Q1 (a) Function Creation: same_frequency

In [2]:
```
def same_frequency(num1,num2):
    num1=str(num1)
    num2=str(num2)
    for i in range(10):
        if num1.count(f'{i}')!=num2.count(f'{i}'):
            return False
    return True
```

In [5]:
```
print(same_frequency(551122,221515))
print(same_frequency(321142,3212215))
print(same_frequency(12345,31354))
print(same_frequency(1212, 2211))
```

```
True
False
False
True
```

Q1 (b) Matplotlib Data Visualization of K-means Clusters

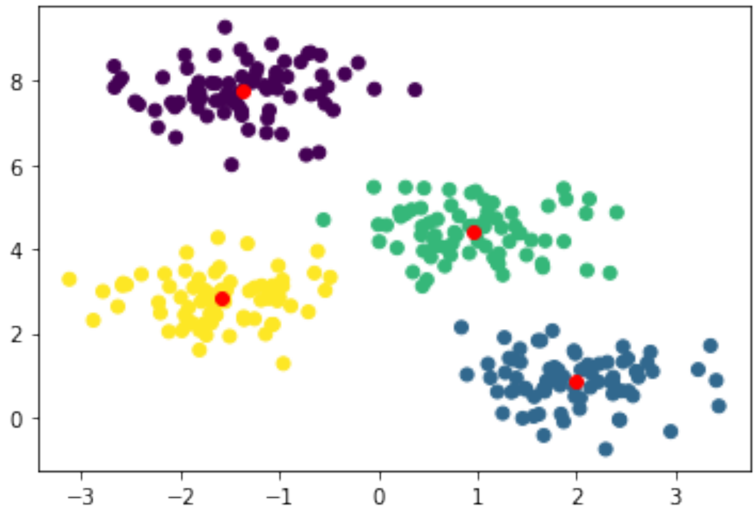
In [6]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [7]:
```
T = np.load('kmeans.npz')
```

In [8]:
```
data=T['data']
pred=T['pred']
centers=T['centers']
```

In [9]:
```
plt.scatter(data[:,0],data[:,1],c=pred)
plt.scatter(centers[:,0],centers[:,1],color='red')
```

Out[9]: <matplotlib.collections.PathCollection at 0x7fba612675b0>



Q2 Regularized Linear Regression

(a)

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt
```

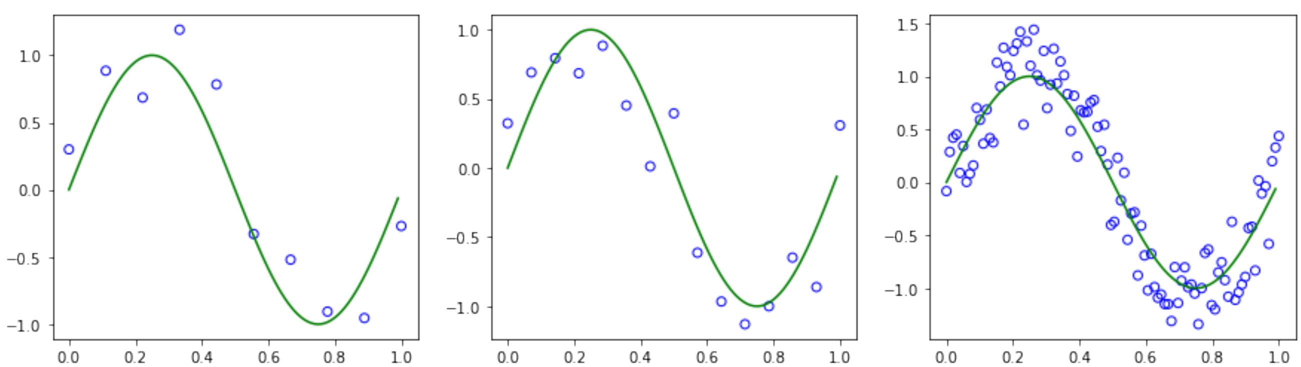
In [4]:
```
data10=np.loadtxt('data10.txt')
data15=np.loadtxt('data15.txt')
data100=np.loadtxt('data100.txt')
```

In [5]:
```
x=np.arange(0,1,0.01)
```

In [6]:
```
ground_true=np.sin(2*x*np.pi)
```

In [7]:
```
plt.figure(figsize=(15,4))
plt.figure(1)
ax1 = plt.subplot(131)
plt.scatter(data10[:,0],data10[:,1],c='none',marker='o',edgecolors='b')
plt.plot(x,ground_true,c="g")
ax2=plt.subplot(132)
plt.scatter(data15[:,0],data15[:,1],c='none',marker='o',edgecolors='b')
plt.plot(x,ground_true,c="g")
ax3=plt.subplot(133)
plt.scatter(data100[:,0],data100[:,1],c='none',marker='o',edgecolors='b')
plt.plot(x,ground_true,c="g")
```

Out[7]: [<matplotlib.lines.Line2D at 0x7f7a60594790>]



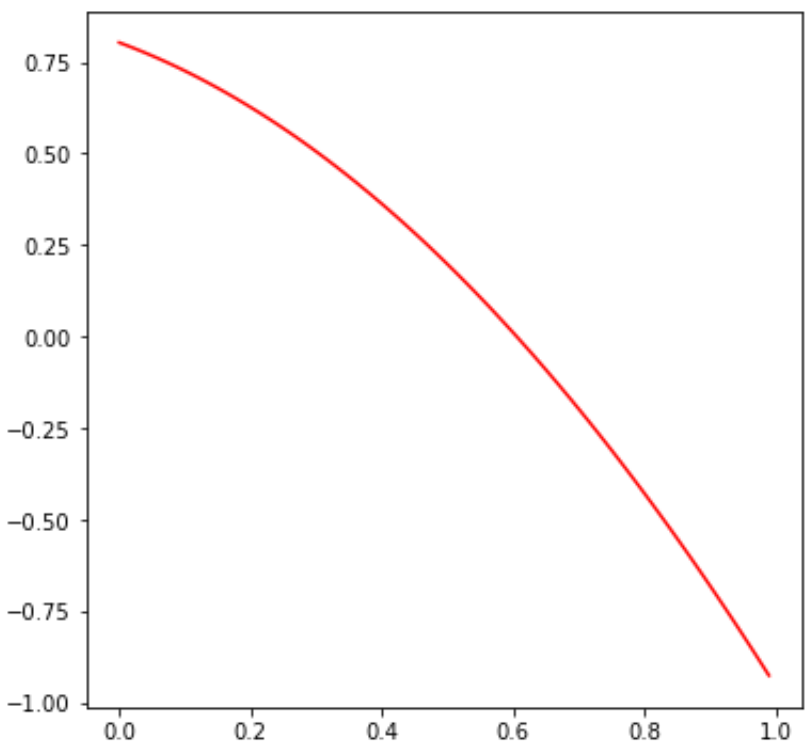
(b)

In [8]:
```
def cCalculation(m,filename,lamda):
    data=np.loadtxt(filename)
    A=np.empty(shape=(len(data),0))
    for i in range(m,-1,-1):
        x=((data[:,0])**i).reshape(len(data),1)
        A=np.hstack((A,x))
    b=data[:,1]
    I=np.eye(m+1,m+1)
    I[m,m]=0
    c=(np.linalg.inv(((A.T).dot(A)+lamda*I))).dot((A.T)).dot(b)
    y_predict=A.dot(c)
    print('y_predict:',y_predict)
    print('\nc:',c)
    x=np.arange(0,1,0.01)
    pol=np.poly1d(c)
    Y=pol(x)
    plt.figure(figsize=(6,6))
    plt.figure(1)
    plt.plot(x,Y,c="r")
    return
```

m=2, data_number=10, lamda=exp(-10)

In [9]:
```
cCalculation(2,'data10.txt',np.exp(-10))
```

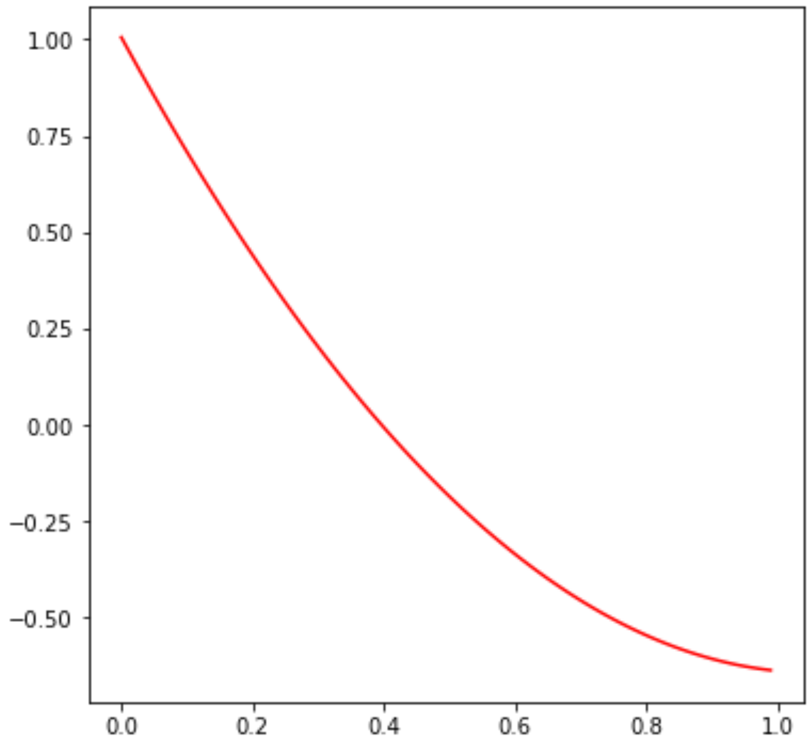
```
y_predict: [ 0.80278999  0.71571757  0.60169663  0.46072716  0.29280917  0.096
06461
 -0.12599322 -0.37499958 -0.65095446 -0.95385787]

c: [-1.09360143 -0.66304643  0.80278999]
```



m=2, data_number=15, lamda=exp(-10)

In [10]:
```
cCalculation(2,'data15.txt',np.exp(-10))
```

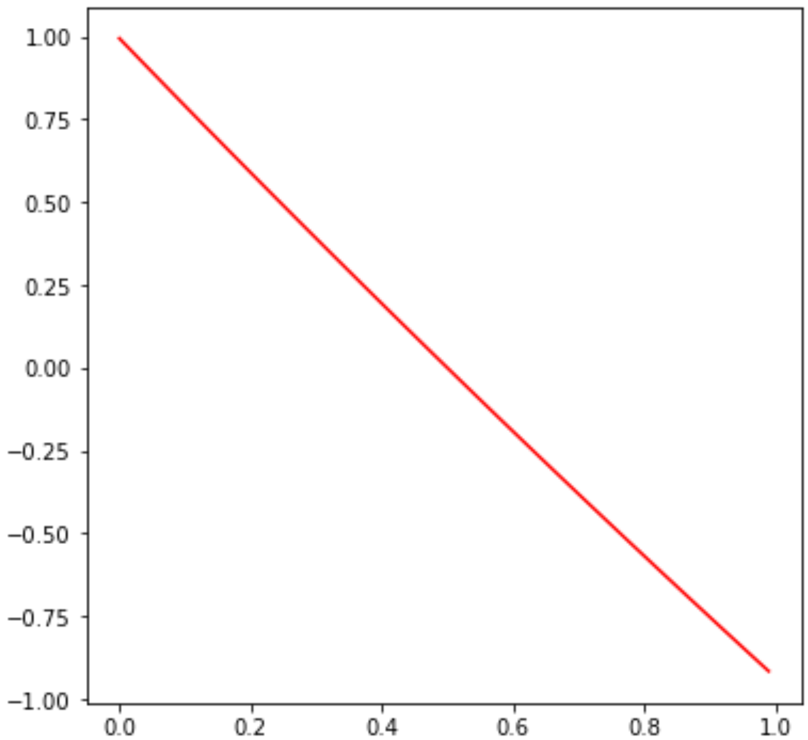
```
y_predict: [ 1.00509481  0.78992556  0.58926218  0.40522039  0.23377944  0.079
70159
 -0.06135339 -0.18546731 -0.29470403 -0.39028631 -0.46955909 -0.53475541
 -0.58406426 -0.61887462 -0.63821953]

c: [ 1.47561981 -3.11893414  1.00509481]
```



m=2, data_number=100, lamda=exp(-10)

In [11]:
```
cCalculation(2,'data100.txt',np.exp(-10))
```

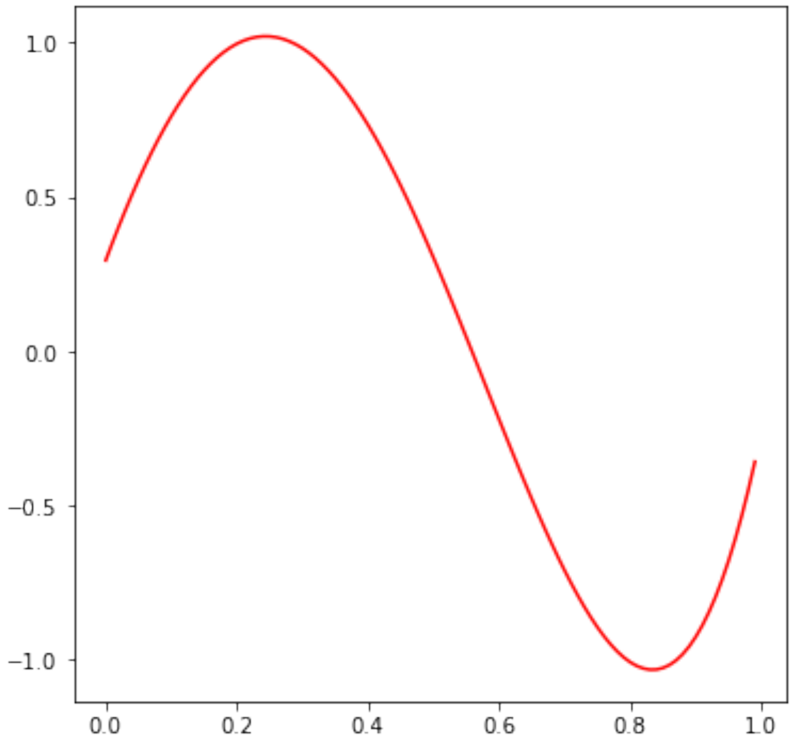
```
y_predict: [ 0.99431686  0.97362671  0.95296128  0.93232055  0.91170453  0.891
11321
  0.87054661  0.85000472  0.82948754  0.80899506  0.7885273   0.76828653
  0.74806998  0.72787766  0.70770955  0.68555262  0.66543538  0.64534237
  0.62527358  0.60522901  0.58520867  0.56521254  0.54524064  0.52529297
  0.50536951  0.48348169  0.4636091   0.44376073  0.42393659  0.40413667
  0.38436097  0.36460949  0.34488224  0.32517921  0.3055004   0.28388169
  0.26425374  0.24465002  0.22507053  0.20551525  0.1859842   0.16647737
  0.14699476  0.12753638  0.10810222  0.08675262  0.06736932  0.04801025
  0.0286754   0.00936477 -0.00992163 -0.02918382 -0.04842178 -0.06763552
 -0.08682503 -0.10790552 -0.12704417 -0.14615859 -0.1652488  -0.18431478
 -0.20335654 -0.22237407 -0.24136739 -0.26033648 -0.27928135 -0.30009273
 -0.31898673 -0.33785651 -0.35670206 -0.3755234  -0.39432051 -0.4130934
 -0.43184206 -0.45056651 -0.46926673 -0.489809   -0.50845835 -0.52708348
 -0.54568439 -0.56426108 -0.58281355 -0.60134179 -0.61984581 -0.63832561
 -0.65678118 -0.67705434 -0.69545905 -0.71383953 -0.73219579 -0.75052784
 -0.76883565 -0.78711925 -0.80537862 -0.82361377 -0.8418247  -0.86182875
 -0.87998881 -0.89812465 -0.91623626 -0.93432366]

c: [ 0.12111225 -2.04975277  0.99431686]
```



m=6, data_number=10, lamda=exp(-10)

In [12]:
```
cCalculation(6,'data10.txt',np.exp(-10))
```

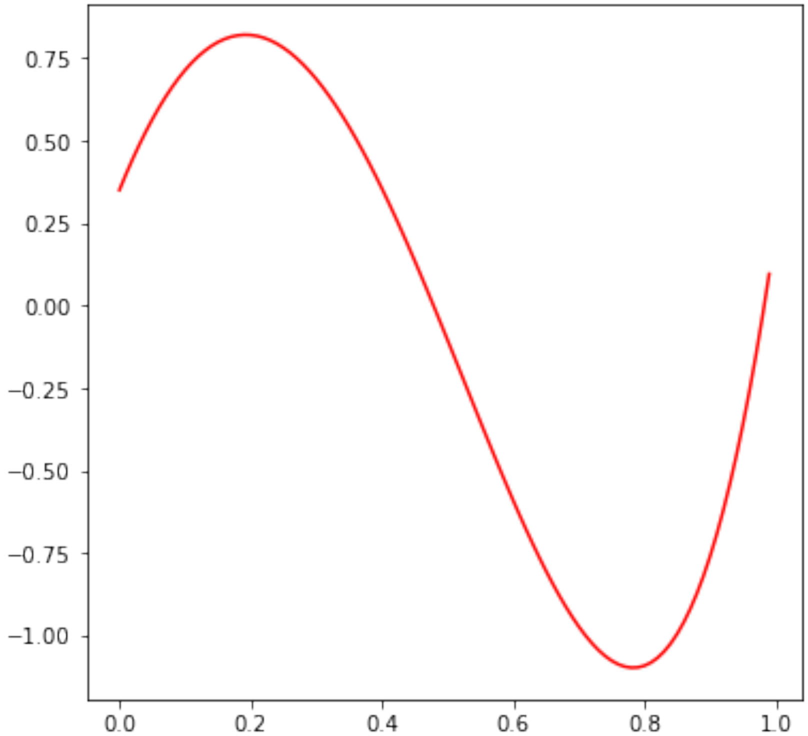
```
y_predict: [ 0.29506395  0.80003219  1.01499813  0.92426119  0.56208621  0.013
68128
 -0.56195235 -0.96758507 -0.95795735 -0.25862818]

c: [-3.86535379  9.54404978  6.21580073 -8.51711016 -9.64611561  5.71503691
  0.29506395]
```



m=6, data_number=15, lamda=exp(-10)

In [13]:
```
cCalculation(6,'data15.txt',np.exp(-10))
```

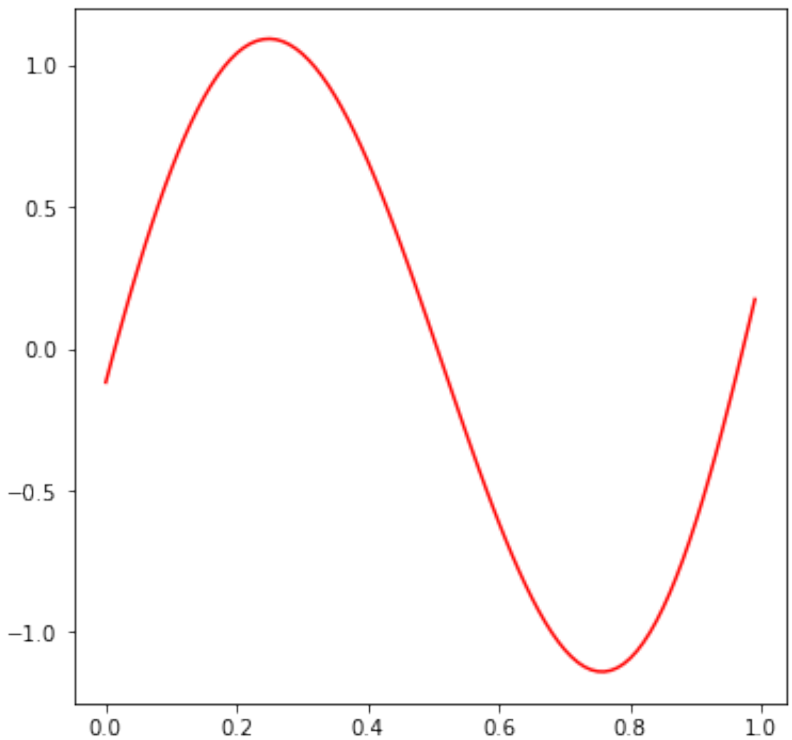
```
y_predict: [ 0.35031548  0.63624437  0.79060018  0.81529702  0.71841802  0.517
82605
  0.23075939 -0.10465812 -0.45431077 -0.77689376 -1.00989592 -1.09787721
 -0.97080641 -0.55017711  0.2308588 ]

c: [ -2.03675      5.68598222   6.16034482  -2.12834588 -12.72211711
   4.92142926   0.35031548]
```



m=6, data_number=100, lamda=exp(-10)

In [14]:
```
cCalculation(6,'data100.txt',np.exp(-10))
```

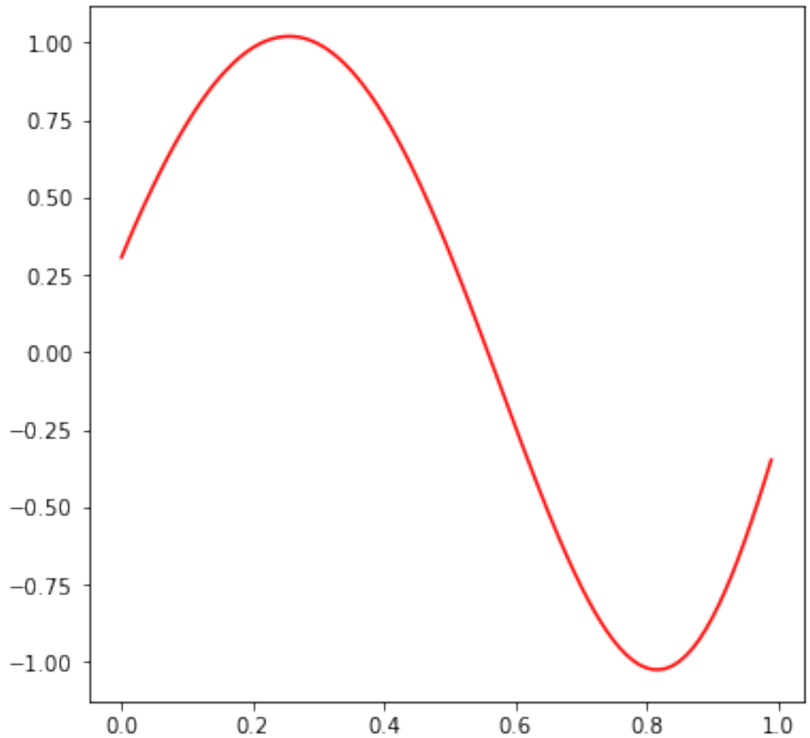
```
y_predict: [-0.11929706 -0.03220277  0.0528708   0.13571484  0.21612889  0.293
92115
  0.36890879  0.44091829  0.50978569  0.57535687  0.63748779  0.69548286
  0.7498547   0.80049394  0.84730165  0.89425962  0.93274638  0.96716278
  0.99745401  1.02357625  1.04549671  1.06319368  1.07665654  1.08588574
  1.09089278  1.09155118  1.08777817  1.07988847  1.06793838  1.05199483
  1.03213531  1.00844773  0.9810302   0.94999091  0.91544787  0.87355634
  0.83208214  0.78753016  0.74005632  0.68982504  0.637009    0.58178876
  0.52435242  0.46489529  0.40361946  0.33436447  0.26995514  0.20439239
  0.13790179  0.07071351  0.00306176 -0.06481567 -0.13267795 -0.2002818
 -0.26738197 -0.34031612 -0.40555515 -0.46952451 -0.53197737 -0.59266816
 -0.65135322 -0.70779149 -0.76174523 -0.81298075 -0.86126914 -0.91071604
 -0.95209603 -0.98985809 -1.02380113 -1.05373295 -1.07947108 -1.10084364
 -1.11769028 -1.12986303 -1.13722726 -1.13963088 -1.13652371 -1.12828573
 -1.11484707 -1.0961562  -1.07218097 -1.04290966 -1.00835209 -0.96854066
 -0.92353147 -0.86811535 -0.81248543 -0.75199468 -0.6868084  -0.61712155
 -0.54315995 -0.46518151 -0.38347752 -0.2983739  -0.21023245 -0.11024561
 -0.01706974  0.07777976  0.17377992  0.27036499]

c: [-24.94871017  30.77134599  30.38930654 -35.69319503  -8.84522894
   8.71614366  -0.11929706]
```



m=9, data_number=10, lamda=exp(-10)

In [15]:
```
cCalculation(9,'data10.txt',np.exp(-10))
```

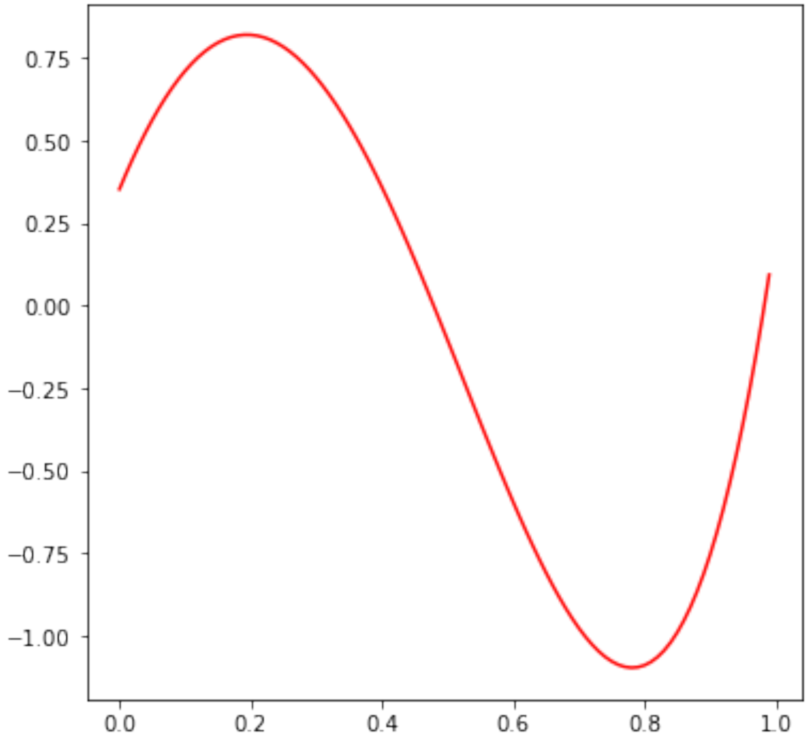
```
y_predict: [ 0.30633599  0.77608761  1.00591854  0.94504555  0.59060008  0.012
60142
 -0.60135441 -0.99269289 -0.89932454 -0.27921735]

c: [-3.5338486  -1.84896842  2.29997278  6.47141056  6.80582425  0.25321228
 -9.30526316 -6.83132438  5.10343135  0.30633599]
```



m=9, data_number=15, lamda=exp(-10)

In [16]:
```
cCalculation(9,'data15.txt',np.exp(-10))
```

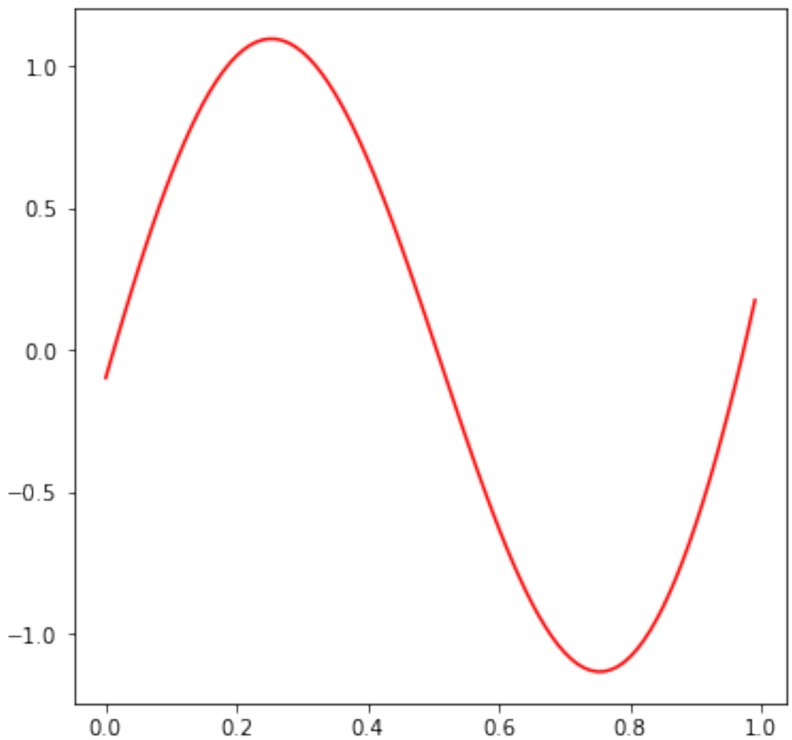
```
y_predict: [ 0.35216585  0.6332145   0.78786632  0.8154001   0.72142887  0.521
9194
  0.23348417 -0.10504289 -0.45787615 -0.78176767 -1.01298786 -1.09669608
 -0.96589879 -0.54632004  0.22681027]

c: [  2.31628497  -3.48724108  -2.32384348   2.433444     6.2482091
   4.82652127  -2.94622439 -11.99855474   4.80604877   0.35216585]
```



m=9, data_number=100, lamda=exp(-10)

In [17]:
```
cCalculation(9,'data100.txt',np.exp(-10))
```

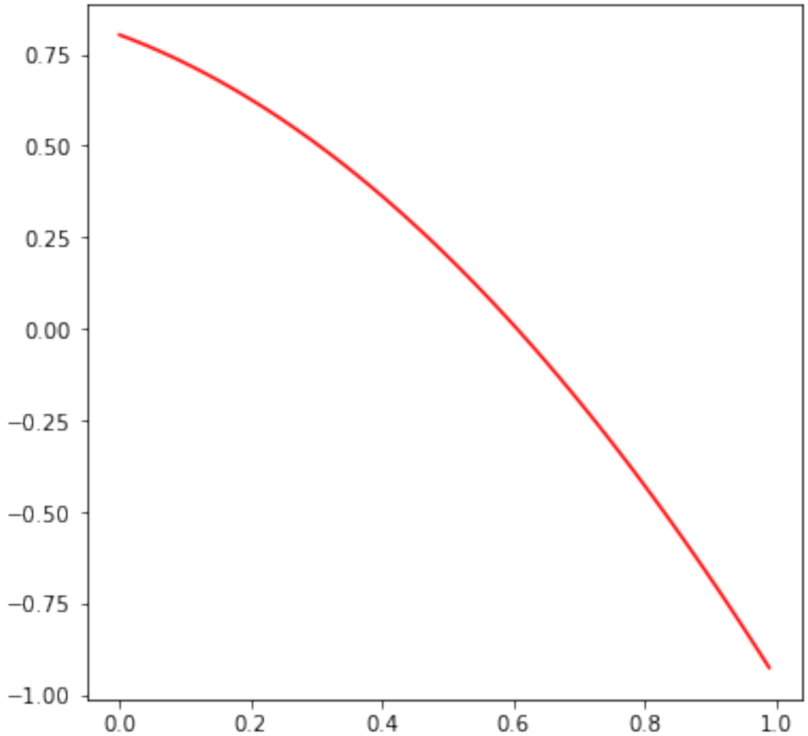
```
y_predict: [-0.0974112  -0.0164568   0.06315383  0.14119174  0.21743403  0.291
66432
  0.36367313  0.43325837  0.50022574  0.56438918  0.62557131  0.6830452
  0.73727734  0.78812308  0.83544766  0.88328945  0.92282697  0.95849225
  0.99019473  1.0178554   1.04140712  1.06079482  1.07597578  1.0869198
  1.09360938  1.09604886  1.0938044   1.08733333  1.07667139  1.06186742
  1.04298335  1.02009413  0.99328765  0.96266457  0.92833818  0.8864522
  0.84477248  0.79981828  0.75175194  0.70074691  0.64698732  0.59066748
  0.53199146  0.47117246  0.40843228  0.33747354  0.27145539  0.20425122
  0.13611107  0.06729004 -0.00195257 -0.07135406 -0.14064919 -0.2095711
 -0.27785215 -0.35190197 -0.41796809 -0.48256911 -0.54544418 -0.60633657
 -0.66499465 -0.72117279 -0.7746324  -0.82514281 -0.87248222 -0.92064032
 -0.96064317 -0.9968533  -1.02909428 -1.05720274 -1.08102902 -1.10043779
 -1.11530857 -1.12553619 -1.13103112 -1.13152192 -1.12685776 -1.11728496
 -1.10277958 -1.08333334 -1.05895323 -1.0296609  -0.99549193 -0.95649484
 -0.91272999 -0.85916665 -0.80563042 -0.74757246 -0.6850847  -0.61826136
 -0.54719625 -0.47197978 -0.3926957  -0.30941736 -0.22220376 -0.12177096
 -0.02639613  0.07287195  0.17608333  0.28332912]

c: [ 17.53893754 -22.00647044 -21.06911983   7.03985659  33.8281617
  20.79696185 -38.38285113  -5.43884906   8.07411308  -0.0974112 ]
```



m=2, data_number=10, lamda=0

In [18]:
```python
cCalculation(2,'data10.txt',0)
```

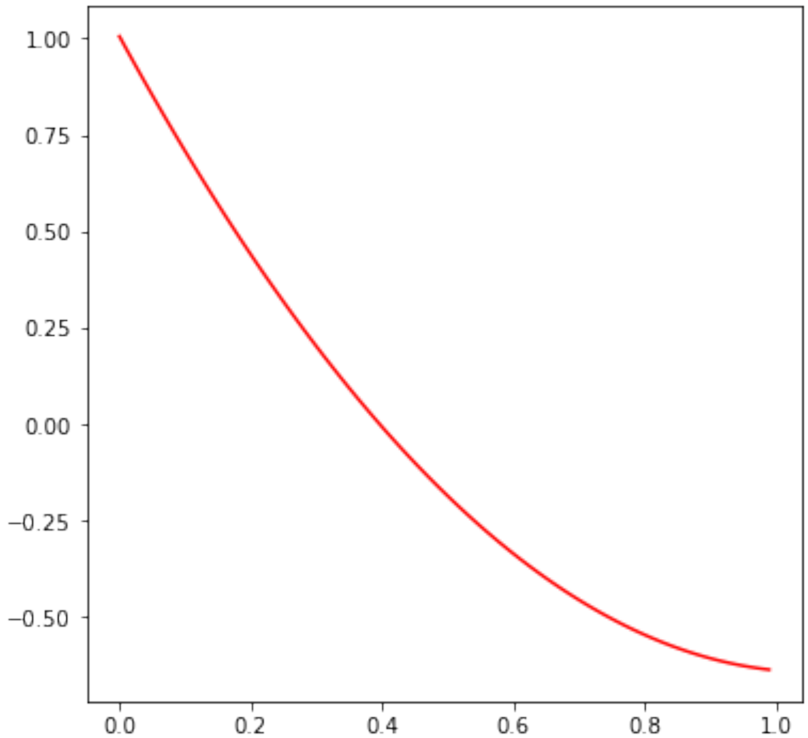
```
y_predict: [ 0.80276877  0.71571706  0.60171083  0.46075009  0.29283483  0.096
08696
 -0.12598016 -0.3750018  -0.65097795 -0.95390862]

c: [-1.09384447 -0.66283292  0.80276877]
```



m=2, data_number=15, lamda=0

In [19]:
```
cCalculation(2,'data15.txt',0)
```

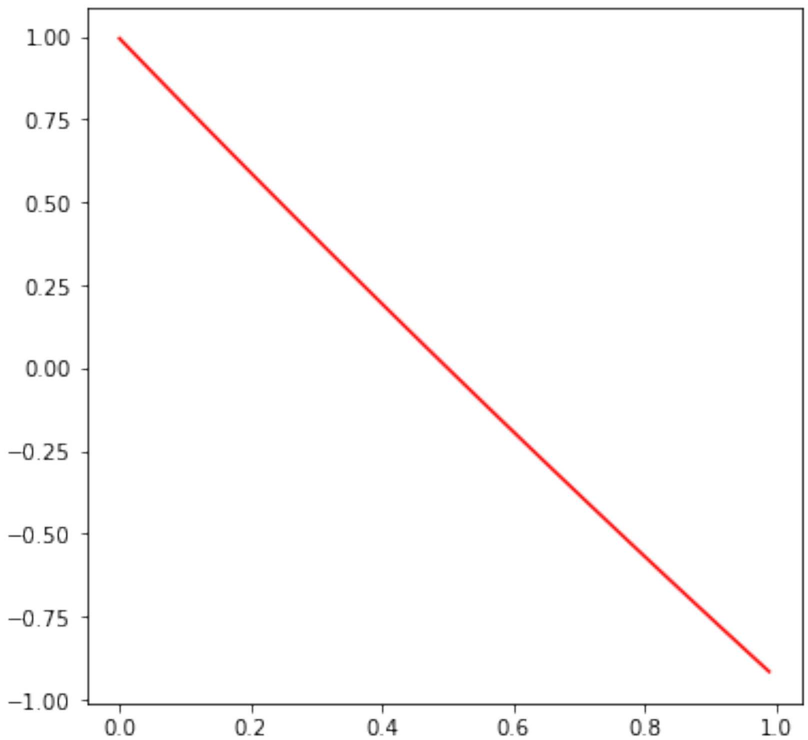
```
y_predict: [ 1.00544487  0.79013983  0.58936014  0.40522268  0.2337047   0.079
57059
 -0.06152145 -0.18565221 -0.2948862  -0.39044572 -0.46967636 -0.53480995
 -0.58403723 -0.61874488 -0.63796881]

c: [ 1.47756096 -3.12097465  1.00544487]
```



m=2, data_number=100, lamda=0

In [20]:
```
cCalculation(2,'data100.txt',0)
```

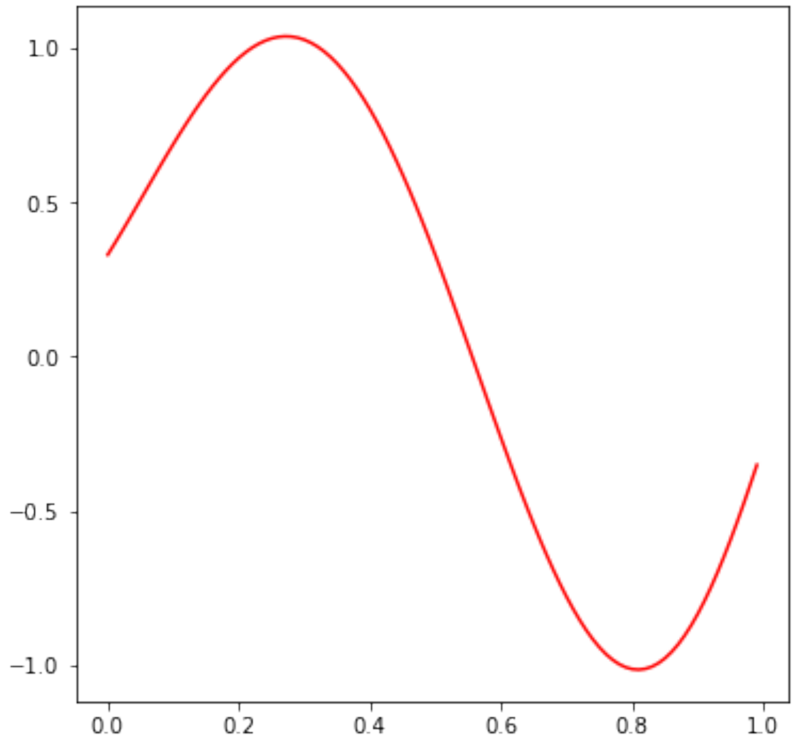
```
y_predict: [ 0.99435046  0.9736585   0.95299128  0.9323488   0.91173107  0.891
13809
  0.87056984  0.85002634  0.82950759  0.80901358  0.78854431  0.76830209
  0.74808412  0.72789041  0.70772096  0.68556258  0.66544407  0.64534981
  0.62527981  0.60523406  0.58521258  0.56521534  0.54524237  0.52529365
  0.50536919  0.4834803   0.46360677  0.44375751  0.4239325   0.40413174
  0.38435525  0.36460301  0.34487502  0.3251713   0.30549183  0.28387242
  0.26424389  0.24463962  0.2250596   0.20550384  0.18597233  0.16646508
  0.14698209  0.12752335  0.10808887  0.08673896  0.06735542  0.04799614
  0.02866111  0.00935034 -0.00993618 -0.02919843 -0.04843644 -0.06765018
 -0.08683967 -0.10792009 -0.12705864 -0.14617293 -0.16526297 -0.18432875
 -0.20337027 -0.22238754 -0.24138055 -0.26034931 -0.2792938  -0.30010473
 -0.31899829 -0.33786759 -0.35671264 -0.37553343 -0.39432996 -0.41310224
 -0.43185026 -0.45057402 -0.46927353 -0.48981497 -0.50846353 -0.52708784
 -0.5456879  -0.5642637  -0.58281524 -0.60134252 -0.61984555 -0.63832432
 -0.65677884 -0.67705079 -0.69545436 -0.71383368 -0.73218875 -0.75051955
 -0.7688261  -0.7871084  -0.80536644 -0.82360022 -0.84180974 -0.8618122
 -0.87997079 -0.89810511 -0.91621519 -0.934301  ]

c: [ 0.12128272 -2.04993418  0.99435046]
```



m=6, data_number=10, lamda=0

In [21]:
```
cCalculation(6,'data10.txt',0)
```

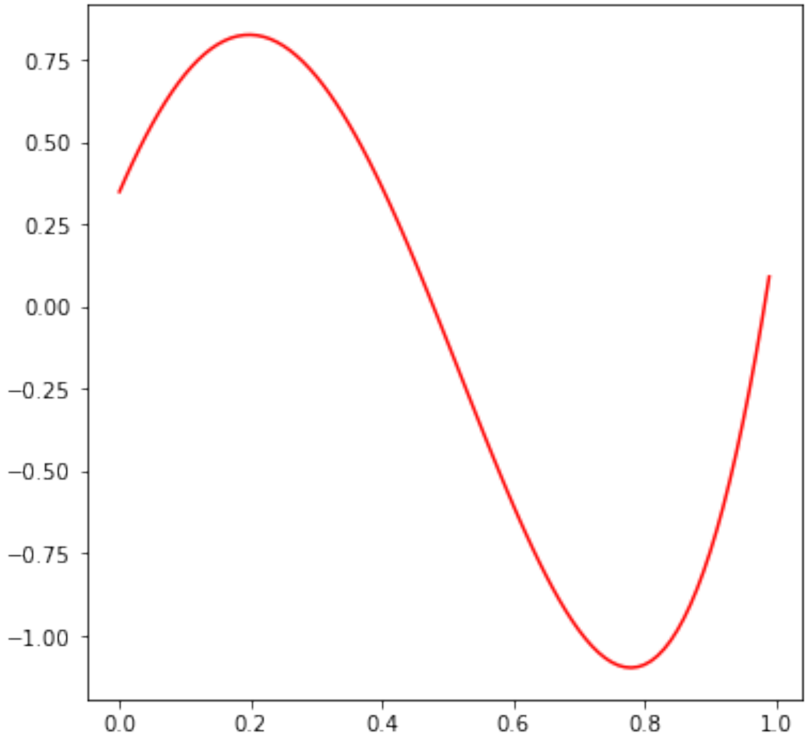
```
y_predict: [ 0.32921152  0.72536445  1.00259332  0.98282545  0.61605892 -0.002
59931
 -0.63222282 -0.99499334 -0.87448325 -0.28775494]

c: [-35.27877246  70.13917649 -14.44920489 -29.3399683    4.9174934
   3.39430931   0.32921152]
```



m=6, data_number=15, lamda=0

In [22]:
```
cCalculation(6,'data15.txt',0)
```

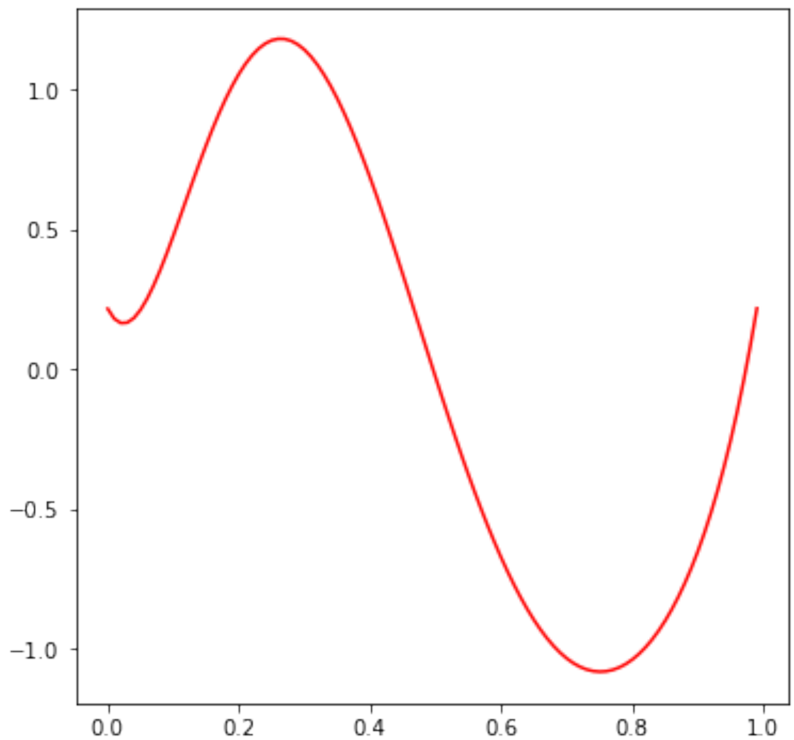
```
y_predict: [ 0.34863886  0.625894    0.78751577  0.82201963  0.73068628  0.529
0663
  0.23556548 -0.10837394 -0.46492725 -0.78954277 -1.01807349 -1.09615618
 -0.95897046 -0.53752069  0.21987843]

c: [ -1.25424225  -1.98660859  21.50872416 -13.83638845  -9.16016882
   4.59992351   0.34863886]
```



m=6, data_number=100, lamda=0

In [23]:
```
cCalculation(6,'data100.txt',0)
```

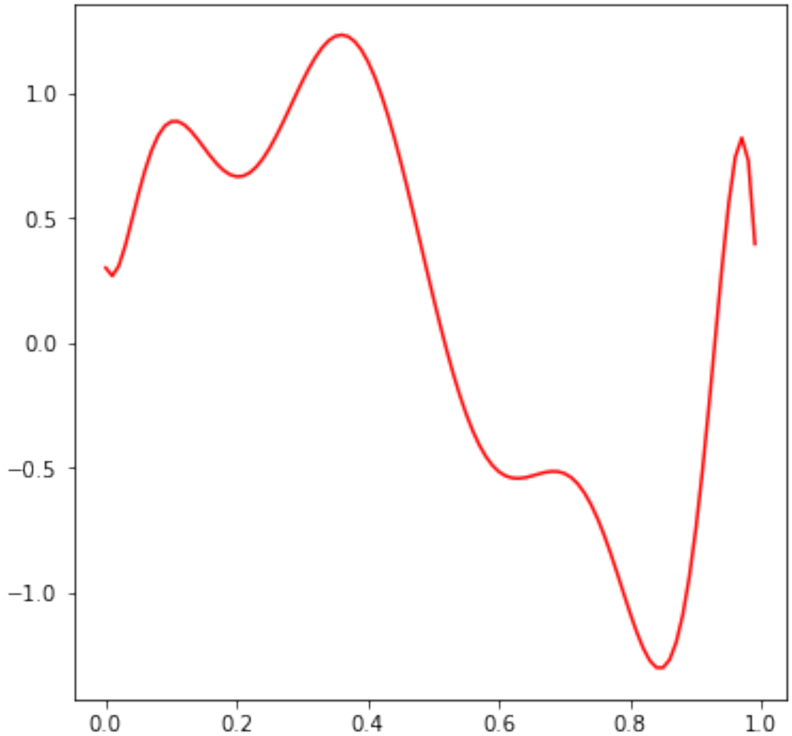
```
y_predict: [ 0.21653074  0.18079188  0.16535969  0.16750891  0.18470464  0.214
59484
  0.25500285  0.30392016  0.35949922  0.42004644  0.48401537  0.54934095
  0.61540671  0.68109961  0.74542303  0.81354274  0.87223118  0.927138
  0.97768408  1.02337817  1.06381175  1.0986539   1.12764643  1.15059904
  1.16738468  1.17864607  1.18232321  1.17979312  1.171142    1.15649757
  1.13602517  1.10992402  1.07842367  1.04178047  1.00027428  0.94935951
  0.89863856  0.8440377   0.78590111  0.72458148  0.66043757  0.59383173
  0.5251277   0.45468843  0.3828741   0.30271433  0.22916205  0.15531425
  0.08150167  0.0080441  -0.06475076 -0.13658839 -0.20718839 -0.27628525
 -0.34362904 -0.41540384 -0.47832479 -0.53882187 -0.59671317 -0.65183367
 -0.70403516 -0.75318592 -0.79917036 -0.84188845 -0.88125506 -0.92060302
 -0.95271607 -0.98129913 -1.00631841 -1.02774958 -1.04557629 -1.05978859
 -1.07038113 -1.07735138 -1.08069754 -1.08018866 -1.07590965 -1.06798217
 -1.0563839  -1.04108103 -1.02202548 -0.99915193 -0.97237478 -0.94158492
 -0.90664642 -0.86322303 -0.8189908  -0.76997989 -0.71590672 -0.65644183
 -0.59120575 -0.5197647  -0.44162622 -0.35623462 -0.26296631 -0.15044027
 -0.03827168  0.08419082  0.21790077  0.36390549]

c: [ 1.83364292e+02 -6.27665357e+02  8.24548712e+02 -4.89239289e+02
  1.13777608e+02 -4.63859121e+00  2.16530744e-01]
```



m=9, data_number=10, lamda=0

In [24]:
```
cCalculation(9,'data10.txt',0)
```

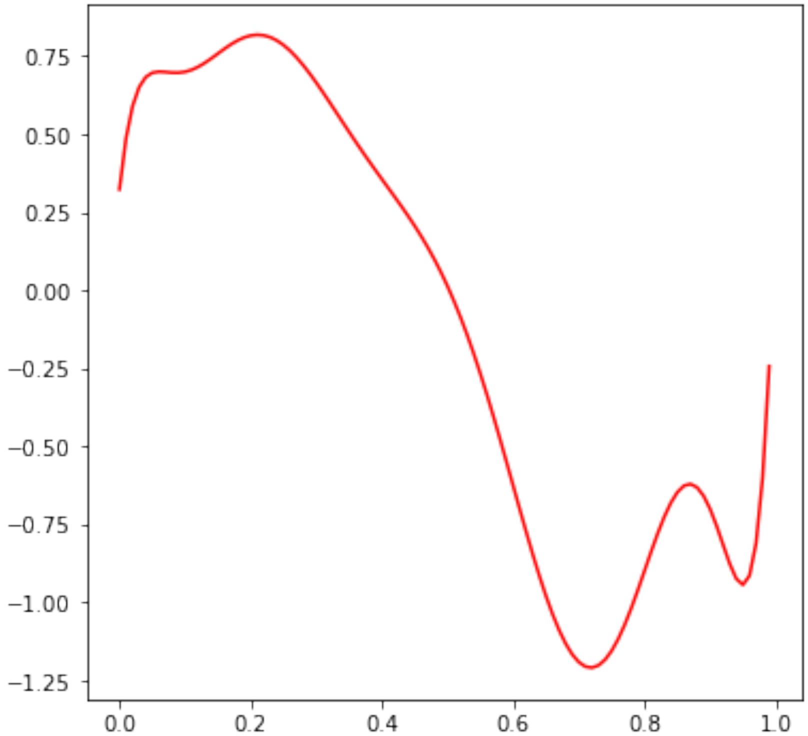
```
y_predict: [ 0.29999643  0.88502911  0.68489416  1.19022593  0.78268675 -0.328
71933
 -0.52019604 -0.90594892 -0.95406147 -0.27023967]

c: [-3.90366240e+04  1.66965375e+05 -2.95037663e+05  2.77784787e+05
 -1.49717413e+05  4.60874184e+04 -7.60300497e+03  5.64761774e+02
 -8.20655012e+00  2.99996426e-01]
```



m=9, data_number=15, lamda=0

In [25]:
```
cCalculation(9,'data15.txt',0)
```

```
y_predict: [ 0.32266937  0.69853493  0.74691844  0.81782954  0.70471203  0.484
75722
  0.27071138  0.01231392 -0.41227747 -0.93570244 -1.20841443 -0.98357985
 -0.63131972 -0.87124438  0.30976435]

c: [ 1.75224232e+04 -7.39433217e+04  1.29498703e+05 -1.22037802e+05
  6.71343619e+04 -2.18695730e+04  4.08377423e+03 -4.08625454e+02
  2.00467830e+01  3.22669370e-01]
```

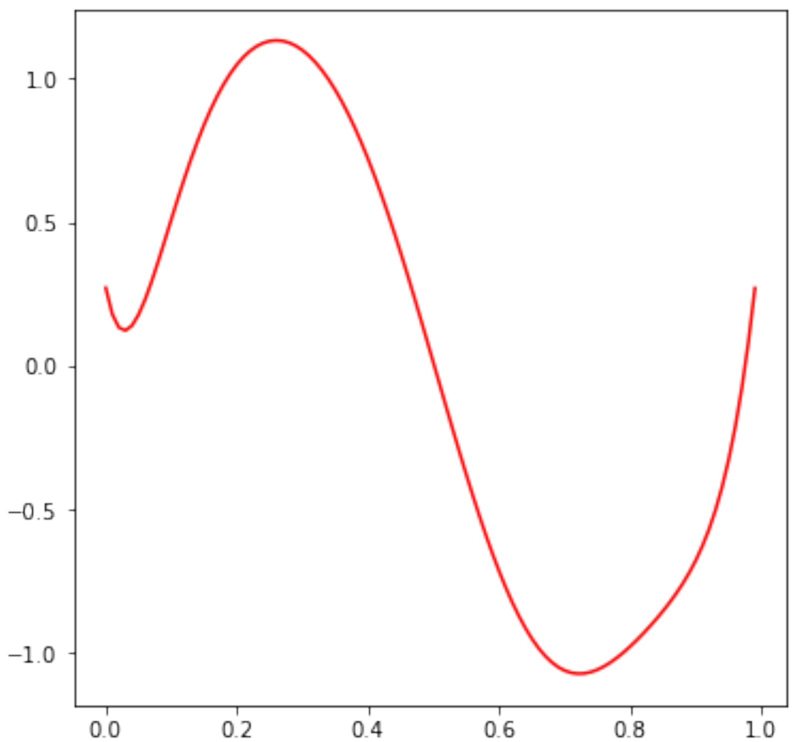


m=9, data_number=100, lamda=0

In [26]:
```
cCalculation(9,'data100.txt',0)
```

```
y_predict: [ 0.27158292  0.17889524  0.13267156  0.12286915  0.14102939  0.180
08197
  0.23416741  0.29847636  0.36910492  0.44292452  0.51746572  0.59009925
  0.66015381  0.72658862  0.78866397  0.85132377  0.90285155  0.94904794
  0.98987393  1.0253671   1.05561647  1.08074206  1.10087876  1.11616393
  1.12672836  1.13303709  1.1340522   1.13065669  1.12291417  1.11087209
  1.09456456  1.07401601  1.04924546  1.02027124  0.9871158   0.94585346
  0.90403647  0.85818593  0.80838999  0.75476301  0.69744847  0.63662129
  0.57248947  0.50529498  0.4353139   0.3554868   0.28070037  0.20419
  0.12635809  0.04763093 -0.03154522 -0.11070864 -0.18938708 -0.26710302
 -0.34337934 -0.42505872 -0.49679337 -0.56567479 -0.63128875 -0.69325118
 -0.75121359 -0.80486806 -0.85395179 -0.89825104 -0.93760438 -0.97505498
 -1.00374234 -1.02733784 -1.04591087 -1.05958591 -1.0685398  -1.07299778
 -1.07322815 -1.06953576 -1.06225395 -1.05051928 -1.03685699 -1.02071121
 -1.00241965 -0.98228651 -0.96056625 -0.93744617 -0.91302785 -0.88730747
 -0.86015539 -0.82829915 -0.79703686 -0.76290749 -0.72503681 -0.6823116
 -0.63335719 -0.57651562 -0.50982452 -0.43099722 -0.33740421 -0.21381916
 -0.07904279  0.0809954   0.27047729  0.49398385]
```

```
c: [-2.21760981e+03  1.15794264e+04 -2.48690007e+04  2.86932303e+04
 -1.95216513e+04  8.13293966e+03 -2.07021513e+03  2.84955010e+02
 -1.18520406e+01  2.71582923e-01]
```



1)When data is scarce, I recommend to use low degree polynomials. Because using high degree polynomials will lead to overfitting. 2)Regularization can help smoothing the curve.

(c)

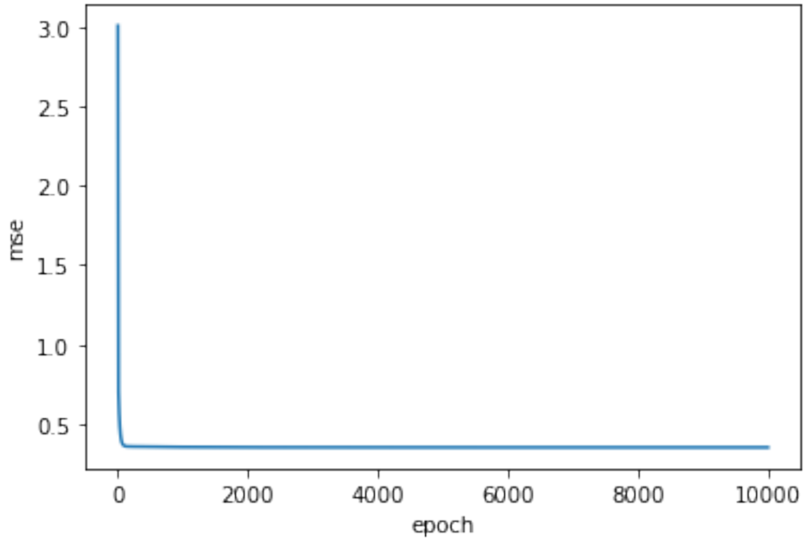
dE(c)/d(c)=2A'Ac-2A'b+2*lambda*I*c

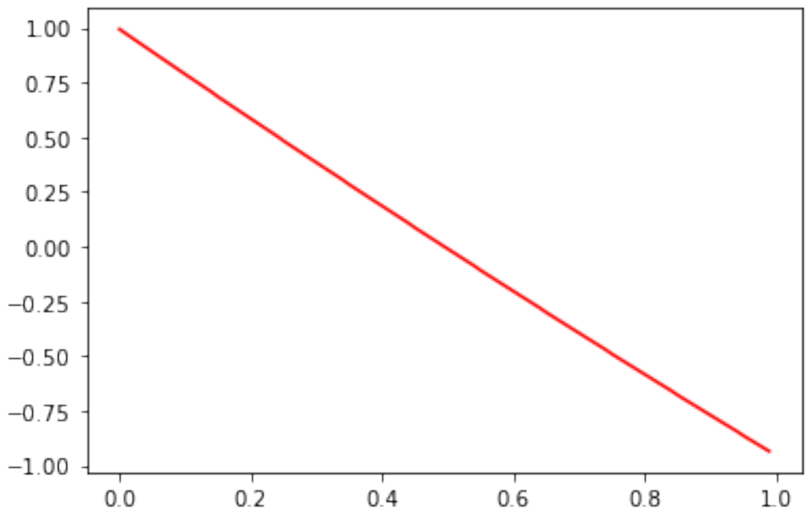
In [27]:
```
def batch_GD(m,filename,lamda,epochs,learning_rate):
    data=np.loadtxt(filename)
    A=np.empty(shape=(len(data),0))
    for i in range(m,-1,-1):
        x=((data[:,0])**i).reshape(len(data),1)
        A=np.hstack((A,x))
    b=(data[:,1]).reshape(len(data),1)
    I=np.eye(m+1,m+1)
    I[m,m]=0
    c=np.ones([m+1,1])
    mse_list=[]
    epoch_list=[]
    for i in range(epochs):
        c=c-learning_rate*(2*(A.T)@A@c-2*(A.T)@b+2*lamda*I@c)
        mse=np.mean(np.square(b-A@c))

        if i%10==0:
            mse_list.append(mse)
            epoch_list.append(i)
    return c,mse,mse_list,epoch_list,A
```

m=2, data_num=100, lambda=0, epoch=10000, learning_rate=0.001

In [28]:
```
c,mse,mse_list,epoch_list,A=batch_GD(2,"data100.txt",0,10000,0.001)
plt.figure(1)
plt.xlabel("epoch")
plt.ylabel("mse")
plt.plot(epoch_list,mse_list)
x=np.arange(0,1,0.01)
Y=A@c
plt.figure(2)
plt.plot(x,Y,c="r")
```

Out[28]: [<matplotlib.lines.Line2D at 0x7f7a80a629d0>]

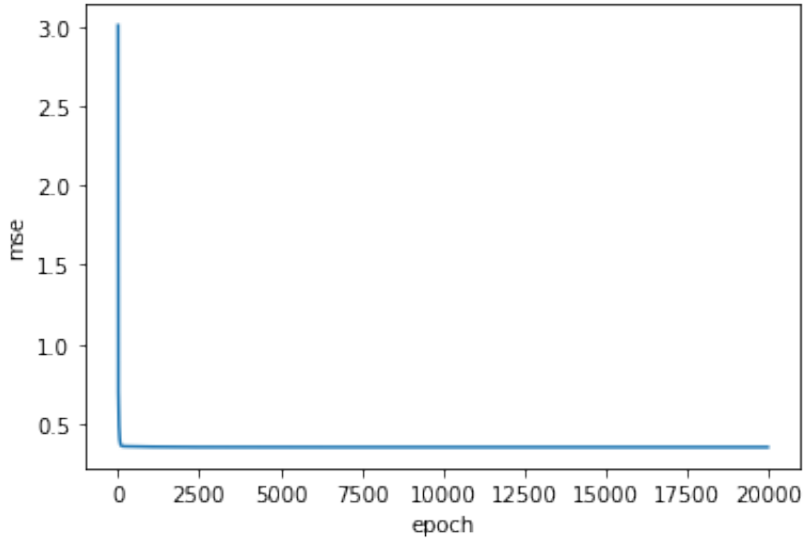


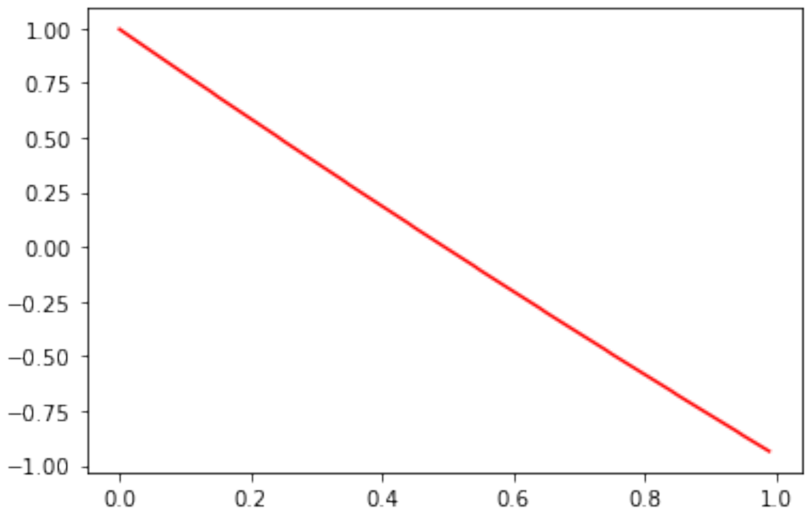


m=2, data_num=100, lambda=exp(-10), epoch=10000, learning_rate=0.001

In [29]:
```
c,mse,mse_list,epoch_list,A=batch_GD(2,"data100.txt",np.exp(-10),20000,0.001)
plt.figure(1)
plt.xlabel("epoch")
plt.ylabel("mse")
plt.plot(epoch_list,mse_list)
x=np.arange(0,1,0.01)
Y=A@c
plt.figure(2)
plt.plot(x,Y,c="r")
```

Out[29]: [<matplotlib.lines.Line2D at 0x7f7a51d70610>]

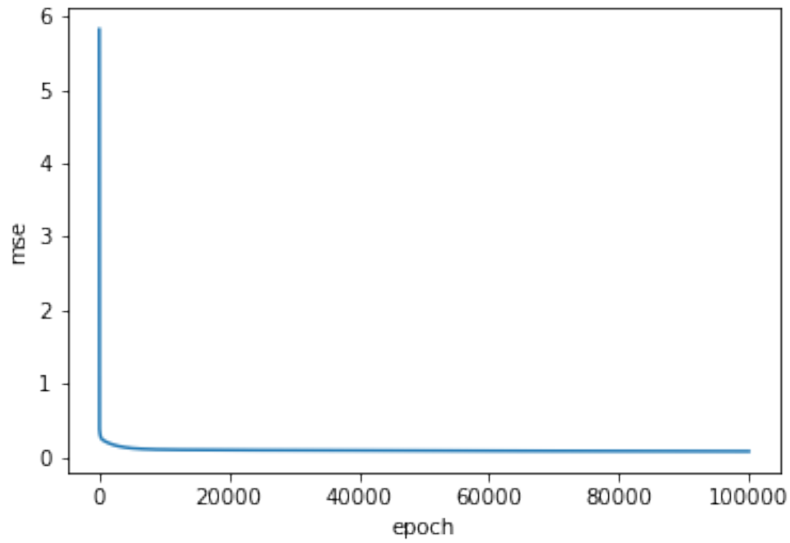


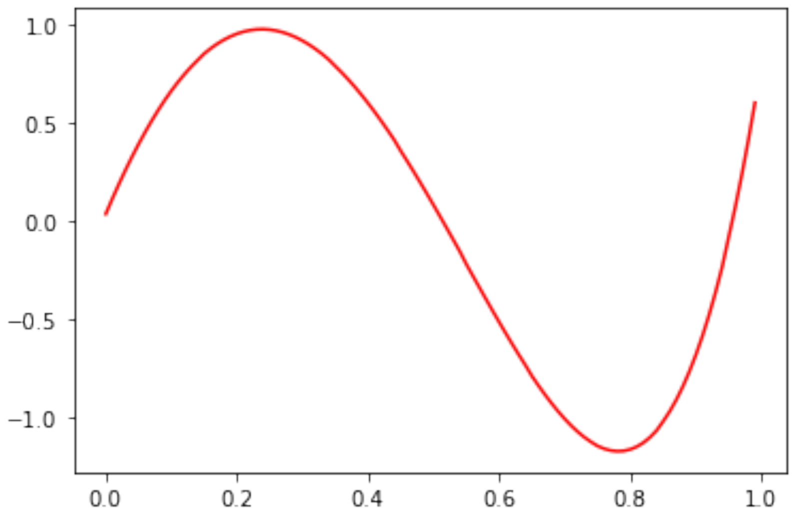


m=6, data_num=100, lambda=0, epoch=100000, learning_rate=0.001

In [30]:
```
c,mse,mse_list,epoch_list,A=batch_GD(6,"data100.txt",0,100000,0.001)
plt.figure(1)
plt.xlabel("epoch")
plt.ylabel("mse")
plt.plot(epoch_list,mse_list)
x=np.arange(0,1,0.01)
Y=A@c
plt.figure(2)
plt.plot(x,Y,c="r")
```

Out[30]: [<matplotlib.lines.Line2D at 0x7f7a3001d160>]

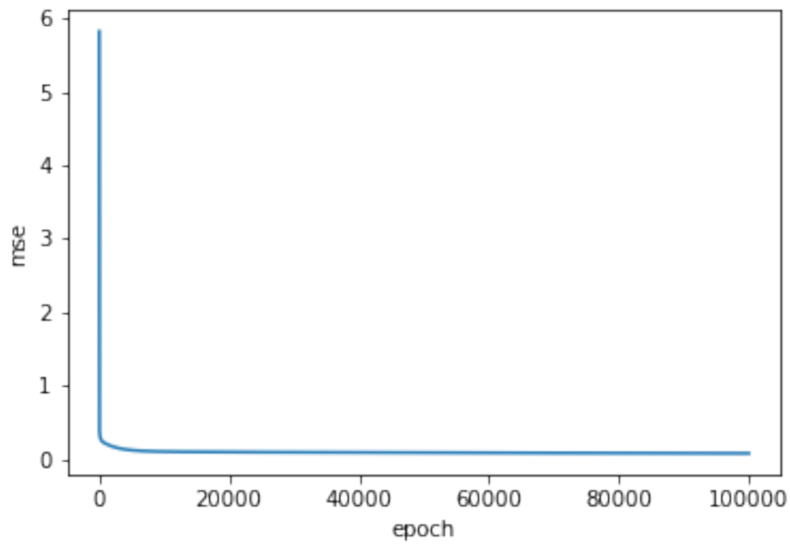


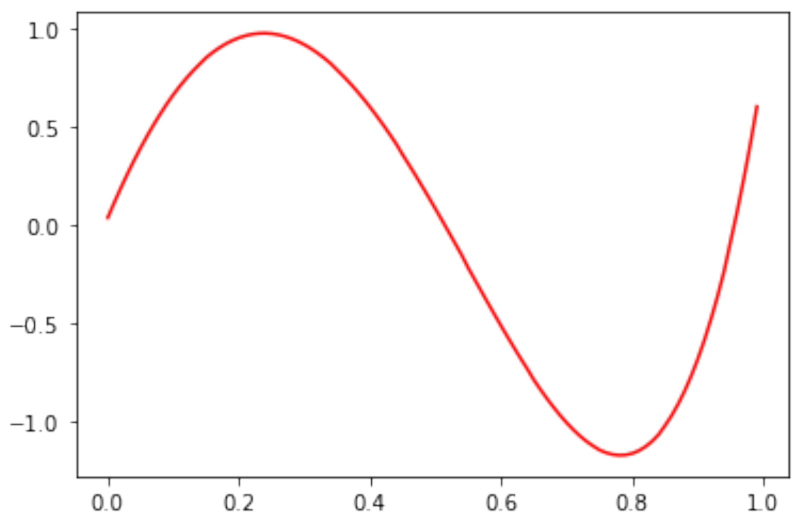


m=6, data_num=100, lambda=exp(-10), epoch=100000, learning_rate=0.001

In [31]:
```
c,mse,mse_list,epoch_list,A=batch_GD(6,"data100.txt",np.exp(-10),100000,0.001
plt.figure(1)
plt.xlabel("epoch")
plt.ylabel("mse")
plt.plot(epoch_list,mse_list)
x=np.arange(0,1,0.01)
Y=A@c
plt.figure(2)
plt.plot(x,Y,c="r")
```

Out[31]: [<matplotlib.lines.Line2D at 0x7f7a700a24c0>]

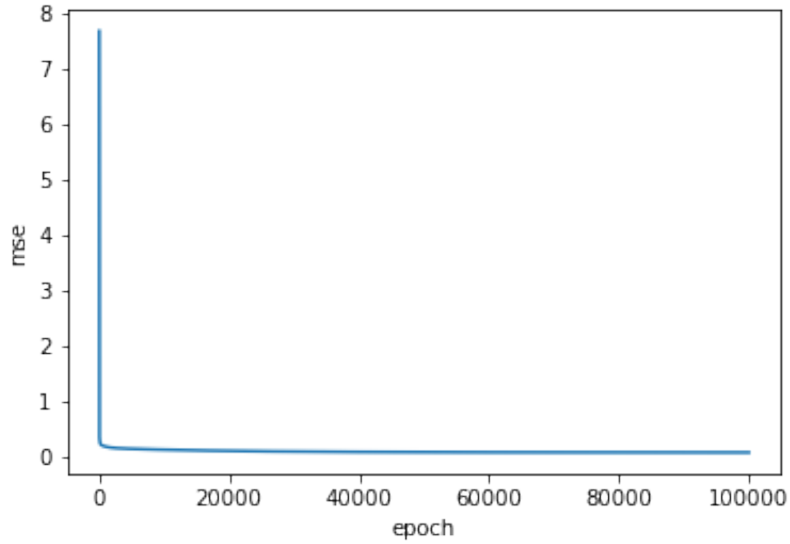


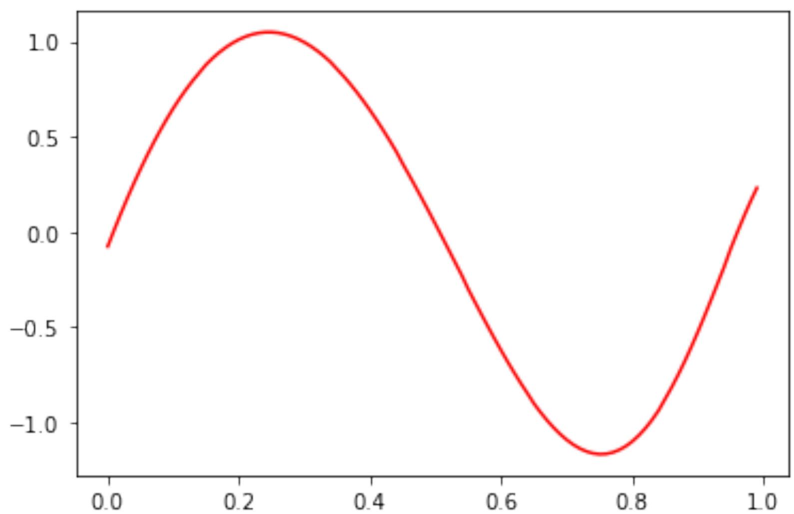


m=9, data_num=100, lambda=0, epoch=100000, learning_rate=0.001

In [32]:
```
c,mse,mse_list,epoch_list,A=batch_GD(9,"data100.txt",0,100000,0.001)
plt.figure(1)
plt.xlabel("epoch")
plt.ylabel("mse")
plt.plot(epoch_list,mse_list)
x=np.arange(0,1,0.01)
Y=A@c
plt.figure(2)
plt.plot(x,Y,c="r")
```

Out[32]: [<matplotlib.lines.Line2D at 0x7f7a30071700>]



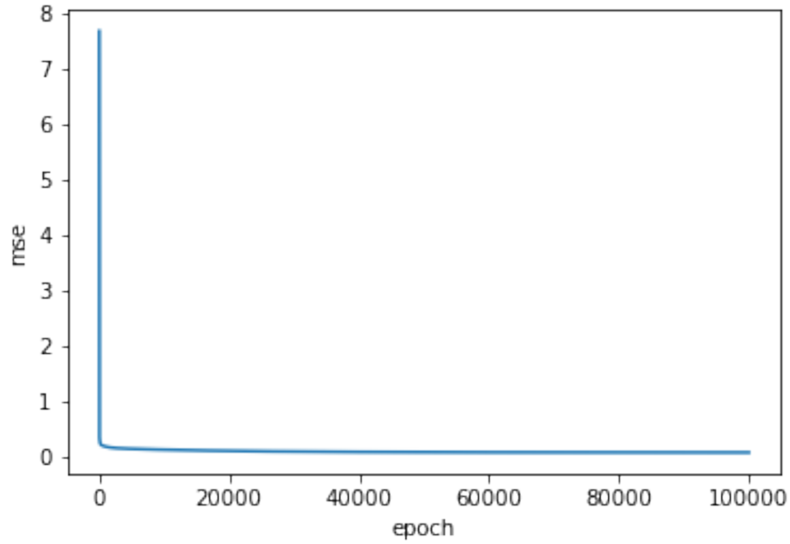


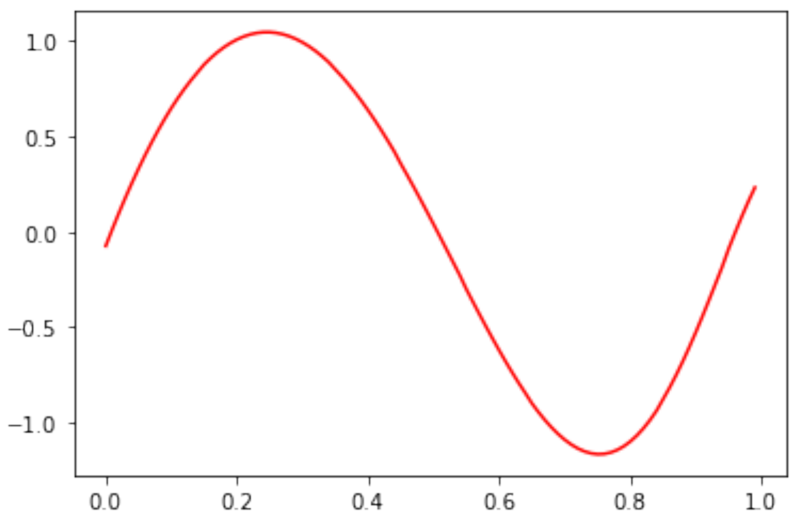
m=9, data_num=100, lambda=exp(-10), epoch=100000, learning_rate=0.001

In [33]:

```
c,mse,mse_list,epoch_list,A=batch_GD(9,"data100.txt",np.exp(-10),100000,0.001
plt.figure(1)
plt.xlabel("epoch")
plt.ylabel("mse")
plt.plot(epoch_list,mse_list)
x=np.arange(0,1,0.01)
Y=A@c
plt.figure(2)
plt.plot(x,Y,c="r")
```

Out[33]:
```
[<matplotlib.lines.Line2D at 0x7f7a80a8aa30>]
```





I think this gradient descent can not get stuck in a local minimum. Because this is linear regression model so that E(c) is always quadratic of c which means there is only one global minimum.

In [ ]:

Q3 Temperature Field

(a)

In [2]:
```
import numpy as np
```

In [22]:
```
A=np.array([[0,0,0,1],[8,6,1,1],[5,2,8,1],[8,2,6,1],
            [5,1,2,1],[3,3,3,1],[9,8,2,1],[3,6,5,1],
            [4,6,9,1],[1,8,2,1],[1,1,2,1],[6,4,2,1]])
```

In [23]:
```
b=np.array([10,15,20,22,16,23,18,19,25,20,28,27])
```

In [24]:
```
I_mod=np.eye(4,4)
I_mod[3,3]=0
```

I calculate

In [25]:
```
c=(np.linalg.inv(((A.T).dot(A)+np.exp(-10)*I_mod))).dot((A.T)).dot(b)
```

In [26]:
```
T_5_5_5=c[0]*5+c[1]*5+c[2]*5+c[3]
```

In [27]:
```
T_5_5_5
```

Out[27]:
```
21.389072800503914
```

I apply L2 regularization when solve the equation, and predict the temperature in (5,5,5) is 21.389072800503914 Celsius

(b)

Direction should be minus gradient of T(x,y,z), and we should normalize it

In [36]:
```
d=np.array([c[0],c[1],c[2]])
```

In [37]:
```
unit=-d/(np.sqrt(c[0]**2+c[1]**2+c[2]**2))
```

In [38]:
```
unit
```

Out[38]:
```
array([ 0.17934912, -0.15690718, -0.97119207])
```

In [ ]:

In [ ]: