REVIEW OF THE MATHEMATICAL CHALLENGES ON THE PRICING OF ASIAN OPTIONS

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Abstract

The evolution and innovation in financial contracts and products are one of the foundations on providing traction to world's economic development.
The modern era of finances has seen emerge many different kinds of products, that aimed to optimize and add dynamism to trading, optimizing capital flows and an efficient movement of wealth in the markets.
From within this framework of innovation, we will be focusing on a product known as Asian option, also known as an average option. A "path-depending exotic option".

Hence, in this paper, we'll try to learn about its history and the reasons that motivated the creation of this specific product, we will analyze its core differences with some of the most known products, such as American's and European's call and put, as well as running simulations to illustrate and successfully observe it's particularities in a comprehensible and pedagogical way, with special attention on the challenges that this specific type of asset presents from a mathematical perspective.

Definition of Asian option

An Asian option, which is also known as an average option it's a "path-depending exotic option".
This means that either the settlement price (the average price at which a contract trades) or the strike of the option (price at which a specific derivative contract can be exercised), is formed by aggregating the underlying asset prices during the option lifetime.

There are two different categories:

- **Average price options (APO):** The payoff at maturity is equal to the average price of the underlying asset over its lifetime, minus the fixed strike price.

$$\begin{cases} \boldsymbol{Average\ price\ call \rightarrow S = max\left(S_{avg} - K, 0\right)} \\ \boldsymbol{Average\ price\ put \rightarrow S = max\left(K - S_{avg}, 0\right)} \end{cases}$$

- **Average Strike Options (ASO):** The payoff is equal to the price of the underlying asset at maturity minus a variable strike, which is equal to the average price of the underlying asset over a period.

$$\begin{cases} \boldsymbol{Average\ strike\ call \rightarrow S = max\left(S_T - S_{avg}, 0\right)} \\ \boldsymbol{Average\ strike\ put \rightarrow S = max\left(S_{avg} - S_T, 0\right)} \end{cases}$$

For this type of option, it does not exist any closed form analytical formula for calculating the theoretical option value. There exist closed form **approximation** formulas for pricing this kind of option.

The main aim of any of this techniques is to reduce risk on the operations of the agents involved in commodities and energy markets, who tend to be exposed to average prices due to their business cycles and their exposure to currency fluctuations.

History of the Asian option

The Asian option has a relatively short history compared to other more average products. Found within the group of "Exotic" products, it was firstly developed in 1987 by Mark Standish and David Spaughton, two analysts in the London-based Bankers & Trust.
The origin of the name is due to the fact that it was firstly developed when the two analysts were working in Tokyo (Asia). It is however used world-wide as a tool to price options linked to the average price of an asset with a specific set of characteristics that will be expanded further on in this paper.

Originally, it was formulated to price options linked to the average price of crude oil. However, since Corporations and exporters for the commodities and energy Markets tend to be exposed to average prices, the popularity of this options grew very quickly.

The first closed-form analysis of geometric mean Asian options was performed by Kemna and Vorst in 1990. (Peter Buchen, An introduction to Exotic option pricing, CRC Financial Mathematics Series)
However, múltiple studies from early dates about arithmetic mean Asian options – the particular type we will be studying in this paper – can be found, and a long list of papers with different mathematical approaches have emerged over the years. We'll be giving a closer look to them further on in this paper.

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Mathematical challenges on the valuation of Asian options

From the mathematical standpoint, Asian options are very challenging. Even when we assume that stock prices follow a Black-Scholes model (and then they have a log-normal distribution), the law of the arithmetic average of the assets is not known. More precisely, this is the law of the sum of correlated log-normal distribution, which has not a known density function. This implies that it is not possible to find a closed-form formula for option prices as in the case of classical call and put options.

Several studies are devoted to studying the construction of numerical methods to approximate option prices:

- Monte Carlo simulation with different variance reduction techniques, e.g. Kemna and Vorst, Clelow and Carverhill and Dufresne;
- to assume a given form for the density of the average in order to obtain closed form approximate solution: Turnbull and Wakeman, Levy, Vorst and the references therein, Milevsky and Posner, Dufresne;
- Methods based in partial differential equations, as in Barraquand and Pudet or in Zvan and al.;Rogers and Shi and Alziary and al.;
- to use the Laplace transform with respect to time to maturity of the option price as in Geman and Yor;
- to give lower and upper bounds for the option price using a conditioning technique as in Curran, Rogers and Shi and more recently Thompson.

We will study the implementation of Monte Carlo methods, which are based on the law of big numbers which we studied in our statistics and probability course. Through them, we will evaluate and compare the differences between the outputs and their variances when applied difference variance reduction techniques, mainly antithetic variance reduction techniques.
These techniques basically aim to reduce the variance of the range of outputs by **reducing the number of required generated samples** to reach a reliable estimator, hence economically improving the accuracy of the simulation.

The underlying principle of the antithetic technique consists of generating a path, and then taking the inverse of the given path and performing an average of both paths:

$$\widehat{\theta_1} = avg(\{s_1, s_2, \dots, s_n\})$$
$$\widehat{\theta_2} = avg(\{-s_1, -s_2, \dots, -s_n\})$$

$$\hat{\theta} = \frac{\widehat{\theta_1} + \widehat{\theta_2}}{2}$$

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Using Black-Scholes model for pricing options

The Black-Scholes model for calculating the premium of an option was introduced in 1973 in a paper entitled, "The Pricing of Options and Corporate Liabilities" published in the Journal of Political Economy. The formula was developed Fischer Black, Myron Scholes and Robert Merton and is maybe the world's most popular pricing model and was worthy of '97's Nobel Prize in Economics.

In a nutshell, the Black-Scholes model is used to calculate the theoretical price of European put and call options, ignoring any dividends paid during the option's lifetime.

The model makes certain important assumptions:

- The options are European and can only be exercised at expiration
- No dividends are paid out during the life of the option
- Efficient markets (market movements cannot be predicted)
- No commissions
- The risk-free rate and volatility of the underlying are known and constant
- Follows a lognormal distribution; that is, returns on the underlying are normally distributed.

And takes the following variables into consideration:

- Current underlying price
- Options strike price
- Time until expiration, expressed as a percent of a year
- Implied volatility
- Risk-free interest rates

The Black Scholes equation looks like follows:

$$\frac{\ln\left(S/K\right)+\left(r+s^2/2\right)t}{s\cdot\sqrt{t}}$$

which, once arranged to our case, situation, and information constraints, can be coded as:

```
math.exp((r-sigma**2/2)*((T*index)/N)+sigma*row))
```

where **row** is the given t moment value generated in the Brownian generator, r is the interest rate (0 in our case), T is the distribution function, Sigma is our standard deviation, index is the current t moment, and N is the computation of N = round(T/dt) where dt is the change magnitude in the Brownian motion generator, in our case 0.0025, and which is hence not equivalent to the N to the BS formula shown above.

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Black Scholes Equation in pricing Asian Options

The Black-Scholes formula will tend to valuate the Asian option at a higher market price. This outcome is to be expected since the Black-Scholes formula applies to standard European options which only, implicitly, considers the underlying asset price at maturity of the option as settlement price. This price is in average higher than the Asian option settlement price when the underlying asset price has a positive drift. However, for some volatility scenarios where there is a drastic volatility shift and the period with higher volatility is before the average period of the option, even the Black-Scholes formula will underestimate the option value.
(based in The effect of volatility on the pricing of Asian options by Erik Wiklund)

Monte-Carlo method

From among the multiple approaches available nowadays in order to enhance the quality of the approximation to the estimated Price of the option, we will use the Monte-Carlo method, due to its easy implementation and effectiveness.

The Monte-Carlo method has a long history and should be well known by now so we'll introduce it briefly. Monte-Carlo methods were initially used to option pricing by Boyle in 1977. We will use It to estimate our mean price.

In short, these methods are used to solve numerical problems that are too complicated to be solved analytically, thus, they are solved through the generation of N amount of numbers from a defined range and normal distribution density function, using heavy computational setups.

In our case, we will generate N amount of series of Brownian motion processes that will emulate the behaviors of an Asian option. A Brownian motion is a stochastic or Wsangiener process that behaves in total chaos and has, therefore, no exact predictability.

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Antithetic variance reduction

Among the multiple methods used to reduce variance in the Monte-Carlo methods, we'll use the one based on antithetic variables.
The main reason for the choice is the simplicity of its mechanics.
Antithetic variates are based on the premise that under certain conditions using N/2 pairs of sample paths that are negatively correlated rather than N individually generated paths leads to a tighter confidence interval.

To see this, we consider our generated path:

```
def blackscholes(x):
    newrow = []
    S_0 = 100
    for index, row in x.iterrows():
        newrow.append(S0*math.exp((r-sigma**2/2)*((T*index)/N)+sigma*row))
    return pd.Series(newrow)
```

but this time we invert our sigma value:

```
def blackscholes_AT(x):
    newrow = []
    S_0 = 100
    for index, row in x.iterrows():
        newrow.append(S0*math.exp((r-sigma**2/2)*((T*index)/N)+(-sigma)*row))
    return pd.Series(newrow)
```

Once we get an inversely generated path, we will then compute an antithetic mean composed by the (regular mean + the antithetic mean)/2, which will output a path with a narrower confidence interval.

We can find this in our session analsys notebook in the 6th line, as:
mean = (float(row['Regular']) + float(row['Antithetic']))/2

for example, we've been able to verify that, in one of our simulations:
Our antithetic paths were 95% of the times between 6.2611390257720938 and 6.2613714217939105;
Our regular path was between 6.2796480203486222 and 6.2802274155902289.

```
In [21]: mean_confidence_interval(Payoffs_Evolution['ATMean'])
Out[21]: (6.2612552237830021, 6.2611390257720938, 6.2613714217939105)
```

```
In [23]: mean_confidence_interval(Payoffs_Evolution['RegularMean'])
Out[23]: (6.2799377179694256, 6.2796480203486222, 6.2802274155902289)
```

Hence proving effectively lowering the interval.

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Simulations methodology

In order to perform our analysis, we've developed a series of scripts that generate and develop a series of computations over time, storing data every N moment, in order to be able to, afterward, observe its evolution and compare the effects of the variance reduction methods over the same Brownian path-dependent generated series.

These computations will be performed with a Python running over a rented Google Cloud computing (n1-highcpu-8 (8 vCPUs, 7.2 GB memory)).
And its analysis will be performed over an ipython notebook for higher comprehension.
We will also be using some common libraries such as Pandas, Numpy and matplotlib for plotting of charts and enhanced processing and data manipulation capabilities.

Due to time and computational constrains, we've limited the number of simulations to 1 million of series per session. And averaged 50 hours per session.
Additionally, a simulation of 2 million series has been generated, spending a total time of nearly 5 days.

We've run a good amount of simulations with different series ranges in order to compare and evaluate the behavior of the variance and its correlations. We chosen the following set of parameters for no special reason but simulation homogeneity:

- r = 0.00 (interest rate)
- strike price = 95
- T = 0.5
- sigma = 0.2
- S0 = 100 (Start price)
- dt = 0.0025 (change magnitude)

The script, generated the following files in order to track the evolution of the simulation (AT for antithetic method):

Files containing a single column with the payoffs from every simulated series.

- payoffs.csv
- payoffsAT.csv

Evolution of the mean of the obtained payoffs:

- meanevolution.csv
- meanevolution_AT.csv

	N	TIMER	Mean	Variance	Standard Dev.
0	100	0 days 00:00:18.624291000	6.545219866	[49.2575420374]	[6.97921672536]
1	200	0 days 00:00:39.823498000	6.459697182	[46.281131229]	[6.80302368282]

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Every Brownian set used

- browniansets.csv:

0 0.0309453436411 -0.031580994578 -0.0196681867769 -0.0744374580025 -0.0192172987174

Simulations Generator code

Public GIT repository for code can be found at
https://github.com/sineline/Seminar_Paper_153958

```
from __future__ import division
import Queue
import threading
import numpy as np
import pandas as pd
import math
from sys import stdout
import uuid
import os
import sys
import datetime
from string import Template


class DeltaTemplate(Template):
    delimiter = "%"

def strfdelta(tdelta, fmt):
    d = {"D": tdelta.days}
    d["H"], rem = divmod(tdelta.seconds, 3600)
    d["M"], d["S"] = divmod(rem, 60)
    d["M"] = str(d["M"]).zfill(2)
    d["H"] = str(d["H"]).zfill(2)
    d["S"] = str(d["S"]).zfill(2)
    t = DeltaTemplate(fmt)
    return t.substitute(**d)


# In[3]:

q = Queue.Queue()

# definim variables output
path="output/"
file_name = os.path.join(path,str(uuid.uuid4()))
meanevolution = []
meanevolution_AT = []
# Definim parametres
quantitat_series = int(sys.argv[1]); # Numero de series a generar.

print "quantitat series a simular: %d" % quantitat_series

r = 0.00
strike_price = 95
T = 0.5
sigma = 0.2
S0 = 100
dt = 0.0025
```

```
N = int(round(T/dt))
t = np.linspace(0, T, N)


# Definim genereadors
# Definim la funcion que farem servir per generar les series:
def genBrownian():
    # trajectoria d'un moviment Brownia

    W0 = 0
    W = np.random.standard_normal(size = N)
    W = W0+np.cumsum(W)*np.sqrt(dt) ### standard brownian motion ###
    #print W
    return W

# Generem l'array amb les series:
sets = [] # Definim el contenidor de les series
set_means = [] # Definim el contenidor de les mitjanes aritmetiques
start = datetime.datetime.now()

print "session id:" , file_name
print "\nstart time: " , start
# Creation of Dataframes from generated lists
npsets = pd.DataFrame(sets)
npmeans = pd.DataFrame(set_means)

# Calculem pay offs
payoffs = []
payoffsBS = []
payoffsBS_Antithetic = []
blackscholes_serie = []

def blackscholes(x):
    newrow = []
    S_0 = 100
    for index, row in x.iterrows():
        newrow.append(S0*math.exp((r-sigma**2/2)*((T*index)/N)+sigma*row))
    return pd.Series(newrow)

def blackscholes_AT(x):
    newrow = []
    S_0 = 100
    for index, row in x.iterrows():
        newrow.append(S0*math.exp((r-sigma**2/2)*((T*index)/N)+(-sigma)*row))
    return pd.Series(newrow)

def newrow():
    serie = genBrownian()
    sets.append(serie)
    payoffsBS.append(max(np.mean(blackscholes(pd.DataFrame(serie)))-
strike_price,0))
    payoffsBS_Antithetic.append(max(np.mean(blackscholes_AT(pd.DataFrame(serie)))-
strike_price,0))

# for index, row in npsets.iterrows():
rowindex = 0
timer = datetime.datetime.now()

while rowindex <= quantitat_series:
    t = threading.Thread(target=newrow)
    t.daemon = True
    t.start()
    elapsed = ""
    rowindex+=1
    secondsdiff = datetime.datetime.now()-start
```

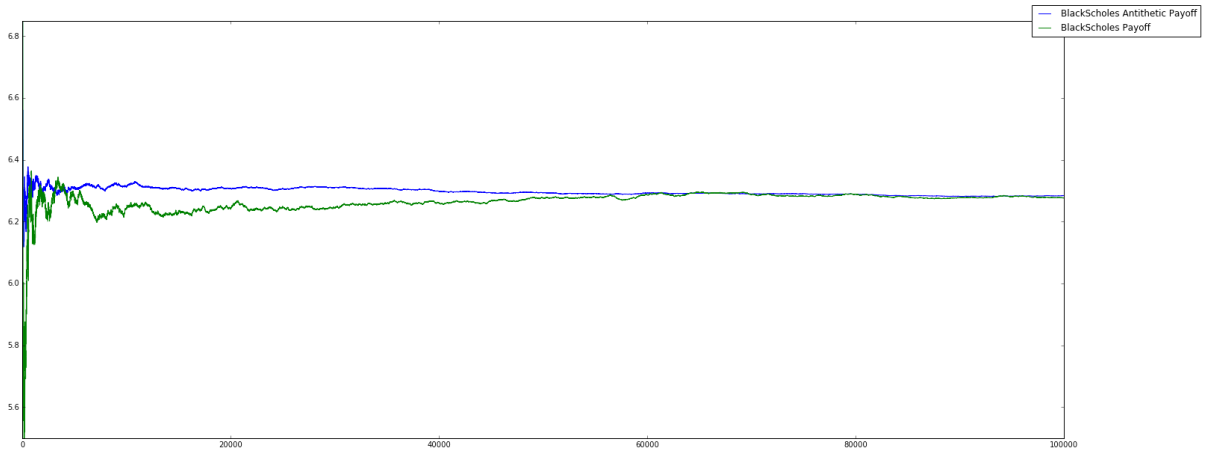
```
    if int(rowindex % 100) == 0:
        elapsed = datetime.datetime.now() - timer
        timer = datetime.datetime.now()
        timediff = (quantitat_series/rowindex)*secondsdiff.total_seconds()
        timeleft = datetime.timedelta(minutes=(timediff/60)) -
(datetime.datetime.now()-start)
        meanevolution.append([rowindex,datetime.datetime.now()-
start,np.mean(payoffsBS),np.var(pd.DataFrame(payoffsBS)),np.std(pd.DataFrame(payoff
sBS))])
        meanevolution_AT.append([rowindex,datetime.datetime.now()-
start,np.mean(payoffsBS_Antithetic),np.var(pd.DataFrame(payoffsBS_Antithetic)),np.s
td(pd.DataFrame(payoffsBS_Antithetic))])
        final = start + datetime.timedelta(seconds=(timediff))
        os.system('cls')  # for Windows
        os.system('clear')  # for Linux/OS X
        stdout.write("\rTime: %s , Mean: %f , Mean AT: %f , Last N computed: %d.
100 Rows iteration time: %s  \b  "
                     "At this speed, computation will finish in %s minutes, the:
%s, at %s" %
                     (datetime.datetime.now()-start, np.mean(payoffsBS),
np.mean(payoffsBS_Antithetic),rowindex, elapsed, strfdelta(timeleft,"%D days
%H:%M:%S"),final.strftime("%D"), final.strftime("%H:%M")))
        stdout.flush()

# Imprimim resultats i exportem outputs:
print "\nFinished at", datetime.datetime.now()
print "total time elapsed: %d", start - datetime.datetime.now()

# Mitjana Payoffs
print np.mean(pd.DataFrame(payoffsBS))
pd.DataFrame(payoffsBS).to_csv(str(file_name)+"_payoffs.csv", sep='\t',
encoding='utf-8')
pd.DataFrame(payoffsBS_Antithetic).to_csv(str(file_name)+"_payoffsAT.csv",
sep='\t', encoding='utf-8')
pd.DataFrame(meanevolution).to_csv(str(file_name)+"_meanevolution.csv", sep='\t',
encoding='utf-8')
pd.DataFrame(meanevolution_AT).to_csv(str(file_name)+"_meanevolution_AT.csv",
sep='\t', encoding='utf-8')
pd.DataFrame(sets).to_csv(str(file_name)+"_browniansets.csv", sep='\t',
encoding='utf-8')

print np.var(pd.DataFrame(payoffsBS))
print np.std(pd.DataFrame(payoffsBS))
print
np.mean(pd.concat([pd.DataFrame(payoffsBS),pd.DataFrame(payoffsBS_Antithetic)]))
```

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

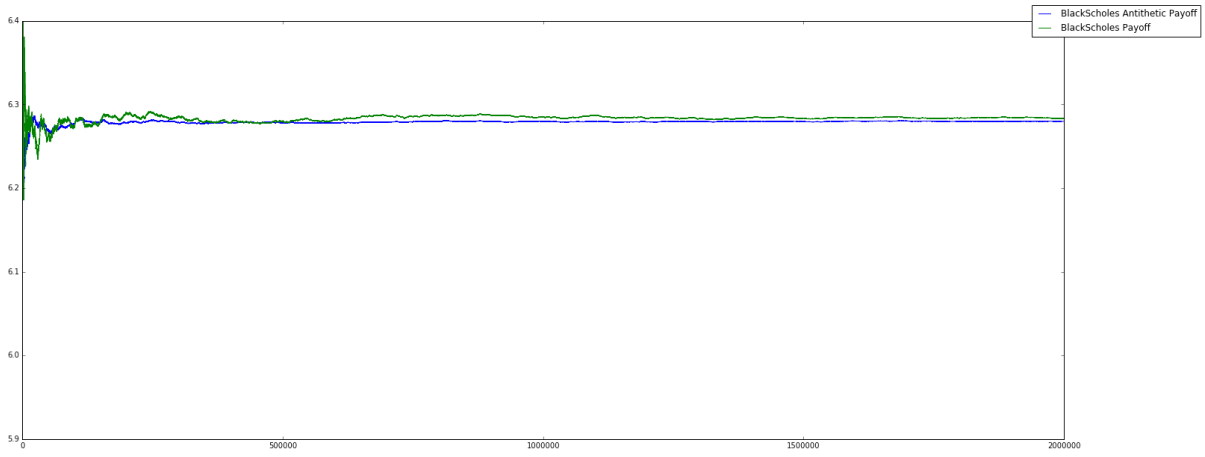
Analysis Results

We've obtained the following means and variances:

Session id	N	Regular Mean	AT Mean	Regular Variance	AT Variance	AT Variance < Reg. Variance
1d392808	99999	6.27772	6.271578	43.537304	43.5308	TRUE
f1da1b0d	99999	6.264059	6.271394	43.637263	43.4143	TRUE
ae088eb1	100000	6.333441	6.281625	44.083238	43.3386	TRUE
f0f60d13	100000	6.276719	6.284222	43.734101	43.922	FALSE
7549a7df	1999990	6.283273	6.27977	43.730709	43.6964	TRUE



Session f0f60d13



Session 7549a7df

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Session Analysis code

```
# coding: utf-8

# In[ ]:

import sys,pandas as pd
import numpy as np
import math
import matplotlib
get_ipython().magic(u'matplotlib inline')
import pylab
import matplotlib.pyplot as plt
from pylab import rcParams
rcParams['figure.figsize'] = 25, 10


# In[ ]:

session_id = ''
Payoffs_Evolution_Without_Antithetic =
pd.read_csv('output/'+session_id+'_payoffs.csv',sep="\t",index_col='Unnamed: 0')
Payoffs_Evolution_Antithetic =
pd.read_csv('output/'+session_id+'_payoffsAT.csv',sep="\t",index_col='Unnamed: 0')
Payoffs_Evolution = pd.DataFrame()
Payoffs_Evolution['Regular'] = Payoffs_Evolution_Without_Antithetic
Payoffs_Evolution['Antithetic'] = Payoffs_Evolution_Antithetic


# In[ ]:

Payoffs_Evolution.head(5)


# In[ ]:

fig2, ax = plt.subplots()
ax.plot(Payoffs_Evolution['Regular'], label="BlackScholes Payoff")
ax.plot(Payoffs_Evolution['Antithetic'], label="BlackScholes Antithetic Payoff")
plt.legend(bbox_to_anchor=(1.05, 1), loc=10, borderaxespad=0.)
plt.show()


# In[ ]:

Means_Aggregates = pd.DataFrame()


# In[ ]:

for index, row in Payoffs_Evolution.iterrows():
    mean = (float(row['Regular']) + float(row['Antithetic']))/2
    Payoffs_Evolution.set_value(index, 'ATMean_T', mean)


# In[ ]:

Payoffs_Evolution['RegularMean'] =
Payoffs_Evolution['Regular'].expanding(min_periods=1).mean()
Payoffs_Evolution['ATMean'] =
Payoffs_Evolution['ATMean_T'].expanding(min_periods=1).mean()


# In[ ]:
```

```
Payoffs_Evolution['RegularVar'] =
Payoffs_Evolution['Regular'].expanding(min_periods=1).var()
Payoffs_Evolution['ATVar'] =
Payoffs_Evolution['Antithetic'].expanding(min_periods=1).var()


# In[ ]:

np.var(Payoffs_Evolution['Antithetic'])


# In[ ]:

np.var(Payoffs_Evolution['Regular'])


# In[ ]:

# Do we manage to reduce the variance when we apply Antithetic variance reduction
methods?
np.var(Payoffs_Evolution['Antithetic'])<np.var(Payoffs_Evolution['Regular'])


# In[ ]:

Payoffs_Evolution.tail(5)


# In[ ]:

fig2, ax = plt.subplots()
ax.plot(Payoffs_Evolution['ATMean'], label="BlackScholes Antithetic Payoff")
ax.plot(Payoffs_Evolution['RegularMean'], label="BlackScholes Payoff")

plt.legend(bbox_to_anchor=(1.05, 1), loc=10, borderaxespad=0.)
plt.ylim([5.9,6.4])
plt.show()
```

https://github.com/sineline/Review-of-the-mathematical-challenges-on-the-pricing-of-asian-options-

Conclusions

The Monte Carlo method has been proved to be a powerful tool in numerical option pricing. Nevertheless, as we can see in the numerical examples, the number of needed simulations use to be very high. In order to overcome this problem, several variance reduction methods have been proposed in the literature. In this work, we have implemented the antithetic variables method.

Bibliografia

https://en.wikipedia.org/wiki/Antithetic_variates
http://www.goddardconsulting.ca/option-pricing-monte-carlo-vr.html#antitheticvariates
https://www.rocq.inria.fr/mathfi/Premia/free-version/doc/premia-doc/pdf_html/asian_doc/asian_doc.html
https://en.wikipedia.org/wiki/Antithetic_variates
http://www.math.kth.se/matstat/seminarier/reports/M-exjobb12/120f412a.pdf
http://www.fincad.com/resources/resource-library/wiki/asian-options
http://www.wikinvest.com/wiki/Exotic_Options_-_Asian_Options
http://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=4301159&fileOId=4301160
https://www.rocq.inria.fr/mathfi/Premia/free-version/doc/premia-doc/pdf_html/asian_doc/
http://quantlabs.net/academy/download/free_quant_instituitional_books_/[Nielsen]%20Pricing%20Asian%20Options.pdf
http://homepage.ntu.edu.tw/~jryanwang/course/Financial%20Computation%20or%20Financial%20Engineering%20(graduate%20level)/FE_Ch10%20Asian%20Options.pdf
(Peter Buchen, An introduction to Exotic option pricing, CRC Financial Mathematics Series)
http://www.fincad.com/resources/resource-library/wiki/asian-options
http://www.wikinvest.com/wiki/Exotic_Options_-_Asian_Options
http://www.stat.columbia.edu/~vecer/asian-vecer.pdf
http://www.math.kth.se/matstat/seminarier/reports/M-exjobb12/120412a.pdf
http://www.riskencyclopedia.com/articles/asian-option-average-option/
http://www.businessdictionary.com/definition/Asian-option.html
http://www.sdgm.com/support/glossary.aspx?term=Asian
http://www.fincad.com/resources/resource-library/wiki/asian-options
https://www.diva-portal.org/smash/get/diva2:301070/FULLTEXT01.pdf
https://en.wikipedia.org/wiki/Black%E2%80%93Scholes_model
http://www2.warwick.ac.uk/fac/soc/wbs/subjects/finance/research/wpaperseries/1994/94-54.pdf
http://www.math.kth.se/matstat/seminarier/reports/M-exjobb12/120412a.pdf
http://www.investopedia.com/university/options-pricing/black-scholes-model.asp
http://www.investopedia.com/terms/b/blackscholes.asp
http://bradley.bradley.edu/~arr/bsm/pg04.html
http://www.investopedia.com/terms/a/asianoption.asp
http://www.math.kth.se/matstat/seminarier/reports/M-exjobb12/120412a.pdf
http://stackoverflow.com/questions/13202799/python-code-geometric-brownian-motion-whats-wrong
http://www.espenhaug.com/black_scholes.html