Universidad Politécnica de Cataluña

Facultad de Matemáticas y Estadística

El problema de la Braquistócrona

Práctica IV

Autores:

Eduardo Peña

Laia Pomar

Diciembre 2019

1. Introducción

El objetivo de esta práctica es, dados dos puntos $O = (0, 0)$ i $F = (a, b)$, encontrar la curva de descenso más rápido de un cuerpo entre O y F sobre el cual solo actúa la fuerza de la gravedad.

La curva que trazaría un objeto bajo estas condiciones es conocida como *braquistócrona*, que corresponde a una cicloide invertida, que tal y como se menciona en el enunciado del problema, la cicloide es la curva definida por un punto de un círculo que gira sin deslizar por una línea recta.

La formulación de este problema de optimización no lineal, que nos es proporcionada en el enunciado del mismo, es, tomando como aceleración de la gravedad $g = 9,8m/s^2$:

$$\begin{aligned} min\ & \frac{1}{\sqrt{2g}} \int_a^0 \sqrt{\frac{1 + f'(x)^2}{f(x)}}\, dx \\ s.a\ \ & f(0) = 0\ ,\ \ f(a) = b. \end{aligned}$$

Como al largo del curso solo hemos trabajado con problemas de dimensión finita, para poder solucionarlo será necesario hacer una discretización del problema para poder aplicar los procesos de optimización que hemos trabajado en clase. Además, a partir de discretizar el problema, el enunciado plantea 3 variantes de este problema de optimización. Estos son:

1. Considerar las abscisas x_i como parámetros fijos y las ordenadas y_i como variables, es decir, encontrar $y = f(x)$.
2. Considerar las ordenadas y_i como parámetros y que x_i sean las variables, es decir, encontrar la función inversa $x = f^{-1}(y)$.
3. Considerar tanto a x_i como a y_i variables.

Finalmente, después de discretizar el problema y solucionar las tres variantes del problema se hará un análisis de los resultados obtenidos y una comparación entre las variantes del problema.

2. Formulación

Tal y como hemos mencionado en la introducción, el problema necesita ser discretizado para poder trabajar con él. Entonces, tal y como expone la formulación del enunciado, si una curva viene descrita por un conjunto de puntos $\{(x, f(x))\} \subset \mathbb{R}^2$, entonces el tiempo que tarda un cuerpo en recorrer dicha curva deslizándose sin rozamiento por la acción de la gravedad viene dada por:

$$\frac{1}{\sqrt{2g}} \int_a^0 \sqrt{\frac{1 + f'(x)^2}{f(x)}}\, dx \approx \frac{1}{\sqrt{2g}} \sum_{i=0}^{n-1} \sqrt{\frac{1 + f'(x)}{f(x)}} dx$$

De esta manera hemos aproximado la integral usando sumas de Riemann, en concreto la suma superior de la función integrada respecto a la partición $\mathcal{P} = \{x_0, \ldots, x_n\}$ del intervalo $[0,\ a]$. Dados una serie de puntos $(x_i, y_i) \in \mathcal{P}$, también podemos aproximar el diferencial dx y la derivada de f en un punto x_i, como:

$$f'(x_i) \approx \frac{f(x_{i+1} - f(x_i)}{x_{i+1} - x_i} = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \qquad dx \approx (x_{i+1} - x_i)$$

En consecuencia obtenemos la siguiente función objetivo que utilizaremos para la resolución del problema. Nótese que esta función es una aproximación, pero que esta mejora cuanto más refinemos la partición $\mathcal{P}$.

$$\frac{1}{\sqrt{2g}}\int_a^0 \sqrt{\frac{1+f'(x)^2}{f(x)}}\,dx \approx \frac{1}{\sqrt{2g}}\sum_{i=0}^{n-1}\sqrt{\frac{1+(\frac{y_{i+1}-y_i}{x_{i+1}-x_i})^2}{y_i}}(x_{i+1}-x_i) =$$

$$= \frac{1}{\sqrt{2g}}\sum_{i=0}^{n-1}\sqrt{\frac{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2}{y_i}}$$

2.1. Formulación variante 1: x_i fijadas y y_i variables

Esta es la primera variante de las tres propuestas para esta práctica. En esta variante se consideran como parámetros fijos los puntos del dominio x_i, y entonces, se resolverá el problema de optimización para determinar los valores y_i que minimizan el tiempo total. Consideramos una partición de $n+1$ puntos del intervalo $[0, a]$ de manera que $x_0 = 0$ y que $x_n = a$.

Teniendo en cuenta que la función objetivo a minimizar es:

$$\frac{1}{\sqrt{2g}}\sum_{i=0}^{n-1}\sqrt{\frac{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2}{y_i}}$$

Hay que considerar que $y_i \neq 0\ \forall i$. Como no consideramos ordenadas negativas y para que $y_i \neq 0\ \forall i$, impondremos un valor de tolerancia $\varepsilon = 10^{-12}$ y la condición de que $y_i \geq \varepsilon\ \forall i$. Inicializaremos entonces $y_0 = \varepsilon$ en vez de igual a 0. La última de las condidicones es que $y_n = b$. Con todo esto la formulación del problema nos queda de la siguiente manera:

$$(P_1)\begin{cases} min & \frac{1}{\sqrt{2g}}\sum_{i=0}^{n-1}\sqrt{\frac{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2}{y_i}} \\ s.a: & \\ & y_0 = \varepsilon \\ & y_n = b \\ & y_i \geq \varepsilon \quad i \in \{0,\ldots,n\} \end{cases}$$

2.2. Formulación variante 2: x_i variables y_i fijadas

En esta segunda variante del problema, fijamos los puntos de la imagen y los consideramos como parámetros, y se resolverá el problema de optimización para determinar los valores x_i que minimizan el tiempo total. Una vez más se ha de escoger una partición de $n+1$ puntos del intervalo, esta vez, $[0, b]$. Por lo que respecta a las restricciones, primeramente imponemos que los valores iniciales y finales de las abscisas de la curva sean $x_0 = 0$ y $x_n = a$, y como no tiene sentido físico que x_i sea negativa, como última restricción imponemos que $x_i \geq 0$. Con todo lo anterior el modelo nos queda de la forma siguiente:

$$(P_2)\begin{cases} min & \frac{1}{\sqrt{2g}}\sum_{i=0}^{n-1}\sqrt{\frac{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2}{y_i}} \\ s.a: & \\ & x_0 = 0 \\ & x_n = a \\ & x_i \geq 0 \quad i \in \{0,\ldots,n\} \end{cases}$$

2.2.1. Convexidad del problema en la versión 2

Veamos que el problema en esta segunda versión es un problema de optimización convexa. Sabemos que, un problema de optimización no lineal es convexo si su función objetivo y su región factible son convexas. Como podemos observar, las restricciones de esta versión del problema vienen dadas por funciones lineales, y por tanto la región factible es convexa. Estudiemos ahora la convexidad de la función objetivo:

$$f(x_0, \ldots, x_n) = \frac{1}{\sqrt{2g}} \sum_{i=0}^{n-1} \sqrt{\frac{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}{y_i}}$$

Nótese que $f \in \mathcal{C}^2$, ya que como podemos observar es suma y comoposición de funciones de clase $\mathcal{C}^2$. Por tanto y como hemos visto en clase, f será convexa si, y solo si, $\nabla^2 f(x) \succcurlyeq 0$ en la región factible. Debido a la linealidad de las funciones convexas, solo hace falta ver que la función;

$$f_k(x_0, \ldots, x_n) = \sqrt{\frac{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}{y_i}}$$

es convexa $\forall i \in \{0, \ldots, n\}$. Procedemos a calcular $\nabla^2 f_k(x)$.

$$\nabla^2 f_k(x) = \frac{1}{\sqrt{y_i}} \begin{pmatrix} \frac{(y_{i+1}-y_i)^2}{\sqrt{((x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2)^3}} & \frac{-(y_{i+1}-y_i)^2}{\sqrt{((x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2)^3}} \\ \frac{-(y_{i+1}-y_i)^2}{\sqrt{((x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2)^3}} & \frac{(y_{i+1}-y_i)^2}{\sqrt{((x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2)^3}} \end{pmatrix}$$

Por el criterio de Sylvester podemos observar que la matriz es efectivamente semidefinida positiva ya que todos los menores principales de la matrix son mayores o igual a 0. El menor de dimensión 1 x 1 es obviamente positivo y el menor de dimensión 2 x 2, es decir el determinante de la matriz, es 0.

Como hemos visto tanto la función objetivo como el conjunto de las restricciones son convexas y, por tanto, queda demostrado que el problema en su segunda versión es un problema de optimización no lineal convexo, y por tanto, cualquier mínimo local de $f(x)$ será también un mínimo absoluto de problema.

2.3. Formulación variante 3: x_i y y_i variables

En esta última versión del problema consideramos como variables tanto a x como y, y los únicos parámetros de este modelo serán n, a, b, g y ε.

La función objetivo de esta tercera versión es la misma que la utilizada en las dos versiones anteriores. Respecto a las restricciones, nuevamente es necesario que las ordenadas sean no nulas e impondremos de nuevo que $y_i \geq \varepsilon = 10^{-12}$, que implica que $y_0 = \varepsilon$. También se vuelve a repetir que $x_0 = 0$, $x_n = a$, $y_n = b$ y que todas las ordenadas sean $y_i \geq \varepsilon \ \forall i$ y que todas las abscisas $x_i \geq 0 \ \forall i$, por las razones indicadas en los apartados anteriores.

En este última versión del problema añadiremos dos restricciones extra. Para evitar que la diferencia de dos valores sucesivos de las ordenadas y de las abscisas sea confundido con un 0 de máquina, que en ese caso la función no sería inyectiva, impondremos que $x_{i+1} - x_i \geq \delta$ y que $y_{i+1} - y_i \geq \delta$, con $\delta = 10^{-6}$. Con todo esto la formulación final de esta versión del problema nos queda:

$$(P_3) \begin{cases} min \ \ \frac{1}{\sqrt{2g}} \sum_{i=0}^{n-1} \sqrt{\frac{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2}{y_i}} \\ s.a: \\ \qquad x_0 = \varepsilon \\ \qquad x_n = a \\ \qquad y_0 = \varepsilon \\ \qquad y_n = b \\ \qquad x_i \geq \varepsilon \quad i \in \{0, \ldots, n\} \\ \qquad y_i \geq \varepsilon \quad i \in \{0, \ldots, n\} \\ x_{i+1} - x_i \geq \delta \quad i \in \{0, \ldots, n-1\} \\ y_{i+1} - y_i \geq \delta \quad i \in \{0, \ldots, n-1\} \end{cases}$$

2.4. Inyectividad de la cicloide

El hecho de saber que la cicloide es la curva solución del problema nos permite hacer algunas observaciones. Nótese que la cicloide viene dada por la parametrización $c(t) = (x(t), y(t)) = (r(t - sin(t)), r(1 - cos(t)))$. Hay que ver primero que $y(t) = r(1 - cos(t)) \leq 2r$, y la igualdad se da cuando $cos(t) = -1$. Por tanto, el máximo de la cicloide está en el punto $t = \pi$. Podemos ver que $x(\pi) = \pi r$ y que $y(\pi) = 2r$, y es inmediato comprobar que sea M el máximo de la cicloide con $M = (\alpha, \beta)$, sus coordenadas satisfacen que $\frac{\alpha}{\beta} = \frac{\pi}{2}$. De esta manera cuando planteemos el problema con unos parámetros a, b tal que $\frac{a}{b} > \frac{\pi}{2}$, el intervalo $[0, a]$ contiene un subintervalo donde decrece y uno donde crece.

Visto esto, veamos como se comportan las tres variantes del problema cuando $\frac{a}{b} > \frac{\pi}{2}$. En la primera esto no genera ningún tipo de problema, ya que se tratan como variables las ordenadas $y_i = f(x_i)$. Sin embargo, en la segunda variable del problema, al fijar las ordenadas y encontrando las abscisas como $x_i = f^{-1}(y_i)$, vemos que si la función no es inyectiva entonces la función no tendrá inversa (debido a que dejará de ser inyectiva). Pero la cicloide invertida esperada en estos casos contiene parejas de puntos con la misma ordenada y_i y esto entra en contradicción con que la función sea inyectiva. Esto mismo sucede por la tercera variante del problema por la misma razón (ya que las ordenadas y_i vuelven a ser variables), debido a que la función no es inyectiva por $\frac{a}{b} > \frac{\pi}{2}$.

Como podremos ver en el apartado **Ejemplos de aplicación de los modelos** (y como se mencionará en este apartado), si ejecutamos los modelos 2 y 3 cuando $\frac{a}{b} > \frac{\pi}{2}$, veremos que la curva obtenida aproxima una cicloide con máximo en la ordenada b, seguido de un segmento paralelo al eje de abscisas que lo une con el punto $F = (a, b)$. Obviamente, este resultado dista mucho de una braquistócrona, y los modelos 2 y 3 no serán útiles para resolver el problema en estos casos.

3. Modelos AMPL

Para encontrar una solución a este problema en sus tres variantes ha sido necesaria la utilización y la implementación de AMPL, y por esa razón, se ha trasladado la formulación de este problema a este lenguaje de programación, culminando en el resultado siguiente:

3.1. Variante 1: x_i fijadas y y_i variables

```
param a;
param b;
param n;

param g:= 9.81;
param epsilon:= 1e-12;

set indice:= 0..n;

param x{i in indice} := i*a/n;

var y{i in indice} >= epsilon;

#Funcion objetivo
minimize c:
(1/sqrt(2*g))*(sum{i in indice: i != n}sqrt(((x[i+1] - x[i])^2 + (y[i+1] - y[i])^2 )/y[i]) ) ;

#Restricciones
subject to y0_igual_a_origen:
  y[0] = epsilon;
subject to yn_igual_a_destino:
  y[n] = b;

```

3.2. Variante 2: x_i variables y_i fijadas

```
param a;
param b;
param n;

param g:= 9.81;
param epsilon:= 1e-12;

set indice:= 0..n;

param y{i in indice} := i*b/n + epsilon;

var x{i in indice} >= 0;

#Funcion objetivo
minimize c:
(1/sqrt(2*g))*(sum{i in indice: i != n}sqrt(((x[i+1] - x[i])^2 + (y[i+1] - y[i])^2 )/y[i]) ) ;

#Restricciones
subject to x0_igual_a_cero:
  x[0] = 0;
subject to xn_igual_a_a:
  x[n] = a;
subject to interv_x{i in indice: i != n}:
  x[i+1] - x[i] >= epsilon;
```

3.3. Variante 3: x_i y y_i variables

```
param a;
param b;
param n;

param g:= 9.81;
param epsilon:= 1e-12;

set indice:= 0..n;

var y{i in indice} >= epsilon;

var x{i in indice} >= epsilon;

#Funcion objetivo
minimize c:
(1/sqrt(2*g))*(sum{i in indice: i != n}sqrt(((x[i+1] - x[i])^2 + (y[i+1] - y[i])^2 )/y[i]) ) ;

#Restricciones
subject to x0_igual_a_cero:
  x[0] = epsilon;
subject to xn_igual_a_a:
  x[n] = a;
subject to y0_igual_a_cero:
  y[0] = epsilon;
subject to yn_igual_a_b:
  y[n] = b;
subject to interv_x{i in indice: i != n}:
  x[i+1] - x[i] >= 1e-6;
subject to interv_y{i in indice: i != n}:
  y[i+1] - y[i] >= 1e-6;

```

4. Ejemplos de aplicación de los modelos

A continuación se muestran los resultados obtenidos al aplicar los modelos en AMPL de las 3 variaciones del problema. Para hacerlo hemos dado valor a los parámetros de entrada n, a, b y los hemos resuelto usando el solver MINOS. Para facilitar la representación gráfica en MatLab de los resultados de nuestros modelos, hemos creado un archivo .run que dispone los datos de manera entendible para MatLab creando un archivo de nombre *results.m*. Este archivo se encuentra en el anexo de este informe.

En los ejemplos de a continuación, se comparan en una mísma gráfica, no solo los resultados de los 3 modelos sino que también se compara con la cicloide teórica que debería resultar de cada entrada de datos. La cicloide viene dada por la parametrización $c(t) = (x(t), y(t)) = (r(t - sin(t)), r(1 - cos(t)))$. Esta curva se ha calculado primero encontrando el radio de la cicloide resolviendo el sistema:

$$\begin{cases} a = r(t - sin(t)) \\ b = r(1 - cos(t)) \end{cases}$$

Y finalmente, una vez calculado el radio, para obtener la cicloide resolvemos este último sistema:

$$\begin{cases} x = r(t - sin(t)) \\ y = r(1 - cos(t)) \end{cases}$$

Se ha utilizado el motor de conocimiento computacional **Wolfram Alpha** para la obtención de los resultados anteriores.

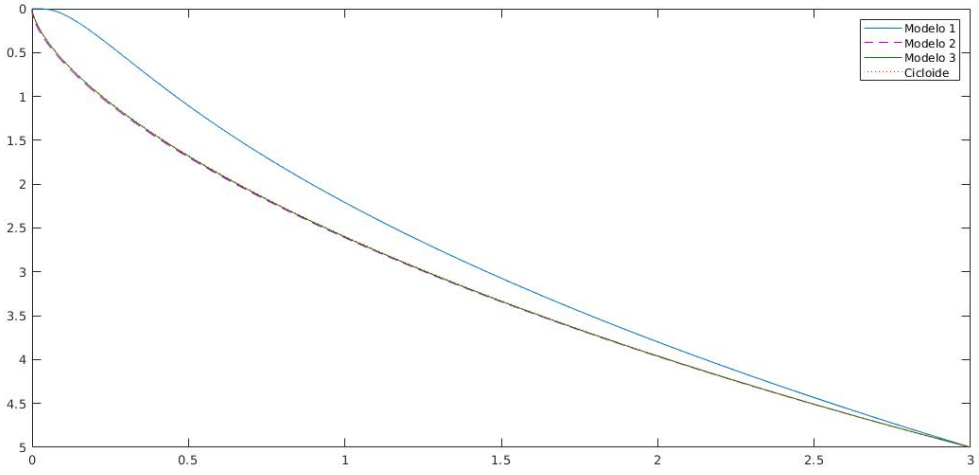

Figura 1: Comparación de los modelos con la cicloide, tomando $(a, b) = (3,5)$ i $n = 100$

El primer caso que hemos elegido para estudiar es F = (3,5) y $n = 100$. En este caso el único modelo que no se ajusta perfectamente a la cicloide es el primero, que empieza por encima y, aun que se mantiene así durante toda la gráfica, se va acercando poco a poco a la gráfica de la cicloide. Esto es debido a que la partición es poco fina y por lo tanto la suma de Riemann se distancia del valor de la integral.

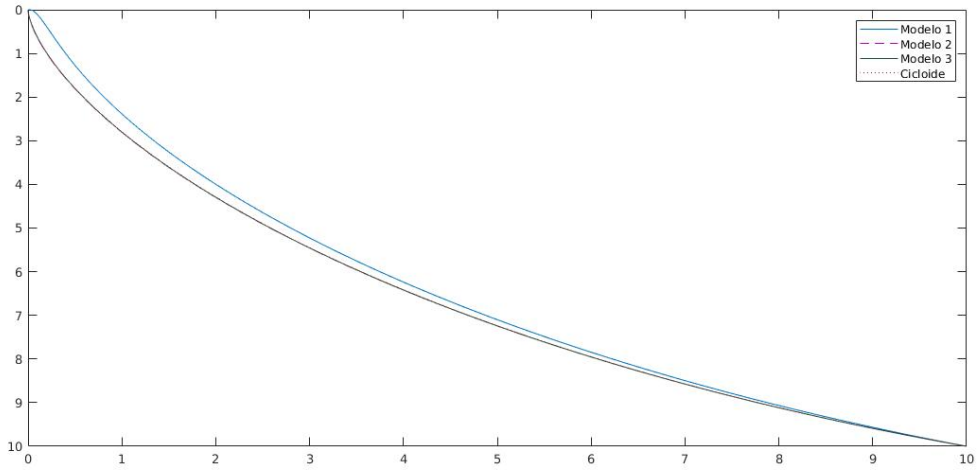

Figura 2: Comparación de los modelos con la cicloide, tomando $(a, b) = (10,10)$ i $n = 500$

En el siguiente ejemplo hemos escogido $F = (10, 10)$ y $n = 500$. Lo interesante de este caso respecto al anterior es ver que, al augmentar la n y por lo tanto hacer una partición más fina, la aproximación del modelo 1 se asemeja más a la cicloide. Los modelos 2 y 3, que ya se ajustaban mucho a la cicloide en la Figura 1, se mantienen más o menos igual.

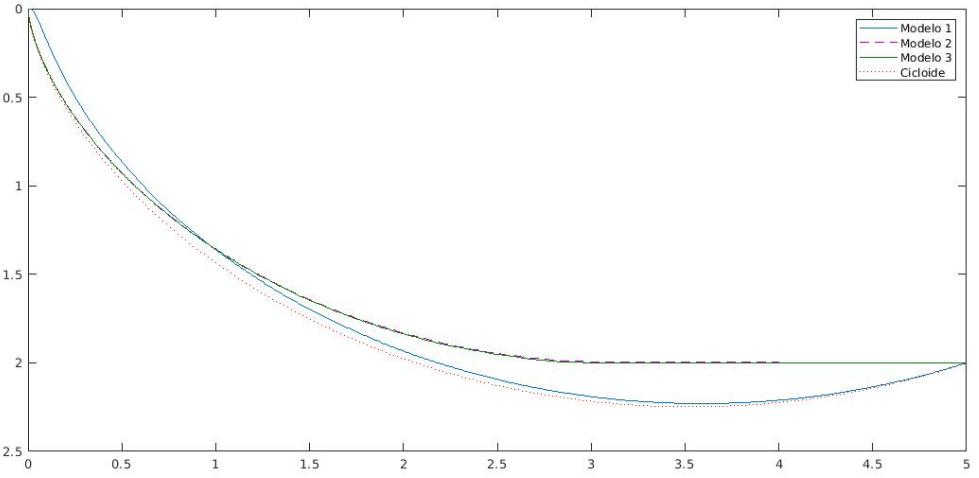

Figura 3: Comparación de los modelos con la cicloide, tomando (a,b) = (5,2) i n = 500

En el último caso hemos escogido los valores F = (5,2) y n = 500 de manera que $a/b \geq \pi/2$. De esta manera, tal y como hemos estudiado en la sección **Inyectividad de la cicloide**, la gráfica del modelo 2 no es inyectiva, y se desvia significativamente de la cicloide en la parte final. Además, podemos observar que el modelo 3 también sufre una desviación respecto la cicloide. Esto es debido a la restricción $y_{k+1} - y_k \geq \varepsilon$. En cambio el modelo 1 solo difiere ligeramente de la cicloide en la parte inicial, pero conforme avanza la gráfica se va aproximando correctamente ya que la función $f(x) = y$ está bien definida.

5. Conclusiones

En los resultados de esta práctica hemos podido ver que los tres modelos dan una aproximación bastante buena de una cicloide. Pero, para escoger que aproximación es mejor debemos tener en cuenta los parámetros de entrada.

Tal y como hemos visto en el último caso que hemos analizado si $a/b > \pi/2$, las aproximaciones de los modelos 2 y 3 difieren significantemente de la cicloide, mientras que la del modelo 1 se mantiene bastante fiel. Por lo tanto, en estos casos, es mejor usar el primer modelo.

Sin embargo, en los casos en los que no se da este problema, los modelos 2 y 3 se ajustan casi perfectamente a la cicloide incluso con particiones no muy finas, en cambio la precisión del modelo 1 disminuye bastante si lo hace la n, sobretodo en la sección inicial de la gráfica.

6. Anexo: tratamiento de los datos y archivo .Run

A continuación se encuentra el archivo .run mencionado con anterioridad que crea el archivo .m nombrado *results.m* donde se encuentran los datos de los resultados de los 3 modelos para una cierta entrada, de manera que se puedan graficar en MatLab cómodamente. Para crear la gráfica des de MatLab solamente es necesario abrir *results.m* en el programa y ejecutarlo.

```
reset;
option solver minos;
model variante1.mod;
data var1.dat;
option minos_options 'superbasics_limit=1000';
solve;
option display_precision 12;

printf " x1 = [ %f", x[0] > results.m;
for{i in indice: i != 0 }{
  printf ", %f", x[i] > results.m;
}
printf "]; \n" >results.m;
printf " y1 = [ %f", y[0] > results.m;
for{i in indice: i != 0 }{
  printf ", %f", y[i] > results.m;
}
printf "]; \n" >results.m;
close results.m;
reset;
option solver minos;
model variante2.mod;
data var1.dat;
option minos_options 'superbasics_limit=1000';
solve;
option display_precision 10;
printf " x2 = [ %f", x[0] >> results.m;
for{i in indice: i != 0 }{
  printf ", %f", x[i] >> results.m;
}
printf "]; \n" >>results.m;
printf " y2 = [ %f", y[0] >> results.m;
for{i in indice: i != 0 }{
  printf ", %f", y[i] >> results.m;
}
printf "]; \n" >>results.m;
close results.m;
reset;
option solver minos;
model variante3.mod;
data var1.dat;
option minos_options 'superbasics_limit=1000';
solve;
option display_precision 10;
printf " x3 = [ %f", x[0] >> results.m;
for{i in indice: i != 0 }{
  printf ", %f", x[i] >> results.m;
}
printf "]; \n" >>results.m;
printf " y3 = [ %f", y[0] >> results.m;
for{i in indice: i != 0 }{
  printf ", %f", y[i] >> results.m;
}
printf "]; \n" >>results.m;
printf "plot(x1,-y1); \n hold on; \n  plot(x2,-y2); \n  plot(x3,-y3);\n " >> results.m;
close results.m;

```