Metapost et les inéquations

Christophe Poulain

1er août 2004

1 Présentation

Cela fait longtemps que je voulais produire avec MetaPost des représentations graphiques des solutions d'inéquations du 1er degré à 1 inconnue telles que celles ci-dessous.

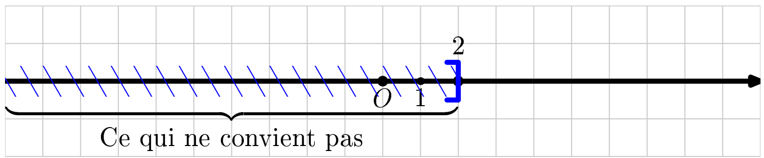


Fig. 1 – $x \geqslant 2$

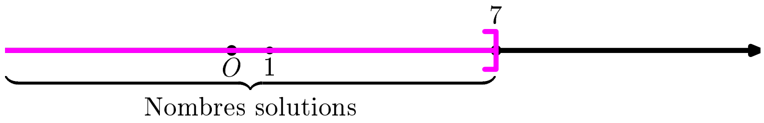


Fig. 2 – $x < 7$

Je voulais parvenir en une seule macro à faire afficher le résultat. Pourquoi ne pas faire une figure puis l'adapter ensuite suivant les cas car, finalement, les inéquations du 1er degré à une inconnue n'interviennent qu'à partir de la 3e et leurs représentations graphiques relativement peu ; donc le nombre de cas à traiter reste faible.
C'est exact mais les corrections de devoirs dans lesquelles intervenaient des inéquations me laissaient un goût d'inachevé car les élèves devaient faire ces représentations graphiques eux-mêmes[1]. Et puis, j'ai crée une fiche d'exercices[2] et pour la correction, j'aurais bien aimé un transparent avec la correction des représentations graphiques.

2 Détail des commandes de la macro

Avant toute chose, cette macro dépend de quatre fichiers[3] :
- les fichiers personnels[4] geometriesyr13.mp, papiers2.mp, constantes.mp
- le fichier TEX.mp livré avec MetaPost.

Il y a 3 paramètres de présentations qui sont :

[1]Cela peut-être un bon exercice mais lors d'une correction de Brevet Blanc, il y a déjà beaucoup de choses à corriger...
[2]Disponible à l'adresse *www.melusine.eu.org/syracuse/poulecl/college/3/*
[3]C'est un inconvénient mais libre ensuite à chacun d'adapter à ses propres pratiques
[4]Disponibles à l'adresse *www.melusine.eu.org/syracuse/poulecl/macros/*

style qui peut prendre les valeurs **hachures** (par défaut, voir figure n°1) ou **repasse** (voir figure n°2).

affichagesol qui peut prendre les valeurs **non** (par défaut, voir figure n°1) ou **oui** (voir figure n°2).

affichagevaleur qui peut prendre les valeurs **oui** (par défaut, voir figures n°1 et n°2) ou **non**. Ce dernier cas peut se présenter lorsque l'on a des valeurs particulières (π, $\sqrt{2}$,...) à placer : il faut alors utiliser
`label.top(btex $\pi$ etex,placepoint(valeur,0.5))`

et deux paramètres pour la représentation graphique en elle-même :

pos qui prend les valeurs **inf** (voir figure n°2) ou **sup** (voir figure n°1).

sol qui prend les valeurs **oui** (voir figuren°2) ou **non** (voir figure n°1).

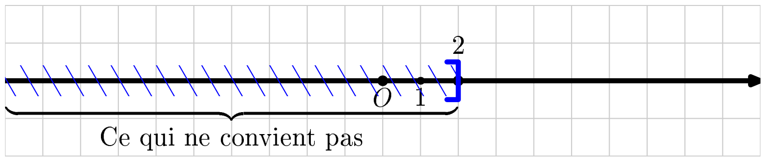
Finalement, on appelle la macro par

```
draw soline(7,120,red)
```

où 7 représente la valeur « limite », 120 l'angle des hachures (en degré) et *red* la couleur.

3 Exemples

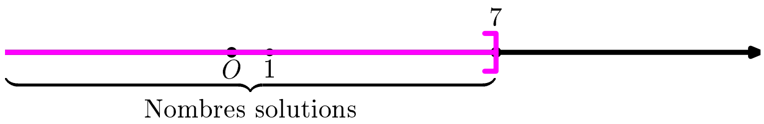
3.1 Exemple 1




```
figure(0,0,10cm,2cm);
trace grille(0.5) withcolor gris;
origine((10,2));
pos:="sup";
sol:="non";
draw soline(2,120,bleu);
fin;
```

La grille définit les unités : ici, $0,5\,cm$ sur chacun des axes. L'origine est définie dans les unités de la grille à partir de l'origine du cadre (en l'occurence, le coin en bas à gauche).

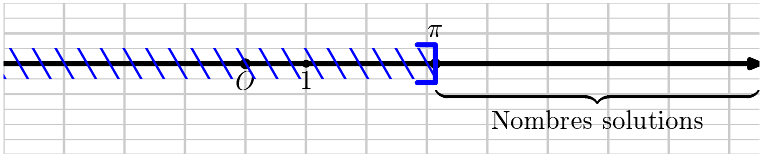
3.2 Exemple 2




```
figure(0,0,10cm,2cm);
z.origine=u*(3,1);
x.u:=5mm;
style:="repasse";
affichagesolu:="oui";
pos:="inf";
sol:="oui";
draw soline(7,60,violet);
fin;
```

Lorsque la grille (ou le papier millimétré, ou le papier cahier,...) ne définissent par les unités, il faut utiliser **z.origine** pour définir l'origine de la droite graduée et **x.u** pour définir l'unité de longueur sur cette droite graduée. L'unité « verticale » pour le placement des hachures, du crochet,... est de toutes façons réglée à 5mm.
On remarquera également le changement de style.

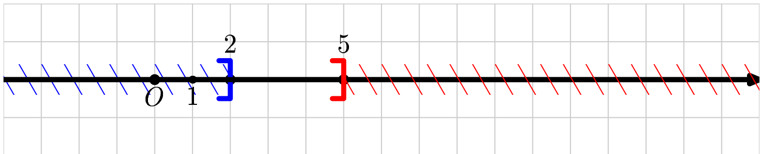
3.3 Exemple 3




```
figure(0,0,10u,2u);
trace papiercahier withcolor gris;
origine((4,1.5));
style:="hachures";
affichagesolu:="oui";
affichageval:="non";
pos:="sup";
sol:="non";
draw soline(pi,120,bleu);
label.top(btex $\pi$ etex,placepoint(pi,0.5));
fin;
```

Un exemple avec du papier cahier (pour ressembler au travail des élèves) avec un panachage (hachurage et affichage de l'intervalle solution). Egalement présent le cas de la valeur délicate à afficher.

3.4 Exemple 4




```
figure(0,0,10u,2u);
trace grille(0.5) withcolor gris;
origine((4,2));
style:="hachures";
affichagesolu:="rien";%ou autre chose que oui ou non
affichageval:="oui";
pos:="sup";
sol:="non";
draw soline(2,120,bleu);
pos:="inf";
sol:="oui";
draw soline(5,120,rouge);
fin;
```

On peut même résoudre graphiquement les systèmes d'inéquations du 1$^{\text{er}}$ degré à 1 inconnue.

4 La macro

```
input geometriesyr13;
input TEX;

string pos,sol,style,affichagesolu,affichageval;

style="hachures";
affichagesolu="non";
affichageval="oui";

%Macro pour représenter graphiquement les solutions d'une inéquation
vardef soline(expr valeur,ang,couleur)=
  save $;
  picture dep,$;
  pair ww;
  y.u:=5mm;
  dep=image(
    drawarrow (xpart(z.so),ypart(placepoint(0,0)))
    --(xpart(z.ne),ypart(placepoint(0,0))) withpen pencircle scaled 2bp;
    graduantx.bot;
    pair O;
    O=placepoint(0,0);
    marque_p:="plein";
    pointe(O);
    label.bot(btex $O$ etex,O);
    );
  $=image(
    trace dep;
    ww=placepoint(valeur,0);
    pointe(ww);
    marque_p:="non";
    if pos="inf":
      if style="hachures":
        draw hachurage(placepoint(valeur,0.4)--placepoint(valeur,-0.4)--
          (xpart(z.ne),ypart(placepoint(0,-0.4)))--
          (xpart(z.ne),ypart(placepoint(0,0.4)))--cycle,ang,0.3,0)
        withcolor couleur
      elseif style="repasse":
        draw (xpart(z.so),ypart(placepoint(0,0)))--(placepoint(valeur,0))
        withpen pencircle scaled 2bp withcolor couleur
      fi;
      if sol="non":
        draw placepoint(valeur+0.3,0.5)--placepoint(valeur,0.5)
        --placepoint(valeur,-0.5)--placepoint(valeur+0.3,-0.5)
        withpen pencircle scaled 2bp withcolor couleur
      elseif sol="oui":
        draw placepoint(valeur-0.3,0.5)--placepoint(valeur,0.5)
        --placepoint(valeur,-0.5)--placepoint(valeur-0.3,-0.5)
        withpen pencircle scaled 2bp withcolor couleur
      fi;
      if affichagesolu="non":
```

```
    long:=abs((xpart(z.ne),ypart(placepoint(0,-0.4)))-
      placepoint(valeur,-0.4));
    label.bot(TEX("$\underbrace{\hbox to"&
        decimal(long)&"pt{}}_{\hbox{Ce qui ne convient pas}}$"),
      iso(placepoint(valeur,-0.4),(xpart(z.ne),
          ypart(placepoint(0,-0.4)))));
  elseif affichagesolu="oui":
    long:=abs((xpart(z.so),ypart(placepoint(0,-0.4)))-
      placepoint(valeur,-0.4));
    label.bot(TEX("$\underbrace{\hbox to"&
        decimal(long)&"pt{}}_{\hbox{Nombres solutions}}$"),
      iso(placepoint(valeur,-0.4),(xpart(z.so),
          ypart(placepoint(0,-0.4)))));
  fi;
  if affichageval="oui":
    label.top(TEX("$"&decimal(valeur)&"$"),placepoint(valeur,0.5));
  fi;
elseif pos="sup":
  if style="hachures":
    draw hachurage(placepoint(valeur,0.4)--placepoint(valeur,-0.4)
      --(xpart(z.so),ypart(placepoint(0,-0.4)))--
      (xpart(z.so),ypart(placepoint(0,0.4)))--cycle,ang,0.3,0)
    withcolor couleur
  elseif style="repasse":
    draw placepoint(valeur,0)--(x.ne,ypart(placepoint(0,0)))
    withpen pencircle scaled 2bp withcolor couleur
  fi;
  if sol="non":
    draw placepoint(valeur-0.3,0.5)--placepoint(valeur,0.5)
    --placepoint(valeur,-0.5)--placepoint(valeur-0.3,-0.5)
    withpen pencircle scaled 2bp withcolor couleur
  elseif sol="oui":
    draw placepoint(valeur+0.3,0.5)--placepoint(valeur,0.5)
    --placepoint(valeur,-0.5)--placepoint(valeur+0.3,-0.5)
    withpen pencircle scaled 2bp withcolor couleur
  fi;
  if affichagesolu="non":
    long:=abs((xpart(z.so),ypart(placepoint(0,-0.4)))-
      placepoint(valeur,-0.4));
    label.bot(TEX("$\underbrace{\hbox to"&
        decimal(long)&"pt{}}_{\hbox{Ce qui ne convient pas}}$"),
      iso(placepoint(valeur,-0.4),(xpart(z.so),
          ypart(placepoint(0,-0.4)))));
  elseif affichagesolu="oui":
    long:=abs((xpart(z.ne),ypart(placepoint(0,-0.4)))-
      placepoint(valeur,-0.4));
    label.bot(TEX("$\underbrace{\hbox to"&
        decimal(long)&"pt{}}_{\hbox{Nombres solutions}}$"),
      iso(placepoint(valeur,-0.4),(xpart(z.ne),
          ypart(placepoint(0,-0.4)))));
  fi;
  if affichageval="oui":
```

```
        label.top(TEX("$"&decimal(valeur)&"$"),placepoint(valeur,0.5));
      fi;
    fi;
    );
  $
enddef;
```