Certamen Online Práctico - 3.5 - INF-285/ILI-285, v1.0
SCT - Lu.03.08.20

<u>**Instrucciones**</u>: ***Usted tiene que mostrar todo su trabajo de forma clara y ordenada para obtener todos los puntos***. *Este certamen consta de 3 preguntas, las cuales serán entregadas todas juntas en un PDF. Sus desarrollos y código sebe ser subido a la plataforma Aula en los tiempos indicados. Puntos parciales serán entregados a preguntas incompletas. Respuestas finales sin desarrollo o* ***sin nombre*** *reciben 0 puntos.* Copy-and-Paste *de algoritmos reciben 0 puntos. ¡Éxito!*

1. Considere la siguiente función:

$$f(x) = \exp(x) - x^x,$$

la cual tiene una raíz en $x = \exp(1)$, el número de Euler, o simplemente e. Al evaluar $f(e)$ obtenemos $\exp(e) - \exp(e) = 0$. Note que usaremos exp, $\exp(1)$ y e para denotar a la constante de Euler, que es aproximadamente:

$$\exp(1) = 2.7182818284590452353602874713526624977572470936999595749669676277240 7\ldots.$$

Claramente podemos notar que $\exp(x) \geq x^x$ para $x \in]0, e]$, y $x^x \geq \exp(x)$ para $x \in [e, \infty[$. Ahora, queremos estudiar como se modifica la raíz de $f(x)$ cuando le agregamos una constante δ, es decir, queremos cuantificar numéricamente que le ocurre a la raíz de la siguiente función:

$$f_\delta(x) = f(x) + \delta = \exp(x) - x^x + \delta. \tag{1}$$

Donde denotamos la raíz de $f_\delta(x)$ como $r(\delta)$. En particular conocemos que $r(0) = e$, i.e. la raíz de la función original. En cambio cuando δ es distinto a 0, su raíz cambia. Por simplicidad considere que $\delta > 0$.

(a) ✍ Proponga un algoritmo, que tenga **convergencia lineal**, para obtener la raíz de $f_\delta(x)$. El **input** del algoritmo debe ser δ y el **output** la raíz obtenida. Usted debe determinar el número de iteraciones o error para obtener una aproximación *razonable*. El algoritmo debe tener definido completamente todos los parámetros extras para su ejecución.

(b) ✍ Proponga un algoritmo, que tenga **convergencia superior a lineal**, para obtener la raíz de $f_\delta(x)$. El **input** del algoritmo debe ser δ y el **output** la raíz obtenida. Usted debe determinar el número de iteraciones o error para obtener una aproximación *razonable*. El algoritmo debe tener definido completamente todos los parámetros extras para su ejecución.

(c) ⌨ Implemente alguno de los 2 algoritmos propuestos y ejecútelo para obtener la raíz de $f_{1+j}(x)$, donde j representa el último dígito de su rol.

(d) ⌨ Grafique la función $r(\delta)$ para $\delta \in [0, 100]$, utilice por lo menos 100 puntos para el plot.

2. Sea A_i, B_i, C_i, X, $Y \in \mathbb{R}^{n\times n}$ para $i \in \{1, 2\}$. Considere la siguiente variante de la ecuación de Sylvester:

$$\begin{aligned} A_1\,X + Y\,B_1 &= C_1 \\ X\,A_2 + B_2\,Y &= C_2 \end{aligned}$$

donde A_1 y B_2 son simétricas y definida positivas, y C_1 y C_2 son matrices no nulas. Recordar que no está permitido obtener la inversa de una matriz.

(a) ✍ Proponga un algoritmo basado en una iteración de punto fijo de alta dimensión que obtenga una aproximación de X y Y. Este será denominado como Algoritmo 1.

(b) ✍ Proponga un algoritmo basado en GMRes que obtenga X y Y. Este será denominado como Algoritmo 2.

(c) ⌨ Implemente Algoritmo 1.

(d) ⌨ Implemente Algoritmo 2.

(e) ⌨ Considere $n = 20$ y las matrices generadas en el código adjunto adjunto. Muestre como converge (o diverge) el valor del residuo relativo del sistema de ecuaciones lineales anteriormente descrito.

```
import numpy as np
np.random.seed(0)
n = 20
A1 = np.random.rand(n,n)
A1 = np.dot(A1.T,A1)
B1 = np.random.rand(n,n)
C1 = np.zeros((n,n))
C1[0,0] = 1
A2 = np.random.rand(n,n)
B2 = np.random.rand(n,n)
B2 = np.dot(B2.T,B2)
C2 = np.random.rand(n,n)
```