**Problem 1:** Desarrollar un programa que calcule los $k$ valores propios más grandes en magnitud de una matriz $A$ mediante el método de la potencia. **(5 puntos)**

**Solution.** Este algoritmo consiste de dos pedazos funciones principales, `iterative_power` y `get_m_largest_eigenvalues`. El primer método tiene la función de encontrar el eigenvalor de mayor magnitud. La segunda función utiliza la primera de manera iterativa. Después de encontrar el más grande, utilizamos una técnica de *deflación*, que consiste en aplicar el mismo método de la potencia, pero con una matriz modificada a la original. Específicamente, si queremos el eigenvalor $m$ más grande, los eigenvalores de la nueva matriz son exactamente los mismos, excepto los $m-1$ más grandes, que son reemplazados por el eigenvalor 0. Esto asegura que la aplicación del método de la potencia encuentre el eigenvalor especificado.
**Ejemplos:**

1. **Eigen_3x3.txt**

$$\begin{bmatrix} 5.0 & -1.778 & 0.0 \\ -1.778 & 9.0 & -1.778 \\ 0.0 & -1.778 & 10.0 \end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ Ejecutando nuestro programa con

```
make run-p1 ARGS="t5a/Eigen_3x3.txt t5a/x0_3.txt 3 1e-6 500",
```

obtenemos los 3 eigenvalores más grandes (en este caso, todos).

| Eigenvector | Eigenvalue |
|---|---|
| $\begin{bmatrix} 0.18 & -0.64 & 0.74 \end{bmatrix}$ | 11.538403 |
| $\begin{bmatrix} -0.37 & 0.92 & -0.17 \end{bmatrix}$ | 8.213811 |
| $\begin{bmatrix} 0.91 & -0.42 & 0.02 \end{bmatrix}$ | 4.247786 |

Table 1: Eigenvalores de Eigen_3x3.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

```
python3 p1_verify.py t5a/Eigen_3x3.txt 3 1
```

Este nos genera los valores

| Eigenvector | Eigenvalue |
|---|---|
| $\begin{bmatrix}0.18 & -0.64 & 0.74\end{bmatrix}$ | 11.538405 |
| $\begin{bmatrix}-0.37 & 0.92 & -0.17\end{bmatrix}$ | 8.213809 |
| $\begin{bmatrix}0.91 & -0.42 & 0.02\end{bmatrix}$ | 4.247786 |

Table 2: Eigenvalores de Eigen_3x3.txt verificados con numpy.

Podemos observar muy poca diferencia entre los eigenvalores, por lo que podemos asumir que han sido verificados como correctos.

2. **Eigen_5x5.txt**

$$\begin{bmatrix}0.2 & 0.1 & 1 & 1 & 0\\ 0.1 & 4 & -1 & 1 & -1\\ 1 & -1 & 60 & 0 & -2\\ 1 & 1 & 0 & 8 & 4\\ 0 & -1 & -2 & 4 & 700\end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix}1 & 1 & 1 & 1 & 1\end{bmatrix}^T$ Ejecutando nuestro programa con

`make run-p1 ARGS="t5a/Eigen_5x5.txt t5a/x0_5.txt 3 1e-6 500"`,

obtenemos los 3 eigenvalores más grandes (en este caso, todos).

| $n$ | Eigenvalue |
|---|---|
| 1 | 700.030781 |
| 2 | 60.028366 |
| 3 | 8.338448 |

Table 3: Eigenvalores de Eigen_5x5.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

`python3 p1_verify.py t5a/Eigen_5x5.txt 3 1`

Este nos genera los valores

| n | Eigenvalue |
|---|---|
| 1 | 700.030781 |
| 2 | 60.028366 |
| 3 | 8.338447 |

Table 4: Eigenvalores de Eigen_5x5.txt verificados con numpy.

Podemos observar muy poca diferencia entre los eigenvalores, por lo que podemos asumir que han sido verificados como correctos.

3. **Eigen_50x50.txt**

$$\begin{bmatrix} 10.000 & 0.084 & 0.039 & \cdots & 0.052 & 0.078 & 0.007 \\ 0.084 & 20.000 & 0.095 & \cdots & 0.040 & 0.053 & 0.040 \\ 0.039 & 0.095 & 30.000 & \cdots & 0.084 & 0.013 & 0.052 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0.081 & 0.044 & 0.095 & \cdots & 380.000 & 0.038 & 0.081 \\ 0.092 & 0.092 & 0.005 & \cdots & 0.041 & 490.000 & 0.092 \\ 0.007 & 0.040 & 0.052 & \cdots & 0.078 & 0.078 & 500.000 \end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \end{bmatrix}^T$ Ejecutando nuestro programa con

```
make run-p1 ARGS="t5a/Eigen_50x50.txt t5a/x0_50.txt 7 1e-6 500",
```

obtenemos los73 eigenvalores más grandes (en este caso, todos).

| $n$ | Eigenvalue |
|---|---|
| 1 | 500.001591 |
| 2 | 490.001430 |
| 3 | 480.000520 |
| 4 | 470.001067 |
| 5 | 460.001265 |
| 6 | 450.000481 |
| 7 | 440.000418 |

Table 5: Eigenvalores de Eigen_50x50.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

```
python3 p1_verify.py t5a/Eigen_50x50.txt 7 1
```

Este nos genera los valores

| n | Eigenvalue |
|---|---|
| 1 | 500.001543 |
| 2 | 490.001431 |
| 3 | 480.000527 |
| 4 | 470.001058 |
| 5 | 460.001269 |
| 6 | 450.000483 |
| 7 | 440.000419 |

Table 6: Eigenvalores de Eigen_50x50.txt verificados con numpy.

Podemos observar muy poca diferencia entre los eigenvalores, por lo que podemos asumir que han sido verificados como correctos.

4. **Eigen_125x125.txt**

$$\begin{bmatrix} 9.566e-05 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1.965e-04 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 3.802e-04 & 1.356e-06 & 0 \\ 0 & 0 & 0 & \cdots & 1.356e-06 & 3.701e-04 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 3.522e-04 \end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$ Ejecutando nuestro programa con

```
make run-p1 ARGS="t5a/Eigen_125x125.txt t5a/x0_125.txt 7 1e-6 500",
```

obtenemos los 7 eigenvalores más grandes.

| $n$ | Eigenvalue |
|---|---|
| 1 | 1.000000 |
| 2 | 0.000598 |
| 3 | 0.000581 |
| 4 | 0.000561 |
| 5 | 0.000556 |
| 6 | 0.000551 |
| 7 | 0.000548 |

Table 7: Eigenvalores más grandes de Eigen_125x125.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

```
python3 p1_verify.py t5a/Eigen_125x125.txt 18 1
```

Este nos genera los valores

| n | Eigenvalue |
|---|---|
| 1 | 1.000000 |
| 2 | 1.000000 |
| ⋮ | ⋮ |
| 12 | 1.000000 |
| 13 | 0.000598 |
| 14 | 0.000581 |
| 15 | 0.000561 |
| 16 | 0.000556 |
| 17 | 0.000551 |
| 18 | 0.000548 |

Table 8: Eigenvalores más grandes de Eigen_125x125.txt verificados con numpy.

Por la naturaleza del algoritmo, no se va a encontrar el mismo eigenvalor en la siguiente iteración. Entonces, encontramos los 7 eigenvalores más grandes no repetidos. Como podemos confirmar, estos corresponden a los que regresa numpy.

◇

**Problem 2:** Desarrollar un programa que calcule los $k$ valores propios más pequeños en magnitud de una matriz $A$ mediante el método de la potencia inversa. **(5 puntos)**

**Solution.** Para encontrar los eigenvalores más pequeños, realizamos el método de la potencia inversa. Después de cada iteración, realizamos un ajuste en los vectores para ortogonalizarlos con los eigenvalores anteriores, asegurando que el nuevo eigenvalor se encuentre en otro eigenespacio, permitiendo encontrar los otros eigenvalores más pequeños.

1. **Eigen_3x3.txt**

$$\begin{bmatrix} 5.0 & -1.778 & 0.0 \\ -1.778 & 9.0 & -1.778 \\ 0.0 & -1.778 & 10.0 \end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ Ejecutando nuestro programa con

`make run-p2 ARGS="t5a/Eigen_3x3.txt t5a/x0_3.txt 3 1e-6 500"`,

obtenemos los 3 eigenvalores más pequeños (en este caso, todos).

| Eigenvector | Eigenvalue |
|---|---|
| $\begin{bmatrix} 0.91 & -0.42 & 0.02 \end{bmatrix}$ | 4.247786 |
| $\begin{bmatrix} -0.37 & 0.92 & -0.17 \end{bmatrix}$ | 8.213811 |
| $\begin{bmatrix} 0.18 & -0.64 & 0.74 \end{bmatrix}$ | 11.538403 |

Table 9: 3 eigenvalores más pequeños de Eigen_3x3.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

`python3 p1_verify.py t5a/Eigen_3x3.txt 3 0`

Este nos genera los valores

| Eigenvector | Eigenvalue |
|---|---|
| $\begin{bmatrix}0.91 & -0.42 & 0.02\end{bmatrix}$ | 4.247786 |
| $\begin{bmatrix}-0.37 & 0.92 & -0.17\end{bmatrix}$ | 8.213809 |
| $\begin{bmatrix}0.18 & -0.64 & 0.74\end{bmatrix}$ | 11.538405 |

Table 10: 3 eigenvalores más pequeños de Eigen_3x3.txt verificados con numpy.

Podemos observar muy poca diferencia entre los eigenvalores, por lo que podemos asumir que han sido verificados como correctos.

2. **Eigen_5x5.txt**

$$\begin{bmatrix}0.2 & 0.1 & 1 & 1 & 0\\ 0.1 & 4 & -1 & 1 & -1\\ 1 & -1 & 60 & 0 & -2\\ 1 & 1 & 0 & 8 & 4\\ 0 & -1 & -2 & 4 & 700\end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix}1 & 1 & 1 & 1 & 1\end{bmatrix}^T$ Ejecutando nuestro programa con

`make run-p2 ARGS="t5a/Eigen_5x5.txt t5a/x0_5.txt 3 1e-6 500"`,

obtenemos los 3 eigenvalores más pequeños.

| n | Eigenvalue |
|---|---|
| 1 | 0.057075 |
| 2 | 3.745331 |
| 3 | 8.338447 |

Table 11: Eigenvalores de Eigen_5x5.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

`python3 p1_verify.py t5a/Eigen_5x5.txt 3 0`

Este nos genera los valores

| n | Eigenvalue |
|---|---|
| 1 | 0.057075 |
| 2 | 3.745331 |
| 3 | 8.338447 |

Table 12: 3 eigenvalores más pequeños de Eigen_5x5.txt verificados con numpy.

Podemos observar muy poca diferencia entre los eigenvalores, por lo que podemos asumir que han sido verificados como correctos.

3. **Eigen_50x50.txt**

$$\begin{bmatrix} 10.000 & 0.084 & 0.039 & \cdots & 0.052 & 0.078 & 0.007 \\ 0.084 & 20.000 & 0.095 & \cdots & 0.040 & 0.053 & 0.040 \\ 0.039 & 0.095 & 30.000 & \cdots & 0.084 & 0.013 & 0.052 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0.081 & 0.044 & 0.095 & \cdots & 380.000 & 0.038 & 0.081 \\ 0.092 & 0.092 & 0.005 & \cdots & 0.041 & 490.000 & 0.092 \\ 0.007 & 0.040 & 0.052 & \cdots & 0.078 & 0.078 & 500.000 \end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$ Ejecutando nuestro programa con

```
make run-p2 ARGS="t5a/Eigen_50x50.txt t5a/x0_50.txt 7 1e-6 500",
```

obtenemos los 7 eigenvalores más pequeños.

| $n$ | Eigenvalue |
|---|---|
| 1 | 9.998050 |
| 2 | 19.998794 |
| 3 | 29.998932 |
| 4 | 39.999764 |
| 5 | 49.998366 |
| 6 | 59.999846 |
| 7 | 69.999136 |

Table 13: 7 eigenvalores más pequeños de Eigen_50x50.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

```
python3 p1_verify.py t5a/Eigen_50x50.txt 7 0
```

Este nos genera los valores

| n | Eigenvalue |
|---|---|
| 1 | 9.998050 |
| 2 | 19.998794 |
| 3 | 29.998931 |
| 4 | 39.999763 |
| 5 | 49.998366 |
| 6 | 59.999845 |
| 7 | 69.999135 |

Table 14: Eigenvalores de Eigen_50x50.txt verificados con numpy.

Podemos observar muy poca diferencia entre los eigenvalores, por lo que podemos asumir que han sido verificados como correctos.

4. **Eigen_125x125.txt**

$$\begin{bmatrix} 9.566e-05 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1.965e-04 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 3.802e-04 & 1.356e-06 & 0 \\ 0 & 0 & 0 & \cdots & 1.356e-06 & 3.701e-04 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 3.522e-04 \end{bmatrix}$$

Como vector inicial, usaremos $\mathbf{x}_0 = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \end{bmatrix}^T$ Ejecutando nuestro programa con

```
make run-p2 ARGS="t5a/Eigen_125x125.txt t5a/x0_125.txt 7 1e-9 500",
```

obtenemos los 7 eigenvalores más pequeños.

| $n$ | Eigenvalue |
|---|---|
| 1 | 0.000002 |
| 2 | 0.000008 |
| 3 | 0.000017 |
| 4 | 0.000018 |
| 5 | 0.000026 |
| 6 | 0.000036 |
| 7 | 0.000038 |

Table 15: Eigenvalores de Eigen_125x125.txt

Como verificación, utilizamos la librería de `numpy`. Corremos el programa (antes activando el virtual environment, e instalando numpy mediante `pip install numpy` si es necesario) mediante

```
python3 p1_verify.py t5a/Eigen_125x125.txt 7 0
```

Este nos genera los valores

| n | Eigenvalue |
|---|---|
| 1 | 0.000002 |
| 2 | 0.000008 |
| 3 | 0.000017 |
| 4 | 0.000018 |
| 5 | 0.000026 |
| 6 | 0.000036 |
| 7 | 0.000038 |

Table 16: Eigenvalores de Eigen_125x125.txt verificados con numpy.

Para esta matriz, usamos una tolerancia de menor magnitud para aproximar los eigenvalores de manera más cercana a los reales. Con esto, verificamos que los calculados son correctos.

◇