Métodos Numéricos Avanzados
PRÁCTICAS

Tema 1. RESOLUCIÓN NUMÉRICA DE SISTEMAS NO LINEALES

Cuarto Curso de Grado en Física
(Primer cuatrimestre)

PROF.: JOSÉ ANTONIO SÁNCHEZ PELEGRÍN

Sistemas no lineales (I)

Curso 2023-2024

Universidad de Córdoba
Dpto. Informática y Análisis Numérico

Objetivo y ejecución

- En esta práctica se programarán el método de punto fijo y el método de Newton, con algunas de sus variantes, para la resolución de sistemas de ecuaciones no lineales.
- Cada uno de los ficheros creados durante la práctica se subirá a la página Moodle de la asignatura al enlace que se habilitará, más adelante, para ello.

Práctica 1.1. Escribe una función de MATLAB, llamada `sisnolin_puntofijo`, que resuelva un sistema no lineal algebraico por el método de punto fijo, en sus dos versiones de Jacobi y de Gauss-Seidel. El encabezamiento será:

```

function [x,nit,norma,salida]=...
             sisnolin_puntofijo(fung,x0,metodo,maxiter,toler,detalle)
%---------------------------------------------------------------------
%% Aproxima el punto fijo de una función vectorial fung(x)
%  usando el método de punto fijo (opción Jacobi o Gauss-Seidel)
%  Datos
%  -----
% fung      = función vectorial para buscarle el punto fijo
% x0        = vector inicial
% metodo          = 1  (Jacobi)
%                 = 2  (Gauss-Seidel)
% maxiter   = numero máximo de iteraciones admitidas
% toler       = tolerancia de parada
% detalle      = 0 No se muestran las iteraciones por pantalla
%              = 1 Se muestran las iteraciones por pantalla
%--------------------
% Se ha de tener la función:
% fung.m función y=fung(x) donde se define la función
%                               de punto fijo
%---------------------------------------------------------------------
% Argumentos de salida:
%---------------------------------------------------------------------
% x       = punto fijo aproximado
% nit     = iteraciones realizadas hasta alcanzar la tolerancia
% norma   = norma de la diferencia de las dos últimas iteraciones
% salida    = 1 si se alcanza la tolerancia
%           = 0 si no se alcanza la tolerancia en las iteraciones dadas.
%---------------------------------------------------------------------
```

Práctica 1.2. Escribe un programa principal que se llame `resolucion_puntofijo.m` y use la función anterior para aproximar la solución del sistema no lineal

$$\left\{\begin{array}{lcr} 15x_1 + x_2^2 - 4x_3 & = & 13, \\ x_1^2 + 10x_2 - x_3 & = & 11, \\ x_2^2 - 25x_3 & = & -22. \end{array}\right.$$

Toma como punto inicial $\mathbf{x}^0 = (0,0,0)$ y como tolerancia $\varepsilon = 10^{-6}$.

Práctica 1.3. Implementa el algoritmo de Newton-Raphson escribiendo una función que tendrá el siguiente encabezado

```
function [x,nit,norma,conver]=...
                 newtonsis(fun,jacfun,x0,maxiter,toler,detalle)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Método de Newton para sistemas de ecuaciones
%-------------------------------------------------------------------
% Argumentos de entrada
%-------------------------------------------------------------------
% fun         = función vectorial que define el sistema (f(x)=0)
% jacfun      = jacobiano de la función f
% x0          = vector inicial
% maxiter     = numero maximo de iteraciones admitidas
% toler       = tolerancia de parada
% detalle     = 0 No se muestran las iteraciones
%             = 1 Se muestran las iteraciones
%--------------------
% Se han de tener las funciones:
% fun.m  donde se define el sistema
% jacfun.m  donde se define la matriz jacobiana del sistema.
%
%--------------------------------------------------------------------
% Argumentos de salida:
%--------------------------------------------------------------------
% x      = solución aproximada
% nit    = iteraciones realizadas hasta alcanzar la tolerancia
% norma  = norma de la diferencia de las dos últimas iteraciones
% conver = 1 si se alcanza la tolerancia
%        = 0 si no se alcanza la tolerancia en las iteraciones dadas.
%--------------------------------------------------------------------
```

Práctica 1.4. Queremos encontrar los puntos de intersección de las cónicas de ecuaciones

$$\begin{aligned} x^2 - y - 1 &= 0, \\ (x-2)^2 + (y-0.5)^2 - 1 &= 0. \end{aligned}$$

- Dibuja las cónicas usando la función `contour` de MATLAB como se hizo en teoría.
- Plantea explícitamente las funciones f_k para el sistema de ecuaciones que expresa la intersección de las cónicas y halla la matriz Jacobiana.
- Utilizan el método de Newton-Raphson, que has implementado en el ejercicio anterior, para calcular los puntos de intersección de las cónicas. Utiliza diferentes valores iniciales para cada uno de ellos.