Práctica 3

1) Sistemas de ecuaciones no lineales.

Utilizar el método de Newton-Raphson para determinar la solución del sistema: $\left.\begin{array}{l} x\,y - x = 0 \\ x + y - 2 = 0 \end{array}\right\}$

a) Partiendo de $\vec{x}_0 = \begin{pmatrix} 0.5 \\ 0.7 \end{pmatrix}$. Utilizar como criterio de parada que la distancia de la solución aproximada obtenida en la *k*-ésima iteración $\vec{x}_k$ a la verdadera solución $\vec{s}$, es decir $\|\vec{x}_k - \vec{s}\|_2$, sea menor que 0.001, sabiendo que la sucesión converge a $\vec{s} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

b) ¿Qué ocurre si $\vec{x}_0 = \begin{pmatrix} 0.25 \\ 1.5 \end{pmatrix}$?.

(Solución: a) hacen falta 4 iteraciones; b) converge a otra solución del problema $\vec{s} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$)

2) Fractal de Julia

Representar el fractal de Julia para $c = 1 + 0.25i$ con la función $g(z) = c * \text{sen}(z)$ dibujado en la región del plano complejo $[-1.5, 1.5] \times [-1.5, 1.5]$ (dividiendo en 333 partes iguales) con un número máximo de iteraciones igual a 256, con $\text{mod} = 13$. Se van efectuando las iteraciones de punto fijo:

- Si en la *k*-ésima iteración se obtiene que $|z_k| > \text{mod}$, entonces se devuelve el valor de k y se deja de iterar.
- Si se completan las 256 iteraciones y en todo momento $|z_k| \leq \text{mod}$ se devuelve un 0.

Nota: En C la función $\sin(x)$ se usa para la función seno con valores reales. Cuando z es un número complejo se debe usar la función $c\sin(z)$.

Por otro lado, usar el script de Gnuplot:

```
set pm3d map
set size square
splot "datos.dat"
```

3) Fractal de Newton-Raphson

Se considera la ecuación: $z^5 + 1 = 0$ con z complejo.

Con lápiz y papel deducir cuáles son las 5 raíces complejas de la ecuación anterior.

Una vez calculadas, se tiene que representar el fractal de Newton-Raphson para la ecuación anterior en la región del plano complejo: $[-1,\ 1] \times [-1,\ 1]$. Para ello se divide en 333 partes iguales y para cada

punto z_0 de esta región se va a calcular la sucesión de N-R con un número máximo de iteraciones igual a 30 y con $tol = 0.01$. Si efectuadas 30 iteraciones resulta que se está cerca de la raíz primera se devuelve un 1, si se está cerca de la segunda raíz se devuelve un 2, etc., en caso contrario (no está cerca de ninguna de las cinco raíces) se devuelve un 6.

Decir que un número z "esta cerca de una raíz" s, se debe interpretar de la siguiente forma:

$|creal(z) - creal(s)| < tol$ y $|cimag(z) - cimag(s)| < tol$

Usar el mismo script de Gnuplot del programa anterior.

(Sol: una raíz de la ecuación es -1, otra es $-0.309017 - 0.951057\,i$, etc.)

4) Sistema de ecuaciones lineales tridiagonal.

Se trata de hacer un programa que sirva para resolver cualquier sistema tridiagonal. Para evitar la introducción de datos desde el teclado, se escribirá la matriz *A* de los coeficientes del sistema y *B* los términos independientes al comienzo del programa, pero el código debe ser general, de manera que sólo modificando *A* y *B* debe seguir funcionando correctamente. Resolver el sistema tridiagonal siguiente:

$$\left.\begin{aligned} 4x_1 + x_2 &= 9 \\ 2x_1 + 3x_2 + x_3 &= 7 \\ x_2 + x_3 + x_4 &= 0 \\ 2x_3 + x_4 + 2x_5 &= 1 \\ 3x_4 + x_5 &= -2 \end{aligned}\right\}$$

La solución que debe obtenerse es : $x_1 = 2 \quad x_2 = 1 \quad x_3 = 0 \quad x_4 = -1 \quad x_5 = 1$

5) Sistema tridiagonal.

Modificar el programa anterior para resolver un sistema tridiagonal de 100 ecuaciones con 100 incógnitas del siguiente tipo:

$$\begin{pmatrix} 4 & 2 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 4 & 2 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 4 & 2 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 2 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & & & & & & & & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 & 4 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{98} \\ x_{99} \\ x_{100} \end{pmatrix} = \begin{pmatrix} 1^2 \,/\, 10 \\ 2^2 \,/\, 10 \\ 3^2 \,/\, 10 \\ 4^2 \,/\, 10 \\ \vdots \\ 98^2 \,/\, 10 \\ 99^2 \,/\, 10 \\ 100^2 \,/\, 10 \end{pmatrix}$$

(Solución: por ejemplo: $x_{25} = 8.82099125\,;\quad x_{100} = 227.564529$).

6) Métodos iterativos.

Se considera el sistema lineal: $\left.\begin{array}{l} 4x_1 + x_2 + x_3 = 9 \\ x_1 + 3x_2 + x_3 = 10 \\ x_1 + x_2 + 5x_3 = 18 \end{array}\right\}$, cuya solución exacta es:

$x_1 = 1 \quad x_2 = 2 \quad x_3 = 3$.

a) Comprobar que hacen falta 23 iteraciones del método de Jacobi para conseguir que $\left\|\vec{x}^{(n)} - \vec{s}\right\|_2$ sea menor que 10^{-6}, siendo $\vec{s}$ = vector solución = $(1,2,3)$ y partiendo de $\vec{x}^{(0)} = (0,0,0)$.

b) Comprobar que sólo hacen falta 9 iteraciones del método de Gauss-Seidel para conseguir el mismo objetivo.

Se trata de ver a continuación, que tanto el método de Jacobi como Gauss-Seidel son convergentes siempre, independientemente del valor elegido para $\vec{x}^{(0)}$, al tener la matriz de los coeficientes del sistema la diagonal estrictamente dominante:

c) Comprueba que también converge a la solución cuando $\vec{x}^{(0)} = (-527,1000,-80000)$ (muy alejado de la solución) y que hacen falta 38 iteraciones del método de Jacobi para que conseguir que $\left\|\vec{x}^{(n)} - \vec{s}\right\|_2$ sea menor que 10^{-6}.

d) Comprueba que el método de Gauss-Seidel necesitaría 13 iteraciones para conseguir el mismo objetivo.

7) Sistema tridiagonal grande resuelto con un método iterativo.

El sistema tridiagonal de 100 ecuaciones con 100 incógnitas del ejercicio 2 se desea resolver ahora mediante un método iterativo. Efectuar 50 iteraciones del método de Gauss-Seidel partiendo de $\vec{x}^{(0)} = \vec{0}$.

(Solución: igual que la obtenida en el ejercicio 2, $x_{25}^{(50)} = 8.82099125$; $x_{100}^{(50)} = 227.564529$).