Prácticas de Matlab
Resolución de EDO con métodos implícitos
Hoja 5 B

1.1 Práctica 1 {Ejemplo resolver un sistema mendiante un punto fijo}

Escribid en el apéndice A1 una función que implemente el método de punto fijo para sistemas.

$$\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}_{k+1} = G \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}_{k} \tag{1}$$

Resolved el siguiente ejemplo

$$\begin{array}{rcll} f_1(x,y) & = & x^2 - 2x - y + 0.5 & \text{parábola} \\ f_2(x,y) & = & x^2 + 4y^2 - 4 & \text{elipse} \end{array} \tag{2}$$

Es decir, $f_1(x,y) = 0$ y $f_2(x,y) = 0$ Usando

$$\begin{array}{rcl} x & = & \frac{x^2 - y + 0.5}{2} \\ y & = & \frac{-x^2 - 4y^2 + 8y + 4}{8} \end{array} \tag{3}$$

Para el punto inicial $(p_0, q_0) = (0, 1)$ la sucesión converge a $(-0.2, 1)$ **PERO** para el punto inicial $(p_0, q_0) = (2, 0)$ la sucesión diverge.

1.1.1 Solución

$$\begin{aligned} x &= (-0.222315, 0.993812,) \\ ev &= 4 \\ loop &= 4 \\ x_{vect} &= \begin{pmatrix} 0.000000, 1.000000 \\ -0.250000, 1.000000 \\ -0.218750, 0.992188 \\ -0.222168, 0.993988 \\ -0.222315, 0.993812 \end{pmatrix} \end{aligned}$$

1.2 Práctica 2 (Ejemplo: Una iteración tipo Newton para sistemas)

El método de Newton para sistemas esta dado por

$$\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}_{k+1} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}_{k} - J_k^{-1} \begin{pmatrix} f_1(x_1, ..., x_n) \\ \vdots \\ f_n(x_1, ..., x_n) \end{pmatrix}_{k} \qquad J = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix} \tag{4}$$

aplicalo para el siguiente sistemaaplicalo para el siguiente sistema

$$\left.\begin{array}{rcl} e^x + xy - 1 & = & 0 \\ \operatorname{sen}(xy) + x + y - 1 & = & 0 \end{array}\right\} \tag{5}$$

partiendo del punto $(x_0, y_0) = (0.5, 0.5)$. **Importante:** el jacobiano puedes calcularlo con el comando simbólico, pero no uses variables *simbólicas* en el bucle. ¡ Tampoco uses la inversa del jacobiano! El comando **inv(J)** está **prohibido**. Transforma $J^{-1}F$ en un sistema lineal equivalente y aplica un comandos intrínseco de Matlab para resolver dicho sistema.

1.2.1 Solución

$$
\begin{aligned}
x &= (0.0; 1.0) \\
ev &= 6 \\
loop &= 3 \\
x_{vect} &= \begin{pmatrix} 0.500000, 0.500000 \\ 0.005441, 0.827896 \\ -0.000503, 0.999982 \\ 0.000000, 1.000000 \end{pmatrix}
\end{aligned}
$$