proglin.xws
La machine à résoudre les problèmes de programmation linéaire

Guillaume CONNAN
http://gconnan.free.fr

7 février 2008 - Version 1.1

I - Ça sert à quoi ?

Les exercices de programmation linéaire au lycée sont assez standardisés, que ce soit dans les sections STG ou ES. Ce petit programme **XCAS** permet d'obtenir rapidement le polygone des contraintes et la droite qui va permettre de lire le « bénéfice » maximum ou la « dépense » minimum avec une mise à l'échelle automatisée. On se contentera de « faire bouger » la droite d'optimisation à l'aide d'un curseur et d'obtenir les coordonnées du point optimum.

II - Comment ça marche ?

a. Principe général

On entre une série d'instructions correspondant au système de contraintes :

$$\begin{cases} a_1x+b_1y+c_1 & > \text{ ou } < & 0 \\ a_2x+b_2y+c_2 & > \text{ ou } < & 0 \\ \cdots & & \\ x & > \text{ ou } < & k_1 \\ x & > \text{ ou } < & k_2 \end{cases}$$

avec les coefficients b_i positifs, ainsi que l'expression donnant le « bénéfice » ou la « dépense » en fonction de x et y. On entre alors :

```
proglin([a1x+b1y+c1,a2x+b2y+c2,...],[p,m,...],[k1,k2,...],[p,m,...],A*x+B*y)
```

avec `p` pour >0 et `m` pour <0.

b. Comment l'utiliser ?

On ouvre une fenêtre de géométrie en tapant Alt + G .
On crée d'abord un curseur représentant le « bénéfice » ou la « dépense » :

```
R:=element(0 .. Bmax)
```

`Bmax` étant la valeur maximum du curseur automatiquement calculée.
On entre ensuite les données :

```
proglin([a1x+b1y+c1,a2x+b2y+c2,...],[p,m,...],[k1,k2,...],[p,m,...],A*x+B*y)
```

On utilise le curseur pour faire « bouger » la droite d'optimisation. On détermine ainsi le point optimum. S'il s'agit de l'intersection de D_1 et D_3 par exemple, on obtient ses coordonnées en tapant :

```
I:=inter(D[1],D[3])
```

Si **XCAS** répond par exemple (1200,300), pour avoir la valeur du « bénéfice » ou de la « dépense » on entre :

```
Be(1200,300)
```

III - Un exemple

Une usine syldave produit des androïdes de deux sortes : des soldats et des ouvriers, à partir de microprocesseurs. Elle dispose au plus de 60 000 *microprocesseurs par jour.*
Il faut 30 microprocesseurs pour le cerveau d'un soldat et 80 microprocesseurs pour un cerveau d'ouvrier.
Il faut une heure de travail pour produire 20 soldats et 2 heures pour produire 20 ouvriers.
L'usine emploie 9 techniciens travaillant 10 heures par jour.
Il ne faut pas produire plus de 500 ouvriers par jour, sinon ils se regroupent dans un syndicat et ça devient ingérable.
Un soldat rapporte un bénéfice de 8€ et un ouvrier 20 €.
Comment répartir la production quotidienne pour obtenir le bénéfice maximum ?
Ce problème classique amène au système :

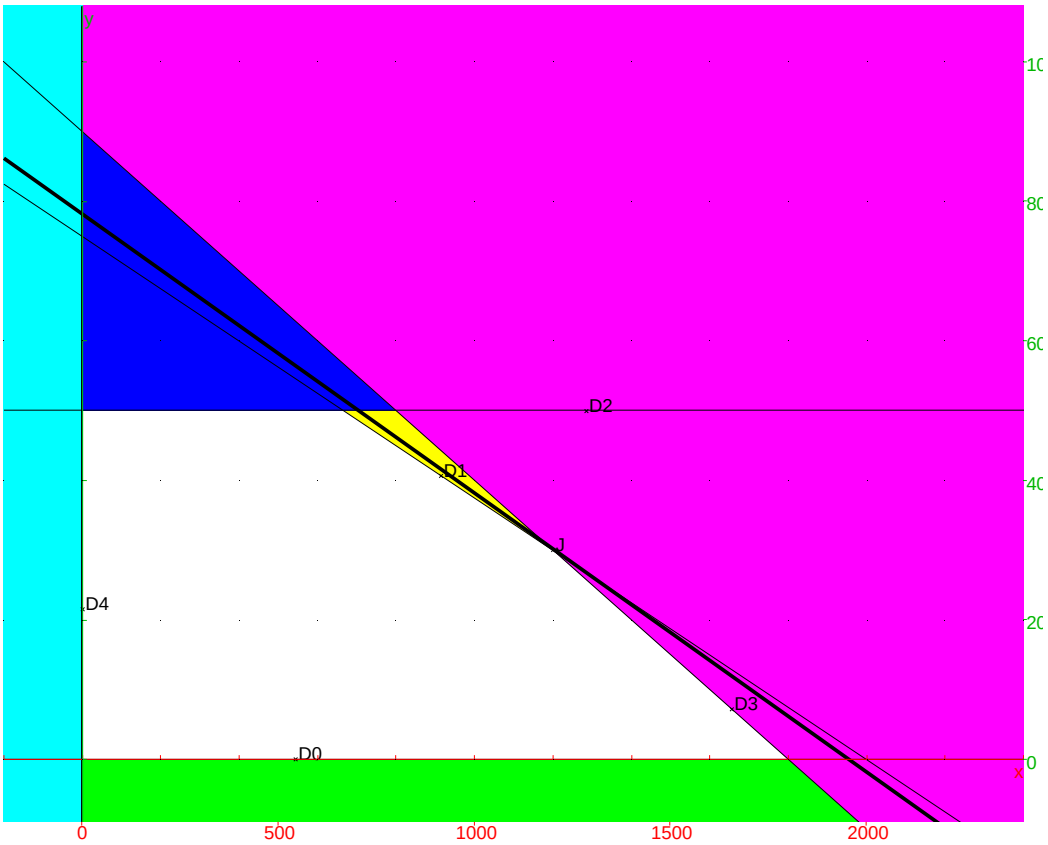
$$\begin{cases} x & \geqslant & 0 \\ y & \geqslant & 0 \\ 30x+80y-60\,000 & \leqslant & 0 \\ y-500 & \leqslant & 0 \\ 0,05x+0,1y-90 & \leqslant & 0 \end{cases}$$

De plus, le bénéfice est donné par $B = 8x+20y$.
On entre :

```
R:=element(0 .. Bmax)
```

```
proglin([y,30*x+80*y-60000,y-500,0.05*x+0.1*y-90],[p,m,m,m],[0],[p],8*x+20*y)
```

On obtient :



On fait varier la « droite-bénéfice » avec le curseur et on lit que le maximum sera obtenu à l'intersection des droites D_3 et D_1.
On obtient ses coordonnées en entrant :

```
I:=inter(D[1],D[3])
```

qui renvoie :

Réponse du logiciel

```
[point(1200.0,300.0)]
```

On obtient le bénéfice correspondant en entrant :

```
Be(1200,300)
```

et on obtient :

Réponse du logiciel

```
15600
```

IV - Insertion dans un document LaTeX

On peut bien sûr fabriquer un environnement `proglin` ainsi : on commence par créer un fichier qui sera interprété par **giac/XCAS** :

```
\begin{VerbatimOut}{proglin.cxx}
proglin(L,S,X,XS,B):={
n:=nops(L)-1;
D:=NULL;DX:=NULL;
P:=NULL;PX:=NULL;
F:=[];
Le:=NULL;
Intx:=NULL;
Inty:=NULL;
W:=[0,0,0,0];

for(k:=0;k<=n;k++){L[k]:=solve(L[k],y)[0]; if((size(solve(L[k],x))!=0) and (solve(L[k],x)!=[x])){
    Intx:=Intx,solve(L[k],x)[0]};
                  F:=append(F,unapply(L[k],x));Inty:=Inty,F[k](0)}

W[0]:=-0.1*maxnorm([Intx]); W[1]:=1.2*(max(maxnorm(X),maxnorm([Intx])));
W[2]:=-0.1*maxnorm([Inty]); W[3]:=1.2*maxnorm([Inty]);

if(nops(X)>0){
nx:=nops(X)-1;
for(k:=0;k<=nx;k++){
 Le:=Le,legende(X[k]+i*(k+1)*W[3]/(k+5),"D"+(n+1+k));
 DX:=DX,droite(x=X[k]);
 PX:=PX,if(XS[k]==p){display(polygone(1.2*W[0]+i*6*W[2],1.2*W[0]+1.2*W[3]*i,X[k]+1.2*W[3]*i,X[k]+i
     *6*W[2]),(k+5+nx)+rempli);}
              else{display(polygone(1.2*W[1]+i*6*W[2],1.2*W[1]+1.2*W[3]*i,X[k]+1.2*W[3]*i,X[k]+i*6*W
                  [2]),(k+5+nx)+rempli);}
  }
}
 for(k:=0;k<=n;k++){
 Le:=Le,legende(((5-k)*W[0]+(k+2)*W[1])/(7)+i*F[k](((5-k)*W[0]+(k+2)*W[1])/(7)),"D"+k);
 D:=D,droite(y=L[k]);
 P:=P,if(S[k]==m){display(polygone(1.2*W[0]+i*F[k](1.2*W[0]),1.2*W[0]+1.2*W[3]*i,1.2*W[1]+1.2*W[3]*i
     ,1.2*W[1]+i*F[k](1.2*W[1])),(k+2)+rempli);}
               else{display(polygone(1.2*W[0]+i*F[k](1.2*W[0]),1.2*W[0]+6*W[2]*i,1.2*W[1]+6*W[2]*i
                   ,1.2*W[1]+i*F[k](1.2*W[1])),(k+2)+rempli);}
```

```
  }
xyztrange(W[0],W[1],W[2],W[3],-10,10,-1,6,W[0],W[1],W[2],W[3],1);
Be:=unapply(B,x,y):;
//fB:=unapply(solve(B,y)[0],x);return(fB);
Bmax:=Be(maxnorm([Intx]),maxnorm([Inty]));
D:=D,DX;
P,PX,DX,D,display(droite(B=R),line_width_4),Le;
}:;
\end{VerbatimOut}
```

Puis on donne quelques instructions à **XCAS/giac** pour savoir trouver et interpréter ce fichier :

```
\begin{VerbatimOut}{proglin.giac}
maple_mode(0);
read("proglin.cxx");
Sortie:=fopen("XCASproglin.tex");
Resultat:=read("proglin.user");
Resultat:=latex(Resultat);
fprint(Sortie,Unquoted,Resultat);
fclose(Sortie);
\end{VerbatimOut}
```

On fabrique enfin un environnement LaTeX qui lira nos instructions et les enverra à **XCAS/giac** qui rendra la politesse à LaTeX en lui envoyant la figure au format PSTricks :

```
\newenvironment{proglin}
{\VerbatimEnvironment\begin{VerbatimOut}{proglin.user}}
{\end{VerbatimOut}
          \immediate\write18{giac  <proglin.giac}
          \begin{center}
            \input{XCASproglin}
          \end{center}
}
```

Ainsi, en reprenant l'exemple étudié :

```
\begin{proglin}
proglin([y,30*x+80*y-60000,y-500,0.05*x+0.1*y-90],[p,m,m,m],[0],[p],8*x+20*y)
\end{proglin}
```

donne :