Álgebra Lineal Computacional

Primer Cuatrimestre 2023

Trabajo Práctico N° 1.

El objetivo de este trabajo práctico es implementar el método de la potencia para encontrar el máximo autovalor de una matriz cuyas dimensiones son de un tamaño considerable. Además se quiere estudiar si existen casos en los cuales es particularmente beneficioso usar este método.

Ejercicio 1.
Desarrollar un programa que dada una una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ y un entero positivo k, realice k iteraciones del método de la potencia con un vector aleatorio inicial $v \in \mathbb{R}^n$. El programa debe devolver un vector $\boldsymbol{a} \in \mathbb{R}^k$, donde a_i sea la aproximación al autovalor obtenida en el paso i.

Recomendaciones:
Pueden comparar con np.linalg.eigvals() para verificar resultados.

Ejercicio 2.
Se quiere probar el método con distintos tipos de matrices. Para ello implementar una función que genere las siguientes 4 matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\mathbf{C}$ y $\mathbf{D}$, que tenga como parámetro de entrada el tamaño de las matrices. Los elementos de cada matriz se definen según:

- matriz $\boldsymbol{A}$: Los elementos a_{ij} son tomados al azar utilizando la función *random*() de la librería *numpy* de Python.
- matriz $\boldsymbol{B}$: Los elementos b_{ij} son tomados al azar utilizando la función *random*() de la librería *numpy* de Python y además se cumple que $b_{ij} = b_{ji}$.
- matriz $\mathbf{C}$: Los elementos $c_{ij} = b_{ij}$ si $i \neq j$ y $c_{ij} = (b_{ij} + 100)$ si $i = j$.
- matriz $\mathbf{D}$ Los elementos $d_{ij} = b_{ij}$ si $i \neq j$ y $d_{ij} = (b_{ij} + 1000)$ si $i = j$.

En este ejercicio trabajamos con $\boldsymbol{A}$, $\boldsymbol{B}$, $\mathbf{C}$ y $\mathbf{D} \in \mathbb{R}^{100\times 100}$.

(a) Se pide realizar 100 iteraciones con el método de la potencia y graficar en cada caso el valor de la aproximación al autovalor en el paso i en función del número de iteraciones.

(b) ¿Qué puede concluir al comparar los gráficos de las distintas matrices?

(c) ¿Considera que el método converge rápidamente?

Si es necesario puede generar más de una matriz para cada caso, es decir, varias matrices de tipo $\boldsymbol{A}$, $\boldsymbol{B}$, $\mathbf{C}$ y $\mathbf{D}$.

Ejercicio 3.
Está probado que la velocidad de convergencia del método está dada por la relación entre el segundo autovalor de mayor módulo y el primer autovalor de mayor módulo.
Más precisamente, el error en cada paso se multiplica aproximadamente por $(\frac{\lambda_2}{\lambda_1})^2$.

Si se define el vector de errores $\mathbf{e} \in \mathbb{R}^{100}$, tal que sus elementos se calculan como:

$$e_i = |\lambda_{\max} - a_i|$$

donde $\lambda_{\max}$ es el autovalor de mayor módulo de $\boldsymbol{A}$, y a_i es la aproximación del autovalor en la iteración i (ver ejercicio 1).

(a) Graficar los errores $\log(e_i)$ en función del número de iteración para los casos de las matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\mathbf{C}$ y $\mathbf{D}$ del ejercicio 2.

(b) Sabiendo que el factor por el que se multiplica el error es aproximadamente $(\frac{\lambda_2}{\lambda_1})^2$, la pendiente de la recta obtenida debería ser aproximadamente $2 \cdot log(\frac{\lambda_2}{\lambda_1})$.
Para comparar los valores obtenidos experimentalmente, en el mismo gráfico representar $\log(e_i)$ del ítem (a) y la función $y(x) = 2 \cdot log(\frac{\lambda_2}{\lambda_1}) \cdot x + log(e_0)$.

<u>*Recomendaciones:*</u>
Se puede usar la función sorted() para ordenar los coeficientes de un vector. En este caso que trabajamos con una matriz simétrica también puede ser útil la función np.linalg.eigh().

(c) ¿Qué puede concluir a partir de los gráficos anteriores respecto a cada uno de los distintos tipos de matrices que analizó?

<u>*Observación:*</u>
Esperamos que las respuestas a las preguntas las hagan de manera simple y coloquial, no pretendemos formalismo en el texto a presentar.