```
import numpy as np
import matplotlib.pyplot as plt
```

In [20]:

```
# creamos un DF

# creamos un DF con numpy

df = pd.DataFrame(np.array([[7,6.5,9.2,8.6,8],
                            [7.5,9.4,7.3,7,7],
                            [7.6,9.2,8,8,7.5],
                            [5,6.5,6.5,7,9],
                            [6,6,7.8,8.9,7.3],
                            [7.8,9.6,7.7,8,6.5],
                            [6.3,6.4,8.2,9,7.2],
                            [7.9,9.7,7.5,8,6],
                            [6,6,6.5,5.5,8.7],
                            [6.8,7.2,8.7,9,7]]),

                 index = ['Lucia','Pedro','Ines','Luis','Andres','Ana','Carlos','Jose','Sonia','Maria'],
                 columns = ['Matematicas','Ciencias','Español','Historia','EFisica'])

df
```

Out[20]:

	Matematicas	Ciencias	Español	Historia	EFisica
Lucia	7.0	6.5	9.2	8.6	8.0
Pedro	7.5	9.4	7.3	7.0	7.0
Ines	7.6	9.2	8.0	8.0	7.5
Luis	5.0	6.5	6.5	7.0	9.0
Andres	6.0	6.0	7.8	8.9	7.3
Ana	7.8	9.6	7.7	8.0	6.5
Carlos	6.3	6.4	8.2	9.0	7.2
Jose	7.9	9.7	7.5	8.0	6.0
Sonia	6.0	6.0	6.5	5.5	8.7
Maria	6.8	7.2	8.7	9.0	7.0

In [23]:
```
label = df.index.tolist()
label
```
Out[23]:
```
['Lucia',
 'Pedro',
 'Ines',
 'Luis',
 'Andres',
 'Ana',
 'Carlos',
 'Jose',
 'Sonia',
 'Maria']
```
In [27]:
```
from numpy import array
from sklearn.decomposition import TruncatedSVD

A = df
A
```
Out[27]:

	Matematicas	Ciencias	Español	Historia	EFisica
Lucia	7.0	6.5	9.2	8.6	8.0
Pedro	7.5	9.4	7.3	7.0	7.0
Ines	7.6	9.2	8.0	8.0	7.5
Luis	5.0	6.5	6.5	7.0	9.0
Andres	6.0	6.0	7.8	8.9	7.3
Ana	7.8	9.6	7.7	8.0	6.5

In [28]:
```
from numpy import diag
from numpy import zeros
from scipy.linalg import svd
```
In [29]:
```
U, s, VT = svd(A)
Sigma = zeros((A.shape[0], A.shape[1]))
Sigma[:A.shape[1],:A.shape[1]] = diag(s)
```
In [30]:
```
print('Matriz U:')
print(U)
```
```
Matriz U:
[[-3.30904778e-01 -2.98442471e-01  1.95957496e-01  5.49129984e-01
  -3.41498230e-01  3.14150349e-02 -1.23395489e-01  1.44668847e-01
  -3.94701586e-01 -3.89792548e-01]
 [-3.20798129e-01  3.55909651e-01 -2.40201517e-01  6.92729616e-02
  -2.04368756e-01 -4.82498850e-01  2.12566116e-01 -5.29979923e-01
  -2.91960519e-01  1.48477689e-01]
 [-3.38871518e-01  2.14752307e-01 -1.01355072e-01 -1.12970083e-02
  -1.76844101e-01 -1.27934010e-01 -3.25968835e-01 -5.72499990e-02
   7.00049221e-01 -4.25915443e-01]
 [-2.86319969e-01 -3.47796853e-01 -4.56775340e-01 -6.26123724e-01
  -3.29136285e-01  7.19838459e-02 -5.36517108e-04  2.16809671e-01
  -1.85075729e-01 -4.01432467e-02]
```

In [31]:
```python
print('Matriz Sigma:')
print(Sigma)
```

```
Matriz Sigma:
[[53.21335049  0.          0.          0.          0.        ]
 [ 0.          5.35631448  0.          0.          0.        ]
 [ 0.          0.          3.80560674  0.          0.        ]
 [ 0.          0.          0.          1.46904724  0.        ]
 [ 0.          0.          0.          0.          0.47799818]
 [ 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.        ]]
```

In [32]:
```python
print('matriz VT:')
print(VT)
```

```
matriz VT:
[[-0.40556463 -0.45759727 -0.46123811 -0.47103299 -0.43761787]
 [ 0.30388113  0.70497323 -0.20845983 -0.2165757  -0.56595801]
 [-0.00426062 -0.27229676  0.31573346  0.5963283  -0.6859601 ]
 [ 0.56293162 -0.34606734  0.50555254 -0.5444058  -0.1066974 ]
 [ 0.65288853 -0.31576789 -0.62332408  0.28149203  0.07909827]]
```

In [33]:
```python
# recostruir la matriz original

B = U.dot(Sigma.dot(VT))
B
```

Out[33]:
```
array([[7. , 6.5, 9.2, 8.6, 8. ],
       [7.5, 9.4, 7.3, 7. , 7. ],
       [7.6, 9.2, 8. , 8. , 7.5],
       [5. , 6.5, 6.5, 7. , 9. ],
       [6. , 6. , 7.8, 8.9, 7.3],
       [7.8, 9.6, 7.7, 8. , 6.5],
       [6.3, 6.4, 8.2, 9. , 7.2],
       [7.9, 9.7, 7.5, 8. , 6. ],
       [6. , 6. , 6.5, 5.5, 8.7],
       [6.8, 7.2, 8.7, 9. , 7. ]])
```

In [34]:
```python
# reduccion a dos dimenciones

n_elements = 2
Ureduced2 = U[:, :n_elements]
print('U reducida:\n', Ureduced2)
```

```
U reducida:
 [[-0.33090478 -0.29844247]
 [-0.32079813  0.35590965]
 [-0.33887152  0.21475231]
 [-0.28631997 -0.34779685]
 [-0.3037477  -0.3046643 ]
```

In [35]:
```python
SigmaReduced2 = Sigma[:n_elements, :n_elements]
print('Matriz Sigma Reducida:\n', SigmaReduced2)
```

```
Matriz Sigma Reducida:
 [[53.21335049  0.        ]
 [ 0.          5.35631448]]
```

In [36]:
```python
VTReduced2 = VT[:n_elements, :]
print('Matriz VT Reducida:\n', VTReduced2)
```

```
Matriz VT Reducida:
 [[-0.40556463 -0.45759727 -0.46123811 -0.47103299 -0.43761787]
 [ 0.30388113  0.70497323 -0.20845983 -0.2165757  -0.56595801]]
```

In [37]:
```python
AReduced2 = Ureduced2.dot(SigmaReduced2.dot(VTReduced2))
print('Matriz A transformada:\n', AReduced2)
```

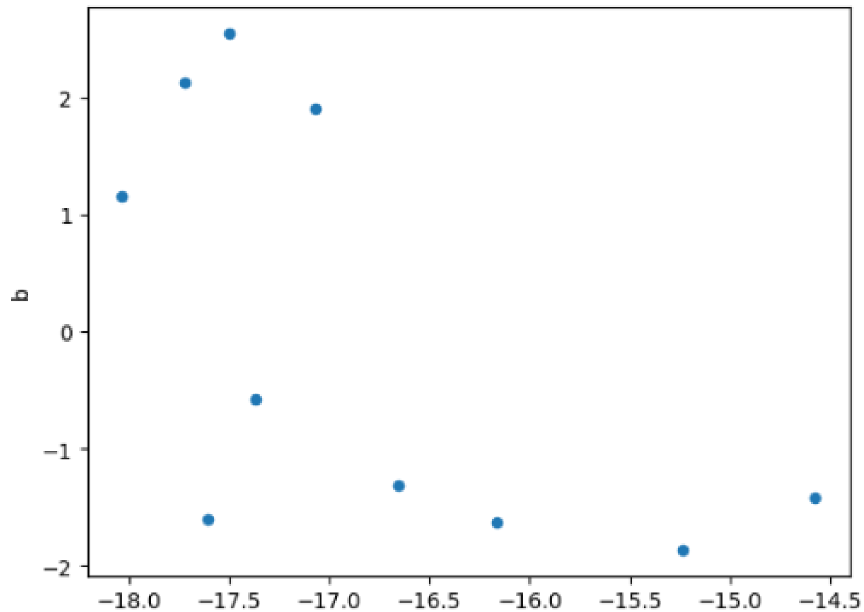
```
Matriz A transformada:
 [[6.65563617 6.93068907 8.45496902 8.64041633 8.61053016]
 [7.50259774 9.15546106 7.47627701 7.62801112 6.39154032]
 [7.66288835 9.06253486 8.0774837  8.24477427 7.24032869]
 [5.61309793 5.65867129 7.41578628 7.58014066 7.72189396]
 [6.05941975 6.24591247 7.7953721  7.96693512 7.99698132]
 [7.83156613 9.60455223 7.73118336 7.88753689 6.55416982]
 [6.35785122 6.70021013 7.95486409 8.12861624 8.02875678]
 [7.86973795 9.80171752 7.53881312 7.68951159 6.21497236]
 [5.48053658 5.67061811 7.01789357 7.17215282 7.18015541]
 [6.86687283 7.53702066 8.13213661 8.30697925 7.93006916]]
```


[6.86687283 7.53702066 8.13213661 8.30697925 7.93006916]]

In [38]:
```python
# obtenemos T2

T2 = Ureduced2.dot(SigmaReduced2)
print('Matriz T (2 dimensiones):\n', T2)
```

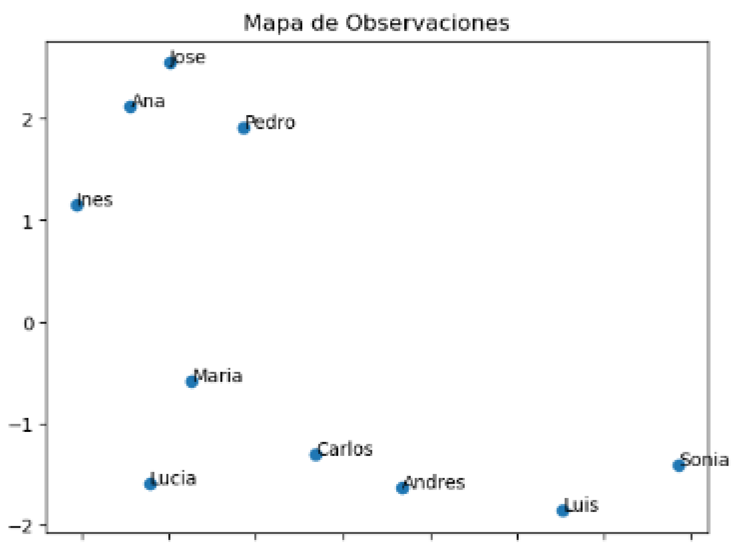
```
Matriz T (2 dimensiones):
 [[-17.60855193  -1.59855173]
 [-17.07074326   1.90636402]
 [-18.03248887   1.15028089]
 [-15.23604486  -1.86290932]
 [-16.1634327   -1.6318778 ]
 [-17.7206514    2.12154237]
 [-16.65597775  -1.30716988]
 [-17.49591103   2.54709878]
 [-14.57497601  -1.41686385]
 [-17.36793336  -0.58228904]]
```

In [39]:
```
# graficamos T2

df = pd.DataFrame(T2, columns = ['a','b'])
df.plot(kind = 'scatter', x = 'a', y = 'b')
plt.show()
```



In [40]:

```python
# colocamos etiquetas

df = pd.DataFrame(T2, columns = ['a','b'])
x = df.iloc[:,0]
y = df.iloc[:,1]

# convertimos x y y a arreglos np

x = x.to_numpy()
y = y.to_numpy()

fig, ax = plt.subplots()
ax.set_title('Mapa de Observaciones')
ax.scatter(x, y)

# agregamos las etiquetas

for i, txt in enumerate(label):
    ax.annotate(txt, (x[i], y[i]))
```



In [41]:
```python
# Hacemos una reduccion de U, Sigma, VT, a 3 dimesiones para mejorar la division de los grupos

# U
n_elements = 3
UReduced3 = U[:, :n_elements]
print('U Reducida:\n', UReduced3)

# Sigma
SigmaReduced3 = Sigma[:n_elements, :n_elements]
print('Matriz Sigma Reducida:\n', SigmaReduced3)

# VT
VTReduced3 = VT[:n_elements, :]
print('VT Reducida:\n', VTReduced3)

# A reducida

AReduced3 = UReduced3.dot(SigmaReduced3.dot(VTReduced3))
print('Matriz A Transformada:\n', AReduced3)
```

```
U Reducida:
 [[-0.33090478 -0.29844247  0.1959575 ]
 [-0.32079813  0.35590965 -0.24020152]
 [-0.33887152  0.21475231 -0.10135507]
 [-0.28631997 -0.34779685 -0.45677534]
 [-0.3037477  -0.3046643   0.28988541]
 [-0.33301138  0.39608249  0.02516069]
 [-0.31300374 -0.24404278  0.32780981]
 [-0.328788    0.47553197  0.09142545]
 [-0.27389698 -0.26452216 -0.60309016]
```

In [42]:
```python
T3 = UReduced3.dot(SigmaReduced3)
print('Matriz T (3 dimensiones):\n', T3)
```

```
Matriz T (3 dimensiones):
 [[-17.60855193  -1.59855173   0.74573717]
 [-17.07074326   1.90636402  -0.91411251]
 [-18.03248887   1.15028089  -0.38571754]
 [-15.23604486  -1.86290932  -1.73830731]
 [-16.1634327   -1.6318778    1.10318989]
 [-17.7206514    2.12154237   0.09575169]
 [-16.65597775  -1.30716988   1.24751522]
 [-17.49591103   2.54709878   0.34792931]
 [-14.57497601  -1.41686385  -2.29512399]
 [-17.36793336  -0.58228904   1.32260625]]
```

In [43]:
```python
# importamos libreria para grafica 3D

from mpl_toolkits import mplot3d
%matplotlib inline
```

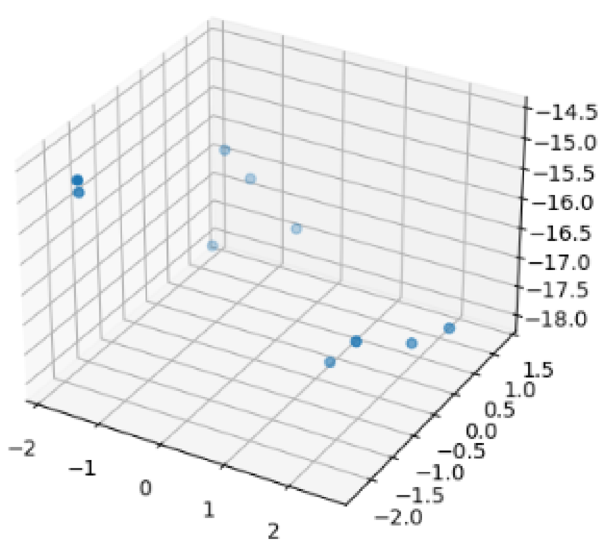
In [44]:
```python
# T3 lo convertimos en DF

df3 = pd.DataFrame(T3, columns = ['a','b','c'])
df3
```

Out[44]:

	a	b	c
0	-17.608552	-1.598552	0.745737
1	-17.070743	1.906364	-0.914113
2	-18.032489	1.150281	-0.385718
3	-15.236045	-1.862909	-1.738307
4	-16.163433	-1.631878	1.103190
5	-17.720651	2.121542	0.095752
6	-16.655978	-1.307170	1.247515
7	-17.495911	2.547099	0.347929
8	-14.574976	-1.416864	-2.295124
9	-17.367933	-0.582289	1.322606

In [45]:
```python
# grafica 3D

fig, plt.figure()
ax = plt.axes(projection = '3d')
xline = df3['b']
yline = df3['c']
zline = df3['a']
ax.scatter3D(xline, yline, zline)
```

Out[45]: <mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x1f60ada1d50>



In [46]:

```python
# calculamos los errores para 2 dimensiones

error2 = B - AReduced2

# convertimos a DF

dferror2 = pd.DataFrame(error2, columns = ['Matematicas','Ciencias','Español','Historia','EFisica'])
dferror2
```

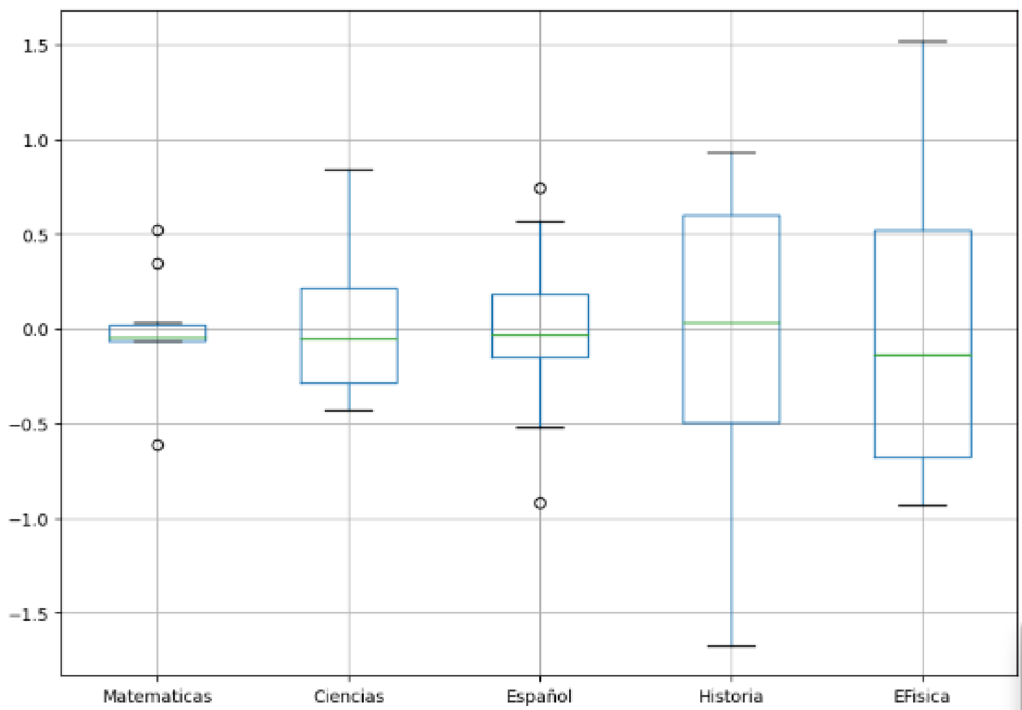
Out[46]:

	Matematicas	Ciencias	Español	Historia	EFisica
0	0.344364	-0.430689	0.745031	-0.040416	-0.610530
1	-0.002598	0.244539	-0.176277	-0.628011	0.608460
2	-0.062888	0.137465	-0.077484	-0.244774	0.259671
3	-0.613098	0.841329	-0.915786	-0.580141	1.278106
4	-0.059420	-0.245912	0.004628	0.933065	-0.696981
5	-0.031566	-0.004552	-0.031183	0.112463	-0.054170
6	-0.057851	-0.300210	0.245136	0.871384	-0.828757
7	0.030262	-0.101718	-0.038813	0.310488	-0.214972
8	0.519463	0.329382	-0.517894	-1.672153	1.519845
9	-0.066873	-0.337021	0.567863	0.693021	-0.930069

In [47]:

```python
# graficamos

plt.figure(figsize = (10,7))
dferror2.boxplot()
```

Out[47]: <Axes: >



In [48]:
```
# calculamos la media del error de cada columna(variable)

print('Errores promedio por columna')
error2.mean(axis = 0)
```

```
Errores promedio por columna
```

Out[48]:
```
array([-2.04650684e-05,  1.32612606e-02, -1.94778869e-02, -2.45074289e-02,
        3.30602005e-02])
```

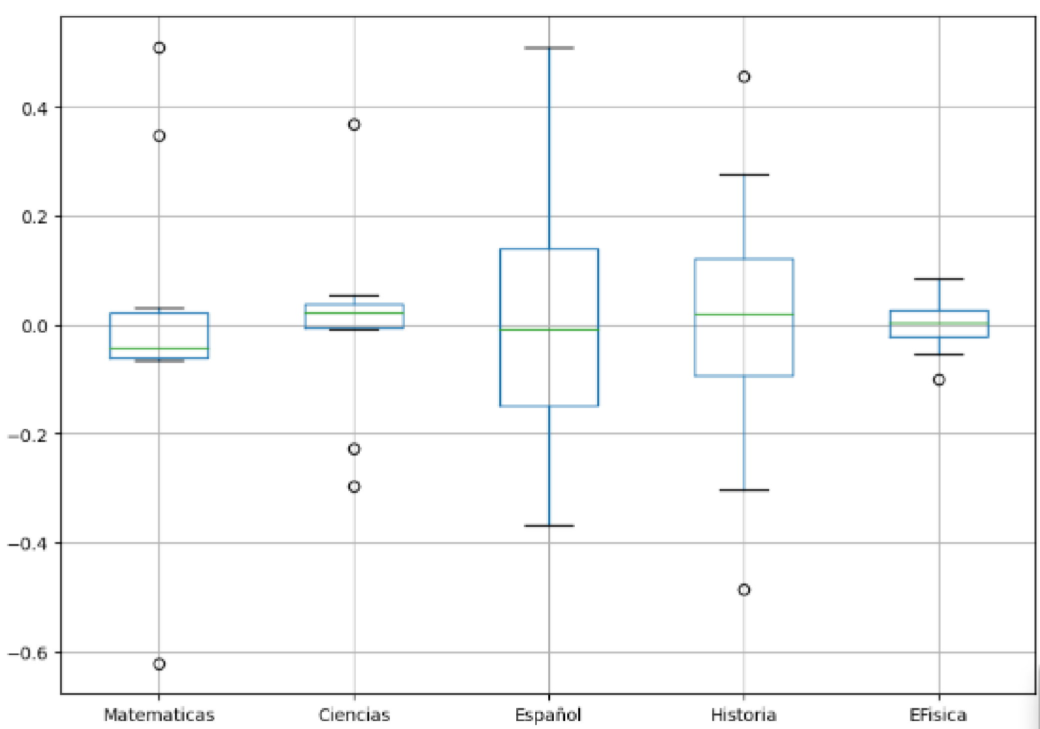
In [51]:
```
# calculamos los errores para 3 dimensiones

error3 = B - AReduced3

# convertimos a DF

dferror3 = pd.DataFrame(error3, columns = ['Matematicas','Ciencias','Español','Historia','EFisica'])
dferror3
```

Out[51]:

	Matematicas	Ciencias	Español	Historia	EFisica
0	0.347541	-0.227627	0.509577	-0.485121	-0.098984
1	-0.006492	-0.004371	0.112339	-0.082900	-0.018585
2	-0.064532	0.032436	0.044300	-0.014760	-0.004916
3	-0.620504	0.367993	-0.366944	0.456461	0.085697
4	-0.054719	0.054483	-0.343686	0.275202	0.059763
5	-0.031158	0.021521	-0.061415	0.055364	0.011512
6	-0.052536	0.039484	-0.148746	0.127455	0.026989
7	0.031744	-0.006977	-0.148666	0.103008	0.023693
8	0.509685	-0.295573	0.206754	-0.303505	-0.054519
9	-0.061238	0.023121	0.150272	-0.095687	-0.022614

In [52]:
```
# graficamos

plt.figure(figsize = (10,7))
dferror3.boxplot()
```

Out[52]: <Axes: >



In [53]:
```
# calculamos la media del error de cada columna(variable)

print('Errores promedio por columna')
error3.mean(axis = 0)
```

```
Errores promedio por columna
```

Out[53]:
```
array([-0.00022094,  0.00044883, -0.00462162,  0.00355172,  0.00078359])
```

In [ ]: