Assignment 08 Answer

January 2, 2024

1 Assignment 8 Answer

- There are some lecture references in this document. They are notes to myself and can be ignored.

[ ]:
```python
from scipy.integrate import odeint
import numpy as np
import sympy as sp
import matplotlib.pyplot as plt
# pip install phaseportrait
import phaseportrait
```

1.1 Q1

1.1.1 1

$A = \begin{bmatrix} 5 & 1 & 0 \\ 0 & 5 & 1 \\ 0 & 0 & 5 \end{bmatrix}$

$\lambda_1 = 5$

$(A - \lambda_1 I)x = \left(\begin{bmatrix} 5 & 1 & 0 \\ 0 & 5 & 1 \\ 0 & 0 & 5 \end{bmatrix} - \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ 0 \end{bmatrix}$

The kernel is a line: $ker(A - \lambda_1 I) = \begin{bmatrix} \mathbb{R} \\ 0 \\ 0 \end{bmatrix}$

$(A - \lambda_1 I)^2 x = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_3 \\ 0 \\ 0 \end{bmatrix}$

The kernel is a plane that also contains the line: $ker(A - \lambda_1 I)^2 = \begin{bmatrix} \mathbb{R} \\ \mathbb{R} \\ 0 \end{bmatrix}$

$(A - \lambda_1 I)^3 x = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

The kernel is the entire space: $ker(A-\lambda_1 I)^3 = \begin{bmatrix}\mathbb{R}\\\mathbb{R}\\\mathbb{R}\end{bmatrix}$

$(A-\lambda_1 I)^3 = (A-\lambda_1 I)^4 = \dots = (A-\lambda_1 I)^n = \begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\end{bmatrix}$

The maximum dimension of the kernel is 3, which is the dimension of A.

1.1.2 2

$$\dot{x} = \begin{bmatrix}5 & 1 & 0\\0 & 5 & 1\\0 & 0 & 5\end{bmatrix}\begin{bmatrix}\mathbb{R}\\0\\0\end{bmatrix} = \begin{bmatrix}5\mathbb{R}\\0\\0\end{bmatrix} = \begin{bmatrix}\mathbb{R}\\0\\0\end{bmatrix}$$

$$\dot{x} = \begin{bmatrix}5 & 1 & 0\\0 & 5 & 1\\0 & 0 & 5\end{bmatrix}\begin{bmatrix}\mathbb{R}\\\mathbb{R}\\0\end{bmatrix} = \begin{bmatrix}5\mathbb{R}+\mathbb{R}\\5\mathbb{R}+\mathbb{R}\\0\end{bmatrix} = \begin{bmatrix}\mathbb{R}\\\mathbb{R}\\0\end{bmatrix}$$

$$\dot{x} = \begin{bmatrix}5 & 1 & 0\\0 & 5 & 1\\0 & 0 & 5\end{bmatrix}\begin{bmatrix}\mathbb{R}\\\mathbb{R}\\\mathbb{R}\end{bmatrix} = \begin{bmatrix}5\mathbb{R}+\mathbb{R}\\5\mathbb{R}+\mathbb{R}\\5\mathbb{R}\end{bmatrix} = \begin{bmatrix}\mathbb{R}\\\mathbb{R}\\\mathbb{R}\end{bmatrix}$$

All the subspaces are invariant for the system $\dot{x} = Ax$.

This means that starting in an eigenspace of A, the free evolution will stay in that eigenspace.

This can also be done in python:

[ ]:
```python
# Represents the sum of the kernels of the given matrix
# So the kernel [[1, 0, 0], [0, 1, 0]] would become [1, 1, 0]
# If kernel(A) contains entries bigger than one, check with nullspace()
# to make sure that information was not lost.
def kernel(matrix):
    subspace_list = sp.Matrix(matrix).nullspace()
    if len(subspace_list) == 0:
        return sp.Matrix([0])''
    subspace = sp.Matrix(subspace_list[0])
    for i in range(1,len(subspace_list)):
        subspace += sp.Matrix(subspace_list[i])
    return subspace

def dim(kernel):
    return len([x for x in kernel if x != 0])
```

[ ]:
```python
A = sp.Matrix([
    [5, 1, 0],
    [0, 5, 1],
    [0, 0, 5],
])
print('A = ')
```

```
A
```

A =

[ ]:
$$\begin{bmatrix}5 & 1 & 0\\0 & 5 & 1\\0 & 0 & 5\end{bmatrix}$$

[ ]:
```
I = sp.eye(3)
print('I = ')
I
```

I =

[ ]:
$$\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}$$

[ ]:
```
kernel(A-5*I)
```

[ ]:
$$\begin{bmatrix}1\\0\\0\end{bmatrix}$$

[ ]:
```
dim(kernel(A-5*I))
```

[ ]: 1

[ ]:
```
kernel((A-5*I)**2)
```

[ ]:
$$\begin{bmatrix}1\\1\\0\end{bmatrix}$$

[ ]:
```
dim(kernel((A-5*I)**2))
```

[ ]: 2

[ ]:
```
kernel((A-5*I)**3)
```

[ ]:
$$\begin{bmatrix}1\\1\\1\end{bmatrix}$$

[ ]:
```
dim(kernel((A-5*I)**3))
```

[ ]: 3

[ ]:
```
kernel((A-5*I)**4)
```

[ ]:

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

[ ]:
```
dim(kernel((A-5*I)**4))
```

[ ]: 3

[ ]:
```
dim(kernel((A-5*I)**1)) < dim(kernel((A-5*I)**2)) < dim(kernel((A-5*I)**3)) ==␣
↪dim(kernel((A-5*I)**4))
```

[ ]: True

1.2 Q2

1.2.1 1

Since the matrix is massive and the process is mostly the same i will use python to generate the kernels and instead comment on what is different. I will also display the kernels as row vectors to save vertical space. They will have the form $[x_1, x_2, ..., x_n]$

[ ]:
```
A = sp.Matrix(
    [
        [5, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 5, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 5, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 5, 1, 0, 0, 0, 0],
        [0, 0, 0, 0, 5, 1, 0, 0, 0],
        [0, 0, 0, 0, 0, 5, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 4, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 4, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 4],
    ]
)
print('A = ')
A
```

A =

[ ]:
$$\begin{bmatrix}5&0&0&0&0&0&0&0&0\\0&5&1&0&0&0&0&0&0\\0&0&5&0&0&0&0&0&0\\0&0&0&5&1&0&0&0&0\\0&0&0&0&5&1&0&0&0\\0&0&0&0&0&5&0&0&0\\0&0&0&0&0&0&4&0&0\\0&0&0&0&0&0&0&4&1\\0&0&0&0&0&0&0&0&4\end{bmatrix}$$

[ ]:
```
lambda1 = 5
lambda2 = 4
I = sp.eye(9)
```

[ ]:
```
A-lambda1*I
```

[ ]:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

[ ]:
```
kernel(A-lambda1*I).T
```

[ ]:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Proof that this is a $ker(A - \lambda_1 I)$:

$M \times ker(M) = 0$

[ ]:
```
M = (A - lambda1*I)
(M@kernel(M)).T
```

[ ]:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

[ ]:
```
dim(kernel(A-lambda1*I))
```

[ ]: 3

For $(A - \lambda_1 I)^2$:

[ ]:
```
(A-lambda1*I)**2
```

[ ]:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

[ ]:
```
kernel((A-lambda1*I)**2).T
```

[ ]:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

[ ]:
```
dim(kernel((A-lambda1*I)**2))
```

[ ]: 5

For $(A - \lambda_1 I)^3$:

[ ]:
```
(A-lambda1*I)**3
```

[ ]:
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

[ ]:
```
kernel((A-lambda1*I)**3).T
```

[ ]:
$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

[ ]:
```
dim(kernel((A-lambda1*I)**3))
```

[ ]: 6

For $(A - \lambda_1 I)^4$:

[ ]:
```
(A-lambda1*I)**4
```

[ ]:
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

[ ]:
```
kernel((A-lambda1*I)**4).T
```

[ ]:
$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

[ ]:
```
dim(kernel((A-lambda1*I)**4))
```

[ ]: 6

Printing the kernels next to eachother to compare them:

[ ]:
```
kernel((A-lambda1*I)**1).T
```

[ ]: $\begin{bmatrix}1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
kernel((A-lambda1*I)**2).T
```

[ ]: $\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
kernel((A-lambda1*I)**3).T
```

[ ]: $\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
kernel((A-lambda1*I)**4).T
```

[ ]: $\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
dim(kernel((A-5*I)**1)) < dim(kernel((A-5*I)**2)) < dim(kernel((A-5*I)**3)) ==␣
  ↪dim(kernel((A-5*I)**4))
```

[ ]: True

A contains 3 subsystems for $\lambda_1 = 5$. The kernel stops growing at $n = 3$ for $(A - \lambda_1 I)^n$.

The minimal polynomial for A is $(s-5)^3(s-4)^2$.
The multiplicity of the eigenvalue 5 is 3.

[ ]:
```
def characteristic(matrix):
    I = sp.eye(matrix.shape[0])
    M = matrix - sp.symbols('s')*I
    return M.det()
```

[ ]:
```
characteristic(A)
```

[ ]: $(4-s)^3\,(5-s)^6$

The characteristic polynomial shows the biggest dimension the kernel assosiated with λ_1 can have is 6. This corresponds with what we found above. After this, the kernel stops growing and stays at dimension 6.

1.2.2 2

When printing the kernels next to eachother in Q1, it is clear that $ker(M^n)$ is nested into $ker(M^{n+1})$ for all $n > 1$ where $M = A - \lambda_1 I$.

Is the subspace $ker(A - \lambda_1 I)$ invariant for the system $\dot{x} = Ax$?

[ ]:
```
kernel(A - lambda1*I).T
```

[ ]: $\begin{bmatrix}1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
(A@kernel(A - lambda1*I)).T
```

[ ]: $\begin{bmatrix}5 & 5 & 0 & 5 & 0 & 0 & 0 & 0 & 0\end{bmatrix}$

Yes! The derivatives of any initial condition that falls within the subsystem $ker(A-\lambda_1 I)$ will also fall within the subsystem $ker(A-\lambda_1 I)$.

Is the subspace $ker(A-\lambda_1 I)^2$ invariant for the system $\dot{x} = Ax$?

[ ]:
```
kernel((A - lambda1*I)**2).T
```

[ ]: $\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
(A@kernel((A - lambda1*I)**2)).T
```

[ ]: $\begin{bmatrix}5 & 6 & 5 & 6 & 5 & 0 & 0 & 0 & 0\end{bmatrix}$

Yes!

Is the subspace $ker(A-\lambda_1 I)^3$ invariant for the system $\dot{x} = Ax$?

[ ]:
```
kernel((A - lambda1*I)**3).T
```

[ ]: $\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
(A@kernel((A - lambda1*I)**3)).T
```

[ ]: $\begin{bmatrix}5 & 6 & 5 & 6 & 6 & 5 & 0 & 0 & 0\end{bmatrix}$

Yes!

All the subsystems are invariant for the system $\dot{x} = Ax$.

Lets confirm this for $\lambda_2 = 4$ as well:

[ ]:
```
kernel(A - lambda2*I).T
```

[ ]: $\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0\end{bmatrix}$

[ ]:
```
kernel((A - lambda2*I)**2).T
```

[ ]: $\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1\end{bmatrix}$

[ ]:
```
kernel((A - lambda2*I)**3).T
```

[ ]: $\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1\end{bmatrix}$

[ ]:
```
dim(kernel((A - lambda2*I)**1)) < dim(kernel((A - lambda2*I)**2)) ==
 ↪dim(kernel((A - lambda2*I)**3))
```

[ ]: `True`

The same conclusions from λ_1 can also be drawn for λ_2.

1.3 Q3

- Lecture 201 Towards generalized eigenspaces
- Lecture 206 diagonalizability

[ ]:
```
A = sp.Matrix([[5, 1, -2, 4], [0, 5, 2, 2], [0, 0, 5, 3], [0, 0, 0, 4]])
A
```

[ ]:

$$\begin{bmatrix} 5 & 1 & -2 & 4 \\ 0 & 5 & 2 & 2 \\ 0 & 0 & 5 & 3 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

This is an upper triangular matrix. The eigenvalues are the diagonal entries. This is confirmed by the characteristic polynomial:

[ ]:
```
characteristic(A)
```

[ ]:

$$(4 - s)(5 - s)^3$$

[ ]:
```
lambda1 = 4
lambda2 = 5
```

[ ]:
```
I = sp.eye(4)
```

A is not on Jordan form. Finding the Jordan form of A:

[ ]:
```
v1 = kernel(A-lambda1*I)
v1.T
```

[ ]:

$$\begin{bmatrix} -14 & 4 & -3 & 1 \end{bmatrix}$$

Due to the implementation of the kernel() function, lets check the proper kernel to make sure that no information was lost:

[ ]:
```
sp.Matrix((A-lambda1*I).nullspace()).T
```

[ ]:

$$\begin{bmatrix} -14 & 4 & -3 & 1 \end{bmatrix}$$

Good! The kernel is a single vector, and no information was lost.

[ ]:
```
M = A - lambda2*I
```

[ ]:
```
v2 = kernel(M)
v2.T
```

[ ]:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

$Mv_3 = v_2$

[ ]:
```
v3 = M.gauss_jordan_solve(v2)[0].subs({"tau0": 0})
v3.T
```

[ ]: $\begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}$

[ ]:
```
v4 = M.gauss_jordan_solve(v3)[0].subs({"tau0": 0})
v4.T
```

[ ]: $\begin{bmatrix} 0 & 1 & \frac{1}{2} & 0 \end{bmatrix}$

[ ]:
```
T = sp.Matrix([v1, v2, v3, v4]).reshape(4, 4).T
T
```

[ ]:
$$\begin{bmatrix} -14 & 1 & 0 & 0 \\ 4 & 0 & 1 & 1 \\ -3 & 0 & 0 & \frac{1}{2} \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

[ ]:
```
J = T.inv()@A@T
J
```

[ ]:
$$\begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 5 & 1 & 0 \\ 0 & 0 & 5 & 1 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Double checking the answer:

[ ]:
```
A.jordan_form()[1]
```

[ ]:
$$\begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 5 & 1 & 0 \\ 0 & 0 & 5 & 1 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Analysis:

Same procedure as in Q2:

For $\lambda_1 = 4$:

[ ]:
```
characteristic(J)
```

[ ]: $(4 - s)(5 - s)^3$

The characteristic polynomial indicates that the kernel assosiated with $\lambda_1 = 4$ can have a maximum dimension of 1.

[ ]:
```
J - lambda1*I
```

[ ]:
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

[ ]:
```
kernel(J - lambda1*I).T
```

[ ]: $\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
kernel((J - lambda1*I)**2).T
```

[ ]: $\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}$

[ ]:
```
dim(kernel((A - lambda1*I)**1)) == dim(kernel((A - lambda1*I)**2))
```

[ ]: True

For $\lambda_2 = 5$:

[ ]:
```
characteristic(J)
```

[ ]: $(4 - s)(5 - s)^3$

The characteristic polynomial indicates that the kernel assosiated with $\lambda_2 = 5$ can have a maximum dimension of 3.

[ ]:
```
J - lambda2*I
```

[ ]:

$$\begin{bmatrix}-1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0\end{bmatrix}$$

[ ]:
```
kernel(J - lambda2*I).T
```

[ ]: $\begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix}$

[ ]:
```
kernel((J - lambda2*I)**2).T
```

[ ]: $\begin{bmatrix}0 & 1 & 1 & 0\end{bmatrix}$

[ ]:
```
kernel((J - lambda2*I)**3).T
```

[ ]: $\begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix}$

[ ]:
```
kernel((J - lambda2*I)**4).T
```

[ ]: $\begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix}$

[ ]:
```
M = (J - lambda2*I)
dim(kernel(M**1)) < dim(kernel(M**2)) < dim(kernel(M**3)) == dim(kernel(M**4))
```

[ ]: True

The same conclusions from Q2 can be drawn for Q3, except in Q3 A had to be transformed into its Jordan form J.