In [1]:

```
import numpy as np
import scipy.stats as sps
import pandas as pd
import matplotlib.pyplot as plt
%pylab inline
```

Populating the interactive namespace from numpy and matplotlib

In [2]:

```
frame = pd.read_csv('1.csv', header=None, sep=',')
```

In [3]:

```
frame
```

Out[3]:

	0	1	2	3	4
0	-3.559408	1	6.490758	1.587476	0.009951
1	-9.077184	1	7.850516	0.827435	0.009930
2	30.869241	1	2.284262	8.049759	0.010058
3	0.106257	1	3.669403	1.108870	0.009974
4	22.341931	1	9.837528	9.758840	0.009999
5	6.015793	1	2.502938	1.976068	0.009995
6	-11.980174	1	7.676827	0.045813	0.009924
7	23.218360	1	2.289537	6.153489	0.010039
8	0.100527	1	9.173865	3.799071	0.009946
9	27.269879	1	6.980626	9.509448	0.010025
10	31.384797	1	5.253003	9.726376	0.010045
11	-6.598528	1	8.293633	1.714535	0.009934
12	14.765551	1	0.629582	3.208768	0.010026
13	31.617425	1	4.029495	9.137723	0.010051
14	10.312853	1	3.179003	3.411363	0.010002
15	27.938261	1	7.372958	9.891985	0.010025
16	18.003122	1	9.684659	8.575995	0.009989
17	21.561170	1	3.254789	6.251579	0.010030
18	14.236649	1	8.096850	6.833287	0.009988
19	10.063359	1	1.041901	2.236739	0.010012
20	22.421952	1	7.616674	8.635158	0.010010
21	23.523121	1	6.807816	8.531124	0.010017
22	36.827251	1	0.366934	8.599674	0.010082
23	17.422619	1	0.118845	3.601322	0.010035
24	15.538030	1	7.978095	7.088774	0.009991
25	6.986903	1	7.141642	4.536752	0.009974
26	4.582070	1	1.605496	1.163719	0.009996
27	3.495847	1	0.041959	0.156801	0.010001
28	32.652026	1	0.161229	7.503491	0.010073

29	24.699623	1	5.858630	8.330007	0.010025
...	...	...	...	...	...
970	23.999781	1	0.122286	5.300533	0.010052
971	4.447489	1	1.540333	1.073524	0.009995
972	13.782428	1	6.647461	5.967926	0.009993
973	23.571652	1	5.561695	7.905869	0.010023
974	3.548737	1	8.698049	4.450115	0.009957
975	24.815181	1	0.922556	5.898733	0.010050
976	30.161103	1	3.874863	8.718890	0.010049
977	34.798001	1	1.196903	8.567049	0.010074
978	15.742766	1	3.912702	5.116913	0.010012
979	30.231119	1	0.288979	6.932487	0.010067
980	1.639139	1	6.218564	2.752985	0.009965
981	15.780397	1	7.330278	6.859377	0.009995
982	13.730348	1	2.205524	3.749475	0.010015
983	-11.054633	1	8.624019	0.773332	0.009921
984	7.231460	1	7.750226	4.917447	0.009972
985	12.176965	1	4.506823	4.525971	0.010000
986	-3.436271	1	9.170954	2.960589	0.009938
987	23.403990	1	1.937634	6.053728	0.010041
988	10.649717	1	6.575947	5.191179	0.009986
989	24.984630	1	1.339273	6.135282	0.010048
990	23.375080	1	5.860149	7.990056	0.010021
991	-10.110461	1	8.168728	0.769192	0.009926
992	15.802764	1	0.888886	3.628765	0.010027
993	1.940771	1	7.244078	3.298679	0.009960
994	-5.416513	1	6.918600	1.320754	0.009944
995	8.130829	1	1.365504	1.911510	0.010005
996	37.530816	1	1.073912	9.162200	0.010081
997	13.617988	1	3.291409	4.244624	0.010010
998	2.845115	1	4.807232	2.323936	0.009975
999	19.916243	1	3.628238	6.031448	0.010024

1000 rows × 5 columns

In [4]:

```
data = pd.DataFrame.as_matrix(frame)
```

Найдем оценку параметра θ методом наименьших квадратов:

In [5]:

```
z = data[:, 1:]
y = data[:, 0]
theta_est = np.linalg.inv(z.T.dot(z)).dot(z.T).dot(y)
print theta_est
```

```
[  3.98035920e+01  -2.03507682e+00   4.03605975e+00  -3.67138513e+03]
```

$det(Z^T Z)$:

In [6]:

```
np.linalg.det(z.T.dot(z))
```

Out[6]:

```
0.66090295317101178
```

Заметим, что матрица $Z^T Z$ невырождена.

Рассмотрим оценку $\widehat{\theta} = (Z^T Z + \lambda I)^{-1} Z^T Y$

In [7]:

```
lambda_ = np.arange(0, 100, 0.1)
theta = np.empty((lambda_.size, data[:, 0].size, 4))
z1 = np.empty((data[:, 0].size - 1, 4))
for i in range(data[:, 0].size):
    for j in range(lambda_.size):
        z1 = np.delete(z, i, 0)
        y1 = np.delete(y, i, 0)
        theta[j][i] = np.linalg.inv(z1.T.dot(z1) + lambda_[j] *
                                    np.eye(4)).dot(z1.T).dot(y1)
```

In [12]:

```
theta
```

Out[12]:

```
array([[[ 4.08992564e+01,  -2.03616731e+00,   4.03714457e+00,
         -3.78094154e+03],
        [ 1.92687409e+01,  -2.01461753e+00,   4.01562763e+00,
         -1.61793370e+03],
        [ 1.99834800e+01,  -2.01521078e+00,   4.01618278e+00,
         -1.68938131e+03],
        ...,
        [ 6.51486104e+01,  -2.06039260e+00,   4.06141649e+00,
         -6.20591850e+03],
        [ 5.07857676e+01,  -2.04605639e+00,   4.04706364e+00,
         -4.76961915e+03],
        [ 4.38456993e+01,  -2.03912563e+00,   4.04010798e+00,
         -4.07559047e+03]],

       [[ 3.08750268e+00,  -1.99815114e+00,   3.99946949e+00,
           3.09346194e-02],
        [ 3.08709240e+00,  -1.99822906e+00,   3.99958325e+00,
           3.09307254e-02],
        [ 3.08733340e+00,  -1.99810752e+00,   3.99942347e+00,
           3.09331262e-02],
        ...,
        [ 3.08713348e+00,  -1.99812937e+00,   3.99949061e+00,
           3.09306961e-02],
        [ 3.08726815e+00,  -1.99815441e+00,   3.99950122e+00,
           3.09321798e-02],
        [ 3.08749870e+00,  -1.99816307e+00,   3.99948601e+00,
           3.09345522e-02]],

       [[ 3.08548460e+00,  -1.99794070e+00,   3.99960501e+00,
           3.09146185e-02],
        [ 3.08507508e+00,  -1.99801865e+00,   3.99971868e+00,
           3.09106287e-02],
        [ 3.08531599e+00,  -1.99789687e+00,   3.99955863e+00,
           3.09130320e-02],
        ...,
        [ 3.08511474e+00,  -1.99791872e+00,   3.99962599e+00,
           3.09108035e-02],
        [ 3.08524931e+00,  -1.99794389e+00,   3.99963676e+00,
           3.09122180e-02],
        [ 3.08548102e+00,  -1.99795248e+00,   3.99962129e+00,
           3.09145693e-02]],


       ...,
       [[ 1.90378485e+00,  -1.86145294e+00,   4.06376270e+00,
           1.90970864e-02],
        [ 1.90372037e+00,  -1.86151858e+00,   4.06382279e+00,
           1.90964429e-02],
        [ 1.90375828e+00,  -1.86126456e+00,   4.06350856e+00,
           1.90968168e-02],
        ...,
        [ 1.90314674e+00,  -1.86134234e+00,   4.06370718e+00,
```

```
   1.90907024e-02],
 [  1.90329634e+00,  -1.86144235e+00,   4.06380481e+00,
    1.90922008e-02],
 [  1.90381397e+00,  -1.86138088e+00,   4.06366511e+00,
    1.90973760e-02]],

[[  1.90308567e+00,  -1.86135903e+00,   4.06378557e+00,
    1.90900938e-02],
 [  1.90302133e+00,  -1.86142463e+00,   4.06384561e+00,
    1.90894519e-02],
 [  1.90305903e+00,  -1.86117055e+00,   4.06353129e+00,
    1.90898236e-02],
 ...,
 [  1.90244747e+00,  -1.86124841e+00,   4.06373001e+00,
    1.90837090e-02],
 [  1.90259712e+00,  -1.86134845e+00,   4.06382769e+00,
    1.90852079e-02],
 [  1.90311469e+00,  -1.86128693e+00,   4.06368793e+00,
    1.90903824e-02]],

[[  1.90238706e+00,  -1.86126517e+00,   4.06380839e+00,
    1.90831070e-02],
 [  1.90232287e+00,  -1.86133074e+00,   4.06386839e+00,
    1.90824665e-02],
 [  1.90236036e+00,  -1.86107659e+00,   4.06355398e+00,
    1.90828362e-02],
 ...,
 [  1.90174877e+00,  -1.86115452e+00,   4.06375280e+00,
    1.90767213e-02],
 [  1.90189848e+00,  -1.86125461e+00,   4.06385052e+00,
    1.90782208e-02],
 [  1.90241598e+00,  -1.86119303e+00,   4.06371070e+00,
    1.90833947e-02]]])
```

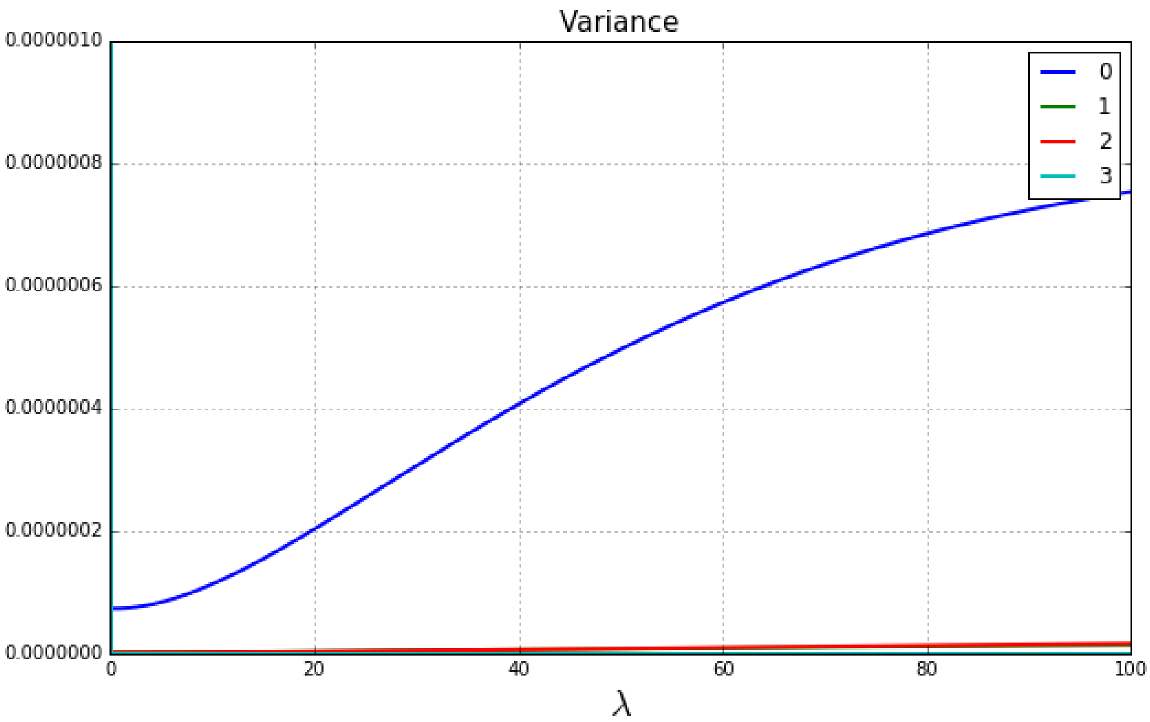
Для каждого λ найдем ошибку дисперсии:

In [8]:

```
var = np.empty((lambda_.size, 4))
for i in range(lambda_.size):
    var[i] = (theta[i]**2).mean(0) - (theta[i].mean(0))**2
```

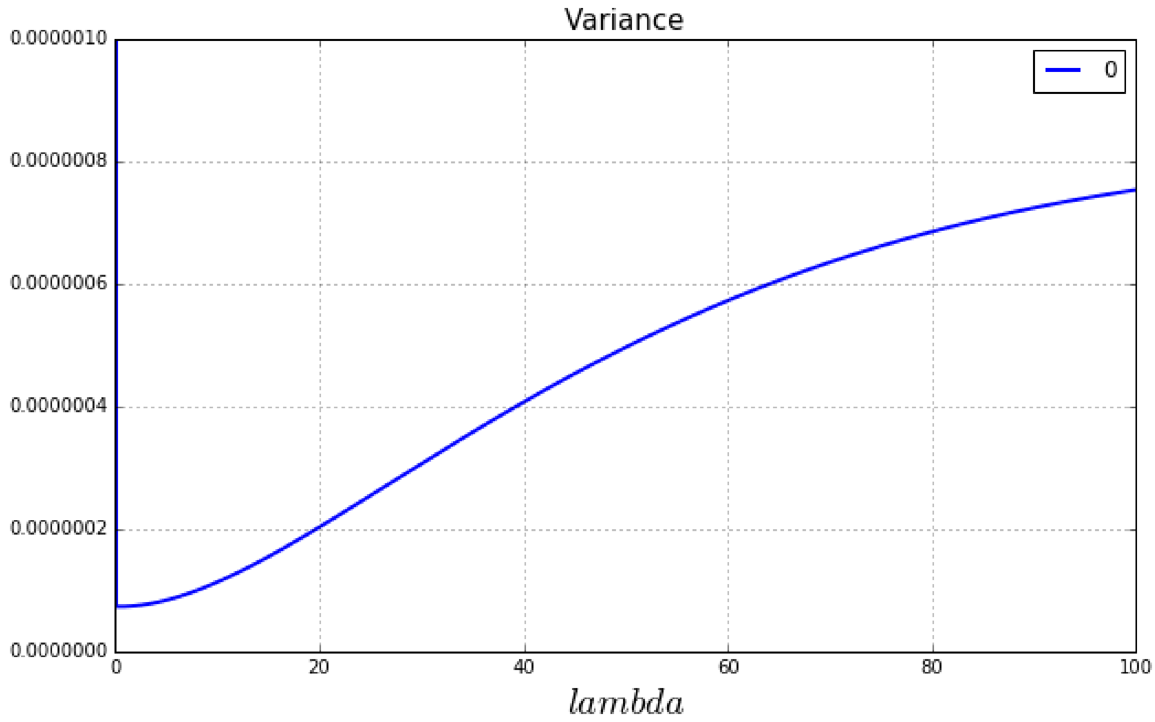
In [19]:

```
plt.figure(figsize=(10, 6))
for i in range(4):
    plt.plot(lambda_, var[:, i], linewidth = 2, label=str(i))
plt.legend()
plt.xlim(0, 100)
plt.ylim(0, 0.000001)
plt.xlabel(r'$\lambda$', fontsize = 20)
plt.title('Variance', fontsize = 15)
plt.grid()
plt.show()
```



In [26]:

```
plt.figure(figsize=(10, 6))
plt.plot(lambda_, var[:, 0], linewidth = 2, label=str(0))
plt.legend()
plt.xlim(0, 100)
plt.ylim(0, 0.000001)
plt.xlabel(r'$lambda$', fontsize = 20)
plt.title('Variance', fontsize = 15)
plt.grid()
plt.show()
```



In [27]:

```
var
```

Out[27]:

```
array([[  1.01110748e+02,   1.01099630e-04,   1.01077960e-04,
          1.01110487e+06],
       [  7.30693195e-08,   1.27052946e-09,   1.23581856e-09,
          7.27716769e-12],
       [  7.30353005e-08,   1.27086519e-09,   1.23531407e-09,
          7.28622750e-12],
       ...,
       [  7.52744441e-07,   1.38503080e-08,   1.61028701e-08,
          7.52752891e-11],
       [  7.52996018e-07,   1.38615377e-08,   1.61183138e-08,
          7.53004465e-11],
       [  7.53246769e-07,   1.38727625e-08,   1.61336366e-08,
          7.53255239e-11]])
```

Из графиков видно, что при λ стремящемся к 0, дисперсия уменьшается для каждой координаты, но при этом при $\lambda = 0$ дисперсия велика. Оптимальное λ - близкое к 0, например, $\lambda = 1$.