Curso de Métodos Numéricos

DEMAT, Universidad de Guanajuato

Clase 5: Solución de ecuaciones no lineales (Parte 2)

- Ejemplos del método de Newton–Raphson
- Método de Newton-Raphson para funciones implícitas
- Método de la secante
- Repaso de algebra lineal
- Escritura de archivos de texto y apuntadores a funciones
- Graficación usando Gnuplot

MAT–251

Dr. Joaquín Peña Acevedo
CIMAT A.C.
e-mail: joaquin@cimat.mx

Resumen

Algoritmo 1: Método de Newton-Raphson

Entrada: Una función $f : D \subset \mathbb{R} \to \mathbb{R}$ y su derivada $f' : D \to \mathbb{R}$. Un punto $x_0 \in D$. El número máximo de iteraciones N y una tolerancia $\tau > 0$.

Resultado: Aproximación de la raíz x^* de la función f.

Hacer $f_0 = f(x_0)$;

for $k = 1, 2, \ldots, N$ **do**

  if $|f_k| < \tau$ **then**

    $x^* = x_k$;

    Terminar;

  else

    $f'_k = f'(x_k)$;

    $x_{k+1} = x_k - \dfrac{f_k}{f'_k}$ si $f'_k \neq 0$;

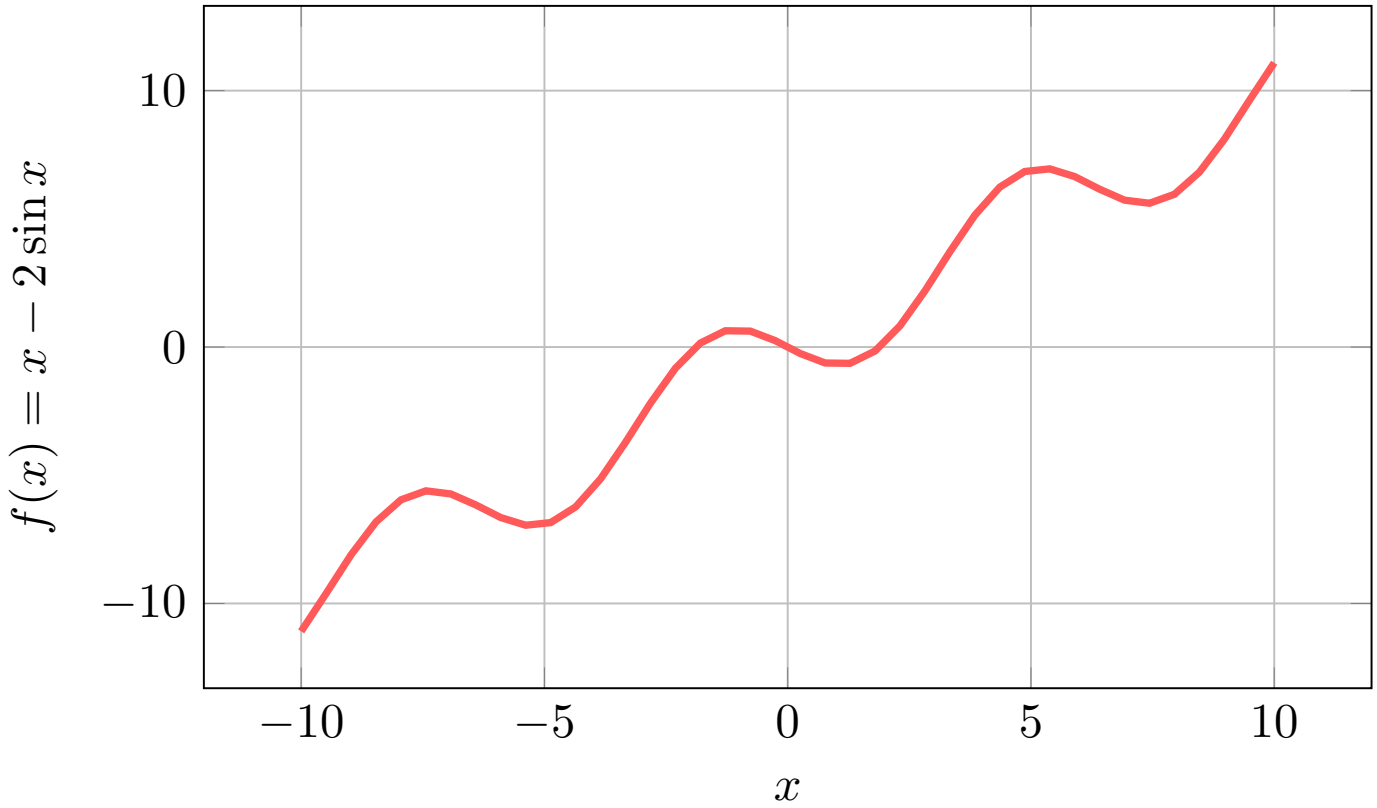
    $f_{k+1} = f(x_{k+1})$;

  end

end

Ejemplo 3 (I)

Al calcular la raíz de $x - 2\sin(x) = 0$ se tiene el siguiente comportamiento.



Ejemplo 3 (II)

k	x_k	$f(x_k)$	$f'(x_k)$
1	1.1000	-6.8241e-01	9.2808e-02
2	8.4530	6.8012e+00	2.1277e+00
3	5.2564	6.9677e+00	-3.5168e-02
4	203.3842	2.0192e+02	2.3654e+00
5	118.0193	1.1998e+02	5.8385e-01
6	-87.4709	-8.8419e+01	-7.6115e-01
7	-203.6359	-2.0256e+02	2.6864e+00
8	-128.2322	-1.2715e+02	2.6805e+00
9	-80.7976	-8.2344e+01	-2.6848e-01
10	-387.5048	-3.8928e+02	1.9269e+00
11	-185.4782	-1.8573e+02	2.9846e+00
12	-123.2497	-1.2458e+02	2.4936e+00
13	-73.2891	-7.5006e+01	2.0255e+00
14	-36.2577	-3.8241e+01	7.4189e-01
15	15.2874	1.4471e+01	2.8257e+00
16	10.1662	1.1517e+01	2.4750e+00
17	5.5129	6.9056e+00	-4.3544e-01
18	21.3716	2.0210e+01	2.6283e+00

Ejemplo 3 (III)

k	x_k	$f(x_k)$	$f'(x_k)$
19	13.6822	1.1886e+01	1.2107e-01
20	-84.4915	-8.3841e+01	2.8911e+00
21	-55.4920	-5.7233e+01	1.6392e-02
22	3435.9663	3.4376e+03	-1.8598e-01
23	21919.2339	2.1920e+04	2.8842e+00
24	14319.1544	1.4320e+04	-9.4962e-01
25	29398.3953	2.9400e+04	-6.1758e-01
26	77002.7809	7.7001e+04	2.3983e+00
27	44895.9860	4.4895e+04	2.7410e+00
28	28516.7622	2.8518e+04	2.7300e+00
29	18070.5576	1.8070e+04	-9.8641e-01
30	36389.7828	3.6391e+04	2.5101e+00
31	21891.8200	2.1890e+04	2.7971e-01
32	-56369.0215	-5.6368e+04	2.6779e+00
33	-35319.5791	-3.5318e+04	1.4437e+00
34	-10855.5555	-1.0858e+04	1.4323e+00
35	-3275.1312	-3.2731e+03	1.0417e+00
36	-133.0772	-1.3127e+02	1.4727e-01
37	758.2943	7.6014e+02	1.7795e+00
38	331.1252	3.3303e+02	1.6154e+00

Ejemplo 3 (IV)

k	x_k	$f(x_k)$	$f'(x_k)$
42	27.9465	2.7302e+01	2.8935e+00
43	18.5106	1.9176e+01	-8.8617e-01
44	40.1493	3.8874e+01	2.5407e+00
45	24.8486	2.5409e+01	-9.1981e-01
46	52.4732	5.0865e+01	2.1894e+00
47	29.2406	3.0886e+01	2.1367e+00
48	14.7857	1.3192e+01	2.2081e+00
49	8.8113	7.6600e+00	2.6354e+00
50	5.9047	6.6437e+00	-8.5847e-01
51	13.6438	1.1882e+01	5.2737e-02
52	-211.6678	-2.1352e+02	1.7598e+00
53	-90.3366	-8.8945e+01	2.4363e+00
54	-53.8290	-5.4648e+01	2.8246e+00
55	-34.4818	-3.4330e+01	2.9943e+00
56	-23.0164	-2.4726e+01	2.0378e+00
57	-10.8826	-1.2870e+01	1.2255e+00
58	-0.3807	3.6245e-01	-8.5680e-01
59	0.0423	-4.2292e-02	-9.9821e-01
60	-0.0001	5.0601e-05	-1.0000e+00
61	0.0000	-8.6372e-14	-1.0000e+00

Hay propuestas para acelerar el método de Newton, por ejemplo:

Ejemplo 3 (V)

Trevor J.McDougall, Simon J.Wotherspoon. "A simple modification of Newton's method to achieve convergence of order $1+\sqrt{2}$". *Applied Mathematics Letters*. Vol. 29,2014, pp. 20-25.

En el artículo proponen

$$
\begin{aligned}
x_0^* &= x_0 - \frac{f(x_0)}{f'(x_0)} \\
x_1 &= x_0 - \frac{f(x_0)}{f'\left(\frac{1}{2}[x_0 + x_0^*]\right)} \\
&\vdots \\
x_k^* &= x_k - \frac{f(x_k)}{f'\left(\frac{1}{2}[x_{k-1} + x_{k-1}^*]\right)} \\
x_{k+1} &= x_k - \frac{f(x_k)}{f'\left(\frac{1}{2}[x_k + x_k^*]\right)}
\end{aligned}
$$

Ejemplo 3 (VI)

Aplicando este algoritmo a la ecuación $x - 2\sin(x) = 0$ se obtiene lo siguiente:

k	x_k	$f(x_k)$	$f'(x_k)$
0	1.1000	-6.8241e-01	9.2808e-02
1	1.8827	-2.0803e-02	1.6137e+00
2	1.8827	-2.0803e-02	8.7187e-01
3	1.9066	1.8242e-02	1.6364e+00
4	1.8954	-1.3417e-04	1.6484e+00
5	1.8955	-8.5285e-07	1.6380e+00
6	1.8955	4.0422e-11	1.6380e+00

Comparamos el método de Newton-Raphson con esta propuesta para el caso en que $f(x) = (x-2)^9$ y $f'(x) = 9(x-2)^8$.

Ejemplo 3 (VII)

	NR		NR-Modif	
k	x_k	$f(x_k)$	x_k	$f(x_k)$
0	10.0000	1.3422e+08	10.0000	1.3422e+08
1	9.1111	4.6498e+07	8.5958	2.3626e+07
2	8.3210	1.6109e+07	8.5958	2.3626e+07
3	7.6187	5.5807e+06	8.3486	1.6753e+07
4	6.9944	1.9334e+06	7.7440	6.8068e+06
5	6.4394	6.6980e+05	7.3206	3.4170e+06
6	5.9462	2.3205e+05	6.8879	1.5927e+06
7	5.5077	8.0390e+04	6.5039	7.6253e+05
8	5.1180	2.7850e+04	6.1455	3.6160e+05
9	4.7715	9.6484e+03	5.8172	1.7206e+05
10	4.4636	3.3426e+03	5.5144	8.1773e+04
11	4.1898	1.1580e+03	5.2357	3.8880e+04
12	3.9465	4.0118e+02	4.9791	1.8483e+04
13	3.7302	1.3898e+02	4.7429	8.7872e+03
14	3.5380	4.8150e+01	4.5254	4.1775e+03
15	3.3671	1.6681e+01	4.3251	1.9860e+03
16	3.2152	5.7789e+00	4.1407	9.4418e+02

Ejemplo 3 (VIII)

	NR		NR-Modif	
k	x_k	$f(x_k)$	x_k	$f(x_k)$
17	3.0802	2.0020e+00	3.9710	4.4887e+02
18	2.9602	6.9359e-01	3.8147	2.1340e+02
19	2.8535	2.4029e-01	3.6708	1.0145e+02
20	2.7586	8.3245e-02	3.5383	4.8230e+01
21	2.6744	2.8839e-02	3.4163	2.2929e+01
22	2.5994	9.9910e-03	3.3040	1.0901e+01
23	2.5328	3.4613e-03	3.2006	5.1823e+00
24	2.4736	1.1991e-03	3.1054	2.4637e+00
25	2.4210	4.1542e-04	3.0177	1.1713e+00
26	2.3742	1.4392e-04	2.9370	5.5683e-01
27	2.3326	4.9859e-05	2.8627	2.6472e-01
28	2.2957	1.7273e-05	2.7943	1.2585e-01
29	2.2628	5.9841e-06	2.7313	5.9831e-02
30	2.2336	2.0731e-06	2.6733	2.8444e-02
31	2.2077	7.1822e-07	2.6199	1.3523e-02
32	2.1846	2.4882e-07	2.5708	6.4288e-03
33	2.1641	8.6201e-08	2.5255	3.0563e-03
34	2.1458	2.9863e-08	2.4838	1.4530e-03

Ejemplo 3 (IX)

	NR		NR-Modif	
k	x_k	$f(x_k)$	x_k	$f(x_k)$
35	2.1296	1.0346e-08	2.4455	6.9076e-04
36	2.1152	3.5842e-09	2.4101	3.2840e-04
37	2.1024	1.2417e-09	2.3776	1.5612e-04
38	2.0911	4.3018e-10	2.3477	7.4222e-05
39	2.0809	1.4903e-10	2.3201	3.5286e-05
40	2.0719	5.1630e-11	2.2947	1.6775e-05
41	2.0639	1.7887e-11	2.2713	7.9750e-06
42	2.0568	6.1966e-12	2.2498	3.7914e-06
43	2.0505	2.1468e-12	2.2300	1.8025e-06
44	2.0449	7.4372e-13	2.2118	8.5691e-07
45	2.0399	2.5765e-13	2.1950	4.0738e-07
46	2.0355	8.9262e-14	2.1795	1.9367e-07
47	2.0315	3.0924e-14	2.1653	9.2074e-08
48	2.0280	1.0713e-14	2.1522	4.3773e-08
49	2.0249	3.7115e-15	2.1401	2.0810e-08
50	2.0222	1.2858e-15	2.1290	9.8932e-09

Ejemplo 3 (X)

Nota: Sobre la tolerancia τ para terminar el algoritmo cuando $|f(x_k)| < \tau$, usualmente se toma τ que dependa del épsilon de la máquina, por ejemplo usando doble precisión:

$$\begin{aligned} \tau &= \sqrt{\epsilon_m} \approx 1.4901 \times 10^{-8}, \\ \tau &= \sqrt[3]{\epsilon_m^2} \approx 3.6669 \times 10^{-11}, \\ \tau &= \sqrt[3]{\epsilon_m} \approx 6.0555 \times 10^{-6}. \end{aligned}$$

Ejemplo 4 (I)

Encuentre una aproximación para $\sqrt{R}$, donde R es un número positivo.

Newton-Raphson para funciones implícitas (I)

En este caso se quiere determinar el valor de la función $y = y(x)$ de modo que $F(x, y) = 0$.

Una manera para determinar una pareja (x^*, y^*) en el que se cumpla la ecuación implícita, es fijar un valor x, y dado este valor, se resuelve la ecuación para y, por lo que se puede aplicar el método de Newton de la siguiente forma

$$y_{k+1} = y_k - \frac{F(x, y_k)}{F_y(x, y_k)}.$$

Ejemplo.
Considere la ecuación no lineal

$$y + x = e^{x-y}.$$

Fijamos el valor $x_0 = 1$ y $y_0 = 5$. Definimos

$$F(x, y) = y + x - e^{x-y}.$$

Newton-Raphson para funciones implícitas (II)

La sucesión de puntos es

$$y_{k+1} = y_k - \frac{y_k + x_0 - e^{x_0 - y_k}}{1 + e^{x_0 - y_k}}.$$

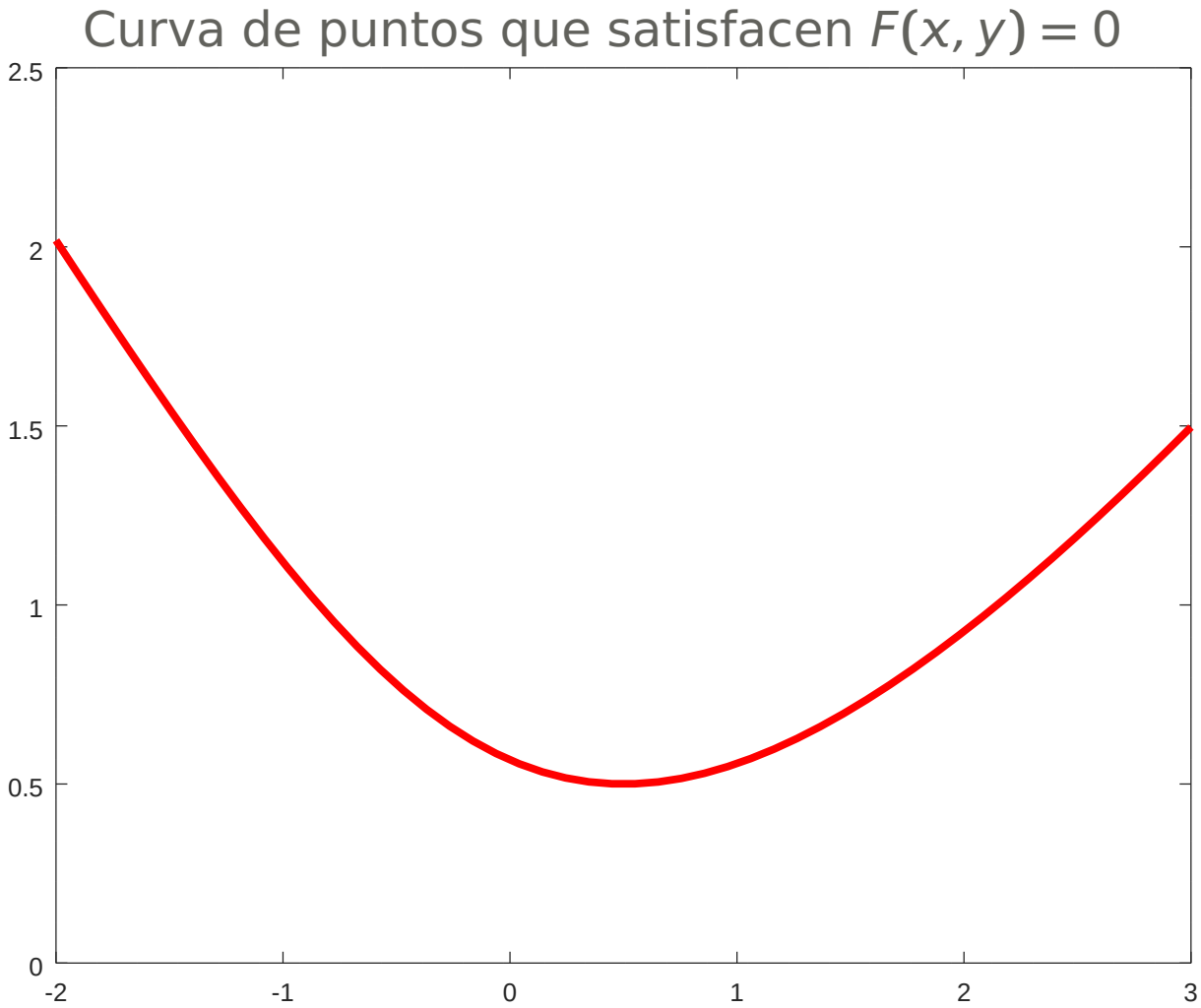
k	y_k	$F(x_0, y_k)$	$F_y(x_0, y_k)$	y_{k+1}
1	5.0000	5.9817e+00	1.0183e+00	-0.8741
2	-0.8741	-6.3890e+00	7.5149e+00	-0.0239
3	-0.0239	-1.8080e+00	3.7841e+00	0.4539
4	0.4539	-2.7268e-01	2.7266e+00	0.5539
5	0.5539	-8.3536e-03	2.5622e+00	0.5571
6	0.5571	-8.2939e-06	2.5572e+00	0.5571
7	0.5571	-8.1901e-12	2.5571e+00	0.5571

Variando el valor de x_0 se puede generar la curva de nivel 0 de la función $F(x, y)$.

Newton-Raphson para funciones implícitas (III)

Problema	Num. Iter.	x_0	y_k	$F(x_0, y_k)$
1	4	-2.0000	2.0180e+00	-3.0247e-14
2	4	-1.8980	1.9199e+00	4.5103e-17
3	4	-1.7959	1.8227e+00	-5.5511e-17
4	4	-1.6939	1.7266e+00	-6.2450e-17
5	4	-1.5918	1.6317e+00	4.8572e-17
6	4	-1.4898	1.5382e+00	-1.3878e-17
7	4	-1.3878	1.4465e+00	9.7145e-17
8	4	-1.2857	1.3569e+00	-1.1102e-16
9	4	-1.1837	1.2697e+00	0.0000e+00
10	4	-1.0816	1.1853e+00	5.5511e-17
11	4	-0.9796	1.1041e+00	6.9389e-17
12	4	-0.8776	1.0265e+00	5.5511e-17
13	4	-0.7755	9.5305e-01	-5.5511e-17
14	4	-0.6735	8.8411e-01	5.5511e-17
15	4	-0.5714	8.2012e-01	0.0000e+00
⋮	⋮	⋮	⋮	⋮
41	4	2.0816	9.6695e-01	-1.8519e-13
42	4	2.1837	1.0195e+00	-2.8022e-13
43	4	2.2857	1.0739e+00	-4.0989e-13
44	4	2.3878	1.1299e+00	-5.8087e-13
45	4	2.4898	1.1876e+00	-8.0380e-13
46	4	2.5918	1.2467e+00	-1.0854e-12
47	4	2.6939	1.3073e+00	-1.4335e-12
48	4	2.7959	1.3692e+00	-1.8581e-12
49	4	2.8980	1.4323e+00	-2.3670e-12
50	4	3.0000	1.4967e+00	-2.9701e-12

Newton-Raphson para funciones implícitas (IV)

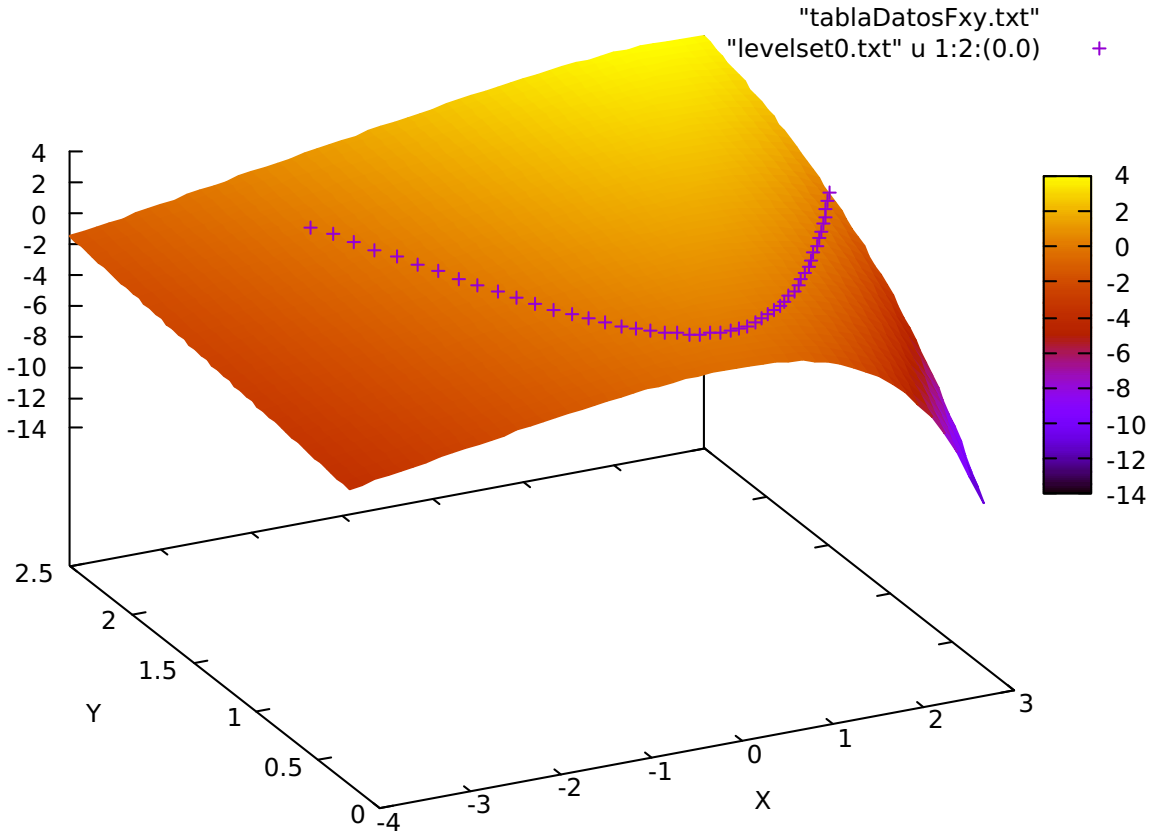


Newton-Raphson para funciones implícitas (V)

La gráfica 3D muestra que hay un cruce por el plano $z = 0$. Para identificar esta parte, se grafica los puntos encontrados por el algoritmo sobre la supercie usando el código:

```
set xlabel 'X'
set ylabel 'Y'
splot "tablaDatosFxy.txt"  with pm3d, \
      "levelset0.txt" u 1:2:(0.0) w p
```

Newton-Raphson para funciones implícitas (VI)



Método de la secante (I)

En este caso, tomamos una aproximación de la derivada

$$f'(x) \approx \frac{f(x+h)-f(x)}{h}$$

En particular, si tomamos dos puntos consecutivos de la sucesión generada para hacer la aproximación, obtenemos

$$f'(x_n) \approx \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}}$$

Reemplazamos en el método de Newton-Raphson la evaluación de la derivada por la aproximación anterior. Lo que resulta es el método de la secante para generar la secuencia de puntos:

$$x_{n+1} = x_n - \frac{x_n-x_{n-1}}{f(x_n)-f(x_{n-1})} f(x_n)$$

Método de la secante (II)

Algoritmo 2: Método de la secante

Entrada: Una función continua $f : D \subset \mathbb{R} \to \mathbb{R}$ y dos puntos $x_0, x_1 \in D$.
Número máximo de iteraciones N y una tolerancia $\tau > 0$.

Resultado: Aproximación de la raíz x^* de la función f.

Hacer $f_0 = f(x_0)$, $f_1 = f(x_1)$;

for $k = 1, 2, \ldots, N$ **do**

  if $|f_k| < \tau$ **then**

    $x^* = x_k$;

    Terminar;

  else

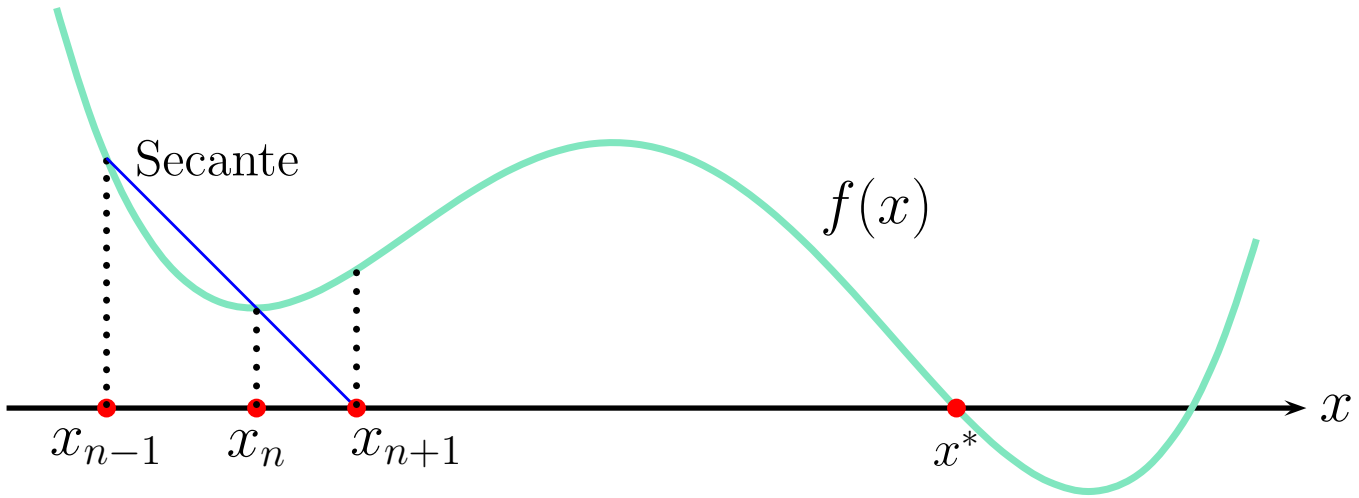
    $$x_{k+1} = x_k - f_k \frac{x_k - x_{k-1}}{f_k - f_{k-1}};$$

    $f_{k+1} = f(x_{k+1})$;

  end

end

Método de la secante (III)



Ejemplo. Consideremos la función $f(x) = x - 2\sin x$:

k	x_k	$f(x_k)$	$\frac{f(x_k)-f(x_{k-1})}{x_k-x_{k-1}}$
0	1.1000	-6.8241e-01	
1	8.4530	6.8012e+00	1.0178e+00
2	1.7705	-1.8975e-01	1.0462e+00
3	1.9519	9.5351e-02	1.5719e+00
4	1.8912	-6.9897e-03	1.6871e+00
5	1.8954	-2.2057e-04	1.6339e+00
6	1.8955	5.4763e-07	1.6379e+00
7	1.8955	-4.2673e-11	1.6380e+00

Unidad 3:

Solución de sistemas de ecuaciones lineales

Conceptos de algebra lineal

Sea V un espacio vectorial sobre el campo $\mathbb{F}$, con $\dim V = n$. Sea $\mathcal{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m\} \subset V$. Entonces

- $\mathcal{V}$ es un conjunto generador si
- $\mathcal{V}$ es un conjunto linealmente independiente si
- $\mathcal{V}$ es una base para V si
- $\mathcal{V}$ es un conjunto de vectores ortogonales si
- $\mathcal{V}$ es un conjunto ortonormal si

Conceptos de algebra lineal

Sea V un espacio vectorial sobre el campo $\mathbb{F}$, con $\dim V = n$. Sea $\mathcal{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m\} \subset V$. Entonces

- $\mathcal{V}$ es un conjunto generador si para todo $v \in V$ existen $\alpha_1, \ldots, \alpha_m \in \mathbb{F}$ tales que $v = \alpha_1 \boldsymbol{v}_1 + \cdots + \alpha_m \boldsymbol{v}_m$.
- $\mathcal{V}$ es un conjunto linealmente independiente si $m \leq n$ y no existen $\alpha_1, \ldots, \alpha_m \in \mathbb{F}$, no todos nulos, tales que $\alpha_1 \boldsymbol{v}_1 + \cdots + \alpha_m \boldsymbol{v}_m = \boldsymbol{0}$.
- $\mathcal{V}$ es una base para V si $m = n$, es un conjunto generador linealmente independiente.
- $\mathcal{V}$ es un conjunto de vectores ortogonales si $\boldsymbol{v}_j^\top \boldsymbol{v}_i = 0$ para $j \neq i$.
- $\mathcal{V}$ es un conjunto ortonormal si es ortogonal y $\boldsymbol{v}_i^\top \boldsymbol{v}_i = 1$.

Matrices

$\boldsymbol{A} \in \mathbb{R}^{m \times n}, \quad \boldsymbol{v} \in \mathbb{R}^{n},$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_n \end{bmatrix}, \qquad \boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix},$$

con $\boldsymbol{A}_i \in \mathbb{R}^m$, entonces el producto de $\boldsymbol{A}$ y $\boldsymbol{v}$ es

Matrices

$\boldsymbol{A} \in \mathbb{R}^{m \times n}, \quad \boldsymbol{v} \in \mathbb{R}^{n},$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_n \end{bmatrix}, \qquad \boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix},$$

con $\boldsymbol{A}_i \in \mathbb{R}^m$, entonces el producto de $\boldsymbol{A}$ y $\boldsymbol{v}$ es

$$\boldsymbol{A}\boldsymbol{v} = \sum_{j=1}^{n} v_j \boldsymbol{A}_j$$

- El espacio columna $\mathcal{C}(\boldsymbol{A})$ de $\boldsymbol{A}$ es
- El espacio fila $\mathcal{R}(\boldsymbol{A})$ de $\boldsymbol{A}$ es
- El rango de la matriz es

Matrices

$\boldsymbol{A} \in \mathbb{R}^{m \times n}, \quad \boldsymbol{v} \in \mathbb{R}^n,$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_n \end{bmatrix}, \qquad \boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix},$$

con $\boldsymbol{A}_i \in \mathbb{R}^m$, entonces el producto de $\boldsymbol{A}$ y $\boldsymbol{v}$ es

$$\boldsymbol{A}\boldsymbol{v} = \sum_{j=1}^{n} v_j \boldsymbol{A}_j$$

- El espacio columna $\mathcal{C}(\boldsymbol{A})$ de $\boldsymbol{A}$ es el conjunto de todas las combinaciones lineales de sus columnas.
- El espacio fila $\mathcal{R}(\boldsymbol{A})$ de $\boldsymbol{A}$ es el conjunto de todas las combinaciones lineales de sus filas y $\dim \mathcal{C}(\boldsymbol{A}) = \dim \mathcal{R}(\boldsymbol{A})$.
- El rango de la matriz es la dimensión del espacio fila y $\text{rank}(\boldsymbol{A}) \leq \min\{m, n\}$ y es de rango completo si $\text{rank}(\boldsymbol{A}) = \min\{m, n\}$.

Matrices

El espacio nulo de $\boldsymbol{A}$ es el conjunto

Matrices

El espacio nulo de $\boldsymbol{A}$ es el conjunto

$$null(\boldsymbol{A}) = Ker(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{R}^n \ : \ \boldsymbol{A}\boldsymbol{v} = \boldsymbol{0}\}$$

Se puede ver que $Ker(\boldsymbol{A})$ es un subespacio de $\mathbb{R}^n$ y su dimensión se llama la nulidad de $\boldsymbol{A}$.

$$rank(\boldsymbol{A}) + null(\boldsymbol{A}) = n$$

Sistemas de ecuaciones lineales (I)

Consideremos el caso en que tenemos tantas ecuaciones como incógnitas, de la forma

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \qquad i = 1, 2, \ldots, n.$$

Entonces podemos definir

$$\boldsymbol{A} \in \mathbb{R}^{n \times n}, \qquad \boldsymbol{b} \in \mathbb{R}^n.$$

$$\boldsymbol{A} = [a_{ij}] = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}, \qquad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

Queremos determinar $\boldsymbol{x} \in \mathbb{R}^n$ tal que

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}.$$

Sistemas de ecuaciones lineales (II)

Para una matriz $\boldsymbol{A}$ cuadrada decimos que es invertible si existe una matriz $\boldsymbol{A}^{-1}$ tal que

$$\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}.$$

El producto de matrices invertibles, es invertible.

Si $\boldsymbol{A}$ es invertible, entonces $\alpha\boldsymbol{A}$, para $\alpha \neq 0$, y $\boldsymbol{A}^{\top}$ son invertibles. Además,

- $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$ sólo tiene la solución trivial.
- $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ sólo tiene solución única.
- $\boldsymbol{A}$ es de rango completo.
- $\det \boldsymbol{A} \neq 0$.

Sistemas de ecuaciones lineales

- Se dice que el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente si al menos tiene una solución.

Sistemas de ecuaciones lineales

- Se dice que el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente si al menos tiene una solución.
- Si $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente, sus soluciones son de la forma

$$\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z}, \qquad \boldsymbol{Ay} = \boldsymbol{b}, \qquad \boldsymbol{z} \in Ker(\boldsymbol{A}).$$

Sistemas de ecuaciones lineales

- Se dice que el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente si al menos tiene una solución.
- Si $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente, sus soluciones son de la forma

$$\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z}, \qquad \boldsymbol{Ay} = \boldsymbol{b}, \qquad \boldsymbol{z} \in Ker(\boldsymbol{A}).$$

- Cuando $\boldsymbol{A}$ es invertible, la solución de $\boldsymbol{Ax} = \boldsymbol{b}$ es

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$$

Sistemas de ecuaciones lineales

- Se dice que el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente si al menos tiene una solución.
- Si $\boldsymbol{Ax} = \boldsymbol{b}$ es consistente, sus soluciones son de la forma

$$\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z}, \qquad \boldsymbol{Ay} = \boldsymbol{b}, \qquad \boldsymbol{z} \in Ker(\boldsymbol{A}).$$

- Cuando $\boldsymbol{A}$ es invertible, la solución de $\boldsymbol{Ax} = \boldsymbol{b}$ es

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$$

- El cálculo de la inversa de una matriz no es trivial.

$$\boldsymbol{A}^{-1} = \frac{\operatorname{adj} \boldsymbol{A}}{\det \boldsymbol{A}}, \qquad (\operatorname{adj} \boldsymbol{A})_{ij} = (-1)^{i+j} \det \boldsymbol{A}(j|i)$$

Cálculo del determinante (I)

La expansión por cofactores para calcular el determinante de una matriz $\boldsymbol{A} = [a_{ij}]$ de tamaño $n \times n$ desarrollada por la fila i-ésima es

$$\det \boldsymbol{A} = \sum_{k=1}^{n} a_{ik}(-1)^{i+k} \det \boldsymbol{A}(i,k)$$

donde $\boldsymbol{A}(i,k)$ es la matriz formada al remover la fila i y la columna k de $\boldsymbol{A}$. Si $f(n-1)$ es el número de multiplicaciones para calcular $\det \boldsymbol{A}(i,k)$. La fórmula de recurrencia es

$$f(n) = nf(n-1) + n, \quad \text{con} \quad f(2) = 2.$$

Esto da

$$f(n) = n! \sum_{k=1}^{n-1} \frac{1}{k!} = \lfloor n!(e-1) \rfloor - 1$$

En resumen, el cálculo del determinante es de orden $O(n!)$.

Sistemas de ecuaciones lineales (II)

Por ejemplo, para una matriz de tamaño $n = 50$ se tiene que $f(50) \approx 10^{62}$.

Si un procesador i7 puede realizar del orden de 200 gigaFLOPS (FLOPS = numéro de operaciones de punto flotante por segundo), es decir, del orden de 10^{11} FLOPS.
Entonces el tiempo de cómputo para calcular el determinante sería del orden de

$$\frac{10^{62}\ \text{FLOP}}{10^{11}\ \text{FLOPS}} = 10^{51}\ \text{segundos.}$$

Sistemas de ecuaciones lineales (II)

Por ejemplo, para una matriz de tamaño $n = 50$ se tiene que $f(50) \approx 10^{62}$.

Si un procesador i7 puede realizar del orden de 200 gigaFLOPS (FLOPS = numéro de operaciones de punto flotante por segundo), es decir, del orden de 10^{11} FLOPS.
Entonces el tiempo de cómputo para calcular el determinante sería del orden de

$$\frac{10^{62}\ \text{FLOP}}{10^{11}\ \text{FLOPS}} = 10^{51}\ \text{segundos.}$$

(El universo tiene una edad del orden de 10^{17} segundos).

- Hay que usar otras estrategias para calcular la solución del sistema que no esté basada en el cálculo de determinantes.

Apuntadores a funciones (I)

Para reutilizar el código de los algoritmos, no es una buena práctica que queden sólo se aplican una función particular. Por ejemplo:

```
double fncf1(double x) {
    return(x*x + 1);
}

double der_fncf1(double x) {
    return(2*x);
}

double Newton_Raphson(double x0, int Nmax, double tol) {
    double fk = fncf1(x0);
    double derfk = der_fncf1(x0);
    // ...
}
```

En este caso, este algoritmo no se puede aplicar a otra función que esté declarada en el mismo código.
Para evitar esto, se requiere que tanto la función y su derivada sean pasados al algoritmos como paramétros.

Apuntadores a funciones (II)

- Un apuntador a una función es una variable que almacena la dirección de una función.
- Mediante el apuntador a la función se puede llamar la función y ejecutarla.
- Esto permite que en el algoritmo se codifique el uso de la función mediante el apuntador, sin que quede fijo en el algoritmo el uso de una función particular.

El siguiente ejemplo muestra como usar apuntadores a funciones.

```
void evalFnc(double x0, double x1, int n, double (*ptr_fnc)(double)) {
    double x, y;
    printf("      x                y\n");
    printf("-----------------------------\n");
    for(int i=0; i<n; i++) {
        x = x0 + i*(x1-x0)/(n-1);
        y = ptr_fnc(x);
        printf("% 6.4e    % 6.4e\n", x, y);
    }
}
```

Apuntadores a funciones (III)

```
double fnc1(double x) {
    return(2*x + 1);
}

double fnc2(double x) {
    return(sin(x));
}

int main() {
    printf("Evaluando la funcion 2x+1:\n");
    evalFnc(1, 4, 5, &fnc1);

    printf("Evaluando la funcion sin(x):\n");
    evalFnc(0, M_PI, 5, &fnc2);

    printf("Evaluando la funcion sin(x):\n");
    evalFnc(0, M_PI, 5, &sin);
    return(0);
}
```

Escritura de archivos de texto

El siguiente código muestra una función que crea un archivo de texto, cuyo nombre está dado por la cadena *cname*, para almacenar los *numData* primeros datos de dos arreglos *vx* y *vy*.

```
void generaTabla(double *vx, double *vy, int numData,
                 char *cname)
{
    int i;
    FILE *file;

    file = fopen(cname, "w");
    for(i = 0; i < numData; i++)
        fprintf(file, "% .8f   % .6e\n", vx[i], vy[i]);

    fclose(file);
}
```

Gnuplot

- Gnuplot es programa que puede generar gráficas 2D y 3D a partir de datos proporcionados en archivos o funciones que tienen expresiones analíticas. Tambien puede hacer ajuste de datos.
- Las gráficas generadas se pueden almacenar en diferentes formatos.
- Para Windows, se puede obtener el instalador `gp528-win64-mingw.exe` y el manual de usuario en el sitio:

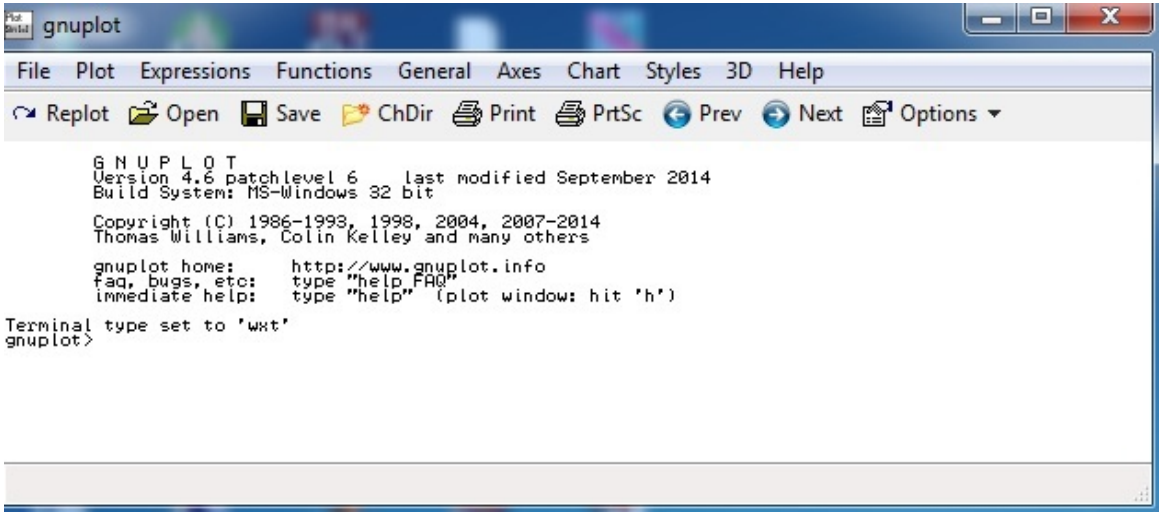
  `https://sourceforge.net/projects/gnuplot/files/gnuplot/5.2.8/`

- Para Ubuntu, se puede usar el administrador de paquetes para instalarlo, o ejecutar desde la linea de comandos:

  ```
  sudo apt-get install gnuplot
  ```

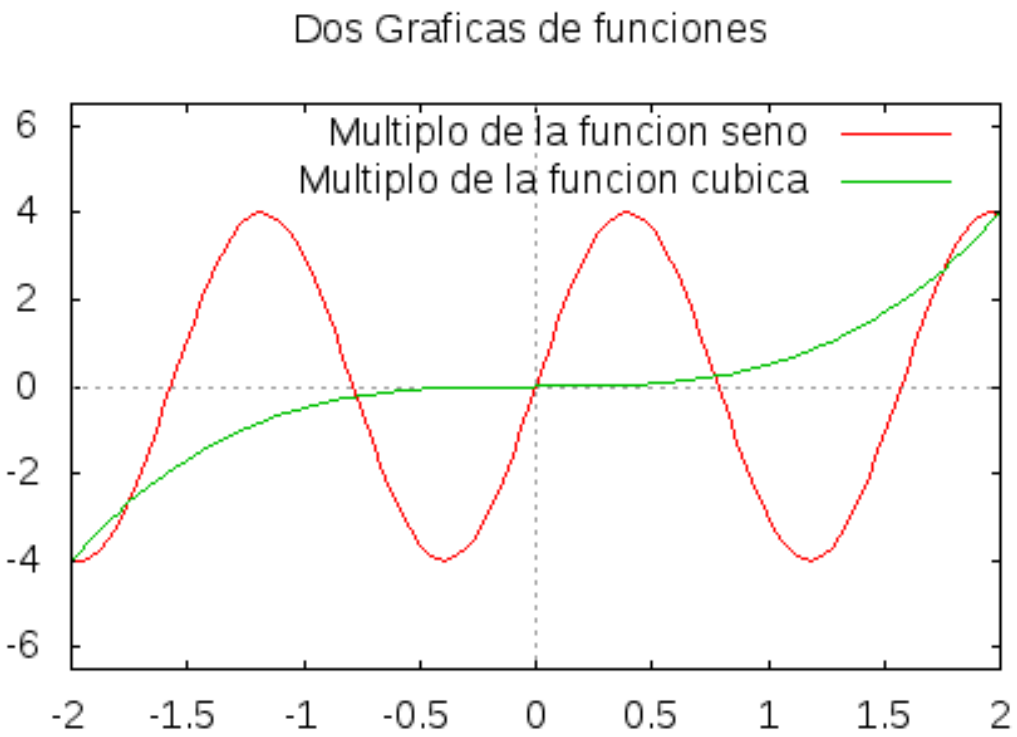
Gnuplot

- Para reproducir los ejemplos, se puede
  - Mandar a ejecutar gnuplot desde una consola y dar las instrucciones dentro del ambiente de gnuplot.
  - Si las instrucciones ya están en un archivo, digamos `prueba.gpl`, dentro del ambiente de gnuplot se puede ejecutar
    `load prueba.gpl`
    para generar la gráfica, o desde la línea de comandos se puede invocar el comando
    `gnuplot prueba.gpl`
- En Windows es mejor ejecutar la interfaz gráfica de la aplicación y escribir los comandos en ella.



Ejemplo 2D

```
set title "Dos Graficas de funciones"
set xrange [-2:2]
set yrange [-6.5:6.5]
set zeroaxis
plot  4*sin(4*x) title 'Multiplo de la funcion seno', \
      0.5*x**3  title 'Multiplo de la funcion cubica'
```



Para generar una imagen de salida

Para generar un archivo EPS con las graficas, hay que agregar las instrucciones:

```
# Genera un archivo EPS
set terminal postscript eps color  lw 2
set output 'graficas1.eps'
replot
set term x11
set output
```

Para generar un archivo PNG que tenga un tamaño de 460 pixeles de ancho y 320 pixeles de alto es:

```
# Genera un archivo PNG
set terminal png nocrop enhanced size 460,320
set output 'graficas1.png'
replot
set term x11
set output
```

Gráficas con datos tomados de un archivo (I)

Usando el código `generaTabla.c`, que genera un archivo de texto que tiene 4 columnas:

- La primera columna tiene los valores x_j de una partición uniforme $a = x_1 < x_2 < \ldots < x_m = b$ de un intervalo $[a.b]$ con m puntos. Estos datos se proporcionan desde la línea de comandos.
- Las columnas 2, 3 y 4 tienen los valores de las funciones $4\sin(4x)$, $0.5x^3$, y $\sqrt{|x|}$, evaluadas en los puntos x_j. Por ejemplo, las primeras líneas del archivo `tabla.txt`:

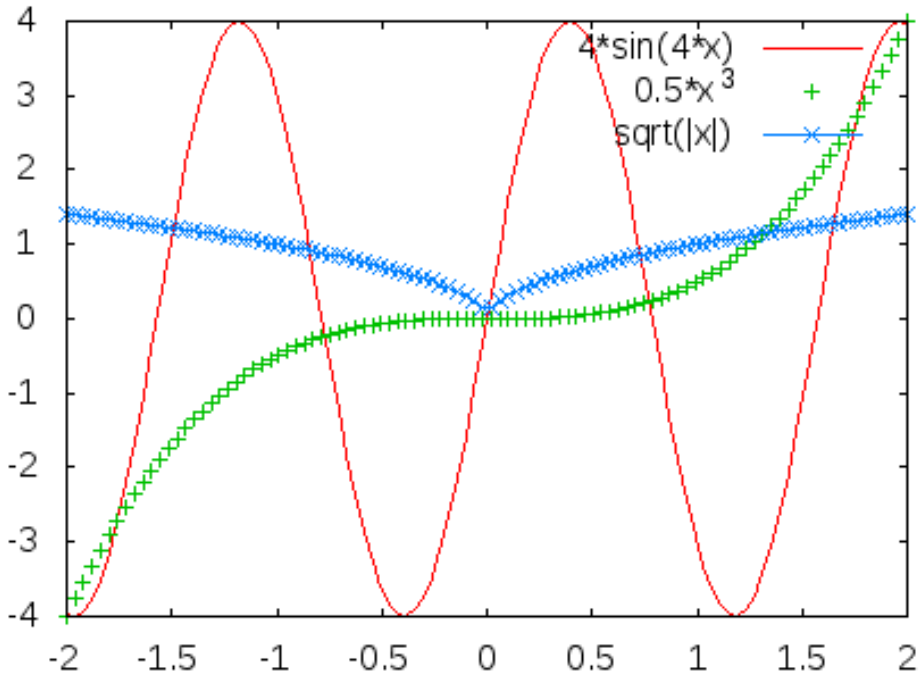
```
#              x              f1(x)              f2(x)              f3(x)
-2.00000000e+00  -3.95743299e+00  -4.00000000e+00  1.41421356e+00
-1.95959596e+00  -3.99951343e+00  -3.76244024e+00  1.39985569e+00
-1.91919192e+00  -3.93735463e+00  -3.53447752e+00  1.38534902e+00
-1.87878788e+00  -3.77257664e+00  -3.31591396e+00  1.37068883e+00
-1.83838384e+00  -3.50947407e+00  -3.10655169e+00  1.35587014e+00
-1.79797980e+00  -3.15490414e+00  -2.90619283e+00  1.34088769e+00
-1.75757576e+00  -2.71810801e+00  -2.71463951e+00  1.32573593e+00
```

Gráficas con datos tomados de un archivo (II)

El siguiente script genera una gráfica en la que aparecen las gráficas de las tres funciones utilizas el archivo de texto generado.
Hay que especificar cuáles son las columnas que se tienen que tomar para generar una gráfica.

```
plot  "tabla.txt" using 1:2 title '4*sin(4*x)' w l, \
      "tabla.txt" using 1:3 title '0.5*x^3' w p, \
      "tabla.txt" using 1:4 title 'sqrt(|x|)' w linespoints
set terminal png nocrop enhanced size 460,320
set output 'graficas2.png'
replot
set term x11
set output
```

Gráficas con datos tomados de un archivo (III)



Creando un histograma con Gnuplot (I)

Usamos el código `randomNormal.cpp` para generar números aleatorios con distribución normal:

- El programa recibe desde la línea de comandos el número de puntos que se quieren generar y la desviación estándar de la normal.
- Usa la librería GSL para generar los números.
- Los números generados se escriben en el archivo de texto `muestras.txt`.

El siguiente script de Gnuplot usa el archivo generado para crear el histograma.

```
# Numero de subintervalos
n = 40
# Valor minimo
min=-8.0
# Valor maximo
max=8.0
# Tama~no de los subintervalos
width=(max-min)/n

# Funciones para mapear un valor a los intervalos
```

Creando un histograma con Gnuplot (II)

```
bin_index(x,width) = floor(x/width)
bin_value(x,width) = width*bin_index(x,width) + width/2.0

set terminal png nocrop enhanced size 640,480
set output 'histograma.png'
set xrange [min:max]
set yrange [0:]
set xtics min,(max-min)/5,max
set boxwidth width*0.9
set style fill solid 0.5 #fillstyle
set tics out nomirror
set xlabel "x"
set ylabel "Frecuencia"
plot './muestras.txt' using (bin_value($1,width)):(1.0) smooth freq w boxe
set term x11
set output
```

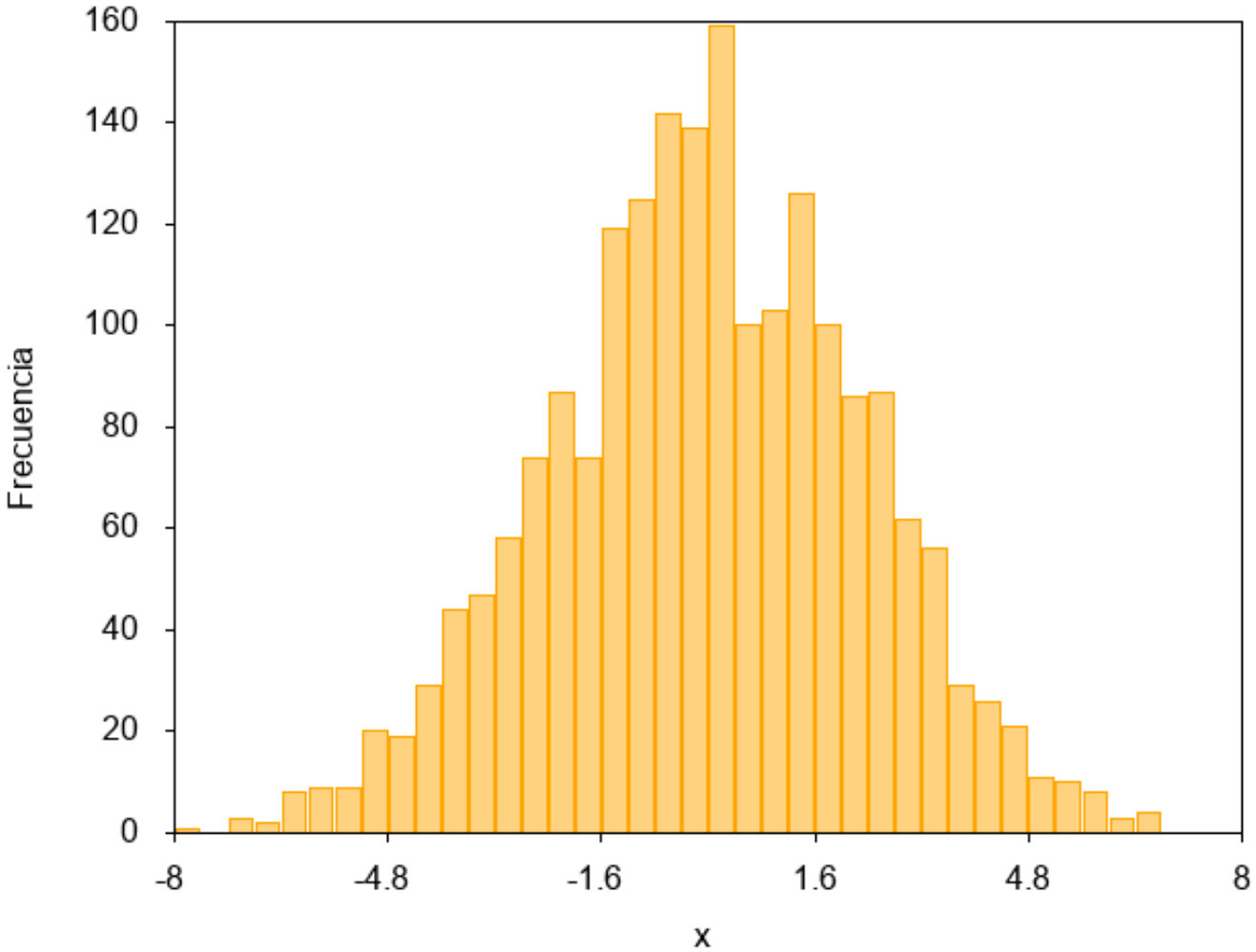
Sobre el script de Gnuplot:

- Se fija la cantidad y el ancho de los intervalos que tendrá el histograma.
- La función `bin_index(x,width)` devuelve el índice del bin al que el valor x será asignado.

Creando un histograma con Gnuplot (III)

- La función `bin_value(x,width)` devuelve el valor del bin al que el valor x es asignado.
- La opción `smooth freq` se usa para que cuente los puntos que tienen el mismo bin.
- Note que las instrucciones después de `using` van entre paréntesis. Esto le indica a Gnuplot que los interprete como expresiones numéricas, en lugar de los índices de las columnas del archivo.
  - `$1` indica el número de la columna del archivo que se va a usar.
  - `(bin_value($1,width))` indica que le pase el valor de la columna 1 a la función y use el valor que retorna como la coordenada X.
  - `(1.0)` indica que se incremente el contador (la coordenada Y) en 1.0 para cada ocurrencia x.

Creando un histograma con Gnuplot (IV)

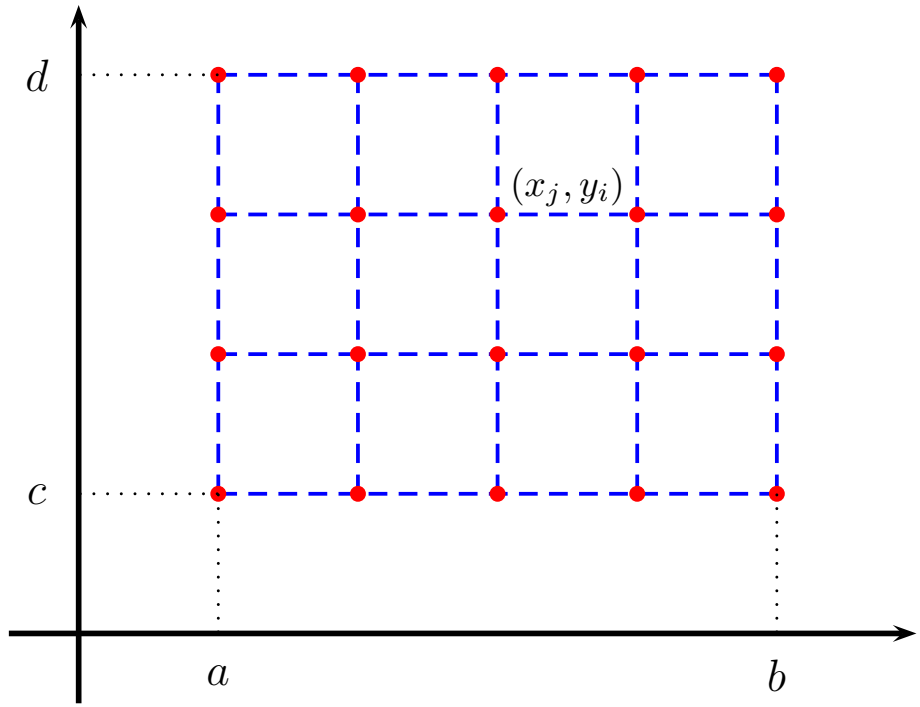


Gráficas 3D (I)

El código `generaDatosSuperficie.c` genera una tabla de valores para generar la gráfica de una función de dos variables.

- Desde la línea de comandos se proporciona el intervalo para x, el intervalo para y, el número de puntos en cada intervalo y el nombre del archivo de salida.
- El archivo de salida tiene tres columnas: una para los valores x_j, otra para los valores y_i y la tercera con los valores $f(x_j, y_i)$. Los puntos (x_j, y_i) son los nodos de una retícula regular.

Gráficas 3D (II)



- Los datos en el archivo se deben listar por bloques formados por las filas o por columnas de la retícula.
- En el archivo cada bloque de datos se separa por un línea vacía. Por ejemplo:

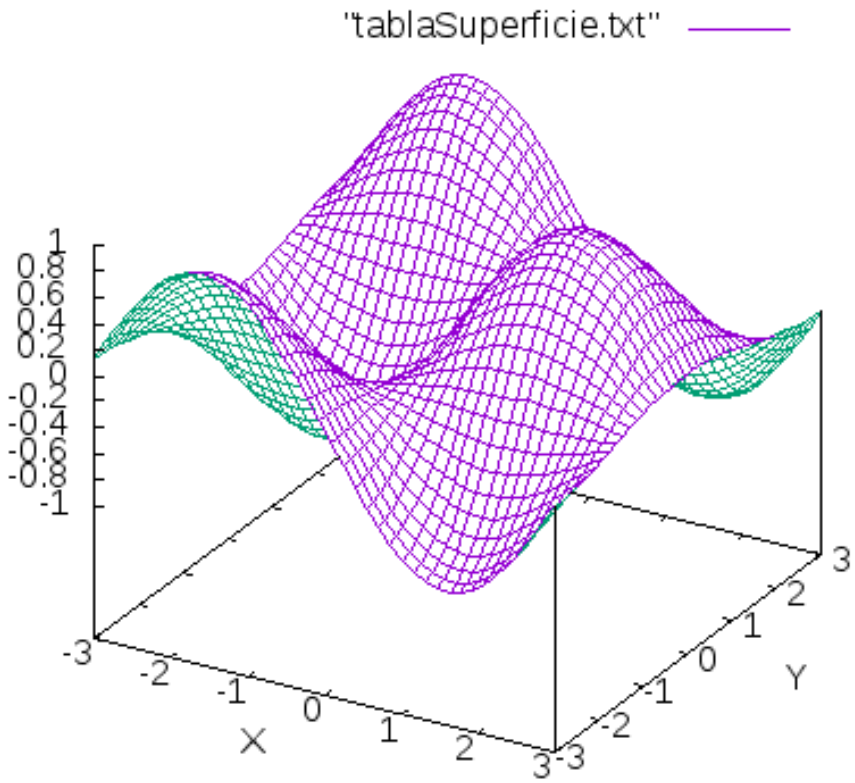
Gráficas 3D (III)

```
#              x                   y                   z
-2.00000000e+00   -3.00000000e+00    9.00197630e-01
-2.00000000e+00    0.00000000e+00   -9.09297427e-01
-2.00000000e+00    3.00000000e+00    9.00197630e-01

-6.66666667e-01   -3.00000000e+00    6.12181465e-01
-6.66666667e-01    0.00000000e+00   -6.18369803e-01
-6.66666667e-01    3.00000000e+00    6.12181465e-01

 6.66666667e-01   -3.00000000e+00   -6.12181465e-01
 6.66666667e-01    0.00000000e+00    6.18369803e-01
 6.66666667e-01    3.00000000e+00   -6.12181465e-01

 2.00000000e+00   -3.00000000e+00   -9.00197630e-01
 2.00000000e+00    0.00000000e+00    9.09297427e-01
 2.00000000e+00    3.00000000e+00   -9.00197630e-01
```

- Con el archivo de texto generado se puede visualizar la malla, o la superficie de la función, o su proyección al plano.

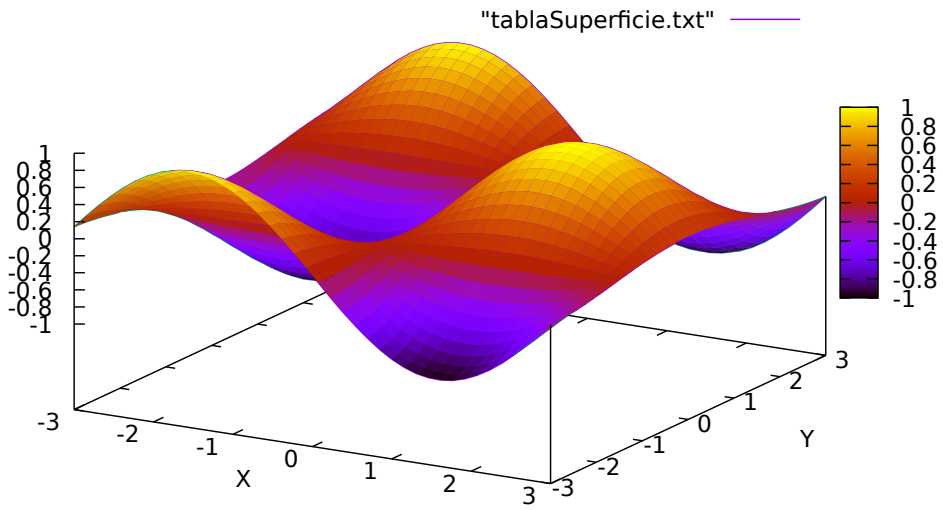
Gráfica 3D: Malla

```
set xlabel ’X’
set ylabel ’Y’
set hidden3d
splot "tablaSuperficie.txt" with lines
```



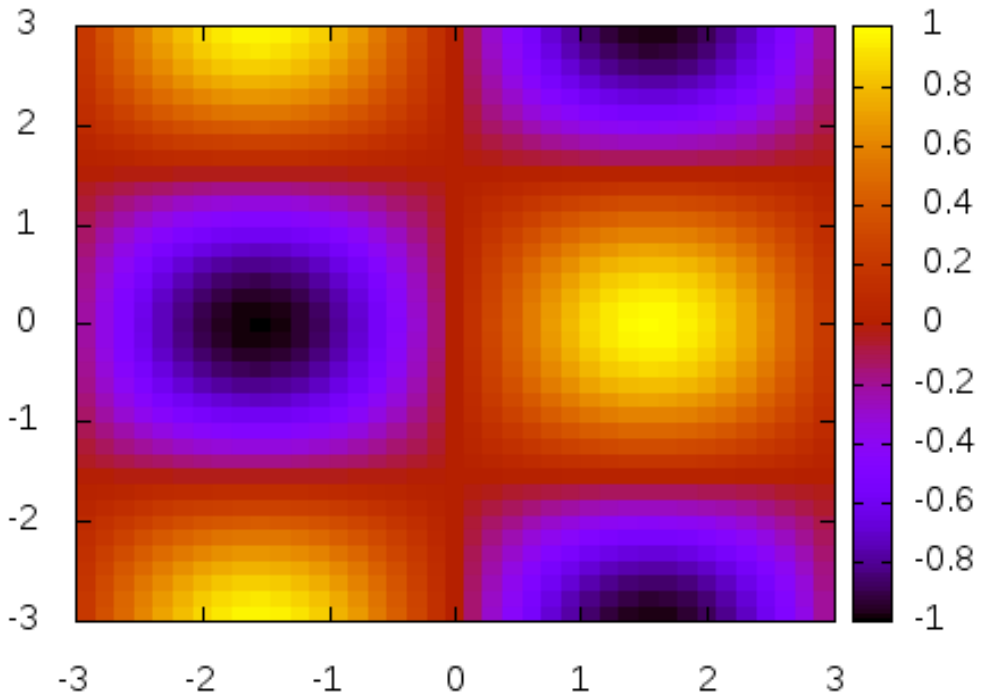
Gráfica 3D: Superficie

```
set xlabel ’X’
set ylabel ’Y’
set hidden3d
splot "tablaSuperficie.txt" with pm3d
```



Gráfica 3D: Proyección 2D

```
set terminal png nocrop enhanced size 460,400
set output 'grafica4.png'
set pm3d map
splot "tablaSuperficie.txt"
set term x11
set output
```



Animaciones (I)

Se puede crear un GIF animado a partir de varias imágenes generadas con Gnuplot.
Por ejemplo, usando el código `generaScriptAnimacion1.cpp`, que hace lo siguiente:

- Crea el archivo de texto `anim01.gpl` que tiene las instrucciones para crear cada cuadro del GIF animado.
- Cada cuadro es creado por una instrucción `plot`. En este caso cada instrucción lee un archivo que tiene una tabla con coordenadas de puntos sobre la curva

$$\begin{aligned} x &= \cos t - \cos 60t \\ y &= 2\sin t - \sin 60t \end{aligned}$$

- Los archivos con las coordenadas son generados por el programa en C

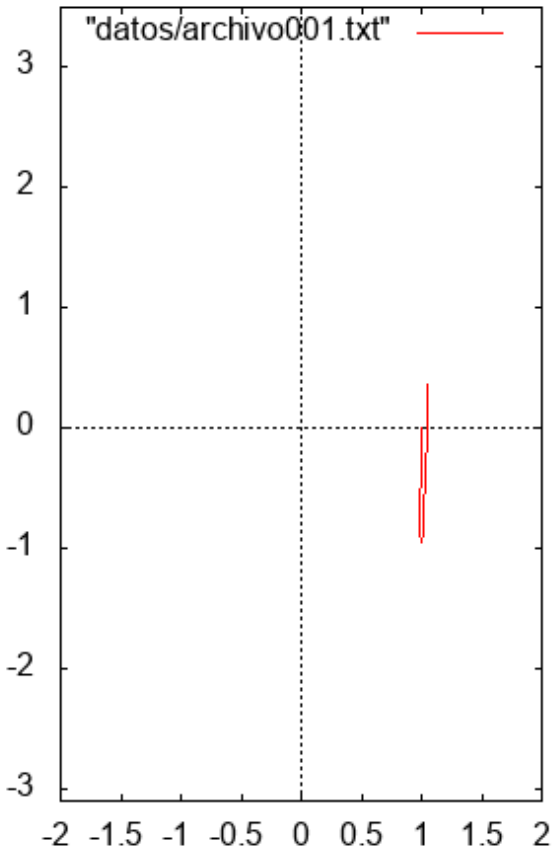
Una parte del script generado `anim01.gpl` se muestra a continuación:

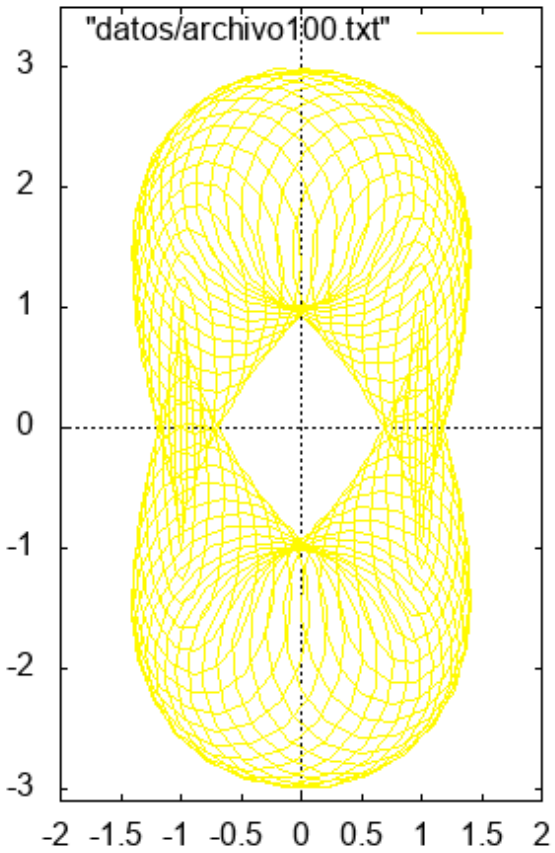
Animaciones (II)

```
set terminal gif animate delay 25
set size ratio -1
set output "curvaParametrica1.gif"
set xrange [-2.0:2.0]
set yrange [-3.1:3.5]
set xzeroaxis
set yzeroaxis

plot 'datos/archivo001.txt' w l lw 1 lc rgb '#FF0300'
plot 'datos/archivo002.txt' w l lw 1 lc rgb '#FF0500'
plot 'datos/archivo003.txt' w l lw 1 lc rgb '#FF0800'
plot 'datos/archivo004.txt' w l lw 1 lc rgb '#FF0A00'
```

Las gráficas del primer y ultimo cuadro se muestran a continuación.

Animaciones (III)





Animaciones (IV)

El ejemplo anterior muestra que se puede generar un cuadro leyendo la información de un archivo de texto, pero en este caso el último archivo generado tiene la información de todos los anteriores.

Queremos usar sólo ese archivo. El código `generaScriptAnimacion2.cpp` modifica las intrucciones para leer solo unas filas del archivo generado con las coordenadas de la curva paramétrica. Una parte del script que genera:

```
set terminal gif animate delay 25
set size ratio -1
set output "curvaParametrica2.gif"
set xrange [-2.0:2.0]
set yrange [-3.1:3.5]
set xzeroaxis
set yzeroaxis

plot 'datos/archivo100.txt' every ::1::10 w l lw 1 lc rgb '#FF0300'
plot 'datos/archivo100.txt' every ::1::20 w l lw 1 lc rgb '#FF0500'
plot 'datos/archivo100.txt' every ::1::30 w l lw 1 lc rgb '#FF0800'
```