Computational Finance - Lab Assignment 2

Contact: k.chatziandreou@uva.nl // Submit until 06.05.2025

1 Option Pricing — Realized Variance Swap and Carr–Madan [6p]

Consider an underlying asset price process $(S_t)_{t\geq 0}$ given by

$$dS_t = rS_t\,dt + \sigma_t S_t\,dB_t, \tag{1}$$

where $(B_t)_{t\geq 0}$ is a standard Brownian motion, and $(\sigma_t)_{t\geq 0}$ is an adapted stochastic volatility process. The riskless asset is priced as $A_t := e^{rt},\ t \in [0,T]$. We consider a realized variance swap with payoff given by the accumulated variance:

$$R^2_{0,T} = \int_0^T \sigma_t^2\,dt. \tag{2}$$

1. Show that the payoff $\int_0^T \sigma_t^2\,dt$ satisfies

$$\int_0^T \sigma_t^2\,dt = 2\int_0^T \frac{dS_t}{S_t} - 2\log\left(\frac{S_T}{S_0}\right). \tag{3}$$

2. Show that the price of the variance swap at time $t \in [0,T]$, given by

$$V_t := e^{-r(T-t)}\mathbb{E}_{\mathbb{Q}}\left[\int_0^T \sigma_t^2\,dt \mid \mathcal{F}_t\right], \tag{4}$$

satisfies the decomposition

$$V_t = L_t + 2(T-t)re^{-r(T-t)} + 2e^{-r(T-t)}\int_0^t \frac{dS_u}{S_u}, \tag{5}$$

where

$$L_t := -2e^{-r(T-t)}\mathbb{E}_{\mathbb{Q}}\left[\log\left(\frac{S_T}{S_0}\right) \mid \mathcal{F}_t\right] \tag{6}$$

is the price at time t of the log-contract with payoff $-2\log(S_T/S_0)$.

3. Show that the following portfolio constructed at time $t \in [0,T]$ hedges the realized variance swap:
   - One log-contract with value L_t.
   - $2e^{-r(T-t)}/S_t$ shares of the underlying asset S_t.
   - $2e^{-rT}\left(\int_0^t \frac{dS_u}{S_u} + (T-t)r - 1\right)$ invested in the riskless asset $A_t = e^{rt}$.

4. Show that the above portfolio is self-financing.

5. Consider the Heston model given by the stochastic differential equation

$$dV_t = \kappa(\theta - V_t)\,dt + \sigma\sqrt{V_t}\,dW_t, \tag{7}$$

where $\kappa, \theta, \sigma > 0$. Compute the expected realized variance over the interval $[0,T]$ under this model. *Hint:* Use Fubini's theorem:

$$\frac{1}{T}\mathbb{E}\left[\int_0^T V_t\,dt\right] = \frac{1}{T}\int_0^T \mathbb{E}[V_t]\,dt = \frac{1}{T}\int_0^T u(t)\,dt, \tag{8}$$

where $u(t) := \mathbb{E}[V_t]$. Derive an ODE for $u(t)$ and solve it.

6. Compute the variance swap rate

$$VS_T := \frac{1}{T}\mathbb{E}\left[\lim_{N\to\infty}\sum_{k=1}^{N}\left(\frac{S_{kT/N} - S_{(k-1)T/N}}{S_{(k-1)T/N}}\right)^2\right] = \frac{1}{T}\mathbb{E}\left[\int_0^T \frac{1}{S_t^2}(dS_t)^2\right], \tag{9}$$

on an index whose dynamics are given by the Heston-type model:

$$\begin{cases} dS_t = (r - \alpha V_t)S_t\,dt + S_t\sqrt{\beta + V_t}\,dB_t, \\ dV_t = \kappa(\theta - V_t)\,dt + \sigma\sqrt{V_t}\,dW_t, \end{cases} \tag{10}$$

where B_t and W_t are standard Brownian motions with correlation $\rho \in [-1,1]$. *Hint:* Use the previously derived expression for $u(t) := \mathbb{E}[V_t]$.

2 Variance Reduction: Asian Options under the Heston Model [6p]

Monte Carlo methods require a large number of simulations to achieve accurate estimates for option prices. Variance reduction techniques, such as **Control Variates**, are therefore crucial to enhance computational efficiency. In this assignment, you will price an arithmetic-average Asian call option under the **Heston stochastic volatility model** and apply the geometric-average Asian option under the BS model as a control variate.

2.1 Task 1: Monte Carlo Simulation under Heston Model

Implement a Monte Carlo simulator for the Heston model to price an arithmetic-average Asian call option. Follow these steps:

1. Simulate paths (S_t, v_t) under the Heston model using Euler and Milstein discretization with correlated Brownian increments. Ensure non-negativity for v_t.
2. Calculate the arithmetic average price along each path and the corresponding option payoff.
3. Compute the Monte Carlo estimator and its standard error using M simulated paths. Compare Euler versus Milstein scheme.
4. Verify your implementation by temporarily setting volatility-of-volatility $\xi = 0$ (reducing Heston to GBM) and comparing your results against known or approximated benchmarks.

2.2 Task 2: Analytical Geometric-Asian Price as Control Variate Reference

Implement the analytical pricing formula for the geometric-average Asian call under the Black–Scholes model:

1. Choose and justify a suitable volatility σ for the GBM control variate.
2. Calculate the geometric Asian option price analytically using the provided closed-form solution.
3. Test your implementation on known scenarios to ensure accuracy.

2.3 Task 3: Implementing the Control Variate Monte Carlo

Integrate the geometric Asian control variate into your Monte Carlo simulation:

1. Simulate parallel Black–Scholes (GBM) paths using identical Brownian increments from the Heston paths.
2. Compute arithmetic-average Asian payoffs (Heston) and geometric-average Asian payoffs (GBM) simultaneously.
3. Use the control variate estimator with initially choosing $c = 1$:

$$\hat{C}_{\text{CV}} = \bar{Y} + c\,(C_G - \bar{X}),$$

4. Compute variance and standard error of both the plain and control variate estimators, and compare their performance.

2.4 Task 4: Experiments and Comparative Analysis

(a) Variance Reduction Efficacy: Evaluate the performance of control variates by comparing the estimated prices and standard errors with and without control variates across various numbers of simulation paths (M). Summarize your findings clearly in tables or plots.

(b) Impact of Varying Heston Parameters: Perform sensitivity analyses by varying key parameters:

- Volatility-of-volatility ξ: Compare low ($\xi = 0.1$) and high ($\xi = 1.0$) scenarios.
- Correlation ρ: Evaluate negative correlation ($\rho = -0.9$) versus zero or positive correlation ($\rho = 0$ or 0.5).
- Strike K: Analyze in-the-money ($K < S_0$), at-the-money ($K \approx S_0$), and out-of-the-money ($K > S_0$) cases.

(c) Additional Analysis: Explore further aspects, such as the impact of averaging frequency, alternative control variates (!), or estimating the optimal control variate coefficient c^*.

3 The Temperature Derivative Market — Pricing of Asian Temperature Risk [6p]

The majority of futures and options written on temperature indices are traded on the CME (Chicago Mercantile Exchange), while a significant portion of the market outside these standardized indices operates over-the-counter (OTC). The standardized open market offers advantages such as reduced transaction costs and increased transparency but is largely limited to CME-listed temperature-based derivatives.
The CME marketplace provides futures and options based on Heating Degree Days (HDD), Cooling Degree Days (CDD), and the Cumulative Average Temperature (CAT) index, covering 19 cities worldwide and two contract seasons, which can vary depending on the specific contract. Despite their availability, daily trading volumes remain relatively low, with multiple days often passing without a single transaction. Nonetheless, in 2023, driven by increased climate volatility and interest in weather risk management, the CME reported a growing open interest in these contracts.
A weather derivative is a financial contract that facilitates the transfer of weather-related risk between parties seeking to hedge against such uncertainties. These contracts are typically structured around three core components:

- **Underlying meteorological parameter:** This refers to a physical atmospheric quantity monitored throughout the contract period to determine whether a payout is triggered. The most commonly used parameter is the average daily temperature, defined as the arithmetic mean of the daily maximum and minimum temperatures (i.e., the air temperature measured at two meters above the earth's surface).
- **Weather index:** A functional aggregation of the underlying meteorological parameter over the risk or contract period. Examples include cumulative temperature indices such as HDD and CDD.
- **Payoff function:** A predefined rule that links the weather index to the actual monetary payout, often through linear or piecewise-linear functions.

While nearly any quantifiable weather risk can be hedged using a weather derivative, provided there is a willing counterparty, certain industries have emerged as key users of such instruments:

- **Energy companies** use weather derivatives to hedge against revenue fluctuations due to mild weather, since consumer energy demand is highly sensitive to temperature variations.
- **Agricultural producers and transportation companies** rely on weather stability and use derivatives to mitigate the effects of temperature extremes or other unfavorable meteorological conditions.
- **Retail businesses** often experience seasonal sales patterns influenced by weather and use derivatives to manage such demand-side risks.
- **Leisure and tourism infrastructure** (e.g., hotels, amusement parks, and ski resorts) depend on predictable and favorable weather conditions for optimal operations.
- **Financial institutions** (e.g., investment banks, asset managers, insurers, and reinsurers) may include weather derivatives as part of broader portfolio risk management strategies.

Temperature-Based Weather Derivatives

Temperature derivatives are by far the most widely used class of weather derivatives. This popularity stems from the fact that temperature fluctuations significantly impact a broad range of economic sectors. The underlying parameter in these contracts is typically the average daily temperature, calculated as the mean of the daily maximum and minimum temperatures:

$$T_t := \frac{T_t^{\max} + T_t^{\min}}{2}. \tag{11}$$

This average temperature is then aggregated to compute widely used weather indices such as HDD (Heating Degree Days), CDD (Cooling Degree Days), and CAT (Cumulative Average Temperature), defined as:

$$HDD(\tau_1, \tau_2) := \int_{\tau_1}^{\tau_2} \max(c - T_u, 0)\, du,$$
$$CDD(\tau_1, \tau_2) := \int_{\tau_1}^{\tau_2} \max(T_u - c, 0)\, du,$$
$$CAT(\tau_1, \tau_2) := \int_{\tau_1}^{\tau_2} T_u\, du,$$

where c is a baseline reference temperature, typically 18°C, and T_u denotes the average temperature on day u. The integration period $[\tau_1, \tau_2]$ corresponds to:

- October to April for HDD contracts (heating season),
- April to October for CDD contracts (cooling season).

The CAT index measures the cumulative average temperature over the period $[\tau_1, \tau_2]$. It is used in place of the CDD index for a group of nine European cities: Amsterdam, Essen, Paris, Barcelona, London, Rome, Berlin, Madrid, Oslo, and Stockholm.
Using the identity

$$\max(T_u - c, 0) - \max(c - T_u, 0) = T_u - c,$$

one obtains the HDD–CDD parity:

$$CDD(\tau_1, \tau_2) - HDD(\tau_1, \tau_2) = CAT(\tau_1, \tau_2) - c(\tau_2 - \tau_1). \tag{12}$$

This identity implies that it is sufficient to analyze only the HDD and CAT indices, as CDD can be derived from them.

Objective of the Exercise This exercise is structured around the following goals:

- Retrieve historical weather data (e.g., daily 2-meter air temperature in Amsterdam).
- Construct a suitable stochastic model for temperature dynamics.
- Estimate and calibrate the model under the real-world measure $\mathbb{P}$.
- Use the calibrated model to price weather derivative options based on temperature indices.

3.1 Weather Data - Temperature

With the following code snippet you can download data for temperature data. For a full documentation you can also refer to the following guthub API page Open meteo github API page.
Forecast data:

```
import openmeteo_requests
import requests_cache
import pandas as pd
from retry_requests import retry

# Setup the Open-Meteo API client with cache and retry on error
cache_session = requests_cache.CachedSession('.cache', expire_after = 3600)
retry_session = retry(cache_session, retries = 5, backoff_factor = 0.2)
openmeteo = openmeteo_requests.Client(session = retry_session)

# Make sure all required weather variables are listed here
# The order of variables in hourly or daily is important to assign them correctly below
url = "https://historical-forecast-api.open-meteo.com/v1/forecast"
params = {
    # Amsterdam area (you can change )
        "latitude": 52.37,
        "longitude": 4.89,
        "start_date": "2020-08-10",
        "end_date": "2024-08-23",
        "hourly": "temperature_2m",
        "daily": "temperature_2m_mean"
}
responses = openmeteo.weather_api(url, params=params)

# Process first location. Add a for-loop for multiple locations or weather models
response = responses[0]
print(f"Elevation {response.Elevation()} m asl")
print(f"Timezone {response.Timezone()} {response.TimezoneAbbreviation()}")
print(f"Timezone difference to GMT+0 {response.UtcOffsetSeconds()} s")

# Process hourly data. The order of variables needs to be the same as requested.
hourly = response.Hourly()
```

```
hourly_temperature = hourly.Variables(0).ValuesAsNumpy()

hourly_data = {"date": pd.date_range(
        start = pd.to_datetime(hourly.Time(), unit = "s", utc = True),
        end = pd.to_datetime(hourly.TimeEnd(), unit = "s", utc = True),
        freq = pd.Timedelta(seconds = hourly.Interval()),
        inclusive = "left"
)}
hourly_data["temperature_2m"] = hourly_temperature

hourly_dataframe = pd.DataFrame(data = hourly_data)
print(hourly_dataframe)

# Process daily data. The order of variables needs to be the same as requested.
daily = response.Daily()
daily_temperature = daily.Variables(0).ValuesAsNumpy()

daily_data = {"date": pd.date_range(
        start = pd.to_datetime(daily.Time(), unit = "s", utc = True),
        end = pd.to_datetime(daily.TimeEnd(), unit = "s", utc = True),
        freq = pd.Timedelta(seconds = daily.Interval()),
        inclusive = "left"
)}
daily_data["temperature_2m_mean"] = daily_temperature

daily_dataframe = pd.DataFrame(data = daily_data)
print(daily_dataframe)
```

We are only be using the daily temperature data for this exercise.

3.2 Stochastic Modeling of Temperature dynamics

Let us now discuss the modeling of the temperature dynamics. We will consider the modeling of this stochastic process using a stochastic differential equation. This differential equation essential describes the evolution of the temperature through time but also taking into account this random fluctuations that the temperature dynamics have in real life. We want these dynamics to capture to main characteristics of temperature data. Namely the mean-reverting property which essentially will describe the speed of mean-reversion of the temperature to its seasonal value and also the seasonality function $S(t)$ which for now will be denoted by $\bar{\mu}(t)$ and will describe the seasonal (deterministic) evolution of the temperature. To do so we consider the following stochastic differential equation which models the dynamics of the temperatures.

$$dT_t = \kappa\left(\bar{\mu}(t) - T_t\right)dt + \sigma(t)dW_t \tag{13}$$

where $W = (W_t)_{t\geq 0}$ is the driving "noise" which is taken to be a Winer process; α is the speed of mean-reversion and $\bar{\mu}(t)$ is the seasonal mean and $\sigma(t)$ the volatility of the process (which can also be a function of time) but we have to ensure that it is positive. You can just assume that it is a positive constant. To capture fully the mean-reverting dynamics of the temperature it is important to ensure that

$$\mathbb{E}\left[T_t\right] = \bar{\mu}(t) - \int_s^t e^{-\kappa(t-k)}dW_k \tag{14}$$

The solution of this SDE can also be found using Ito's formula. Regarding the deterministic seasonal mean $\bar{\mu}(t)$, having a closer look at the "denoised" temperature time series it reveals that temperatures have uniform peaks and very weak rising trends. This inspires us to model the trend with a linear function and the seasonality with a single single sine function. Thus the determistic seasonal temperature which we have denoted by $\bar{\mu}(t)$ is modeled by

$$\bar{\mu}(t) = \alpha + \beta t + \gamma\sin\left(\omega t + \phi\right) \tag{15}$$

where the period $\omega = \frac{2\pi}{365}$ and the phase shift ϕ captures the fact that the minimum and maximum temperatures do not necessarily occur on the first of January or the first of July respectively. Using historical data we can estimate these parameters by employing a least-squares method. This can be done by observing that the equation (15) can be rewritten as

$$\bar{\mu}(t) = \alpha_1 + \alpha_2 + \alpha_3 sin(\omega t) + \alpha_4 cos(\omega t) \tag{16}$$

we find the vector $\xi = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ that minimizes the estiamtion errors. Then

$$A = \alpha_1, \quad B = \alpha_2, \quad C = \sqrt{\alpha_3^2 + \alpha_4^2}, \quad \phi = tan^{-1}\left(\frac{\alpha_4}{\alpha_3}\right) - \pi$$

We refer to the following paper for some further analysis on this and some reasonable values of the parameter sets for the numerical simulations (Temperature modeling reference).
To address the issue that $\mathbb{E}(T_t) \neq \bar{\mu}(t)$ we can define

$$dT_t = \left(\frac{d\bar{\mu}(t)}{dt} + \kappa\left(\bar{\mu}(t) - T_t\right)\right)dt + \sigma(t)dW_t \tag{17}$$

We could use Ito's formula, but it turns out that we can now solved this SDE using the traditional integrating factor method. Multiplying through by $e^{\int_0^t \kappa du}$ we obtain, where the left hand side of the above expression is just the differential of a product:

$$e^{\int_0^t \kappa du} d\bar{\mu(u)} - e^{\int_0^t \kappa du}\kappa(\bar{\mu(u)} - T_u)du + e^{\int_0^t \kappa du} dT_u = e^{\int_0^t \kappa du}\sigma_t dW_u \tag{18}$$

$$d\left[e^{\int_0^t \kappa du}(\bar{\mu(u)} - T_u)\right] = e^{\int_0^t \kappa du}\sigma_t dW_u \tag{19}$$

If we consider ito process $Z_t = e^{\int_0^t \kappa du}(\bar{\mu(u)} - T_u)$
Then it's dynamics are as the equation above $dZ_t = d\left[e^{\int_0^t \kappa du}(\bar{\mu(u)} - T_u)\right] = e^{\int_0^t \kappa du}\sigma_t dW_u$

$$Z_t = Z_0 - \int_0^t e^{\int_0^t \kappa du}\sigma_t dW_u \tag{20}$$

Now if substituting $Z_t = e^{\int_0^t \kappa du}(\bar{\mu(u)} - T_u)$ into the equation above with $\bar{\mu_0} = T_0$:

$$e^{\int_0^t \kappa du}(\bar{\mu(t)} - T_t) = e^{\int_0^t \kappa du}(\bar{\mu_0} - T_0) - \int_0^t e^{\int_0^t \kappa du}\sigma_t dW_u \tag{21}$$

Finally rearranging we get:

$$T_t = \bar{\mu(t)} + e^{-\int_0^t \kappa du}\int_0^t e^{\int_0^t \kappa du}\sigma_t dW_u \tag{22}$$

This implies immediately that $\mathbb{E}[T_t] = \bar{\mu(t)}$.

3.3 Estimate and calibrate the model under the real-world measure $\mathbb{P}$

Objective: Calibrate a deterministic seasonal model for temperature and fit an autoregressive (AR) process to the residuals.

Task 1: Deterministic Model Fitting

1. Specify the model:
$$T(t) = a + b\,t + \alpha \sin\left(\frac{2\pi}{365.25}t + \theta\right)$$
2. Use `scipy.optimize.curve_fit` to estimate the parameters a, b, α, and θ.
3. Compute the model fit and extract residuals.

Task 2: Autoregressive Modeling of Residuals

1. Fit an AR model to the residuals.
2. Determine the order of the AR process using information criteria (e.g., AIC).
3. Estimate the mean-reversion parameter κ.
4. Estimate the seasonal volatility $\sigma(t)$ using Fourier series.

Example Code Snippet

```
import numpy as np
from scipy.optimize import curve_fit

% Define the deterministic model function
def model_fit(x, a, b, a1, b1):
    omega = 2*np.pi/365.25
    return a + b*x + a1*np.cos(omega*x) + b1*np.sin(omega*x)

% Convert dates to ordinal numbers and fit model
first_ord = daily_dataframe.index[0].toordinal()
xdata = np.array([date.toordinal() - first_ord for date in daily_dataframe.index])
ydata = daily_dataframe['temperature_2m_mean']

params_all, cov = curve_fit(model_fit, xdata, ydata, method='lm')
daily_dataframe['model'] = model_fit(xdata, *params_all)
daily_dataframe['residuals'] = daily_dataframe['temperature_2m_mean'] - daily_dataframe['model

print("Estimated parameters:", params_all)
```

The question now is how can we estimate our mean reversion parameter. Recall that we have the following SDE for the temperature dynamics:

$$dT_t = \left[\frac{d\bar{\mu}(t)}{dt} + \kappa(\bar{\mu}(t) - T_t)\right] dt + \sigma_t dW_t$$

First, consider the Euler Discretization of our SDE over the interval between $t \in [i-1, i]$
Euler Discretization of our SDE over the interval between $t \in [i-1, i]$

$$T_i - T_{i-1} = \bar{\mu}_i - \bar{\mu}_{i-1} + \kappa(\bar{\mu}_{i-1} - T_{i-1}) + \sigma_i z_i$$

$$T_i - \bar{\mu}_i = T_{i-1} + \bar{\mu}_{i-i} - \kappa(T_{i-1} - \bar{\mu}_{i-1}) + \sigma_i z_i$$

Where $z_t \sim N(0,1)$. Now let's create a term for when we de-trend and remove seasonality from our original DAT series. Let's call this transformation $\hat{T}_t = T_t - \bar{\mu}(t)$, therefore we get:

$$\hat{T}_i = \hat{T}_{i-1} - \kappa(\hat{T}_{i-1}) + \sigma_i z_i$$

This can be modelled as an AR(1) process:

$$\hat{T}_i = \gamma \hat{T}_{i-1} + e_i$$

where $\kappa = 1 - \gamma$, and $e_i = \sigma_i z_i$. This is a linear approach to estimating the speed of mean reversion. Now fit the AR(1) model to the transformed DAT series $\hat{T}_i$ (de-trended and removed seasonality).
Temperature volatility modeling
We will also follow a semi-parametric approach to recover the volatility $\sigma(t)$ function, by means of Fourier series. In our implementation we will consider that the temperature volatility is just a deterministic function of time. You will model the volatility by

$$\sigma(t) = V + Ut + \sum_{i=1}^{I} c_i \sin(i\omega t) + \sum_{j=1}^{J} d_j \cos(j\omega t) \tag{23}$$

with $V + Ut$ representing a linear term to capture the trend in the annual volatility (i.e the fact that there is higher volatility in the winter months than in summer).

3.4 Pricing of Weather Derivatives

We will now turn our attention to the pricing of HDD and CDD contracts. Recall that the reference temperature for weather derivatives contracts traded at the CME is 18^oC and as a result the payoff function for a degree day contract is defined as

$$\mathcal{C} = N \max\{H_n - K, 0\} \tag{24}$$

where N is the tick size (notional value), K the strike level and n is the number of days in the contract period and H_n the number of heating degree days in the contract period

$$H_n = \sum_{i=1}^{n} \max\{18 - T_t, 0\} \tag{25}$$

from the temperature time series you can clearly see that for the winter months $\mathbb{P}(\max\{18 - T_t, 0\} =) \approx 0$. Hence one can also use a variance reduction method in his estimator.
The goal is to use MC methods to compute the price at time t of an HDD call option in a time space of length n, assuming a constant interest rate r, using the Estimators

$$C_{HDD}(t) = e^{-r(t_n - t)} \frac{1}{M} \sum_{i=1}^{M} N\mathbb{E}\left[\max\{H_n^i - K, 0\}\right] \tag{26}$$

where M the number of MC paths generated.

A Assignment 1

Realized Variance Swaps - Another look at historical volatility We have already seen in the previous assignment some of the possible estimators one can construct for realized variance. Now our goal is to study realized variance swaps.
(Realized) variance swaps are forward contracts that allow for the exchange of the estimated volatility against a fixed value κ_σ. These constracts a re typically written on the realized variance over a given period of time, i.e have a terminal payoff of

$$\frac{1}{T}\left[\sum_{k=1}^{N}\left(\log\frac{S_{t_k}}{S_{t_{k-1}}}\right)^2 - \frac{1}{N-1}\left(\frac{S_T}{S_0}\right)^2\right] - \kappa_\sigma^2 \tag{27}$$

and they can be priced using the expected value of of their payoff

$$\frac{1}{T}\mathbb{E}_{\mathbb{Q}}\left[\sum_{k=1}^{N}\left(\log\frac{S_{t_k}}{S_{t_{k-1}}}\right)^2 - \frac{1}{N-1}\left(\frac{S_T}{S_0}\right)^2\right] - \kappa_\sigma^2 \tag{28}$$

where κ_σ is the volaitlity level. Note that the above payoff has to be multiplied by the Vega notional [1], which is part of the contract, in order to convert it into currency units. Being a swap the strike is choosen such that the contract haw no value at inception, i.e $\kappa_\sigma^2 = \frac{1}{T}\mathbb{E}_{\mathbb{Q}}\left[\sum_{k=1}^{N}\left(\log\frac{S_{t_k}}{S_{t_{k-1}}}\right)^2 - \frac{1}{N-1}\left(\frac{S_T}{S_0}\right)^2\right]$.
We can equivalently calculate the realized variance as follows:

$$\sigma_R^2 := A\sum_{k=1}^{N}\frac{1}{N}\log\left(\frac{S_{t_k}}{S_{t_{k-1}}}\right)^2 \tag{29}$$

where A is an annualization factor (usually 252 working days), $t_0 < t_1 < \dots t_n$ are sampling dates specified in the contract and (S_t) denotes the stock price process under consideration. In practice however, pricing of variance swaps, namely computing the expectation of the realized variance, is performed by approximating the discrete sampling above, by its continuous version.
More precisely take a partition $0 = t_0 < t_1 < \cdots < t_n = T$, then the following limits holds in probability:

$$\lim_{n\to+\infty}\sum_{k=1}^{N}\log\left(\frac{S_{t_k}}{S_{t_{k-1}}}\right)^2 = \langle\log(S), \log(S)\rangle_T \tag{30}$$

Computing the quadratic variation on the right is an easier exercise using the tools from stochastic Calculus and generally yields to simple closed-form expressions.

[1] Vega notional represents the expected profit and loss of a variance swap for a 1% change in volatility from the strike price. It helps traders assess the magnitude of the trade. For example, with a Vega notional of Eur10,000, a one-point difference in volatility would result in a profit or loss close to Eur10,000.

B Assignment 2

Asian options are special cased of average value options, whose payoffs are determined by the difference between the average underlying asset price over a certain time interval and a strike price K. Due to their dependence on averaged asset prices; Asian options are less volatile than plain vanilla options whose claim payoffs depend only on the terminal value of the underlying asset. Asian options were first traded in Tokyo in 1987, and have become particularly popular in commodities trading.

Arithmetic Asian Options: Given an underlying asset price process $(S_t)_{t\in[0,T]}$, the payoff of the Asian call option on S with exercise date T and strike price K is given by

$$C = \left(\frac{1}{T}\int_0^T S_t dt - K\right)^+ \tag{31}$$

Similarly the payoff of the Asian put option on S

$$C = \left(K - \frac{1}{T}\int_0^T S_t dt\right)^+ \tag{32}$$

Due to their dependence on averaged asset prices, Asian options are less volatile than plain vanilla options whose payoffs depend only on the terminal value of the underlying asset. Related exotic options include the Asian-American options, or Hawaiian options, that combine an Asian claim payoff with American style exercise. An option on average is an option whose payoff has the form

$$C = \phi(\Lambda_T, S_T) \tag{33}$$

where

$$\Lambda_T = S_0 \int_0^T e^{\sigma B_u + ru - \sigma^2/2udu} = \int_0^T S_u du \tag{34}$$

For example when $\phi(y,x) = (y/T - K)^+$ this yields the Asian call option with payoff

$$\left(\frac{1}{T}\int_0^T S_u du - K\right)^+ = \left(\frac{\Lambda_T}{T} - K\right)^+ \tag{35}$$

which is path-depend option whose price at time $t \in [0,T]$ is given by

$$e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\left(\frac{1}{T}\int_0^T S_u du - K\right)^+ \mid \mathcal{F}_t\right] \tag{36}$$

Throughout we will assume that the underlying asset price process S_t is a geometric Brownian motion satisfying

$$dS_t = rS_t dt + \sigma S_t dB_t \tag{37}$$

where B_t is a standard Brownian motion under the risk-neutral probability measure $\mathbb{Q}$. Using the time homogeneity of the process S (we assume constant coefficients in the SDE), the option payoff $C = \phi(\Lambda_T, S_T)$ can be priced as

$$\begin{aligned}
e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(\Lambda_T, S_T) \mid \mathcal{F}_t\right] &= e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(\Lambda_t + \int_t^T S_u du, S_T) \mid \mathcal{F}_t\right] \\
&= e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(y + x\int_t^T \frac{S_u}{S_t} du, x\frac{S_T}{S_t}) \mid y = \Lambda_t, x = S_t\right] \\
&= e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(y + x\int_0^{T-t} \frac{S_u}{S_0} du, x\frac{S_{T-t}}{S_0}) \mid y = \Lambda_t, x = S_t\right]
\end{aligned}$$

Using the Markov property of the process (S_t, Λ_t) we can write down the option process as a function

$$\begin{aligned}
f(t, S_t, \Lambda_t) &= e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(\Lambda_T, S_T) \mid \mathcal{F}_t\right] \\
&= e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(\Lambda_T, S_T) \mid S_t, \Lambda_t\right]
\end{aligned}$$

of (t, S_t, Λ_t) where the function $f(t,x,y)$ is given by

$$f(t,x,y) = e^{-(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\phi(y + x\int_0^{T-t}\frac{S_u}{S_0}du, x\frac{S_{T-t}}{S_0}) \mid y = \Lambda_t, x = S_t\right]$$

There exists no easily tractable **closed-form** solution for the price of an arithmetically averaged Asian option.
Geometric Asian options On the other hand, replacing the arithmetic average

$$\frac{1}{T}\sum_{k=1}^{n} S_{t_k}\left(t_k - t_{k-1}\right) \approx \frac{1}{T}\int_0^T S_u du \tag{38}$$

with the **geometric** average

$$\begin{aligned}
\prod_{k=1}^{n} S_{t_k}^{(t_k - t_{k-1})/T} &= \exp\left(\log\prod_{k=1}^{n} S_{t_k}^{(t_k - t_{k-1})/T}\right)\\
&= \exp\left(\frac{1}{T}\sum_{k=1}^{n}\log S_{t_k}^{t_k - t_{k-1}}\right)\\
&= \exp\left(\frac{1}{T}\sum_{k=1}^{n}(t_k - t_{k-1})\log S_{t_k}\right)\\
&\approx \exp\left(\frac{1}{T}\int_0^T \log S_u du\right)
\end{aligned}$$

leads to closed form solutions using the Black Scholes formula. We want to compute the price

$$e^{(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\left(\exp\left(\frac{1}{T}\int_0^T \log S_u du - K\right)\right)^+ \mid \mathcal{F}_t\right] \tag{39}$$

at time t of the geometric Asian option with maturity T, where $S_t = S_0 e^{rt + \sigma B_t - \frac{\sigma^2}{2}t}$.
We let

$$\begin{aligned}
G_T = \exp\left(\frac{1}{T}\int_0^T \log S_u du\right) &= \exp\left(\frac{1}{T}\int_0^t \log S_u du + \frac{1}{T}\int_t^T \log S_u du\right)\\
&= \exp\left(\frac{1}{T}\int_0^t \log S_u + \frac{T-t}{T}\log S_t + \frac{1}{T}\int_t^T \log\frac{S_u}{S_t} du\right)\\
&= \exp\left(\frac{1}{T}\int_0^t \log S_u du + \frac{T-t}{T}\log S_t + \frac{1}{T}\int_t^t \left(r(u-t) + (B_u - B_t)\,\sigma - (u-t)\sigma^2/2\right) du\right)\\
&= \exp\left(\frac{1}{T}\int_0^t \log S_u du + \frac{T-t}{T}\log S_t + \frac{1}{T}\int_0^{T-t} \left(ru - \sigma^2 u/2\right) du + \frac{\sigma}{T}\int_t^T (B_u - B_t)\, du\right)\\
&= (S_t)^{(T-t)/T}\exp\left(\frac{1}{T}\int_0^t \log S_u + \frac{(T-t)^2}{2T}\left(r - \sigma^2/2\right) + \frac{\sigma}{T}\int_t^t (B_u - B_t)\, du\right)
\end{aligned}$$

where $\int_t^T B_u du$ is centered Gaussian with conditional variance

$$\begin{aligned}
\mathbb{E}\left[\left(\int_t^T B_u du\right)^2 \mid \mathcal{F}_t\right] &= \mathbb{E}\left[\left(\int_t^T (B_u - B_t)\, du\right)^2 \mid \mathcal{F}_t\right]\\
&= \mathbb{E}\left[\left(\int_t^T (B_u - B_t)\, du\right)^2\right]\\
&= \mathbb{E}\left[\left(\int_0^{T-t} (B_u - B_t)\, du\right)^2\right]\\
&= \int_0^{T-t}\int_0^{T-t} \mathbb{E}\left[B_s B_u\right] ds du\\
&= 2\int_0^{T-t}\int_0^{u} s\, ds du = \int_0^{T-t} u^2 du = \frac{(T-t)^3}{3}
\end{aligned}$$

Hence letting

$$m := \frac{1}{T}\int_0^t \log S_u du + \frac{T-t}{T}\log S_t + \frac{(T-t)^2}{2T}\left(r-\sigma^2/2\right), \quad X := \frac{\sigma}{T}\int_t^T B_u du$$

and $\nu^2 = (T-t)\,\sigma^2/3$ we find that

$$\begin{aligned}
&e^{(T-t)r}\mathbb{E}_{\mathbb{Q}}\left[\left(\exp\left(\frac{1}{T}\int_0^T \log S_u du - K\right)\right)^+ \mid \mathcal{F}_t\right] = \\
&= (S_t)^{(T-t)/T}\, e^{-(T-t)r}\exp\left(\frac{1}{T}\int_0^t \log S_u du + \frac{(T-t)^2}{4T}\left(2r-\sigma^2\right) + \frac{\sigma^2}{6}(T-t)\right) \\
&\times \Phi\left(\frac{(T-t)\sigma^2/3 + \frac{1}{T}\int_0^t \log S_u du + \log\frac{S_t^{(T-t)/T}}{K} + \frac{(T-t)^2}{2T}\left(r-\sigma^2/2\right)}{\sigma\sqrt{(T-t)/3}}\right) \\
&- Ke^{-(T-t)r}\Phi\left(\frac{\frac{1}{T}\int_0^t \log S_u du + \log\frac{S_t^{(T-t)/T}}{K} + \frac{(T-t)^2}{2T}\left(r-\sigma^2/2\right)}{\sigma\sqrt{(T-t)/3}}\right)
\end{aligned}$$

for $0 \le t \le T$. In case $t = 0$ we can find the price of the call option price on geometric averages.

$$\begin{aligned}
C &= e^{-rT}\mathbb{E}_{\mathbb{Q}}\left[\left(\exp\left(\frac{1}{T}\int_0^T \log S_u du - K\right)\right)^+ \mid \mathcal{F}_0\right] \\
&= S_0 e^{-T(r+\sigma^2/6)/2}\Phi\left(\frac{\log(S_0/K) + T\left(r+\sigma^2/6\right)/2}{\sigma\sqrt{T/3}}\right) \\
&- Ke^{-rT}\Phi\left(\frac{\log(S_0/K) + T\left(r-\sigma^2/6\right)/2}{\sigma\sqrt{T/3}}\right)
\end{aligned}$$

We could get the same expression if we consider the calculations immediately to the geometric averages $\prod_{k=1}^N S_{t_k}^{(t_k-t_{k-1})/T}$. Then for $\tilde{\sigma} = \sigma\sqrt{\frac{2N+1}{6(N+1)}}$

$$\begin{aligned}
C &= e^{-rT}\mathbb{E}_{\mathbb{Q}}\left[\left(\exp\left(\prod_{k=1}^N S_{t_k}^{(t_k-t_{k-1})/T} - K\right)\right)^+ \mid \mathcal{F}_0\right] \\
&= S_0 e^{(\tilde{r}-r)T}\Phi\left(\tilde{d}_1\right) \\
&- Ke^{-rT}\Phi\left(\tilde{d}_2\right)
\end{aligned}$$

where

$$\begin{aligned}
\tilde{d}_1 &= \frac{\log\frac{S_0}{K} + \left(\tilde{r} + \frac{1}{2}\tilde{\sigma}^2\right)T}{\sqrt{T}\tilde{\sigma}} \\
\tilde{d}_2 &= \frac{\log\frac{S_0}{K} + \left(\tilde{r} - \frac{1}{2}\tilde{\sigma}^2\right)T}{\sqrt{T}\tilde{\sigma}}
\end{aligned}$$

where

$$\begin{aligned}
\tilde{\sigma} &= \tilde{\sigma} = \sigma\sqrt{\frac{2N+1}{6(N+1)}} \\
\tilde{r} &= \frac{(r - \frac{1}{2}\sigma^2) + \tilde{\sigma}^2}{2}
\end{aligned}$$

C Assignment 3

In this exercise you will implement a complete workflow from retrieving weather data to pricing weather derivatives based on temperature dynamics. The exercise is structured into four main tasks:

- **Data Retrieval:** Obtain daily temperature data (2m) for Amsterdam using the Open-Meteo API.
- **Stochastic Modeling:** Construct a model for temperature dynamics that captures both trend and seasonality.
- **Model Fitting under $\mathbb{P}$:** Calibrate the deterministic trend and seasonal model along with an autoregressive process to model the residuals.
- **Option Pricing:** Price weather derivatives (options) using Monte Carlo simulations based on the estimated model.

Part A: Data Retrieval

Objective: Retrieve daily mean temperature data from the Open-Meteo API for Amsterdam (latitude: 52.37, longitude: 4.89) over a given period.

Instructions

1. Use the Open-Meteo API endpoint (e.g., `https://historical-forecast-api.open-meteo.com/v1/forecast`) with the specified parameters.
2. Retrieve both hourly and daily data; for this exercise, focus on the `temperature_2m_mean` variable available in the daily data.
3. Process the JSON response to extract and format the data into a `pandas.DataFrame`.
4. Handle missing data appropriately (e.g., using interpolation).

Example Code Snippet

```
import openmeteo_requests
import requests_cache
import pandas as pd
from retry_requests import retry

# Setup the API client with cache and retry
cache_session = requests_cache.CachedSession('.cache', expire_after=3600)
retry_session = retry(cache_session, retries=5, backoff_factor=0.2)
openmeteo = openmeteo_requests.Client(session=retry_session)

url = "https://historical-forecast-api.open-meteo.com/v1/forecast"
params = {
    "latitude": 52.37,
    "longitude": 4.89,
    "start_date": "2020-08-10",
    "end_date": "2024-08-23",
    "hourly": "temperature_2m",
    "daily": "temperature_2m_mean"
}
responses = openmeteo.weather_api(url, params=params)
response = responses[0]

# Process daily data
daily = response.Daily()
daily_temperatures = daily.Variables(0).ValuesAsNumpy()

daily_data = {
    "date": pd.date_range(
```

```
        start=pd.to_datetime(daily.Time(), unit="s", utc=True),
        end=pd.to_datetime(daily.TimeEnd(), unit="s", utc=True),
        freq=pd.Timedelta(seconds=daily.Interval()),
        inclusive="left"
    ),
    "temperature_2m_mean": daily_temperatures
}

daily_dataframe = pd.DataFrame(daily_data)
daily_dataframe = daily_dataframe.set_index(pd.to_datetime(daily_dataframe['date'])).dropna()
print(daily_dataframe.head())
```

Part B: Constructing a Stochastic Model

Objective: Construct a model for temperature dynamics that includes both a deterministic component (trend and seasonality) and a stochastic component to capture the residual variation.

Task 1: Exploratory Data Analysis (EDA)

1. Plot the time series, rolling mean, and rolling variance.
2. Decompose the time series into trend, seasonality, and residual components using the `seasonal_decompose` function.

Task 2: Residual Analysis

1. Visualize the residuals using histograms and Q-Q plots.
2. Compute the autocorrelation (ACF) and partial autocorrelation (PACF) plots.
3. Use statistical tests (e.g., the Augmented Dickey-Fuller test) to evaluate the stationarity of the residuals.

Example Code Snippet

```
from statsmodels.tsa.seasonal import seasonal_decompose
import matplotlib.pyplot as plt

# Decompose the time series (assume 365 days per year)
decompose_result = seasonal_decompose(daily_dataframe['temperature_2m_mean'],
                                      model='additive', period=365)
trend = decompose_result.trend
seasonal = decompose_result.seasonal
residual = decompose_result.resid

% Plot the decomposition
decompose_result.plot()
plt.show()

% Plot residual distribution
plt.hist(residual.dropna(), bins=60, figsize=(8,6))
plt.title("Residual Distribution")
plt.xlabel("Residual")
plt.ylabel("Frequency")
plt.show()
```

Part C: Fitting the Model under $\mathbb{P}$

Objective: Calibrate a deterministic seasonal model for temperature and fit an autoregressive (AR) process to the residuals.

Task 1: Deterministic Model Fitting

1. Specify the model:
$$T(t) = a + b\,t + \alpha \sin\left(\frac{2\pi}{365.25}t + \theta\right)$$
2. Use `scipy.optimize.curve_fit` to estimate the parameters a, b, α, and θ.
3. Compute the model fit and extract residuals.

Task 2: Autoregressive Modeling of Residuals

1. Fit an AR model to the residuals.
2. Determine the order of the AR process using information criteria (e.g., AIC).
3. Estimate the mean-reversion parameter κ.

Example Code Snippet

```
import numpy as np
from scipy.optimize import curve_fit

% Define the deterministic model function
def model_fit(x, a, b, a1, b1):
    omega = 2*np.pi/365.25
    return a + b*x + a1*np.cos(omega*x) + b1*np.sin(omega*x)

% Convert dates to ordinal numbers and fit model
first_ord = daily_dataframe.index[0].toordinal()
xdata = np.array([date.toordinal() - first_ord for date in daily_dataframe.index])
ydata = daily_dataframe['temperature_2m_mean']

params_all, cov = curve_fit(model_fit, xdata, ydata, method='lm')
daily_dataframe['model'] = model_fit(xdata, *params_all)
daily_dataframe['residuals'] = daily_dataframe['temperature_2m_mean'] - daily_dataframe['model

print("Estimated parameters:", params_all)
```

Part D: Pricing Temperature Options

Objective: Price weather derivatives based on the temperature model using Monte Carlo simulation.

Task 1: Monte Carlo Simulation of Temperature Paths

1. Develop a simulation algorithm based on an Euler discretization scheme.
2. Use the previously calibrated deterministic model for the trend and seasonality.
3. Incorporate the stochastic AR model for the residuals.

Task 2: Define Option Payoff Functions

Introduce the following common option payoffs for weather derivatives:

- **Call Option with Cap:**
$$\xi = \min\{\alpha \max(DD - K, 0), C\}$$
- **Put Option with Floor:**
$$\xi = \min\{\alpha \max(K - DD, 0), F\}$$
- **Collar:**
$$\xi = \min\{\alpha \max(DD - K_1, 0), C\} - \min\{\beta \max(K_2 - DD, 0), F\}$$

where DD represents the cumulative degree days over the period of interest.