Tarea 3

Iván Vega Gutiérrez

30 de septiembre de 2021

Ejercicio 1. Considera el sistema de ecuaciones lineales.

$$\begin{pmatrix} 3 & -2 & 1 \\ 1 & 1 & -2 \\ 3 & 5 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 5 \end{pmatrix}$$

esto es Ax=b.

Usando los comandos del lenguaje de programación de su preferencia obtenga:

1. La traza de la matriz **A**.
2. El determinante de la matriz **A**.
3. Los valores y vectores impropio de **A**.
4. El kernel y nulidad de **A**.
5. El rango de **A**. Verificar que $\text{rango}(\mathbf{A}) + \text{nulidad}(\mathbf{A}) = n$.
6. Resolver (si es posible) el sistema **Ax = b** usando la inversa de la matriz.

Solución Primero importamos la librería Numpy y Scipy para poder trabajar con matrices y utilizar los distintos métodos que necesitaremos.

[1]:
```
import numpy as np
from scipy.linalg import null_space
A = np.array([[3, -2, 1], [1, 1, -2], [3, 5, -1]])
b = np.array([1, 2, 5])
```

1. La traza de la matriz A

[2]:
```
Traza_A = A.trace()
print("La traza de la matriz A es {}".format(Traza_A))
```

```
La traza de la matriz A es 3
```

2. El determinante de la matriz A

[3]:
```
Determinante_A = np.linalg.det(A)
print("El determinante de la matriz A es {}".format(Determinante_A))
```

```
El determinante de la matriz A es 38.99999999999999
```

3. Los valores y vectores propios de A

[4]:
```
valores_propios, vectores_propios = np.linalg.eig(A)
print("Los valores propios de la matriz A son \n{} ".format(valores_propios))
print("Los vectores propios de la matriz A son \n{}".format(vectores_propios))
```

```
Los valores propios de la matriz A son
[ 3.6931754+0.j         -0.3465877+3.23108268j -0.3465877-3.23108268j]
Los vectores propios de la matriz A son
[[ 0.86480376+0.j          0.19889163-0.09211908j  0.19889163+0.09211908j]
 [-0.04991915+0.j         -0.22666322-0.47545995j -0.22666322+0.47545995j]
 [ 0.49962239+0.j         -0.82129034+0.j         -0.82129034-0.j        ]]
```

4. El kernel y nulidad de A

[5]:
```
Kernel_A = null_space(A)
Nulidad_A = len(Kernel_A[0])
print("El Kernel de A es \n {} \n y la nulidad es {}".format(Kernel_A,␣
 ↪Nulidad_A))
```

```
El Kernel de A es
 []
 y la nulidad es 0
```

5. El rango de A. Verificar que rango(A) + nulidad(A) = n

[6]:
```
Rango_A = np.linalg.matrix_rank(A)
print("El rango de la matriz A  es {}".format(Rango_A))
print("Tenemos que rango(A) + nulidad(A) = {}".format(Rango_A + Nulidad_A))
```

```
El rango de la matriz A  es 3
Tenemos que rango(A) + nulidad(A) = 3
```

6. Resolver (si es posible) el sistema Ax = b usando la inversa de la matriz.

[7]:
```
sol = np.linalg.solve(A,b)
print("La solución del sistema dado es \n {}".format(sol))
```

```
La solución del sistema dado es
 [ 0.76923077  0.46153846 -0.38461538]
```

Ejercicio 2 Repetir el Ejercicio 1 con el sistema de ecuaciones lineales

$$\begin{pmatrix} 1 & 2 & -5 \\ 2 & 4 & -10 \\ 2 & 2 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 10 \\ 5 \\ 20 \end{pmatrix}$$

Para resolver el ejercicio 2, tenemos lo siguiente.

[8]:
```
A = np.array([[1, 2, -5], [2, 4, -10], [2, 2, 5]])
b = np.array([10, 5, 20])
```

1. La traza de la matriz A.

[9]:
```
Traza_A = A.trace()
print("La traza de la matriz A es {}".format(Traza_A))
```

```
La traza de la matriz A es 10
```

2. El determinante de la matriz A.

[10]:
```
Determinante_A = np.linalg.det(A)
print("El determinante de la matriz A es {}".format(Determinante_A))
```

```
El determinante de la matriz A es 0.0
```

3. Los valores y vectores propios de A.

[11]:
```
valores_propios, vectores_propios = np.linalg.eig(A)
print("Los valores propios de la matriz A son \n{} ".format(valores_propios))
print("Los vectores propios de la matriz A son \n{}".format(vectores_propios))
```

```
Los valores propios de la matriz A son
[-2.23568936e-16+0.j          5.00000000e+00+5.47722558j
  5.00000000e+00-5.47722558j]
Los vectores propios de la matriz A son
[[-7.97452223e-01+0.00000000e+00j  4.01609664e-01-6.71103486e-17j
   4.01609664e-01+6.71103486e-17j]
 [ 5.98089167e-01+0.00000000e+00j  8.03219329e-01+0.00000000e+00j
   8.03219329e-01-0.00000000e+00j]
 [ 7.97452223e-02+0.00000000e+00j -3.92357903e-16-4.39941345e-01j
  -3.92357903e-16+4.39941345e-01j]]
```

4. El kernel y nulidad de A.

[12]:
```
Kernel_A = null_space(A)
Nulidad_A = len(Kernel_A[0])
print("El Kernel de A es \n {} \n y la nulidad es {}".format(Kernel_A,␣
 ↪Nulidad_A))
```

```
El Kernel de A es
 [[-0.79745222]
 [ 0.59808917]
 [ 0.07974522]]
 y la nulidad es 1
```

5. El rango de A. Verificar que rango(A) + nulidad(A) = n

[13]:
```
Rango_A = np.linalg.matrix_rank(A)
print("El rango de la matriz A  es {}".format(Rango_A))
print("Tenemos que rango(A) + nulidad(A) = {}".format(Rango_A + Nulidad_A))
```

```
El rango de la matriz A  es 2
Tenemos que rango(A) + nulidad(A) = 3
```

6. Resolver (si es posible) el sistema Ax = b usando la inversa de la matriz.

[14]:
```
sol = np.linalg.solve(A,b)
print("La solución del sistema dado es \n {}".format(sol))
```

```
---------------------------------------------------------------------------
LinAlgError                               Traceback (most recent call last)
<ipython-input-14-5a38e7050bfe> in <module>
----> 1 sol = np.linalg.solve(A,b)
      2 print("La solución del sistema dado es \n {}".format(sol))

<__array_function__ internals> in solve(*args, **kwargs)

~/opt/anaconda3/lib/python3.8/site-packages/numpy/linalg/linalg.py in solve(a, b)
    391     signature = 'DD->D' if isComplexType(t) else 'dd->d'
    392     extobj = get_linalg_error_extobj(_raise_linalgerror_singular)
--> 393     r = gufunc(a, b, signature=signature, extobj=extobj)
    394
    395     return wrap(r.astype(result_t, copy=False))

~/opt/anaconda3/lib/python3.8/site-packages/numpy/linalg/linalg.py in␣
 ↪_raise_linalgerror_singular(err, flag)
     86
     87 def _raise_linalgerror_singular(err, flag):
---> 88     raise LinAlgError("Singular matrix")
     89
     90 def _raise_linalgerror_nonposdef(err, flag):

LinAlgError: Singular matrix
```

Al intentar resolver el sistema nos arroja un error, el cual nos dice que se trata de una matriz singular, por tanto no es posible hallar una solución, lo cual nos podemos percatar desde el momento en que el determinante es cero.

Ejercicio 3 Explicar en que consiste el sistema de aritmética de punto flotante IEEE.

Solución. El sistema de aritmética de punto flotante surgió en 1985, las siglas se deben a Institute for Electrical and Electronic Engineers, y fue este insitituto quién publicó un informe llamado Binary Floating Point Arithmetic Standard 754-1985. El informe proporcionaba las normas para los números binarios y con punto decimal flotante, formatos para el intercambio de datos, algoritmos para redondear operaciones aritméticas y para el manejo de excepciones.

La aritmética de punto flotante es la aritmética que utiliza la computadora, al igual que las representaciones de números reales, la aritmética que utiliza la computadora no es exacta. La aritmética de punto flotante consiste en una aritmética con respecto a la representación punto

flotante de los número que se desean operar. Dados dos valores x e y, si queremos sumarlos, lo primero que hará la computadora será pasar los números x e y en su representación de punto flotante, ya que es la forma en la que opera. Posteriormente, sumará dichas representaciones y esa suma la volverá a convertir en su forma de punto flotante. Es decir

$$x\tilde{+}y = fl(fl(x) + fl(y))$$

donde $\tilde{+}$ denota la suma que realiza la computadora. Notemos que esta aritmética conlleva dos errores, el primero es cuando se hacen las representaciones de x e y y el siguiente error es cuando hace la representación de la suma. Sin embargo el error de la aritmética de punto flotante está acotado por un número muy pequeño que se llama presición de la máquina y se denota por $\epsilon_{machine}$, se tiene que

$$\left|\frac{(x\tilde{+}y) - (x+y}{x+y}\right| \leq \epsilon_{machine}$$

Por lo tanto, los cálculos realizados son buenas aproximaciones.

Ejercicio 4 Realiza el ejercicio 5.1 pag. 92 de libro Numerical Linear Alge- bra. G. Allaire, S. M. Kaber.
Ejercicio 5.1. Compile el siguiente código y comente los resultados.

1. Precisión de punto flotante (presición de la máquina).

```
a=eps;b=0.5*eps;X=[2, 1;2, 1];
A=[2, 1;2, 1+a];norm(A–X)
B=[2, 1;2, 1+b];norm(X–B)
```

2. Cotas de los números punto-flotante.

```
rM=realmax, 1.0001*rM, rm=realmin, .0001*rm
```

3. Infinity y "Not a number".

```
A=[1 2 0 3]; B=1./A, isinf(B), C=A.*B
```

4. ¿Singular o no?

```
A=[1 1; 1 1+eps];inv(A), rank(A)
B=[1 1; 1 1+.5*eps];inv(B), rank(B)
```

Solución El código se compiló en el software de Matlab 2016.

1. Para empezar, tenemos que *eps* es una constante que vale $2{,}2204e-16$, es decir un número muy pequeño, por lo tanto la variable b es un número extremadamente pequeño.
   Por como están definidas las matices A, B y X, sus componentes solo difieren en la entrada (2,2).
   El resultado de la norma de A menos X es

   $$2{,}2204e-16$$

   Mientras que la norma de X menos B es

   $$0$$

   Intuitivamente uno pensaría que los resultados deberían ser iguales, sin embargo como $b < \epsilon_{machine}$, se tiene que $1+b=1$, para la computadora la matriz B es igual a la matriz X, porque prácticamente estamos sumando cero.

2. realmax devuelve el número de punto flotante finito más grande en IEEE® de doble precisión y realmin devuelve el número de punto flotante normalizado positivo más pequeño en IEEE® de doble precisión. Sus valores son

$$rM = 1{,}7977e + 308$$

y

$$rm = 2{,}2251e - 308$$

Al tratar de calular 1.00001*rM el resultado es Inf, que nos representa infinito, lo cual es bastante razonable. Sin embargo, cuando se calcula .0001*rm el resultado es 2.2251e-312.

3. Cuando se calcula B el resultado es el siguiente

$$[1{,}0000, 0{,}5000, Inf, 0{,}3333]$$

Prácticamente se saca el inverso multiplicativo de cada entrada del vector A, sin embargo la tercer entrada es 0, por lo tanto la computadora interpreta $\frac{1}{0}$ como infinito. Posteriormente se hace la comparación lógica isinf(B)

$$[0, 0, 1, 0]$$

Donde uno representa verdadero y cero falso, lo cual es cierto, ya que la tercera entrada es infinito. Por último cuando se hace el producto de A por B, el resultado es

$$[1, 1, NaN, 1]$$

Donde NaN, nos representa Not a Number, esto sucede porque queremos multiplicar cero por infinito, pero infinito no es un número, por eso el motivo del resultado.

4. Dada la matriz A, nos preguntamos si es invertible, cuando intentamos hallar la inversa de la matriz A nos aparece el siguiente mensaje:
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 5.551115e-17.

$$1{,}0e + 15 * \begin{pmatrix} 4{,}5036 & -4{,}5036 \\ -4{,}5036 & 4{,}5036 \end{pmatrix}$$

Definamos a la matriz C como

$$C = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Notemos que la matriz A es casi igual a la matriz C y sabemos que det(C) = 0, por lo tanto C no es invertible, sin embargo, la computadora toma esta diferencia de manera significativa y es capaz de diferenciarla de la matriz C y dar un resultado para inv(A), no obstante en el mensaje que está en rojo nos advierte de un posible resultado erróneo.

Por otro lado, cuando intentamos hallar inv(B), nos da un resultado con entradas Inf, al igual que si queremos hallar inv(C). Esto tiene que ver con el primer inciso, esto se debe a que la entrada (2,2) de la matriz A lo toma como 1, es decir 0.5eps no lo reconoce como tal, por lo tanto para la computadora A=C.