Pricing a European Option: Monte Carlo vs. Black–Scholes

Francesco Trombini

September 2025

Abstract

This paper addresses the pricing of a European option using two different approaches: the closed-form Black–Scholes formula and Monte Carlo simulation. We implement both methods under the assumption that the underlying asset follows a Geometric Brownian Motion. A series of numerical experiments is carried out by varying the number of simulations N, with the aim of analyzing the convergence of the Monte Carlo estimator to the theoretical benchmark. We evaluate absolute and relative errors, standard errors, and 95% confidence intervals. The results confirm that the Monte Carlo method converges to the Black–Scholes price and that the width of the confidence intervals shrinks as N increases, accordingly with the $1/\sqrt{N}$ convergence rate. We provide graphical evidence through convergence plots, price comparison figures, and payoff histograms, together with a summary table of the results. Finally, we discuss the advantages and limitations of the two approaches, showing that Black–Scholes is well-suited for standard vanilla contracts, while Monte Carlo is better suited for more complex payoffs, such as exotic and path-dependent derivatives.

Contents

1 Introduction

The valuation of derivative securities is a cornerstone of modern quantitative finance. Among the wide variety of derivatives, the European call and put options represent the simplest and most studied contracts, making them a natural benchmark for testing analytical and numerical pricing methods. Since the seminal work of Black and Scholes (1973), the closed-form solution for European options has become a standard reference, providing not only accurate prices under specific assumptions but also laying the foundation for much of option theory.

Alongside analytical formulas, numerical methods play a crucial role in option pricing. Among them, Monte Carlo (MC) simulation is one of the most widely used tools thanks to its generality, simplicity, and natural suitability for high-dimensional problems. While the Black–Scholes (BS) formula delivers an exact solution for plain vanilla European options, Monte Carlo is particularly appealing because it extends straightforwardly to exotic and path-dependent derivatives where closed-form solutions are not available.

The main purpose of this work is to compare the Monte Carlo estimator with the Black–Scholes formula in the context of European option pricing. By doing so, we aim to provide a clear and quantitative picture of the convergence properties of Monte Carlo, its statistical error, and its practical performance in relation to the analytical benchmark.

Contributions

The contributions of this paper are summarized as follows:

- We implement and compare the Black–Scholes closed-form formula and a Monte Carlo estimator for both call and put European options.
- We analyze convergence as the number of simulations N increases, studying absolute and relative errors, standard errors, and 95% confidence intervals.
- We present graphical evidence of convergence, distribution of payoffs, and a summary table of numerical results.
- We discuss the advantages and limitations of each method and provide insights on when Monte Carlo is preferable in practice.

Roadmap

The remainder of the paper is structured as follows. Section 2 introduces the theoretical background, recalling the dynamics of the Geometric Brownian Motion, the Black–Scholes formula, and the Monte Carlo estimator.

Section 3 presents the methodology and implementation details, including the experimental setup and pseudocode. Section 4 reports the experimental results and provides a quantitative comparison between Monte Carlo and Black–Scholes. Section 5 discusses the main findings, while Section 6 concludes the paper and outlines possible extensions.

2 Theoretical Background

2.1 Geometric Brownian Motion

We assume that the underlying asset price S_t follows a Geometric Brownian Motion (GBM) under the risk-neutral measure:

$$dS_t = rS_t\,dt + \sigma S_t\,dW_t,$$

where r is the risk-free interest rate, σ is the volatility, and W_t is a standard Wiener process.

The analytical solution at maturity T is given by:

$$S_T = S_0 \exp\Big((r - \tfrac{1}{2}\sigma^2)T + \sigma\sqrt{T}Z\Big), \qquad Z \sim \mathcal{N}(0,1).$$

2.2 Black–Scholes Formula

The closed-form solution for a European call option is:

$$C = S_0 N(d_1) - Ke^{-rT}N(d_2),$$

where

$$d_1 = \frac{\ln(S_0/K) + (r + \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}, \qquad d_2 = d_1 - \sigma\sqrt{T}.$$

The corresponding European put option price is:

$$P = Ke^{-rT}N(-d_2) - S_0 N(-d_1).$$

2.3 Monte Carlo Estimator

The Monte Carlo method approximates the price of an option by simulating a large number N of possible terminal asset prices $S_T^{(i)}$. For each simulation we compute the discounted payoff:

$$X^{(i)} = e^{-rT}\max(S_T^{(i)} - K, 0) \quad \text{(call)}, \qquad X^{(i)} = e^{-rT}\max(K - S_T^{(i)}, 0) \quad \text{(put)}.$$

The Monte Carlo estimator is then:

$$\widehat{V}_N = \frac{1}{N}\sum_{i=1}^{N} X^{(i)}.$$

Its statistical uncertainty can be quantified by the standard error (SE):

$$\mathrm{SE} = \frac{\widehat{\mathrm{sd}}(X)}{\sqrt{N}},$$

and the 95% confidence interval is:

$$\widehat{V}_N \pm 1.96 \cdot \mathrm{SE}.$$

2.4 Notation

Table 1 summarizes the notation used throughout the paper.

Symbol	Description
S_0	Initial asset price
K	Strike price
T	Time to maturity
r	Risk-free interest rate
σ	Volatility of the underlying asset
N	Number of Monte Carlo simulations
$N(\cdot)$	Cumulative distribution function of the standard normal

Table 1: Summary of notation used in the paper.

3 Methodology and Implementation

3.1 Baseline Parameters

The numerical experiments are conducted under a baseline scenario with the following parameters: initial stock price $S_0 = 100$, strike price $K = 100$, time to maturity $T = 1$ year, risk-free rate $r = 5\%$, and volatility $\sigma = 20\%$. These values are standard in the option pricing literature and allow us to generate results that are both interpretable and consistent with theoretical expectations.

3.2 Pipeline of the Analysis

The analysis proceeds through the following steps:

1. Compute the analytical Black–Scholes price for the call and put options.

2. Generate Monte Carlo estimates by simulating the terminal asset price S_T using normally distributed random draws.

3. For each simulation, compute the discounted payoff and the Monte Carlo estimator.

4. Repeat the simulation for different numbers of draws $N \in \{10^3, 2 \cdot 10^3, 3 \cdot 10^3, \ldots, 10^7\}$.

5. For each N, calculate the absolute and relative error with respect to the Black–Scholes benchmark, the standard error, and the 95% confidence interval.

6. Store the results in tables and visualize them using convergence plots, price comparison graphs, and payoff histograms.

3.3 Pseudocode of the Monte Carlo Estimator

The core of the Monte Carlo procedure can be summarized in the following pseudocode:

```
Input: S0, K, T, r, sigma, N, option_type
Generate N draws Z ~ Normal(0,1)
Compute terminal prices:
    S_T = S0 * exp((r - 0.5*sigma^2)*T + sigma*sqrt(T)*Z)
Compute payoff:
    If option_type = call: X = max(S_T - K, 0)
    If option_type = put:  X = max(K - S_T, 0)
Discounted mean payoff:
    MC_price = exp(-r*T) * mean(X)
Compute standard error:
    SE = exp(-r*T) * std(X)/sqrt(N)
Compute confidence interval:
    CI95 = [MC_price - 1.96*SE, MC_price + 1.96*SE]
Output: MC_price, SE, CI95
```

This procedure is repeated for each value of N, allowing us to study the convergence of the Monte Carlo estimates towards the Black–Scholes benchmark.

4 Experiments

4.1 Baseline Scenario

All experiments are carried out using the baseline parameters described in Section 3: $S_0 = 100$, $K = 100$, $T = 1$, $r = 0.05$, and $\sigma = 0.20$. The Black–Scholes formula provides the analytical benchmark, while the Monte Carlo estimator is computed for increasing numbers of simulations N ranging from 10^3 up to 10^7.

4.2 Performance Metrics

To evaluate the performance of the Monte Carlo method, we adopt the following metrics:

- **Absolute error:**
$$\text{AbsErr} = \left|\widehat{V}_N - V_{\text{BS}}\right|$$
where $\widehat{V}_N$ is the Monte Carlo estimate and V_{BS} the Black–Scholes benchmark.

- **Relative error:**
$$\text{RelErr} = \frac{\left|\widehat{V}_N - V_{\text{BS}}\right|}{V_{\text{BS}}}.$$

- **Standard error (SE)** of the Monte Carlo estimator, quantifying statistical uncertainty.

- **95% Confidence Interval (CI)** for the estimated price, obtained as:
$$\widehat{V}_N \pm 1.96 \cdot \text{SE}.$$

4.3 Stress Test (Optional)

In addition to the baseline, we may consider variations in key parameters to assess their impact on the convergence of Monte Carlo:

- *Volatility scenarios:* a low-volatility case $\sigma = 0.10$ and a high-volatility case $\sigma = 0.40$.

- *Maturity scenarios:* a short-term case $T = 0.25$ and a long-term case $T = 2$.

These stress tests provide further insights into how variance in the payoff distribution affects the efficiency of Monte Carlo simulation.

5 Results

5.1 Convergence Analysis

Figure 1 shows the absolute error of the Monte Carlo (MC) estimator compared to the Black–Scholes (BS) benchmark as a function of the number of simulations N, plotted on a log–log scale. The errors exhibit a downward trend as N increases, consistent with the theoretical convergence rate of order $1/\sqrt{N}$. Due to the inherent randomness of Monte Carlo simulation, the error does not decrease monotonically; rather, it oscillates around the expected convergence path. Nevertheless, the overall slope is close to the theoretical benchmark, confirming that the Monte Carlo method converges reliably to the Black–Scholes price as the number of simulations grows.

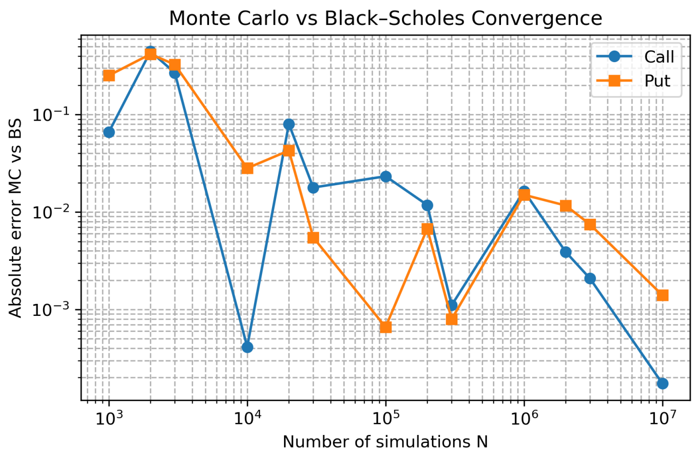

Figure 1: Convergence of Monte Carlo error compared to Black–Scholes.

5.2 Price Comparison

In addition to analyzing the error, it is instructive to observe the actual price estimates. Figure 2 plots the Monte Carlo estimates for the call and put options against the corresponding Black–Scholes benchmarks, across different N. The horizontal dashed lines represent the analytical BS prices. The figure illustrates how the Monte Carlo estimates oscillate around the true values and become increasingly accurate as N increases. This visualization provides an intuitive complement to the error analysis, directly showing the convergence of the estimates to the analytical benchmarks.

5.3 Distribution of Payoffs

Monte Carlo simulation also provides insight into the distribution of payoffs. Figure 3 reports the histogram of simulated payoffs for both call and put options using $N = 10^6$ simulations. The call distribution is skewed to the right, reflecting the unbounded upside potential of the underlying asset, while the put distribution is more concentrated near zero, consistent with the limited downside protection offered by the contract. This visualization helps to interpret the statistical properties of the Monte Carlo estimator and highlights why variance reduction techniques may be valuable in practice.

5.4 Summary Table

Table 2 and table 3 summarizes the numerical results obtained for different simulation sizes N. For each case, the table reports the Monte Carlo

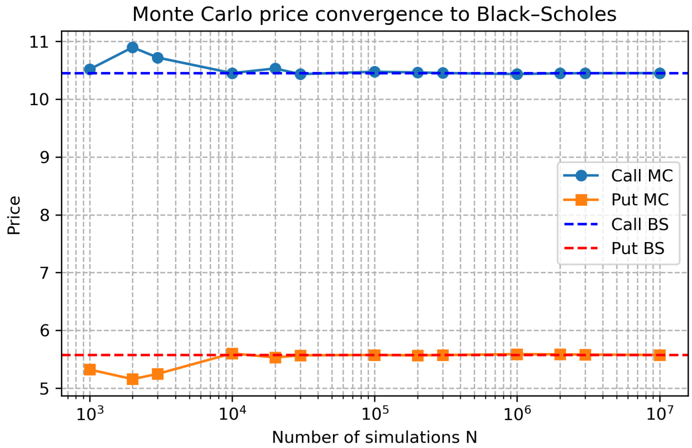

Figure 2: Convergence of Monte Carlo prices towards Black–Scholes benchmarks.

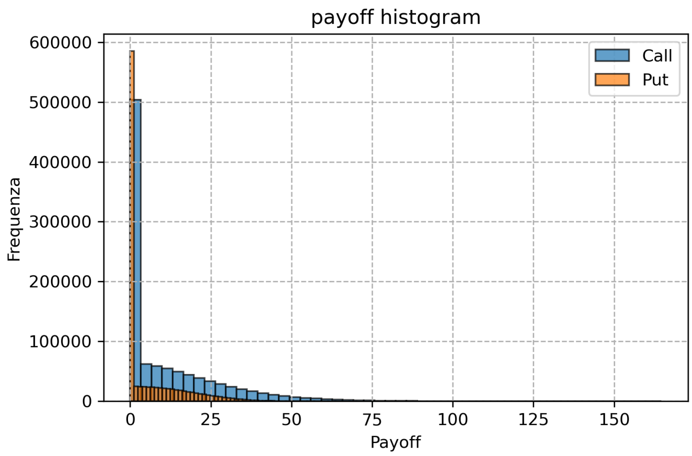

Figure 3: Distribution of simulated payoffs for call and put options ($N = 10^6$).

estimates, standard errors, confidence intervals, and absolute and relative errors compared to the Black–Scholes benchmarks. As expected, increasing N leads to narrower confidence intervals and smaller errors. For $N \geq 10^5$, the Monte Carlo estimates are already extremely close to the analytical prices, with relative errors below 0.1%. This confirms the efficiency of the method in approximating the Black–Scholes formula, albeit at a higher computational cost.

N	Call MC	Call SE	Call CI low	Call CI up	Call —err—	Call rel err
1000	10.516569	0.472961	9.589566	11.443572	0.065986	0.006314
2000	10.896039	0.337289	10.234952	11.557126	0.445456	0.042625
3000	10.719031	0.273481	10.183008	11.255054	0.268447	0.025687
10000	10.450170	0.147813	10.160457	10.739883	0.000414	0.000040
20000	10.530679	0.104895	10.325085	10.736272	0.080095	0.007664
30000	10.432731	0.084884	10.266358	10.599105	0.017852	0.001708
100000	10.473892	0.046589	10.382577	10.565207	0.023308	0.002230
200000	10.462392	0.032887	10.397934	10.526850	0.011809	0.001130
300000	10.451689	0.026844	10.399075	10.504303	0.001105	0.000106
1000000	10.434158	0.014700	10.405346	10.462969	0.016426	0.001572
2000000	10.446680	0.010408	10.426280	10.467080	0.003904	0.000374
3000000	10.448503	0.008496	10.431852	10.465155	0.002080	0.000199
10000000	10.450757	0.004654	10.441635	10.459880	0.000174	0.000017

Table 2: Monte Carlo vs Black–Scholes (Call options).

N	Put MC	Put SE	Put CI low	Put CI up	Put —err—	Put rel err
1000	5.321528	0.259992	4.811943	5.831113	0.251998	0.045213
2000	5.155901	0.185282	4.792747	5.519054	0.417625	0.074930
3000	5.247244	0.152079	4.949169	5.545320	0.326282	0.058541
10000	5.601717	0.086945	5.431305	5.772130	0.028191	0.005058
20000	5.530763	0.061097	5.411013	5.650514	0.042763	0.007672
30000	5.568031	0.049889	5.470249	5.665812	0.005495	0.000986
100000	5.574186	0.027382	5.520518	5.627853	0.000660	0.000118
200000	5.566778	0.019357	5.528839	5.604718	0.006748	0.001211
300000	5.572731	0.015805	5.541754	5.603708	0.000795	0.000143
1000000	5.588557	0.008668	5.571569	5.605546	0.015031	0.002697
2000000	5.585204	0.006130	5.573189	5.597220	0.011678	0.002095
3000000	5.580993	0.005003	5.571187	5.590800	0.007467	0.001340
10000000	5.574929	0.002738	5.569563	5.580295	0.001403	0.000252

Table 3: Monte Carlo vs Black–Scholes (Put options).

5.5 Key Observations

From the empirical results, the following points emerge:

- Monte Carlo estimates converge to the Black–Scholes price with a rate consistent with $1/\sqrt{N}$.
- Confidence intervals shrink as N increases, providing tighter bounds around the true price.

- The distribution of payoffs is highly skewed for calls and more concentrated for puts, explaining the variance structure of the estimator.
- While Black–Scholes delivers exact prices instantly, Monte Carlo requires a large N for high accuracy, underlining the trade-off between flexibility and efficiency.

6 Discussion

The results obtained in Section 5 provide clear evidence of the relative strengths and weaknesses of the Monte Carlo (MC) and Black–Scholes (BS) methods for option pricing. In this section we highlight the main insights that can be drawn from the numerical experiments.

6.1 Advantages of Monte Carlo

Monte Carlo simulation offers several important benefits:

- **Generality:** the method is not tied to a specific payoff structure and can be easily extended to exotic and path-dependent options (e.g., Asian, barrier, lookback).
- **Simplicity:** implementation only requires simulating the underlying asset under the risk-neutral measure and computing discounted payoffs.
- **Parallelization:** since simulations are independent, Monte Carlo is naturally parallelizable and benefits from modern multi-core CPUs and GPUs.

6.2 Limitations of Monte Carlo

At the same time, Monte Carlo has notable drawbacks:

- **Slow convergence:** accuracy improves only at the rate of $1/\sqrt{N}$, which implies that achieving one extra decimal digit of precision requires a hundredfold increase in simulations.
- **Computational cost:** large-scale simulations can be time consuming and memory intensive, particularly when high accuracy is required.
- **Variance sensitivity:** the distribution of payoffs, especially for calls with long maturities or high volatilities, can exhibit heavy skewness, which inflates the variance of the estimator.

6.3 When to Use Each Method

For plain vanilla European options, Black–Scholes is clearly the method of choice: it provides exact results instantly and does not require simulations. However, the assumptions behind the model (constant volatility, log-normal dynamics, no jumps) limit its applicability in more complex settings. Monte Carlo, while less efficient in this simple case, becomes indispensable for pricing derivatives where closed-form solutions are unavailable.

6.4 Variance Reduction Techniques

The efficiency of Monte Carlo can be improved through variance reduction strategies. Two widely used methods are:

- *Antithetic variates:* for each simulated standard normal Z, also simulate $-Z$, and average the results. This symmetry reduces the variance of the estimator without biasing the mean.
- *Control variates:* exploit the correlation between the payoff and a variable with known expectation (e.g., the terminal stock price S_T). Adjusting the estimator using this auxiliary information can significantly reduce variance.

Although variance reduction was not implemented in the experiments of this paper, it is worth noting that these techniques can substantially accelerate convergence and make Monte Carlo more competitive even for relatively simple payoffs.

7 Conclusions and Future Work

This paper has compared the pricing of European call and put options using the analytical Black–Scholes (BS) formula and the Monte Carlo (MC) simulation method. The analysis confirmed that Monte Carlo estimates converge towards the Black–Scholes benchmark at the expected statistical rate of $1/\sqrt{N}$. As the number of simulations increases, the absolute and relative errors diminish, and the confidence intervals shrink accordingly. For $N \geq 10^5$, the Monte Carlo prices were already very close to the analytical values, with relative errors below 0.1%, confirming the accuracy of the method.

The comparison highlighted a clear trade-off: Black–Scholes is unmatched in speed and precision for plain vanilla options, while Monte Carlo offers much greater flexibility and generality. In practice, Monte Carlo becomes the method of choice when dealing with exotic contracts, path-dependent structures, or models where closed-form solutions do not exist.

Several directions for future work naturally emerge from this study. First, the implementation of variance reduction techniques (such as antithetic variates, control variates, or importance sampling) would improve the efficiency of the Monte Carlo estimator. Second, quasi-Monte Carlo methods based on low-discrepancy sequences (e.g., Sobol or Halton) could accelerate convergence beyond the $1/\sqrt{N}$ rate. Finally, extending the framework to more realistic dynamics for the underlying asset, such as stochastic volatility (e.g., Heston model) or jump-diffusion processes, would provide further insights into the robustness of the two approaches.

Overall, this work illustrates both the elegance of the Black–Scholes formula and the versatility of Monte Carlo simulation, reinforcing their complementary roles in the modern toolkit of quantitative finance.

8 Reproducibility and Setup

All experiments presented in this paper were conducted on a personal laptop with the following specifications: Acer Nitro V 15, NVIDIA RTX 4060 GPU, 16 GB RAM, Windows 11 operating system.

The code was implemented in `Python 3.11` within an Anaconda environment (`torch311`). The following main libraries were used: `numpy`, `scipy`, `matplotlib`, and `pandas`. A fixed random seed was employed in the baseline experiments to guarantee reproducibility of the numerical results.

The project is organized into separate folders for clarity:

- `src` – source code, including the Black–Scholes formula, the Monte Carlo estimator, and the experiment runner.
- `data` – numerical results exported as `.csv`.
- `figures` – graphical outputs such as convergence plots and payoff histograms.
- `paper` – LaTeX source files and exported tables.

All experiments can be reproduced by cloning the repository available at:
`https://github.com/USERNAME/option-pricing-mc-vs-bs`

The baseline results can then be obtained by running:

```
python src/run_experiments.py
```

This script generates the tables and figures used in the paper and saves them in the corresponding folders. The provided `requirements.txt` file ensures that all necessary dependencies can be installed with:

```
pip install -r requirements.txt
```

9 References

References

[1] Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3), 637–654.

[2] Boyle, P. P. (1977). Options: A Monte Carlo approach. *Journal of Financial Economics*, 4(3), 323–338.

[3] Glasserman, P. (2004). *Monte Carlo Methods in Financial Engineering.* Springer, New York.

[4] Higham, D. J. (2004). An introduction to financial option valuation: mathematics, stochastics and computation. Cambridge University Press.

[5] Virtanen, P. et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17, 261–272.

A Appendix

A.1 Derivation of the Terminal Asset Price under GBM

Starting from the Geometric Brownian Motion (GBM) dynamics under the risk-neutral measure:

$$dS_t = rS_t\,dt + \sigma S_t\,dW_t,$$

we can apply Itô's Lemma to $\ln(S_t)$. This yields:

$$d\ln(S_t) = \Big(r - \tfrac{1}{2}\sigma^2\Big)dt + \sigma\,dW_t.$$

Integrating from 0 to T gives:

$$\ln(S_T) - \ln(S_0) = \Big(r - \tfrac{1}{2}\sigma^2\Big)T + \sigma W_T,$$

with $W_T \sim \mathcal{N}(0, T)$. Therefore,

$$S_T = S_0 \exp\Big((r - \tfrac{1}{2}\sigma^2)T + \sigma\sqrt{T}Z\Big), \qquad Z \sim \mathcal{N}(0, 1).$$

A.2 Standard Error and Confidence Intervals

Let $X^{(i)}$ denote the discounted payoff of the option in the i-th Monte Carlo replication, and let $\overline{X}_N$ be the sample mean:

$$\overline{X}_N = \frac{1}{N}\sum_{i=1}^{N} X^{(i)}.$$

The variance of the estimator is

$$\mathrm{Var}(\overline{X}_N) = \frac{\mathrm{Var}(X)}{N}.$$

Hence, the standard error is given by

$$\mathrm{SE} = \frac{\widehat{\mathrm{sd}}(X)}{\sqrt{N}}.$$

A 95% confidence interval for the option price is then

$$\widehat{V}_N \pm 1.96 \cdot \mathrm{SE}.$$

A.3 Extended Pseudocode

For completeness, we provide extended pseudocode for the experiment runner that executes both Black–Scholes and Monte Carlo pricing across multiple values of N:

```
Input: S0, K, T, r, sigma, list of N
Compute BS_call, BS_put

For each N in list:
    For option_type in {call, put}:
        Generate N draws Z ~ Normal(0,1)
        Compute terminal prices S_T
        Compute discounted payoffs X
        Compute MC_price = mean(X)
        Compute SE and CI
        Compute absolute and relative error w.r.t. BS
    Store results in table

Output: CSV with results, figures for convergence and histograms,
        LaTeX table for inclusion in the paper
```