In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as sts
%matplotlib inline
```

Выборка размера $N = 10000$ из $R[0, \theta]$, где $\theta = 100$

In [37]:
```
N = 10000
a = 0
b = 100
uniform_rv = sts.uniform(a, b - a)
sample = uniform_rv.rvs(N)
```

В массивах хранятся элементы, соответсвующие определенным оценкам, для каждого $n \leq N$

$first \sim 2\bar{X}$

$second \sim \bar{X} + \frac{X_{(n)}}{2}$

$third \sim (n+1)X_{(1)}$

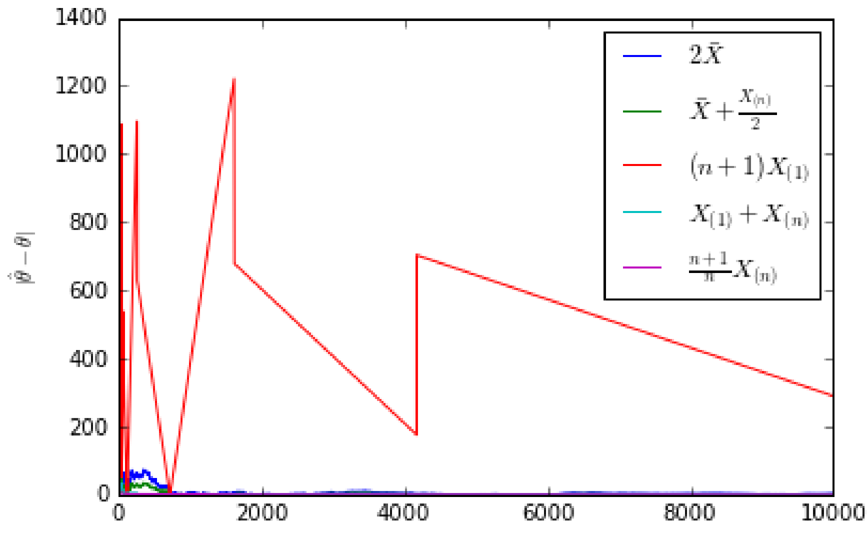
$fourth \sim X_{(1)} + X_{(n)}$

$fifth \sim \frac{n+1}{n}X_{(n)}$

In [38]:
```
avrg = float(sample[0])
min_el = avrg
max_el = avrg

first = np.array([2*avrg])
second = np.array([avrg + max_el/2])
third = np.array([2*min_el])
fourth = np.array([min_el + max_el])
fifth = np.array([2*max_el])
for x in xrange(1,10000):
    avrg = (avrg*x + sample[x])/(x+1)
    if(sample[x] > max_el):
        max_el = sample[x]
    if(sample[x] < min_el):
        min_el = sample[x]

    first = np.append(first, 2*avrg)
    second = np.append(second, avrg + max_el/2)
    third = np.append(third, (x + 2)*min_el)
    fourth = np.append(fourth, min_el + max_el)
    fifth = np.append(fifth, (x + 2)/(x + 1) * max_el)
```

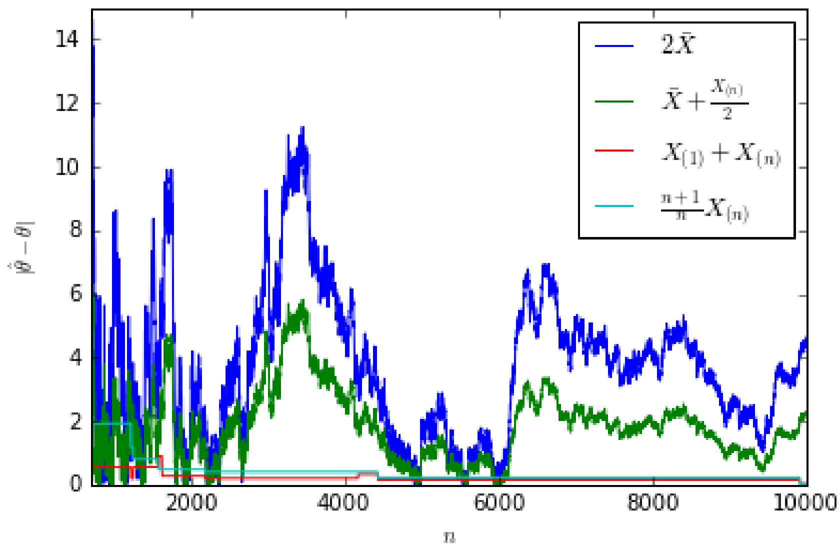
Построение графиков модуля разности оценки и θ|

In [98]:
```
x = np.linspace(0, N, N)
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(third - b), label = '$(n+1)X_{(1)}{}$')
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.legend()
plt.show()
```



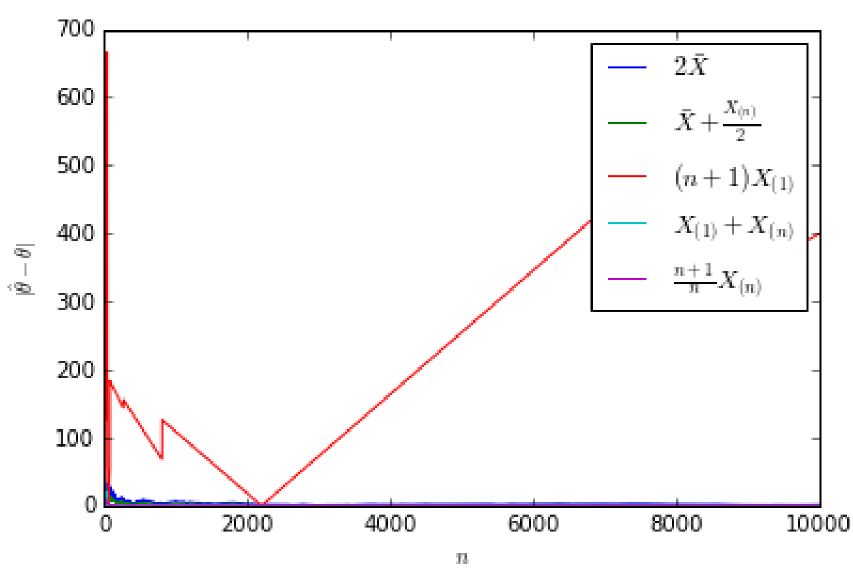
Для наглядности без третьей оценки

In [99]:
```
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.legend()
plt.xlim(700, N)
plt.ylim(0, 15)
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.show()
```

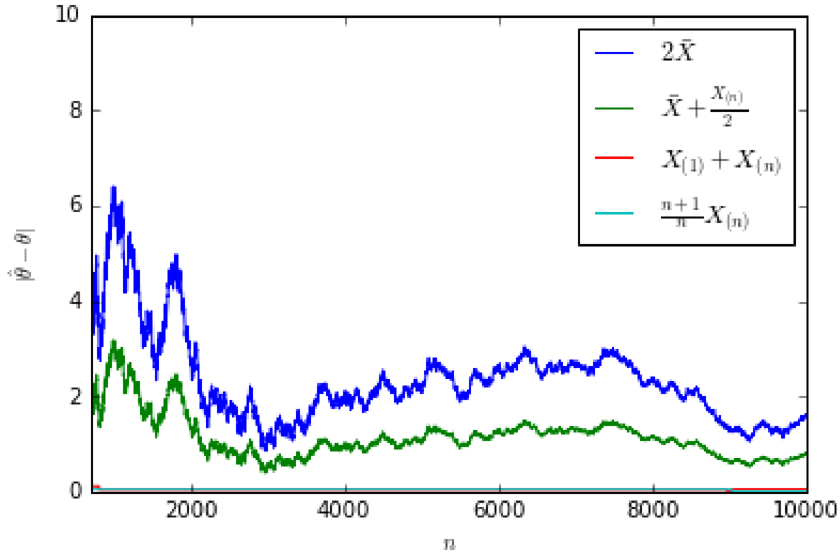


$$\theta = 200$$

In [100]:
```
b = 200
uniform_rv = sts.uniform(a, b - a)
sample = uniform_rv.rvs(N)

avrg = float(sample[0])
min_el = avrg
max_el = avrg

first = np.array([2*avrg])
second = np.array([avrg + max_el/2])
third = np.array([2*min_el])
fourth = np.array([min_el + max_el])
fifth = np.array([2*max_el])
for x in xrange(1,10000):
    avrg = (avrg*x + sample[x])/(x+1)
    if(sample[x] > max_el):
        max_el = sample[x]
    if(sample[x] < min_el):
        min_el = sample[x]

    first = np.append(first, 2*avrg)
    second = np.append(second, avrg + max_el/2)
    third = np.append(third, (x + 2)*min_el)
    fourth = np.append(fourth, min_el + max_el)
    fifth = np.append(fifth, (x + 2)/(x + 1) * max_el)

x = np.linspace(0, N, N)
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(third - b), label = '$(n+1)X_{(1)}{}$')
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.legend()
plt.show()
```

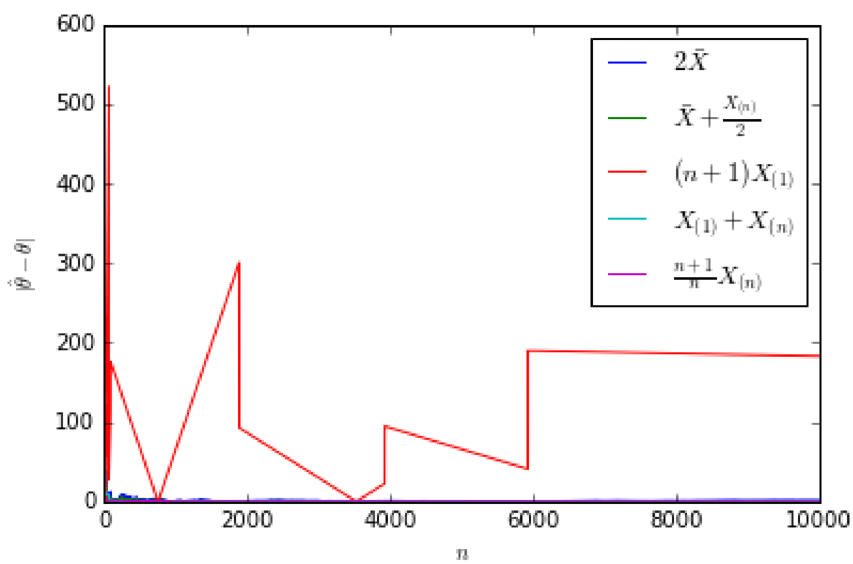


In [101]:
```
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.legend()
plt.xlim(700, N)
plt.ylim(0, 10)
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.show()
```

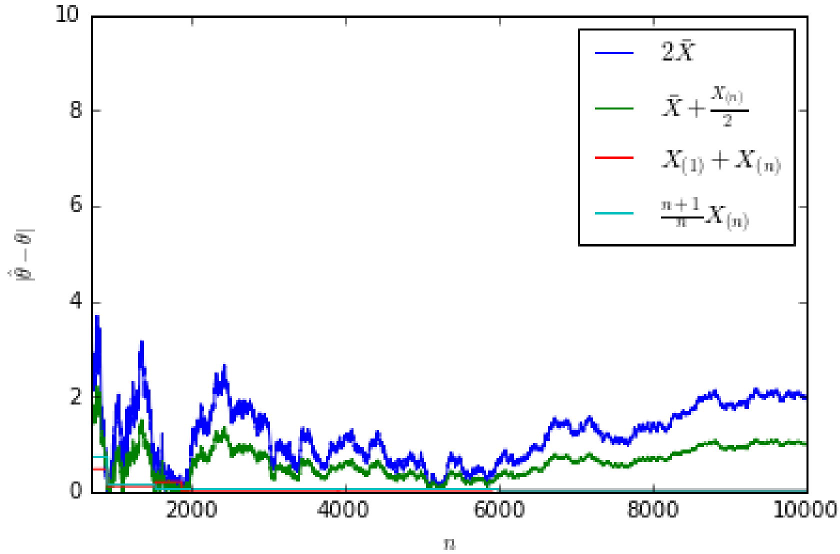


$\theta = 50$

In [102]:
```
b = 200
uniform_rv = sts.uniform(a, b - a)
sample = uniform_rv.rvs(N)

avrg = float(sample[0])
min_el = avrg
max_el = avrg

first = np.array([2*avrg])
second = np.array([avrg + max_el/2])
third = np.array([2*min_el])
fourth = np.array([min_el + max_el])
fifth = np.array([2*max_el])
for x in xrange(1,10000):
    avrg = (avrg*x + sample[x])/(x+1)
    if(sample[x] > max_el):
        max_el = sample[x]
    if(sample[x] < min_el):
        min_el = sample[x]

    first = np.append(first, 2*avrg)
    second = np.append(second, avrg + max_el/2)
    third = np.append(third, (x + 2)*min_el)
    fourth = np.append(fourth, min_el + max_el)
    fifth = np.append(fifth, (x + 2)/(x + 1) * max_el)

x = np.linspace(0, N, N)
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(third - b), label = '$(n+1)X_{(1)}{}$')
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.legend()
plt.show()
```

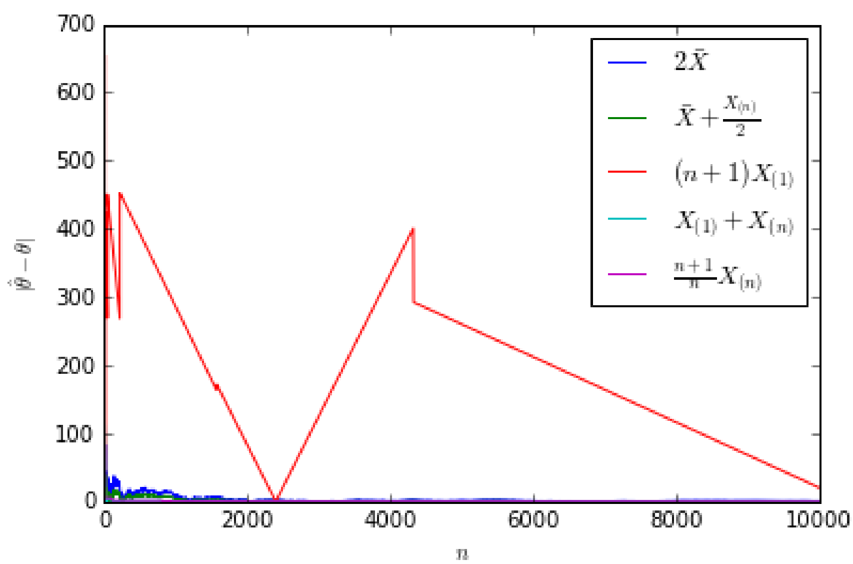


In [103]:
```
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.legend()
plt.xlim(700, N)
plt.ylim(0, 10)
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.show()
```

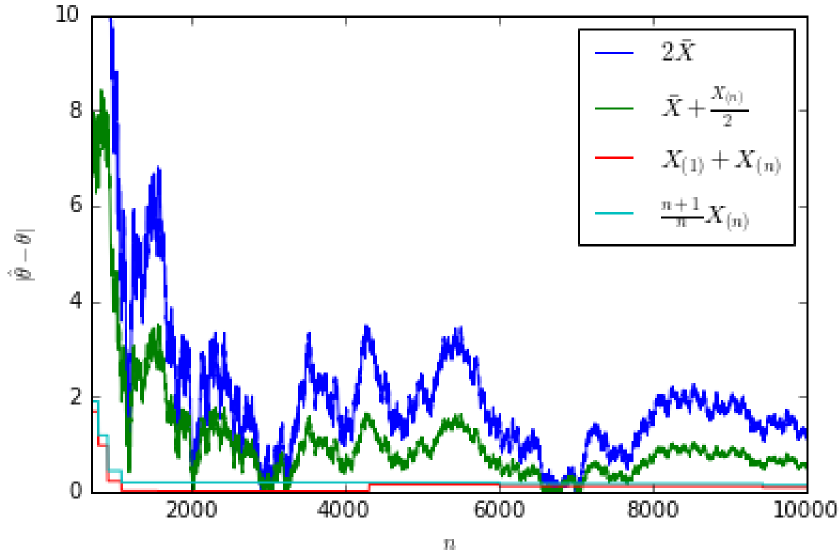


$$\theta = 500$$

In [104]:
```
b = 500
uniform_rv = sts.uniform(a, b - a)
sample = uniform_rv.rvs(N)

avrg = float(sample[0])
min_el = avrg
max_el = avrg

first = np.array([2*avrg])
second = np.array([avrg + max_el/2])
third = np.array([2*min_el])
fourth = np.array([min_el + max_el])
fifth = np.array([2*max_el])
for x in xrange(1,10000):
    avrg = (avrg*x + sample[x])/(x+1)
    if(sample[x] > max_el):
        max_el = sample[x]
    if(sample[x] < min_el):
        min_el = sample[x]

    first = np.append(first, 2*avrg)
    second = np.append(second, avrg + max_el/2)
    third = np.append(third, (x + 2)*min_el)
    fourth = np.append(fourth, min_el + max_el)
    fifth = np.append(fifth, (x + 2)/(x + 1) * max_el)

x = np.linspace(0, N, N)
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(third - b), label = '$(n+1)X_{(1)}{}$')
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.legend()
plt.show()
```

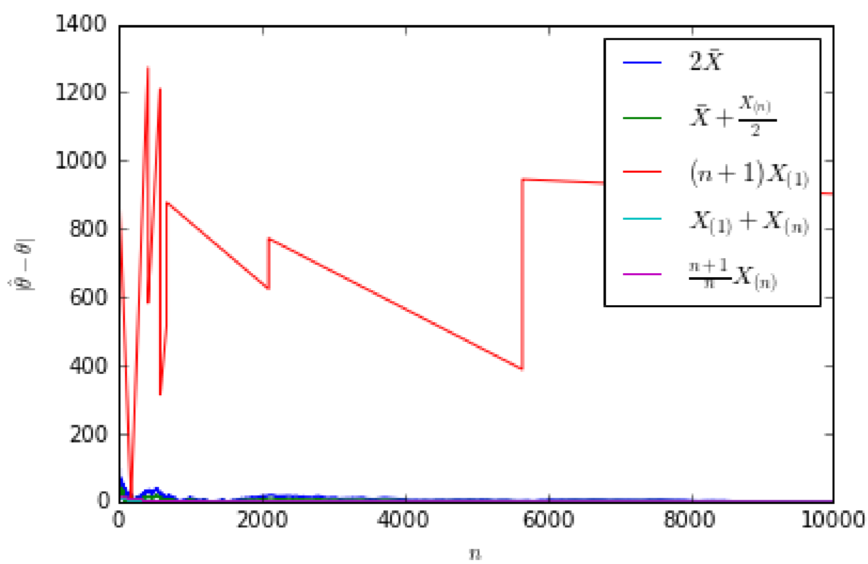


In [105]:
```
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.legend()
plt.xlim(700, N)
plt.ylim(0, 10)
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.show()
```

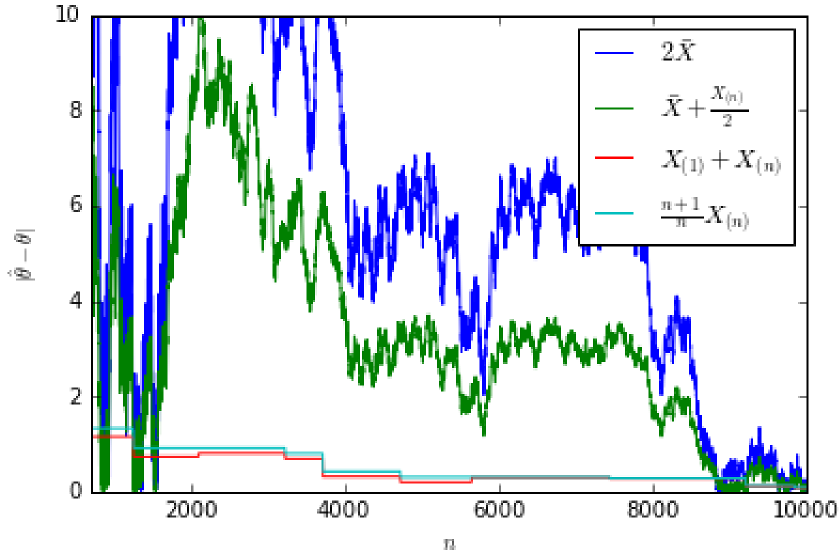


$\theta = 1000$

In [106]:
```
b = 1000
uniform_rv = sts.uniform(a, b - a)
sample = uniform_rv.rvs(N)

avrg = float(sample[0])
min_el = avrg
max_el = avrg

first = np.array([2*avrg])
second = np.array([avrg + max_el/2])
third = np.array([2*min_el])
fourth = np.array([min_el + max_el])
fifth = np.array([2*max_el])
for x in xrange(1,10000):
    avrg = (avrg*x + sample[x])/(x+1)
    if(sample[x] > max_el):
        max_el = sample[x]
    if(sample[x] < min_el):
        min_el = sample[x]

    first = np.append(first, 2*avrg)
    second = np.append(second, avrg + max_el/2)
    third = np.append(third, (x + 2)*min_el)
    fourth = np.append(fourth, min_el + max_el)
    fifth = np.append(fifth, (x + 2)/(x + 1) * max_el)

x = np.linspace(0, N, N)
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(third - b), label = '$(n+1)X_{(1)}{}$')
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.legend()
plt.show()
```



In [107]:
```
plt.plot(x, np.abs(first - b), label = '$2\\bar{X}$')
plt.plot(x, np.abs(second - b), label = '$\\bar{X}+\\frac{X_{(n)}{}}{2}
plt.plot(x, np.abs(fourth - b), label = '$X_{(1)}{} + X_{(n)}{}$')
plt.plot(x, np.abs(fifth - b), label = '$\\frac{n+1}{n}X_{(n)}{}$')
plt.legend()
plt.xlim(700, N)
plt.ylim(0, 10)
plt.ylabel('$|\hat{\\theta} - \\theta|$')
plt.xlabel('$n$')
plt.show()
```



Вывод

Лучше всего приближает значение параметра θ оценка $X_{(1)} + X_{(n)}$. Также достаточно неплохо приближает оценка $\frac{n+1}{n}X_{(n)}$. Хуже всех приближает параметр θ оценка $(n+1)X_{(1)}$.