondiCotes:=1;UnitCotes:="cm";
draw Triangle(D,E,F)()()() withcolor red;
EchelleAngles:=0.6;AutoAngleDroit:=true;
ArrondiAngles:=2;angle_cote:=0;
draw Triangle(G,H,I)
           ("$O$")("$a$","~","$c$")();
fin;
```



10.6 Figures définies à l'aide de propriétés géométriques

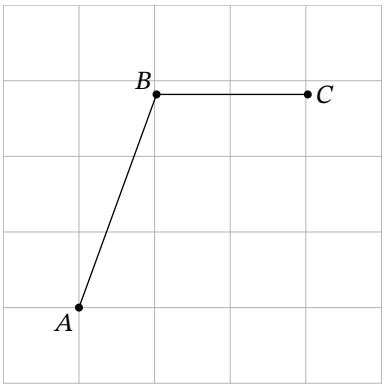
Segments

`defSegmentL(<A>,<B>)(<long>,<ptdep>,<angle>)`

Définit les points A et B de telle sorte que le segment $[AB]$ mesure `long`, commence à `ptdep` et fasse un angle `angle` avec l'horizontale. Les valeurs `ptdep` et `angle` peuvent être omises, les valeurs par défaut sont respectivement `(0,0)` et `0`.
Il n'est pas nécessaire de déclarer les points A et B. S'ils existent déjà, leur valeur n'est pas modifiée.

Exemple 94

```
repere(0,5,1cm,0,5,1cm);
draw quadrillage(1,1);
defSegmentL(A,B)(3,(1,1),70);
defSegmentL(B,C)(2);
draw A--B--C;
nomme.llft(A);nomme.ulft(B);
nomme.rt(C);
fin;
```

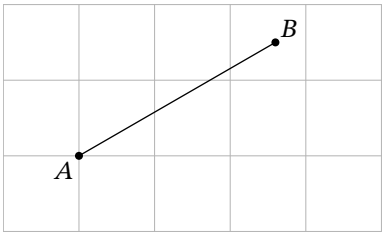


`segmentL(<A>,<B>)(<long>,<ptdep>,<angle>)` `path`

Définit les points A et B comme précédemment et renvoie la ligne `A--B`.

Exemple 95

```
repere(0,5,1cm,0,3,1cm);
draw quadrillage(1,1);
draw segmentL(A,B)(3,(1,1),30);
nomme.llft(A);nomme.urt(B);
fin;
```

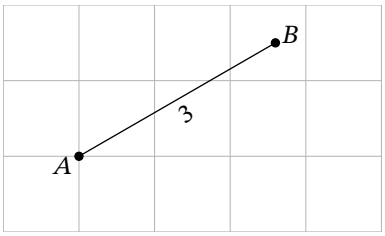


`SegmentL(<A>,<B>)(<long>,<ptdep>,<angle>)(<points>)(<cote>)` `picture`

Définit les points A et B comme précédemment et renvoie le segment complet (voir 10.5).

Exemple 96

```
repere(0,5,1cm,0,3,1cm);
draw quadrillage(1,1);
AffCotes:=true;
draw SegmentL(A,B)(3,(1,1),30)()();
fin;
```



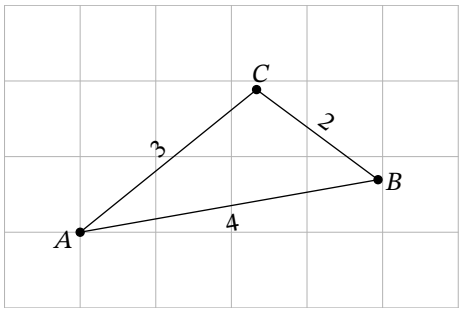
Triangles

`defTriangleLLL(<A>,<B>,<C>)(<longAB>,<longAC>,<longBC>,<ptdepart>,<angle>)`

Définit les points A, B et C de telle sorte que $AB =$ `longAB`, $AC =$ `longAC`, $BC =$ `longBC`, `A=ptdepart` et l'angle entre $[AB]$ et l'horizontale vaut `angle`.
Les valeurs `ptdep` et `angle` peuvent être omises, leurs valeurs par défaut sont respectivement `(0,0)` et `0`.
Il n'est pas nécessaire de déclarer les points A, B et C. Si certains de ces points existent déjà, leur valeur n'est pas modifiée, les autres valeurs sont calculées au mieux.

Exemple 97

```
repere(-1,5,1cm,-1,3,1cm);
draw quadrillage(1,1);
defTriangleLLL(A,B,C)(4,3,2,10);
AffCotes:=true;
draw Triangle(A,B,C)()()();
fin;
```

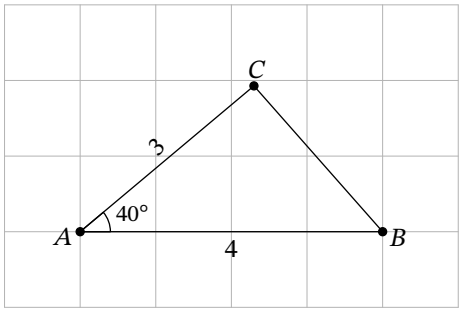


`defTriangleLLA(<A>,<B>,<C>)(<longAB>,<longAC>,<angleA>,<ptdepart>,<angle>)`

Définit les points A, B et C de telle sorte que $AB =$ `longAB`, $AC =$ `longAC`, $\widehat{BAC} =$ `angleA`, `A=ptdepart` et l'angle entre $[AB]$ et l'horizontale vaut `angle`.
Les valeurs `ptdep` et `angle` peuvent être omises, leurs valeurs par défaut sont respectivement `(0,0)` et `0`.
Il n'est pas nécessaire de déclarer les points A, B et C. Si certains de ces points existent déjà, leur valeur n'est pas modifiée, les autres valeurs sont calculées au mieux.

Exemple 98

```
repere(-1,5,1cm,-1,3,1cm);
draw quadrillage(1,1);
defTriangleLLA(A,B,C)(4,3,40);
AffCotes:=true;AffAngles:=true;
draw Triangle(A,B,C)()("")("~");
fin;
```

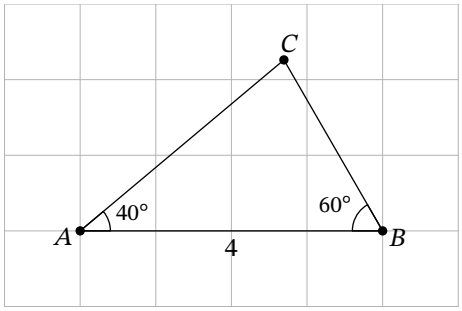


`defTriangleLAA(<A>,<B>,<C>)(<longAB>,<angleA>,<angleB>,<ptdepart>,<angle>)`

Définit les points A, B et C de telle sorte que AB = `longAB`, $\widehat{BAC}$ = `angleA`, $\widehat{CBA}$ = `angleB`, `A=ptdepart` et l'angle entre $[AB]$ et l'horizontale vaut `angle`.
Les valeurs `ptdep` et `angle` peuvent être omises, leurs valeurs par défaut sont respectivement `(0,0)` et `0`.
Il n'est pas nécessaire de déclarer les points A, B et C. Si certains de ces points existent déjà, leur valeur n'est pas modifiée, les autres valeurs sont calculées au mieux.

Exemple 99

```
repere(-1,5,1cm,-1,3,1cm);
draw quadrillage(1,1);
defTriangleLAA(A,B,C)(4,40,60);
AffCotes:=true;AffAngles:=true;
draw Triangle(A,B,C)()("","")("~","~");
fin;
```



`triangleLLL(<A>,<B>,<C>)(<def>)` `(closed) path`

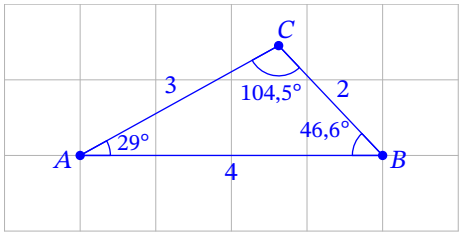
Définit les points A, B et C comme précédemment et renvoie la ligne `A--B--C--cycle`.

`TriangleLLL(<A>,<B>,<C>)(<def>)(<points>)(<cotes>)(<angles>)` `picture`

Définit les points A, B et C comme précédemment et renvoie le triangle complet (voir 10.5).

Exemple 100

```
repere(-1,5,1cm,-1,2,1cm);
draw quadrillage(1,1);
angle_cote:=0;
AffCotes:=true;AffAngles:=true;
draw TriangleLLL(A,B,C)(4,3,2)()()()
                              withcolor blue;
fin;
```



`triangleLLA(<A>,<B>,<C>)(<def>)` `(closed) path`

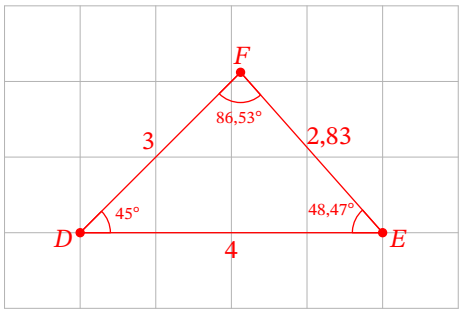
Définit les points A, B et C comme précédemment et renvoie la ligne `A--B--C--cycle`.

`TriangleLLA(<A>,<B>,<C>)(<def>)(<points>)(<cotes>)(<angles>)` `picture`

Définit les points A, B et C comme précédemment et renvoie le triangle complet (voir 10.5).

Exemple 101

```
repere(-1,5,1cm,-1,3,1cm);
draw quadrillage(1,1);
angle_cote:=0;
AffCotes:=true;AffAngles:=true;
ArrondiAngles:=2;
EchelleAngles:=0.6;
draw TriangleLLA(D,E,F)(4,3,45)()()()
                              withcolor red;
fin;
```



`triangleLAA(<A>,<B>,<C>)(<def>)` `(closed) path`

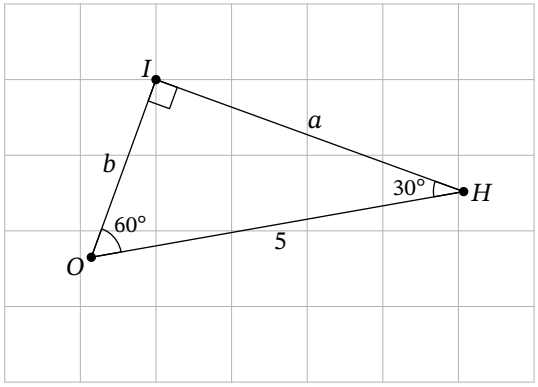
Définit les points A, B et C comme précédemment et renvoie la ligne `A--B--C--cycle`.

`TriangleLAA(<A>,<B>,<C>)(<def>)(<points>)(<cotes>)(<angles>)` `picture`

Définit les points A, B et C comme précédemment et renvoie le triangle complet (voir 10.5).

Exemple 102

```
repere(-1,6,1cm,-2,3,1cm);
draw quadrillage(1,1);
angle_cote:=0;
AffCotes:=true;AffAngles:=true;
ArrondiAngles:=2;
EchelleAngles:=0.8;
AutoAngleDroit:=true;
pair I;
I:=(1,2);
draw TriangleLAA(G,H,I)(5,60,30,10)
("$O$")("$a$","$b$","~")();
fin;
```

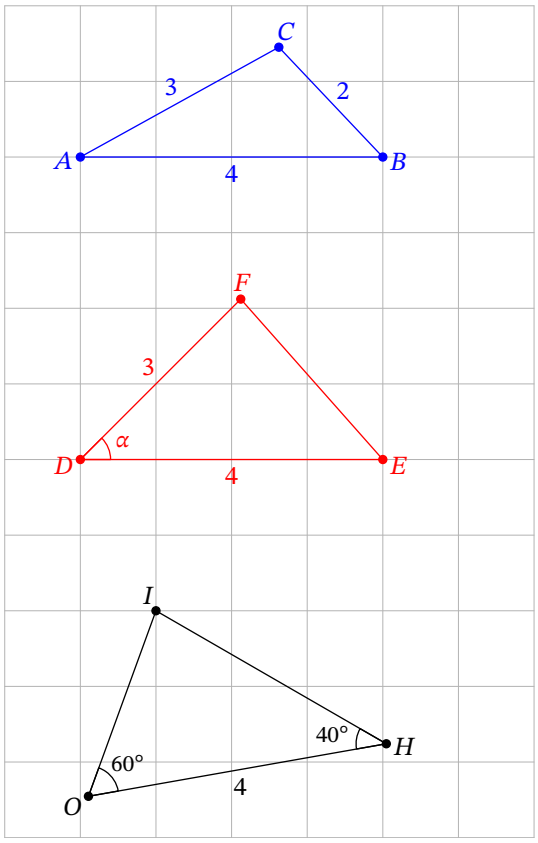


Paramètres

Nom	Type	Défaut	
`AffDonnees`	`string`	`false`	Booléen qui permet de n'afficher que les valeurs des côtés et des angles correspondant aux données.

Exemple 103

```
repere(-1,6,1cm,-5,6,1cm);
angle_cote:=0;
AffDonnees:=true;
draw quadrillage(1,1);
draw TriangleLLL(A,B,C)(4,3,2,(0,4))()()()
                              withcolor blue;
draw TriangleLLA(D,E,F)(4,3,45)
()()("$\alpha$","$\beta$","$\gamma$")
                              withcolor red;
pair I; I:=(1,-2);
draw TriangleLAA(G,H,I)(4,60,40,10)
                ("$O$")("$a$","$b$","~")();
fin;
```



10.7 Transformations

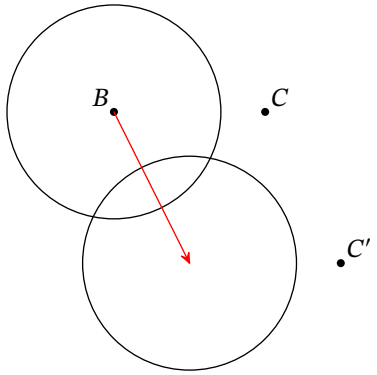
Un certain nombre de transformations sont prédéfinies dans METAPOST. Elles sont ici reprises et, parfois, légèrement modifiées, notamment en ce qui concerne la syntaxe.

`translation(<vect>)(<obj>)` `pair, path ou picture`

Image de `obj` par la translation de vecteur `vect`.

Exemple 104

```
repere();
pair A,B,C,C',u;
path c;
u=(1,-2);
A=(1,1);B=(2,2);C=(4,2);
nomme.ulft(B);
c=cercle(B,A);draw c;
draw translation(u)(c);
C'=translation(u)(C);
nomme.urt(C);nomme.urt(C');
drawvecteur(B,u) withcolor red;
fin;
```

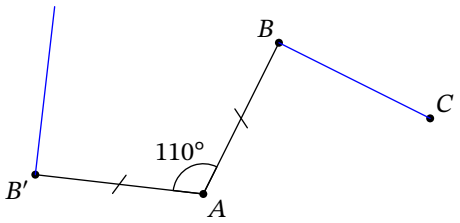


`rotation(<pt>,<ang>)(<obj>)` — `pair, path ou picture`

Image de `obj` par la rotation de centre `pt` et d'angle `ang`.

Exemple 105

```
repere();
pair A,B,C,B';
A=(2,1);B=(3,3);C=(5,2);
nomme.lrt(A);nomme.ulft(B);
nomme.urt(C);
B'=rotation(A,110)(B);
nomme.llft(B');
draw B--A--B';
nomme(B,A,B',"\ang{110}",1);
draw marquesegment(B,A,A,B');
path seg; seg:=B--C;
draw seg withcolor blue;
draw rotation(A,110)(seg) withcolor blue;
fin;
```

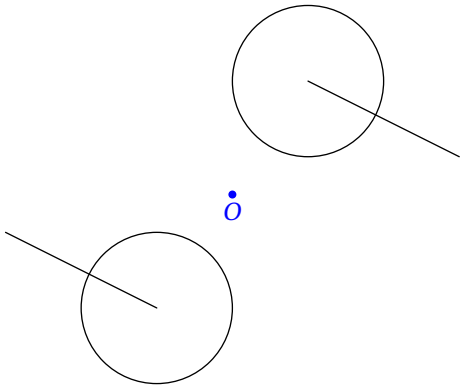


`symetrie(<pt>)(<obj>)` — `pair, path ou picture`

Image de `obj` par la symétrie de centre `pt`.

Exemple 106

```
repere();
 pair C,D,O;
 path s,c;
 C=(3,3);D=(1,4);
 O=(0,2.5);
 s=C--D;
 c=cercle(D,1);
 draw s;draw c;
 nomme.bot(O) withcolor blue;
 draw symetrie(O)(s);
 draw symetrie(O)(c);
fin;
```

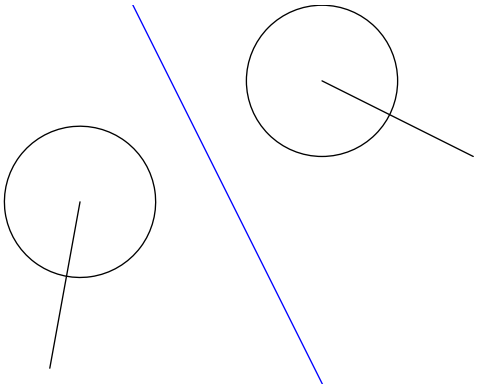


`symetrie(<A>,<B>)(<obj>)` — `pair, path ou picture`

Image de `obj` par la symétrie d'axe `(AB)`.

Exemple 107

```
repere(-4,4,1cm,0,5,1cm);
 pair C,D,O,I;
 path s,c;
 C=(3,3);D=(1,4);
 O=(1,0);I=(-1,4);
 s=C--D;
 c=cercle(D,1);
 draw s;draw c;
 draw droite(O,I) withcolor blue;
 draw symetrie(O,I)(s);
 draw symetrie(O,I)(c);
fin;
```

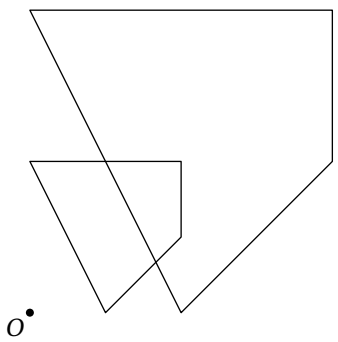


`homothetie(<pt>,<rap>)(<obj>)` — `pair, path ou picture`

Image de `obj` par l'homothétie de centre `pt` et de rapport `rap`.

Exemple 108

```
repere();
 pair O,A,B,C,D;
 O=(1,1);
 A=(2,1);B=(3,2);C=(3,3);D=(1,3);
 path p;p:= polygone(A,B,C,D);
 nomme.llft(O);
 draw p;
 draw homothetie(O,2)(p);
fin
```



`similitude(<pt>,<rap>,<ang>)(<obj>)` — `pair, path ou picture`

Image de `obj` par la similitude de centre `pt`, de rapport `rap` et d'angle `ang`.

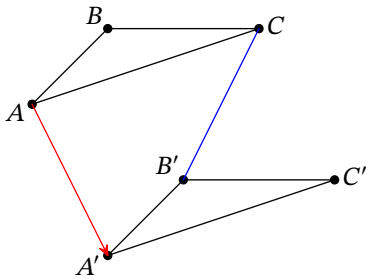
À titre expérimental, on dispose du paramètre suivant qui permet de définir les points créés par l'image d'une figure par transformation.

Paramètres

Nom	Type	Défaut	
`extratransf`	`string`	`""`	Chaine qui désigne les caractères à ajouter au nom des points lors du dessin de l'image de certaines figures (notamment `Triangle`, `Segment`) par transformation. La valeur peut être `"'"`, `"''"`, ... ou un nombre. Les nouveaux points sont créés.

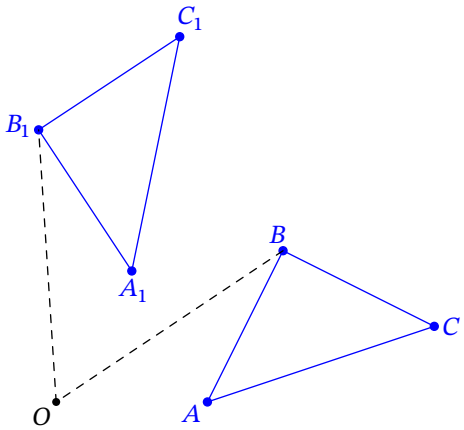
Exemple 109

```
repere();
extratransf:="'";
pair A,B,C,u;
u=(1,-2);
A=(1,1);B=(2,2);C=(4,2);
draw Triangle(A,B,C)()()();
draw translation(u)(Triangle(A,B,C)()()());
draw B'--C withcolor blue;
drawvecteur(A,u) withcolor red;
fin;
```



Exemple 110

```
repere();
extratransf:="1";
pair A,B,C,D,u,O;
A=(2,1);B=(3,3);C=(5,2);
O=(0,1);
nomme.llft(O);
draw Triangle(A,B,C)()()() withcolor blue;
draw rotation(O,60)(Triangle(A,B,C)()()())
                                withcolor blue;
draw B--O--B1 dashed evenly;
fin;
```



11 Dessin à main levée

L'idée de dessiner une figure comme si elle était faite à main levée vient de Christophe POULAIN dans les macros de `geometriesyr16.mp` [1] . Cette idée est reprise par Toby THURSTON dans le document « Drawing with MetaPost » [2]. Le principe est de modifier le tracé des chemins en changeant légèrement les angles de départ et d'arrivée pour créer une déformation qui rappelle le dessin à main levée.

C. POULAIN ajoute un angle de 5°. Dans son document, T. THURSTON reprend l'idée et ajoute une composante aléatoire.

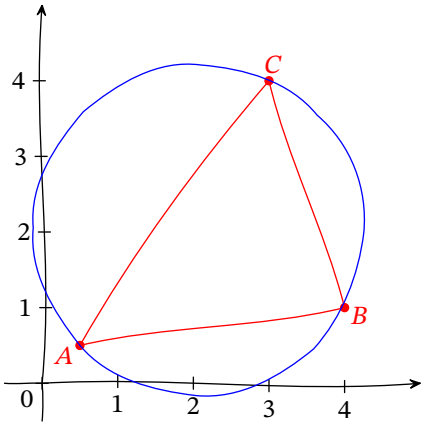
Dans la version proposée ici, la valeur par défaut est aléatoire mais peut être modifiée.

`drawmainlevee`

Dessine une figure en représentant les chemins comme s'ils étaient dessinés à main levée

Exemple 111

```
repere(-0.5,5,1cm,-0.5,5,1cm);
pair A,B,C,D;
A=(0.5,0.5);B=(4,1);C=(3,4);
drawmainlevee axes(1,1);
drawmainlevee Triangle(A,B,C)()()() withcolor red;
drawmainlevee cercle(A,B,C) withcolor blue;
fin;
```

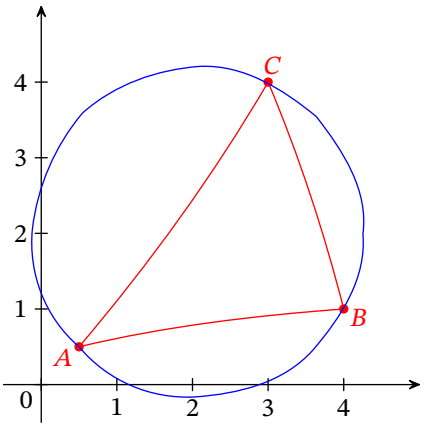


Paramètres

Nom	Type	Défaut	
`type_trace`	`string`	`""`	Paramètre qui peut prendre la valeur `"mainlevee"` pour indiquer que tous les dessins qui suivent seront faits « à main levée ». Toute autre valeur donne un dessin normal.

Exemple 112

```
repere(-0.5,5,1cm,-0.5,5,1cm);
pair A,B,C,D;
A=(0.5,0.5);B=(4,1);C=(3,4);
draw axes(1,1);
type_trace:="mainlevee";
draw Triangle(A,B,C)()()() withcolor red;
draw cercle(A,B,C) withcolor blue;
fin;
```



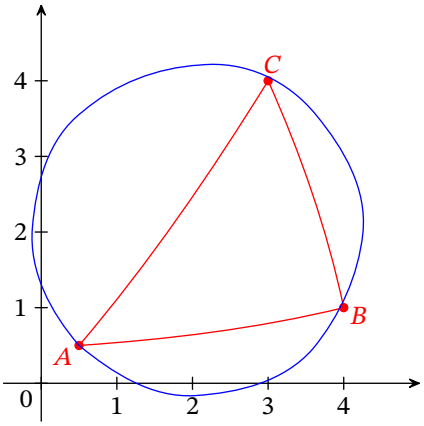
1. https://melusine.eu.org/syracuse/poulecl/geometriesyr16/
2. https://github.com/thruston/Drawing-with-Metapost/blob/master/Drawing-with-Metapost.pdf

`mainlevee(<objet>)` — `path ou picture`

Transforme le chemin ou la figure en un objet de même nature « dessiné à main levée ».

Exemple 113

```
repere(-0.5,5,1cm,-0.5,5,1cm);
pair A,B,C,D;
A=(0.5,0.5);B=(4,1);C=(3,4);
draw axes(1,1);
picture tri;
path cer;
tri:=mainlevee(Triangle(A,B,C)()()());
cer:=mainlevee(cercle(A,B,C));
draw tri withcolor red;
draw cer withcolor blue;
fin;
```

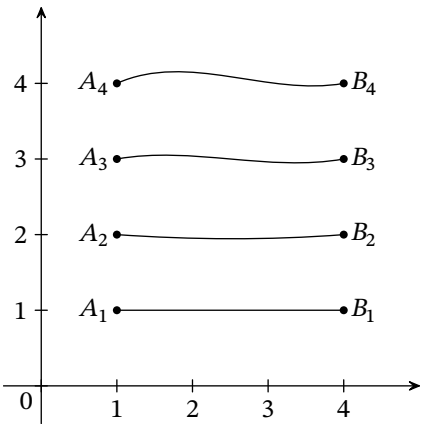


`anglemainlevee(<nombre>)`

Fonction qui permet de définir comment seront dessinés les figures à main levée : le `nombre` sera ajouté à l'angle de départ et à l'angle d'arrivée des chemins pour créer une déformation. `nombre` peut contenir un calcul avec les fonctions `normaldeviate` et `uniformdeviate()`.
Au chargement de `repere`, la fonction est appelée de la façon suivante :
`anglemainlevee(5*signe(normaldeviate)+normaldeviate);`

Exemple 114

```
repere(-0.5,5,1cm,-0.5,5,1cm);
pair A[],B[];
draw axes(1,1);
for i=1 upto 4:
  A[i]:=(1,i);B[i]:=(4,i);
  nomme.lft(A[i]);nomme.rt(B[i]);
endfor
draw A[1]--B[1];
drawmainlevee A[2]--B[2];
anglemainlevee(10);
drawmainlevee A[3]--B[3];
anglemainlevee(uniformdeviate 20 + 10);
drawmainlevee A[4]--B[4];
fin;
```



12 Divers

12.1 Compilation avec `mpost`

Il est possible de compiler une série de figure dans un document `mpost`. Chaque figure devra alors débuter par une instruction `repere()` et se terminer par `fin`.

Il est possible de placer ces commandes en dehors d'un environnement `beginfig()-endfig`, la numérotation est alors automatique et est incrémentée d'une unité à chaque commande `repere`.

```
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.

12.2 Composition des étiquettes

Tous les textes et étiquettes peuvent être composés en utilisant la macro ci-dessous.

`LaTeX(<ch>)` `picture`

Figure formée de la chaîne `ch` composée avec LaTeX et mise à l'échelle `defaultscale`. Cette macro utilise la commande `textext` de `luamplib` dans le cas de l'utilisation de LuaLaTeX et `textext` de `latexmp` dans le cas d'une compilation METAPOST « standard ». Ce dernier cas nécessite alors deux compilations.

Exemple 115

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

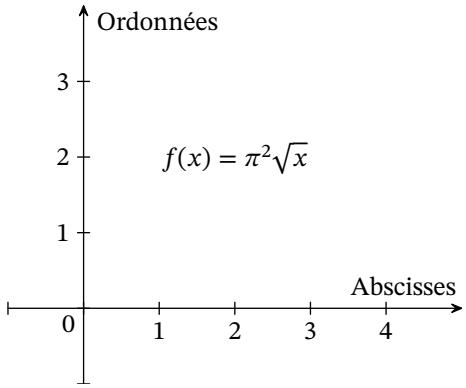
$\frac{1}{2}$ $\frac{1}{3}$ $\frac{1}{4}$ $\frac{1}{5}$ $\frac{1}{6}$

`label.<pos>(<fig>,<point>)`

Commande de METAPOST qui permet de placer la figure `fig` au niveau du point `point`.

Exemple 116

```
repere(-1,5,1cm,-1,4,1cm);
draw axes(1,1);
label.ulft("Abscisses",(5,0.1));
label.lrt("Ordonnées",(0.1,4));
label("$f(x)=\pi^2\sqrt{x}$",(2,2));
fin;
```

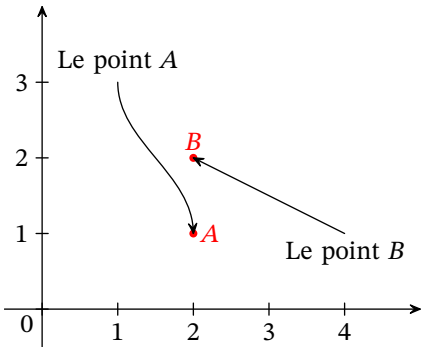


`legende.<pos>(<fig>,<p>)`

Figure formée du chemin `p` dessiné avec une flèche et de la figure ou de la chaine `fig` située à la position `pos` par rapport au premier point du chemin.

Exemple 117

```
repere(-0.5,5,1cm,-0.5,4,1cm);
 draw axes(1,1);
 pair A,B;
 A=(2,1);B=(2,2);
 nomme.rt(A) withcolor red;
 nomme.top(B) withcolor red;
 legende.top("Le␣point␣$A$",(1,3){down}..{down}A);
 legende.bot("Le␣point␣$B$",(4,1)--B);
fin;
```



12.3 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	grisfonce	vertfonce

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

12.4 Remplissage

`avec`

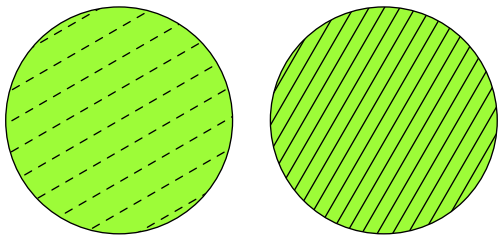
Pour remplir des chemins fermés avec autre chose que de la couleur, `repere` permet l'utilisation de la syntaxe `fill p avec motif` où `motif` est un des motifs décrits ci-dessous ou même une figure quelconque. Cette instruction peut être complétée par des options de dessin (`withpen`, `withcolor`...).

`hachures(<pas>,<angle>)` `picture`

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.

Exemple 118

```
repere();
path c[];
c1=fullcircle scaled 3;
c2=c1 shifted (3.5,0);
fill c1 withcolor lime;
fill c1 avec hachures(10,30) dashed evenly;
fill c2 withcolor lime;
fill c2 avec hachures();
draw c1;draw c2;
fin;
```

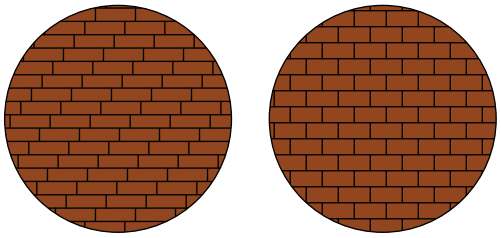


`briques(<larg>,<haut>,<dec>)` `picture`

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.

Exemple 119

```
repere();
path c[];
c1=fullcircle scaled 3;
c2=c1 shifted (3.5,0);
fill c1 withcolor (0,0.65,0.8,0.48);
fill c1 avec briques(15,5,4);
fill c2 withcolor (0,0.65,0.8,0.48);
fill c2 avec briques();
draw c1;draw c2;
fin;
```

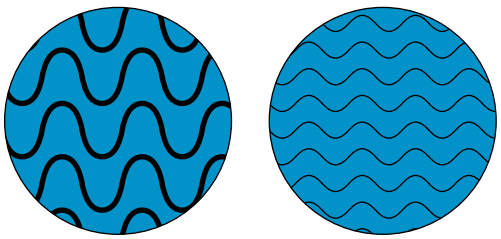


`vagues(<per>,<amp>,<dec>)` `picture`

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

Exemple 120

```
repere();
path c[];
c1=fullcircle scaled 3;
c2=c1 shifted (3.5,0);
fill c1 withcolor (1,0,0,0.2);
fill c1 avec vagues(30,10,20)
          withpen pencircle scaled 2;
fill c2 withcolor (1,0,0,0.2);
fill c2 avec vagues();
draw c1;draw c2;
fin;
```



12.5 Couleur de fond

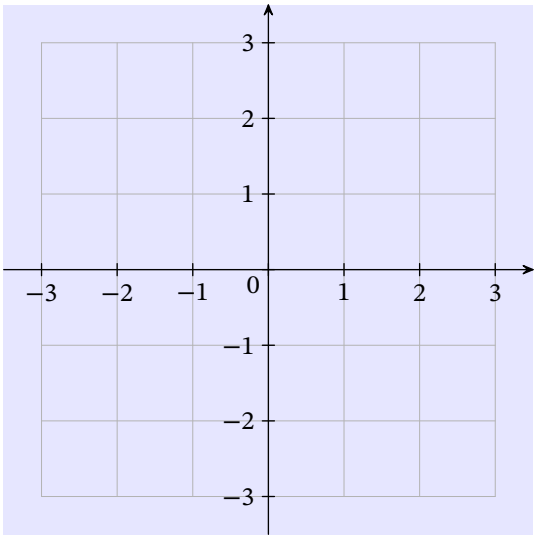
Il est possible de modifier la couleur de fond en modifiant la couleur `fond_coul`.

Paramètres

Nom	Type	Défaut	
`fond_coul`	`color`	`white`	Couleur du fond de la figure

Exemple 121

```
repere(-3.5,3.5,1cm,-3.5,3.5,1cm);
 fond_coul:=(0.9,0.9,1);
 setquad(-3,3,-3,3);
 draw quadrillage(1,1);
 draw axes(1,1);
fin;
```



12.6 Francisation

Quelques mot-clefs ont été traduits.

Mots-clefs

Nom français	Nom original
`couleur`	`withcolor`
`epaisseur`	`withpen pencircle scaled`

Deuxième partie

Tableaux et grilles

Il est possible d'utiliser `repere` pour représenter des tableaux ou « damiers » et placer des objets dans chacune des cases. La figure devra alors débuter par une commande `tableau` au lieu de la commande `repere` et la numérotation sera automatique.

13 Définition et grilles

13.1 Définition du tableau

`tableau(<n>,<m>,<u>)`

Débute une figure et définit un tableau de n colonnes et m lignes. La largeur des colonnes est égale à la hauteur des lignes et vaut `u`.

`tableau(<n>,<m>,<ux>,<uy>)`

Débute une figure et définit un tableau de n colonnes et m lignes. La largeur des colonnes vaut `ux` et la hauteur des lignes vaut `uy`.

`fin`

Termine la figure.

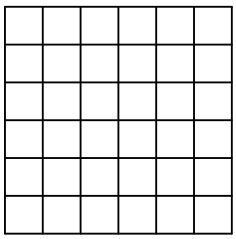
13.2 Grille

`grille(<x>,<y>)` `picture`

Quadrillage avec un pas de `x` sur les colonnes et de `y` sur les lignes.

Exemple 122

```
tableau(6,6,0.5cm);
 draw grille(1,1);
fin;
```



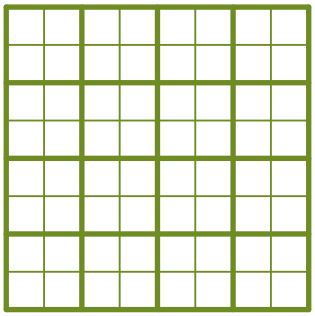
La couleur et l'épaisseur des lignes peuvent être modifiées localement mais on peut aussi les changer globalement.

Paramètres

Nom	Type	Défaut	
`grille_coul`	`color`	`black`	Couleur des traits de la grille
`grille_ep`	`numeric`	`0.7`	Épaisseur des traits de la grille

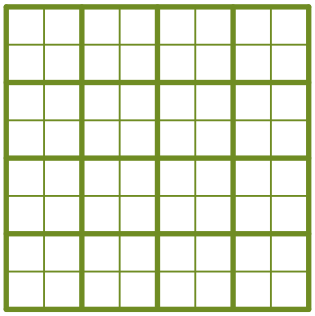
Exemple 123

```
tableau(8,8,0.5cm);
draw grille(1,1) couleur olive;
draw grille(2,2) couleur olive epaisseur 2;
fin;
```



Exemple 124

```
tableau(8,8,0.5cm);
grille_coul:=olive;
draw grille(1,1);
grille_ep:=2;
draw grille(2,2);
fin;
```

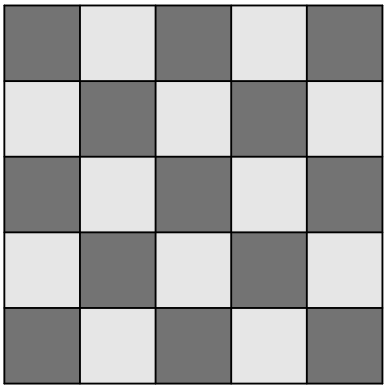


`damier(<x>,<y>)` `picture`

Figure formée de la grille et des cases coloriées alternativement.

Exemple 125

```
tableau(5,5,1cm);
  draw damier(1,1);
fin;
```

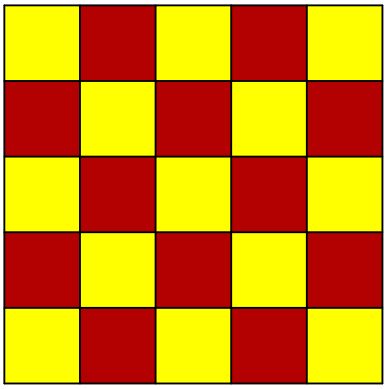


Paramètres

Nom	Type	Défaut	
`table_coul[1]`	`color`	`0.45white`	Couleur de la case située en bas à gauche
`table_coul[2]`	`color`	`0.9white`	Couleur de la case située à côté de la précédente

Exemple 126

```
tableau(5,5,1cm);
  table_coul[1]:=jaune;
  table_coul[2]:=0.7rouge;
  draw damier(1,1);
fin;
```



13.3 Grille partielle

`lignesh(<suite de 0 et 1>)` `picture`

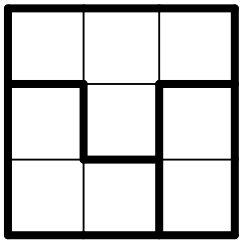
Figure qui décrit les lignes horizontales qui doivent être tracées. De gauche à droite et de haut en bas.

`lignesv(<suite de 0 et 1>)` `picture`

Figure qui décrit les lignes verticales qui doivent être tracées. De gauche à droite et de haut en bas.

Exemple 127

```
tableau(3,3,1cm);
  draw grille(1,1);
  draw lignesh(1,1,1,
               1,0,1,
               0,1,0,
               1,1,1) epaisseur 3;
  draw lignesv(1,0,0,1,
               1,1,1,1,
               1,0,1,1) epaisseur 3;
fin;
```



14 Repérage et cases

14.1 Coordonnées

`coordx` `picture`

Figure formée des numéros de colonnes placée par défaut en bas.

`coordy` `picture`

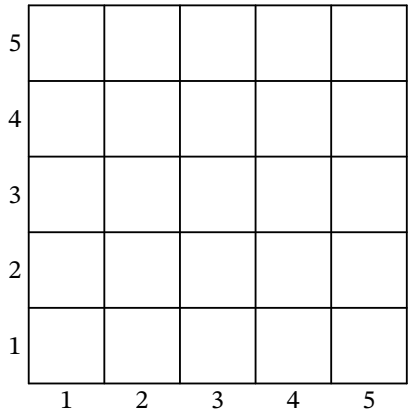
Figure formée des numéros de lignes placée par défaut à gauche.

`coord` `picture`

Figure formée des deux figures précédentes.

Exemple 128

```
tableau(5,5,1cm);
  draw grille(1,1);
  draw coord;
fin;
```

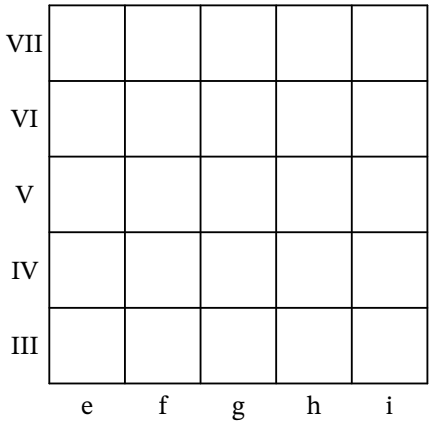


Paramètres

Nom	Type	Défaut	
style_coord_x	string	"1"	Variable qui indique comment doivent être composées les coordonnées des colonnes. Les valeurs possibles sont "1" (nombres), "a" (lettres minuscules), "A" (lettres majuscules), "i" (chiffres romains minuscules) et "I" (chiffres romains majuscules).
style_coord_y	string	"1"	Même chose pour les lignes.
deb_coord_x	numeric	1	Variable qui indique la première valeur des colonnes.
deb_coord_y	numeric	1	Variable qui indique la première valeur des lignes.
align_coord_y	string	"c"	Variable qui indique comment doivent être alignés horizontalement les numéros de lignes. Les valeurs possibles sont "c" (centré), "g" ou "l" (gauche) et "d" ou "r" (droite).
place_coord	string	"bg"	Variable qui indique où les coordonnées doivent être affichées. On peut utiliser une combinaison des lettres "b" (en bas), "h" (en haut), "g" (à gauche) et "d" (à droite).
inverse_coord_x	boolean	false	Variable qui indique si le sens par défaut dans lequel les numéros de colonnes doivent être écrits (de gauche à droite) doit être inversé.
inverse_coord_y	boolean	false	Variable qui indique si le sens par défaut dans lequel les numéros de lignes doivent être écrits (de bas en haut) doit être inversé.

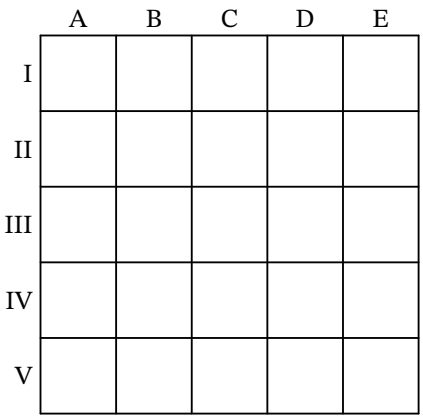
Exemple 129

```
tableau(5,5,1cm);
  draw grille(1,1);
  style_coord_x:="a";
  style_coord_y:="I";
  deb_coord_x:=5;
  deb_coord_y:=3;
  draw coord;
fin;
```



Exemple 130

```
tableau(5,5,1cm);
  draw grille(1,1);
  style_coord_x:="A";
  style_coord_y:="I";
  place_coord:="hg";
  inverse_coord_y:=true;
  align_coord_y:="r";
  draw coord;
fin;
```

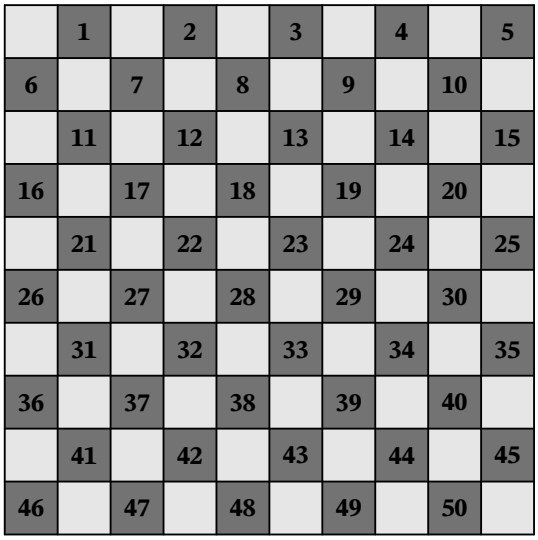


`numerotationdames` picture

Figure formée par les numéros des cases comme au jeu de dames.

Exemple 131

```
tableau(10,10,0.7cm);
 draw damier(1,1);
 draw numerotationdames;
fin;
```



14.2 Placement d'objets dans les cases

`case(<x>,<y>)` picture

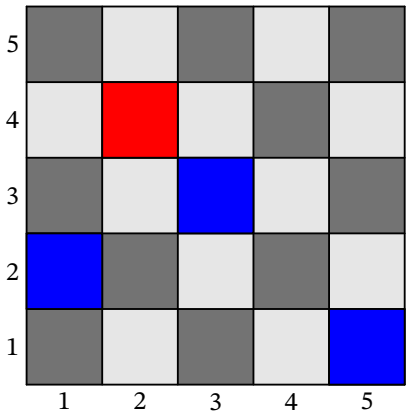
Figure formée de la case de coordonnées `(x,y)` remplie ainsi que de son contour dessiné avec l'épaisseur et la couleur de la grille.

`cases((<x1>,<y1>),(<x2>,<y2>),...)` `picture`

Toutes les cases `(x1,y1)`, `(x2,y2)`...

Exemple 132

```
tableau(5,5,1cm);
 draw damier(1,1);
 draw coord;
 draw case(2,4) couleur rouge;
 draw cases((1,2),(3,3),(5,1)) couleur bleu;
fin;
```



On peut utiliser une variante de `label` pour placer des objets dans les cases :

`tablabel(<obj>,<x>,<y>)`

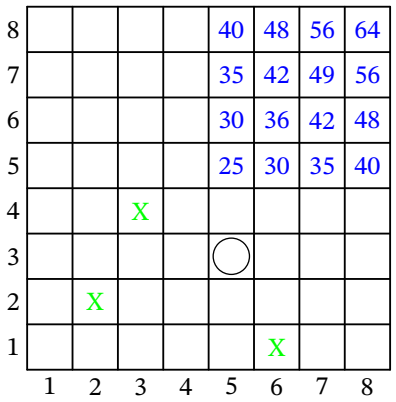
Place l'objet `obj` dans la case de coordonnées `(x,y)`. L'objet peut être une chaine de caractère, un chemin ou une figure.

`tablabels(<obj>)((<x1>,<y1>),(<x2>,<y2>)...)`

Place l'objet `obj` dans toutes les cases indiquées.

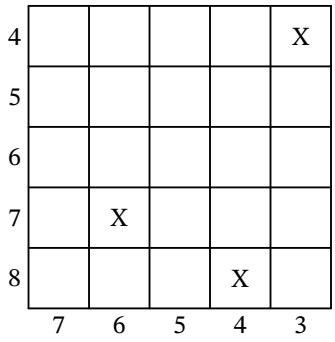
Exemple 133

```
tableau(8,8,0.6cm);
  draw grille(1,1);
  draw coord;
  tablabel(fullcircle scaled 0.8,5,3);
  tablabels("X")((3,4),(6,1),(2,2)) couleur vert;
  for i=5 upto 8:
    for j=5 upto 8:
      tablabel(decimal(i*j),i,j) couleur bleu;
    endfor
  endfor
fin;
```



Exemple 134

```
tableau(5,5,0.8cm);
  draw grille(1,1);
  inverse_coord_x:=true;
  inverse_coord_y:=true;
  deb_coord_x:=3;
  deb_coord_y:=4;
  draw coord;
  tablabels("X")((3,4),(6,7),(4,8));
fin;
```



15 Pixel art

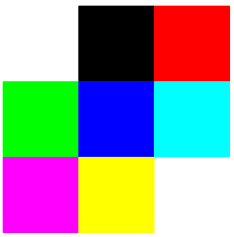
Il est possible d'utiliser les tableaux pour dessiner des figures en pixel art.

`pixart(<liste d'entiers> ou <liste de couleurs>)` `picture`

Figure composée de carrés remplis d'une couleur qui est soit la couleur indiquée explicitement soit une couleur qui dépend de l'entier indiqué (celles-ci sont définies au départ mais peuvent être modifiées). On peut remplacer un entier par _ pour que la case ne soit pas remplie. Les pixels doivent être donnés de gauche à droite et de haut en bas.

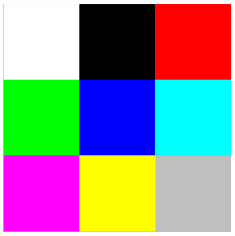
Exemple 135

```
tableau(3,3,1cm);
draw pixart(0,1,2,
            3,4,5,
            6,7,_);
fin;
```



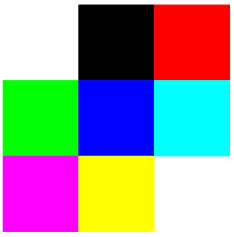
Exemple 136

```
tableau(3,3,1cm);
fond_coul:=grisclair;
draw pixart(0,1,2,
            3,4,5,
            6,7,_);
fin;
```



Exemple 137

```
tableau(3,3,1cm);
draw pixart(blanc,noir,rouge,
            vert,bleu,cyan,
            magenta,jaune,_);
fin;
```



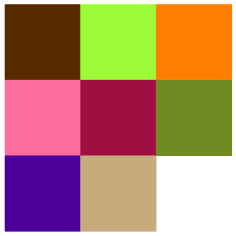
`pixcoul(<liste de couleurs>)`

Macro qui permet de définir les couleurs qui seront utilisées. La première correspond à 0, la deuxième à 1...
Au départ la macro est appelée de la façon suivante :

```
pixcoul(blanc,noir,rouge,vert,bleu,cyan,magenta,jaune);
```

Exemple 138

```
tableau(3,3,1cm);
pixcoul(marron,lime,orange,rose,
        pourpre,olive,violet,beige);
draw pixart(0,1,2,
            3,4,5,
            6,7,_);
fin;
```

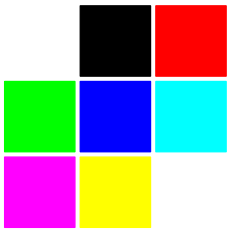


Paramètres

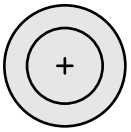
Nom	Type	Défaut	
`pixart_ep`	`numeric`	`0.3`	Écart laissé vide entre les cases.

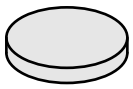
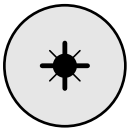
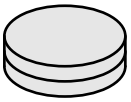
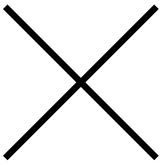
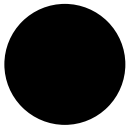
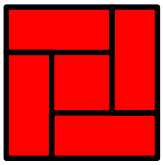
Exemple 139

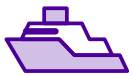
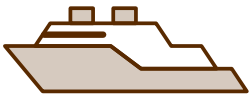
```
tableau(3,3,1cm);
pixart_ep:=2;
draw pixart(0,1,2,
            3,4,5,
            6,7,_);
fin;
```



16 Quelques dessins

Syntaxe	Exemple	Résultat
`Pion(<v>)(<coul1>,<coul2>)`	`Pion(1)(noir,0.85blanc)`	
	`Pion(1)(0.9blanc,noir)`	

Syntaxe	Exemple	Résultat
	`Pion(2)(noir,0.85blanc)`	
	`Pion(2)(0.9blanc,noir)`	
`Dame(<v>)(<coul1>,<coul2>)`	`Dame(1)(noir,0.85blanc)`	
	`Dame(1)(0.9blanc,noir)`	
	`Dame(2)(noir,0.85blanc)`	
	`Dame(2)(0.9blanc,noir)`	
`Croix`		
`Trou`		
`Marque(<v>)(<coul>)`	`Marque(1)((0.9,0.7,0.7))`	
	`Marque(2)((0.8,0.65,0.5))`	
`Mur(<coul1>,<coul2>)`	`Mur(rouge,noir)`	
`Plot(<coul1>,<coul2>)`	`Plot(0.7rouge,0.5rouge)`	
`Caisse(<coul1>,<coul2>)`	`Caisse(orange,marron)`	

Syntaxe	Exemple	Résultat
`Robotdroite(<coul1>,<coul2>)`	`Robotdroite(noir,0.6vert)`	
`Robotgauche(<coul1>,<coul2>)`	`Robotgauche(marine,0.6rouge)`	
`Bateau(<numcases>,<coul>)`	`Bateau(1,violet)`	
	`Bateau(2,marron)`	
`Eau(<coul>)`	`Eau(marine)`	
`Fleche(<coul1>,<coul2>)`	`Fleche(jaune,marine)`	

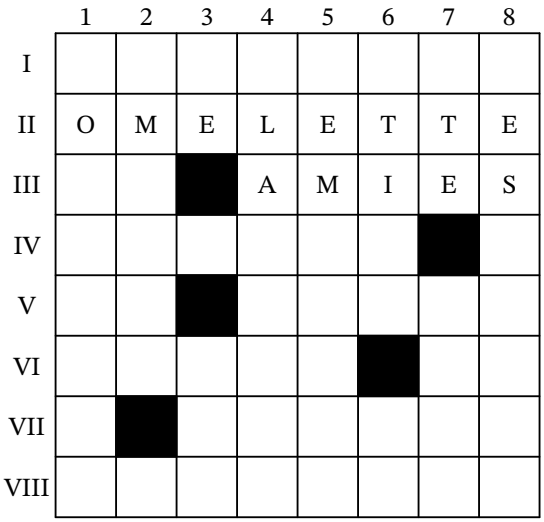
17 Exemples

17.1 Mots croisés

Exemple 140

```
def moth(expr ch,n,m)=
 for i=1 upto length ch:
  tablabel(substring(i-1,i) of ch,n+i-1,m);
 endfor
enddef;

tableau(8,8,0.8cm);
 draw grille(1,1);
 inverse_coord_y:=true;
 style_coord_y:="I";
 place_coord:="hg";
 draw coord;
 draw cases((3,3),(3,5),(2,7),(6,6),(7,4));
 moth("OMELETTE",1,2);
 moth("AMIES",4,3);
fin;
```



17.2 Sudokus

Exemple 141

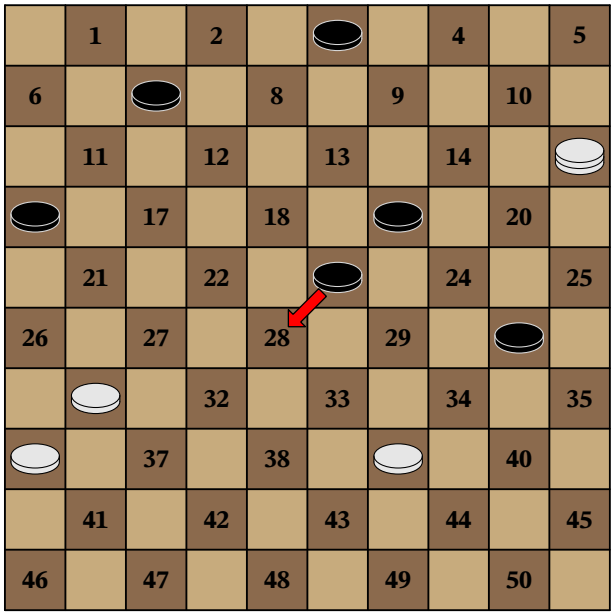
```
tableau(9,9,0.8cm);
 draw grille(1,1);
 draw grille(3,3) epaisseur 2;
 tablabels("1",(1,4),(4,8),(6,5),(7,3));
 tablabels("2",(3,9),(5,6),(6,2));
 tablabels("3",(1,1),(4,2),(9,6));
 tablabels("4",(3,8),(8,7));
 tablabels("5",(2,7),(4,9),(6,6),(7,1));
 tablabels("6",(2,4),(5,8));
 tablabels("7",(6,3),(7,2),(8,9),(9,5));
 tablabels("8",(1,3),(3,7),(5,5),(8,8));
 tablabels("9",(3,2));
fin;
```

		2	5				7	
		4	1	6			8	
	5	8					4	
				2	5			3
				8	1			7
1	6							
8					7	1		
		9	3		2	7		
3						5		

17.3 Dames

Exemple 142

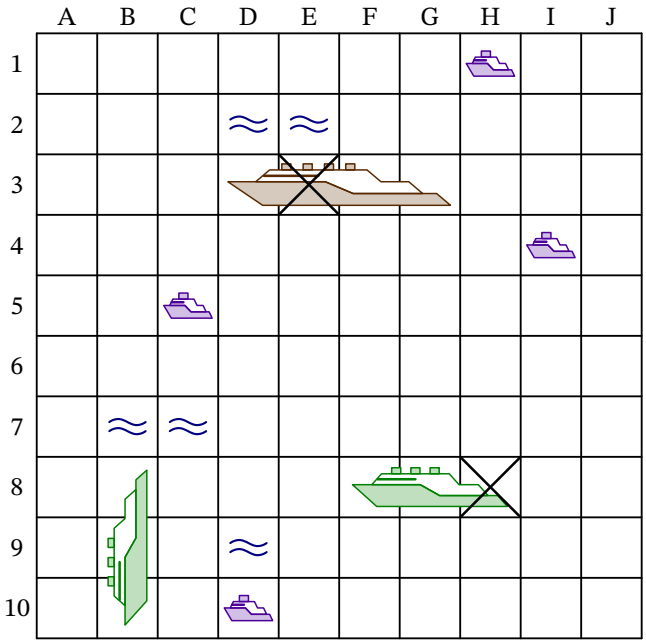
```
tableau(10,10,0.8cm);
 picture pionblanc,pionnoir,dameblanche;
 picture fleche;
 pionblanc:=Pion(2)(0.9blanc,noir);
 pionnoir:=Pion(2)(noir,0.85blanc);
 dameblanche:=Dame(2)(0.9blanc,noir);
 table_coul[1]:=0.7marron+0.3blanc;
 table_coul[2]:=beige;
 fleche:=Fleche(rouge,noir) rotated 135;
 draw damier(1,1);
 draw numerotationdames;
 tablabels(pionblanc,31,36,39);
 tablabels(pionnoir,3,7,16,19,23,30);
 tablabels(dameblanche,15);
 tablabel(fleche,5.5,5.5);
fin;
```



17.4 Bataille navale

Exemple 143

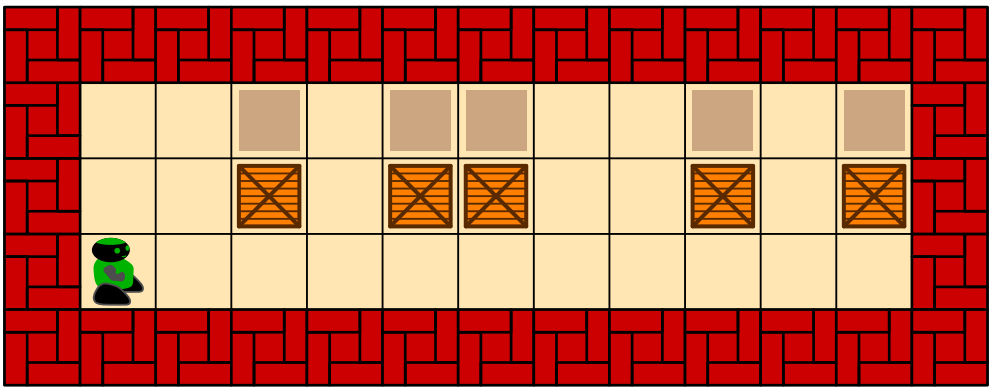
```
tableau(10,10,0.8cm);
 draw grille(1,1);
 inverse_coord_y:=true;
 style_coord_x:="A";
 place_coord:="hg";
 draw coord;
 tablabel(Bateau(4,marron),5.5,3);
 tablabel(Bateau(3,0.5vert),7,8);
 tablabel(Bateau(3,0.5vert) rotated 90,2,9);
 tablabels(Bateau(1,violet),(4,10),(3,5),(8,1),(9,4));
 tablabels(Eau(marine),(2,7),(3,7),(4,9),(4,2),(5,2));
 tablabels(Croix,(5,3),(8,8));
fin;
```



17.5 Algoréa

Exemple 144

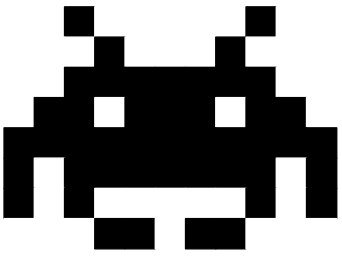
```
tableau(13,5,1cm);
 table_coul[1]:=(1,0.9,0.7);
 table_coul[2]:=(1,0.9,0.7);
 draw damier(1,1);
 tablabels(Mur(0.8rouge,noir),(1,2),(1,3),(1,4),(13,2),(13,3),(13,4));
 for i=1 upto 13:
  tablabels(Mur(0.8rouge,noir),(i,1),(i,5));
 endfor
 tablabel(Robotdroite(noir,0.7vert),2,2);
 for i=4,6,7,10,12:
  tablabel(Caisse(orange,marron),i,3);
  tablabel(Marque(2)((0.8,0.65,0.5)),i,4);
 endfor
fin;
```



17.6 Pixel art

Exemple 145

```
tableau(11,8,0.4cm);
pixart_ep:=0;
draw pixart(0,0,1,0,0,0,0,0,1,0,0,
            0,0,0,1,0,0,0,1,0,0,0,
            0,0,1,1,1,1,1,1,1,0,0,
            0,1,1,0,1,1,1,0,1,1,0,
            1,1,1,1,1,1,1,1,1,1,1,
            1,0,1,1,1,1,1,1,1,0,1,
            1,0,1,0,0,0,0,0,1,0,1,
            0,0,0,1,1,0,1,1,0,0,0);
fin;
```



Exemple 146

```
tableau(14,15,0.3cm);
pixart_ep:=0;
color b,n,v,r,c;
b:=blanc;n:=noir;v:=0.6vert;
r:=0.7rouge;c:=(1,0.2,0.2);
draw pixart(_,_,_,_,_,_,_,_,_,_,_,n,n,_,
            _,_,_,_,_,_,_,_,_,_,_,n,v,v,n,
            _,_,_,_,_,_,_,_,n,n,n,n,v,v,n,
            _,_,_,_,_,_,_,n,v,v,v,v,v,n,_,
            _,_,_,_,_,n,v,n,n,n,v,n,_,_,
            _,_,n,n,n,v,n,_,_,n,v,n,_,_,
            _,n,c,c,v,c,n,_,n,v,n,_,_,_,
            n,c,c,c,c,c,c,n,v,n,n,_,_,_,
            n,c,c,c,c,c,n,c,v,c,c,n,_,_,
            n,c,b,c,c,n,c,c,v,c,c,c,n,_,
            n,c,c,b,r,n,c,c,c,c,c,c,n,_,
            _,n,r,r,r,n,c,c,c,c,c,r,n,_,
            _,_,n,n,n,n,c,c,c,b,r,r,n,_,
            _,_,_,_,_,_,_,n,r,r,r,r,n,_,_,
            _,_,_,_,_,_,_,_,n,n,n,n,_,_,_
            );
fin;
```