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

c_data1 = pd.read_csv("climate_change_1.csv")
c_data1
```

Out[2]:

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols	Temp
0	1983	5	2.556	345.96	1638.59	303.677	191.324	350.113	1366.1024	0.0863	0.109
1	1983	6	2.167	345.52	1633.71	303.746	192.057	351.848	1366.1208	0.0794	0.118
2	1983	7	1.741	344.15	1633.22	303.795	192.818	353.725	1366.2850	0.0731	0.137
3	1983	8	1.130	342.25	1631.35	303.839	193.602	355.633	1366.4202	0.0673	0.176
4	1983	9	0.428	340.17	1648.40	303.901	194.392	357.465	1366.2335	0.0619	0.149
5	1983	10	0.002	340.30	1663.79	303.970	195.171	359.174	1366.0589	0.0569	0.093
6	1983	11	-0.176	341.53	1658.23	304.032	195.921	360.758	1366.1072	0.0524	0.232
7	1983	12	-0.176	343.07	1654.31	304.082	196.609	362.174	1366.0607	0.0486	0.078
8	1984	1	-0.339	344.05	1658.98	304.130	197.219	363.359	1365.4261	0.0451	0.089
9	1984	2	-0.565	344.77	1656.48	304.194	197.759	364.296	1365.6618	0.0416	0.013
10	1984	3	0.131	345.46	1655.77	304.285	198.249	365.044	1366.1697	0.0383	0.049
11	1984	4	0.331	346.77	1657.68	304.389	198.723	365.692	1365.5660	0.0352	-0.019
12	1984	5	0.121	347.55	1649.33	304.489	199.233	366.317	1365.7783	0.0324	0.065
13	1984	6	-0.142	346.98	1634.13	304.593	199.858	367.029	1366.0956	0.0302	-0.016
14	1984	7	-0.138	345.55	1629.89	304.722	200.671	367.893	1366.1145	0.0282	-0.024
15	1984	8	-0.179	343.20	1643.67	304.871	201.710	368.843	1365.9781	0.0260	0.034
16	1984	9	-0.082	341.35	1663.60	305.021	202.972	369.800	1365.8669	0.0239	0.025
17	1984	10	0.016	341.68	1674.65	305.158	204.407	370.782	1365.7869	0.0220	-0.035
18	1984	11	-0.351	343.06	1677.10	305.263	205.893	371.770	1365.6802	0.0202	-0.123
19	1984	12	-0.611	344.54	1672.15	305.313	207.308	372.701	1365.7617	0.0188	-0.282
20	1985	1	-0.561	345.25	1663.42	305.301	208.537	373.623	1365.6082	0.0164	-0.001

21	1985	2	-0.602	346.06	1666.21	305.243	209.543	374.681	1365.7085	0.0160	-0.155
22	1985	3	-0.737	347.66	1678.34	305.165	210.368	376.004	1365.6570	0.0141	-0.032
23	1985	4	-0.484	348.20	1675.24	305.093	211.111	377.635	1365.5120	0.0138	-0.042
24	1985	5	-0.731	348.92	1666.83	305.045	211.823	379.539	1365.6366	0.0128	0.001
25	1985	6	-0.086	348.40	1659.40	305.027	212.512	381.642	1365.6964	0.0126	-0.049
26	1985	7	-0.156	346.66	1654.25	305.049	213.165	383.905	1365.6509	0.0121	-0.042
27	1985	8	-0.392	344.85	1654.41	305.126	213.803	386.223	1365.7499	0.0116	0.013
28	1985	9	-0.541	343.20	1668.31	305.250	214.501	388.500	1365.6653	0.0102	-0.035
29	1985	10	-0.140	343.08	1681.56	305.395	215.327	390.676	1365.5269	0.0101	-0.008
...	...	...	...	...	...	...	...	...	...	...	...
278	2006	7	0.628	382.38	1765.95	319.872	249.247	539.725	1365.8212	0.0038	0.456
279	2006	8	0.759	380.45	1762.66	319.930	248.981	539.682	1365.7067	0.0041	0.482
280	2006	9	0.793	378.92	1776.04	320.010	248.775	539.566	1365.8419	0.0043	0.425
281	2006	10	0.892	379.16	1789.02	320.125	248.666	539.488	1365.8270	0.0044	0.472
282	2006	11	1.292	380.18	1791.91	320.321	248.605	539.500	1365.7039	0.0049	0.440
283	2006	12	0.951	381.79	1795.04	320.451	248.480	539.377	1365.7087	0.0054	0.518
284	2007	1	0.974	382.93	1799.66	320.561	248.372	539.206	1365.7173	0.0054	0.601
285	2007	2	0.510	383.81	1803.08	320.571	248.264	538.973	1365.7145	0.0051	0.498
286	2007	3	0.074	384.56	1803.10	320.548	247.997	538.811	1365.7544	0.0045	0.435
287	2007	4	-0.049	386.40	1802.11	320.518	247.574	538.586	1365.7228	0.0045	0.466
288	2007	5	0.183	386.58	1795.65	320.445	247.224	538.130	1365.6932	0.0041	0.372
289	2007	6	-0.358	386.05	1781.81	320.332	246.881	537.376	1365.7616	0.0040	0.382
290	2007	7	-0.290	384.49	1771.89	320.349	246.497	537.113	1365.7506	0.0040	0.394
291	2007	8	-0.440	382.00	1779.38	320.471	246.307	537.125	1365.7566	0.0041	0.358
292	2007	9	-1.162	380.90	1794.21	320.618	246.214	537.281	1365.7159	0.0042	0.402
293	2007	10	-1.142	381.14	1802.38	320.855	246.189	537.380	1365.7388	0.0041	0.362
294	2007	11	-1.177	382.42	1803.79	321.062	246.178	537.319	1365.6680	0.0042	0.266
295	2007	12	-1.168	383.89	1805.58	321.217	246.261	537.052	1365.6927	0.0040	0.226
296	2008	1	-1.011	385.44	1809.92	321.328	246.183	536.876	1365.7163	0.0038	0.074

292	2007	9	-1.162	380.90	1794.21	320.618	246.214	537.281	1365.7159	0.0042	0.402
293	2007	10	-1.142	381.14	1802.38	320.855	246.189	537.380	1365.7388	0.0041	0.362
294	2007	11	-1.177	382.42	1803.79	321.062	246.178	537.319	1365.6680	0.0042	0.266
295	2007	12	-1.168	383.89	1805.58	321.217	246.261	537.052	1365.6927	0.0040	0.226
296	2008	1	-1.011	385.44	1809.92	321.328	246.183	536.876	1365.7163	0.0038	0.074

297	2008	2	-1.402	385.73	1803.45	321.345	245.898	536.484	1365.7366	0.0036	0.198
298	2008	3	-1.635	385.97	1792.84	321.295	245.430	535.979	1365.6726	0.0034	0.447
299	2008	4	-0.942	387.16	1792.57	321.354	245.086	535.648	1365.7146	0.0033	0.278
300	2008	5	-0.355	388.50	1796.43	321.420	244.914	535.399	1365.7175	0.0031	0.283
301	2008	6	0.128	387.88	1791.80	321.447	244.676	535.128	1365.6730	0.0031	0.315
302	2008	7	0.003	386.42	1782.93	321.372	244.434	535.026	1365.6720	0.0033	0.406
303	2008	8	-0.266	384.15	1779.88	321.405	244.200	535.072	1365.6570	0.0036	0.407
304	2008	9	-0.643	383.09	1795.08	321.529	244.083	535.048	1365.6647	0.0043	0.378
305	2008	10	-0.780	382.99	1814.18	321.796	244.080	534.927	1365.6759	0.0046	0.440
306	2008	11	-0.621	384.13	1812.37	322.013	244.225	534.906	1365.7065	0.0048	0.394
307	2008	12	-0.666	385.56	1812.88	322.182	244.204	535.005	1365.6926	0.0046	0.330

308 rows × 11 columns

Problem 1

In [3]:
```
c_data1_train = c_data1.iloc[0:284]
# traning data set inclueds up to and including the data of 2006
c_data1_test = c_data1.iloc[284:308]
# testing data set
c_data1_test.head()
```

Out[3]:

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols	Temp
284	2007	1	0.974	382.93	1799.66	320.561	248.372	539.206	1365.7173	0.0054	0.601
285	2007	2	0.510	383.81	1803.08	320.571	248.264	538.973	1365.7145	0.0051	0.498
286	2007	3	0.074	384.56	1803.10	320.548	247.997	538.811	1365.7544	0.0045	0.435
287	2007	4	-0.049	386.40	1802.11	320.518	247.574	538.586	1365.7228	0.0045	0.466
288	2007	5	0.183	386.58	1795.65	320.445	247.224	538.130	1365.6932	0.0041	0.372

In [4]:
```
def closed_form_1(x_train,y_train):
    x_train["constant"]=1
    col = x_train.columns
    x_mat = np.matrix(x_train).T
    y_mat = np.matrix(y_train).T
    x = x_mat.T
    coeff = np.dot(np.dot(x_mat,x).I,np.dot(x_mat,y_mat))
    result = pd.Series(coeff.A1,index=col, name="coeff")
    return result
coeff = closed_form_1(c_data1_train.loc[:,"MEI":"Aerosols"], c_data1_train.Temp)
coeff
```

Out[4]:
```
MEI            0.064205
CO2            0.006457
CH4            0.000124
N2O           -0.016528
CFC-11        -0.006630
CFC-12         0.003808
TSI            0.093141
Aerosols      -1.537613
constant    -124.594261
Name: coeff, dtype: float64
```

In [15]:
```
def r2(coeff,x_train,y_train):
    coeff = np.matrix(coeff)
    x_train["constant"]=1
    x_train=np.matrix(x_train)
    y_train=y_train.values
    y_train_hat = (np.dot(x_train,coeff.T)).A1
    cov_yymean = ((y_train_hat-y_train_hat.mean())**2).sum()
    var_y = ((y_train-y_train.mean()**2)).sum()
    r2 = cov_yymean/var_y
    return r2

r2_train = r2(coeff,c_data1_train.loc[:,"MEI":"Aerosols"],c_data1_train.Temp)
r2_test = r2(coeff,c_data1_test.loc[:,"MEI":"Aerosols"],c_data1_test.Temp)
print('The r2 of train data is',r2_train)
print('The r2 of test data is',r2_test)
```

```
('The r2 of train data is', 0.1317103586452159)
('The r2 of test data is', 0.009032992652029991)
```

In [21]:
```
def t_value(coeff,x_train,y_train):
    x_train["constant"]=1
    coeff = np.matrix(coeff)
    index = x_train.columns
    x_train = np.matrix(x_train)
    y_train = np.matrix(y_train).T
    y_train_hat = np.dot(x_train,coeff.T)
    s2 = np.power(y_train-y_train_hat,2).sum()/(x_train.shape[0]-x_train.shape[1])
    t = coeff/np.power(s2*(np.dot(x_train.T,x_train)).I.diagonal(),0.5)
    result=pd.Series(t.A1, index=index, name="tvalue")
    return result

t_value(coeff,c_data1_train.loc[:,"MEI":"Aerosols"],c_data1_train.Temp)
```

Out[21]:
```
MEI          9.923226
CO2          2.826420
CH4          0.240469
N2O         -1.929726
CFC-11      -4.077834
CFC-12       3.757293
TSI          6.312561
Aerosols    -7.210301
constant    -6.265174
Name: tvalue, dtype: float64
```

According to t value, MEI, CO2 , N2O, TSI, Aerosols,constact can be regarded as siginificant value.

In [23]:
```
c_data2 = pd.read_csv("climate_change_2.csv")
c_data2
```

Out[23]:

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols	NO	Temp
0	1983	5	2.556	345.96	1638.59	303.677	191.324	350.113	1366.1024	0.0863	2.63859	0.109
1	1983	6	2.167	345.52	1633.71	303.746	192.057	351.848	1366.1208	0.0794	2.63371	0.118
2	1983	7	1.741	344.15	1633.22	303.795	192.818	353.725	1366.2850	0.0731	2.63322	0.137
3	1983	8	1.130	342.25	1631.35	303.839	193.602	355.633	1366.4202	0.0673	2.63135	0.176
4	1983	9	0.428	340.17	1648.40	303.901	194.392	357.465	1366.2335	0.0619	2.64840	0.149
5	1983	10	0.002	340.30	1663.79	303.970	195.171	359.174	1366.0589	0.0569	2.66379	0.093
6	1983	11	-0.176	341.53	1658.23	304.032	195.921	360.758	1366.1072	0.0524	2.65823	0.232
7	1983	12	-0.176	343.07	1654.31	304.082	196.609	362.174	1366.0607	0.0486	2.65431	0.078
8	1984	1	-0.339	344.05	1658.98	304.130	197.219	363.359	1365.4261	0.0451	2.65898	0.089
9	1984	2	-0.565	344.77	1656.48	304.194	197.759	364.296	1365.6618	0.0416	2.65648	0.013
10	1984	3	0.131	345.46	1655.77	304.285	198.249	365.044	1366.1697	0.0383	2.65577	0.049
11	1984	4	0.331	346.77	1657.68	304.389	198.723	365.692	1365.5660	0.0352	2.65768	-0.019
12	1984	5	0.121	347.55	1649.33	304.489	199.233	366.317	1365.7783	0.0324	2.64933	0.065
13	1984	6	-0.142	346.98	1634.13	304.593	199.858	367.029	1366.0956	0.0302	2.63413	-0.016
14	1984	7	-0.138	345.55	1629.89	304.722	200.671	367.893	1366.1145	0.0282	2.62989	-0.024
15	1984	8	-0.179	343.20	1643.67	304.871	201.710	368.843	1365.9781	0.0260	2.64367	0.034
16	1984	9	-0.082	341.35	1663.60	305.021	202.972	369.800	1365.8669	0.0239	2.66360	0.025
17	1984	10	0.016	341.68	1674.65	305.158	204.407	370.782	1365.7869	0.0220	2.67465	-0.035
18	1984	11	-0.351	343.06	1677.10	305.263	205.893	371.770	1365.6802	0.0202	2.67710	-0.123
19	1984	12	-0.611	344.54	1672.15	305.313	207.308	372.701	1365.7617	0.0188	2.67215	-0.282
20	1985	1	-0.561	345.25	1663.42	305.301	208.537	373.623	1365.6082	0.0164	2.66342	-0.001

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols	NO	Temp
299	2008	4	-0.942	387.16	1792.57	321.354	245.086	535.648	1365.7146	0.0033	2.79257	0.278
300	2008	5	-0.355	388.50	1796.43	321.420	244.914	535.399	1365.7175	0.0031	2.79643	0.283
301	2008	6	0.128	387.88	1791.80	321.447	244.676	535.128	1365.6730	0.0031	2.79180	0.315
302	2008	7	0.003	386.42	1782.93	321.372	244.434	535.026	1365.6720	0.0033	2.78293	0.406
303	2008	8	-0.266	384.15	1779.88	321.405	244.200	535.072	1365.6570	0.0036	2.77988	0.407
304	2008	9	-0.643	383.09	1795.08	321.529	244.083	535.048	1365.6647	0.0043	2.79508	0.378
305	2008	10	-0.780	382.99	1814.18	321.796	244.080	534.927	1365.6759	0.0046	2.81418	0.440
306	2008	11	-0.621	384.13	1812.37	322.013	244.225	534.906	1365.7065	0.0048	2.81237	0.394
307	2008	12	-0.666	385.56	1812.88	322.182	244.204	535.005	1365.6926	0.0046	2.81288	0.330

308 rows × 12 columns

In [24]:
```
c_data2_train = c_data2.iloc[0:284]
c_data2_test = c_data2.iloc[284:308]
c_data2_test.head()
```

Out[24]:

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols	NO	Temp
284	2007	1	0.974	382.93	1799.66	320.561	248.372	539.206	1365.7173	0.0054	2.79966	0.601
285	2007	2	0.510	383.81	1803.08	320.571	248.264	538.973	1365.7145	0.0051	2.80308	0.498
286	2007	3	0.074	384.56	1803.10	320.548	247.997	538.811	1365.7544	0.0045	2.80310	0.435
287	2007	4	-0.049	386.40	1802.11	320.518	247.574	538.586	1365.7228	0.0045	2.80211	0.466
288	2007	5	0.183	386.58	1795.65	320.445	247.224	538.130	1365.6932	0.0041	2.79565	0.372

In [33]:
```
coeff = closed_form_1(c_data2_train.loc[:,"MEI":"NO"], c_data2_train.Temp)
coeff
```

Out[33]:
```
MEI           0.230112
CO2           0.006167
CH4          -0.006012
N2O          -0.016552
CFC-11       -0.006657
CFC-12        0.003825
TSI           0.093162
Aerosols     -1.537646
NO            6.125000
constant   -130.750000
Name: coeff, dtype: float64
```

In [27]:
```
r2_train_data2 = r2(coeff,c_data2_train.loc[:,"MEI":"NO"],c_data2_train.Temp)
r2_test_data2 = r2(coeff,c_data2_test.loc[:,"MEI":"NO"],c_data2_test.Temp)
print('The r2 of train data2 is',r2_train_data2)
print('The r2 of test data2 is',r2_test_data2)
```

```
('The r2 of train data2 is', 0.30559429925520726)
('The r2 of test data2 is', 0.09516910092806588)
```

In [31]:
```
t_value(coeff,c_data2_train.loc[:,"MEI":"NO"],c_data2_train.Temp)
```

Out[31]:
```
MEI         16.522157
CO2          1.253986
CH4         -0.000027
N2O         -0.897771
CFC-11      -1.902073
CFC-12       1.753051
TSI          2.933252
Aerosols    -3.349718
NO           0.000027
constant    -0.000578
Name: tvalue, dtype: float64
```

In [35]:
```
coeff2 = closed_form_1(c_data2_test.loc[:,"MEI":"NO"], c_data2_test.Temp)
coeff2
```

Out[35]:
```
MEI          -0.161763
CO2           0.001384
CH4          -1.809082
N2O          -0.115593
CFC-11        0.037829
CFC-12       -0.073145
TSI          -1.060037
```

Out[35]:
```
MEI           -0.161763
CO2            0.001384
CH4           -1.809082
N2O           -0.115593
CFC-11         0.037829
CFC-12        -0.073145
TSI           -1.060037
Aerosols     148.424188
NO          1807.250000
constant    -290.000000
Name: coeff, dtype: float64
```

In [36]:
```
t_value(coeff2,c_data2_test.loc[:,"MEI":"NO"],c_data2_test.Temp)
```

Out[36]:
```
MEI         -1.502220
CO2          0.032923
CH4               NaN
N2O         -0.308392
CFC-11       0.107474
CFC-12      -0.190509
TSI         -0.552328
Aerosols     0.960511
NO                NaN
constant          NaN
Name: tvalue, dtype: float64
```

The linear model with sum of squre error cannot solve overfitting problem.

Problem 2

- The loss function with L_1 regularization

$$(Y - X\beta)'(Y - X\beta) + \lambda'|\beta|$$

- The loss function with L_2 regularization

$$(Y - X\beta)'(Y - X\beta) + \lambda\beta'\beta$$

In [51]:
```
def closed_form_2(x_train,y_train,lamda):
    x_train["constant"]=1
    col = x_train.columns
    x_train = np.matrix(x_train)
    y_train = np.matrix(y_train).T
    coeff = np.dot((np.dot(x_train.T,x_train) + lamda*np.eye(x_train.shape[1])).I,np.dot(x_train.T,y_train))
    result = pd.Series(coeff.A1, index=col,
                       name="corcoe")
    return result

for lamda in [10,1,0.1,0.01,0.001]:
    coeff = closed_form_2(c_data2_train.loc[:,"MEI":"NO"], c_data2_train.Temp,lamda)
    print('when lamda is',lamda)
    print(coeff)
    print('\n')
```

```
('when lamda is', 10)
MEI         0.040543
CO2         0.008146
CH4         0.000205
N2O        -0.016081
CFC-11     -0.006361
CFC-12      0.003689
TSI         0.001265
Aerosols   -0.024433
NO         -0.000220
constant   -0.000220
Name: corcoe, dtype: float64
```

```
('when lamda is', 1)
MEI          0.043956
CO2          0.008043
CH4          0.000216
N2O         -0.016930
CFC-11      -0.006466
CFC-12       0.003769
TSI          0.001469
Aerosols    -0.211773
NO          -0.002294
constant    -0.002294
Name: corcoe, dtype: float64

('when lamda is', 0.1)
MEI          0.050687
CO2          0.006989
CH4          0.000156
N2O         -0.014816
CFC-11      -0.006079
CFC-12       0.003661
TSI          0.001380
Aerosols    -0.871360
NO          -0.025040
constant    -0.025040
Name: corcoe, dtype: float64

('when lamda is', 0.01)
MEI          0.054653
CO2          0.006351
CH4          0.000341
N2O         -0.013486
CFC-11      -0.005839
CFC-12       0.003591
TSI          0.001642
Aerosols    -1.265462
NO          -0.261767
constant    -0.261768
Name: corcoe, dtype: float64

('when lamda is', 0.001)
MEI          0.055576
CO2          0.006261
CH4          0.002613
N2O         -0.013402
CFC-11      -0.005832
CFC-12       0.003589
TSI          0.004989
Aerosols    -1.332782
NO          -2.539573
constant    -2.539576
Name: corcoe, dtype: float64
```

In [45]:

```
r2(coeff,c_data2_test.loc[:,"MEI":"NO"],c_data2_test.Temp)
```

Out[45]:

```
0.09516910092806588
```

2.3 L2 can decrease the coefficient of umimportant varibales and get the coeffs more closer. In this dataset, it can present more robust performance.

2.4 The answers can be seen above.

Problem 3

3.1 see the chart as enclosed

In [84]:
```
import numpy as np
import pandas as pd
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn import linear_model

def var_if(x,lamda =10):
    col = list(range(x.shape[1]))
    flag = 1
    while (flag):
        flag = 0
        varif = [variance_inflation_factor(x.iloc[:,col].values, ix) for ix in range(x.iloc[:,col].shape[1])]
        maxvarif = max(varif)
        maxix = varif.index(maxvarif)
        if maxvarif > lamda:
            del col[maxix]
            flag = 1
    print('Remain Variables:', list(X.columns[col]))
    return list(x.columns[col])

data3 = pd.read_csv("climate_change_1.csv")
X = data3.loc[:,"MEI":"Aerosols"]
y = data3.loc[:,"Temp"]

X_train3 = X[0:284]
X_test3 = X[284:308]
y_train3 = y[0:284]
y_test3 = y[284:308]
res = var_if(X_train3)
print(res)
```

```
('Remain Variables:', ['MEI', 'CFC-12', 'Aerosols'])
['MEI', 'CFC-12', 'Aerosols']
```

In [96]:
```
variables = ['MEI','CFC-12','Aerosols']
x_4 = data3.loc[:,variables]
y_4 = data3.Temp
x_train4 = x_4[0:284]
X_test = x_4[284:208]
y_train4 = y_4[0:284]
y_test4 = y_4[284:308]

coeff =closed_form_1(x_train4, y_train4)
R2_4 = r2(coeff,x_train4, y_train4,)
print(R2_4)
```

```
0.10518055026754325
```

Problem 4

In [16]:

```
from numpy.linalg import inv
from numpy import dot
import matplotlib.pyplot as plt

x_g = c_data1_train.loc[:,"MEI":"Aerosols"]
x_g.insert(0,"ones",1)
x_g = np.matrix(x_g.values)
y_g = c_data1_train.Temp
y_g = np.matrix(y_g.values)

for i in range(1,9):
    x_g[:,i] = (x_g[:,i] - min(x_g[:,i])) / (max(x_g[:,i]) - min(x_g[:,i]))

beita = dot(dot(inv(dot(x_g.T,x_g)),x_g.T),y_g.T)


def costfunc(x_g,y_g,beita):
    inner = np.power(((x_g*beita.T)-y_g),2)
    return np.sum(inner)/(2*len(x_g))

def gradientDescent(x, y, theta, alpha, iters):
    temp = np.matrix(np.zeros(theta.shape))
    parameters = int(theta.ravel().shape[1])
    cost = np.zeros(iters)
    for i in range(iters):
        error = (x * theta.T) - y
        for j in range(parameters):
            term = np.multiply(error, x[:,j])
            temp[0,j] = theta[0,j] - ((alpha / len(x)) * np.sum(term))

        theta = temp
        cost[i] = costfunc(x, y, theta)
    return theta,cost
```

In [*]:

```
alpha = 0.001
iters = 100000
theta = np.matrix(np.zeros(9))

gradient,cost = gradientDescent(x_g
                                , y_g, theta, alpha, iters)

fig, bx = plt.subplots(figsize=(6,6))
bx.plot(np.arange(iters), cost, 'r')
bx.set_xlabel('Iterations')
bx.set_ylabel('Cost')
bx.set_title('Cost with gradientDescent')
plt.show()
```