Lecture CFD

Solution of Nonlinear equations

Autor: GABusta

Solve the next Nonlinear problem,

$$\underline{F}(\underline{U}) - \underline{R} = 0$$

Where,

$$\underline{F}(\underline{U}) = \begin{bmatrix} {U_1}^2 + {U_2}^2 - 5.\left[sen^2\left(U_3.\frac{\pi}{10}\right)\right] \\ {U_1}^2 + {U_2}^2 - e^{\frac{U_3}{5}} \\ U_1 + {U_2}^2 - \frac{1}{9}.(U_3 - 10)^2 \end{bmatrix} \qquad \underline{U} = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} \qquad \underline{R} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} \qquad \underline{U^0} = \begin{bmatrix} 1 \\ 1 \\ 7 \end{bmatrix}$$

The tolerance for all the study cases is $\boldsymbol{tol} = \mathbf{1.10^{-10}}$

We need to solve the problem using three diferent methods of solutions (and compare each other):

- Newton – Raphson method
- Modified Newton – Raphson method
- BFGS (quasi Newton) method

Important:

The next problem corresponds to a one practical job during a phd course, its programming entirely in Python.

Feel free to make any sugestions, questions or comments that you need.

---

1. NEWTON – RAPHSON method:

Problema estacionario

$$\underline{\underline{K}} \cdot \underline{U} = \underline{R} \longrightarrow \underline{F}(\underline{U}) = \underline{R} \longrightarrow \underline{R} - \underline{F}(\underline{U}) = \underline{0}$$

Linealizando

$$\underline{F} = \underline{F}^{(k-1)} + \left.\frac{\partial \underline{F}}{\partial U_j}\right|^{(k-1)} \Delta U_j{}^{(k)}$$

k es el número de iteración

Suma NEQ términos

$$\Delta U_j{}^{(k)} = U_j{}^{(k)} - U_j{}^{(k-1)}$$

Siendo el procedimiento para la resolución el siguiente,

- Se calcula la matriz tangente $\underline{\underline{K_T}}(\underline{U}) = \frac{\partial \underline{F}(\underline{U})}{\partial U_j}$

$$\underline{\underline{K_T}}(\underline{U}) = \left[\frac{\partial \underline{F}(\underline{U})}{\partial U_1} \quad \frac{\partial \underline{F}(\underline{U})}{\partial U_2} \quad \frac{\partial \underline{F}(\underline{U})}{\partial U_3}\right]$$

- Se evalúan

$$\underline{F}^{(k-1)} \qquad y \qquad \underline{\underline{K_T}}(\underline{U})^{(k-1)}$$

- Y se calcula el diferencial,

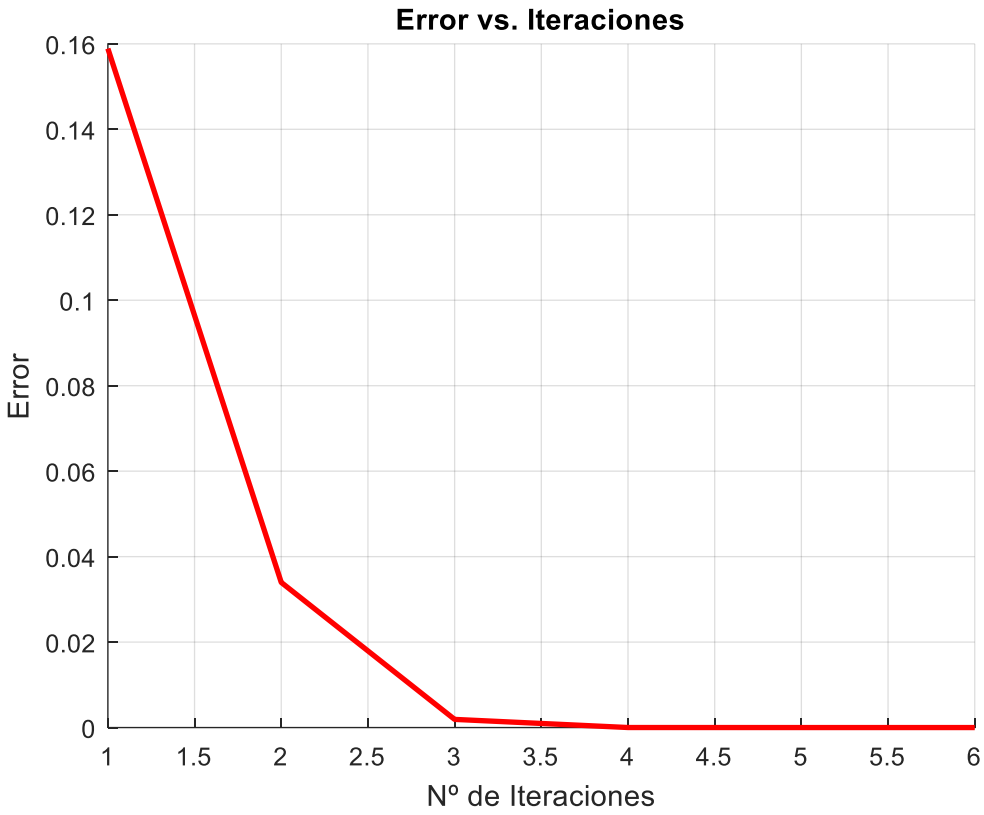
$$\Delta U_j{}^{(k)} = \left(\underline{\underline{K_T}}(\underline{U})^{(k-1)}\right)^{-1} \cdot \left(\underline{R} - \underline{F}^{(k-1)}\right)$$

- Conociendo además,

$$U_j{}^{k} = \Delta U_j{}^{(k)} + U_j{}^{(k-1)}$$

Los Resultados Obtenidos son los siguientes,

Iteraciones "k"	U1	U2	U3	Error	Tiempo (seg.)
6	1.797	1.244	6.646	5.988 e-20	0.3525



Se puede observar que la solución converge en 6 iteraciones

2. Método NEWTON – RAPHSON (modificado):

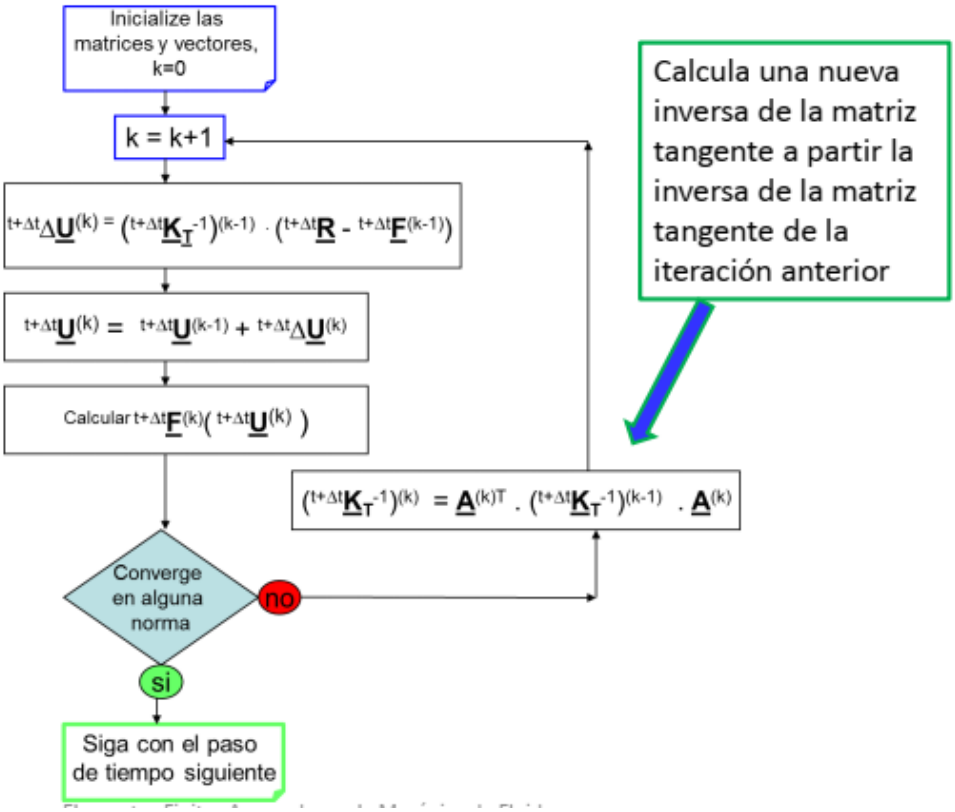
$$ {}^{t+\Delta t}\underline{\underline{\mathbf{K}}}_T{}^{(k-1)} \cdot {}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)} = {}^{t+\Delta t}\underline{\mathbf{R}} - {}^{t+\Delta t}\underline{\mathbf{F}}^{(k-1)} $$

Mantengo f' o la matriz tangente ${}^{t+\Delta t}\underline{\underline{\mathbf{K}}}_T$ **constante** durante las iteraciones o los modifico cada n iteraciones

Los Resultados Obtenidos son los siguientes,

N_KT (actualización)	Iteraciones realizadas	U1	U2	U3	Error	Tiempo (seg.)
2	7	1.797	1.244	6.646	4.52e-17	0.1868
4	9	1.797	1.244	6.646	5.766e-16	0.1838
8	11	1.797	1.244	6.646	2.324e-11	0.1799
16	17	1.797	1.244	6.646	6.843e-11	0.2136
32	33	1.797	1.244	6.646	4.522e-17	0.2742

3. Método BFGS (Cuasi Newton):



El algoritmo para el cálculo mediante el método BFGS, es el mostrado en la gráfica superior, donde:

$$\underline{\underline{\mathbf{A}}}^{(k)} = \underline{\underline{\mathbf{I}}} + \underline{\mathbf{v}}^{(k)} \ \underline{\mathbf{w}}^{(k)^T}$$

$$\underline{\mathbf{v}}^{(k)} = -c^{(k)} \ {}^{t+\Delta t}\underline{\underline{\mathbf{K}}}_T^{(k-1)} \ . \ {}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)} - {}^{t+\Delta t}\Delta\underline{\mathbf{F}}^{(k)}$$

$$c^{(k)} = \left[\frac{{}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)^T} \ . \ {}^{t+\Delta t}\Delta\underline{\mathbf{F}}^{(k)}}{{}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)^T} \ . \ {}^{t+\Delta t}\underline{\underline{\mathbf{K}}}_T^{(k-1)} \ . \ {}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)}} \right]^{1/2}$$

$$\underline{\mathbf{w}}^{(k)} = \frac{{}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)}}{{}^{t+\Delta t}\Delta\underline{\mathbf{U}}^{(k)^T} \ . \ {}^{t+\Delta t}\Delta\underline{\mathbf{F}}^{(k)}}$$

Recordando que para evitar el mal condicionamiento de la matriz actualizada, se calcula el número de condicionamiento de la misma y solo se actualiza si,

$$c^{(k)} > 10^5$$

Los Resultados Obtenidos son los siguientes,

Iteración	U1	U2	U3	Error	Tiempo (seg.)
15	1.797	1.244	6.646	8.118e-11	**0.3710**

Se puede observar que se necesitan de 15 iteraciones para lograr la convergencia de la solución,

4. Comparación entre métodos:

A continuación se listan los resultados obtenidos mediante los distintos métodos de solución,

Método		K	U1	U2	U3	Error	Tiempo (seg.)
Newton-Raphson		**6**	1.797	1.244	6.646	5.988 e-20	**0.3525**
Newton-Raphson (Modificado)	N= 2	7	1.797	1.244	6.646	4.52e-17	0.1868
	N= 4	9	1.797	1.244	6.646	5.766e-16	0.1838
	N= 8	11	1.797	1.244	6.646	2.324e-11	0.1799
	N= 16	17	1.797	1.244	6.646	6.843e-11	0.2136
	N= 32	33	1.797	1.244	6.646	4.522e-17	0.2742
BFGS		**15**	1.797	1.244	6.646	8.118e-11	**0.3710**

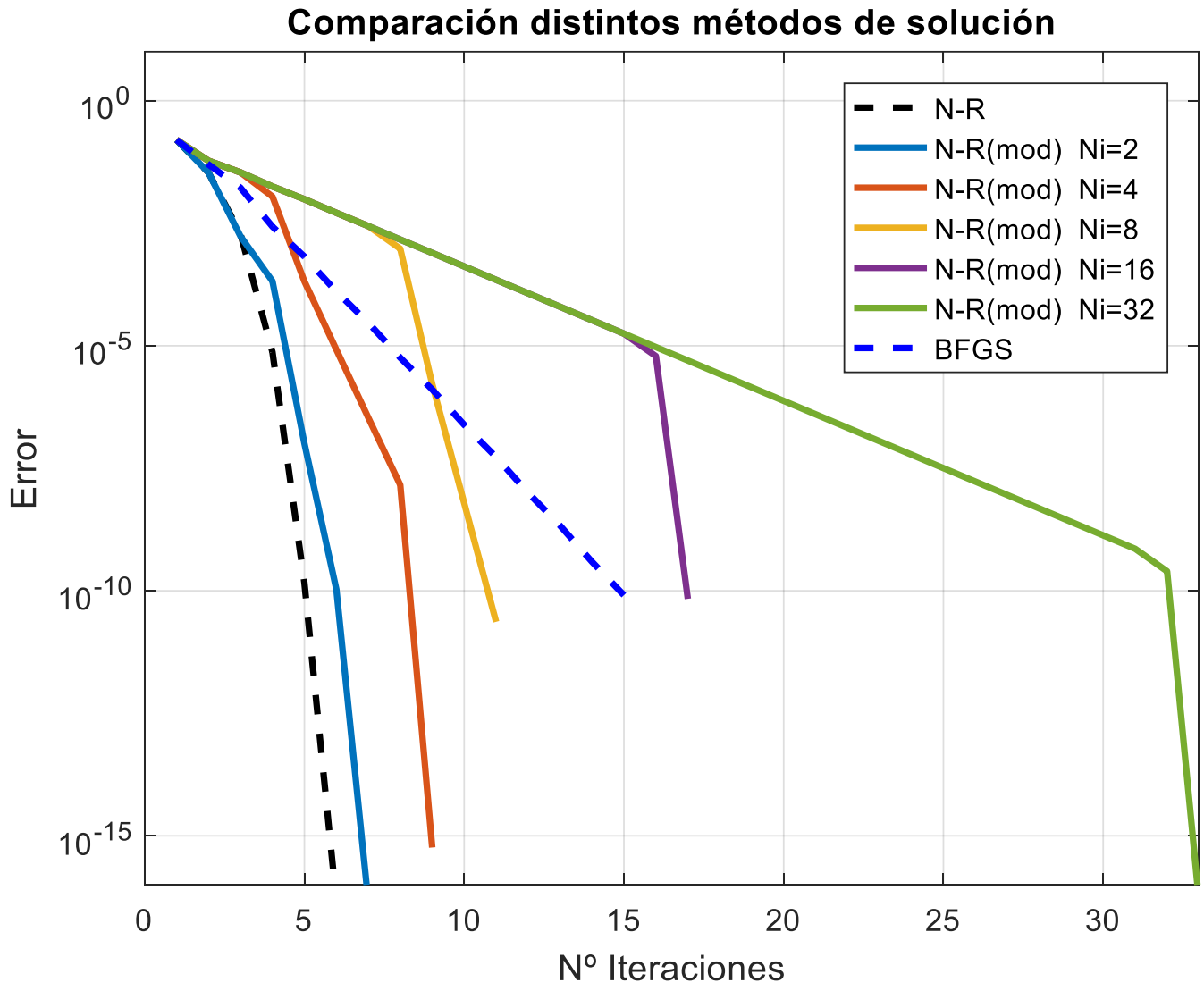
La comparación de los distintos métodos permite arribar a las siguientes conclusiones:

- **El método de Newton – Raphson** requiere de menos iteraciones para lograr la convergencia, obteniéndose (para este problema) un tiempo de resolución más grande que con el resto de los métodos. En todas las iteraciones se debió calcular la matriz tangente " KT " inversa.

- **El método de Newton – Raphson (Modificado)**, calcula la nueva matriz tangente inversa en la cantidad de intervalos detallados (ver tabla superior), obteniéndose tiempos aceptables de cálculo y distinta cantidad de iteraciones necesarias para lograr la convergencia de la solución. Como detalle, no se tiene que calcular en cada iteración la matriz tangente inversa, solamente en los intervalos deseados. Se ve reflejado en los tiempos de cálculo, ya que fueron los menores obtenidos.

- **El método BFGS**, si bien requiere de más iteraciones para lograr la convergencia y tiene un tiempo de cálculo mayor (para este problema), tiene como ventaja que sólo calcula la matriz tangente inversa una sola vez y luego realiza aproximaciones sucesivas a ésta en cada nueva iteración.

Colocando los valores obtenidos del Error en una escala Semi-Logarítmica, Se pueden ver en una misma gráfica los errores obtenidos y las iteraciones necesarias para lograr la convergencia.



5. Distintos métodos variando Uo

Se puede observar en la siguiente gráfica, el error obtenido por los distintos métodos cuando el vector Inicial está formado por

$$U_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

En todos los casos la respuesta es oscilatoria, NO logrando la convergencia de la solución a la tolerancia especificada.