|  |
| --- |
| Métodos Númericos Avanzados |
| Trabajo Práctico Nro. 1 |
| Grcar Matrix - Autovalores |

|  |
| --- |
|  |

**Grupo Nro. 3:**

**Maximiliano J. Valverde (51.158)**

**Ma. Florencia Besteiro (51117)**

INDICE

[**Indice** 1](#_Toc432462117)

[Introducción 2](#_Toc432462118)

[Grcar Matrix 2](#_Toc432462119)

[2](#_Toc432462120)

[Primera Aproximación 3](#_Toc432462121)

[Teorema de Laplace 3](#_Toc432462122)

[Menor Complementario 3](#_Toc432462123)

[Adjunto de un Elemento 4](#_Toc432462124)

[Algoritmo 4](#_Toc432462125)

[4](#_Toc432462126)

[Dificultades Encontradas 4](#_Toc432462127)

[Segunda Aproximación 6](#_Toc432462128)

[Descomposición QR 6](#_Toc432462129)

[Resultados Obtenidos 7](#_Toc432462130)

[Observación 8](#_Toc432462131)

[Apéndice 8](#_Toc432462132)

[Algoritmo para Teorema de Laplace 8](#_Toc432462133)

[Algoritmo para Descomposición QR con Single Shift y Double Shift 9](#_Toc432462134)

[Cuadro de resultados nro. 1 10](#_Toc432462135)

[Bibliografía 11](#_Toc432462136)

# Introducción

El objetivo de este trabajo consiste, en obtener los autovalores de cierta matriz. Para ello, se debieron aplicar los conceptos vistos en la materia para su cálculo, a la vez que investigar sobre cómo efectivizar los algoritmos para poder obtener resultados en un tiempo razonable.

# Grcar Matrix

La matriz de Grcar es una matriz de testeo creada por Joseph Grcar y es usada para testear algoritmos de descomposición y obtención de autovalores. La misma tiene la siguiente forma:

# 

La diagonal principal y las tres supradiagonales, están pobladas por 1’s. La diagonal por debajo de la principal de -1’s y el resto de la matriz, posee 0’s.

# Primera Aproximación

En primera instancia se pensó en obtener los autovalores a través del polinomio característico. Para ello se consideró la siguiente fórmula, donde es nuestra matriz, es la identidad y es un vector no nulo:



Luego, para obtener los autovalores, es necesario encontrar aquellos valores de que satisfacen dicha fórmula, por lo que obtenemos una nueva expresión,



donde representa el polinomio característico de A.

Pero debido a las dimensiones de nuestra matriz, nos encontramos en la necesidad de buscar un algoritmo eficiente para encontrar el determinante. Para ello, recurrimos al Teorema de Laplace.

## Teorema de Laplace

Este teorema permite simplificar el cálculo de determinantes de matrices de altas dimensiones mediante la descomposición de la misma en matrices de menor dimensión.

El mismo afirma que el determinante de una matriz, es igual a la suma de los determinantes de los adjuntos de cualquier fila o columna de la matriz, lo que reduce un determinante de dimensión a determinantes de dimensión .

### Menor Complementario

Partiendo de una matriz cuadrada: , de orden , se llama menor complementario del elemento  a_{ij} \; , y lo representamos  \alpha_{ij} \;  al determinante de la matriz cuadrada de orden que resulta de eliminar de la matriz A la fila y la columna .

### Adjunto de un Elemento

El Adjunto de un elemento se representa por  A_{ij} \;  y se define de la siguiente forma,

    A_{ij} = (-1)^{(i+j)} \; \alpha_{ij} 

donde  \alpha_{ij} \;  es el menor complementario del elemento.

### Algoritmo

Siguiendo este teorema, entonces es posible crear un algoritmo recursivo que permita obtener

el determinante de matrices de grandes dimensiones de la siguiente manera:

## \det (A_{j,j}) = \left \{ \begin{array}{llcl} si & j = 1 & \to & a_{1,1} \\ \\ si & j > 1 & \to & \displaystyle \sum_{k=1}^j \; (-1)^{(1+k)} \cdot a_{1,k} \cdot \det( \alpha_{1,k}) \end{array} \right .

## 

## 

donde se va acumulando y calculando el determinante para cada adjunto, el cual va disminuyendo de dimensión con cada paso recursivo.

## Dificultades Encontradas

Durante las pruebas, este algoritmo mostró buenos resultados para valores pequeños de , pero el tiempo de ejecución aumentaba exponencialmente hacia valores inadmisibles a partir de . Debido a la naturaleza del algoritmo, se pensó en paralelizar el cálculo de los determinantes de los adjuntos utilizando *threads*, pero finalmente se decidió buscar un método alternativo y descartar el Teorema de Laplace.

# Segunda Aproximación

## Descomposición QR

Se descompone la matriz A en un producto de dos matrices Q y R, donde Q es una matriz ortogonal (la inversa es igual a la transpuesta de la matriz) y R una triangular superior.

El algoritmo básico consiste en:

1. Obtener la descomposición de A en Q y R.
2. A’ = R\*Q
3. Si A’ es una matriz donde sólo la diagonal principal tiene números distintos de cero o muy pequeños, entonces los números de la diagonal son los autovalores de la matriz. Si eso no ocurre, vuelvo al paso 1).

El algoritmo de factorización QR utilizado, fue el de Gram-Schmidt visto en clase.

Este algoritmo, converge de manera muy lenta, por lo que a esto, le agregamos el concepto de “Single Shift” y “Double Shift”.

Luego de obtener A’, analizamos los últimos elementos de la misma:

|  |  |  |  |
| --- | --- | --- | --- |
| ……. | ……. | …... | ……. |
| ……. | ……. | ….... | ……. |
| ……. | ……. | a | b |
| ……. | ……. | c | d |

Si se cumple la condición 1:

Significa que el valor “d” es autovalor de la matriz A. Luego de guardar dicho valor, aplicamos deflación sobre la matriz, esto es, hacemos más pequeña su dimensión eliminando la última fila y columna. El que se menciona, es una cota de tolerancia. En nuestras pruebas usamos los valores 10-6, 10-7 y 10-8.

Si esta no se cumple, analizamos los últimos elementos de la matriz A de dimensión 3x3:

|  |  |  |  |
| --- | --- | --- | --- |
| ……. | ……. | …... | ……. |
| ……. | f | g | h |
| ……. | i | a | b |
| ……. | j | c | d |

La condición que debe cumplirse para aplicar “Double Shift” es:

Si la misma se cumple, significa que los autovalores de la matriz formada por los últimos elementos (a,b,c,d), son autovalores de A, los cuales los obtenemos con el polinomio característico. Una vez guardados, aplicamos deflación en la matriz eliminando las dos últimas filas y columnas. De esta manera, la matriz A va disminuyendo su tamaño y agilizando la obtención de autovalores de la misma.

En nuestro algoritmo se utilizó QR junto con Householder para aumentar la convergencia el mismo.

### Resultados Obtenidos

Con este algoritmo, pudimos obtener los autovalores de una matriz de dimensión de 50X50 en 82 segundos, lo que en comparación con lo obtenido en los intentos anteriores, fue un avance positivo. Sin embargo, cuando intentamos calcular los autovalores para una matriz de 100x100, el tiempo tendió a infinito (consideramos “infinito” más de una hora de ejecución).

*Ver en Apéndice, cuadro de resultados nro. 1*

Luego de aplicar la mejora en eficiencia al algoritmo aplicando householder, pudimos obtener resultados para una dimensión de n = 100 en un tiempo finito.

*Ver en Apéndice, cuadro de resultados nro. 2*

# Observación

Durante el período de pruebas en la última instancia (QR +Householder + Simple Shift + Double Shift), variamos la dimensión de la matriz desde n = 100 con aumentos de 10, es decir, n = 100 , n = 110, n = 120, etc.

*Ver en Apéndice, cuadro de resultados nro. 3*

Como se puede observar, para n = 100 el tiempo de ejecución supera los diez mil segundos, mientras que para n = 110, el tiempo de ejecución es de 580 segundos. A su vez, con n = 120, el tiempo de ejecución sigue sin superar los mil segundos.

Si bien analizamos esta situación no pudimos encontrar con el motivo de la diferencia tan grande que existe entre los tiempos de ejecución ya que estas pruebas se realizaron sin ningún tipo de modificación en el código entre una y otra.

A su vez, al hacer un segundo análisis, descubrimos que esto también sucede para valores de n menores. Los tiempos de ejecución aumentan cuando n es mayor, pero existen “picos” de tiempo para ciertos valores de n.

*Ver en Apéndice, cuadro comparativo*

# Apéndice

## Algoritmo para Teorema de Laplace

function determinant = Laplace(matrix, n)

determinant = 0;

%CASO BASE - Matriz de orden 2

if (n == 2)

determinant = det(matrix);

return;

end

%PASO RECURSIVO - Calculo el adjunto y ejecuto laplace.

%Se ejecuta por profundidad. Se acumula el determinante resultado de

%toda la rama de matrices que corresponden a cada columna. Luego avanza

%el loop y se prosigue con la siguiente.

for col = 1 : n

firstElement = matrix(1,col);

[sign, tempAdjunt] = Adjunt(matrix, col);

%Ejecuto Laplace con el adjunto de orden n-1

determinant = determinant + firstElement \* (sign \* Laplace(tempAdjunt, n-1));

end

end

## Algoritmo para Descomposición QR con Single Shift y Double Shift

function out = eigenValues(matrix)

n = size(matrix,1);

out = [];

tic;

loops = 0;

error = 10 ^ (-7);

A = matrix;

while( n > 2)

[Q,R] = householderQR(A);

A = R \* Q;

%Single Shift%

a = A(n-1,n-1);

b = A(n-1,n);

c = A(n,n-1);

d = A(n,n);

%Double Shift%

f = A(n-2,n-2);

i = A(n-1,n-2);

if( abs(c) < error \* ( abs(a) + abs(d) ))

out = [out ;d];

A = A(1:n-1,1:n-1);

n = n - 1;

elseif ( abs(i) < error \* ( abs(f) + abs(a) ))

aux = lambdas(a,b,c,d);

out = [out ; aux];

A = A(1:n-2,1:n-2);

n = n - 2;

end

loops = loops + 1;

end

if( n == 2)

a = A(1,1);

b = A(1,2);

c = A(2,1);

d = A(2,2);

aux = lambdas(a,b,c,d);

out = [out ; aux];

elseif( n == 1)

out = [out ; A(1,1)];

end

end

function values = lambdas(a,b,c,d)

x1 = -a-d;

x0 = (a\*d) - (b\*c);

p = [1 x1 x0];

values = roots(p);

end

## Cuadro de resultados nro. 1

## Cuadro de resultados nro. 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **n = 100 , Error = 10^(-6)** | | **n = 100 , Error = 10^(-7)** | | **n = 100 , Error = 10^(-8)** | |
| **NUESTRO** | **MATLAB** | **NUESTRO** | **MATLAB** | **NUESTRO** | **MATLAB** |
| 1.6071 - 0.0471i | 1.6070 - 0.0476i | 1.6070 - 0.0475i | 1.6070 - 0.0476i | 1.6070 - 0.0476i | 1.6070 - 0.0476i |
| 1.6071 + 0.0471i | 1.6070 + 0.0476i | 1.6070 + 0.0475i | 1.6070 + 0.0476i | 1.6070 + 0.0476i | 1.6070 + 0.0476i |
| 1.6077 - 0.1426i | 1.6078 - 0.1425i | 1.6078 - 0.1425i | 1.6078 - 0.1425i | 1.6078 - 0.1425i | 1.6078 - 0.1425i |
| 1.6077 + 0.1426i | 1.6078 + 0.1425i | 1.6078 + 0.1425i | 1.6078 + 0.1425i | 1.6078 + 0.1425i | 1.6078 + 0.1425i |
| 1.6094 - 0.2368i | 1.6094 - 0.2368i | 1.6094 - 0.2368i | 1.6094 - 0.2368i | 1.6094 - 0.2368i | 1.6094 - 0.2368i |
| 1.6094 + 0.2368i | 1.6094 + 0.2368i | 1.6094 + 0.2368i | 1.6094 + 0.2368i | 1.6094 + 0.2368i | 1.6094 + 0.2368i |
| 1.6118 - 0.3302i | 1.6118 - 0.3302i | 1.6118 - 0.3302i | 1.6118 - 0.3302i | 1.6118 - 0.3302i | 1.6118 - 0.3302i |
| 1.6118 + 0.3302i | 1.6118 + 0.3302i | 1.6118 + 0.3302i | 1.6118 + 0.3302i | 1.6118 + 0.3302i | 1.6118 + 0.3302i |
| 1.6151 - 0.4222i | 1.6151 - 0.4222i | 1.6151 - 0.4222i | 1.6151 - 0.4222i | 1.6151 - 0.4222i | 1.6151 - 0.4222i |
| 1.6151 + 0.4222i | 1.6151 + 0.4222i | 1.6151 + 0.4222i | 1.6151 + 0.4222i | 1.6151 + 0.4222i | 1.6151 + 0.4222i |
| 1.6194 - 0.5123i | 1.6194 - 0.5123i | 1.6194 - 0.5123i | 1.6194 - 0.5123i | 1.6194 - 0.5123i | 1.6194 - 0.5123i |
| 1.6194 + 0.5123i | 1.6194 + 0.5123i | 1.6194 + 0.5123i | 1.6194 + 0.5123i | 1.6194 + 0.5123i | 1.6194 + 0.5123i |
| 1.6247 - 0.6000i | 1.6247 - 0.6000i | 1.6247 - 0.6000i | 1.6247 - 0.6000i | 1.6247 - 0.6000i | 1.6247 - 0.6000i |
| 1.6247 + 0.6000i | 1.6247 + 0.6000i | 1.6247 + 0.6000i | 1.6247 + 0.6000i | 1.6247 + 0.6000i | 1.6247 + 0.6000i |
| 1.0016 - 1.4537i | 1.0016 - 1.4537i | 1.0016 - 1.4537i | 1.0016 - 1.4537i | 1.0016 - 1.4537i | 1.0016 - 1.4537i |
| 1.0016 + 1.4537i | 1.0016 + 1.4537i | 1.0016 + 1.4537i | 1.0016 + 1.4537i | 1.0016 + 1.4537i | 1.0016 + 1.4537i |
| 0.9295 - 1.5046i | 0.9295 - 1.5046i | 0.9295 - 1.5046i | 0.9295 - 1.5046i | 0.9295 - 1.5046i | 0.9295 - 1.5046i |
| 0.9295 + 1.5046i | 0.9295 + 1.5046i | 0.9295 + 1.5046i | 0.9295 + 1.5046i | 0.9295 + 1.5046i | 0.9295 + 1.5046i |
| 1.0757 - 1.4043i | 1.0757 - 1.4043i | 1.0757 - 1.4043i | 1.0757 - 1.4043i | 1.0757 - 1.4043i | 1.0757 - 1.4043i |
| 1.0757 + 1.4043i | 1.0757 + 1.4043i | 1.0757 + 1.4043i | 1.0757 + 1.4043i | 1.0757 + 1.4043i | 1.0757 + 1.4043i |
| 1.6311 - 0.6846i | 1.6311 - 0.6846i | 1.6311 - 0.6846i | 1.6311 - 0.6846i | 1.6311 - 0.6846i | 1.6311 - 0.6846i |
| 1.6311 + 0.6846i | 1.6311 + 0.6846i | 1.6311 + 0.6846i | 1.6311 + 0.6846i | 1.6311 + 0.6846i | 1.6311 + 0.6846i |
| 0.8597 - 1.5565i | 0.8597 - 1.5565i | 0.8597 - 1.5565i | 0.8597 - 1.5565i | 0.8597 - 1.5565i | 0.8597 - 1.5565i |
| 0.8597 + 1.5565i | 0.8597 + 1.5565i | 0.8597 + 1.5565i | 0.8597 + 1.5565i | 0.8597 + 1.5565i | 0.8597 + 1.5565i |
| 1.1513 - 1.3573i | 1.1513 - 1.3573i | 1.1513 - 1.3573i | 1.1513 - 1.3573i | 1.1513 - 1.3573i | 1.1513 - 1.3573i |
| 1.1513 + 1.3573i | 1.1513 + 1.3573i | 1.1513 + 1.3573i | 1.1513 + 1.3573i | 1.1513 + 1.3573i | 1.1513 + 1.3573i |
| 0.7923 - 1.6087i | 0.7923 - 1.6087i | 0.7923 - 1.6087i | 0.7923 - 1.6087i | 0.7923 - 1.6087i | 0.7923 - 1.6087i |
| 0.7923 + 1.6087i | 0.7923 + 1.6087i | 0.7923 + 1.6087i | 0.7923 + 1.6087i | 0.7923 + 1.6087i | 0.7923 + 1.6087i |
| 1.2278 - 1.3136i | 1.2278 - 1.3136i | 1.2278 - 1.3136i | 1.2278 - 1.3136i | 1.2278 - 1.3136i | 1.2278 - 1.3136i |
| 1.2278 + 1.3136i | 1.2278 + 1.3136i | 1.2278 + 1.3136i | 1.2278 + 1.3136i | 1.2278 + 1.3136i | 1.2278 + 1.3136i |
| 1.6386 - 0.7655i | 1.6386 - 0.7655i | 1.6386 - 0.7655i | 1.6386 - 0.7655i | 1.6386 - 0.7655i | 1.6386 - 0.7655i |
| 1.6386 + 0.7655i | 1.6386 + 0.7655i | 1.6386 + 0.7655i | 1.6386 + 0.7655i | 1.6386 + 0.7655i | 1.6386 + 0.7655i |
| 0.7276 - 1.6607i | 0.7276 - 1.6607i | 0.7276 - 1.6607i | 0.7276 - 1.6607i | 0.7276 - 1.6607i | 0.7276 - 1.6607i |
| 0.7276 + 1.6607i | 0.7276 + 1.6607i | 0.7276 + 1.6607i | 0.7276 + 1.6607i | 0.7276 + 1.6607i | 0.7276 + 1.6607i |
| 1.3044 - 1.2740i | 1.3044 - 1.2740i | 1.3044 - 1.2740i | 1.3044 - 1.2740i | 1.3044 - 1.2740i | 1.3044 - 1.2740i |
| 1.3044 + 1.2740i | 1.3044 + 1.2740i | 1.3044 + 1.2740i | 1.3044 + 1.2740i | 1.3044 + 1.2740i | 1.3044 + 1.2740i |
| 0.6655 - 1.7121i | 0.6655 - 1.7121i | 0.6655 - 1.7121i | 0.6655 - 1.7121i | 0.6655 - 1.7121i | 0.6655 - 1.7121i |
| 0.6655 + 1.7121i | 0.6655 + 1.7121i | 0.6655 + 1.7121i | 0.6655 + 1.7121i | 0.6655 + 1.7121i | 0.6655 + 1.7121i |
| 1.6471 - 0.8418i | 1.6471 - 0.8418i | 1.6471 - 0.8418i | 1.6471 - 0.8418i | 1.6471 - 0.8418i | 1.6471 - 0.8418i |
| 1.6471 + 0.8418i | 1.6471 + 0.8418i | 1.6471 + 0.8418i | 1.6471 + 0.8418i | 1.6471 + 0.8418i | 1.6471 + 0.8418i |
| 1.3798 - 1.2395i | 1.3798 - 1.2395i | 1.3798 - 1.2395i | 1.3798 - 1.2395i | 1.3798 - 1.2395i | 1.3798 - 1.2395i |
| 1.3798 + 1.2395i | 1.3798 + 1.2395i | 1.3798 + 1.2395i | 1.3798 + 1.2395i | 1.3798 + 1.2395i | 1.3798 + 1.2395i |
| 0.6062 - 1.7625i | 0.6062 - 1.7625i | 0.6062 - 1.7625i | 0.6062 - 1.7625i | 0.6062 - 1.7625i | 0.6062 - 1.7625i |
| 0.6062 + 1.7625i | 0.6062 + 1.7625i | 0.6062 + 1.7625i | 0.6062 + 1.7625i | 0.6062 + 1.7625i | 0.6062 + 1.7625i |
| 1.4521 - 1.2110i | 1.4521 - 1.2110i | 1.4521 - 1.2110i | 1.4521 - 1.2110i | 1.4521 - 1.2110i | 1.4521 - 1.2110i |
| 1.4521 + 1.2110i | 1.4521 + 1.2110i | 1.4521 + 1.2110i | 1.4521 + 1.2110i | 1.4521 + 1.2110i | 1.4521 + 1.2110i |
| 1.6564 - 0.9123i | 1.6564 - 0.9123i | 1.6564 - 0.9123i | 1.6564 - 0.9123i | 1.6564 - 0.9123i | 1.6564 - 0.9123i |
| 1.6564 + 0.9123i | 1.6564 + 0.9123i | 1.6564 + 0.9123i | 1.6564 + 0.9123i | 1.6564 + 0.9123i | 1.6564 + 0.9123i |
| 0.5498 - 1.8116i | 0.5498 - 1.8116i | 0.5498 - 1.8116i | 0.5498 - 1.8116i | 0.5498 - 1.8116i | 0.5498 - 1.8116i |
| 0.5498 + 1.8116i | 0.5498 + 1.8116i | 0.5498 + 1.8116i | 0.5498 + 1.8116i | 0.5498 + 1.8116i | 0.5498 + 1.8116i |
| 0.4963 - 1.8591i | 0.4963 - 1.8591i | 0.4963 - 1.8591i | 0.4963 - 1.8591i | 0.4963 - 1.8591i | 0.4963 - 1.8591i |
| 0.4963 + 1.8591i | 0.4963 + 1.8591i | 0.4963 + 1.8591i | 0.4963 + 1.8591i | 0.4963 + 1.8591i | 0.4963 + 1.8591i |
| 1.5191 - 1.1888i | 1.5191 - 1.1888i | 1.5191 - 1.1888i | 1.5191 - 1.1888i | 1.5191 - 1.1888i | 1.5191 - 1.1888i |
| 1.5191 + 1.1888i | 1.5191 + 1.1888i | 1.5191 + 1.1888i | 1.5191 + 1.1888i | 1.5191 + 1.1888i | 1.5191 + 1.1888i |
| 1.6661 - 0.9759i | 1.6661 - 0.9759i | 1.6661 - 0.9759i | 1.6661 - 0.9759i | 1.6661 - 0.9759i | 1.6661 - 0.9759i |
| 1.6661 + 0.9759i | 1.6661 + 0.9759i | 1.6661 + 0.9759i | 1.6661 + 0.9759i | 1.6661 + 0.9759i | 1.6661 + 0.9759i |
| 0.4458 - 1.9048i | 0.4458 - 1.9048i | 0.4458 - 1.9048i | 0.4458 - 1.9048i | 0.4458 - 1.9048i | 0.4458 - 1.9048i |
| 0.4458 + 1.9048i | 0.4458 + 1.9048i | 0.4458 + 1.9048i | 0.4458 + 1.9048i | 0.4458 + 1.9048i | 0.4458 + 1.9048i |
| 1.5780 - 1.1724i | 1.5780 - 1.1724i | 1.5780 - 1.1724i | 1.5780 - 1.1724i | 1.5780 - 1.1724i | 1.5780 - 1.1724i |
| 1.5780 + 1.1724i | 1.5780 + 1.1724i | 1.5780 + 1.1724i | 1.5780 + 1.1724i | 1.5780 + 1.1724i | 1.5780 + 1.1724i |
| 1.6751 - 1.0312i | 1.6751 - 1.0312i | 1.6751 - 1.0312i | 1.6751 - 1.0312i | 1.6751 - 1.0312i | 1.6751 - 1.0312i |
| 1.6751 + 1.0312i | 1.6751 + 1.0312i | 1.6751 + 1.0312i | 1.6751 + 1.0312i | 1.6751 + 1.0312i | 1.6751 + 1.0312i |
| 0.3983 - 1.9484i | 0.3983 - 1.9484i | 0.3983 - 1.9484i | 0.3983 - 1.9484i | 0.3983 - 1.9484i | 0.3983 - 1.9484i |
| 0.3983 + 1.9484i | 0.3983 + 1.9484i | 0.3983 + 1.9484i | 0.3983 + 1.9484i | 0.3983 + 1.9484i | 0.3983 + 1.9484i |
| 1.6257 - 1.1602i | 1.6256 - 1.1602i | 1.6256 - 1.1602i | 1.6256 - 1.1602i | 1.6256 - 1.1602i | 1.6256 - 1.1602i |
| 1.6257 + 1.1602i | 1.6256 + 1.1602i | 1.6256 + 1.1602i | 1.6256 + 1.1602i | 1.6256 + 1.1602i | 1.6256 + 1.1602i |
| 1.6821 - 1.0768i | 1.6821 - 1.0768i | 1.6821 - 1.0768i | 1.6821 - 1.0768i | 1.6821 - 1.0768i | 1.6821 - 1.0768i |
| 1.6821 + 1.0768i | 1.6821 + 1.0768i | 1.6821 + 1.0768i | 1.6821 + 1.0768i | 1.6821 + 1.0768i | 1.6821 + 1.0768i |
| 1.6845 - 1.1115i | 1.6845 - 1.1115i | 1.6845 - 1.1115i | 1.6845 - 1.1115i | 1.6845 - 1.1115i | 1.6845 - 1.1115i |
| 1.6845 + 1.1115i | 1.6845 + 1.1115i | 1.6845 + 1.1115i | 1.6845 + 1.1115i | 1.6845 + 1.1115i | 1.6845 + 1.1115i |
| 1.6596 - 1.1492i | 1.6596 - 1.1492i | 1.6596 - 1.1492i | 1.6596 - 1.1492i | 1.6596 - 1.1492i | 1.6596 - 1.1492i |
| 1.6596 + 1.1492i | 1.6596 + 1.1492i | 1.6596 + 1.1492i | 1.6596 + 1.1492i | 1.6596 + 1.1492i | 1.6596 + 1.1492i |
| 0.3539 - 1.9897i | 0.3539 - 1.9897i | 0.3539 - 1.9897i | 0.3539 - 1.9897i | 0.3539 - 1.9897i | 0.3539 - 1.9897i |
| 0.3539 + 1.9897i | 0.3539 + 1.9897i | 0.3539 + 1.9897i | 0.3539 + 1.9897i | 0.3539 + 1.9897i | 0.3539 + 1.9897i |
| 1.6786 - 1.1348i | 1.6786 - 1.1348i | 1.6786 - 1.1348i | 1.6786 - 1.1348i | 1.6786 - 1.1348i | 1.6786 - 1.1348i |
| 1.6786 + 1.1348i | 1.6786 + 1.1348i | 1.6786 + 1.1348i | 1.6786 + 1.1348i | 1.6786 + 1.1348i | 1.6786 + 1.1348i |
| 0.3125 - 2.0286i | 0.3125 - 2.0286i | 0.3125 - 2.0286i | 0.3125 - 2.0286i | 0.3125 - 2.0286i | 0.3125 - 2.0286i |
| 0.3125 + 2.0286i | 0.3125 + 2.0286i | 0.3125 + 2.0286i | 0.3125 + 2.0286i | 0.3125 + 2.0286i | 0.3125 + 2.0286i |
| 0.2743 - 2.0648i | 0.2743 - 2.0648i | 0.2743 - 2.0648i | 0.2743 - 2.0648i | 0.2743 - 2.0648i | 0.2743 - 2.0648i |
| 0.2743 + 2.0648i | 0.2743 + 2.0648i | 0.2743 + 2.0648i | 0.2743 + 2.0648i | 0.2743 + 2.0648i | 0.2743 + 2.0648i |
| 0.2393 - 2.0984i | 0.2393 - 2.0984i | 0.2393 - 2.0984i | 0.2393 - 2.0984i | 0.2393 - 2.0984i | 0.2393 - 2.0984i |
| 0.2393 + 2.0984i | 0.2393 + 2.0984i | 0.2393 + 2.0984i | 0.2393 + 2.0984i | 0.2393 + 2.0984i | 0.2393 + 2.0984i |
| 0.2075 - 2.1291i | 0.2075 - 2.1291i | 0.2075 - 2.1291i | 0.2075 - 2.1291i | 0.2075 - 2.1291i | 0.2075 - 2.1291i |
| 0.2075 + 2.1291i | 0.2075 + 2.1291i | 0.2075 + 2.1291i | 0.2075 + 2.1291i | 0.2075 + 2.1291i | 0.2075 + 2.1291i |
| 0.1790 - 2.1568i | 0.1790 - 2.1568i | 0.1790 - 2.1568i | 0.1790 - 2.1568i | 0.1790 - 2.1568i | 0.1790 - 2.1568i |
| 0.1790 + 2.1568i | 0.1790 + 2.1568i | 0.1790 + 2.1568i | 0.1790 + 2.1568i | 0.1790 + 2.1568i | 0.1790 + 2.1568i |
| 0.1537 - 2.1815i | 0.1537 - 2.1815i | 0.1537 - 2.1815i | 0.1537 - 2.1815i | 0.1537 - 2.1815i | 0.1537 - 2.1815i |
| 0.1537 + 2.1815i | 0.1537 + 2.1815i | 0.1537 + 2.1815i | 0.1537 + 2.1815i | 0.1537 + 2.1815i | 0.1537 + 2.1815i |
| 0.1318 - 2.2030i | 0.1318 - 2.2030i | 0.1318 - 2.2030i | 0.1318 - 2.2030i | 0.1318 - 2.2030i | 0.1318 - 2.2030i |
| 0.1318 + 2.2030i | 0.1318 + 2.2030i | 0.1318 + 2.2030i | 0.1318 + 2.2030i | 0.1318 + 2.2030i | 0.1318 + 2.2030i |
| 0.1132 - 2.2214i | 0.1132 - 2.2214i | 0.1132 - 2.2214i | 0.1132 - 2.2214i | 0.1132 - 2.2214i | 0.1132 - 2.2214i |
| 0.1132 + 2.2214i | 0.1132 + 2.2214i | 0.1132 + 2.2214i | 0.1132 + 2.2214i | 0.1132 + 2.2214i | 0.1132 + 2.2214i |
| 0.0979 - 2.2365i | 0.0979 - 2.2365i | 0.0979 - 2.2365i | 0.0979 - 2.2365i | 0.0979 - 2.2365i | 0.0979 - 2.2365i |
| 0.0979 + 2.2365i | 0.0979 + 2.2365i | 0.0979 + 2.2365i | 0.0979 + 2.2365i | 0.0979 + 2.2365i | 0.0979 + 2.2365i |
| 0.0860 - 2.2483i | 0.0860 - 2.2483i | 0.0860 - 2.2483i | 0.0860 - 2.2483i | 0.0860 - 2.2483i | 0.0860 - 2.2483i |
| 0.0860 + 2.2483i | 0.0860 + 2.2483i | 0.0860 + 2.2483i | 0.0860 + 2.2483i | 0.0860 + 2.2483i | 0.0860 + 2.2483i |
| 0.0775 - 2.2567i | 0.0775 - 2.2567i | 0.0775 - 2.2567i | 0.0775 - 2.2567i | 0.0775 - 2.2567i | 0.0775 - 2.2567i |
| 0.0775 + 2.2567i | 0.0775 + 2.2567i | 0.0775 + 2.2567i | 0.0775 + 2.2567i | 0.0775 + 2.2567i | 0.0775 + 2.2567i |
| 0.0724 - 2.2618i | 0.0724 - 2.2618i | 0.0724 - 2.2618i | 0.0724 - 2.2618i | 0.0724 - 2.2618i | 0.0724 - 2.2618i |
| 0.0724 + 2.2618i | 0.0724 + 2.2618i | 0.0724 + 2.2618i | 0.0724 + 2.2618i | 0.0724 + 2.2618i | 0.0724 + 2.2618i |
| **Tiempo: 1.5806e+03 secs** | | **Tiempo: 2.1048e+03 secs** | | **Tiempo: 8.2478e+03 secs** | |

## Cuadro comparativo

# Bibliografía

* Teorema de Laplace, Wikipedia. (<https://es.wikipedia.org/wiki/Teorema_de_Laplace>)
* Descomposición QR (<http://people.inf.ethz.ch/arbenz/ewp/Lnotes/chapter3.pdf>)
* Single Shift y Double Shift (<http://www.math.usm.edu/lambers/mat610/sum10/lecture16.pdf>)
* Householder (http://www.cs.cornell.edu/~bindel/class/cs6210-f09/lec18.pdf)