Re-exam ALI 2018 Solution

In [1]:

```
import sympy as sp
from sympy import *
init_printing()
from IPython.display import display, Latex, HTML
import numpy as np
```

Assignment 1

In [2]:

```
# a)

# 1: False
# 2: True
# 3: False
# 4: True
# 5: False
# 6: True
# 7: False
# 8: False
# 9: True

# b) # 2 is true

# c)
# It is invertible since it has 4 pivots
# It is diagonalizable since it has 4 distinct eigenvalues

# d) Dim V = 3
```

Assignment 2

In [3]:

```
a, b, x1, x2, x3 = symbols('a b x1 x2 x3')
```

In [4]:

```
# a)
A = Matrix([[-1, 3, 2], [1, 0, 1], [3, 3, a]])
A[1,0]*A.cofactor(1,0)+A[1, 2]*A.cofactor(1,2)
```

Out[4]:

$$-3a + 18$$

In [7]:

```
# b)
B = Matrix([-8, 2, b])
L, U, perm = A.row_join(B).LUdecomposition()
U
```

Out[7]:

$$\begin{bmatrix} -1 & 3 & 2 & -8 \\ 0 & 3 & 3 & -6 \\ 0 & 0 & a-6 & b \end{bmatrix}$$

In [8]:

```
# b) When a != 6

# c) When a = 6 and b != 0

# d) When a = 6 and b = 0
```

A.row_join(B).echelon_form()

Assignment 3

In [10]:

```
A = Matrix([[1, -1, 3], [2, 1, 0], [-1, -5, 9]])
b = Matrix([1, 5, -7])
x, y, z = symbols('x y z')
var = Matrix([x, y, z])

# a)
display(Latex("$${}{} = {}$$".format(latex(A), latex(var), latex(b))))
```

$$\begin{bmatrix} 1 & -1 & 3 \\ 2 & 1 & 0 \\ -1 & -5 & 9 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 5 \\ -7 \end{bmatrix}$$

In [11]:

```
# b)
display(A.row_join(b))
L, U, perm = A.row_join(b).LUdecomposition()
display(U)
```

$$\begin{bmatrix} 1 & -1 & 3 & 1 \\ 2 & 1 & 0 & 5 \\ -1 & -5 & 9 & -7 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 & 3 & 1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

In [12]:

```
# c)
A.row_join(b).rref()
```

Out[12]:

$$\left(\begin{bmatrix}1 & 0 & 1 & 2\\0 & 1 & -2 & 1\\0 & 0 & 0 & 0\end{bmatrix}, \quad (0, \quad 1)\right)$$

In [13]:

```
coef = Matrix([2, 1, 0])
display(Latex("$${} = {}+z{}$$".format(latex(var), latex(coef), latex(A.nullspace()[0
]))))
```

$$\begin{bmatrix}x\\y\\z\end{bmatrix} = \begin{bmatrix}2\\1\\0\end{bmatrix} + z\begin{bmatrix}-1\\2\\1\end{bmatrix}$$

In [16]:

```
# d)
# A inverse does not exist since A has a free variable and is hence not invertible.
```

Assignment 4

In [17]:

```
A = Matrix([[1,4], [3, 2]])

# a)

# First we check whether A is diagonalizable

vecs = A.eigenvects()
vecs
```

Out[17]:

$$\left[\left(-2, \quad 1, \quad \left[\begin{bmatrix}-\frac{4}{3}\\1\end{bmatrix}\right]\right), \quad \left(5, \quad 1, \quad \left[\begin{bmatrix}1\\1\end{bmatrix}\right]\right)\right]$$

In [20]:

```
# Since A has distinct eigenvalues, A is diagnonalizable.

d1 = vecs[1][0]
d2 = vecs[0][0]
q1 = vecs[1][2][0]
q2 = vecs[0][2][0]
Q = q1.row_join(q2)
D = diag(d1, d2)
Qinv = Q**-1
Q*D*Qinv
```

Out[20]:

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

In [25]:

```
# b)
# A^n = Q*D^n*Qinv
display(Latex("$$A^n = Q*D^n*Q^-1$$"))
```

$$A^n = Q * D^n * Q^-1$$

Assignment 5

In [29]:

```
# a)
v1 = Matrix([1, 0, 0, 1])
x2 = Matrix([1, 0, 1, 1])
x3 = Matrix([3, 2, 1, 1])
v2 = x2-x2.project(v1)
v3 = x3-x3.project(v1)-x3.project(v2)
V = v1.row_join(v2).row_join(v3)
display(V.T*V)
# So now, v1, v2, v3 will now form an orthogonal basis for W.

# b)
u1 = v1.normalized()
u2 = v2.normalized()
u3 = v3.normalized()
U = u1.row_join(u2).row_join(u3)
display(U.T*U)
# So u1, u2, u3 form an orthonormal basis for W.
```

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 6 \end{bmatrix}$$

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

Assignment 6

In [30]:

```
from scipy.io import loadmat
Data = loadmat('TV_Viewing.mat')
Data.keys()
```

Out[30]:

```
dict_keys(['__header__', '__version__', '__globals__', 'X'])
```

In [34]:

```
X = Data['X']
x = np.array(X[:,0]).astype(float)
y = np.array(X[:,1]).astype(float)
```

In [35]:

```
# a) Design Matrix for linear model

X1 = Matrix([ones(len(x),1)]).row_join(Matrix(x))
XtX = X1.T*X1
Xty = X1.T*Matrix(y)
Mat, _ = XtX.row_join(Xty).rref()
B1 = Mat[:,-1]
display(Latex("$$y_1(x) = {}+{}x$$".format(round(B1[0],2), round(B1[1], 4))))
```

$$y_1(x) = 37.28 + 0.9981x$$

In [37]:

```
# a) Design matrix for quadratic model

X2 = Matrix([ones(len(x),1)]).row_join(Matrix(x)).row_join(Matrix(x**2))
XtX = X2.T*X2
Xty = X2.T*Matrix(y)
Mat, _ = XtX.row_join(Xty).rref()
B2 = Mat[:,-1]
display(Latex("$$y_2(x) = {}+{}x+{}x^2$$"
              .format(round(B2[0],2), round(B2[1], 4), round(B2[2], 4))))
```

$$y_2(x) = 52.75 + 0.2018x + 0.0088x^2$$

In [39]:

```
# b)

display(Latex(
    "The error of the linear model is {}, and the error of the quadratic is model {}."
    " We conclude that the quadratic model has the best fit for this specific data."
    .format(round((Matrix(y)-X1*B1).norm(), 2), round((Matrix(y)-X2*B2).norm(), 2))))
```

The error of the linear model is 161.79, and the error of the quadratic is model 157.63. We conclude that the quadratic model has the best fit for this specific data.

Assignment 7

In [41]:

```
A = Matrix([[3, 1, 1], [-1, 3, 1]])

AtA = A.T*A
vecs = AtA.eigenvects()
vecs
```

Out[41]:

$$\left[\left(0,\quad 1,\quad \left[\left[\begin{matrix}-\frac{1}{5}\\ -\frac{2}{5}\\ 1\end{matrix}\right]\right]\right),\quad \left(10,\quad 1,\quad \left[\left[\begin{matrix}-2\\ 1\\ 0\end{matrix}\right]\right]\right),\quad \left(12,\quad 1,\quad \left[\left[\begin{matrix}1\\ 2\\ 1\end{matrix}\right]\right]\right.\right.$$

In [43]:

```
s1 = sqrt(vecs[2][0])
s2 = sqrt(vecs[1][0])
s3 = sqrt(vecs[0][0])
v1 = vecs[2][2][0].normalized()
v2 = vecs[1][2][0].normalized()
v3 = vecs[0][2][0].normalized()
u1 = (s1**-1)*A*v1
u2 = (s2**-1)*A*v2
U = u1.row_join(u2)
V = v1.row_join(v2).row_join(v3)
Vt = V.T
S = diag(s1,s2).row_join(zeros(2,1))
display(U*S*Vt)
display(A)
```

$$\left[\begin{matrix}3 & 1 & 1\\ -1 & 3 & 1\end{matrix}\right]$$

$$\left[\begin{matrix}3 & 1 & 1\\ -1 & 3 & 1\end{matrix}\right]$$

In [3]:

```
```

Out[3]:

$$\left[\begin{matrix}-3 & 1\\ 6 & -2\\ 6 & -2\end{matrix}\right]$$

```
In [5]:
```

```
Out[5]:
```

$$\left[\left(0, \quad 2, \quad \left[\left[\begin{matrix}2\\1\\0\end{matrix}\right], \quad \left[\begin{matrix}2\\0\\1\end{matrix}\right]\right]\right), \quad \left(90, \quad 1, \quad \left[\left[\begin{matrix}- \frac{1}{2}\\1\\1\end{matrix}\right]\right]\right)\right]$$

```
In [ ]:
```