In [1]:
```
import numpy as np
import scipy.stats as sps
import matplotlib.pyplot as plt
import pylab
%matplotlib inline
```

Оценка максимального правдоподобия для параметра θ распределения $N(0, \theta)$:

$\theta_1 = S_n^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^2$

В качестве априорного распределения возьмем сопряженное распределение — Inverse-gamma Distribution $\Gamma_{inv}(\alpha_0, \beta_0)$, mean $= \frac{\beta_0}{\alpha_0 - 1}$; $\alpha = \alpha_0 + \frac{n}{2}, \beta = \beta_0 + \frac{\sum_{i=1}^{n} X_i^2}{2}$.

Следовательно, байесовская оценка: $\theta^* = \frac{\beta}{\alpha - 1} = \frac{2\beta_0 + \sum_{i=1}^{n} X_i^2}{2\alpha_0 + n - 2}$."

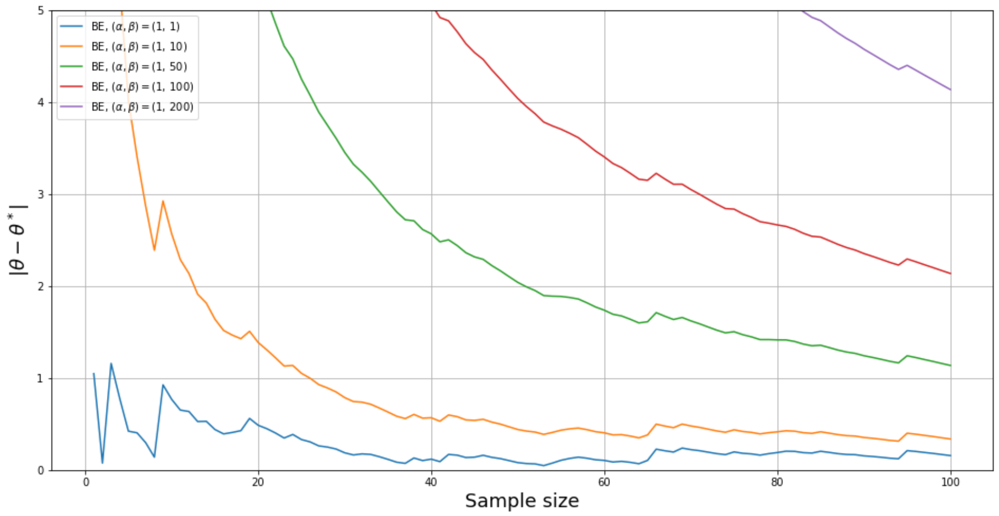
In [2]:
```
# генерация выборки
N = 100
sample = sps.norm.rvs(size = N, loc = 0, scale = 1)

# Выборочные дисперсии
dispersions = np.zeros(N)
for i in range(N):
    dispersions[i] = pylab.var(sample[:(i+1)])

# Байесовская оценка для theta
def BayesEstimation(sample, alpha, beta):
    return ((2. * beta) + sum([i**2 for i in sample]))/((2.
* alpha) + len(sample) - 2.)
```

In [3]:
```
def estimate(params, ylim, ok):
    # Строим графики байесовских оценок для четырех парамет
ров
    plt.figure(figsize = (16, 8))
    for par, col in zip(params, pylab.arange(len(params))):
        estimations = np.array([BayesEstimation(sample[:(i+
1)], par[0], par[1]) for i in range(N)])
        plt.plot(np.arange(1, N+1), abs(estimations-1),
                 label='BE, $(\\alpha, \\beta)=$({}, {})'.f
ormat(par[0], par[1]))

    if ok:
        # Строим график для ОМП
        plt.plot(np.arange(1,N+1), abs(dispersions-1), labe
l='MLE')

    plt.ylim((0, ylim))
    plt.legend()
    plt.xlabel('Sample size', fontsize=18)
    plt.ylabel("$|\\theta - \\theta^*|$", fontsize=18)
    plt.grid()
    plt.show()
```

In [4]:
```
# массивы параметров априорного распределения
params_1 = np.array([(1,1), (1,10), (1,50), (1,100), (1,200
)])
params_2 = np.array([(1,1), (10,1), (50,1), (100,1), (200,1
)])
```

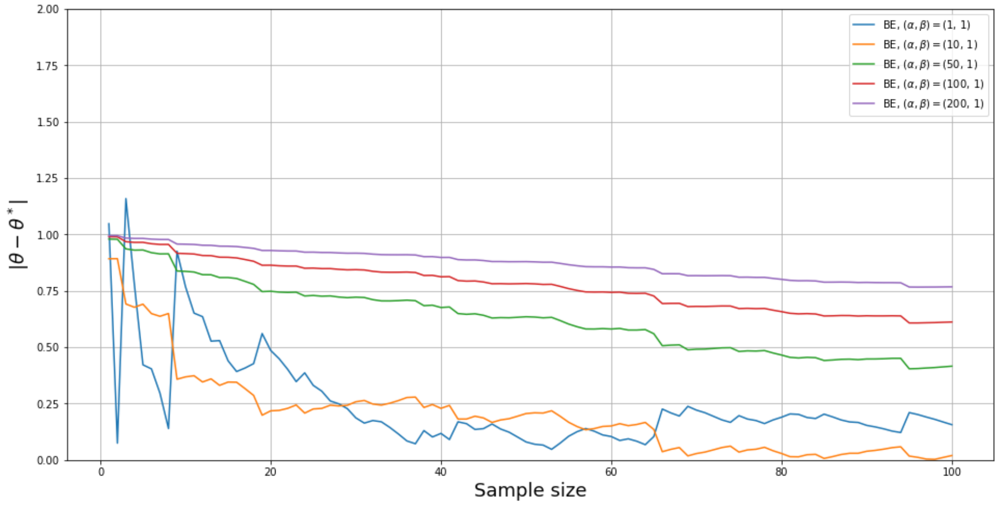
Построим графики при переменном параметре масштаба априорного распределения

In [5]:
```
estimate(params_1, ylim=5, ok=False)
```



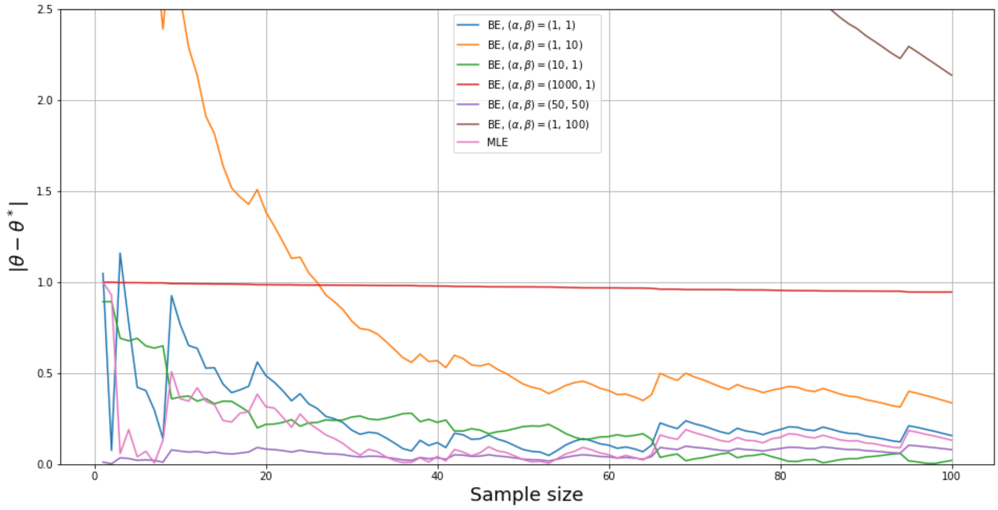
Построим графики при переменном параметре сдвига априорного распределения

In [6]:
```
estimate(params_2, ylim=2, ok=False)
```



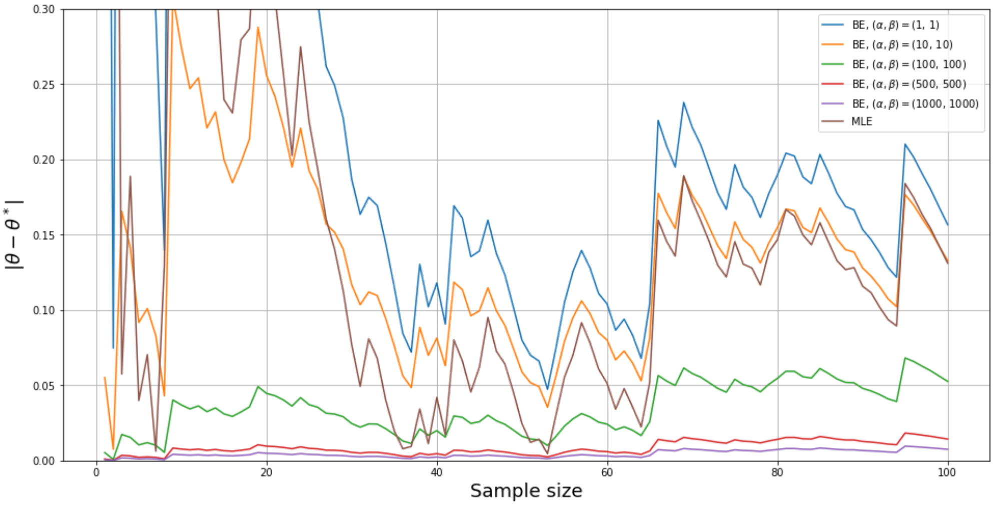
Видно, что при увеличении параметров оценка ухудшается. Построим теперь на одном графике кривые оценки ОМП и байесовских оценок для разных параметров априорного распределения.

In [7]:
```
params_3 = np.array([(1,1), (1,10), (10,1), (1000,1), (50,5
0), (1, 100)])
estimate(params_3, ylim=2.5, ok=True)
```



По этому графику видно, что при большом значении параметра масштаба и маленьком параметре сдвига оценка получается плохой. То же самое можно сказать и про оценку с большим параметром сдвига и маленьким параметром масштаба.
Можно было подумать, что при выборе небольших значениях параметра оценка наилучшая, но это опровергает оценка при значениях (50, 50).

In [8]:
```
params_4 = np.array([(1,1), (10,10), (100,100), (500,500),
(1000,1000)])
estimate(params_4, ylim=0.3, ok=True)
```



Вывод :

1) При значениях параметров (1,1) и (10, 10) байесовская оценка похожа на ОМП и хорошо приближает значение параметра θ.
2) При большой разности параметров байесовская оценка очень плоха.
3) Наилучших оценок байесовским методом можно добиться, используя большие и равные между собой значения параметров априорного распределения.