Deeper Analysis of MC and the antithetic sampling

Roylan Martinez

December 5, 2022

0.0.1 Data needed in first part

[128]:

```
import numpy as np
import matplotlib.pyplot as plt
```

[129]:

```
# Definitions
T = 6

# initial value
x, y = 0.5, 2

# error
epsilon = 0.2

# random_samples RANDOM DATA
random_samples = 100

# function to round
redondear = lambda numero: int(np.floor(numero + 0.5))
```

[130]:

```
# simulations x
uniformx = np.random.uniform(x - epsilon, x + epsilon, random_samples)

# simulations y
uniformy = np.random.uniform(y - epsilon, y + epsilon, random_samples)

# distribution for any (x, y)
xy = np.dstack((uniformx, uniformy))[0]
```

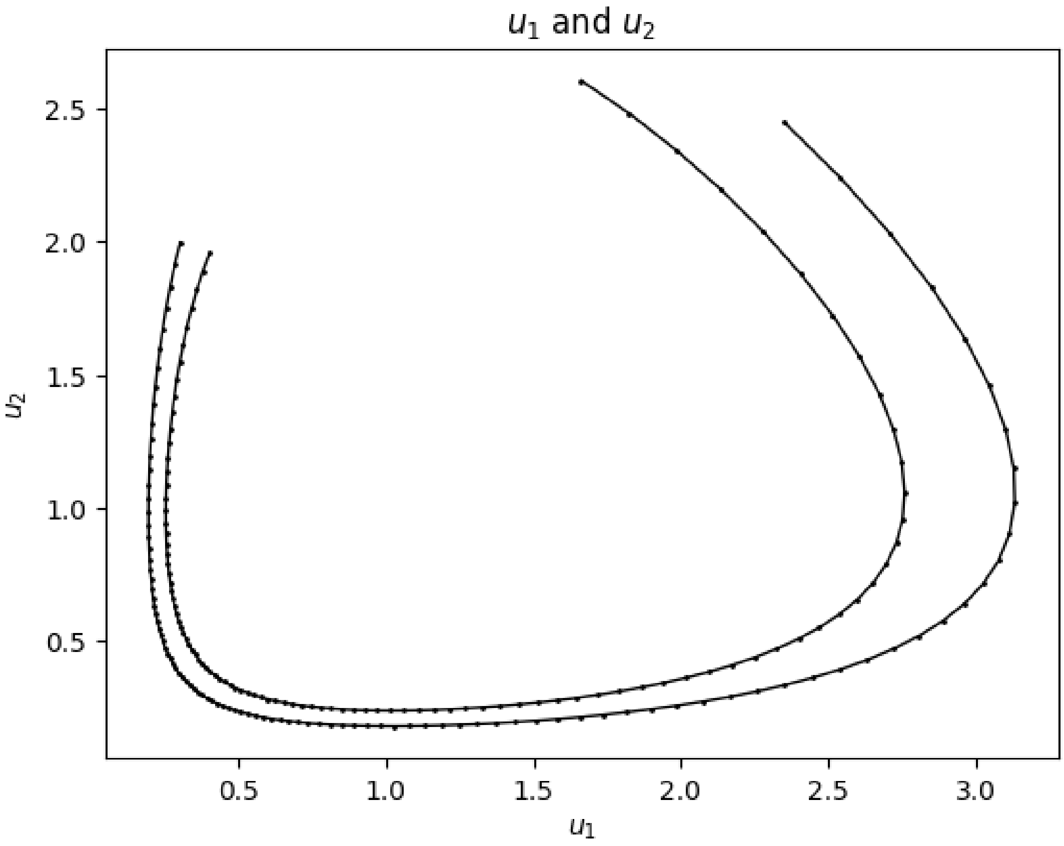
[131]:

```
# function
def function(variable):
    first_row = variable[0] - variable[0] * variable[1]
    second_row = variable[1] * variable[0] - variable[1]
    return np.array([first_row, second_row])


# Function to calculate euler
def euler(M, u0):
    # M
    samples = M

    # dt definition
    dt = T / (10 * np.sqrt(samples))

    # N
    N = redondear(T/dt)

    # results
    results = [u0]

    for i in range(N):

        # Add new results to results
        results.append(results[i] + dt * function(results[i]))

    results = np.array(results)
    return results
```

0.1 First plot (a)

[132]:
```
first, second = euler(random_samples, xy[0]).T
first1, second1 = euler(random_samples, xy[1]).T


# plt.scatter(first, second)
fig, ax = plt.subplots()
ax.plot(first, second, '-o', linewidth=1, markersize=1, color='black')
ax.plot(first1, second1, '-o', linewidth=1, markersize=1, color='black')

ax.set_title(r'$u_1$ and $u_2$')
ax.set_ylabel(r'$u_2$')
ax.set_xlabel(r'$u_1$')
```

[132]:
```
Text(0.5, 0, '$u_1$')
```



0.1.1 Data needed in second part

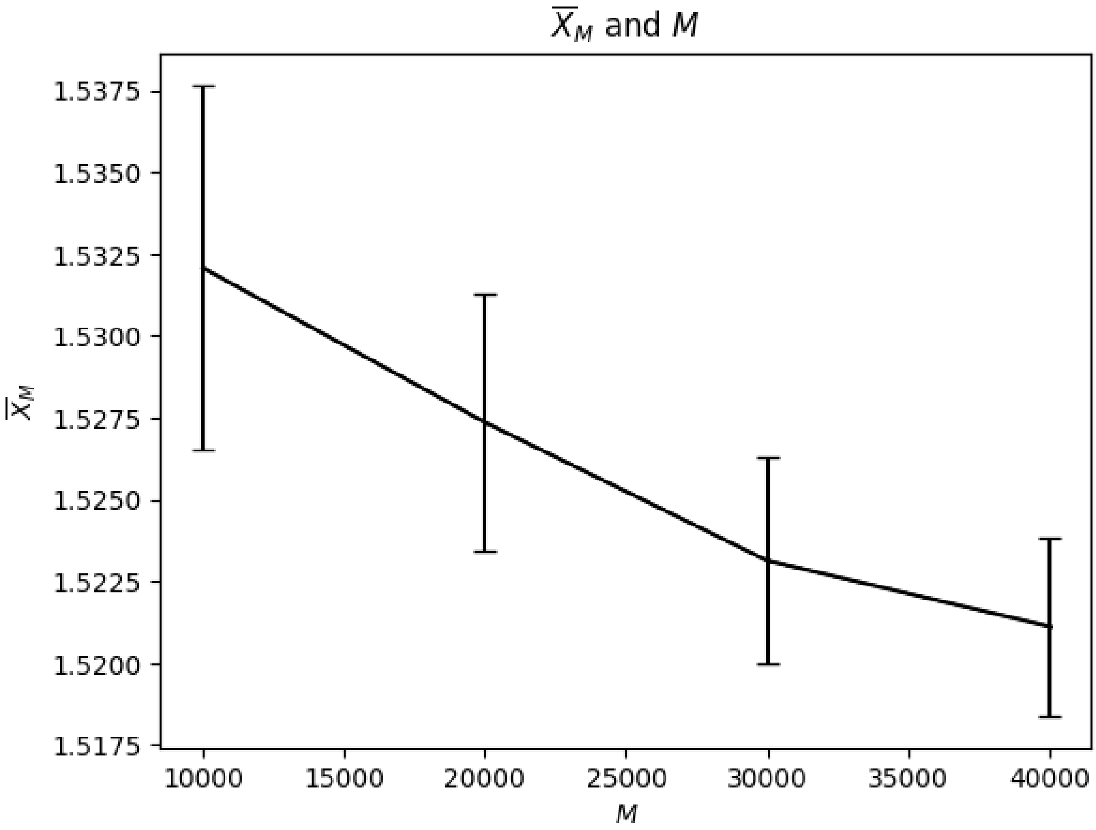
[133]:
```
valoresM = np.arange(1, 5) * 10000

xhat = lambda c: np.mean([euler(c, valor).T[0][-1] for valor in xy])
xhat_sd = lambda c: np.std([euler(c, valor).T[0][-1] for valor in xy], ddof=1)




ultima_lista = np.array([[k, xhat(k)] for k in valoresM])
ultima_lista_sd = np.array([(2/np.sqrt(k)) * xhat_sd(k) for k in valoresM])
```

1 Second plot (b)

[134]:
```
first3, second3 = ultima_lista.T

# plt.scatter(first, second)
fig, ax = plt.subplots()
```

```
ax.plot(first3, second3, '-o', linewidth=1, markersize=1, color='black')
ax.errorbar(first3, second3, ultima_lista_sd, capsize=4, ecolor='black',␣
 ↪color='black')

ax.set_title(r'$\overline{X}_M$ and $M$')
ax.set_ylabel(r'$\overline{X}_M$')
ax.set_xlabel(r'$M$')
```

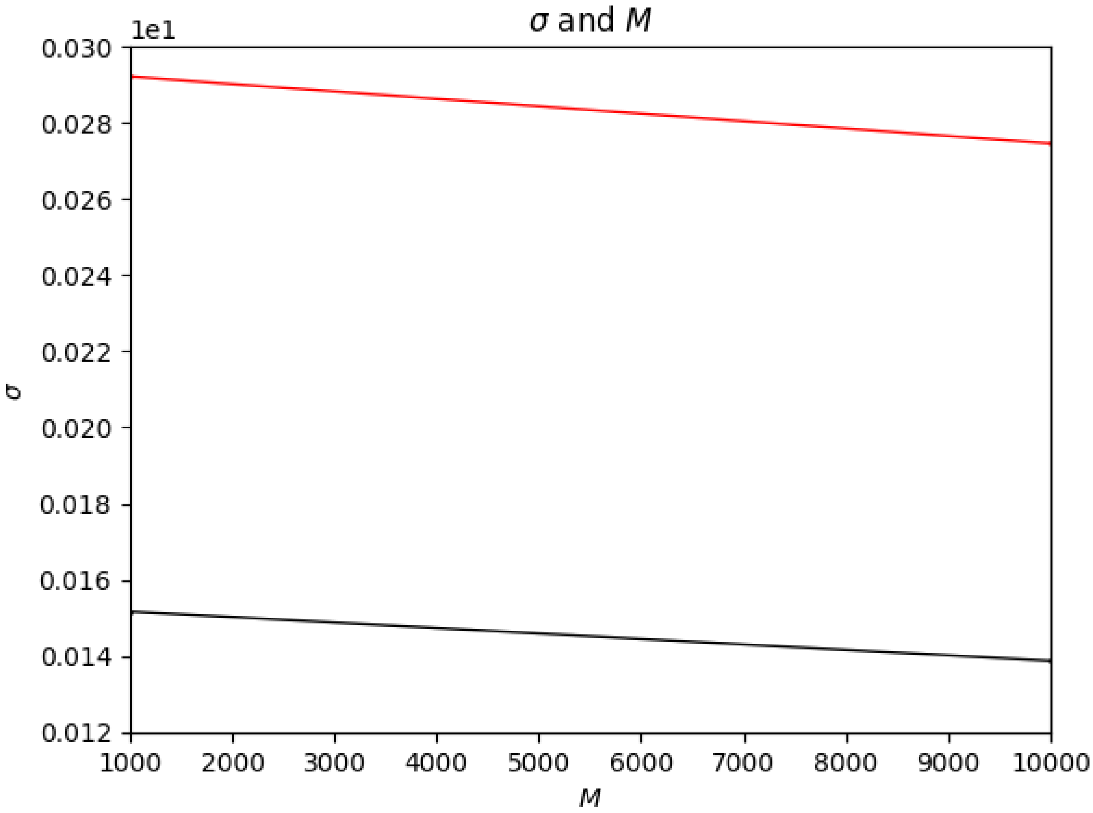
```
[134]: Text(0.5, 0, '$M$')
```



2 Third plot

```
[184]: # test4 = np.array([np.array( 2 * euler(20, xy[0])[0] - element) for element in␣
 ↪euler(20, xy[0])])

antithetic = lambda fixed_u: np.array([np.array( 2 * fixed_u[0] - element) for␣
 ↪element in fixed_u])

# all_xhat = lambda c: np.array([euler(c, valor).T[0][-1] for valor in xy])
```

```
valoresM2 = np.array([100, 1000, 10000])



l1 = lambda mm: np.array([euler(mm, valor).T[0][-1] for valor in xy])
l2 = lambda mm: antithetic(l1(mm))

# print(np.std(l1), np.std(l2), np.std((l1 + l2)*0.5))


y_axis = np.array([np.std(l1( 2 * bigM)) for bigM in valoresM2])
y_axis2 = np.array([np.std((l1(bigM) + l2(bigM)*0.5)) for bigM in valoresM2])

y_axis, y_axis2

# first3, second3 = ultima_lista.T

fig3, ax3 = plt.subplots()
ax3.plot(valoresM2, y_axis, '-o', linewidth=1, markersize=1, color='red')
ax3.plot(valoresM2, y_axis2, '-o', linewidth=1, markersize=1, color='black')

plt.ticklabel_format(axis='both', style='sci')
ax3.set_title(r'$\sigma$ and $M$')
ax3.set_ylabel(r'$\sigma$')
ax3.set_xlabel(r'$M$')
ax3.set_xbound(1000, 10000)
ax3.set_ybound(0.30, 0.12)
ax3.ticklabel_format(axis='y', style='sci', scilimits=(1,1))
```

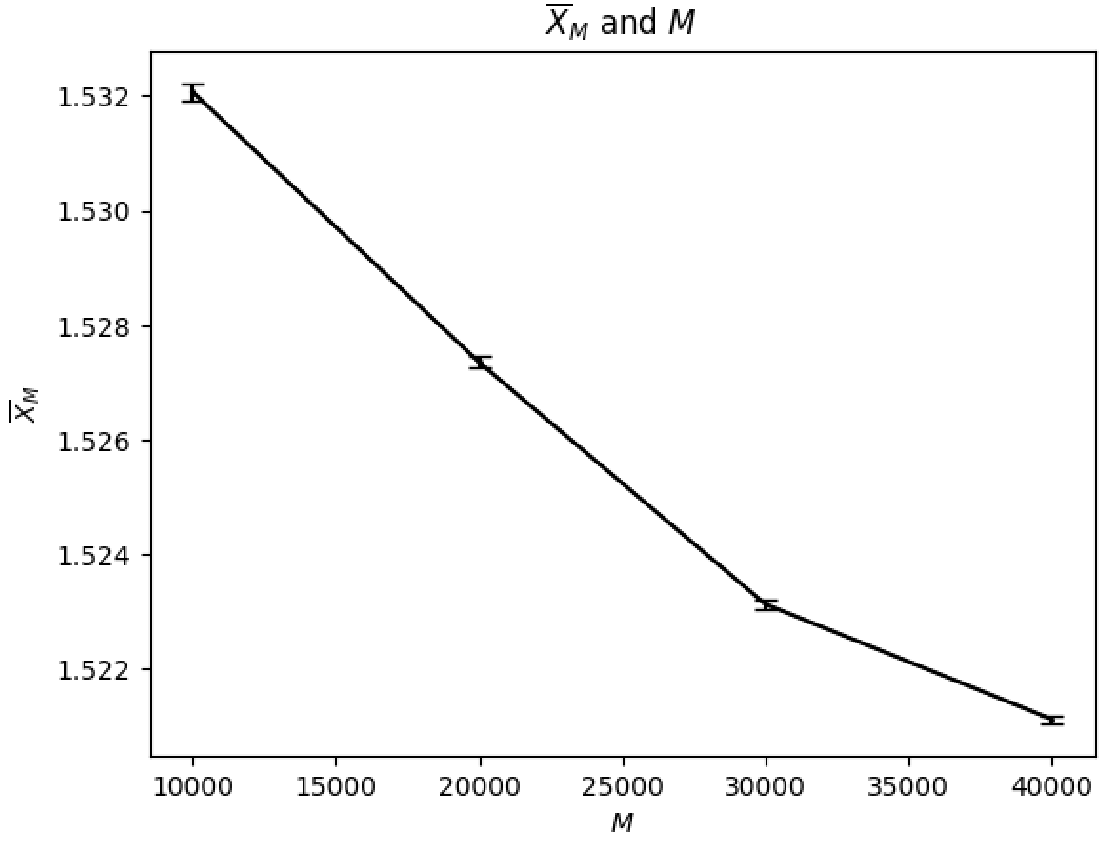


```
[186]: y_axis2
```

```
[186]: array([0.19611491, 0.15167131, 0.13873115])
```

```
[189]: first3, second3 = ultima_lista.T

       # plt.scatter(first, second)
       fig, ax = plt.subplots()
       ax.plot(first3, second3, '-o', linewidth=1, markersize=1, color='black')
       ax.errorbar(first3, second3, ultima_lista_sd * (np.mean(y_axis2))**2,␣
        ↪capsize=4, ecolor='black', color='black')

       ax.set_title(r'$\overline{X}_M$ and $M$')
       ax.set_ylabel(r'$\overline{X}_M$')
       ax.set_xlabel(r'$M$')
```

```
[189]: Text(0.5, 0, '$M$')
```



The reason the standard deviation is 0 is because we are taking the variance of a constant.