Parisian Option Pricing

Constantin Gleyze and Elias Mbarek

Master El Karoui

École Polytechnique and Sorbonne University

Abstract
In this document, we focus on Parisian options. Parisian options are barrier options that are activated or deactivated - depending on the type of option - if the underlying asset remains continuously above or below the barrier for a sufficient duration. Subsequently, we test several methods for pricing these options: an approach using a naive Monte Carlo method, an approach employing the concept of Brownian bridge, and finally an approach via the Laplace transform as developed by C. Labart & J. Lelong in their paper "Pricing Parisian options using Laplace transform". Additionally, we employ a variance reduction technique using an adaptive method to enhance the convergence of our results.

May 8, 2025

[0]Céline Labart & Jérôme Lelong, Pricing Parisian options using Laplace transforms , Bankers Markets & Investors : an academic & professional review, 2009, 99, 24 p., ffhal-00776703.

Contents

1 Introduction

A *Parisian option* is a path-dependent derivative whose payoff is contingent on the underlying asset's historical path: the option delivers its payoff only if the asset price has stayed (*In* option) or not (*Out* option) continuously above (*Up* option) or below (*Down* option) a fixed barrier L for a specified duration D.

Applications range from credit-spread triggers in synthetic CDOs to callable step-up notes.

Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{Q})$ support a standard Brownian motion W. Under the risk-neutral measure $\mathbb{Q}$ the price $X_t = \log S_t$ solves

$$\mathrm{d}S_t = S_t\Big(r - \delta - \tfrac{\sigma^2}{2}\Big)\mathrm{d}t + S_t\sigma\,\mathrm{d}W_t, \qquad S_0 > 0, \tag{1}$$

with short rate r, dividend yield δ, and volatility $\sigma > 0$.

For a fixed barrier level L we define the *Parisian clock*

$$\tau_D \;:=\; \inf\Big\{\, t \geq 0 : \exists\, s \in [d, T] \text{ s.t. } S_u \gtrless L \;\; \forall\, u \in [s-d, s]\Big\}.$$

The central objects of the paper are:

$$V^{\text{Parisian In}} := e^{-rT}\mathbb{E}\big[f(S_T)\,\mathbf{1}_{\{\tau_D \leq T\}}\big], \qquad V^{\text{Parisian Out}} := e^{-rT}\mathbb{E}\big[f(S_T)\,\mathbf{1}_{\{\tau_D > T\}}\big].$$

More specifically, the authors consider eight types of Parisian options:

PDIC: Parisian Down-and-In Call, PDOC: Parisian Down-and-Out Call,
PDIP: Parisian Down-and-In Put, PDOP: Parisian Down-and-Out Put,
PUIC: Parisian Up-and-In Call, PUOC: Parisian Up-and-Out Call,
PUIP: Parisian Up-and-In Put, PUOP: Parisian Up-and-Out Put.

The authors use two main relationships to price Parisian options:

Parameters:

- $x = S_0$: initial underlying price
- T: time to maturity
- K: strike price
- L: barrier level
- D: delay to spend within the barrier region
- r: risk-free interest rate
- δ: continuous dividend yield

1. In–Out Parity Relations (*Proof*)

We denote by BSC and BSP the Black Sholes price of a call and a put.

$$\begin{aligned}
\mathrm{PDIC}\big(x,T;K,L,D;r,\delta\big) \;+\; \mathrm{PDOC}\big(x,T;K,L,D;r,\delta\big) &= \mathrm{BSC}\big(x,T;K;r,\delta\big),\\
\mathrm{PUIC}\big(x,T;K,L,D;r,\delta\big) \;+\; \mathrm{PUOC}\big(x,T;K,L,D;r,\delta\big) &= \mathrm{BSC}\big(x,T;K;r,\delta\big),\\
\mathrm{PDIP}\big(x,T;K,L,D;r,\delta\big) \;+\; \mathrm{PDOP}\big(x,T;K,L,D;r,\delta\big) &= \mathrm{BSP}\big(x,T;K;r,\delta\big),\\
\mathrm{PUIP}\big(x,T;K,L,D;r,\delta\big) \;+\; \mathrm{PUOP}\big(x,T;K,L,D;r,\delta\big) &= \mathrm{BSP}\big(x,T;K;r,\delta\big).
\end{aligned}$$

2. Inversion Relations (*Proof*)

$$\begin{aligned}
\mathrm{PDOP}\big(x,T;K,L,D;r,\delta\big) &= x\,K\ \mathrm{PUOC}\Big(\tfrac{1}{x},\ T;\ \tfrac{1}{K},\ \tfrac{1}{L},\ D;\ \delta,\ r\Big),\\
\mathrm{PUOP}\big(x,T;K,L,D;r,\delta\big) &= x\,K\ \mathrm{PDOC}\Big(\tfrac{1}{x},\ T;\ \tfrac{1}{K},\ \tfrac{1}{L},\ D;\ \delta,\ r\Big),\\
\mathrm{PUIP}\big(x,T;K,L,D;r,\delta\big) &= x\,K\ \mathrm{PDIC}\Big(\tfrac{1}{x},\ T;\ \tfrac{1}{K},\ \tfrac{1}{L},\ D;\ \delta,\ r\Big),\\
\mathrm{PDIP}\big(x,T;K,L,D;r,\delta\big) &= x\,K\ \mathrm{PUIC}\Big(\tfrac{1}{x},\ T;\ \tfrac{1}{K},\ \tfrac{1}{L},\ D;\ \delta,\ r\Big).
\end{aligned}$$

First, we analyze the academic context in which this paper is situated as well as the proposed method (2). Next, we describe the Naive Monte Carlo approach and the variant using the Brownian Bridge, together with the variance-reduction technique employed(3). Finally, we present and comment our results(4).

In the appendix, you will find the proofs of the stated theorems Appendix 1, the rationale for our implementation of tests in Python Appendix 2, as well as the graphical representation (dataframes and charts) of some of our results Appendix 3.

2 Paper's Approach

We begin by recalling the foundational Parisian-pricing techniques in subsection 2.1, then establish our notation and auxiliary processes in subsection 2.2. Building on this, subsection 2.3 uncovers the barrier-sign reduction, which paves the way for the Laplace transform of the Down–Out Put in subsection 2.4. subsection 2.5 then turns to the Parisian Up–In Call and subsection 2.6 explains the Laplace transform inversion method by providing an error-controlled numerical inversion. Finally subsection 2.7 concludes by commenting the practical implementation of our code.

2.1 Techniques for Parisian Option Pricing

Chesney, Jeanblanc-Picqué & Yor (1997) pioneered Parisian valuation via Brownian excursion theory, obtaining Laplace transforms for the hitting time of a Brownian motion below a level during a fixed excursion length. PDE approaches followed (Avellaneda & Wu 1999; Haber 1999) as well as binomial lattices (Costabile 2002). Baldi *et al.* (2000) introduced importance sampling grounded in large-deviation theory to tame Monte-Carlo variance. A decisive step was made by Labart & Lelong (2009) who unified the transform approach for all eight Parisian flavours and proved an explicit exponentially-convergent numerical inversion.

For European payoffs and log-normal dynamics, the Laplace method is regarded as the gold standard because of the following reasons:

- deterministic, controllable accuracy
- closed-form Greeks
- speeds three orders of magnitude faster than unbiased Monte-Carlo at 10^{-6} relative error

Monte-Carlo remains indispensable when (a) model dynamics depart from diffusions with known transforms, or (b) early exercise is embedded.

The main approach of the paper is to use the theory of Brownian meanders and its Laplace transform to get explicit formulas for the prices of some types of Parisian Options like PDIC and PUIC and then using some parity relations and using reduction of cases, to obtain the prices of all possible Parisian Options. After defining and setting up our mathematical framework, we will present the approach of the paper. In fact by expressing the Parisian option value as a convolution-type payoff, which we will do below, it becomes clear immediately why the Laplace transform is advantageous. By one of the fundamental properties of the Laplace transform convolutions are transformed into simple products.

2.2 Notation and auxiliary processes

Under the same assumptions as in the Introduction, we can model model the asset price process $S = (S_t)_{0\le t\le T}$ by

$$S_t \;=\; x\,\exp\!\Big(\big(r-\delta-\tfrac{1}{2}\sigma^2\big)\,t+\sigma W_t\Big),\quad t\in[0,T],$$

Proposition 1. *(Cameron-Martin-Girsanov Theorem) Set*

$$m \;=\; \frac{r-\delta-\frac{1}{2}\sigma^2}{\sigma},$$

and define a new measure $\mathbb{P}$ by

$$\frac{d\mathbb{Q}}{d\mathbb{P}}\Big|_{\mathcal{F}_T} \;=\; \exp\!\Big(m\,Z_T-\tfrac{1}{2}m^2T\Big),\quad Z_t \;=\; W_t+m\,t.$$

Then Z is a $\mathbb{P}$–Brownian motion and under $\mathbb{P}$,

$$S_t \;=\; x\,e^{\sigma Z_t},\quad t\in[0,T].$$

Remark 1. Since the shift mt is deterministic, the natural filtration of Z coincides with $\mathbb{F}$.

Definition 1 (Hitting and Parisian Stopping Times)**.** Let Z be a Brownian motion, $D > 0$, $L > 0$.

An excursion can be described in terms of the Brownian motion Z. For a given barrier L for the process S, we introduce the corresponding barrier b for Z defined by

$$b \;=\; \frac{1}{\sigma}\,\ln\!\left(\frac{L}{x}\right).$$

We therefore define:

$$T_b \;=\; \inf\{u > 0 : Z_u = b\}, \quad g_t^b \;=\; \sup\{u \le t : Z_u = b\}, \quad d_t^b \;=\; \inf\{u \ge t : Z_u = b\}.$$

We also define the *downward Parisian time* and *upward Parisian time* such that:

$$T_b^- \;=\; \inf\bigl\{t > 0 : (t - g_t^b)\,\mathbf{1}_{\{Z_t < b\}} \ge D\bigr\}, \quad T_b^+ \;=\; \inf\bigl\{t > 0 : (t - g_t^b)\,\mathbf{1}_{\{Z_t > b\}} \ge D\bigr\}.$$

Remark 2. One equivalently writes

$$T_b^- = \inf\{t > D : \forall s \in [t-D, t],\, Z_s < b\}, \quad g_t^b = g_{L,t}^S,\; d_t^b = d_{L,t}^S.$$

Definition 2 (Laplace Transform)**.** For $f : [0,\infty) \to \mathbb{R}$, its Laplace transform is

$$\widehat{f}(\lambda) \;=\; \int_0^\infty e^{-\lambda t} f(t)\,dt.$$

Proposition 2. *If f, g have Laplace transforms on (σ_f, ∞) and (σ_g, ∞), then*

$$(f * g)(t) = \int_0^t g(u)\,f(t-u)\,du \quad\Longrightarrow\quad \widehat{f * g}(\lambda) = \widehat{f}(\lambda)\,\widehat{g}(\lambda),$$

valid on $(\max\{\sigma_f, \sigma_g\}, \infty)$.

Definition 3 (θ, k, d)**.** For $\lambda > 0$, strike $K > 0$, initial $x > 0$, and barrier b, set

$$\theta = \sqrt{2\lambda}, \quad k = \frac{1}{\sigma}\ln\!\left(\tfrac{K}{x}\right), \quad d = \frac{b-k}{\sqrt{D}}.$$

Definition 4 (Star Notation)**.** For any function f, define

$$f^\star(t) = e^{\left(r + \frac{1}{2}m^2\right)t}\,f(t).$$

For completeness, rigor and ease-of-reading we provide again the definitions of Parisian Down Call options:

Definition 5 (Parisian Down-and-In Call)**.** Let T_b^- be the first time the process spends a continuous duration D below barrier L. The price of the Parisian Down-and-In call with payoff $(S_T - K)^+$ is

$$\mathrm{PDIC}(x, T; K, L; r, \delta) = e^{-\left(r + \frac{1}{2}m^2\right)T}\,\mathbb{E}^{\mathbb{P}}\bigl[1_{\{T_b^- < T\}}\,(xe^{\sigma Z_T} - K)^+\,e^{mZ_T}\bigr].$$

Equivalently,

$$\mathrm{PDIC}^\star(x, T; K, L; r, \delta) = \mathbb{E}^{\mathbb{P}}\bigl[1_{\{T_b^- < T\}}\,(xe^{\sigma Z_T} - K)^+\,e^{mZ_T}\bigr].$$

Definition 6 (Parisian Down-and-Out Call)**.** The price of the Parisian Down-and-Out call is

$$\mathrm{PDOC}(x, T; K, L; r, \delta) = e^{-\left(r + \frac{1}{2}m^2\right)T}\,\mathbb{E}^{\mathbb{P}}\bigl[1_{\{T_b^- > T\}}\,(xe^{\sigma Z_T} - K)^+\,e^{mZ_T}\bigr],$$

and in the "star" notation,

$$\mathrm{PDOC}^\star(x, T; K, L; r, \delta) = \mathbb{E}^{\mathbb{P}}\bigl[1_{\{T_b^- > T\}}\,(xe^{\sigma Z_T} - K)^+\,e^{mZ_T}\bigr].$$

2.3 Barrier sign reduction

We show that for $b > 0$ the Parisian Down–Out Call price with barrier $L = xe^{\sigma b}$ can be expressed in terms of the corresponding price with barrier at level 0 and equivalently for the Parisian Up-In Call price for $b < 0$. Using these two reductions and the In-Out parities and the Call-Put parities, we can price all of the eight Parisian Flavors.

Proposition 3. Barrier-Sign Reduction for PDOC (*Proof*)

Let

$$T_b \;=\; \inf\{t > 0 : Z_t = b\}, \quad \mu_b(\mathrm{d}u) = \mathbb{P}(T_b \in \mathrm{d}u), \quad D > 0.$$

Define the "zero-barrier" Parisian Down–Out Call

$$\mathrm{PDOC}_0^\star(T; K/L; r, \delta) = \mathbb{E}^{\mathbb{P}}\big[\mathbf{1}_{\{T_0^- \geq T\}}\,(e^{\sigma Z_T} - K/L)^+\,e^{mZ_T}\big].$$

Then for $b > 0$,

$$\mathrm{PDOC}^\star(x, T; K, L) \;=\; L\,e^{mb}\int_0^D \mathrm{PDOC}_0^\star(T - u;\,K/L)\,\mu_b(\mathrm{d}u).$$

Moreover, its Laplace transform in T is

$$\widehat{\mathrm{PDOC}}^\star(x, \lambda; K, L) = L\,e^{mb}\widehat{\mathrm{PDOC}_0}^\star(\lambda; K/L; r, \delta)\int_0^D e^{-\lambda u}\,\mu_b(\mathrm{d}u),$$

where

$$\int_0^D e^{-\lambda u}\,\mu_b(\mathrm{d}u) = e^{-\theta b}\,N\big(\theta\sqrt{D} - b/\sqrt{D}\big) + e^{\theta b}\,N\big(-\theta\sqrt{D} - b/\sqrt{D}\big),$$

with $\theta = \sqrt{2\lambda}$.

Sketch of Two-Step Conditioning.

1. *Condition on the first hitting time T_b* of level b: on $\{T_b \leq D\}$ use $\mathbf{1}_{\{T_b^+ \leq T\}} = \mathbf{1}_{\{T_b \leq D\}}\,\mathbf{1}_{\{T_b^+ - T_b \leq T - T_b\}}$.
2. *Apply the strong Markov property* at $t = T_b$: the post-T_b process $B_s = Z_{T_b+s} - b$ is a $\mathbb{P}$–Brownian motion, and $T_b^+ - T_b = T_0^+(B)$.

Refer to the proof in the appendix for the detail of the calculations and a formalization of the heuristics above. □

We state without proof, as it is similar, the analogous reduction for the Parisian Up–In Call.

Proposition 4. Barrier-Sign Reduction for PUIC

Let

$$T_b \;=\; \inf\{t > 0 : Z_t = b\}, \quad \mu_b(\mathrm{d}u) = \mathbb{P}(T_b \in \mathrm{d}u), \quad D > 0, \quad L = x\,e^{\sigma b}.$$

Define the "zero-barrier" Parisian Up–In Call by

$$\mathrm{PUIC}_0^\star(T; K/L) = \mathbb{E}^{\mathbb{P}}\big[\mathbf{1}_{\{T_0^+ \leq T\}}\,(e^{\sigma Z_T} - K/L)^+\,e^{mZ_T}\big].$$

Then for $b > 0$ the price of the Parisian Up–In Call satisfies

$$\mathrm{PUIC}^\star(x, T; K, L\,; r, \delta) \;=\; L\,e^{mb}\int_0^D \mathrm{PUIC}_0^\star\big(T - u;\,K/L\,\big)\,\mu_b(\mathrm{d}u).$$

Moreover, its Laplace transform in T is

$$\widehat{\mathrm{PUIC}}^\star(x, \lambda; K, L) = L\,e^{mb}\,\widehat{\mathrm{PUIC}_0}^\star(\lambda; K/L; r, \delta)\int_0^D e^{-\lambda u}\,\mu_b(\mathrm{d}u),$$

with the same expression for

$$\int_0^D e^{-\lambda u}\,\mu_b(\mathrm{d}u)$$

as above.

2.4 Laplace transform of the Down-Out Put

Theorem 5. Laplace Transform of PDIC (*Proof*)

For $\lambda > 0$, let $\theta = \sqrt{2\lambda}$, $b = \frac{1}{\sigma}\ln(L/x)$ and define the kernel

$$K_{\lambda,D}(a) = \int_0^{+\infty} v \exp\Big(-\tfrac{v^2}{2D} \;-\; \theta\,|a-v|\Big)\,\mathrm{d}v.$$

Then the Laplace transform of the Parisian Down–In Call satisfies

$$\widehat{\mathrm{PDIC}}^{\star}(x,\lambda;K,L) \;=\; \frac{e^{\theta b}}{D\,\theta\,\psi(\theta\sqrt{D})}\int_k^{\infty} e^{my}\,(xe^{\sigma y}-K)\,K_{\lambda,D}(b-y)\,\mathrm{d}y, \tag{2}$$

where $k = \frac{1}{\sigma}\ln(K/x)$ and $\psi(z) = e^{z^2/2}N(-z)$.

Moreover, for $\lambda > (m+\sigma)^2/2$ the above admits closed-form simplifications:

***Case** $K > L$:*

$$\widehat{\mathrm{PDIC}}^{\star}(x,\lambda) = \frac{\psi(-\theta\sqrt{D})\,e^{2\theta b}}{\theta\,\psi(\theta\sqrt{D})}\,K\,e^{(m-\theta)k}\Big(\tfrac{1}{m-\theta}-\tfrac{1}{m+\sigma-\theta}\Big). \tag{3}$$

***Case** $K \le L$:*

$$\begin{aligned}\widehat{\mathrm{PDIC}}^{\star}(x,\lambda) = \frac{e^{(m+\theta)b}}{\psi\big(\theta\sqrt{D}\big)}\Bigg(&\frac{2K}{m^2-\theta^2}\,\psi\big(m\sqrt{D}\big) - m\sqrt{2\pi D}\,e^{\frac{Dm^2}{2}}\,N\big(m\sqrt{D}+d\big)\Big]\\ &-\frac{2L}{(m+\sigma)^2-\theta^2}\Big[\psi\big((m+\sigma)\sqrt{D}\big)-(m+\sigma)\sqrt{2\pi D}\,e^{\frac{D(m+\sigma)^2}{2}}\,N\big((m+\sigma)\sqrt{D}+d\big)\Big]\Bigg)\\ &+\frac{K\,e^{(m+\theta)k}}{\theta\,\psi\big(\theta\sqrt{D}\big)}\Big(\frac{1}{m+\theta}-\frac{1}{m+\sigma+\theta}\Big)\Big[\psi\big(\theta\sqrt{D}\big)-\theta\sqrt{2\pi D}\,e^{\lambda D}\,N\big(\theta\sqrt{D}-d\big)\Big]\\ &+\frac{e^{\lambda D}\sqrt{2\pi D}}{\psi\big(\theta\sqrt{D}\big)}\,K\,e^{2\theta b}e^{(m-\theta)k}\,N\big(-d-\theta\sqrt{D}\big)\Big(\frac{1}{m+\sigma-\theta}-\frac{1}{m-\theta}\Big).\end{aligned} \tag{4}$$

where $d = (b-k)/\sqrt{D}$.

We also state the result for PUIC. The proof is analogous to the proof of Theorem 5 and we therefore omit it.

2.5 Parisian Up–In Call (PUIC)

Theorem 6. Laplace Transform of PUIC

Let

$$\theta = \sqrt{2\lambda},\quad b = \frac{1}{\sigma}\ln\frac{L}{x},\quad k = \frac{1}{\sigma}\ln\frac{K}{x},\quad d = \frac{b-k}{\sqrt{D}},$$

and define the kernel $K_{\lambda,D}(z)$ as above Then the Laplace transform of the Parisian Up–In Call is

$$\widehat{\mathrm{PUIC}}^{\star}(x,\lambda;K,L) \;=\; \frac{e^{-\theta b}}{D\,\theta\,\psi(\theta\sqrt{D})}\int_k^{\infty} e^{my}\,\big(xe^{\sigma y}-K\big)\,K_{\lambda,D}(y-b)\,\mathrm{d}y.$$

Moreover, for $\lambda > (m+\sigma)^2/2$ one obtains:

***Case** $K > L$:*

$$\widehat{\mathrm{PUIC}}^{\star}(x,\lambda) = \frac{2\, e^{(m-\theta)b}}{\sqrt{2\pi D}\,\psi(\theta\sqrt{D})}\Big[\frac{K}{m^2-\theta^2}\, e^{\frac{Dm^2}{2}}\, m\, N\big(m\sqrt{D}+d\big) - \frac{L}{(m+\sigma)^2-\theta^2}\, e^{\frac{D(m+\sigma)^2}{2}}\,(m+\sigma)\, N\big((m+\sigma)\sqrt{D}+d\big)\Big] + \frac{e^{-2\theta b}}{\psi(\theta\sqrt{D})}\, K\, e^{(m+\theta)k}\, e^{\lambda D}\sqrt{2\pi D}\, N\big(d-\theta\sqrt{D}\big)\left(\tfrac{1}{m+\sigma+\theta} - \tfrac{1}{m+\theta}\right) + \frac{e^{(m-\theta)k}}{\theta\,\psi(\theta\sqrt{D})}\, K\left(\tfrac{1}{m-\theta} - \tfrac{1}{m+\sigma-\theta}\right)\Big[\psi(\theta\sqrt{D}) \;-\; \theta\sqrt{2\pi D}\, e^{\lambda D}\, N\big(d+\theta\sqrt{D}\big)\Big]. \tag{5}$$

***Case** $K \le L$:*

$$\widehat{\mathrm{PUIC}}^{\star}(x,\lambda) = \frac{2\, e^{(m-\theta)b}}{\psi(\theta\sqrt{D})}\Big[\frac{K}{m^2-\theta^2}\,\psi\big(m\sqrt{D}\big) \;-\; \frac{L}{(m+\sigma)^2-\theta^2}\,\psi\big((m+\sigma)\sqrt{D}\big)\Big] + \frac{e^{-2\theta b}\,\psi(-\theta\sqrt{D})}{\theta\,\psi(\theta\sqrt{D})}\, K\, e^{(m+\theta)k}\Big(\tfrac{1}{m+\theta} - \tfrac{1}{m+\theta+\sigma}\Big). \tag{6}$$

2.6 Error-controlled numerical inversion

The main advantage of the Laplace pricing method of the paper is the fast convergence rate of the numerical inversion. In fact we can invert the Laplace transform using a series representation which we can accelerate considerably using the Euler summation technique.

Here are the two theorems behind the numerical inversion to recover the prices of the Parisian options:

Theorem 7 (Numerical Inversion via Euler-Accelerated Trapezoidal Rule)**.** *Let f be a continuous, bounded function satisfying $f(t) = 0$ for $t < 0$, and assume its Laplace transform $\hat{f}(\lambda)$ is analytic on $\{\lambda \in \mathbb{C} : \mathrm{Re}(\lambda) > \frac{(m+\sigma)^2}{2}\}$. Define the trapezoidal approximation with step size π/t as:*

$$f_{\pi/t}(t) = \frac{e^{\alpha t}}{2t}\hat{f}(\alpha) + \frac{e^{\alpha t}}{t}\sum_{k=1}^{\infty}(-1)^k\,\mathrm{Re}\left[\hat{f}\left(\alpha + i\frac{k\pi}{t}\right)\right],$$

where $\alpha > \frac{(m+\sigma)^2}{2}$. Then the discretization error satisfies:

$$|f(t) - f_{\pi/t}(t)| \le \|f\|_\infty \cdot \frac{e^{-2\alpha t}}{1-e^{-2\alpha t}}.$$

Furthermore, for a target relative error ϵ, it suffices to choose $2\alpha t = max\{\frac{(m+\sigma^2)}{2}, \frac{1}{2t}ln(\frac{1+\epsilon}{\epsilon})\}$.

Theorem 8 (Euler Acceleration and Global Error Bound)**.** *Let $f \in C^{q+4}$ and assume that for some $\varepsilon > 0$ and all $k \le q+4$, we have:*

$$f^{(k)}(s) = \mathcal{O}(e^{(\alpha-\varepsilon)s}).$$

Define the truncated series:

$$s_p(t) = \frac{e^{\alpha t}}{2t}\hat{f}(\alpha) + \frac{e^{\alpha t}}{t}\sum_{k=1}^{p}(-1)^k\,\mathrm{Re}\left[\hat{f}\left(\alpha + i\frac{\pi k}{t}\right)\right],$$

and the Euler-accelerated approximation:

$$E(q,p,t) = \sum_{k=0}^{q}\binom{q}{k}2^{-q}s_{p+k}(t).$$

Then the global error satisfies:

$$|f(t) - E(q,p,t)| \leq S_0 \cdot \frac{e^{-2\alpha t}}{1 - e^{-2\alpha t}} + \frac{e^{\alpha t} t |f'(0) - \alpha f(0)|}{\pi^2} \cdot \frac{1}{p!(q+1)!} \cdot \frac{1}{2^q (p+q+2)!} + \mathcal{O}\left(\frac{1}{p^{q+3}}\right).$$

For $2\alpha t = 18.4$ and $q = p = 15$, this gives:

$$|f(t) - E(15,15,t)| \leq S_0 \cdot 10^{-8} + t|f'(0) - \alpha f(0)| \cdot 10^{-11}.$$

2.7 Implementation of the Laplace Pricing Method

To implement Laplace Pricing, two functions, `laplace_P_D_in_call` and `laplace_P_U_in_call`, will play a key role, as explained above. Furthermore, it is important to implement the In-Out Parity and the Black-Scholes pricing as these relationships are used in the pricing of the Parisian Calls to jump from In-options to Out-options. The two pricing functions for the PDIC and the PUIC are implemented as such: If $b \leq 0$, respectively $b \geq 0$, the closed-form expressions of the Laplace transform of these call prices give the exact result. In the other case, we have to reduce the calls to PDOC, respectively, PUOC for $b = 0$ (after having used the In-Out parity). Thus we get the Laplace prices for calls. To price Parisian Puts, we use the same methodology as above, but we have to use the Call-Put parities, by calculating first the Call prices with inverted parameters. By doing this, one has to be careful to exchange r and δ. The discount factor also has to be adjusted this way.

For the inversion of Laplace prices, we use the Euler summation formula of the above theorem, which gives extremely fast and accurate pricing, where one can control the error level using the heuristic for α given in Theorem 7

3 Comparison of Pricing Methods

We begin in subsection 3.1 by setting up our path-simulation framework. We then turn in subsection 3.2 to the construction and analysis of the naive Monte Carlo estimator. In subsection 3.3 we introduce the Brownian-bridge correction to eliminate discretization bias. Finally, subsection 3.4 presents the adaptive antithetic scheme that further reduces variance.

3.1 Path Simulation

We first simulate several price trajectories. For N trajectories and M discretization steps , we simulate $N \times M$ random variables

$$\left(W_j^i\right)_{i\in\{1,\ldots,N\},\, j\in\{1,\ldots,M\}}$$

drawn i.i.d. from a standard normal distribution, i.e.

$$W_j^i \sim \mathcal{N}(0,1).$$

We then construct our trajectories using the following process:

$$S_{t_j}^i = \begin{cases} S_0, & j=0, \\ S_{t_{j-1}}^i \exp\Big(\big(r-\delta-\tfrac{1}{2}\sigma^2\big)\,(t_j-t_{j-1}) + \sigma\,\sqrt{t_j-t_{j-1}}\,W_j^i\Big), & j=1,\ldots,N. \end{cases}$$

3.2 Naive Monte Carlo

3.2.1 Theoretical approach

We use the notation X_k to denote the k-th simulated payoff of our option.

We assume that the sequence $(X_k)_{k=1}^n$ is i.i.d. $\forall n \geq 0$.

By the Central Limit Theorem,

$$\sqrt{n}\,\big(\overline{X}_n - m_X\big) \xrightarrow{\mathcal{L}} \mathcal{N}\big(0,\sigma_X^2\big) \quad \text{as } n \to +\infty,$$

where

$$\overline{X}_n = \frac{1}{n}\sum_{k=1}^{n} X_k, \quad m_X = \mathbb{E}[X_1], \quad \sigma_X^2 = \mathrm{Var}[X_1].$$

If we define the sampling error

$$\varepsilon_n \;=\; \overline{X}_n - m_X,$$

then it follows that

$$\varepsilon_n \xrightarrow{\mathcal{L}} \mathcal{N}\Big(0, \tfrac{\sigma_X^2}{n}\Big).$$

This method presents a high variance and a relatively slow convergence rate ($\mathcal{O}\big(n^{-1/2}\big)$).

3.2.2 Empirical application

Once we have our simulated trajectories, we introduce our condition steps $D = \left\lceil \frac{d}{\Delta t} \right\rceil$, the number of discretization step the excursion should be as well as our delay counter Θ defined as follows:

$$\Theta_0^i = 0, \qquad \Theta_{j+1}^i = \begin{cases} \Theta_j^i + 1, & \text{if } S_{t_j}^i \text{and } S_{t_{j+1}}^i \text{ satisfies the barrier condition (Up or Down)} \\ 0, & \text{otherwise.} \end{cases}$$

Remark 3. In practice, we break the counting loop for Θ as soon as it exceeds D.

We finally compute our prices using a naive Monte Carlo approach :

$$\widehat{V} = e^{-rT}\,\frac{1}{M}\sum_{i=1}^{M} f\big(S^i_{t_N}\big)\,\mathbf{1}_{\left\{\exists\, j\in\{1,\ldots,M\}:\,\Theta^i_j\geq D\right\}} \text{ for In Options.}$$

$$\widehat{V} = e^{-rT}\,\frac{1}{M}\sum_{i=1}^{M} f\big(S^i_{t_N}\big)\,\mathbf{1}_{\left\{\forall\, j\in\{1,\ldots,M\}:\,\Theta^i_j< D\right\}} \text{ for Out Options.}$$

We use these algorithms to compute PUIC and PDIC options. We obtain the other call options by using the In-Out Parity and the Put options using the inverse relations. To obtain the inverse price with $\frac{1}{S_0}$ and such that $r \to \delta$ and $\delta \to r$, we apply the following inversion to our simulated prices:

$$S_t^{\text{inv}} = \frac{S_t \cdot e^{-2(r-\delta)t}}{S_0^2}$$

Time discretisation ignores sub-interval excursions; the bias is $\mathcal{O}(\sqrt{\Delta t})$ (Broadie & Glasserman 1997).

3.3 Brownian Bridge Monte Carlo estimator

3.3.1 Theoretical approach

We define the centered Brownian bridge as

$$Y_t^{W,T} = W_t - \frac{t}{T}W_T$$

such that

$$\mathbb{E}[Y_t^{W,T}] = 0 \quad \text{and} \quad \operatorname{Cov}(Y_t^{W,T}, Y_s^{W,T}) = \min(s,t) - \frac{st}{T} \quad \text{for } s,t\in[0,T].$$

We recall the following key conditional independence result for the Brownian bridge.

Proposition 9 (Bridge of the Euler scheme)**.** [1]

Assume that $\sigma(t,x) = 0$ for every $t\in[0,T]$, $x\in\mathbb{R}$.

1. *The processes*
$$\big(\bar{X}^n_t\big)_{t\in[t_k,t_{k+1}]}, \quad k = 0,\ldots,n-1,$$
are conditionally independent given the filtrating field $\sigma\big(\bar{X}^n_{t_k} : k = 0,\ldots,n\big)$.
2. *Moreover, for each $k = 0,\ldots,n-1$, the conditional law*
$$\mathcal{L}\big((\bar{X}^n_t)_{t\in[t_k,t_{k+1}]} \mid \bar{X}^n_{t_k} = x_k,\ \bar{X}^n_{t_{k+1}} = x_{k+1}\big)$$
coincides with that of the Gaussian bridge
$$\left\{x_k + \frac{t-t_k}{t_{k+1}-t_k}(x_{k+1}-x_k) + \sigma\big(t_k,x_k\big)\,Y^{W,\,t_{k+1}-t_k}_{t-t_k}\right\}_{t\in[t_k,t_{k+1}]},$$
where $(Y_u^{W,L})_{u\in[0,L]}$ is a standard Brownian bridge of length $L = t_{k+1}-t_k$.

[1]See G. Pagès, *Numerical Probability*, Proposition 8.2.

We also have the following equation for all $\lambda \in \mathbb{R}_*$:

$$\mathbb{P}\left(\inf_{t\in[0,T]}\left(x+(y-x)\frac{t}{T}+\lambda Y_t^{W,T}\right)\geqslant z\right)=\begin{cases}1-\exp\left(-\dfrac{2(z-x)(z-y)}{\lambda^2 T}\right) & \text{if } z\leqslant \min(x,y),\\ 0 & \text{otherwise.}\end{cases}$$

In particular, for all $t_1, t_2 \in [0,T]$ such that $t_2 > t_1$, we have:

$$\mathbb{P}\left(\inf_{t\in[t_1,t_2]}\left(x+(y-x)\frac{t-t_1}{t_2-t_1}+\lambda Y_t^{W,t_2-t_1}\right)\geqslant z\right)=\begin{cases}1-\exp\left(-\dfrac{2(z-x)(z-y)}{\lambda^2 (t_2-t_1)}\right) & \text{if } z\leqslant \min(x,y),\\ 0 & \text{otherwise.}\end{cases}$$

Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{Q})$ support a standard Brownian motion W. Under the risk-neutral measure $\mathbb{Q}$ the log-price $X_t = \log S_t$ solves

$$\mathrm{d}X_t = \left(r-\delta-\tfrac{\sigma^2}{2}\right)\mathrm{d}t+\sigma\,\mathrm{d}W_t, \qquad X_0=\log(S_0),$$

with short rate r, dividend yield δ, and volatility $\sigma > 0$.

The dynamic of X allows for the use of the above Brownian bridge properties.

Combining the following properties of the logarithm function:

- The logarithm function is increasing on $]0,+\infty[$, so comparing S_t to L is equivalent to comparing $X_t = \log(S_t)$ to $\log(L)$.
- The logarithm function is injective from $]0,+\infty[$ to $]0,+\infty[$. Combined with the growth, the sup or max (and respectively the inf or min) of S_t and of X_t occur at the same times.
- The logarithm function is continuous on $]0,+\infty[$, which ensures that the sigma-fields are equivalent under the change of variables.

We can then derive:

$$\begin{aligned}
&\mathbb{P}\left(\inf_{t\in[t_k,t_{k+d}]}\bar{S}^n_t\geqslant L \,\middle|\, \bar{S}^n_{t_k},\bar{S}^n_{t_{k+d}}\right)\\
&=\mathbb{P}\left(\inf_{t\in[t_k,t_{k+d}]}\log(\bar{S}^n_t)\geqslant \log(L) \,\middle|\, \log(\bar{S}^n_{t_k}),\log(\bar{S}^n_{t_{k+d}})\right)\\
&=\mathbb{P}\left(\inf_{t\in[t_k,t_{k+d}]}\bar{X}^n_t\geqslant \log(L) \,\middle|\, \bar{X}^n_{t_k},\bar{X}^n_{t_{k+d}}\right)\\
&=\mathbb{P}\left(\inf_{t\in[t_k,t_{k+d}]}\left(\bar{X}^n_{t_k}+\frac{t-t_k}{t_{k+d}-t_k}\left(\bar{X}^n_{t_{k+d}}-\bar{X}^n_{t_k}\right)+\sigma\left(t_k,\bar{X}^n_{t_k}\right)Y^{W,t_{k+d}-t_k}_{t-t_k}\right)\geqslant\log(L)\right)\\
&=\begin{cases}1-\exp\left(-\dfrac{2(\log(L)-\bar{X}^n_{t_k})(\log(L)-\bar{X}^n_{t_{k+d}})}{\sigma^2(t_k,\bar{X}^n_{t_k})(t_{k+d}-t_k)}\right) & \text{if }\log(L)\leqslant\min\left(\bar{X}^n_{t_k},\bar{X}^n_{t_{k+d}}\right) <=> L\leqslant\min\left(\bar{S}^n_{t_k},\bar{S}^n_{t_{k+d}}\right),\\ 0 & \text{otherwise.}\end{cases}\\
&=\begin{cases}1-\exp\left(-\dfrac{2\log(\frac{L}{\bar{S}^n_{t_k}})\log(\frac{L}{\bar{S}^n_{t_{k+d}}})}{\sigma^2(t_k,\bar{X}^n_{t_k})(t_{k+d}-t_k)}\right) & \text{if } L\leqslant\min\left(\bar{S}^n_{t_k},\bar{S}^n_{t_{k+d}}\right),\\ 0 & \text{otherwise.}\end{cases}
\end{aligned}$$

We emphasize that the volatility used here is that of the log-price. In our modeling framework, while the volatility of $(S_t)_{t\in[0,T]}$ is $(\sigma S_t)_{t\in[0,T]}$, the volatility of the log-price $(X_t)_{t\in[0,T]}$ is constant and equal to σ.

We also note that in the case of the Brownian bridge, the distribution of the supremum is known (and can be derived from the equations above). This allows us to deduce the following identity:

$$\mathbb{P}\left(\sup_{t\in[t_k,t_{k+d}]} \bar{S}^n_t \leqslant L \,\middle|\, \bar{S}^n_{t_k}, \bar{S}^n_{t_{k+d}}\right) = \begin{cases} 1-\exp\left(-\dfrac{2\log(\frac{L}{S^n_{t_k}})\log(\frac{L}{S^n_{t_{k+d}}})}{\sigma^2(t_k,\bar{X}^n_{t_k})(t_{k+d}-t_k)}\right) & \text{if } L \geqslant \max\left(\bar{S}^n_{t_k}, \bar{S}^n_{t_{k+d}}\right), \\ 0 \text{ otherwise.} \end{cases}$$

3.3.2 Empirical application

We first simulate N trajectories with by discretizing the interval $[0,T]$ through M steps. We then obtain a (N,M) price matrix. We also simulate $(N-1)\times M$ random variables

$$\left(U^i_j\right)_{i\in\{1,\ldots,N-1\},\, j\in\{1,\ldots,M\}}$$

drawn i.i.d. from a standard continuous uniform distribution, i.e.

$$U^i_j \sim \mathcal{U}([0,1]).$$

Once we have our simulated trajectories, we introduce our condition steps $D = \left\lceil \frac{d}{\Delta t}\right\rceil$, the number of discretization step the excursion should be as well as our delay counter Θ defined as follows:

$$\Theta^i_0 = 0, \qquad \Theta^i_{j+1} = \begin{cases} \Theta^i_j + 1, & \text{if } S^i_{t_j} \text{and } S^i_{t_{j+1}} \text{ satisfies the barrier condition (Up or Down)} \\ & \text{and } U^i_j \leq 1-\exp\left(-\dfrac{2\log(\frac{L}{S^i_{t_j}})\log(\frac{L}{S^i_{t_{j+1}}})}{(t_{j+1}-t_j)\sigma^2}\right) \\ 0, & \text{otherwise.} \end{cases}$$

Remark 4. In practice, we break the counting loop for Θ as soon as it exceeds D.

We finally compute our prices using a naive Monte Carlo approach :

$$\widehat{V} = e^{-rT}\,\frac{1}{M}\sum_{i=1}^{M} f\left(S^i_{t_N}\right)\mathbf{1}_{\left\{\exists\, j\in\{1,\ldots,M\}:\, \Theta^i_j \geq D\right\}} \text{ for In Options.}$$

$$\widehat{V} = e^{-rT}\,\frac{1}{M}\sum_{i=1}^{M} f\left(S^i_{t_N}\right)\mathbf{1}_{\left\{\forall\, j\in\{1,\ldots,M\}:\, \Theta^i_j < D\right\}} \text{ for Out Options.}$$

We use these algorithms to compute PUIC, PDIC, PUIP and PDIP options. We obtain the other call and put options by using the In-Out Parity.

The bridge correction removes discretisation bias entirely, leaving only statistical error $\mathrm{Var}/\sqrt{N}$.

3.4 Variance Reduction - The adaptative anthitetic approach

3.4.1 Theoretical approach

We define X and X' that are identically distributed such that:

$$\mathbb{E}[X] = \mathbb{E}[X'] = m_X, \quad \mathrm{Var}[X] = \mathrm{Var}[X'] = \sigma_X^2,$$

but $\mathrm{Var}[X - X'] > 0$, and we set

$$\Xi \;=\; X - X'.$$

We introduce a dynamic control variate λ and we set:

$$X^\lambda \;=\; X \;-\; \lambda\,\Xi,$$

The variance of X^λ is minimized when

$$\lambda_{\min} = \frac{\mathrm{Cov}(X, \Xi)}{\mathrm{Var}(\Xi)}.$$

We compute the sample mean of the controlled payoffs, using an estimator $\hat{\lambda}_{min}$ for λ_{min}:

$$\overline{X}_n^{\hat{\lambda}_{min}} = \overline{X}_n + \hat{\lambda}_n^{min}\, \Xi_n,$$

where

$$\overline{X}_n = \frac{1}{n}\sum_{k=1}^n X_k, \qquad \hat{\lambda}_n^{min} = \frac{\frac{1}{n}\sum_{k=1}^n X_k\,(X_k - X_k')}{\frac{1}{n}\sum_{k=1}^n (X_k - X_k')^2}, \qquad \Xi_n = \frac{1}{n}\sum_{k=1}^n (X_k - X_k').$$

We have:

$$\sqrt{n}\left(\overline{X}_n^{\hat{\lambda}_{min}} - m_X\right) \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \sigma_{\min}^2\right) \quad \text{as } n \to +\infty,$$

or equivalently

$$\overline{X}_n^{\lambda} - m_X \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \tfrac{\sigma_{\min}^2}{n}\right) \quad \text{as } n \to +\infty,$$

with

$$\sigma_{\min}^2 \le \sigma_X^2.$$

However our estimator

$$\mathbb{E}\left[\overline{X}_n^{\hat{\lambda}_{min}}\right] = m_X - \mathbb{E}\left[\hat{\lambda}_n^{min}\, \Xi_n\right].$$

Indeed, in practice:

$$\mathbb{E}\left[\hat{\lambda}_n^{min}\, \Xi_n\right] \neq 0,$$

To manage this issue, we use the adaptative approach. We define for each k:

$$\widetilde{X}_k = X_k - \widetilde{\lambda}_{k-1}\, \Xi_k,$$

where

$$\widetilde{\lambda}_k = (-k) \vee \left(\hat{\lambda}_n^{min} \wedge k\right),$$

Our estimator is then:

$$\overline{\widetilde{X}}_n = \frac{1}{n}\sum_{k=1}^n \widetilde{X}_k = \overline{X}_n - \frac{1}{n}\sum_{k=1}^n \widetilde{\lambda}_{k-1}\, \Xi_k,$$

This estimator satisfies: [2]

- **Unbiasedness:** $\mathbb{E}\left[\overline{\widetilde{X}}_n\right] = m_X$.
- **Almost sure convergence:** $\overline{\widetilde{X}}_n \xrightarrow{\text{a.s.}} m_X \quad \text{as } n \to +\infty$.
- **Asymptotic normality:** $\sqrt{n}\left(\overline{\widetilde{X}}_n - m_X\right) \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \sigma_{\min}^2\right) \quad \text{as } n \to +\infty$.

[2]See G. Pagès, *Numerical Probability*, §3.2.2.

Then the adaptive estimator of m, defined by

$$\widetilde{X}_M^\lambda = \frac{1}{M}\sum_{k=1}^{M}\widetilde{X}_k,$$

is unbiased

$$\mathbb{E}\big[\widetilde{X}_M^\lambda\big] = m,$$

convergent, i.e.

$$\widetilde{X}_M^\lambda \xrightarrow{\text{a.s.}} m \quad \text{as } M \to +\infty,$$

and asymptotically normal with minimal variance, i.e.

$$\sqrt{M}\,\big(\widetilde{X}_M^\lambda - m\big) \xrightarrow{\mathcal{L}} \mathcal{N}\big(0, \sigma^2_{\min}\big) \quad \text{as } M \to +\infty.$$

3.4.2 Empirical application

We perform only one simulation of $N \times M$ standard normal draws, corresponding to N paths each discretized in M steps.

First, generate

$$(W_j^i)_{i=1,\dots,N,\ j=1,\dots,M} \quad \text{drawn i.i.d with} \quad W_j^i \sim \mathcal{N}(0,1),$$

and use these to build the price matrix S_1. Then construct a second matrix S_2 by replacing each normal draw with its negation, $-W_j^i$. The two matrices S_1 and S_2 are identically distributed, and combining their payoffs will reduce variance via antithetic pairing.

Remark 5. In the case of the Brownian bridge method, we simulate to compute the first payoff row

$$U_j^i{}_{i=1,\dots,N-1,\ j=1,\dots,M} \quad \text{drawn i.i.d with} \quad U_j^i \sim \mathcal{U}([0,1]),$$

and then we take $(1 - U_j^i)_{i=1,\dots,N-1,\ j=1,\dots,M}$ to simulate the second payoff row.

For each path i in S_1, apply the chosen Parisian pricing algorithm to obtain the non-discounted payoff

$$X_1^i \;=\; \text{Payoff}\big(S_1^i, T\big),$$

and similarly from S_2 obtain

$$X_2^i \;=\; \text{Payoff}\big(S_2^i, T\big).$$

Whenever the payoff uses the In–Out parity, remember to convert the Black–Scholes price into a time-T cash flow by multiplying by e^{rT} before forming X_1^i or X_2^i.

We then form the adaptively antithetic combined payoff $\overline{X}$ such that:

$$\overline{X}_i = X_1^i \;-\; \hat{\lambda}_{i-1}^{\min}(X_1^i - X_2^i)$$

Finally, the option price is estimated by the discounted average of these adjusted payoffs:

$$\widehat{P} = e^{-rT}\,\frac{1}{M}\sum_{i=1}^{M}\overline{X}_i.$$

4 Results comments

We first examine the bias of our methods in subsection 4.1, including the individual method biases in subsection 4.1.1 and general observations in subsection 4.1.2. Next, we analyse the variance of each estimator in subsection 4.2. Finally, we compare the computational performance of the methods in subsection 4.3.

4.1 The bias of our methods

We first introduce a general framework for biased Monte Carlo simulation, taken from [3]. Our quantitiy of interest is:

$$I_0 = \mathbb{E}\, Y_0$$

Y_0 being a square integrable real random variable defined on some probability space, which cannot be simulated exactly because of its complexity. However we also introduce a sequence of real square integrable random variables defined on the same probability space $(Y_h)_{h\in\mathbf{H}}$ as $h \to 0$, with $\mathbf{H} \subset (0, +\inf)$ being a set of *bias* parameters, where the union of this set and 0 is compact (to allow for convergence and minimum arguments) and which has the following property:

$$\forall n \in \mathbb{N}^* \ , \quad \frac{\mathbf{H}}{n} \subset \mathbf{H}$$

Due to compactness,t he property above and $\mathbf{H}$ being non-empty, this set of bias parameters as a limit point. In our case we use the following parameters: $h = \frac{T}{n} \in \mathbf{H}$

In our case the random variables are functions of the genuine Euler scheme and of path functionals that appear in the payoff of Parisian option prices.

We now quote the following *bias-variance decomposition* from G. Pages. [4]

The starting point is the bias–variance decomposition of the mean-squared error when using independent copies $\{Y_h^k\}_{k\geq 1}$ of Y_h to approximate

$$I_0 = \mathbb{E}[Y_0]$$

by the Monte Carlo estimator

$$\widehat{I}_M = \frac{1}{M}\sum_{k=1}^{M} Y_h^k.$$

One obtains

$$\mathbb{E}\big[(Y_0 - \widehat{I}_M)^2\big] = \underbrace{\big(\mathbb{E}[Y_0] - \mathbb{E}[Y_h]\big)^2}_{\text{squared bias}} + \underbrace{\frac{\operatorname{Var}(Y_h)}{M}}_{\substack{\text{Monte Carlo}\\ \text{variance}}} \ .$$

4.1.1 The bias for each method

Naive Monte Carlo: First, as expected, we observe that the naive Monte Carlo method exhibits a significant and structural bias i.e $\lim_{h\to\infty}\big(\mathbb{E}[Y_0] - \mathbb{E}[Y_h]\big)^2 > 0$. Indeed, this bias is related to the discretization step: this method only examines the interval endpoints to determine whether the price has remained above or below the barrier. As previously explained, this bias is moreover of order $\mathcal{O}(\sqrt{\Delta t})$. We can therefore write the following approximation with $c_1^{\sqrt{\Delta t}}$ the induced discretization bias:[5]

$$\text{Bias}_{\text{Naive MC}} \ \approx \ c_1^{\sqrt{\Delta t}} \ + \ \mathcal{O}\big(\tfrac{1}{\sqrt{n}}\big), \ c_1^{\sqrt{\Delta t}} \in \mathbb{R}.$$

Therefore, in order to achieve a bias on the order of 10^{-4}, one would need a discretization on the order of 10^8, which is hardly prattical. While the application of variance-reduction techniques does indeed shrink the *Monte*

[3]See G. Pagès, *Numerical Probability*, Chapter §9.2

[4]See G. Pagès, *Numerical Probability*, Chapter §9.3

[5]Inspired by G. Pagès, *Numerical Probability*, Theorem §8.1

Carlo variance component - as clearly demonstrated in Figure 4 - a residual structural bias nevertheless remains. This constant bias, intrinsically linked to the discretization, clearly demonstrates the limitations of a naive Monte Carlo approach for Parisian options, and more generally for path-dependent options.

Brownian bride approach: The full relevance of the Brownian bridge emerges in these simulations. For the Brownian bridge, we can write following approximation with:[6]

$$\mathrm{Bias}_{\text{Naive MC}} \;\approx\; \mathcal{O}\big(\tfrac{1}{\sqrt{n}}\big).$$

The absence of bias i.e $\lim_{h\to\infty}\big(\mathbb{E}[Y_0]-\mathbb{E}[Y_h]\big)^2 = 0$ is clearly visible in the various graphs 5 and 6. From these plots one also sees that the bias - already modest due to the relatively low variance at $M = 10^5$ - is further reduced by variance reduction. In principle, one could eliminate this residual bias entirely by increasing the number of simulated trajectories M to reduce $\mathrm{Var}(Y_h)/M$. This method is very close to the Laplace price despite a relatively low number of simulations and discretization steps (100,000 and 250, respectively). However, its performance should be qualified: it is more cumbersome to implement, requiring an additional `if` loop in the algorithms to verify the condition as well as the extra simulation of a matrix of uniform random variables to model the probabilities. Although slightly heavier, this method nevertheless performs very well for path-dependent options and underscores the relevance of the Brownian bridge approach.

Laplace Transform: Finally, it goes without saying that the function employing the Laplace transform is unbiased, which is one of the undeniable advantages of this method.

4.1.2 General comments on the bias

Negative prices: First, note in Table 1 (our initial simulations without variance reduction) that both the Naive Monte Carlo and Brownian Bridge estimators produce a slightly *negative* price for the PUOC option (approximately -0.0053 and -0.0049, respectively), whereas the true Laplace price is 0.0113. To understand this, recall our parameter set: $S_0 = 100$, $L = 90$, $K = 100$, delay $D/T = 13\%$, $T = 1$ year, and $\sigma = 0.2$. With such moderate volatility and a relatively short maturity, most simulated paths remain near the initial spot. Since the call starts already above the barrier, very few paths ever cross down for a full D, so PUIC (Parisian Up–In Call) events dominate, while genuine PUOC (Up–Out) payoffs almost never occur. Moreover, when the option does go "Up–Out" (hence "Down–In"), the underlying is below $L = 90$ and also below the strike $K = 100$, so even realized payoffs are zero. Consequently, one expects the true PUOC price to be nearly zero.

However, in our Monte Carlo tables PUOC is not computed directly but via the In–Out parity

$$\mathrm{PUIC} + \mathrm{PUOC} = \mathrm{BSC},$$

and because both PUIC estimators are slightly biased high (more so for the naive method, less so for the bridge), parity forces PUOC into negative territory. This artefact of discretisation bias is entirely removed once variance reduction is applied (see Table 2), where the PUOC estimates become non-negative and match the Laplace benchmark.

A very similar phenomenon explains why PDIC (Parisian Down–In Call) prices in Table 1 are extremely close to zero: the combination of few valid down-in events and zero payoffs upon execution yields a near-zero true value, but bias-driven mis-pricing under parity can still push the estimate slightly off-center.

By contrast, PDIP (Parisian Down–In Put) maintains a substantial positive price even in the biased tables. Although down-in events remain rare, once the underlying spends D below the barrier around $L = 90$, the put payoff $(K - S_T)^+$ can be large (since $K = 100$), so each realized event contributes significant value. Thus, despite low probability, the average remains non-negligible.

All these effects are of course reflected in the variances reported in Table 1, to be analyzed in the next section.

[6] Inspired by G. Pagès, *Numerical Probability*, Theorem §8.1

Effect of using a single price-series versus re-simulating per delay without variance reduction: In addition to the parity-driven bias, our bias-versus-delay plots (Figure 7) reveal a second key effect: whether we reuse the same simulated paths for all delays or re-simulate afresh at each delay markedly influences the apparent bias, especially for the Brownian-bridge estimator.

- **Single price-series for all delays.** When we fix one Monte Carlo sample of 100 000 paths (and one corresponding uniform variate array for the bridge) and simply vary the delay D/T from 1% to 25%, both naive and bridge estimators show bias. In the naive case this stems solely from the coarse time mesh ($\Delta t = 1/250$) (see Figure 3), but for the bridge there is a *second* bias introduced by reusing the same uniform variate matrix: correlations across delay levels amplify discretisation error (see Figure 5). In Figure 7(a) one sees the bridge bias up to twice as large as in independent runs. This simulation biais is present wether a variance reduction approach has been applied or not, just a litlle more difficult to read with it for the Brownian bridge because of the near complete absence of bias.
- **Re-simulation per delay.** If instead we regenerate a fresh set of price paths (and fresh uniforms for the bridge) at each delay, the random-number correlations disappear. The naive bias remains essentially unchanged (driven by $\sqrt{\Delta t}$), but the bridge bias collapses back to its nominal discretisation-free level, of order Var/N. This behavior confirms that re-sampling both normals and uniforms for each D restores the statistical independence required for the bridge correction to be unbiased across delay grid points.

In practice, therefore, when studying sensitivity to the delay parameter it is crucial to re-simulate at each delay—or else interpret the inflated bias in the single-sample curves as an artefact of persistence in the underlying random draw. This insight is especially important for moderate path counts (here 10^5 paths and 250 steps), where re-using uniforms can double the effective bias in a Brownian-bridge Monte Carlo.

Bias after variance reduction Once the adaptive - antithetic variance reduction is applied - shrinking variances by factors between 10^4 and 10^5 depending on the option—the bias patterns become more subtle (see Figure **??**). In particular:

- **Naive Monte Carlo:** The residual bias remains virtually identical regardless of whether we re-simulate the paths for each delay or reuse the same sample. This confirms that, beyond any simulation-specific noise, the naive approach harbors an intrinsic $\mathcal{O}(\sqrt{\Delta t})$ discretisation bias that variance reduction cannot remove, rendering it ill-suited for path-dependent payoffs.
- **Brownian Bridge:** Here the only remaining bias is the statistical error Var/N, which, after reduction, drops to close to zero. However, by comparing the two modes (single reuse vs. per-delay re-simulation) one still observes a small difference in bias—precisely the sampling-noise effect caused by reusing the uniform variate matrix. This minor effect disappears when uniforms are also re-drawn for each delay, in line with our earlier findings.

4.2 The variance of our methods

Naive Monte Carlo: The variance for the naive Monte Carlo method is high. As can be seen in Table 1, it is especially large for those options whose condition is realized often with high payoffs. For example, as explained above, the PUIC options are frequently in-the-money with large payoffs, leading to a variance of 194.6. The variance for PUOC is the same, since its payoff is obtained via the In–Out parity (discounting the Black–Scholes price from 0 to T rather than using the call payoff directly, to reduce variance). It might seem contradictory that the variance of PDOC is only 3.328. In fact, this stems from our implementation: for naive Monte Carlo we coded PDIC and PUIC directly and derived PDOC and PUOC via parity (the puts via the inverse relations). Consequently, since PDIC occurs very rarely and with smaller payoffs, its variance is very low, and PDOC's variance is low by consequence.

The PDIP/PDOP and PUIP/PUOP options are obtained via inversion from PUIC and PDIC. One would therefore expect a high variance for PDIP/PDOP and a lower variance for PUIP/PUOP, reflecting the variance of the call used. However, here the strike parameter $K = 100$ exceeds the barrier $L = 90$. Below the barrier, a put's payoff is substantial but remains strictly positive even when the underlying lies between 90 and 100. Therefore, even if the event occurs less frequently (or often with zero payoff for PUIP), the payoffs that do occur are of larger magnitude, explaining the variances of 136.7 and 222.9 for PDIP/PDOP and PUIP/PUOP, respectively.

Finally, after variance reduction (see Table 2), the variances drop by roughly six orders of magnitude for all options, to between 10^{-4} and 10^{-6}, effectively negligible. In Figure 4, the post-reduction confidence intervals are no longer visible because they coincide with the estimated prices (they are plotted but overlap the lines). The relative order among options remains unchanged. Note, however, that variance reduction methods degrade the In–Out parity accuracy from about 10^{-15} to 10^{-5}.

Brownian Bridge: Table 1 shows that the qualitative ordering of variances across options mirrors the naive case. However, the Brownian-bridge method enjoys both a far smaller bias and a lower variance. Comparing confidence bands in Figures 3 and 5 reveals reductions of roughly -13.7% for PDIC/PDOC, 0% for PUIC/PUOC, -32.2% for PDIP/PDOP, and -32.5% for PUIP/PUOP.

The Brownian Bridge uses the same barrier checks as naive Monte Carlo, but adds one extra test based on the uniform variate acceptance probability. Consequently, fewer non - zero payoffs are realized - more paths contribute a payoff of zero—so the overall variance falls. This additional rejection step does not alter the mean (and hence the option price), because the paths discarded by the uniform-based test are almost always those whose payoff would have been near zero anyway (e.g. out-of-the-money calls or puts).

A notable exception is PUIC (and consequently PUOC): as explained above, many PUIC events occur under $S_0 = 100$, $K = 100$, $L = 90$, so $\mathbb{P}\{\inf \text{bridge} > L\} \approx 1$. In that case, the extra test is almost never binding, and both the Brownian Bridge price and its variance remain very close to the naive Monte Carlo values.

Finally, when we apply variance reduction to the Brownian Bridge (see Figure. 2), the variance drops dramatically. However, it decreases relatively less than for naive Monte Carlo and ends up very close, for each option, to the "reduced" Monte Carlo variance. In fact this residual variance corresponds to the σ^2_{min} of the optimal constant for the variance of our error $\sim \mathcal{N}(0, \frac{\sigma^2_{min}}{n})$. The strongest driver to reduce the variable is then the number of simulations as the variance reduction method makes both methods converge to the same values even if they began at different initial variances. Finally, the comments made for the adaptive-Monte Carlo (see Figure 4)plot apply equally to the Brownian Bridge (see Figure. 6).

Laplace Transform. The Laplace-based method has a closed-form solution for the integrals and therefore does not need approximation through MC methods. It has zero variance by construction.

4.3 The speed of our methods

The performances presented here are the result of several optimization choices that have been made. For more details, please consult the Appendix 2dedicated to implementation choice details.

The performances detailed come from a dual-core Intel Core i5-6300U @ 2.4 GHz, 16 GB RAM:

Simulating 500 million centered Gaussian random variables $\left(W^i_j\right)_{i\in\{1,\dots,N\},\, j\in\{1,\dots,M\}} \sim \mathcal{N}(0,1)$ takes between 9 s and 11 s. Generating the 500M associated paths requires approximately 7–10 s. Simulating 500M - 500 uniform variables $\left(U^i_j\right)_{i\in\{1,\dots,N-1\},\, j\in\{1,\dots,M\}} \sim \mathcal{U}([0,1])$ likewise takes around 7–10 s.

Consequently, running 500 million trajectories for the Naive Monte Carlo method takes about 16–20 s in total, while the Brownian-bridge approach requires roughly 23–30 s. These timings can vary slightly depending on other processes occupying RAM (which can reduce parallel-execution speed).

When performing antithetic variance reduction, a second price matrix must be simulated using $\left(-W^i_j\right)_{i\in\{1,\dots,N\},\, j\in\{1,\dots,M\}}$ and as well a second matrix of uniform for the Brownian bridge, $\left(1 - U^i_j\right)_{i\in\{1,\dots,N-1\},\, j\in\{1,\dots,M\}}$ that respectively take 7-10s and 1-2s.

Price simulations:

- *Naive Monte Carlo:* Pricing via Naive Monte Carlo takes rougly 0.5s for calls (directly computed) and roughly 1-1.5 second for puts (due to the matrix inversion step). For the anthitetic price, it takes roughly 1s for calls and 2.5s for puts. To accelerate the put option pricing, one may directly implement the algorithms for PDIP and PUIP and use the parity relation for PDOP and PUOP to obtain the same speed as for call options.
- *Brownian bridge:* The time is between 1.5-3s for both calls and puts and with the anthitetic price it jumps to 3.5 - 5s.
- *Laplace inversion:* This method computes the kernel evaluation through a closed formula and accelerated Euler summation in a few tens of milliseconds. This is by far the fastest pricing approach with simple function evaluations only and a modest sum to calculate the prices. Changing the relative error by tuning α doesn't affect the computation speed at all. In this case, using Python or C++ does not matter that much.

It is important to note that, although some methods are structurally more expensive than others (for example, pricing puts with the Naive Monte Carlo due to the inversion step), the total computation time depends heavily on the chosen simulation parameters. In particular, with $S_0 = 100$, $K = 100$, $L = 90$, and a large ratio D/T, the barrier condition (endpoint checks for Naive Monte Carlo, endpoint checks plus uniform-draw test for the Brownian bridge) will be met more often for certain options. This increases the number of `if` statements executed per path and hence the overall run time.

5 Conclusion

We explained in detail the closed-form Laplace transforms for Parisian options, analyzed the error - controlled inversion formulae, and implemented different Monte Carlo pricing techniques and variance reduction techniques. It is once again clearly confirmed in this paper that the Laplace transform is the gold standard for option pricing of path-dependent options that have a convolution-type expression of the payoff. Our tests also clearly showed the clear advantage of the Brownian Bridge compared to the Naive Monte Carlo, notably due to the lack of any link between its bias and the discretization step.We were also able to confirm the effectiveness of the adaptive antithetic approach for variance reduction applied to path - dependant options, which—without much additional computation time—delivers a significant decrease in variance

This highlights the comparatively strong performance of these tuned Monte Carlo methods for path-dependent options which can be used for extensions of Parisian options that involve more complex types of barriers. Alternatively, it can serve as a fast, highly accurate benchmark against which to compare more sophisticated pricing techniques - such as PDE - based methods or neural-network models—that have become widespread in modern research.

Future research and extensions could include (i) Parisian features under stochastic volatility and Lévy jumps, (ii) early exercise via free-boundary formulations and (iii) a precise Brownian Bridge formulation, which conditions on the stopping time τ_b^- and uses its probability distribution to obtain an exact variance reduced formula for the payoff.

Appendix 1 Proofs of Main Theorems

In Appendix 1, we collect all demonstrations. We prove the In–Out parity in Section Appendix 1.1, derive the inversion relations in Section Appendix 1.2, establish the barrier-sign reduction in Section Appendix 1.3, and conclude with the Laplace transform of the Down–In call in Section Appendix 1.4.

Appendix 1.1 Proof of In–Out Parity

We define:

$$g^S_{L,t} := \sup\{\, u \le t : S_u = L\}, \qquad d^S_{L,t} := \inf\{\, u \ge t : S_u = L\}.$$

Such an excursion can also be described in terms of the Brownian motion W. For a given barrier L for the process S, we introduce the corresponding barrier b for W defined by

$$b \;=\; \frac{1}{\sigma}\,\ln\Bigl(\frac{L}{S_0}\Bigr).$$

We then introduce:

$$T_b^-(W) = \inf\bigl\{\, t > D : W_s < b \text{ for all } s \in [t-D,t]\bigr\}.$$
$$T_b^+(W) = \inf\bigl\{\, t > D : W_s > b \text{ for all } s \in [t-D,t]\bigr\}.$$

$$\begin{aligned}
\mathrm{PDIC}(x,T;K,L,D;r,\delta) + \mathrm{PDOC}(x,T;K,L,D;r,\delta) &= \mathbb{E}^{\mathbb{Q}}\bigl[\mathbf{1}_{\{T_b^- \le T\}}\,(S_T-K)^+\bigr] + \mathbb{E}^{\mathbb{Q}}\bigl[\mathbf{1}_{\{T_b^- \ge T\}}\,(S_T-K)^+\bigr]\\
&= \mathbb{E}^{\mathbb{Q}}\bigl[(S_T-K)^+\bigr]\\
&= \mathrm{BSC}(x,T;K;r,\delta).
\end{aligned}$$

By a similar argument, the other In–Out parity relations follow easily.

Appendix 1.2 Proof of Inversion Relations

Let

$$m \;=\; \frac{1}{\sigma}\Bigl(r-\delta-\tfrac{\sigma^2}{2}\Bigr),$$

and let $\mathbb{P}$ be the probability measure under which

$$Z_t \;:=\; W_t + mt, \quad 0 \le t \le T,$$

is a $\mathbb{P}$-Brownian motion. The change of measure from $\mathbb{P}$ to $\mathbb{Q}$ is given by

$$\left.\frac{d\mathbb{Q}}{d\mathbb{P}}\right|_{\mathcal{F}_T} = \exp\Bigl(mZ_T - \tfrac{m^2}{2}T\Bigr).$$

Under $\mathbb{P}$, the asset price process $S = (S_t)_{0\le t\le T}$ follows

$$\forall\, t \in [0,T], \quad S_t \;=\; x\,e^{\sigma Z_t}.$$

Consider the *Parisian Down-and-Out put*

$$\mathrm{PDOP}(x,T;K,L;r,\delta) = \mathbb{E}\Bigl[\,e^{mZ_T}\,(K - xe^{\sigma Z_T})^+\,\mathbf{1}_{\{\tau_b^- > T\}}\Bigr]\,\exp\Bigl(-\bigl(r+\tfrac{m^2}{2}\bigr)T\Bigr),$$

where

$$\tau_b^- \;:=\; \inf\{\, t \ge 0 : Z \text{ makes an excursion below } b \text{ of length } D\}.$$

Notice that τ_b^- for Z is the same as the first time $-Z$ makes an excursion above $-b$ of length D. Introducing the Brownian motion $W := -Z$, we can rewrite

$$\mathrm{PDOP}(x,T;K,L;r,\delta) = \mathbb{E}\Bigl[\,e^{-mW_T}\,(K - xe^{-\sigma W_T})^+\,\mathbf{1}_{\{\tau_{-b}^+ > T\}}\Bigr]\,\exp\Bigl(-\bigl(r+\tfrac{m^2}{2}\bigr)T\Bigr),$$

where

$$\tau^+_{-b} \; := \; \inf\{\, t \geq 0 : W \text{ makes an excursion above } -b \text{ of length } D\}.$$

Expanding the payoff and factoring, we obtain

$$\mathrm{PDOP}(x, T; K, L; r, \delta) = xK\, \mathbb{E}\Big[e^{-(m+\sigma)W_T} \Big(\tfrac{1}{x} e^{\sigma W_T} - \tfrac{1}{K}\Big)^+ \mathbf{1}_{\{\tau^+_{-b} > T\}}\Big] \; \exp\!\Big(-\big(r + \tfrac{m^2}{2}\big)T\Big).$$

Now set

$$m' := -(m+\sigma), \quad \delta' := r, \quad r' := \delta, \quad b' := -b.$$

One checks easily that

$$m' \; = \; \frac{1}{\sigma}\Big(r' - \delta' - \tfrac{\sigma^2}{2}\Big), \quad r' + \tfrac{m'^2}{2} \; = \; r + \tfrac{m^2}{2},$$

and that the corresponding barrier is $L' = 1/L$. Hence

$$\mathbb{E}\Big[e^{-(m+\sigma)W_T} \Big(\tfrac{1}{x} e^{\sigma W_T} - \tfrac{1}{K}\Big)^+ \mathbf{1}_{\{\tau^+_{-b} > T\}}\Big] \; \exp\!\Big(-\big(r + \tfrac{m^2}{2}\big)T\Big)$$

is precisely the price of the *Up-and-Out call*

$$\mathrm{PUOC}\big(1/x,\, T;\, 1/K,\, 1/L;\, \delta, r\big).$$

Therefore we arrive at the relation

$$\boxed{\mathrm{PDOP}(x, T; K, L; r, \delta) = xK\; \mathrm{PUOC}\big(1/x,\, T;\, 1/K,\, 1/L;\, \delta, r\big).}$$

The three other inverse relations are proved in exactly the same way.

Appendix 1.3 Proof of Barrier-sign reduction for PDOC

We begin by recalling the expression for the value of the Parisian down-and-out call option:

$$\mathrm{PDOC}^\star(x, T; K, L; r, \delta) = \mathbb{E}\left[\mathbf{1}_{\{T_b^- \geq T\}} \left(x e^{\sigma Z_T} - K\right)^+ e^{m Z_T}\right],$$

where Z is a Brownian motion starting at 0, and T_b is the first passage time of Z above the level $b > 0$.

Note that on the set $\{T_b^- \geq T\}$, it must be that $T_b \leq D$, where D is the time horizon. If $T_b^- > D$, then this would imply that Z hasn't crossed b before D, contradicting $T_b^- \geq T$ with $T > D$. Thus,

$$\mathrm{PDOC}^\star(x, T; K, L; r, \delta) = \mathbb{E}\left[\mathbf{1}_{\{T_b^- \geq T\}} \mathbf{1}_{\{T_b \leq D\}} \left(x e^{\sigma Z_T} - K\right)^+ e^{m Z_T}\right].$$

Using the strong Markov property at time T_b, let Z_{T_b} be the value of the process at T_b and define $B_t := Z_{T_b + t} - Z_{T_b}$, a Brownian motion independent of $\mathcal{F}_{T_b}$. On $\{T_b^- \geq T\}$, we have the following relation that is straightforward to prove by a simple manipulation and :

$$T_b^-(Z) - T_b(Z) = T_0^-(B) \quad a.s. \text{ on the set } \quad \{T_b^- \geq T\}$$

In fact we have that:

$$T_b^-(Z) - T_b(Z) = inf\{t - T_b(Z) > 0, \quad (t - g_t^b)\mathbf{1} Z_t < b \geq D\} = inf\{t' > 0, \quad (t' + T_b - g^b_{t'+T_b})\mathbf{1}_{Z_{t'+T_b} - Z_{T_b} < 0}\} = T_0(B),$$

because we have that $((t' + T_b - g^b_{t'+T_b}) = t' - g^b_{t'}$ under the condition of the infimum, where we performed a change of variables and where $Z_{t'+T_b} - Z_{T_b} = B_{t'}$. We thus have the following:

$$\begin{aligned}\mathrm{PDOC}^\star(x, T; K, L; r, \delta) &= \mathbb{E}\left[\mathbf{1}_{\{T_b \leq D\}}\, \mathbb{E}\left[\mathbf{1}_{\{T_b - T_b \geq T - T_b\}} \left(x e^{\sigma(Z_T - Z_{T_b} + b)} - K\right)^+ + e^{m(Z_T - Z_{T_b} + b)} \,\middle|\, \mathcal{F}_{T_b}\right]\right] \\ &= \mathbb{E}\left[\mathbf{1}_{\{T_b \leq D\}}\, \mathbb{E}\left[\mathbf{1}_{\{T_0 \geq T - t\}} \left(x e^{\sigma(B_{T-t} + b)} - K\right)^+ + e^{m(B_{T-t} + b)}\right]\Bigg|_{t = T_b}\right]\end{aligned}$$

Here we used the independence of B and $\mathcal{F}_{T_b}$ in the second equality.

Hence, we are conditioning on $T_b = u$, and we obtain:

$$\mathrm{PDOC}^{\star}(x,T;K,L;r,\delta) = \int_0^D \mathbb{E}\left[\mathbf{1}_{\{T_0 \geq T-u\}}\left(xe^{\sigma(B_{T-u}+b)} - K\right)^+ + e^{m(B_{T-u}+b)}\right]\mu_b(du),$$

where μ_b is the distribution of T_b.

By substituting $b = \frac{1}{\sigma}\ln\frac{L}{x}$, we simplify:

$$\mathrm{PDOC}^{\star}(x,T;K,L;r,\delta) = \int_0^D \mathbb{E}\left[\mathbf{1}_{\{T_0 \geq T-u\}}\left(e^{\sigma B_{T-u}} - \frac{K}{L}\right)^+ e^{mB_{T-u}}\right]\mu_b(du).$$

Using the convolution property of the Laplace transform it is very straightforward to derive the Laplace transform representation of $\mathrm{PDOC}^{\star}(x,T;K,L;r,\delta)$.

Appendix 1.4 Laplace Transform of PDIC

Recall that the value of the Parisian down-and-in call option is given by

$$\mathrm{PDIC}^{\star}(x,T;K,L;r,\delta) = \mathbb{E}\left[\mathbf{1}_{\{T_b^- < T\}}\left(xe^{\sigma Z_T} - K\right)^+ + e^{mZ_T}\right].$$

Conditioning on $\mathcal{F}_{T_b^-}$ yields

$$\mathrm{PDIC}^{\star}(x,T;K,L;r,\delta) = \mathbb{E}\left[\mathbf{1}_{\{T_b^- < T\}}\mathbb{E}\left[\left(xe^{\sigma(Z_T - Z_{T_b^-} + Z_{T_b^-})} - K\right)^+ + e^{m(Z_T - Z_{T_b^-} + Z_{T_b^-})}\,\middle|\,\mathcal{F}_{T_b^-}\right]\right].$$

Let $B_t = Z_{t+T_b^-} - Z_{T_b^-}$, a Brownian motion independent of $\mathcal{F}_{T_b^-}$. Denote $\tau = T_b^-$ and $z = Z_{T_b^-}$. Then,

$$\mathbb{E}\left[\left(xe^{\sigma(Z_T - Z_{T_b^-} + z)} - K\right)^+ + e^{m(Z_T - Z_{T_b^-} + z)}\right] = \mathbb{E}\left[\left(xe^{\sigma(B_{T-\tau}+z)} - K\right)^+ + e^{m(B_{T-\tau}+z)}\right].$$

This conditional expectation admits an explicit representation:

$$\mathbb{E}\left[\left(xe^{\sigma(B_{T-\tau}+z)} - K\right)^+ + e^{m(B_{T-\tau}+z)}\right] = \frac{1}{\sqrt{2\pi(T-\tau)}}\int_{-\infty}^{\infty}\left(xe^{\sigma u} - K\right)^+ e^{mu}e^{-\frac{(u-z)^2}{2(T-\tau)}}du.$$

Define the function $f_x(z) = e^{mz}\left(xe^{\sigma z} - K\right)^+$ and the Gaussian semigroup

$$P_t(f_x)(z) = \frac{1}{\sqrt{2\pi t}}\int_{-\infty}^{\infty} f_x(u)\exp\left(-\frac{(u-z)^2}{2t}\right)du.$$

Then,

$$\mathrm{PDIC}^{\star}(x,T;K,L;r,\delta) = \mathbb{E}\left[\mathbf{1}_{\{T_b^- < T\}}P_{T-T_b^-}(f_x)(Z_{T_b^-})\right].$$

As shown by Chesney et al. (1997), $Z_{T_b^-}$ and T_b^- are independent, which is actually a direct consequence of the Blumenthal-0-1 law. Let $\nu^-(dz)$ be the law of $Z_{T_b^-}$, so

$$\mathrm{PDIC}^{\star}(x,T;K,L;r,\delta) = \int_{-\infty}^{\infty}\mathbb{E}\left[\mathbf{1}_{\{T_b^- < T\}}P_{T-T_b^-}(f_x)(z)\right]\nu^-(dz).$$

Let

$$h_b^-(t,y) = \int_{-\infty}^{\infty} \mathbb{E}\left[\mathbf{1}_{\{T_b^- < t\}}\right] \frac{1}{\sqrt{2\pi(t-T_b^-)}} \exp\left(-\frac{(z-y)^2}{2(t-T_b^-)}\right) \nu^-(dz).$$

From Appendix C of the paper, the density ν^- is given by

$$\nu^-(dz) = \frac{b-z}{D} e^{-\frac{(z-b)^2}{2D}} dz, \quad z < b.$$

Therefore,

$$h_b^-(t,y) = \int_{-\infty}^{b} \frac{b-z}{D} e^{-\frac{(z-b)^2}{2D}} \mathbb{E}\left[\mathbf{1}_{\{T_b^- < t\}}\right] \frac{1}{\sqrt{2\pi(t-T_b^-)}} \exp\left(-\frac{(z-y)^2}{2(t-T_b^-)}\right) dz,$$

and the result follows.

Appendix 2 Implementation Language

The question of whether to implement our Parisian option pricer in Python or in C++ naturally arose. Monte Carlo methods require simulating a very large number of paths and performing numerous barrier checks, and when we need tens or hundreds of millions of points - crucial for accurately detecting barrier crossings in Parisian options - the performance constraints become severe. While Python offers powerful vectorized libraries (notably NumPy), pure Python/NumPy is not sufficient at this scale, and a lower-level language such as C++ might seem the obvious choice.

However, Python remains highly accessible and readable, and it provides excellent tools for results presentation - tables via pandas and customizable plots via Matplotlib - that we did not wish to abandon. C++, by contrast, tends to produce more verbose and less immediately understandable code, and it lacks the interactive data-analysis ecosystem that Python provides.

To balance high numerical performance with Python's ease of use and display capabilities, we chose to remain in the Python ecosystem but to delegate all compute-intensive loops to machine-level code via Numba. Concretely, our strategy was:

- **Python for orchestration and display.** All experiment setup, data-frame management, and plotting remain in pure Python.
- **Numba for numeric kernels.** Every critical routine—path generation, barrier - crossing tests, variance-reduction steps, Laplace - kernel evaluations, and Euler inversion—is decorated with `@njit(parallel=True, cache=True)`. Numba compiles these functions with LLVM into optimized native code, offering:
  - *Automatic multi-threading.* The `parallel=True` flag and use of `prange` transform our Monte Carlo loops into OpenMP-parallel loops without writing any C/C++ or OpenMP directives by hand.
  - *Persistent cache.* The `cache=True` option stores compiled machine code on disk, so recompilation is avoided on subsequent runs and start-up is instantaneous.

Several important points must be noted when working with Numba.

First, Numba is not capable of generating random variables in a fully vectorized way. Instead, random numbers must be generated individually using explicit loops. In such cases, NumPy tends to be faster. For instance, when constructing price matrices, we use NumPy to generate the matrix of standard normal random variables, and then pass it to a Numba-compiled function to compute the price paths. We also use NumPy for the uniform matrix for the Brownian bridge method.

It is also important to understand the limitations of Numba's parallelization capabilities. In particular, loops where the current iteration depends on the outcome of previous iterations cannot be parallelized. For example, in our pricing functions, we can parallelize over Monte Carlo paths, but not over the time discretization steps when we are counting how many consecutive subintervals satisfy a certain condition.

Finally, a practical consideration when using Numba is that compilation is deferred. This means that the first time a Numba-compiled function is called, it incurs an additional overhead due to just-in-time compilation. To ensure optimal performance, each Numba function should be called once initially to trigger compilation, and then called again to benefit from the full speed-up.

Since our Monte Carlo kernels are parallelized with Numba and routinely allocate massive arrays in RAM, efficient memory management is critical - and Python gives us the tools to manage a part of it directly.

Explicit Memory Management with gc: In large-scale Monte Carlo simulations we routinely allocate and discard very large NumPy arrays (Gaussian variables, path matrices, antithetic copies, uniform variates for the bridge correction, etc.). Without intervention, Python's memory can balloon or even leak due to hidden reference cycles. To keep RAM usage under control, we combine two explicit steps after each heavy operation:

- `del` statements to drop all Python references to temporary arrays (so their reference count reaches zero when no longer in use).
- `gc.collect()` calls to trigger the built-in garbage collector, which scans for and frees cyclically referenced objects that simple reference counting cannot reclaim.

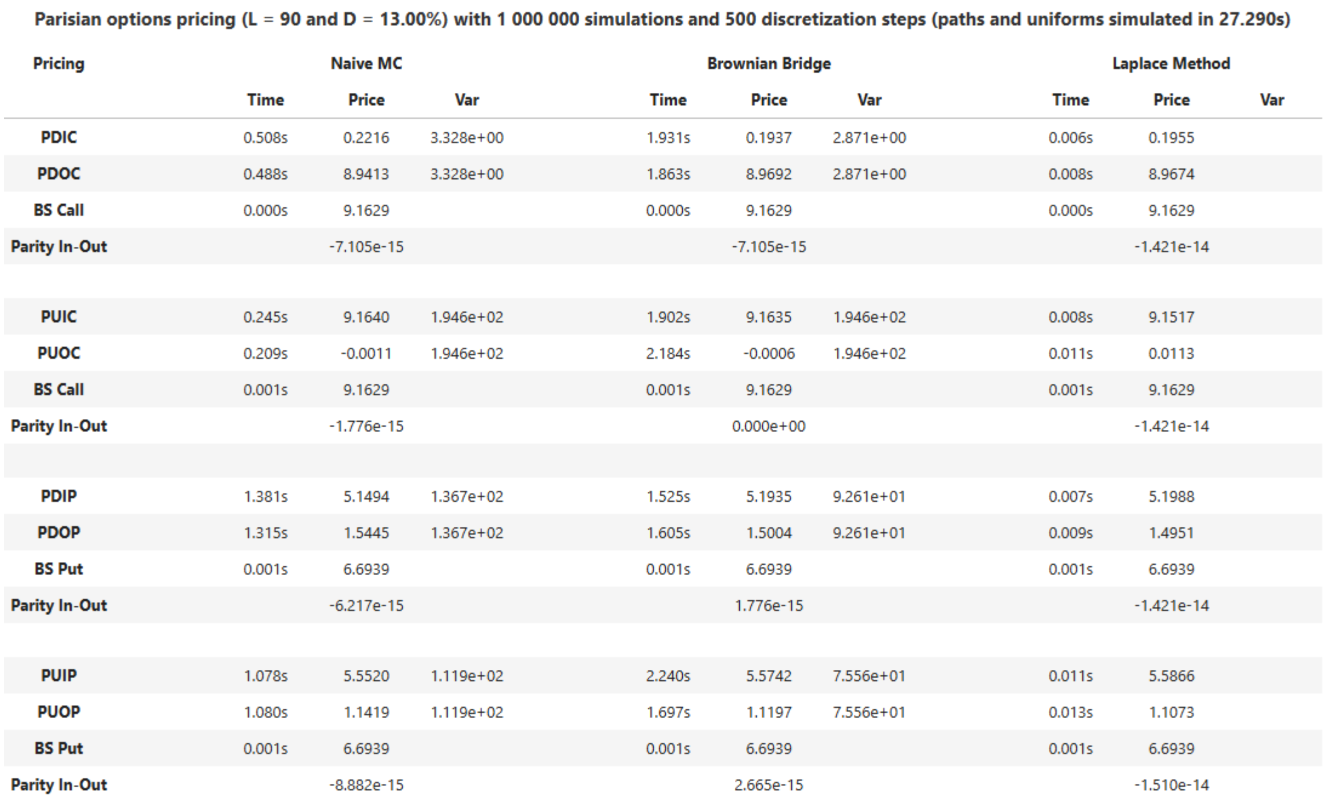
Under the hood, Python uses immediate reference counting plus a generational garbage collector for cycles. By manually invoking `gc.collect()` at key points - right after path generation, after payoff computation, and after any antithetic pairing - we ensure that both standalone (done automatically by Python) and cyclic objects are reclaimed as soon as they become unreachable. This explicit cleanup prevents memory bloat during multi-million-path runs and avoids out-of-memory errors on long experiments.

In summary, combining Python with Numba provides the best of both worlds: the ease of development and readability of Python, along with performance that is often comparable to compiled languages such as C for numerically intensive tasks.

Appendix 3 Results

Appendix 3 presents our numerical findings. First we display raw pricing data for all methods in Section 1, then data with variance reduction in Section 2. Next, we compare Naive Monte Carlo versus Laplace in Section 3 and Section 4, followed by the Brownian bridge comparisons in Sections 5 and 6, and finally the cross-method plots in Sections 7 and 8.

Appendix 3.1 Data Parisian Options Pricing for all methods

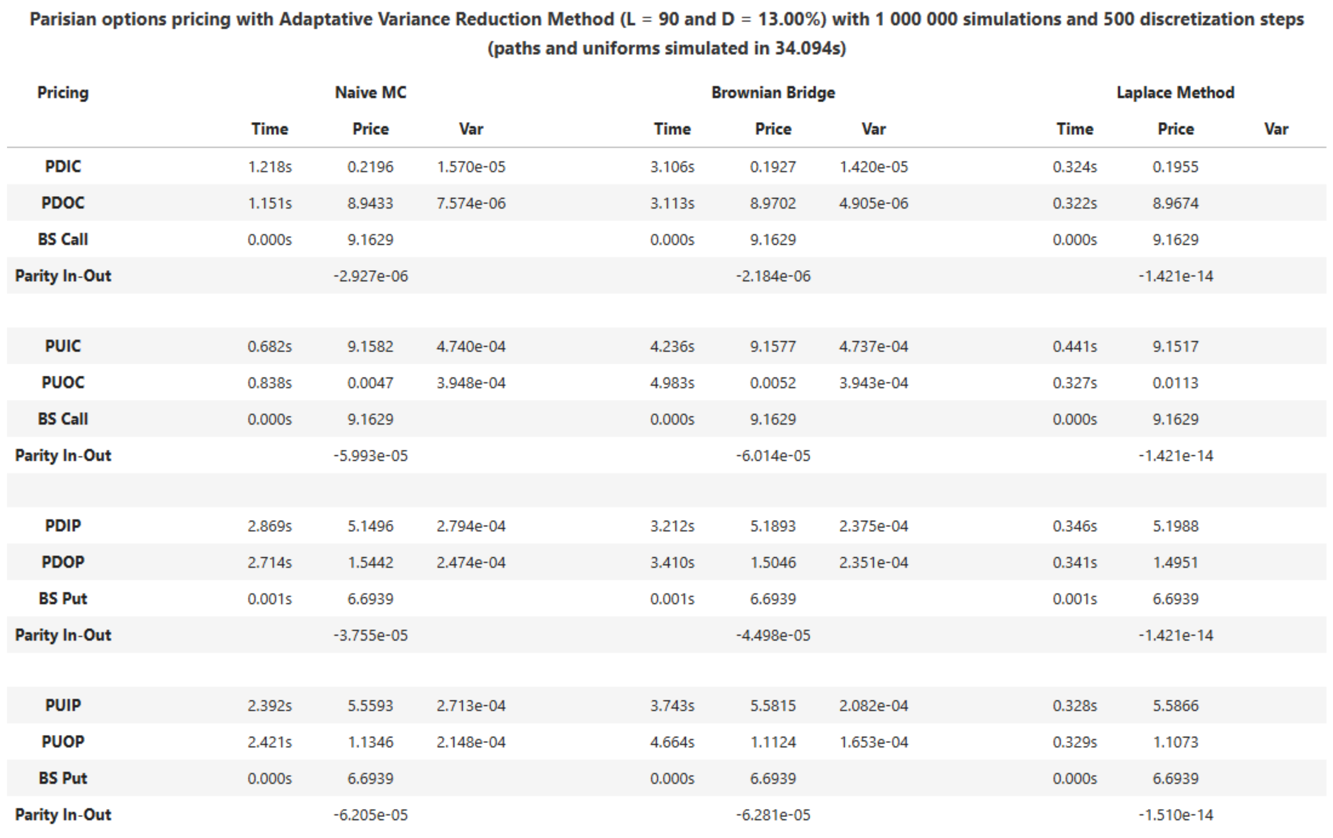


Parisian options pricing (L = 90 and D = 13.00%) with 1 000 000 simulations and 500 discretization steps (paths and uniforms simulated in 27.290s)

Pricing	Naive MC			Brownian Bridge			Laplace Method		
	Time	Price	Var	Time	Price	Var	Time	Price	Var
PDIC	0.508s	0.2216	3.328e+00	1.931s	0.1937	2.871e+00	0.006s	0.1955	
PDOC	0.488s	8.9413	3.328e+00	1.863s	8.9692	2.871e+00	0.008s	8.9674	
BS Call	0.000s	9.1629		0.000s	9.1629		0.000s	9.1629	
Parity In-Out		-7.105e-15			-7.105e-15			-1.421e-14	
PUIC	0.245s	9.1640	1.946e+02	1.902s	9.1635	1.946e+02	0.008s	9.1517	
PUOC	0.209s	-0.0011	1.946e+02	2.184s	-0.0006	1.946e+02	0.011s	0.0113	
BS Call	0.001s	9.1629		0.001s	9.1629		0.001s	9.1629	
Parity In-Out		-1.776e-15			0.000e+00			-1.421e-14	
PDIP	1.381s	5.1494	1.367e+02	1.525s	5.1935	9.261e+01	0.007s	5.1988	
PDOP	1.315s	1.5445	1.367e+02	1.605s	1.5004	9.261e+01	0.009s	1.4951	
BS Put	0.001s	6.6939		0.001s	6.6939		0.001s	6.6939	
Parity In-Out		-6.217e-15			1.776e-15			-1.421e-14	
PUIP	1.078s	5.5520	1.119e+02	2.240s	5.5742	7.556e+01	0.011s	5.5866	
PUOP	1.080s	1.1419	1.119e+02	1.697s	1.1197	7.556e+01	0.013s	1.1073	
BS Put	0.001s	6.6939		0.001s	6.6939		0.001s	6.6939	
Parity In-Out		-8.882e-15			2.665e-15			-1.510e-14	

Figure 1: Parisian Options Pricing with the three methods:
$S_0 = 100, L = 90, D = 13\%, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 1\ 000\ 000, \text{discretization} = 500$

Appendix 3.2 Data Parisian Options Pricing with variance reduction for all methods

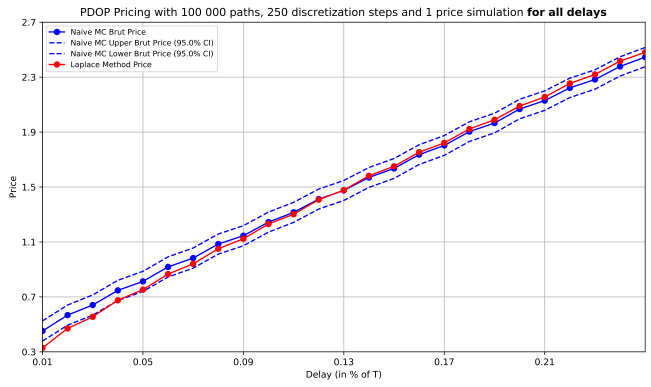


Parisian options pricing with Adaptative Variance Reduction Method (L = 90 and D = 13.00%) with 1 000 000 simulations and 500 discretization steps (paths and uniforms simulated in 34.094s)

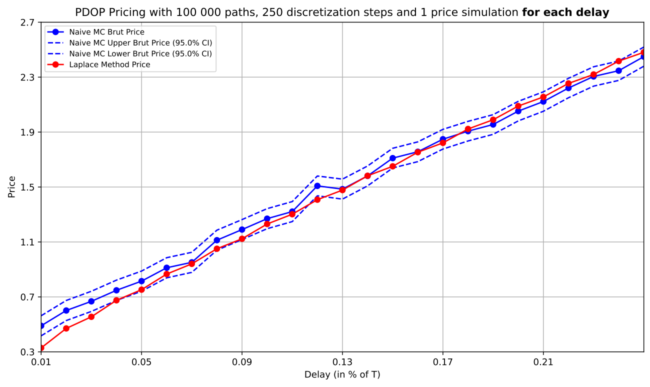
Pricing		Naive MC			Brownian Bridge			Laplace Method	
	Time	Price	Var	Time	Price	Var	Time	Price	Var
PDIC	1.218s	0.2196	1.570e-05	3.106s	0.1927	1.420e-05	0.324s	0.1955	
PDOC	1.151s	8.9433	7.574e-06	3.113s	8.9702	4.905e-06	0.322s	8.9674	
BS Call	0.000s	9.1629		0.000s	9.1629		0.000s	9.1629	
Parity In-Out		-2.927e-06			-2.184e-06			-1.421e-14	
PUIC	0.682s	9.1582	4.740e-04	4.236s	9.1577	4.737e-04	0.441s	9.1517	
PUOC	0.838s	0.0047	3.948e-04	4.983s	0.0052	3.943e-04	0.327s	0.0113	
BS Call	0.000s	9.1629		0.000s	9.1629		0.000s	9.1629	
Parity In-Out		-5.993e-05			-6.014e-05			-1.421e-14	
PDIP	2.869s	5.1496	2.794e-04	3.212s	5.1893	2.375e-04	0.346s	5.1988	
PDOP	2.714s	1.5442	2.474e-04	3.410s	1.5046	2.351e-04	0.341s	1.4951	
BS Put	0.001s	6.6939		0.001s	6.6939		0.001s	6.6939	
Parity In-Out		-3.755e-05			-4.498e-05			-1.421e-14	
PUIP	2.392s	5.5593	2.713e-04	3.743s	5.5815	2.082e-04	0.328s	5.5866	
PUOP	2.421s	1.1346	2.148e-04	4.664s	1.1124	1.653e-04	0.329s	1.1073	
BS Put	0.000s	6.6939		0.000s	6.6939		0.000s	6.6939	
Parity In-Out		-6.205e-05			-6.281e-05			-1.510e-14	

Figure 2: Parisian Options Pricing with the three methods and variance reduction:
$S_0 = 100, L = 90, D = 13\%, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 1\ 000\ 000, \text{discretization} = 500$

Appendix 3.3 Naive Monte Carlo and Laplace Transform methods

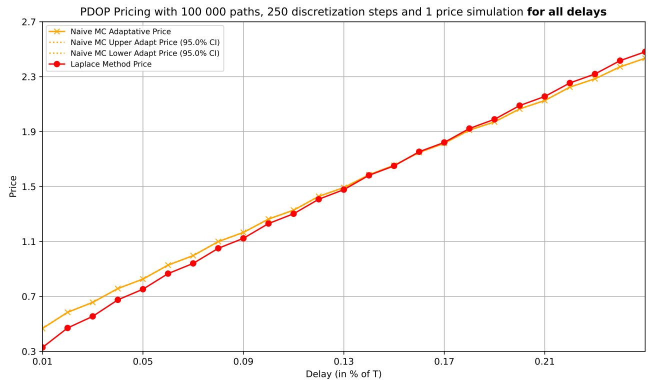


(a) One price simulation for all delays

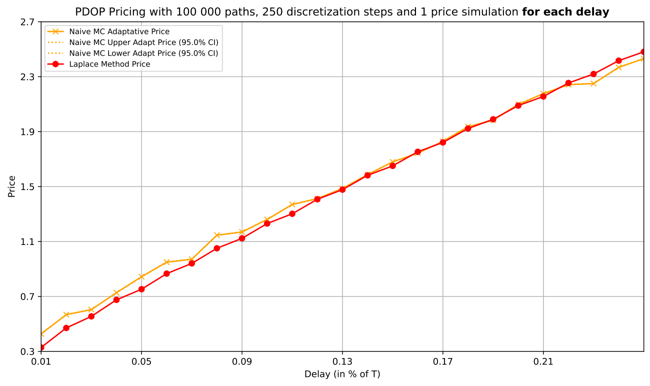


(b) One price simulation for each delay

Figure 3: PDOP pricing using the Naive Monte Carlo method and the Laplace transform method:: $S_0 = 100, L = 90, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 100\ 000, \text{discretization} = 250$



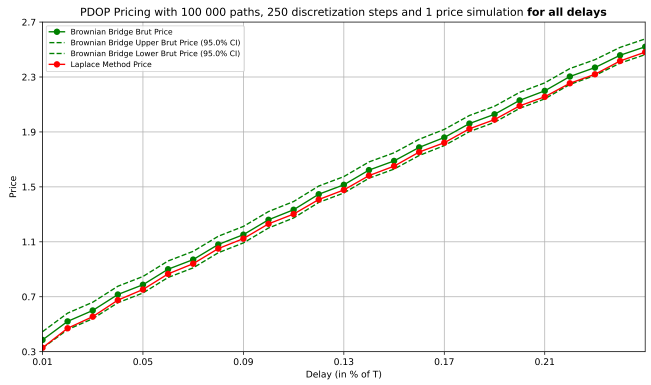
(a) One price simulation for all delays



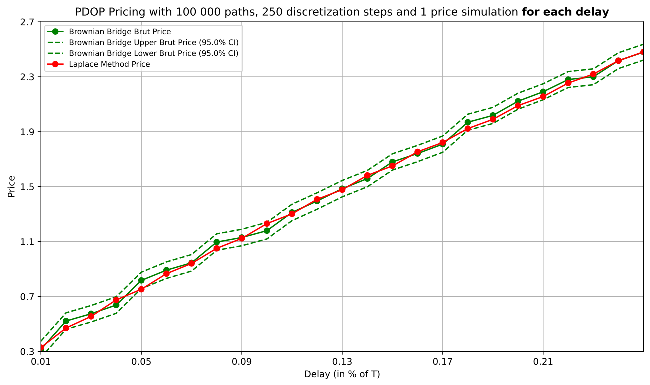
(b) One price simulation for each delay

Figure 4: PDOP pricing using the Naive Monte Carlo method, the Laplace transform method, and variance reduction: $S_0 = 100, L = 90, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 100\ 000, \text{discretization} = 250$

Appendix 3.4 Brownian bridge and Laplace Transform methods

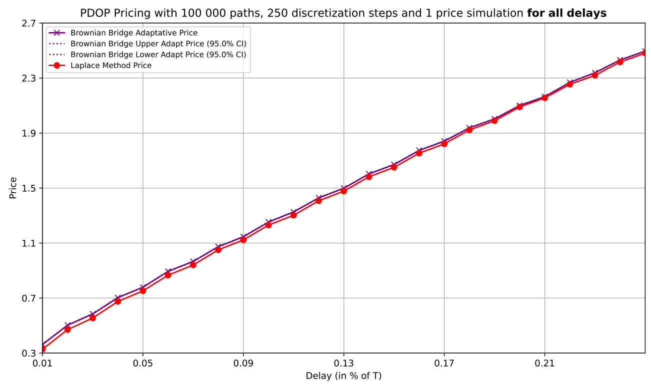


(a) One price simulation for all delays

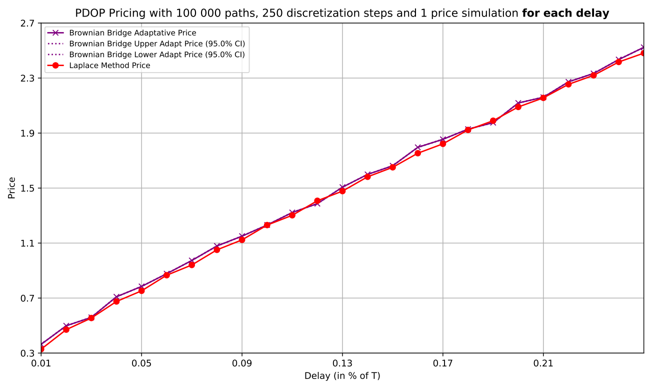


(b) One price simulation for each delay

Figure 5: PDOP pricing using the Brownian bridge method and the Laplace transform method: $S_0 = 100, L = 90, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 100\ 000, \text{discretization} = 250$



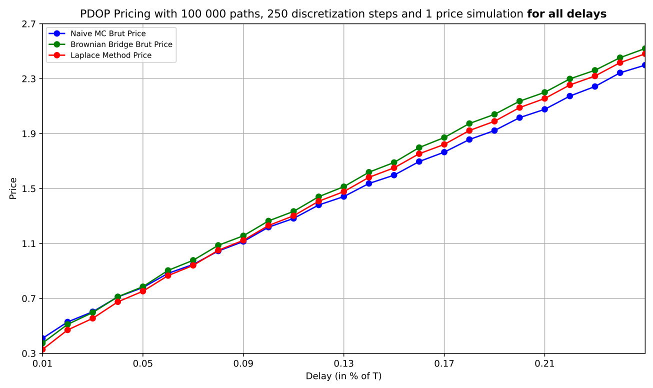
(a) One price simulation for all delays



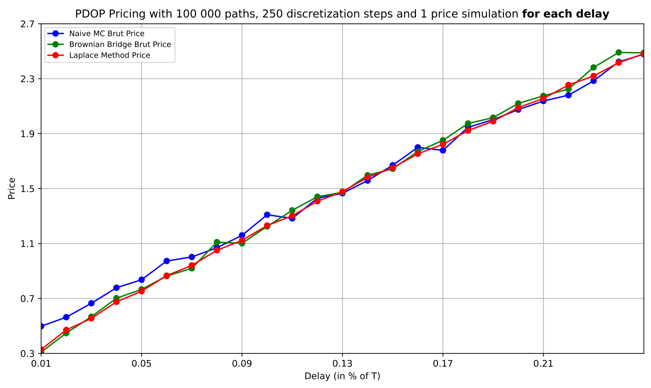
(b) One price simulation for each delay

Figure 6: PDOP pricing using the Brownian bridge method, the Laplace transform method, and variance reduction: $S_0 = 100, L = 90, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 100\ 000, \text{discretization} = 250$

Appendix 3.5 Comparison of the three methods without and with variance reduction

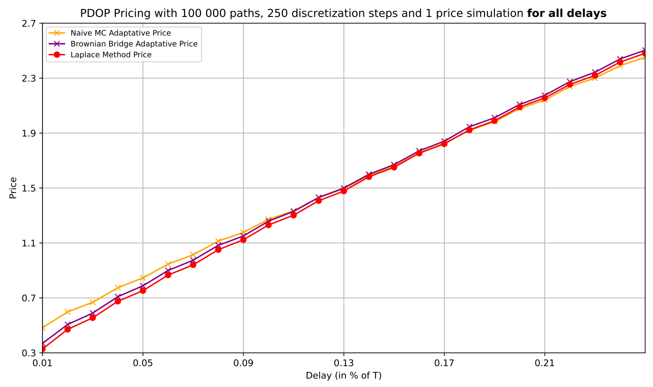


(a) One price simulation for all delays

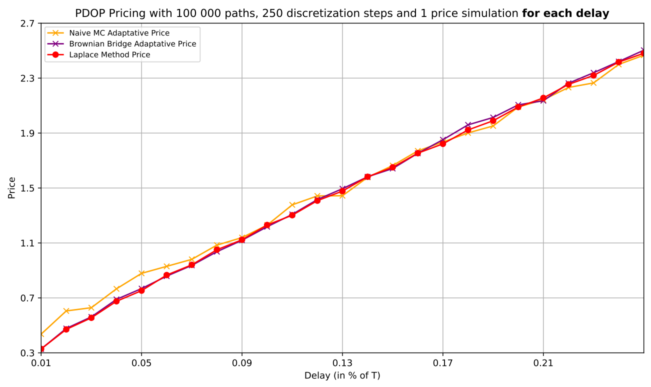


(b) One price simulation for each delay

Figure 7: PDOP Pricing with the three methods:
$S_0 = 100, L = 90, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 100\ 000, \text{discretization} = 250$



(a) One price simulation for all delays



(b) One price simulation for each delay

Figure 8: PDOP Pricing with the three methods and variance reduction:
$S_0 = 100, L = 90, T = 1, \sigma = 0.2, r = 0.025, \delta = 0, \text{paths} = 100\ 000, \text{discretization} = 250$