Answer 3

Answer 3

given → $W \in \mathbb{R}^4$

$W \sim N(0, \gamma^2 I)$ → Σ → covariance matrix

$\vec{D} = (x_1, y_1), (x_2, y_2) - - - , (x_n, y_n)$

$y_i = ax_i^3 + bx_i^2 + cx_i + d + v$

$v \sim N(0, \sigma^2)$

To find → $\underset{w}{\operatorname{argmax}} \; P(\vec{w} \mid \vec{D})$

$$W_{MAP} = \underset{w}{\operatorname{argmax}} \frac{P(\vec{D} \mid \vec{w}) P(\vec{w})}{P(\vec{D})}$$

$$= \underset{w}{\operatorname{argmax}} \left[\ln P(\underset{D_1, D_2, - - - -, D_N}{\vec{D}} \mid \vec{w}) + \ln P(\vec{w}) \right]$$

↓ independent samples

$$= \underset{w}{\operatorname{argmax}} \left[\ln \prod_{i=1}^{N} (\vec{D}_i \mid \vec{w}) + \ln P(\vec{w}) \right]$$

$$= \underset{w}{\operatorname{argmax}} \left[\sum_{i=1}^{N} \ln (\vec{D}_i \mid \vec{w}) + \ln P(\vec{w}) \right]$$

$$\cancel{\ln(y)} \quad D_i \mid \vec{w} \equiv (y_i, x_i) \mid \vec{w}$$

$$P(y_i \mid x_i, w) = a x_i^3 + b x_i^2 + c x_i + d + V$$

→ gaussion with mean $w^T \vec{x}$

→ and standard deviation σ^2

$$\text{where } w^T x = a x_i^3 + b x_i^2 + c x_i + d$$

$$\therefore P(y_i \mid x_i, \cancel{\phi}\, w) = N(w^T x_i, \sigma^2)$$

$$\ln (y_i \mid x_i, w) = \ln \left[\underbrace{\frac{1}{\sqrt{2\pi\sigma^2}}}_{\text{independent of } \vec{w}} e^{-\frac{(y_i - w^T x_i)^2}{2\sigma^2}} \right]$$

$$w_{MAP} = \underset{w}{\arg\max} \left[\sum_{i=1}^{N} - \frac{(y_i - w^T x_i)^2}{2\sigma^2} + \cancel{\frac{1}{2\pi\sigma^2}} \right.$$

$$\ln \left(\underbrace{\left((2\pi)^4 |\Sigma|\right)^{-1/2}}_{\text{independent of } \vec{w}} \exp\left(-\frac{1}{2} (\vec{w})^T \Sigma^{-1} w \right) \right)$$

$$W_{MAP} = \underset{W}{\mathrm{argmax}} \sum_{i=1}^{N} -\frac{(y_i - W^T x_i)^2}{2\sigma^2} - \frac{1}{2}\left(W^T \Sigma^{-1} W\right)$$

$$W_{MAP} = \underset{W}{\mathrm{argmin}} \sum_{i=1}^{N} \frac{(y_i - W^T x_i)^2}{2\sigma^2} + \frac{1}{2}\left(W^T \Sigma^{-1} W\right)$$

where $\Sigma = \gamma^2 I$

$$W_{MAP} = \underset{W}{\operatorname{argmin}} \sum_{i=1}^{W} \frac{(y_i - W^T x_i)^2}{2\sigma^2} + \frac{1}{2}(W^T \Sigma^{-1} W)$$

where $\Sigma = \gamma^2 I$

$$W_{MAP} = \underset{W}{\operatorname{argmin}} \left(\frac{1}{\sigma^2}\right)\left(\frac{1}{2}\right)\left(XW - Y\right)^T \left(XW - Y\right) +$$

$$\frac{1}{2}\left(W^T \Sigma^{-1} W\right)$$

$$= \underset{W}{\operatorname{argmin}} \left(\frac{1}{2\sigma^2}\right)\left(XW - Y\right)^T\left(XW - Y\right) + \frac{1}{2\gamma^2} W^T W$$

Take derivative and set to $0 \rightarrow$

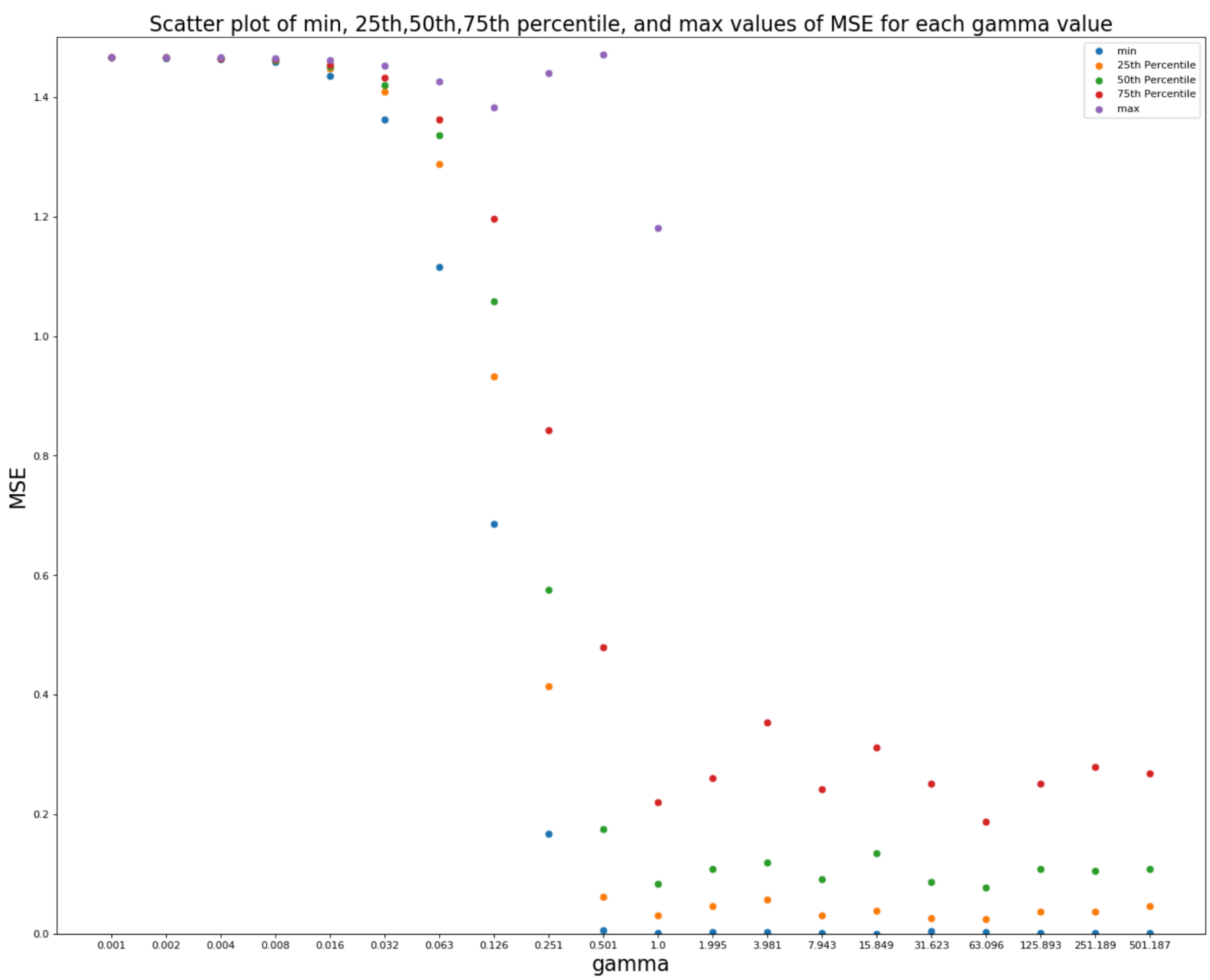
$$\nabla_W f = \frac{1}{\sigma^2}\left(X^T X W - X^T Y\right) + \frac{1}{\gamma^2} W = 0$$

$$\left[\frac{X^T X}{\sigma^2} + \frac{1}{\gamma^2}\right] W = \frac{X^T Y}{\sigma^2}$$

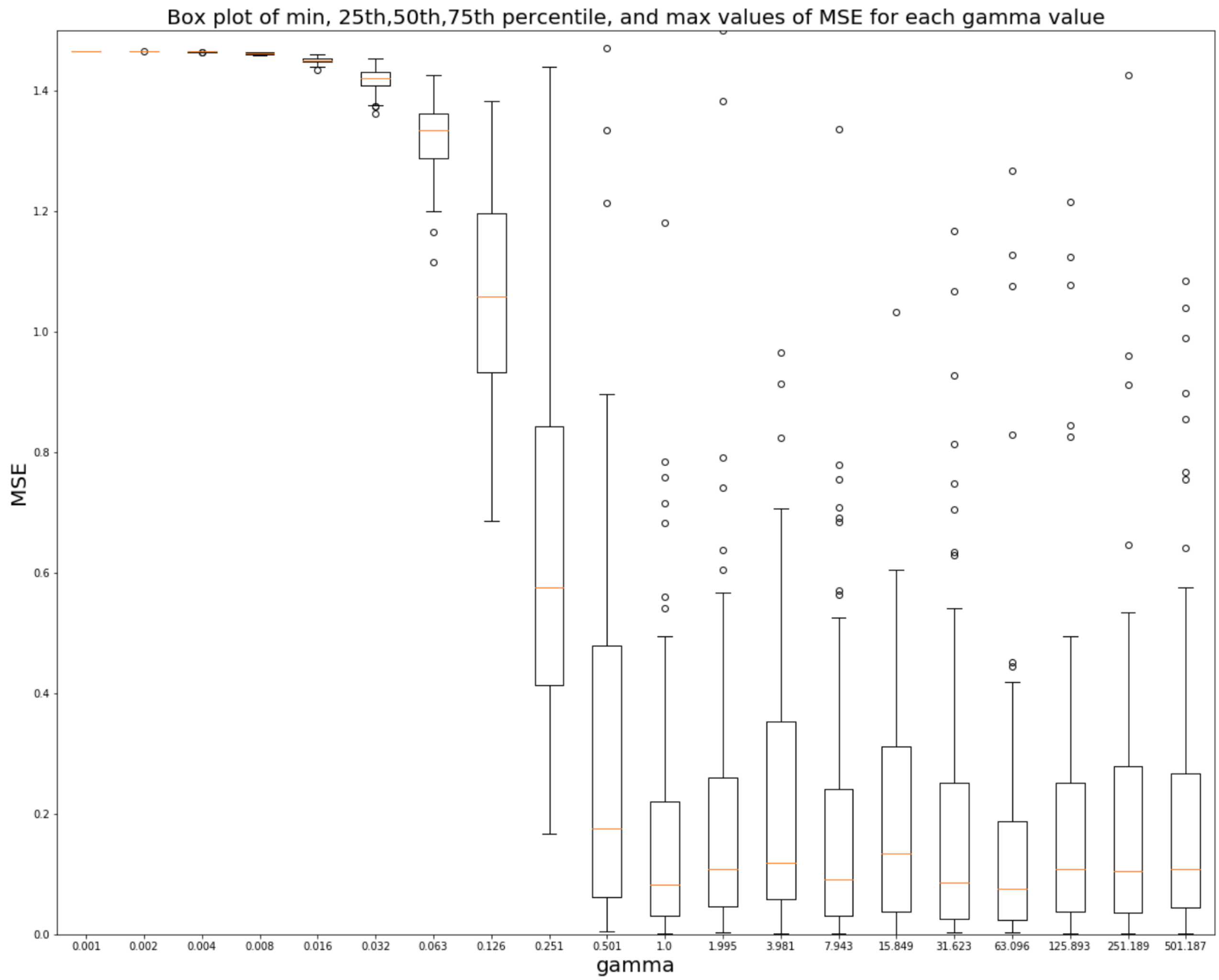
$$\cancel{W = (\gamma^2 X^T X + \sigma^2) X^T Y}$$

$$\left[X^T X + \frac{\sigma^2}{\gamma^2}\right] w = X^T y$$

$$w = \left(X^T X + \frac{\sigma^2}{\gamma^2} I\right)^{-1} \left(X^T y\right)$$



MSE falls as Gamma increases



MSE falls as Gamma increases

In [182]:
```python
'''
ML estimate
'''

C=np.matmul((X_stack.T),X_stack)
A=np.linalg.inv(C)
B=np.matmul(X_stack.T,Y)
w_ml=np.matmul(A,B)
print ("W_ML is \n",w_ml)

print ("\n\nWith Gamma :",gamma,", W_MAP is \n",w_map)
print ("\nAs gamma increases, w_MAP indeed approaches to w_ml")
```

```
W_ML is 
 [[ 1.00279534]
 [-0.19552492]
 [-0.65031524]
 [ 0.13020576]]


With Gamma : 100000.0 , W_MAP is 
 [[ 1.00279534]
 [-0.19552492]
 [-0.65031524]
 [ 0.13020576]]

As gamma increases, w_MAP indeed approaches to w_ml
```

Appendix

Consider least square expression →

$$\underset{w}{\operatorname{argmin}} \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - xw)^2$$

independent of w

in vectorized form →

1000, 4

4, 1

1000, 1

$$\underset{w}{\operatorname{argmin}} f(w) = \frac{1}{2} \cancel{y}(Xw - y)^T (Xw - y)$$

$$= \frac{1}{2} (XW - y)^T (XW - y)$$

→ assume W is feature vector of size 4 ; and 1000 samples

$$\underset{W}{\operatorname{argmin}} \frac{1}{2} (XW - y)^T (XW - y)$$

$$= \frac{1}{2} \underset{W}{\operatorname{argmin}} (W^T X^T X W - W^T X^T y - y^T X W + y^T y)$$

$$= \frac{1}{2} \operatorname{tr} \left(W^T X^T X W - W^T X^T y - y^T X W + y^T y \right)$$

trace

$$= \frac{1}{2} \left(\operatorname{tr} W^T X^T X W - 2 \operatorname{tr} y^T X W \right)$$

$$\nabla_w f = \frac{1}{2} \left(X^T X W + X^T X W - 2 X^T y \right)$$

$$= X^T X W - X^T y$$

formula used for derivative

where $X \in \mathbb{R}^{1000 \times 4}$

$X^T \in \mathbb{R}^{4 \times 1000}$

$Y \in \mathbb{R}^{1000 \times 1}$

$W \in \mathbb{R}^{4 \times 1}$

assuming 1000 samples.

For 1 sample $\rightarrow$ $\vec{x}_i = [x_i^3, x_i^2, x_i, x_i^0]$

$\vec{w} = [a, b, c, d]$

Code for this Question-

Github- https://github.com/rohinarora/EECE5644-Machine_Learning/Exam1/Q3.ipynb

In [1]:
```
import sympy as sym
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
'''
true roots
'''
x=sym.symbols('x')
r1=-0.8
r2=0.2
r3=0.8
y=(x-r1)*(x-r2)*(x-r3)
z=sym.expand(y.simplify())
z
```

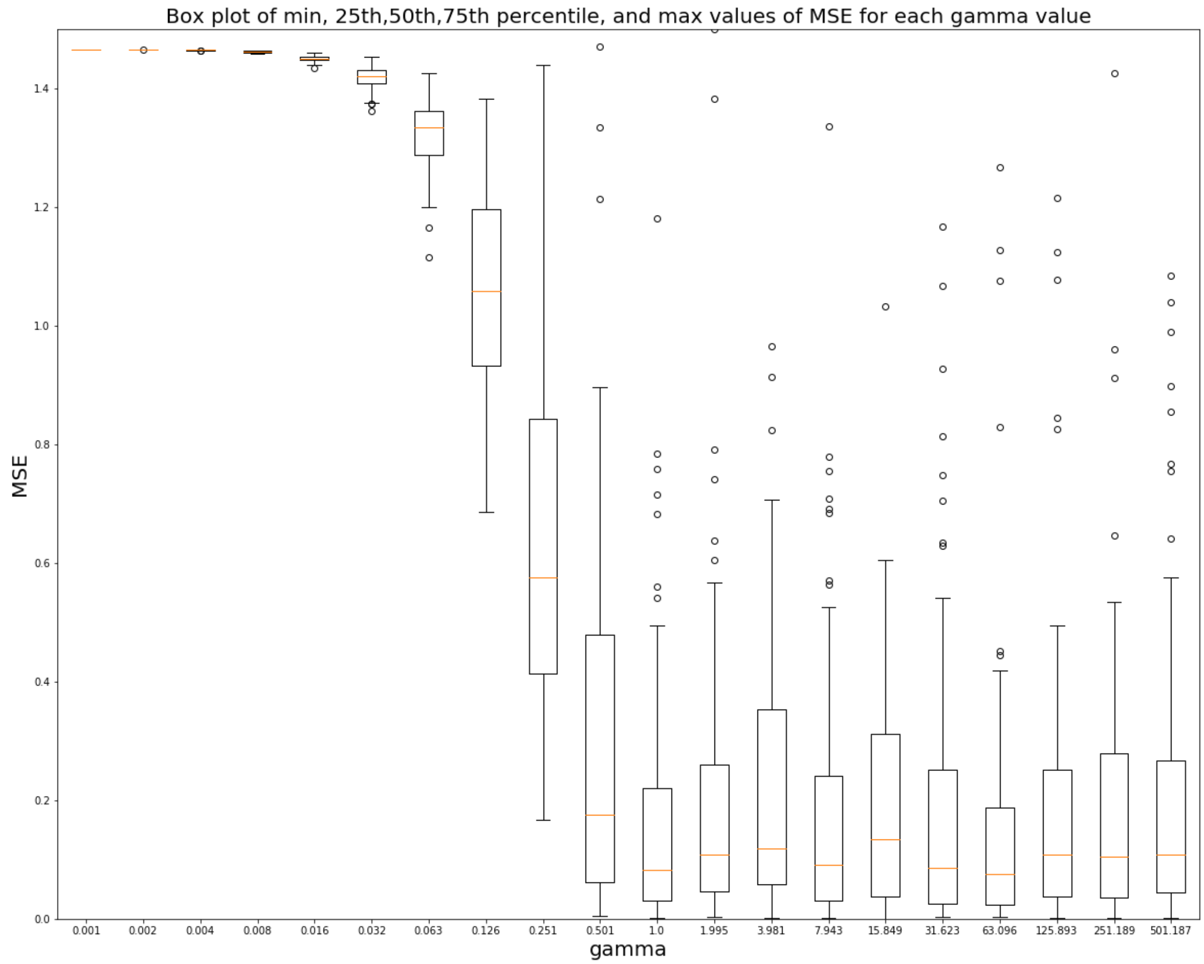
Out[2]: $x^3 - 0.2x^2 - 0.64x + 0.128$

In [3]:
```
'''
true values of W parameter
'''
a=(z.coeff(x**3))
b=(z.coeff(x**2))
c=(z.coeff(x))
d=z.coeff(x,n=0)
w=np.array([a,b,c,d])
w=np.array((w.reshape(-1,1)),dtype=float)
w
```

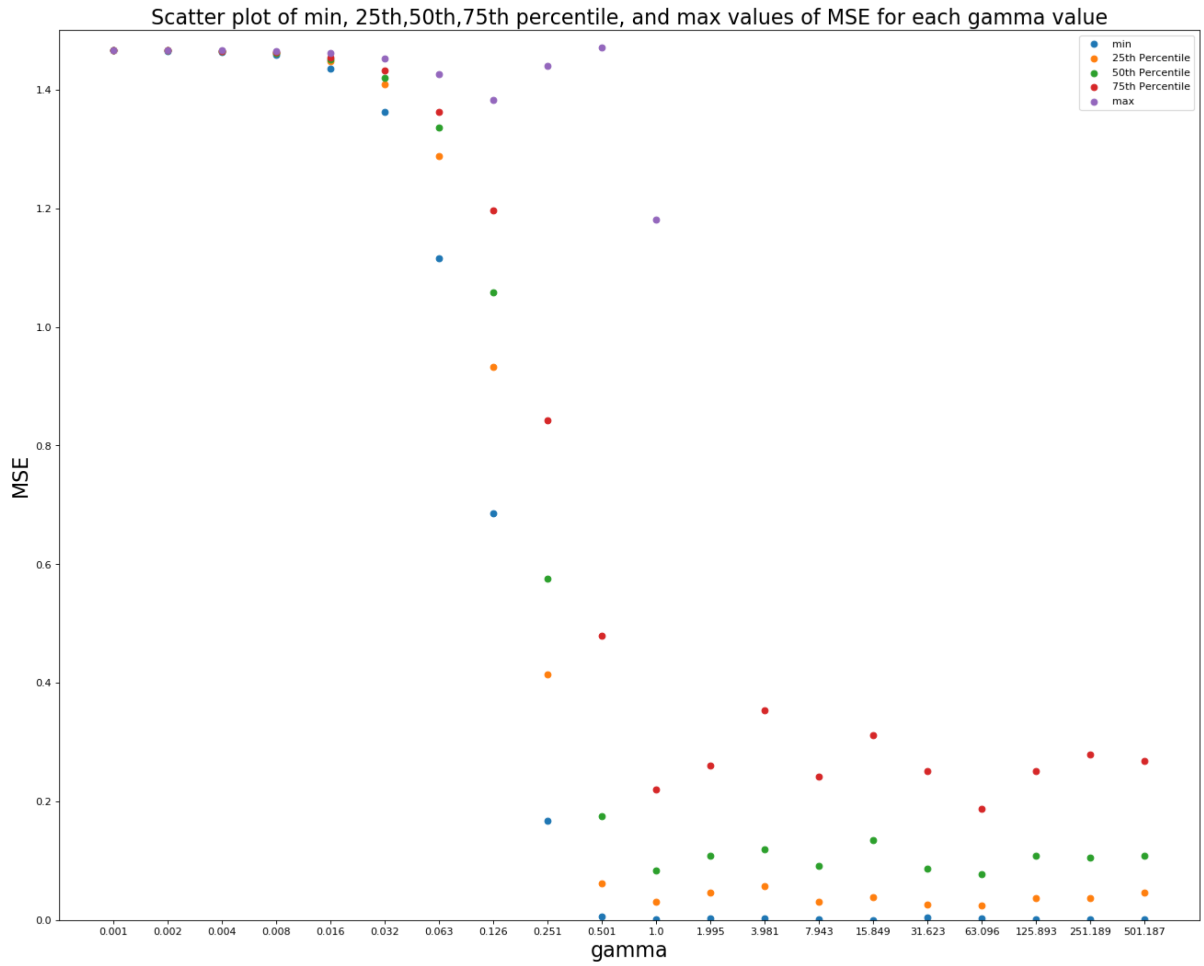
Out[3]:
```
array([[ 1.   ],
       [-0.2  ],
       [-0.64 ],
       [ 0.128]])
```

In [225]:
```
'''
X_stack.shape=(samples,4)
'''

num_samples=10
result={}
sigma=.1 # (Standard deviation of noise) (.01 seems decent) # fixed
gamma_low=-3
gamma_high=3
num_points=20
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    result[gamma]=[]
    for i in range(100):
        Y=[]
        X=[]
        for i in range(num_samples):
            x=np.random.uniform(-1,1)
            y=w[0]*x**3+w[1]*x**2+w[2]*x**1+w[3]*x**0+np.random.normal(s
cale=sigma)
            Y.append(y)
            X.append(x)
        Y=np.array(np.array(Y).reshape(-1,1),dtype=float)
        X=np.array(X).reshape(-1,1)
        X_stack=np.hstack([X**3,X**2,X**1,X**0])
        C=np.matmul((X_stack.T),X_stack)+((sigma**2)/(gamma**2))*np.iden
tity(4)
        A=np.linalg.inv(C)
        B=np.matmul(X_stack.T,Y)
        w_map=np.matmul(A,B)
        result[gamma].append(np.sum(np.square(w_map-w)))
label=[]
result_plot=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    label.append(np.round(gamma,3))
    result_plot.append(result[gamma])
fig1, ax1 = plt.subplots(figsize=(20,16))
from matplotlib.pyplot import figure
plt.ylim(0,1.5)
fig1.text(.35,0.06,'MSE falls as Gamma increases',fontsize=25);
plt.title('Box plot of min, 25th,50th,75th percentile, and max values of
MSE for each gamma value',fontsize=20)
plt.ylabel('MSE',fontsize=20)
plt.xlabel('gamma',fontsize=20)
ax1.boxplot(result_plot,labels=label);

plt.show()
print ("\n\n\n")
from matplotlib.pyplot import figure
fig=figure(num=None, figsize=(20, 16), dpi=80, facecolor='w', edgecolor=
'k')
X=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    X.append(str(np.round(gamma,3)))
```

```
result_min=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    result_min.append((np.array(result[gamma])).min())

result_max=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    result_max.append((np.array(result[gamma])).max())

result_25=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    result_25.append(np.percentile(np.array(result[gamma]),25))
result_50=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    result_50.append(np.percentile(np.array(result[gamma]),50))

result_75=[]
for gamma in np.logspace(gamma_low,gamma_high,num=num_points, endpoint=F
alse):
    result_75.append(np.percentile(np.array(result[gamma]),75))
plt.ylim(0,1.5)
plt.title('Scatter plot of min, 25th,50th,75th percentile, and max value
s of MSE for each gamma value',fontsize=20)
plt.scatter(X,result_min,label='min')
plt.scatter(X,result_25,label='25th Percentile')
plt.scatter(X,result_50,label='50th Percentile')
plt.scatter(X,result_75,label='75th Percentile')
plt.scatter(X,result_max,label='max')
plt.ylabel('MSE',fontsize=20)
plt.xlabel('gamma',fontsize=20)
fig.text(.35,0.06,'MSE falls as Gamma increases',fontsize=25);
plt.legend();
```



MSE falls as Gamma increases



MSE falls as Gamma increases

In [182]:
```
'''
ML estimate
'''

C=np.matmul((X_stack.T),X_stack)
A=np.linalg.inv(C)
B=np.matmul(X_stack.T,Y)
w_ml=np.matmul(A,B)
print ("W_ML is \n",w_ml)

print ("\n\nWith Gamma :",gamma,", W_MAP is \n",w_map)
print ("\nAs gamma increases, w_MAP indeed approaches to w_ml")
```

```
W_ML is
 [[ 1.00279534]
 [-0.19552492]
 [-0.65031524]
 [ 0.13020576]]


With Gamma : 100000.0 , W_MAP is
 [[ 1.00279534]
 [-0.19552492]
 [-0.65031524]
 [ 0.13020576]]

As gamma increases, w_MAP indeed approaches to w_ml
```