Computational Finance and its implementation in Python with applications to option pricing

Andrea Mazzon

Ludwig Maximilians Universität München

Main contents

1. Monte-Carlo method for option pricing and variance reduction techniques
   - The Monte-Carlo method: motivation and a brief overview
   - Variance reduction techniques
     - Introduction
     - Antithetic variables
     - Control variates

2. Option pricing under the Binomial model
   - Motivation and setting
   - Simulation of the Binomial model
   - Calibration of the Binomial model
   - American options valuation

Motivation

- A common problem we face in mathematical finance is the risk neutral valuation of a derivative.
- As you know, the price of a derivative is expressed by the (possibly discounted) expectation of its payoff at maturity, under a pricing measure (also called pricing, or martingale measure).
- That is, we have to compute the expectation of a random variable.
- Problem: most often, there is no way to get an analytic formula for the expectation of complex derivates, or even simpler derivatives written on underlying with non trivial dynamics.
- Broad idea: we can approximate the price by averaging some possible, simulated realizations of the payoff.
- The strong law of large numbers and some other convergence results may help us.

A bit more precisely..

- Consider a random variable $X : \Omega \to \mathbb{R}^N$ defined on a probability space $(\Omega, \mathcal{F}, P)$. The probability measure P may be viewed as a risk neutral measure.
- Also consider a (payoff) function $f : \mathbb{R}^N \to \mathbb{R}$ such that Var$[f(X)] < \infty$.
- The aim is to compute the expectation

$$\theta := \mathbb{E}^P[f(X)] = \int_\Omega f(X)dP.$$

- Suppose there is no analytic formula to derive θ above. We have to find an approximation $\hat{\theta}$.

We can define independent drawings of X

- Given $X:\Omega\to\mathbb{R}$ and $(\Omega,\mathcal{F},P)$ as above, introduce:

$$\tilde{\Omega} := \Omega\times\Omega\times\cdot\cdot\times\Omega = \{\tilde{\omega}=(\omega_1,\dots,\omega_n),\omega_i\in\Omega\},$$
$$\tilde{\mathcal{F}} := \sigma(\mathcal{F}\times\mathcal{F}\times\cdot\cdot\times\mathcal{F}),$$
$$\tilde{P}\left(\prod_{i=1}^n A_i\right) := \prod_{i=1}^n P(A_i), \quad A_i\in\mathcal{F}.$$

- Also define the random variable $\tilde{X}=(\tilde{X}_1,\dots,\tilde{X}_n)$ by $\tilde{X}_i(\tilde{\omega}) := X(\omega_i)$.
- This is a way to see $\tilde{X}(\tilde{\omega})$ as n different realizations $X(\omega_i)$, $i=1,\dots,n$ of one random variable X, or as one realization of n i.i.d. random variables $\tilde{X}_i(\tilde{\omega})$, $i=1,\dots,n$.
- This interpretation is at the base of the Monte-Carlo method, as it permits to exploit the Strong Law of Large Numbers.
- A similar construction and interpretation can be given for a N-dimensional random variable X.

Convergence results for sequences of i.i.d. random variables

Theorem: Strong Law of Large Numbers

Let $(X_i)_{i\in\mathbb{N}}$ be i.i.d. integrable real valued random variables on $(\Omega, \mathcal{F}, P)$, and set

$$\mu := \mathbb{E}^P[X_i], \quad i \in \mathbb{N}.$$

Then

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^n X_i = \mu \quad P-a.s.$$

Theorem: Tschebyscheff Inequality

Let $(X_i)_{i\in\mathbb{N}}$ be i.i.d. square integrable real valued random variables on $(\Omega, \mathcal{F}, P)$, and set

$$\mu := \mathbb{E}^P[X_i], \quad \sigma^2 := \mathbb{E}^P\left[(X_i-\mu)^2\right], \qquad i \in \mathbb{N}.$$

Then for any $\epsilon, \delta > 0$ we have

$$P\left(\left|\frac{1}{n}\sum_{i=1}^n X_i - \mu\right| \geq \epsilon\right) \leq \frac{\sigma^2}{\epsilon^2 n}.$$

and

$$P\left(\left|\frac{1}{n}\sum_{i=1}^n X_i - \mu\right| \geq \frac{\sigma}{\delta^{1/2}n^{1/2}}\right) \leq \delta.$$

Lemma

Let $(X_i)_{i\in\mathbb{N}}$ be a collection of i.i.d. integrable real valued random variables on $(\Omega, \mathcal{F}, P)$, and let $f : \mathbb{R}^N \to \mathbb{R}$. Then the random variables $(f(X_i))_{i\in\mathbb{N}}$ are also i.i.d.

- The lemma above, together with the convergence results of the previous slide, allows us to approximate

  $$\theta := \mathbb{E}^P[f(X)] = \int_\Omega f(X)dP$$

  by

  $$\hat{\theta} := \frac{1}{n}\sum_{i=1}^{n} f(X_i),$$

  where $(X_i)_{i=1,\dots,n}$ are independent realizations of X.
- We can generate numerically n realizations of a random variable X with a given distribution P^X, starting from a sequence of (pseudo!) random numbers.
- One must give a *seed*, i.e., a starting point for the pseudo-random numbers sequence.
- The realizations will not be purely random, and not purely independent.

- Pro:
  - It is very simple to understand and easy to implement.
  - The accuracy does not depend on the domain dimension (i.e., if we simulate N-dimensional random variables the accuracy is the same).
  - The accuracy can be increased by just adding more valuations without loosing the previous estimates.
  - The function f does not need to be continuous, but only square integrable.
- Cons:
  - Look at the Tschebyscheff Inequality: we only have a probabilistic bound. The worst case error is ∞.
  - The estimates depend on the generated random sequence. The sequence is not purely random. First, one has to find a good random number generator.
- There are techniques that can be used to increase the accuracy. In the next slides we will see few of them.

Remark

If X has uniform distribution or has a cumulative distribution function F which is easy to invert (in that case a realization x_i can be generated as $x_i = F^{-1}(u_i)$, with u_i realization of $U \sim U((0,1))$) then approximating $\mathbb{E}[f(X)]$ reduces to approximate

$$\int_0^1 G(x)dx, \tag{1}$$

for $G : [0,1] \to \mathbb{R}$.

Theorem: Koksma-Hlawka inequality

If G has bounded total variation on $(0,1)$, then for any points $x_1, \dots, x_n \in (0,1)$ it holds

$$\left| \frac{1}{n} \sum_{i=1}^{n} G(X_i) - \int_0^1 G(x)dx \right| \leq V(G) D^*(x_1, \dots, x_n),$$

where

$$V(G) = \sup_S \sum_i |G(y_{i+1}) - G(y_i)|$$

over all partitions $S := \{0 = y_1 < y_2 < \cdots < y_n = 1\}$ and $D^*(x_1, \dots, x_n)$ is the star discrepancy

$$D^*(x_1, \dots, x_n) = \sup_{b \in (0,1)} \left| \frac{|\#\{x_i : 0 \leq x_i \leq b\}|}{n} - b \right|.$$

Low-discrepancy sequences

- The result in the previous slide also holds for higher dimensions (here we just wanted to simplify the notation).
- It gives the motivation to look for low discrepancy sequences.
- Most well known low discrepancy sequences: Van der Corput, Halton, Sobol, Hammersley, Sobol, Niederreiter.
- Here we don't focus on Low discrepancy sequences. A bit of references if you want to go deeper on this:
  - J. Dick and F. Pillichshammer, *Digital Nets and Sequences. Discrepancy Theory and Quasi-Monte Carlo Integration*, Cambridge University Press, Cambridge, 2010
  - M. Drmota and R. F. Tichy, *Sequences, discrepancies and applications*, Lecture Notes in Math., 1651, Springer, Berlin, 1997.
  - L. Kuipers, H. Niederreiter, *Uniform distribution of sequences*, Dover Publications, 2005.
  - ...the course *Numerical Methods for Financial Mathematics* at our master!
- We focus instead on variance reduction techniques.

- Consider a random variable $X : \Omega \to \mathbb{R}^N$ defined on a probability space $(\Omega, \mathcal{F}, P)$ and a (payoff) function $f : \mathbb{R}^N \to \mathbb{R}$ such that Var$[f(X)] < \infty$.
- Monte-Carlo method: choosing $n \in \mathbb{N}$ large enough, we approximate

$$\hat{\mu} := \frac{1}{n}\sum_{k=1}^{n} f(X_i) \approx \mu := \mathbb{E}^P[f(X)],$$

where $(X_i)_{i=1,\dots,n}$ are realizations of X, i.e., have same distribution as X.
- The estimator is of course *unbiased*, i.e.,

$$\mathbb{E}^P\left[\hat{\mu}\right] = \mathbb{E}^P\left[\frac{1}{n}\sum_{k=1}^{n} f(X_i)\right] = \mathbb{E}^P\left[f(X)\right] =: \mu$$

- We are interested in the variance of our estimator, i.e., in the quantity

$$\text{Var}\left(\hat{\mu}\right) = \mathbb{E}^P\left[\left(\frac{1}{n}\sum_{k=1}^{n} f(X_i) - \mu\right)^2\right].$$

- We have seen that if $(X_i)_{i=1,\dots,n}$ are independent, we have convergence results for our estimator. Moreover,

$$\text{Var}\,(\hat{\mu}) = \mathbb{E}^P\left[\left(\frac{1}{n}\sum_{k=1}^{n} f(X_i) - \mu\right)^2\right] = \frac{1}{n}\text{Var}[f(X)].$$

- It makes sense: the larger the number n of simulated realizations of X, the smaller the variance of our estimator.
- In particular, we have to increase the number of simulations by a factor of C to reduce the standard deviation by a factor of $\sqrt{C}$.
- The question now is: can we do it better?
- Variance reduction techniques aim to reduce the variance of our estimator, without increasing the number of simulations.

Some variance reduction techniques

Three well known variance reduction techniques are:

- Antithetic variables
- Control variates
- Importance sampling

We will focus mostly on the first two techniques, together with applied examples. Here some references if you want to deepen Importance sampling:

- A, Bouhari. *Adaptative Monte Carlo Method, A Variance Reduction Technique*. Monte Carlo Methods and Their Applications. 10 (1): 1-24, 2004.
- P. J. Smith, M. Shafi, H. Gao. *Quick simulation: A review of importance sampling techniques in communication systems*. IEEE Journal on Selected Areas in Communications. 15 (4): 597-613, 1997.
- Again, the course *Numerical Methods for Financial Mathematics* at our master!

Let's start from a simple result..

Lemma

Let $f, h : \mathbb{R} \to \mathbb{R}$ be two monotone functions, both increasing or both decreasing, and let $X : \Omega \to \mathbb{R}$ be a random variable defined on a probability space $(\Omega, \mathcal{F}, P)$. Then

$$\mathbb{E}^P[f(X)h(X)] \geq \mathbb{E}^P[f(X)]\mathbb{E}^P[h(X)].$$

Proof

The monotonicity assumption on f and h implies that for any $x, y \in \mathbb{R}$ we have

$$(f(x) - f(y))(h(x) - h(y)) \geq 0.$$

Therefore, for any i.i.d. real valued random variables X and Y on $(\Omega, \mathcal{F}, P)$ it holds

$$(f(X) - f(Y))(h(X) - h(Y)) \geq 0$$

and then

$$\mathbb{E}\left[(f(X) - f(Y))(h(X) - h(Y))\right] \geq 0,$$

so that

$$\mathbb{E}\left[f(X)h(X)\right] + \mathbb{E}\left[f(Y)h(Y)\right] \geq \mathbb{E}\left[f(Y)h(X)\right] + \mathbb{E}\left[f(X)h(Y)\right].$$

Since X and Y are identically distributed, it follows that

$$2\mathbb{E}\left[f(X)h(X)\right] \geq 2\mathbb{E}\left[f(Y)h(X)\right],$$

and since they are also independent, this implies that

$$\mathbb{E}^P[f(X)h(X)] \geq \mathbb{E}^P[f(X)]\mathbb{E}^P[h(X)].$$

An interesting consequence

Proposition

Let $f : \mathbb{R} \to \mathbb{R}$ be a monotone function, and $X : \Omega \to \mathbb{R}$ a random variable defined on a probability space $(\Omega, \mathcal{F}, P)$. Then

$$\text{Cov}[f(X), f(-X)] \leq 0.$$

Proof

We have that

$$\text{Cov}[f(X), f(-X)] = \mathbb{E}^P[f(X)f(-X)] - \mathbb{E}^P[f(X)]\mathbb{E}^P[f(-X)].$$

The result then follows since a direct application of the Lemma of the previous slide with $h(x) := -f(-x)$ implies that

$$\mathbb{E}^P[f(X)]\mathbb{E}^P[f(-X)] \geq \mathbb{E}^P[f(X)f(-X)].$$

Application to Monte-Carlo

- Let $f : \mathbb{R} \to \mathbb{R}$ be a monotone function, and let $X : \Omega \to \mathbb{R}$ be a symmetric random variable defined on a probability space $(\Omega, \mathcal{F}, P)$.
- From the last proposition we know that

$$\text{Cov}[f(X), f(-X)] \leq 0.$$

- Idea: choose n even and generate $n/2$ realizations of X, call them $(X_i)_{i=1,\dots,n/2}$. Then define $X_{n/2+i} := -X_i$, $i = 1, \dots, n/2$.
- Since X is symmetric, the estimator is unbiased:

$$\mathbb{E}[\hat{\mu}] = \frac{1}{n}\mathbb{E}\left[\sum_{k=1}^{n/2} f(X_i) + \sum_{k=1}^{n/2} f(-X_i)\right] = \frac{1}{n}\left(\sum_{k=1}^{n/2} \mathbb{E}[f(X_i)] + \sum_{k=1}^{n/2} \mathbb{E}[f(-X_i)]\right) = \mu.$$

- What about the variance?

$$\begin{aligned}
\text{Var}[\hat{\mu}] &= \frac{1}{n^2}\text{Var}\left[\sum_{k=1}^{n/2} f(X_i) + \sum_{k=1}^{n/2} f(-X_i)\right] \\
&= \frac{1}{n^2}\left(n\text{Var}[f(X)] + \text{Cov}\left(\sum_{k=1}^{n/2} f(X_i), \sum_{k=1}^{n/2} f(-X_i)\right)\right) \\
&= \frac{1}{n}\text{Var}[f(X)] + \frac{1}{n}\text{Cov}[f(X), f(-X)] \leq \frac{1}{n}\text{Var}[f(X)].
\end{aligned}$$

Application to Monte-Carlo

- To recap: if X is symmetric, then setting $X_{n/2+i} := -X_i$ for $i = 1, \dots, n/2$ gives us an unbiased estimator $\hat{\mu}$ such that

$$\text{Var}[\hat{\mu}] \leq \frac{1}{n}\text{Var}[f(X)].$$

- But $\frac{1}{n}\text{Var}[f(X)]$ is the variance of the classical estimator, when we generate n i.i.d. realizations of X!
- In this way, we reduce the variance of the estimator.
- This approach is known as Antithetic variables.

Antithetic variables for not symmetric X

- Let $f : \mathbb{R} \to \mathbb{R}$ be a monotone function, and let $X : \Omega \to \mathbb{R}$ be a random variable defined on a probability space $(\Omega, \mathcal{F}, P)$.
- Suppose X to be not symmetric. How can we apply Antithetic variables to reduce the variance of our estimator?
- Call F the cumulative distribution function of X. Suppose that we know (at least a good approximation of) F^{-1}.
- Well known result: let $U \sim$ Unif$(0,1)$ and define $Y := F^{-1}(U)$. Then X and Y have same distribution.
- Let $U \sim$ Unif$(0,1)$. Because of the result above, we have

  $$\mathbb{E}^P[f(X)] = \mathbb{E}^P[h(U)]$$

  with $h(x) = f \circ F^{-1}$.
- Since U is symmetric, we can approximate $\mathbb{E}^P[h(U)]$ by Antithetic variables.

- We want to test the benefits of using Antithetic variables in the valuation of a call option under the Black-Scholes model.
- This is indeed a case when we have of course the benchmark of the analytic formula for a call option.
- In particular, we want to approximate the expectation $\mathbb{E}^P[g(X_T)]$ for $T > 0$, in the case when

$$g(x) = (x - K)^+$$

with $K > 0$ and $X = (X_t)_{0 \le t \le T}$ is a stochastic process with initial value $X_0 = x_0$ and dynamics

$$dX_t = rX_t dt + \sigma X_t dW_t, \quad 0 \le t \le T,$$

where $W = (W_t)_{0 \le t \le T}$ is P-Brownian motion.
- Interpretation: r is the risk free rate and P is the martingale measure, i.e., the probability measure under which the discounted process $(e^{-rt}X_t)_{0 \le t \le T}$ is a martingale.

Valuation with Antithetic variables

- The problem reduces to the valuation of the expectation

$$\mathbb{E}^P[g(X-K)^+]$$

where X is the random variable

$$X = X_0 e^{(r-\sigma^2)T+\sigma\sqrt{T}Z},$$

with $Z \sim \mathcal{N}(0,1)$.

- That is, we have to valuate

$$\mathbb{E}^P\left[f(Z)\right]$$

where

$$f(z) = \left(X_0 e^{(r-\sigma^2)T+\sigma\sqrt{T}z} - K\right)^+.$$

- So, we have a function of a symmetric random variable! We can directly use Antithetic variables.
- We simulate $n/2$ realizations $(z_i)_{i=1,\dots,n/2}$ of a standard normal random variable and then define $z_{i+n/2} = -z_i$, $i = 1,\dots,n/2$.

- In the Python package

  `montecarlovariancereduction.antitheticvariables`

  you can find the code relative to the comparison of Antithetic variables against the standard Monte-Carlo method.
- In particular, in the class `GenerateBlackScholes` we generate the values of

$$X = X_0 e^{(r-\sigma^2)T+\sigma\sqrt{T}Z},$$

  starting from the ones of Z. We do this using both the standard Monte-Carlo approach and the Antithetic variables approach illustrated in the previous slide.
- Note that the method

  `numpy.random.standard_normal(n)`

  generates n returns of a standard normal random variable. In this case, we give no seed: it will be different every time this method is called.

Experiment and results

In

`antitheticVariablesTest`

and

`compareStandardMCWithAV`

we do the following experiment:

- We fix the parameters $X_0 = K = 100$, $T = 3$, $r = 0.05$, $\sigma = 0.5$.
- For any number of simulations $n = 10^3$, 10^4, 10^5 and 10^6, we perform 100 different valuations of the price of the call option, both with the standard and the Antithetic variables Monte-Carlo method.
- We then compute the average percentage error for both the methods.

The following table illustrates the results:

	$n = 10^3$	$n = 10^4$	$n = 10^5$	$n = 10^6$
av. $\%$ error standard MC	6.25	2.07	0.59	0.20
av. $\%$ error AV	5.51	1.77	0.53	0.17

Setting and motivation

- Let $X, Y : \Omega \to \mathbb{R}$ be two random variables defined on a probability space $(\Omega, \mathcal{F}, P)$.
- Suppose you know the analytic value of

$$a := \mathbb{E}^P[X], \qquad \sigma_X^2 := \text{Var}[X], \qquad \sigma_{XY} := \text{Cov}[X, Y],$$

  and also suppose $\sigma_{XY} > 0$.
- Assume you want to approximate

$$b := \mathbb{E}^P[Y]$$

- The goal is to find an unbiased estimator of b which has low variance.

- Idea: simulate n realizations (x_i, y_i), $i = 1, \dots, n$, of (X, Y). Compute then

$$\hat{a} := \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad \hat{b} := \frac{1}{n}\sum_{i=1}^{n} y_i.$$

- Note that

$$\text{Cov}[\hat{a}, \hat{b}] = \frac{1}{n}\sigma_{XY}.$$

- What about an estimator

$$\hat{b}_{CV} := \hat{b} - \beta(\hat{a} - a)$$

  for a given $\beta > 0$?
- It is unbiased:

$$\mathbb{E}^P[\hat{b}_{CV}] = \mathbb{E}^P[\hat{b}] + \beta\mathbb{E}^P[\hat{a} - a] = b.$$

- What about the variance?

$$\text{Var}[\hat{b}_{CV}] = \frac{1}{n}\sigma_Y^2 + \beta^2\frac{1}{n}\sigma_X^2 - 2\beta\frac{1}{n}\sigma_{XY}.$$

- It is minimized by $\beta = \frac{\sigma_{XY}}{\sigma_X^2}$. For such a value of β, we find

$$\text{Var}[\hat{b}_{CV}] = \text{Var}[\hat{b}] - \frac{1}{m}\frac{\sigma_{XY}^2}{\sigma_X^2}.$$

Control variates - 2

- We have seen that taking

$$\hat{b}_{CV} := \hat{b} - \beta(\hat{a} - a), \qquad \beta = \frac{\sigma_{XY}}{\sigma_X^2}$$

gives an optimal variance

$$\text{Var}[\hat{b}_{CV}] = \text{Var}[\hat{b}] - \frac{1}{m}\frac{\sigma_{XY}^2}{\sigma_X^2}.$$

- Note that the gain of the new estimator with respect to the old one only depends on the correlation of X and Y:

$$\frac{\text{Var}[\hat{b}_{CV}]}{\text{Var}[\hat{b}]} = \left(1 - \frac{\sigma_{XY}^2}{\sigma_X^2}\right) = 1 - \rho_{XY}^2.$$

- Problem: we have to compute $\beta = \frac{\sigma_{XY}}{\sigma_X^2}$, but often we don't know σ_X^2 and σ_{XY}.
- Solution: estimate σ_X^2 and σ_{XY} from the generated sample, i.e., set

$$\hat{\sigma}_X^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{a})^2, \qquad \hat{\sigma}_{XY} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{a})(y_i - \hat{b})$$

and choose

$$\beta = \frac{\hat{\sigma}_{XY}}{\hat{\sigma}_X^2}.$$

Control variates - 3

- Note that this last choice of β actually depends on the sample $\{(x_i, y_i), \quad i = 1, \ldots, n.\}$
- For this reason, the associated estimator $\hat{b}_{CV} := \hat{b} - \beta(\hat{a} - a)$ is unbiased only asymptotically.

Exercise: multi-dimensional control variates

Consider now the case when X has values in $\mathbb{R}^N$, $N \geq 1$.
Assume you know the $N \times N$ matrix Cov$(X) =: \Sigma_X$ and the N-dimensional vector Cov$(X) = \sigma_{X,Y}$. Also assume that Σ_X is positive definite.
Consider the estimator

$$\hat{b}_{CV} = \hat{b} - (\hat{a} - a)^T \beta,$$

where β is a N-dimensional vector.
Find the optimal β that minimizes the variance of the estimator above and compute the variance for the optimal β you found.

Application: Cliquet options

- Cliquet options are an example of exotic, path dependent options. In particular, their payoff depends on the returns of the underlying.
- Let $X = (X_t)_{t \in [0,T]}$ be a stochastic process on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, P)$.
- Fix a partition

$$0 = t_0 < t_1 < \cdots < t_N := T$$

  of the interval $[0, T]$.
- For any $n = 1, \dots, N$ define $R_n^* := (R_n)_{[F_\ell, C_\ell]}$ for $F_\ell < C_\ell$, where

$$R_n := \frac{X_{t_n}}{X_{t_{n-1}}} - 1$$

  is the n-return and $(x)_{[a,b]} := \min\left(\max(x, a), b\right)$, $a < b$, is the truncation of x.
- The payoff of the Cliquet option with local floor and cap F_ℓ, C_ℓ, global floor and cap $F_g < C_g$ and monitoring dates $0 < t_1 < \cdots < t_N := T$ is then

$$R_g^* := (R_g)_{[F_g, C_g]}$$

  where

$$R_g = R_1^* + R_2^* + \cdots + R_N^*.$$

Control variates for Cliquet options: motivation

- There is no analytic formula for the expectation of the payoff of a Cliquet option, not even under the Black-Scholes model.
- Observation: there is a of course positive correlation between $R_g^* := (R_g)_{[F_g, C_g]}$ and R_g, and also between R_g^* and R_k^*, $k = 1, \dots, N$, since

$$R_g = R_1^* + R_2^* + \cdots + R_N^*.$$

- Can we find an analytic formula for the expectation of R_g and R_n^*, at least under suitable models as Black-Scholes or Bachelier?

Lemma

Let $b > a$. The truncating function $(x)_{[a,b]} := \min\left(\max(x,a), b\right)$ can be rewritten as

$$(x)_{[a,b]} = a + (x-a)^+ - (x-b)^+.$$

Proof

We have that

$$\begin{aligned} a + (x-a)^+ - (x-b)^+ &= a + \max(x-a, 0) + \min(b-x, 0)^+ \\ &= \max(x,a) + \min(b-x,0)^+. \end{aligned}$$

We then easily see that both $\min\left(\max(x,a), b\right)$ and the function above return a when $x < a$, x if $a \le x \le b$ and b if $x > b$.

- The lemma in the previous slide tells us that, defining $Y_n := R_n + 1$, the quantity R_n^* can be seen as the difference between two payoffs off call options, plus a constant:
$$R_n^* = F_\ell + (Y_n - (F_\ell + 1))^+ - (Y_n - (C_\ell + 1))^+.$$
- That is, we have an analytic formula for the expectation of R_n^*, at least under suitable models like Black-Scholes or Bachelier.
- However, it is it reasonable to expect that R_g^* and R_g are more correlated than R_g^* and R_n^*.
- So, what about an analytic formula for the expectation of
$$R_g = R_1^* + R_2^* + \cdots + R_N^*?$$
This comes directly from the one for R_n^* if the returns are independent!
- This is the case for the Black-Scholes model. But not for Bachelier! (Why?)

- We assume that our underlying X follows dynamics

$$dX_t = rX_t dt + \sigma X_t dW_t, \quad 0 \leq t \leq T$$

under the martingale measure P.

- Then the returns are given by

$$R_n := \frac{X_{t_n}}{X_{t_{n-1}}} - 1 = \exp\left\{\left(r - \frac{1}{2}\sigma^2\right)(t_n - t_{n-1}) + \sigma(W_{t_n} - W_{t_{n-1}})\right\},$$

for any $n = 1, \dots, N$.

- The random variables $Y_n := R_n + 1$, $n = 1, \dots, N$, are independent and log-normally distributed.
- Since

$$R_n^* = F_\ell + (Y_n - (F_\ell + 1))^+ - (Y_n - (C_\ell + 1))^+,$$

we can get $\mathbb{E}[R_n^*]$ via Black-Scholes formula, for any $n = 1, \dots, N$.

- Moreover, since Y_n, $n = 1, \dots, N$, are independent, we can get

$$\mathbb{E}[R_g] = \mathbb{E}[R_1^*] + \dots \mathbb{E}[R_N^*].$$

- In

  `montecarlovariancereduction.controlvariates`

  you can find the code for the application of Control variates in the case of Cliquet option under the Black-Scholes model.
- In particular, in

  `montecarlovariancereduction.controlvariates.cliquetOptionTest`

  we compare the classical Monte-Carlo approach, Monte-Carlo with Antithetic variables and Monte-Carlo with control variates on two aspects, for 30 tests with 10^4 simulations:
  - variance of the estimates
  - time (in seconds) needed for a single estimate.
- The results are shown in the following table.

	classical MC	MC with AV	MC with CV
variance	$3.94 \cdot 10^{-6}$	$1.32 \cdot 10^{-6}$	$4.79 \cdot 10^{-7}$
time	0.21	0.23	0.48

- You can see that Control variates effectively reduce the variance. However, as it is now, it is slower. Exercise: change the implementation (also of the class `CliquetOption` if needed) in order to make the Control variates application faster without loosing accuracy.

Motivation

The multi-period Binomial model for option pricing is widely used by practitioners in financial applications mainly because:

- It is very easy to understand and simulate.
- It is particularly convenient to price options involving a choice of the holder, like American and Bermudan options.
- It approximates the Black-Scholes model when the length of the periods tends to zero.
- Option pricing is not based on pure Monte-Carlo techniques but relies on weighting the payoff relative to any scenario by the (analytic!) probability of the scenario.

The setting

- Consider a multi-period model with times $t = 0, 1, \dots, T$, and consider a probability space $(\Omega, \mathcal{F}, \mathbb{F}, P)$, where $\mathbb{F} = (\mathcal{F}_t)_{t=0,\dots,T}$ is a filtration representing information.
- Suppose there exist:
  - A risk free asset defined by $S_t^0 = (1+\rho)^t$, $t = 0, \dots, T$, with a deterministic interest rate $\rho > 0$.
  - A risky asset adapted to $\mathbb{F}$ defined by

    $$S_t = S_0 \cdot Y_1 \cdot \dots \cdot Y_t, \quad t = 1, \dots, T,$$

    where Y_t can take the two values d, u with $0 < d < 1 + \rho < u$, for any $t = 1, \dots, T$, and $(Y_t)_{t=1,\dots,T}$ are i.i.d. and such that Y_{t+1} is independent of $\mathcal{F}_t$.
- It then holds

$$S_t^0 = S_{t-1}^0(1+\rho), \quad t = 1, \dots, T$$

and

$$S_t = S_{t-1}Y_t, \quad t = 1, \dots, T.$$

Admissible strategies

At every time $t = 0, \dots, T-1$, an investor can construct a portfolio of value V_t, trading on the risk-free asset S^0 and on the risky asset S.

- The value of the portfolio is given by

$$V_t = \alpha_t S_t + \beta_t S_t^0, \quad t = 1, \dots, T,$$

  where $(\alpha_t)_{t=1,\dots,T}$ and $(\beta_t)_{t=1,\dots,T}$ are $\mathbb{F}$-predictable, discrete process.
- The strategy (α, β) must be self-financing: it must hold

$$V_t = \alpha_t S_t + \beta_t S_t^0 = \alpha_{t-1} S_t + \beta_{t-1} S_t^0, \quad t = 1, \dots, \quad T.$$

Definition

A portfolio V is an arbitrage if:

- V is obtained by a self-financing and predictable strategy;
- $P(V_0 = 0) = 1$;
- $P(V_t \geq 0) = 1$ and $P(V_t > 0) > 0$ for some t.

Proposition

The market is arbitrage free only if $d < 1 + \rho < u$.

- Suppose $1 + \rho \leq d < u$, and consider the self-financing portfolio defined by

$$V_t = S_t - \frac{S_0}{S_0^0} S_t^0, \quad t = 0, 1, \dots, T.$$

  Then we have $V_0 = 0$ and

$$V_1 = S_1 - \frac{S_0}{S_0^0} S_1^0 \geq S_0 d - S_0(1 + \rho) \geq 0.$$

- If $d < u \leq 1 + \rho$, changing the signs to the strategy above leads to an arbitrage

In order for the market to be arbitrage-free and complete, there must exist a unique measure $Q \sim P$ such that $\frac{S}{S^0}$ is a martingale, i.e., such that

$$\mathbb{E}^Q\left[\frac{S_{t+1}}{S^0_{t+1}}\Big|\mathcal{F}_t\right] = \frac{S_t}{S^0_t}, \quad t = 0, \dots, T-1. \tag{2}$$

Note that the measure Q is identified by the probability $q := P(Y_t = u)$.
Since

$$\mathbb{E}^Q\left[\frac{S_{t+1}}{S^0_{t+1}}\Big|\mathcal{F}_t\right] = \frac{(qu + (1-q)d)S_t}{S^0_t(1+\rho)}, \quad t = 0, \dots, T-1,$$

equation (2) holds if and only if $qu + (1-q)d = 1 + \rho$, that is,

$$q = \frac{1+\rho-d}{u-d}.$$

Such Q exists and is unique as we have supposed $0 < d < 1 + \rho < u$, and

$$\frac{dQ}{dP}(\omega) = \left(\frac{q}{p}\right)^{n(\omega)}\left(\frac{1-q}{1-p}\right)^{T-n(\omega)},$$

where $p := P(Y_t = u)$ and $n(\omega)$ is the number of times $t = 1, \dots, T$ when $Y_t(\omega) = u$.

Replicating strategy

- Assume we want to find an admissible strategy (α_t, β_t), $t = 0, \dots, T$, such that the value of the portfolio

$$\alpha_t S_t + \beta_t (1+\rho)^t$$

  equals the value V_t of an option at every time $t = 0, \dots, T$.
- From now on, fix $t = 1, \dots, T$, and suppose we know S_{t-1}.
- Call V_t^u the value of the option at time t when $Y_t = u$ and V_t^d the value of the option at time t when $Y_t = d$.
- It must hold

$$\begin{cases} \alpha_t u S_{t-1} + \beta_t (1+\rho)^t = V_t^u, \\ \alpha_t d S_{t-1} + \beta_t (1+\rho)^t = V_t^d. \end{cases}$$

- The solution to the system above is

$$\alpha_t = \frac{V_t^u - V_t^d}{S_{t-1}(u-d)},$$
$$\beta_t = \frac{u V_t^d - d V_t^u}{(1+\rho)^t (u-d)}.$$

  and gives the right replicating strategy.

Option pricing from admissibility

- Remember that our strategy (α_t, β_t), $t = 0, \dots, T$, has to be admissible!
- This means that we must have that

$$
\begin{aligned}
V_{t-1} &= \alpha_{t-1} S_{t-1} + \beta_{t-1}(1+\rho)^{t-1} \\
&= \alpha_t S_{t-1} + \beta_t (1+\rho)^{t-1} \\
&= \frac{V_t^u - V_t^d}{u-d} + \frac{uV_t^d - dV_t^u}{(1+\rho)(u-d)} \\
&= \frac{(1+\rho)(V_t^u - V_t^d) + uV_t^d - dV_t^u}{(1+\rho)(u-d)} \\
&= \frac{(1+\rho-d)V_t^u + (u-1-\rho)V_t^d}{(1+\rho)(u-d)} \\
&= \frac{qV_t^u + (1-q)V_t^d}{1+\rho} \\
&= \frac{1}{1+\rho}\mathbb{E}^Q[V_t|\mathcal{F}_{t-1}].
\end{aligned}
$$

- Then we have that the value $(V_t)_{t=0,\dots,T}$ of the option is a martingale under Q.
- This gives us a pricing theorem.

Theorem

The value V_0 of a contingent claim with maturity T and payoff V_T depending on the realizations of S until time T, is given by

$$V_0 = \frac{1}{(1+\rho)^T}\mathbb{E}^Q[V_T].$$

Remark

Because of the theorem above, we always simulate our Binomial model under the risk neutral measure Q.

- Our main goal here is to get the price of European and (most importantly) American options written on an underlying Binomial model.
- This valuation will approximate the price of the options written on an underlying log-normal model.
- We then simulate the realizations of the underlying model in Python, and get the payoff on the realizations, along with its expectation.
- Remember we have to price under the risk neutral measure Q: then we simulate the realizations of the process under Q.
- The most naive way we can imagine to do this is a brute force Monte-Carlo approximation..

Monte-Carlo method

- Imagine we want to valuate the discounted price of an European option with a given payoff function $f : \mathbb{R} \to \mathbb{R}$, written on the process S, with maturity T.
- Suppose we don't know any analytic formula in order to derive the price as

$$V_0 = \frac{1}{(1+\rho)^T} \mathbb{E}^Q[f(S_T)].$$

- We consider N *states of the world* $\omega_1, \omega_2, \dots, \omega_N \in \Omega$.
- To any $\omega_1, \omega_2, \dots, \omega_N$, we associate a given trajectory of the process $(S_t)_{t=0,\dots,T}$, with dynamics given under the measure Q.
- In particular, we suppose that the trajectories $(S_t(\omega_k))_{t=0,\dots,T}$, $k = 1, 2, \dots, N$ are independent of each other.
- Strong law of large numbers:

$$\frac{1}{n}\sum_{k=1}^{n} f(S_T(\omega_k)) \to \mathbb{E}^Q[f(S_T)] \quad \text{a.s., when } n \to \infty.$$

- The idea is to simulate such trajectories and approximate

$$\mathbb{E}^Q[f(S_T)] \approx \frac{1}{N}\sum_{k=1}^{N} f(S_T(\omega_k)).$$

Monte-Carlo method for the Binomial model

- Our first goal is then to generate a sequence of random numbers in order to simulate N independent trajectories $(S_t(\omega_k))_{t=0,\dots,T}$, $k = 1, 2, \dots, N$ of S under the risk neutral measure Q, and store them in a $(T+1) \times N$ matrix.
- First (not that) bad news: it is not possible to generate a sequence of perfectly random numbers, the best we can get is a sequence of *pseudo*-random numbers.
- Idea: generate (with the help of Python in our case) a sequence of $T \cdot N$ uniformly distributed, pseudo-random numbers $0 < x_{i,j} < 1$, $i = 1, \dots, T$, $j = 1, \dots, N$.
- Fix $\rho > 0$, $u > 1 + \rho$, $d < 1$, $q = \frac{1+\rho-d}{u-d}$.
- For every $i = 1, \dots, T$, $j = 1, \dots, N$, define

$$Y_i(\omega_j) = \begin{cases} u & \text{if } x_{i,j} < q \\ d & \text{if } x_{i,j} \geq q \end{cases}$$

and

$$S_{i+1}(\omega_j) = Y_i(\omega_j) S_i(\omega_j).$$

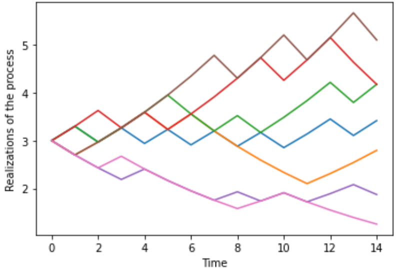
- You can find the code relative to the simulation of the Binomial model with the pure Monte-Carlo approach described above in

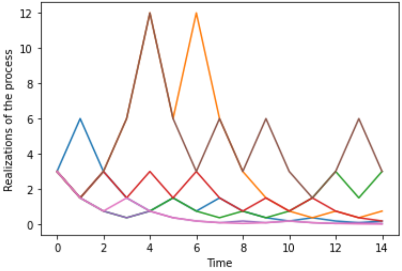
  `binomialmodel.creationandcalibration.binomialModelMonteCarlo`
- Note that the class you find there extends the one in

  `binomialmodel.creationandcalibration.binomialModel`.
- This is done in order to implement in the parent class some methods that do not strictly depend on the way in which we simulate the process.
- In this way, we don't have to copy and paste these methods in every class where we simulate the model in some way: object oriented programming feature.

Some paths

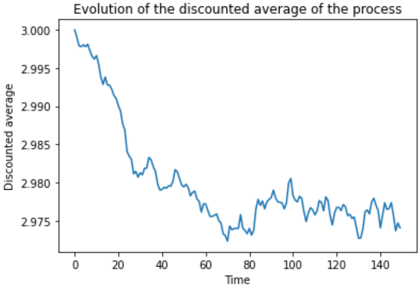
- We plot below some paths of the Binomial model.
- In the figure at the left we take $S_0 = 3$, $u = 1.1$, $d = 0.9$, $r = 0.05$, $T = 150$, having then $q = \frac{1+\rho-d}{u-d} = 0.75$.
- On the right, $S_0 = 3$, $u = 2$, $d = 0.5$, $r = 0.1$, $T = 150$, $q = \frac{1+\rho-d}{u-d} = 0.4$.

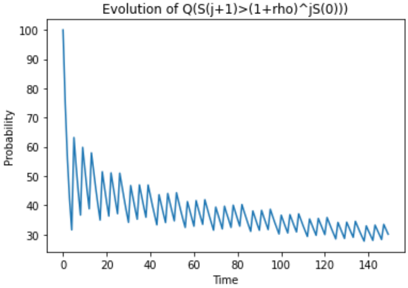




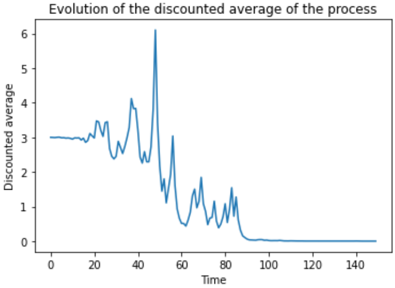
A first test

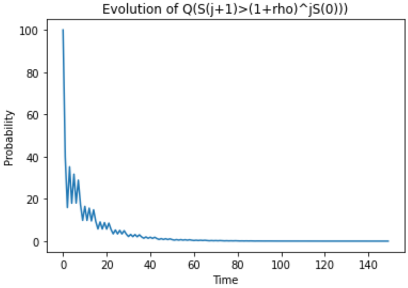
We show here the evolution of the discounted average of the process and of the probability $Q(S_{t_j} > (1+\rho)^{t_j} S_0)$, computed by using the Monte-Carlo method with 10^5 simulations, for $S_0 = 3$, $u = 1.1$, $d = 0.9$, $r = 0.05$, $T = 150$. In this case, we have $q = \frac{1+\rho-d}{u-d} = 0.75$.





Look at the evolution of the same quantities, again computed by using the Monte-Carlo method, choosing now $S_0 = 3$, $u = 2$, $d = 0.5$, $r = 0.1$, $T = 150$, $q = \frac{1+\rho-d}{u-d} = 0.4$.

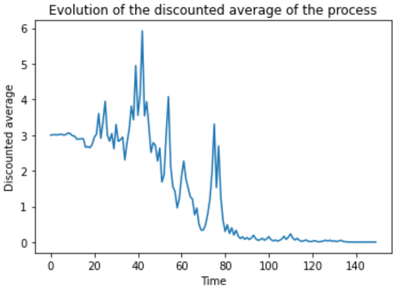


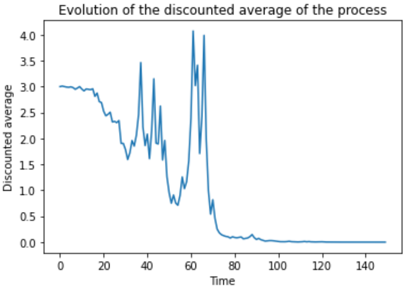


Why is the estimate of the average that inaccurate?

- The anaytic average of the discounted process is equal to S_0, due to many realizations such that $S_{t_j} < (1+\rho)^{t_j} S_0$ and few, extremely high realizations.
- If you buy S at time $t = 0$, and you hold it for 150 time steps, you make a positive gain with a very low probability, but the gain can be extremely high.
- Problem: The approximated average is strongly impacted by whether or not those paths leading to high gains are simulated or not.

Let's choose two different seeds, for the same parameters





Maybe a pure Monte-Carlo approach is not the best solution..

- We have seen that, if the volatility is high, the Monte-Carlo approach can be very inaccurate for many time steps.
- Moreover, it is time consuming (this is a problem common to all brute-force Monte-Carlo approaches)
- Idea: let us exploit some analytic properties of the Binomial model..

Some simple observations

- At the n-th time step, $n+1$ realizations of the process are possible: $S_0u^n, S_0u^{n-1}d, \dots, S_0ud^{n-1}, S_0d^n$.
- The number of ups and downs is given by a Bernoulli distribution:

$$P(S_n = S_0u^kd^{n-k}) = \binom{n}{k} q^k(1-p)^{n-k}.$$

- Using the expression above, we can compute

$$\begin{aligned}\mathbb{E}^Q[f(S_n)] &= \sum_{k=0}^{n} Q(S_n = S_0u^kd^{n-k})f(S_0u^kd^{n-k}) \\ &= \sum_{k=0}^{n} \binom{n}{k} q^k(1-q)^{n-k} f(S_0u^kd^{n-k}).\end{aligned}$$

Implementation in Python

- The idea is then to generate all the possible realizations of the process up to a given time, and to weight them by their probability.
- You can find the code relative to the this approach in

  `binomialmodel.creationandcalibration.binomialModelSmart`,

  whose class also extends the one in

  `binomialmodel.creationandcalibration.binomialModel`.
- Doing some tests in

  `binomialmodel.creationandcalibration.binomialModelSmartTest`.

  you can observe that, in this way, the average of the discounted process is stable.
- Moreover, this approach is of course much faster.

Computation of $Q(S_n > S_0(1+\rho)^n)$, $n = 1, \dots, T$

- Note that for any $k = 0, \dots, n$ it holds

$$S_n = S_0 u^k d^{n-k} > S_0(1+\rho)^n \iff u^k d^{n-k} > (1+\rho)^n$$
$$\iff \left(\frac{u}{d}\right)^k > \left(\frac{1+\rho}{d}\right)^n$$
$$\iff k > n \log_{\frac{u}{d}} \left(\frac{1+\rho}{d}\right).$$

- Then we have

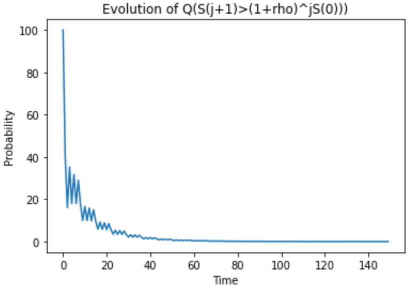
$$Q\left(S_n > S_0(1+\rho)^n\right) = \sum_{k=\bar{k}}^{n} Q(S_n = S_0 u^k d^{n-k})$$
$$= \sum_{k=\bar{k}}^{n} \binom{n}{k} q^k (1-q)^{n-k} f(S_0 u^k d^{n-k}),$$

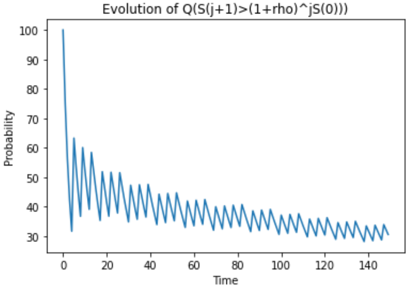
where

$$\bar{k} = \min\left\{k \in \mathbb{N} : \; k > n \log_{\frac{u}{d}} \left(\frac{1+\rho}{d}\right)\right\} \leq n.$$

Evolution of the probability plotted with Python

We show here the evolution of the probability computed above, over 150 time steps. On the left, we have parameters $S_0 = 3$, $u = 1.1$, $d = 0.9$, $\rho = 0.1$, $q = \frac{1+\rho-d}{u-d} = 0.75$. On the right, $S_0 = 3$, $u = 2$, $d = 0.5$, $\rho = 0.05$, $q = \frac{1+\rho-d}{u-d} = 0.4$.





- As seen before, an application of the simulation of the Binomial model in this way is the valuation of European options, under the pricing measure Q.
- In

  `binomialmodel.optionValuation.europeanOption,`

  you can see some methods relative to this.
- In particular, we compute the expectation of the payoff of European options as

$$\mathbb{E}^Q[f(S_n)] = \sum_{k=0}^{n} Q(S_n = S_0 u^k d^{n-k}) f(S_0 u^k d^{n-k})$$
$$= \sum_{k=0}^{n} \binom{n}{k} q^k (1-q)^{n-k} f(S_0 u^k d^{n-k}).$$

- We also compute the value of a general option for every time $t = 0, \dots, T-1$, and the corresponding self-financing, replicating strategy (α_t, β_T), $t = 0, \dots, T-1$, described before.
- As an exercise, you can check if the final value of the portfolio given by that strategy equals the payoff, for an option of your choice.

- Recall that we have

$$S_t = S_0 \cdot Y_1 \cdot \dots \cdot Y_t, \quad t = 1, \dots, T,$$

where

$$Y_t = \begin{cases} u > 1+\rho \text{ with (risk-neutral) probability } q = \frac{1+\rho-d}{u-d} \\ d < 1 \text{ with probability } 1-q \end{cases}, \quad t = 1, \dots, T.$$

- Our goal is to calibrate the up and downs parameters u and d, supposing we know the risk neutral probability $q = \frac{1+\rho-d}{u-d}$ and the interest rate $\rho > 0$, and that we can observe

$$\text{Var}[\log(S_T/S_0)] := \mathbb{E}^Q[\log(S_T/S_0)^2] - \mathbb{E}^Q[\log(S_T/S_0)]^2$$

for a given maturity T.
- Observe first that since

$$\log(S_T/S_0) = \sum_{t=1}^{T} \log(Y_t),$$

and since $(Y_t)_{t=1,\dots,T}$ are equi-distributed, we get

$$\text{Var}[\log(S_T/S_0)] = T^2\text{Var}[\log(Y_T)].$$

One result about variance

Proposition

Let

$$Y_t = \begin{cases} u > 1+\rho \text{ with (risk-neutral) probability } q = \frac{1+\rho-d}{u-d} \\ d < 1 \text{ with probability } 1-q, \end{cases} \quad , \quad t = 1, \dots, T.$$

Then for any $t = 1, \dots, T$ we have

$$\text{Var}[\log Y_t] = q(1-q)\log(u/d)^2.$$

Proof

$$\begin{aligned}
\text{Var}[\log Y_t] &= \mathbb{E}^Q[\log(Y_t)^2] - \mathbb{E}^Q[\log(Y_t)]^2 \\
&= \mathbb{E}^Q[\log(Y_t)^2] - (q\log(u) + (1-q)\log(d))^2 \\
&= q\log(u)^2 + (1-q)\log(d)^2 \\
&\quad - q^2\log(u)^2 - (1-q)^2\log(d)^2 - 2q(1-q)\log(u)\log(d) \\
&= q(1-q)\log(u)^2 + q(1-q)\log(d)^2 - 2q(1-q)\log(u)\log(d) \\
&= q(1-q)\left(\log(u) - \log(d)\right)^2 \\
&= q(1-q)\log(u/d)^2. \qquad (3)
\end{aligned}$$

- Thanks to
$$q = \frac{1+\rho-d}{u-d}$$
and to
$$\text{Var}[\log Y_t] = q(1-q)\log(u/d)^2,$$
along with
$$\sigma^2_{obs} := \text{Var}[\log(S_T/S_0)] = T^2\text{Var}[\log(Y_T)],$$
we can get u and d from q, ρ and σ^2_{obs} by solving the nonlinear nonsystem
$$\begin{cases} \frac{1+\rho-d}{u-d} = q \\ \log(u/d)^2 = \frac{\sigma^2_{obs}}{q(1-q)} \end{cases} \tag{4}$$

- We can find an approximated solution of (4) by the `fsolve` function of Python.
- Look at

`binomialmodel.creationandcalibration.binomialModelCalibration`

to see an implementation of the calibration of u and d as showed above.

- The holder of an American option with payoff f and maturity T on an underlying X has the right, at any time we $t \in [0, T]$, to hold the contract or to exercise the payoff $f(X_t)$.
- The valuation of American options is more complicated than the one of European options, since it involves an optimal exercise problem.
- In order to valuate such an option at time t, indeed, the conditional expectation of the future value of the option at time t has to be computed, and then compared against the present value of the payoff.
- However, the Monte-Carlo computation of a conditional expectation is very time consuming.
- One of the strengths of the Binomial model with respect to other settings is that it permits a favourable pricing of American options.
- Also when dealing with continuous time processes, with suitable dynamics, one may approximate them with a Binomial model in order to get the price.

- At any time $t = 1, \dots, T$, call $S_t(k)$ and $V_t(k)$ the value of the underlying and of the option, respectively, in the scenario with k ups and $t-k$ downs up to time t.
- Idea: proceed backward.
- First we compute the payoff $f(S_T(k)) = f(S_0 u^k d^{T-k})$, for any $k = 0, \dots, T$.
- We have of course $V_T(k) = f(S_T(k))$, for any $k = 0, \dots, T$.
- At time $T-1$, for any $k = 0, \dots, T-1$ we compute

$$
\begin{aligned}
V_{T-1}(k) &= \max\left(f(S_{T-1}(k)), \frac{1}{1+\rho}\left(qV_T(k+1) + (1-q)V_T(k)\right)\right) \\
&= \max\left(f(S_0 u^k d^{T-1-k}), \frac{1}{1+\rho}\left(qV_T(k+1) + (1-q)V_T(k)\right)\right).
\end{aligned}
$$

- For any $t = 1, \dots, T-2$ we compute with the same argument

$$
V_t(k) = \max\left(f(S_0 u^k d^{t-k}), \frac{1}{1+\rho}\left(qV_t(k+1) + (1-q)V_t(k)\right)\right).
$$

- We finally get the value of the option at initial time as

$$
V_0 = \max\left(f(S_0), \frac{1}{1+\rho}\left(qV_1(1) + (1-q)V_1(0)\right)\right).
$$

You can find the code relative to the the valuation of American options in

`binomialmodel.optionValuation.AmericanOption`,

with some tests in

`binomialmodel.optionValuation.AmericanOptionTest`,

Example

We consider a put option with payoff $f(x) = (20 - x)^+$, and choose parameters $T = 3$, $S_0 = 20$, $u = 1.1$, $d = 0.9$, $\rho = 0.05$.
The triangular matrices below show us an analysis of an American option for the such parameters (row 3 shows the values for $t = 3$ and so on).
The upper left and upper right matrices show the amount one would get if exercising the option or holding the contract, respectively; the lower left one the values of the option; the lower right one has 1 in the exercise region and 0 in the hold region

	0	1	2	3
0	0	nan	nan	nan
1	0	2	nan	nan
2	0	0.2	3.8	nan
3	0	0	2.18	5.42

	0	1	2	3
0	0.564464	nan	nan	nan
1	0.123583	1.27551	nan	nan
2	0	0.519048	2.84762	nan
3	0	0	2.18	5.42

	0	1	2	3
0	0	nan	nan	nan
1	0	2	nan	nan
2	0	0.2	3.8	nan
3	0	0	2.18	5.42

	0	1	2	3
0	0	nan	nan	nan
1	0	1	nan	nan
2	0	0	1	nan
3	1	1	1	1

- Consider a continuous, adapted stochastic process $X = (X_t)_{t\geq 0}$ with dynamics

$$dX_t = rX_t dt + \sigma X_t dW_t, \quad t \geq 0,$$

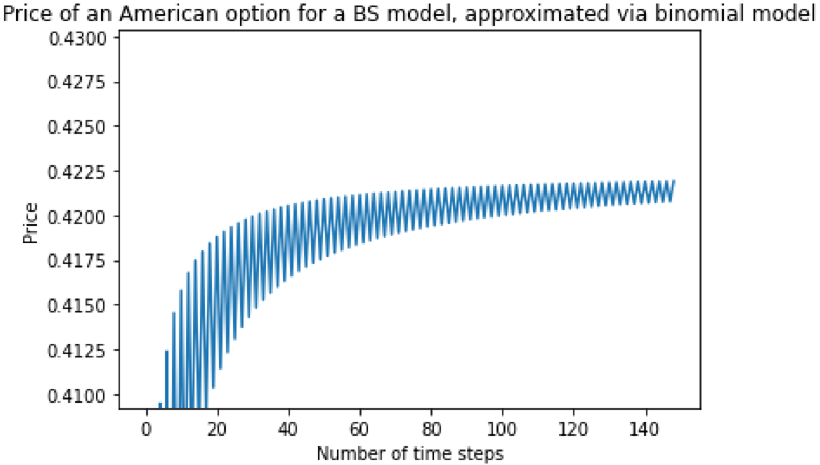
  where $r, \sigma > 0$ and $W = (W_t)_{t\geq 0}$ is a Brownian motion.
- Suppose you want to price an American option with underlying X and maturity $T > 0$.
- It can be seen that the dynamics of $X = (X_t)_{0\leq t\leq T}$ can be approximated by N time steps of a Binomial model with parameters

$$u = e^{\sigma\sqrt{T/N}}, \qquad d = 1/d, \qquad \rho = e^{r\sqrt{T/N}}, \tag{5}$$

  for N large enough, see for example .
- The idea is to approximate the price of the American option of maturity T with the price of an American option with maturity N written a Binomial model with parameters as in (5), for N large enough.
- Indeed, the price of the American option written on the Binomial model can be found as illustrated before.

Example: not such a nice behaviour

We consider an American put option with payoff $f(x) = (1-x)^+$ and maturity $T = 3$, written on a Black-Scholes model with parameters $r = 0.02$, $\sigma = 0.7$.
The plot below shows the approximated price via the derivation under the Binomial model, for an increasing number of times steps up to $N = 150$.



- First idea: we know the analytic price of an European put (or call) option under the Black-Scholes model. For example, call P^E the Black-Scholes formula price of an European put option.
- Also call:
  - P^E_N the price of an European put approximated by the Binomial model with N time steps;
  - P^A the analytic price of an American put;
  - P^A_N the price of an American put approximated by the Binomial model with N time steps.
- Second idea: we know the euristics $P^A - P^A_N \approx P^E - P^E_N$.
- We then approximate

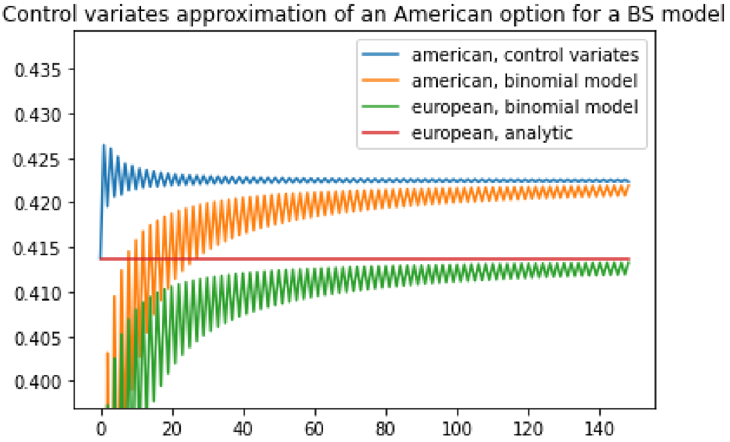
$$P^A \approx P^A_N + (P^E - P^E_N)$$

- This approximates the price of an American put option via control variates.
- Same thing for a call option.

A nicer behaviour with control variates

We consider again an American put option with payoff $f(x) = (1-x)^+$ and maturity $T = 3$, written on a Black-Scholes model with parameters $r = 0.02$, $\sigma = 0.7$: same situation as before

The plot below compares the prices introduced in the previous slide, for an increasing number of times steps up to $N = 150$.



- You can find some experiments relative to the stability of approximations of prices of American options with the Binomial model in

  `binomialmodel.optionValuation.AmericanOptionPriceConvergence`,

- The code performing the control variates approach can be found in

  `binomialmodel.optionValuation.controlVariates`,

  with some tests in

  `binomialmodel.optionValuation.controlVariatesTest`.