Topic 1A: Time Series as Stochastic Processes

Victor M. Preciado

Contents

1 Time-Series Data

Time-series data are prevalent in many fields, including:

- ***Finance:*** Stock prices, exchange rates, and economic indicators.
- ***Signal Processing:*** Audio signals, radar signals, and communication signals.
- ***Engineering:*** Sensor data in manufacturing and energy consumption patterns.
- ***Healthcare:*** Patient vital signs, such as heart rate and blood pressure, monitored over time.
- ***Weather Forecasting:*** Temperature, precipitation, and other meteorological variables.

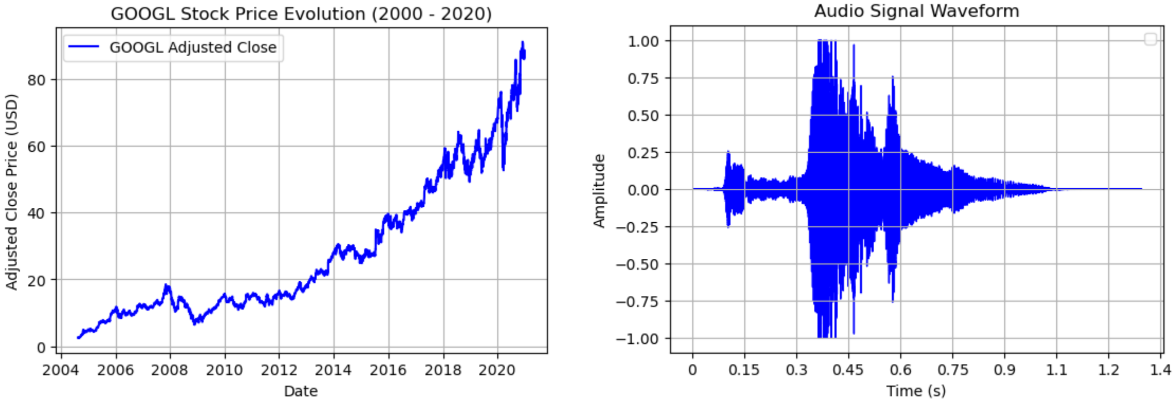

Figure 1: (Left) Evolution of Google stock prices from 2000 to 2020. (Right) Sound waveform.

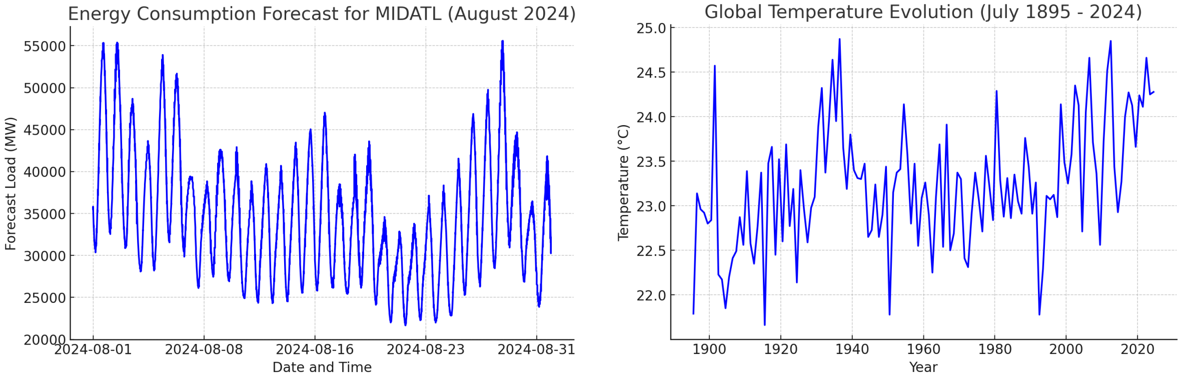

Figure 2: (Left) Energy consumption in the Mid-Atlantic U.S. region during August 2024. (Right) Global temperature evolution from 1895 to 2024

Given the widespread presence of time series data across these diverse domains, our ability to analyze and interpret such data is of paramount importance. Tools for time series analysis allow practitioners to *uncover patterns, identify trends, and make forecasts*, which are critical for informed decision-making. It is worth noting that certain phenomena are inherently more predictable than others. Our ability to forecast a particular event or value depends on several critical factors, such as the quantity and quality of the available data. A central aspect of effective forecasting is determining when accurate predictions are feasible, as opposed to when forecasts offer no advantage over random chance. Reliable forecasts capture authentic patterns and relationships within historical data, without simply replicating past events that are unlikely to recur. Distinguishing between random fluctuations, which should be disregarded, and genuine patterns, which require modeling, is crucial. In this direction, the complexity of time series data requires specialized techniques to extract meaningful insights.

1.1 The Forecasting Process

The forecasting process typically involves the following steps:

1. **Gathering Information:** Two types of information are essential: *statistical data* and *expert knowledge* from those familiar with the system. Expert knowledge helps identify the underlying causal factors, while the data collected must be statistically informative, meaning past observations should provide insights into future outcomes.

2. **Preliminary Analysis:** Graphing the data helps identify trends, seasonality, and potential outliers. It also provides insight into relationships between variables and helps guide model selection.

3. **Model Selection and Fitting:** The choice of model will depend on the availability of data, the strength of relationships between variables, and the specific objectives of the forecast. In this text, a diverse array of models will be explored, each with its own underlying assumptions and methodological approaches. Understanding these assumptions is crucial, as they guide the model's applicability to different types of data and forecasting scenarios.

4. **Model Evaluation:** Once a model has been selected, forecasts will be generated and their accuracy will be assessed as actual data becomes available. In this text, we will cover several techniques to evaluate forecast accuracy, while practical challenges such as missing data and limited time series length must be carefully managed during implementation.

The rigorous analysis and forecasting of time series data require a deep understanding of the underlying stochastic nature of the data. This involves treating time series as random processes, where each observation is seen as a realization of a stochastic process. By exploring time series through this lens, we can better model, predict, and infer patterns that are otherwise obscured by the inherent randomness in the data.

2 Time Series as Stochastic Processes

A time series of length L is a sequence of observations $(y_1, y_2, \ldots, y_L)$ recorded at specific time points $t_1, t_2, \ldots, t_L$. In this text, we will focus on the case in which these time points are uniformly separated, i.e., $t_k - t_{k-1} = T$ for all $k \in \mathbb{N}$, with T being a constant **sampling period**. Statistically, the entries of a time series can be interpreted as realizations of a sequence of random variables[1] $(Y_1, Y_2, Y_3, \ldots)$. This means that the value y_k observed at time t_k can be interpreted as a realization of the random variable Y_k. The entire collection of these random variables, indexed by time, forms

[1] All these random variables are defined on the same probability space $(\Omega, \mathcal{F}, P)$.

what is known as a discrete-time **stochastic process**, denoted by $\mathcal{Y} = \{Y_k : k \in \mathbb{N}\}$. The observed sequence $(y_1, y_2, \ldots, y_L)$ is referred to as a **sample path** of the underlying stochastic process. This statistical perspective enables the use of probabilistic methods to analyze and forecast the behavior of time series data.

2.1 Statistical Properties of Stochastic Processes

Understanding the statistical properties of a random process is crucial in time series analysis, as it allows for rigorous modeling, prediction, and inference by capturing the underlying patterns and dependencies in the data. A complete probabilistic description of a stochastic process $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_L\}$ of length L is provided by its **Joint Distribution (JD)**. The JD is a multivariate cumulative distribution function (CDF) defined as:

$$F_{\mathcal{Y}}(y_1, \ldots, y_L) = \Pr\{Y_1 \le y_1, \ldots, Y_L \le y_L\}.$$

If $(Y_1, \ldots, Y_L)$ are jointly continuous random variables, the $\mathcal{Y}$ has an joint probability density function (PDF), $f_{\mathcal{Y}}(y_1, \ldots, y_L)$. Although this CDF/PDF fully describes the stochastic process—including all possible marginal and conditional distributions—it is often difficult to work with and not easily accessible for time series analysis.

The mean, variance, autocovariance, and autocorrelation are fundamental statistical properties of a random process and are easier to handle than the JD. The following notation is used[2]:

- **Mean**: The mean of the k-th sample of a random process $\mathcal{Y}$ is defined as follows:

  $$\mu_{\mathcal{Y}}(k) = \mathbb{E}[Y_k] = \int_{-\infty}^{\infty} y\, f_{Y_k}(y)\, dy,$$

  where $\mathbb{E}[\cdot]$ denotes the expectation operator.

- **Variance**: The variance of the k-th sample of the random process $\mathcal{Y}$ is defined as follows:

  $$\sigma^2_{\mathcal{Y}}(k) = \mathrm{Var}(Y_k) = \mathbb{E}[(Y_k - \mu_{\mathcal{Y}}(k))^2] = \int_{-\infty}^{\infty} (y - \mu_{\mathcal{Y}}(k))^2\, f_{Y_k}(y)\, dy.$$

  If the variance is constant over time, the process is said to be **homoskedastic**; otherwise, it is called **heteroskedastic**.

- **(Auto)covariance**: The autocovariance between the k-th sample of a random process $\mathcal{Y}$ and the sample lagged by h sampling periods (i.e., a lag of $h \cdot T$ time units) is defined as follows:

  $$C_{\mathcal{Y}}(k,h) = \mathrm{Cov}(Y_k, Y_{k-h}) = \mathbb{E}[(Y_k - \mu_{\mathcal{Y}}(k))(Y_{k-h} - \mu_{\mathcal{Y}}(k-h))] = \mathbb{E}[Y_k Y_{k-h}] - \mu_{\mathcal{Y}}(k)\mu_{\mathcal{Y}}(k-h).$$

  This covariance is used to measure the linear dependence (or lack thereof) between two samples in a stochastic process. Note that even when $C_{\mathcal{Y}}(k,h) = 0$, there can still be nonlinear dependence between Y_k and Y_{k-h}. Furthermore, for $h = 0$, the autocovariance reduces to the variance, $C_{\mathcal{Y}}(k,0) = \sigma^2_{\mathcal{Y}}(k)$.

- **(Auto)correlation**: The autocorrelation function (ACF) is a normalized version of the autocovariance, defined as:

  $$R_{\mathcal{Y}}(k,h) = \frac{\mathrm{Cov}(Y_k, Y_{k-h})}{\sigma_{\mathcal{Y}}(k)\sigma_{\mathcal{Y}}(k-h)} \in [-1, 1].$$

[2] From now on, it is assumed that the mean, variance and autocovariance/autocorrelation exist and are finite.

The autocorrelation assesses how the current value Y_k of a random process is correlated with its own past values.

A random process $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_L\}$ is said to be **Gaussian** if the vector of random variables $[Y_1, Y_2, \ldots, Y_L]^\intercal$ follows a multivariate jointly Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with mean vector $\boldsymbol{\mu}_\mathcal{Y} = [\mathbb{E}[Y_1], \mathbb{E}[Y_2], \ldots, \mathbb{E}[Y_L]]^\intercal \in \mathbb{R}^L$ and autocovariance matrix $\Sigma_\mathcal{Y} \in \mathbb{R}^{L\times L}$. The entries of the covariance matrix $\Sigma_\mathcal{Y}$ are given by the autocovariance function, such that the (i,j)-th entry satisfies: $[\Sigma_\mathcal{Y}]_{i,j} = \text{Cov}(Y_i, Y_j)$. One key implication of a random process being Gaussian is that the entire process is fully characterized by its mean vector $\boldsymbol{\mu}_\mathcal{Y}(k)$ and the autocovariance matrix $\Sigma_\mathcal{Y}$. This implies that knowing these two parameters is sufficient to describe the statistical properties of the process. Moreover, any linear combination of the components of a Gaussian process is also Gaussian. This property simplifies the analysis and modeling of such processes, particularly in time series and signal processing, where many complex behaviors can be reduced to operations involving the mean and covariance.

Example 1: A Simple Random Process

Consider a random process $\mathcal{Y} = \{Y_k : k \in \mathbb{N}\}$ defined by the following recursion:

$$Y_k = \phi Y_{k-1} + \epsilon_k \text{ for all } k \in \mathbb{N}, \text{ with deterministic initial condition } Y_0 = 0,$$

where $|\phi| < 1$ is a constant parameter, and ϵ_k is a sequence of independent and identically distributed (i.i.d.) Gaussian random variables with zero mean and variance σ_ϵ^2. This process is an example of an **autoregressive model of order 1**, denoted by AR(1). Its statistical properties are as follows:

- **Mean**: Since ϵ_k has zero mean, the mean of the process $\mathcal{Y}$ is given by:

  $$\mu_\mathcal{Y}(k) = \mathbb{E}[Y_k] = \phi\mathbb{E}[Y_{k-1}] + \mathbb{E}[\epsilon_k].$$

  Since $Y_0 = 0 = \mathbb{E}[Y_0]$ and $\mathbb{E}[\epsilon_k] = 0$ for all k, it follows that $\mu_\mathcal{Y}(k) = 0$ for all k.

- **Variance**: The variance of the k-th sample is:

  $$\sigma_\mathcal{Y}^2(k) = \text{Var}(Y_k) = \text{Var}(\phi Y_{k-1} + \epsilon_k).$$

  Given that Y_{k-1} and ϵ_k are independent, we have:

  $$\sigma_\mathcal{Y}^2(k) = \phi^2\text{Var}(Y_{k-1}) + \sigma_\epsilon^2 = \phi^2\sigma_\mathcal{Y}^2(k-1) + \sigma_\epsilon^2.$$

  Since $Y_0 = 0 = \sigma_\mathcal{Y}^2(0)$, the solution to this recursion is (exercise):

  $$\sigma_\mathcal{Y}^2(k) = \sigma_\epsilon^2 \sum_{i=0}^{k-1} \phi^{2i} = \sigma_\epsilon^2 \frac{1-\phi^{2k}}{1-\phi^2}.$$

  Therefore, in the limit $k \to \infty$, we have that $\sigma_\mathcal{Y}^2(k) \to \frac{\sigma_\epsilon^2}{1-\phi^2}$.

- **95% Confidence Interval**: Since the noise terms ϵ_k are Gaussian and independent, the random variable Y_k is a linear combination of independent Gaussian random variables. Thus, by the properties of linear combinations of Gaussian variables, Y_k is also

Gaussian for all k. Therefore, a 95% confidence interval for Y_k can be constructed as $\mu_{\mathcal{Y}}(k) \pm 1.96 \cdot \sigma_{\mathcal{Y}}(k)$. Since $\mu_{\mathcal{Y}}(k) = 0$, the 95% confidence interval simplifies to:

$$Y_k \in \left[-1.96 \cdot \sigma_\epsilon \sqrt{\frac{1-\phi^{2k}}{1-\phi^2}}, 1.96 \cdot \sigma_\epsilon \sqrt{\frac{1-\phi^{2k}}{1-\phi^2}} \right].$$

This interval represents the range in which the true value of Y_k will lie with 95% probability, given the known distribution of Y_k.

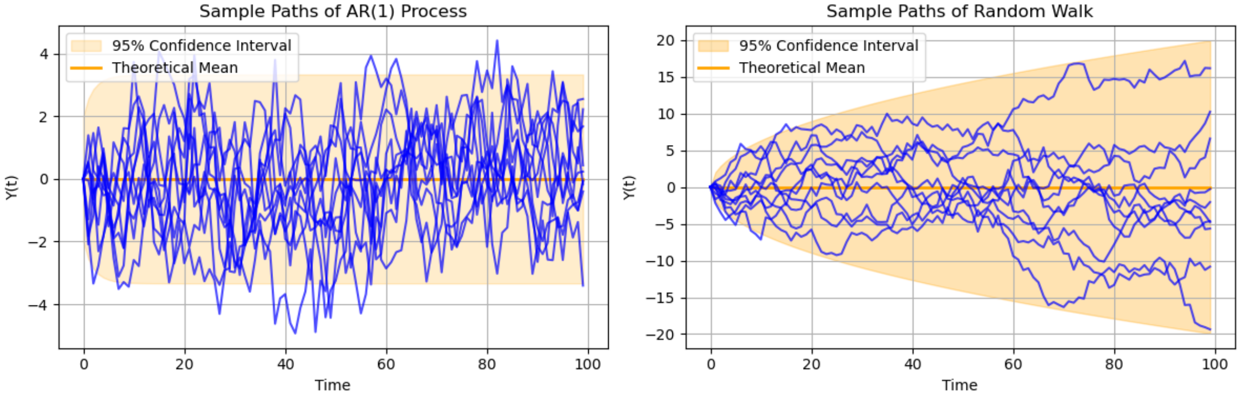

Figure 3: Comparison of 10 sample paths of an AR(1) process (left) and a random walk with zero drift (right), both starting from $Y_0 = 0$. The theoretical means are plotted in orange. The shaded regions represent the 95% confidence intervals based on the theoretical variance, i.e., $[\mu_{\mathcal{Y}}(k) \pm 2 \cdot \sigma_{\mathcal{Y}}(k)]$, for each process.

Example 2: Random Walk with Drift

Consider a random process $\mathcal{Y} = \{Y_k : k \in \mathbb{N}\}$ defined by the following recursion:

$$Y_k = \delta + Y_{k-1} + \epsilon_k \text{ for all } k \in \mathbb{N}, \text{ with initial condition } Y_0 = 0,$$

where δ is a constant called the **drift**, and ϵ_k is a sequence of i.i.d. Gaussian random variables with zero mean and variance σ_ϵ^2. This process is an example of a **random walk with drift** and its statistical properties are as follows:

- **Mean**: The mean of the process $\mathcal{Y}$ is given by:

  $$\mu_{\mathcal{Y}}(k) = \mathbb{E}[Y_k] = \mathbb{E}[\delta + Y_{k-1} + \epsilon_k] = \delta + \mathbb{E}[Y_{k-1}] + \mathbb{E}[\epsilon_k].$$

  Since $\mathbb{E}[\epsilon_k] = 0$, we have $\mu_{\mathcal{Y}}(k) = \delta + \mu_{\mathcal{Y}}(k-1)$. Since $Y_0 = 0$, this recurrence relation gives:

  $$\mu_{\mathcal{Y}}(k) = k\delta.$$

  This indicates that the mean of the process increases linearly over time, reflecting the effect of the drift δ.

- **Variance**: The variance of the k-th sample is:

  $$\sigma_{\mathcal{Y}}^2(k) = \mathrm{Var}(Y_k) = \mathrm{Var}(\delta + Y_{k-1} + \epsilon_k).$$

  Given that Y_{k-1} and ϵ_k are independent (and therefore uncorrelated), we have:

  $$\sigma_{\mathcal{Y}}^2(k) = \mathrm{Var}(Y_{k-1}) + \mathrm{Var}(\epsilon_k) = \sigma_{\mathcal{Y}}^2(k-1) + \sigma_\epsilon^2.$$

  Since $Y_0 = 0$, the variance evolves as:

  $$\sigma_{\mathcal{Y}}^2(k) = k\sigma_\epsilon^2.$$

  Thus, the variance of the process increases linearly with time (and the standard deviation as the square root of time), indicating that the process becomes more variable as time progresses.

- **95% Confidence Interval**: Because ϵ_k are Gaussian random variables, Y_k, as a sum of Gaussian random variables, is also Gaussian at each time k. Using this distribution, a 95% confidence interval for Y_k can be computed as:

  $$Y_k \in [k\delta - 1.96 \cdot \sqrt{k}\sigma_\epsilon, k\delta + 1.96 \cdot \sqrt{k}\sigma_\epsilon].$$

  This interval gives a range in which we expect Y_k to fall with 95% probability at each time step k, accounting for both the drift δ and the increasing variance over time.

2.2 Stationarity in Time Series

Stationarity is a fundamental concept in time series analysis that refers to the idea that the statistical properties of a time series do not change over time. When a process is stationary, it is easier to model and make predictions because its behavior is consistent and predictable over time.

2.2.1 Strong-Sense Stationarity (SSS)

One of the strongest forms of stationarity is known as **strong-sense stationarity (SSS)**. A random process $\mathcal{Y} = \{Y_k : k \in \mathbb{N}\}$ is SSS if the joint distribution of any finite collection of random variables from the process is invariant under shifts in time. Specifically, for any collection of discrete time indices $k_1, k_2, \ldots, k_n$, the joint distribution of the corresponding random variables $(Y_{k_1}, Y_{k_2}, \ldots, Y_{k_n})$ remains unchanged if we shift all time indices by a constant h. Formally, the process is SSS if:

$$\Pr(Y_{k_1} \leq y_{k_1}, Y_{k_2} \leq y_{k_2}, \ldots, Y_{k_n} \leq y_{k_n}) = \Pr(Y_{k_1+h} \leq y_{k_1}, Y_{k_2+h} \leq y_{k_2}, \ldots, Y_{k_n+h} \leq y_{k_n}),$$

for all $n \in \mathbb{N}$, $h \in \mathbb{N}$, and any values $y_{k_1}, y_{k_2}, \ldots, y_{k_n} \in \mathbb{R}$. In other words, the entire joint probability structure of the process does not change over time.

The condition of strong-sense stationarity is very stringent and has several important implications for the random process $\mathcal{Y}$. In particular:

- **Time-Invariance of Means and Variances:** Since the entire joint distribution is invariant under time shifts, this implies that the mean and variance of the process must be constant over time, i.e.,

$$\mu_{\mathcal{Y}}(k) = \mu_{\mathcal{Y}}(k+h) = \mu_{\mathcal{Y}} \text{ and } \sigma_{\mathcal{Y}}(k) = \sigma_{\mathcal{Y}}(k+h) = \sigma_{\mathcal{Y}}, \text{ for all } k, h \in \mathbb{N}.$$

This indicates that the expected value of the process does not change as time progresses.

- **Shift-Invariance of the Covariances and Autocorrelation:** Strong-sense stationarity implies that the pairwise statistical dependencies within the process remain constant over time. As a result, the autocovariance (and autocorrelation) between two random variables Y_k and Y_{k+h} depends only on the time difference h, not on the specific times k or $k+h$. In a stationary process, we can express the autocovariance and autocorrelation functions using a single argument (with a slight abuse of notation), i.e., $C_{\mathcal{Y}}(k,h) = C_{\mathcal{Y}}(h),$ for all $k, h \in \mathbb{N}$.

Example 3: White Noise Process

A simple example of a strongly stationary process is the **white noise process**. In this case, the random variables Y_k are i.i.d. with zero mean and constant variance σ^2. Since each Y_k is independent of all others and has the same joint distribution regardless of time, the white noise process trivially satisfies the condition for strong-sense stationarity. Furthermore, we can compute other relevant statistics, such as the mean, the covariance, and the autocorrelation.

- **Mean:** The mean of the white noise process is constant and given by:

  $$\mu_{\mathcal{Y}}(k) = \mathbb{E}[Y_k] = 0 \text{ for all } k.$$

  This reflects the fact that, on average, the random variables in a white noise process have a value of zero.

- **Covariance:** Due to the independence of the random variables in the white noise process, the covariance is zero for any non-zero lag h:

  $$C_{\mathcal{Y}}(k,h) = \begin{cases} \sigma^2 & \text{if } h = 0, \\ 0 & \text{if } h \neq 0, \end{cases}$$

  where σ^2 is the constant variance of the process. This means that the random variables are uncorrelated unless they coincide in time.

- **Autocorrelation:** Given the form of the covariance function, the autocorrelation function is:

  $$R_{\mathcal{Y}}(h) = \begin{cases} 1 & \text{if } h = 0, \\ 0 & \text{if } h \neq 0. \end{cases}$$

  This indicates that each random variable in the white noise process is only correlated with itself and uncorrelated with all other variables, regardless of the time lag.

In summary, the white noise process is a simple yet fundamental example of a strongly stationary process, characterized by zero mean, constant variance, and no autocorrelation between different time points. In Fig. 4 we include a realization of a sample path of length $L = 1{,}000$. In the right subplot, we also include the empirical autocorrelation function.

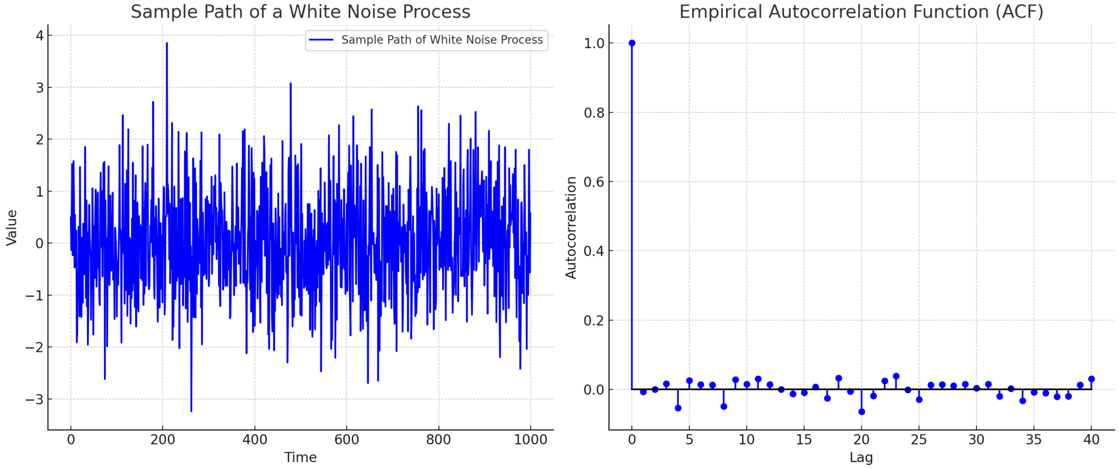

Figure 4: (Left) A typical sample path of a white noise and (right) its empirical ACF.

Strong-sense stationarity plays a critical role in many areas of time series analysis and stochastic processes. Stationarity simplifies modeling because it ensures that the statistical properties of the process remain constant over time. However, due to its stringent requirements, SSS is often difficult to achieve in practice. Many real-world processes, such as stock prices or environmental data, exhibit time-varying trends or volatility, making them non-stationary.

2.2.2 Weak-Sense Stationarity (WSS)

While strong-sense stationarity requires that the entire joint distribution of the process be invariant under time shifts, **Weak-Sense Stationarity (WSS)**, also called *wide-sense stationary*, is a less restrictive condition. WSS requires only that the mean and variance are time-invariant, and the covariance is shift-invariant. These conditions are often sufficient for practical applications and are easier to achieve real-world processes.

A random process $\mathcal{Y} = \{Y_k : k \in \mathbb{N}\}$ is said to be Weak-Sense Stationary if it satisfies the following three conditions:

1. **Time-Invariant Mean and Variance**: The mean and variance of the process must be constant for all k, i.e., $\mu_{\mathcal{Y}}(k) = \mu_{\mathcal{Y}}$ and $\sigma^2_{\mathcal{Y}}(k) = \sigma^2_{\mathcal{Y}}$ for all k. This means the expected value and the volatility of the process does not change over time.

2. **Shift-Invariant Autocovariance/Autocorrelation**: The autocovariance $C_{\mathcal{Y}}(k,h)$ depends only on the time difference (or lag) h, and not on the specific times k or $k+h$. Thus, we have: $C_{\mathcal{Y}}(k,h) = C_{\mathcal{Y}}(h)$ for all k,h. This implies that the covariance between two points in the process is determined only by the lag h, not by their absolute positions in the time series[3].

Note that strong-sense stationarity (SSS) is a stricter condition than weak-sense stationarity (WSS); therefore, if a process is SSS, it is also WSS. However, the reverse implication does not always hold—being WSS does not necessarily imply that the process is SSS. However, there is an

[3] Whenever the autocovariance/autocorrelation is described as a function of a single argument, i.e., the lag h, it implicitly suggests that the stochastic process is WSS.

exception to this rule: If $\mathcal{Y}$ is WSS and *Gaussian*, then $\mathcal{Y}$ is also SSS, since a Gaussian distribution is uniquely defined by the means and the covariance structure.

Weak-sense stationarity is fundamental in time series analysis because it simplifies the modeling and analysis of stochastic processes. Many time series models rely on the WSS assumption, as it ensures that the statistical properties of the process—specifically, the mean, variance, and autocovariance structure—remain constant over time.

Example 4: Autocovariance of a Random Walk

Consider a random walk process defined by:

$$Y_k = Y_{k-1} + \epsilon_k, \quad \epsilon_k \sim \mathcal{N}(0, \sigma_\epsilon^2) \text{ with initial condition } Y_0 = 0,$$

where ϵ_k are i.i.d. Gaussian random variables with zero mean and constant variance σ_ϵ^2. For this random walk, we already showed that the mean is zero for all k. Therefore, the covariance becomes $C_{\mathcal{Y}}(k,h) = \mathbb{E}[Y_k Y_{k-h}]$.
We can express Y_k and Y_{k-h} as a sum of the increments ϵ_k:

$$Y_k = Y_0 + \sum_{i=1}^{k} \epsilon_i \text{ and } Y_{k-h} = Y_0 + \sum_{i=1}^{k-h} \epsilon_i.$$

Now, we compute the covariance:

$$C_{\mathcal{Y}}(k,h) = \mathbb{E}\left[\left(\sum_{i=1}^{k} \epsilon_i\right)\left(\sum_{j=1}^{k-h} \epsilon_j\right)\right] = \sum_{i=1}^{k}\sum_{j=1}^{k-h} \mathbb{E}[\epsilon_i \epsilon_j].$$

Since ϵ_i are i.i.d., $\mathbb{E}[\epsilon_i \epsilon_j] = 0$ for $i \neq j$, and for $i = j$, we have $\mathbb{E}[\epsilon_i^2] = \sigma_\epsilon^2$. Therefore, the only terms that contribute to the sum are those with $i = j$, yielding:

$$C_{\mathcal{Y}}(k,h) = \sum_{i=1}^{k-h} \sigma_\epsilon^2 = (k-h)\sigma_\epsilon^2 \quad \text{for } h \leq k.$$

This shows that the covariance depends linearly on k, the current time, and decreases with increasing lag h. Also, since the covariance depends on both k and the lag h, a random walk is *not* stationary (in any sense).

Example 5: (Asymptotic) Autocovariance of AR(1)

Consider the AR(1) stochastic process, defined by the recursion:

$$Y_k = \phi Y_{k-1} + \epsilon_k \quad \text{with} \quad |\phi| < 1,$$

where ϵ_k is a white noise process with zero mean and variance σ_ϵ^2. As we observe in Fig. 5-(left), the AR(1) process is not stationary for small k, since the deterministic initial condition induces an initial growth of the variance. However, for large k, the AR(1) process becomes WSS exponentially fast, as the influence of the initial condition decays over time. The

analysis of the mean and the variance in Example 1 already established that the mean is zero and the variance converges to the constant $\frac{\sigma_\epsilon^2}{1-\phi^2}$ as $k \to \infty$.
Let us now make an analysis of the autocovariance and the autocorrelation for k large. To derive the autocovariance, we start by writing Y_{k+h} in terms of Y_k using the recursion (exercise):

$$Y_{k+h} = \phi^h Y_k + \phi^{h-1}\epsilon_{k+1} + \phi^{h-2}\epsilon_{k+2} + \cdots + \epsilon_{k+h}.$$

Multiplying this expression by Y_k and taking expectations, we get:

$$\mathbb{E}[Y_k Y_{k+h}] = \mathbb{E}\left[Y_k \left(\phi^h Y_k + \sum_{i=1}^{h} \phi^{h-i}\epsilon_{k+i}\right)\right].$$

Using the fact that Y_k and ϵ_{k+i} are uncorrelated for all i, the cross terms vanish, and we are left with:

$$\mathbb{E}[Y_k Y_{k+h}] = \phi^h \mathbb{E}[Y_k^2].$$

The asymptotic variance is given by $\sigma_{\mathcal{Y}}^2 = \mathbb{E}[Y_k^2] = \frac{\sigma_\epsilon^2}{1-\phi^2}$; hence,

$$\text{Cov}(Y_k, Y_{k+h}) = \mathbb{E}[Y_k Y_{k+h}] = \phi^h \frac{\sigma_\epsilon^2}{1-\phi^2} = C_{\mathcal{Y}}(h),$$

where the autocovariance is a function of the lag h only. The autocorrelation function is the normalized autocovariance:

$$R_{\mathcal{Y}}(h) = \phi^h.$$

For large lags h, if $|\phi| < 1$, the autocorrelation *decays exponentially*, and $R_{\mathcal{Y}}(h) \to 0$ as $h \to \infty$. This reflects that the AR(1) process becomes progressively uncorrelated for distant observations. In Fig. 5 we include a realization of a sample path of length $L = 1,000$, including the evolution of the empirical mean and variance using a rolling window. In the right subplot, we also include the empirical and theoretical autocorrelation function.

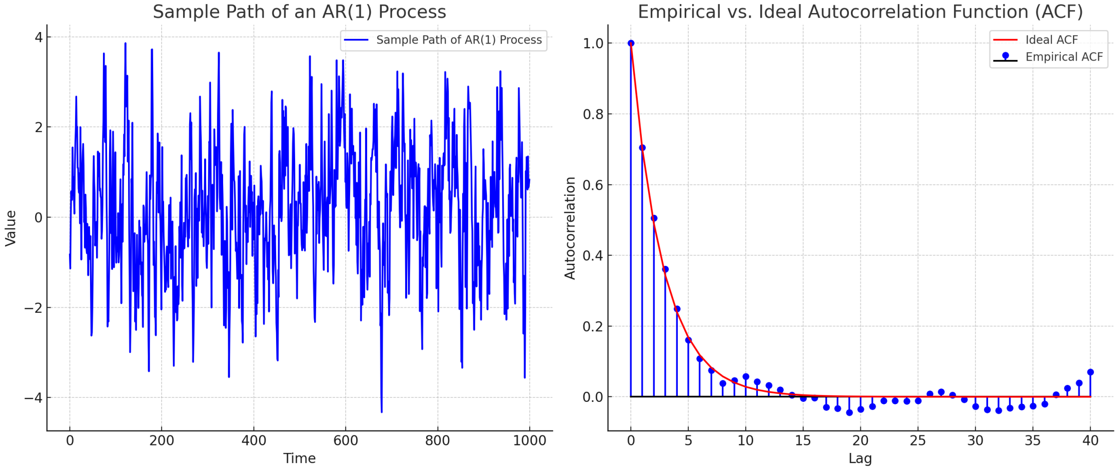

Figure 5: (Left) A sample path of the AR(1) process. (Right) Empirical and theoretical ACF.

2.2.3 Testing Stationarity

In many practical scenarios, we aim to model empirical time series data using models that assume stationarity. Stationarity implies that the statistical properties of the process—such as the mean, variance, and autocovariance—do not change over time. It is crucial to determine whether the observed sample path is likely to be a realization of a stationary stochastic process, as the assumption of stationarity underpins many common modeling techniques. If the time series is not stationary, the results of the model may be misleading, and our analysis would be based on a false premise.

To avoid this, empirical verification of stationarity can be carried out using several methods, as detailed below:

1. **Basic Statistical Tests**: Stationarity implies that the mean, variance, and autocovariance of the time series do not vary with time. A preliminary assessment of stationarity can be performed using the following techniques:

   - **Plot the Time Series**: Visually inspect the time series to check whether it fluctuates around a constant mean and exhibits consistent variability. If clear trends or changes in variance are observed, the process is likely non-stationary.
   - **Rolling Statistics**: Compute and plot the rolling empirical mean and rolling empirical variance using a fixed window size. If the rolling statistics remain approximately constant, this is indicative of stationarity. Significant shifts in these statistics over time suggest non-stationarity.
   - **Autocorrelation Function (ACF)**: Plot the autocorrelation function to assess how the autocorrelations evolve with increasing lags. For a stationary process, the autocorrelation function typically decays towards zero relatively quickly. A persistent autocorrelation at higher lags may signal non-stationarity.

   While these methods provide initial insights, formal statistical tests are required to confirm stationarity.

2. **Augmented Dickey-Fuller (ADF) Test**: The ADF test [1] is a commonly employed hypothesis test to formally assess the stationarity of a time series. The null and alternative hypotheses for the ADF test are:

   $$H_0 : \gamma = 0 \quad \text{(the series is non-stationary) vs. } H_A : \gamma < 0 \quad \text{(the series is stationary)}.$$

   where γ is the associated test statistics[4]. The ADF test provides a p-value and, if the p-value is below a chosen significance level (typically 0.05), we reject the null hypothesis, suggesting that the time series is stationarity. This test is widely used in practice and is implemented in the method `adfuller` in the Python package `statsmodels`. The function returns both the test statistic γ and the p-value, enabling users to make a formal decision on stationarity based on the test results.

2.3 Markov Processes

A **Markov process** is a type of stochastic process where the future behavior of the process depends only on its present state, and not on the sequence of events that preceded it. The process $\{Y_k : k \in$

[4] In simple terms, this test statistic checks whether a time series exhibits random walk behavior, which indicates non-stationarity.

$\mathbb{N}\}$ is said to be **Markovian** if, for all $k \in \mathbb{N}$, the conditional probability distribution of the next observation Y_{k+1} depends only on the current observation Y_k, i.e.,

$$\Pr(Y_{k+1} \leq y \mid Y_k = y_k, Y_{k-1} = y_{k-1}, \dots, Y_1 = y_1) = \Pr(Y_{k+1} \leq y \mid Y_k = y_k),$$

for all $k \in \mathbb{N}$ and $y \in \mathbb{R}$.

Example 6: AR(1) is a Markov Process

Consider the autoregressive model of order 1, AR(1), defined by the recursion:

$$Y_{k+1} = \phi Y_k + \epsilon_k, \quad \epsilon_k \sim \mathcal{N}(0, \sigma^2), \quad |\phi| < 1.$$

The AR(1) model is a Markov process because the conditional distribution of Y_{k+1} depends only on the value of Y_k, not on any earlier values. Specifically, the conditional PDF of Y_{k+1}, given the past values $Y_k, Y_{k-1}, \dots$, depends only on Y_k. Formally, the AR(1) model satisfies:

$$(Y_{k+1} \mid Y_k = y_k, Y_{k-1} = y_{k-1}, \dots) = (Y_{k+1} \mid Y_k = y_k) \sim \mathcal{N}(\phi\, y_1, \sigma^2),$$

which is the defining characteristic of a *Markov process*.

2.3.1 Higher-Order Markov Processes

In some cases, the future state of the process may depend not only on the current state but also on a finite number of previous states. A **higher-order Markov process** of order m satisfies the following condition:

$$\Pr(Y_{k+1} \leq y \mid Y_k, Y_k, \dots, Y_1) = \Pr(Y_{k+1} \leq y \mid Y_k, Y_{k-1}, \dots, Y_{k-m+1}).$$

Thus, in an m-th order Markov process, the conditional distribution given past observations depends solely on the previous m states. However, beyond m steps into the past, no additional information affects the prediction of future values. Markov processes, due to their finite memory property, provide a practical framework for modeling time series with short-term dependencies, making them an essential tool in stochastic modeling.

Example 7: AR(2) is a Second-Order Markov Process

Consider the autoregressive model of order 2, denoted by AR(2) and defined by the recursion:

$$Y_{k+1} = \phi_0 Y_k + \phi_1 Y_{k-1} + \epsilon_k, \quad \epsilon_k \sim \mathcal{N}(0, \sigma^2), \quad |\phi_0| + |\phi_1| < 1.$$

The AR(2) model is a *second-order Markov process* because the conditional distribution of the future value Y_{k+1} depends only on the values of Y_k and Y_{k-1}, not on any earlier values $Y_{k-2}, Y_{k-3}, \dots$ Specifically, the conditional PDF of Y_{k+1}, given the past values $Y_k, Y_{k-1}, Y_{k-2}, \dots$, depends only on Y_k and Y_{k-1}. Formally, the AR(2) model satisfies (exercise):

$$(Y_{k+1} \mid Y_k = y_k, Y_{k-1} = y_{k-1}, Y_{k-2} = y_{k-2}, \dots) = (Y_{k+1} \mid Y_k = y_k, Y_{k-1} = y_{k-1}),$$

which is the defining characteristic of a *second-order Markov process*.

3 Multivariate Time Series

A **multivariate time series** involves observing and analyzing multiple interrelated time-dependent variables simultaneously. Unlike a univariate time series, which focuses on a single variable, multivariate time series account for several variables together, enabling the study of potential interactions and dependencies among them. The strength of multivariate analysis lies in its ability to capture dynamic interdependencies between variables over time. This is especially valuable in fields such as economics, finance, and environmental science, where the behavior of one variable may influence or depend on others. For instance, one might consider three interdependent time series representing the evolution of energy demand, temperature and humidity in a region (see Fig. 6), where changes in one variable could influence the behavior of the others.

Let us consider a collection of C time series of length L, denoted by $\mathcal{Y}^1, \mathcal{Y}^2, \ldots, \mathcal{Y}^C$, co-evolving over discrete time[5]. Using these C time series, we can define a single vector-valued time series as $\boldsymbol{\mathcal{Y}} = (\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_L)$, where each vector in the sequence, $\mathbf{Y}_k = [Y_k^1, Y_k^2, \ldots, Y_k^C]^\intercal$, contain the values of the C time series at a given time k. Specifically, Y_k^i denotes the value of the i-th time series (also called **component**) at time step k, where $i \leq C$ and $k \leq L$.

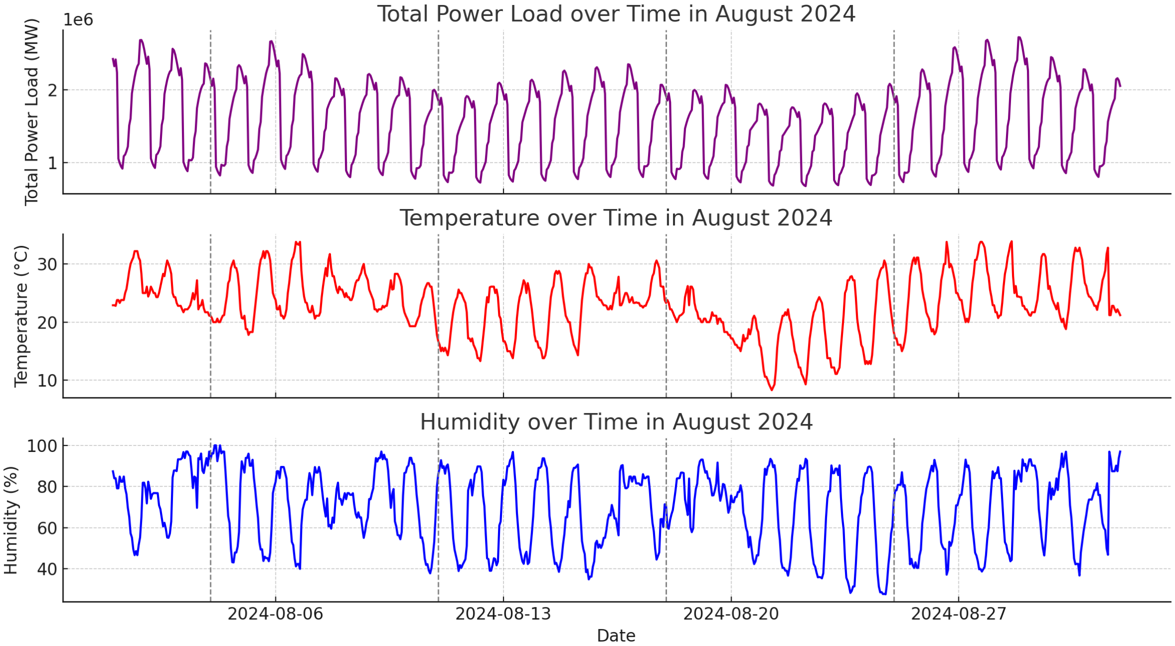

Figure 6: Hourly evolution of a multivariate time series including three components: Power Load (top, in MW), Temperature (middle, in Celsius), and Humidity (in %) in the Mid-Atlantic Region during August 2024. (This multivariate time series is available in GitHub)

[5] In our exposition, we will assume that all $\mathcal{Y}^1, \ldots, \mathcal{Y}^C$ are random processes evolving on the same probability space, and their means, variances, and covariances exist and are finite.

3.1 Statistical Properties of Multivariate Time Series

A complete statistical description of a multivariate random process can be expressed in terms of its joint CDF, as follows. Consider a multivariate process $\boldsymbol{\mathcal{Y}}$ of length L. A complete statistical behavior of the process is encapsulated in the following joint CDF (where the arguments are vectors and the inequalities in the right-hand side are component-wise):

$$F_{\boldsymbol{\mathcal{Y}}}(\mathbf{y}_1, \ldots, \mathbf{y}_L) = \mathbb{P}(\mathbf{Y}_1 \leq \mathbf{y}_1, \ldots, \mathbf{Y}_L \leq \mathbf{y}_L).$$

Notice that this joint distribution involves the entire set of variables across all time steps and all dimensions. Thus, this joint CDF characterizes the interdependencies and stochastic behavior of the entire system, capturing both temporal and cross-variable dependencies.

In practice, specifying the high-dimensional joint distribution of a multivariate process is challenging due to its complexity and the large number of variables involved, particularly as both the number of time steps L and the number of variables C increase. To analyze multivariate time series more effectively, a variety of simplified but powerful statistical tools are employed, each designed to capture the relationships between the individual series and their temporal dynamics. Below, we provide an overview of the most common statistical measures used in the analysis of multivariate stochastic processes:

- **Mean Vector:** The **mean vector $\boldsymbol{\mu}_{\boldsymbol{\mathcal{Y}}}(k)$** provides a summary of the expected values for each component of the multivariate time series at a given time k. Mathematically, it is expressed as:

  $$\boldsymbol{\mu}_{\boldsymbol{\mathcal{Y}}}(k) = \mathbb{E}[\mathbf{Y}_k] \in \mathbb{R}^p.$$

  This vector encapsulates the evolution of the average behavior of each component, offering a foundational perspective on the trends present in the data.

- **Autocovariance Matrix:** The **autocovariance matrix** $\Sigma_{\boldsymbol{\mathcal{Y}}}(k_1, k_2) \in \mathbb{R}^{p \times p}$ quantifies the degree to which the components of the time series co-vary across two time points k_1 and k_2. It is formally defined as:

  $$\Sigma_{\boldsymbol{\mathcal{Y}}}(k_1, k_2) = \mathbb{E}[(\mathbf{Y}_{k_1} - \boldsymbol{\mu}_{\boldsymbol{\mathcal{Y}}}(k_1))(\mathbf{Y}_{k_2} - \boldsymbol{\mu}_{\boldsymbol{\mathcal{Y}}}(k_2))^\intercal].$$

  Notice that the expression inside the expectation is the outer product of two C-dimensional vectors; hence, the resulting object is a $C \times C$ matrix. The entries in this matrix can be interpreted as follows:

  1. The *diagonal elements* of $\Sigma_{\boldsymbol{\mathcal{Y}}}(k_1, k_2)$ are the autocovariance of each individual component, i.e.,

     $$[\Sigma_{\boldsymbol{\mathcal{Y}}}(k_1, k_2)]_{ii} = \mathrm{Cov}(Y^i_{k_1}, Y^i_{k_2}) = \mathbb{E}[(Y^i_{k_1} - \mu_{Y_i}(k_1))(Y^i_{k_2} - \mu_{Y_i}(k_2))^\intercal],$$

     If $k_1 = k_2 = k$, the diagonal terms represent the variance of the individual time series at time k.

  2. The *off-diagonal elements* of $\Sigma_{\boldsymbol{\mathcal{Y}}}(k_1, k_2)$ are called the **cross-covariances**, since they quantify the co-movement between two different time series, as follows:

     $$[\Sigma_{\boldsymbol{\mathcal{Y}}}(k_1, k_2)]_{ij} = \mathrm{Cov}(Y^i_{k_1}, Y^j_{k_2}) = \mathbb{E}[(Y^i_{k_1} - \mu_{Y_i}(k_1))(Y^j_{k_2} - \mu_{Y_j}(k_2))^\intercal],$$

     where $i \neq j$. These elements capture the interdependence between different components of the multivariate series at different time points.

In conclusion, the entries of the autocovariance matrix provides insights into both the variability of each individual series (represented by the diagonal elements) and the co-movement between different series (represented by the off-diagonal elements).

Python Example: VAR(1) Process Analysis

In this example, a synthetic multivariate time series with $C = 3$ components is generated using the so-called **Vector Autoregressive (VAR)** process using Python. In particular, we use the VAR(1) process, defined as the following vector-valued recursion:

$$\mathbf{Y}_k = A\mathbf{Y}_{k-1} + \boldsymbol{\epsilon}_k,$$

where $\mathbf{Y}_k = [Y_k^1, Y_k^2, Y_k^3]^\intercal$ and the matrix A, called **transition matrix**, is given by:

$$A = \begin{bmatrix} 0.7 & 0.1 & 0.0 \\ 0.3 & 0.5 & 0.2 \\ 0.0 & 0.2 & 0.6 \end{bmatrix}.$$

The noise vector $\boldsymbol{\epsilon}_k = [\epsilon_k^1, \epsilon_k^2, \epsilon_k^3]^\intercal$ follows a multivariate normal distribution with zero mean and covariance matrix:

$$\Sigma_\epsilon = \begin{bmatrix} 1 & 0.5 & 0.3 \\ 0.5 & 1 & 0.4 \\ 0.3 & 0.4 & 1 \end{bmatrix}.$$

We will generate a synthetic multivariate time series with three interrelated components, compute the mean vector, covariance matrix, and visualize the relationships using scatter plots and cross-correlation plots. The Python code can be found online in GitHub.

1. **Simulating VAR(1) Process:** In this first code cell, we generate a multivariate time series using the VAR(1) model. This step simulates three variables with a predefined transition matrix.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.api import VAR

# Simulate VAR(1) time series
np.random.seed(42)
L = 1000  # length of time series
C = 3     # number of variables

# Transition matrix for VAR(1) process
A = np.array([[0.7, 0.1, 0.0],
              [0.3, 0.5, 0.2],
              [0.0, 0.2, 0.6]])

# Innovation (noise) covariance
cov = np.array([[1, 0.5, 0.3],
                [0.5, 1, 0.4],
                [0.3, 0.4, 1]])

# Generate random noise
noise = np.random.multivariate_normal(np.zeros(C), cov, L)
```

```

# Initialize the time series data
data = np.zeros((L, C))

# Simulate the VAR(1) process
for t in range(1, L):
    data[t] = A @ data[t-1] + noise[t]

# Create DataFrame
df = pd.DataFrame(data, columns=[’Y1’, ’Y2’, ’Y3’])

# Display the first few rows
df.head()
```

In this cell, we simulate a VAR(1) process for 1000 time steps and 3 variables, stored in the DataFrame `df`. The transition matrix A determines the interaction between variables, while the noise has a predefined covariance matrix. A sample path of this multivariate process is shown in Fig. 7, where each component is indicated with a different color.

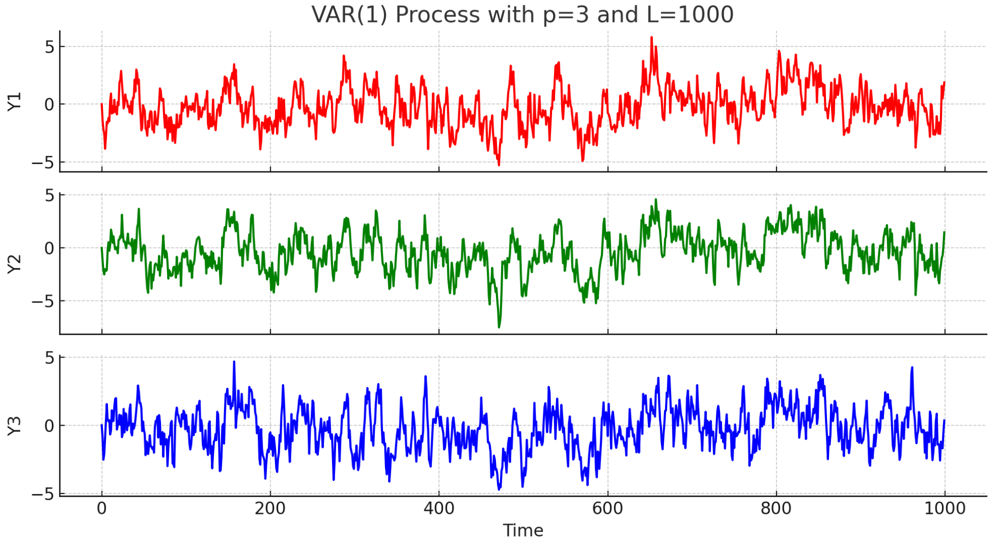

Figure 7: Sample path of VAR(1) with $C = 3$ and $L = 1,000$.

2. **Scatter Plots and Correlation Coefficients:** Scatter plots are used to visualize the pairwise relationships between the variables. The correlation coefficients are computed and displayed in the titles of the plots.

```
# Scatter plots between Y1, Y2, and Y3
plt.figure(figsize=(10, 4))

# Compute the Pearson correlation coefficients
corr_Y1_Y2 = df[’Y1’].corr(df[’Y2’])
corr_Y1_Y3 = df[’Y1’].corr(df[’Y3’])
corr_Y2_Y3 = df[’Y2’].corr(df[’Y3’])

# Y1 vs Y2
plt.subplot(1, 3, 1)
plt.scatter(df[’Y1’], df[’Y2’], alpha=0.5)
plt.title(f’Y1 vs Y2\nCorr: {corr_Y1_Y2:.2f}’)
plt.xlabel(’Y1’)
plt.ylabel(’Y2’)

# Y1 vs Y3
plt.subplot(1, 3, 2)
plt.scatter(df[’Y1’], df[’Y3’], alpha=0.5)
plt.title(f’Y1 vs Y3\nCorr: {corr_Y1_Y3:.2f}’)
plt.xlabel(’Y1’)
plt.ylabel(’Y3’)

# Y2 vs Y3
plt.subplot(1, 3, 3)
plt.scatter(df[’Y2’], df[’Y3’], alpha=0.5)
plt.title(f’Y2 vs Y3\nCorr: {corr_Y2_Y3:.2f}’)
plt.xlabel(’Y2’)
plt.ylabel(’Y3’)

plt.tight_layout()
plt.show()
```

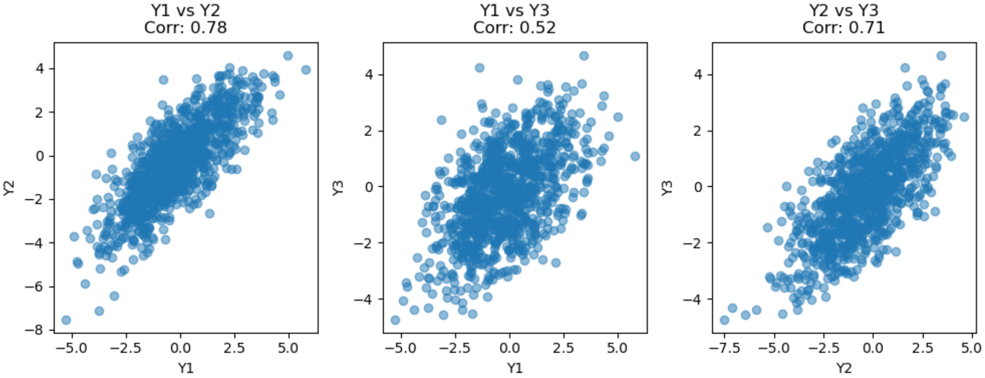

Figure 8: Scatter plots showing relationships between $\mathcal{Y}^1$, $\mathcal{Y}^2$, and $\mathcal{Y}^3$.

3. **Cross-Correlation Functions (CCF):** Finally, we compute and plot the cross-correlation functions for each pair of variables across different time lags using a matrix plot.

```
# Standardizing the series (z-scores)
df[’Y1_z’] = zscore(df[’Y1’])
df[’Y2_z’] = zscore(df[’Y2’])
df[’Y3_z’] = zscore(df[’Y3’])

# List of standardized variables
variables = [’Y1_z’, ’Y2_z’, ’Y3_z’]

# Create subplots for cross-correlation matrix
fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(12, 12))
max_lags = 20  # Maximum number of lags

# Plot cross-correlations for each pair of variables
for i, var1 in enumerate(variables):
    for j, var2 in enumerate(variables):
        if i == j:
            # Autocorrelation on the diagonal
```

```
                  axes[i, j].xcorr(df[var1], df[var2], maxlags=max_lags)
                  axes[i, j].set_title(f'Autocorrelation: {var1}')
            else:
                  # Cross-correlation for off-diagonal elements
                  axes[i, j].xcorr(df[var1], df[var2], maxlags=max_lags)
                  axes[i, j].set_title(f'Cross-correlation: {var1} & {var2}')

            axes[i, j].set_xlabel('Lag')
            axes[i, j].set_ylabel('Cross-correlation')

plt.tight_layout()
plt.show()
```

This cell produces a matrix of subplots that show both the autocorrelations (diagonal) and cross-correlations (off-diagonal) between the variables for different time lags. The cross-correlation function helps identify the lead-lag relationships between variables across different time lags, revealing important temporal dependencies.

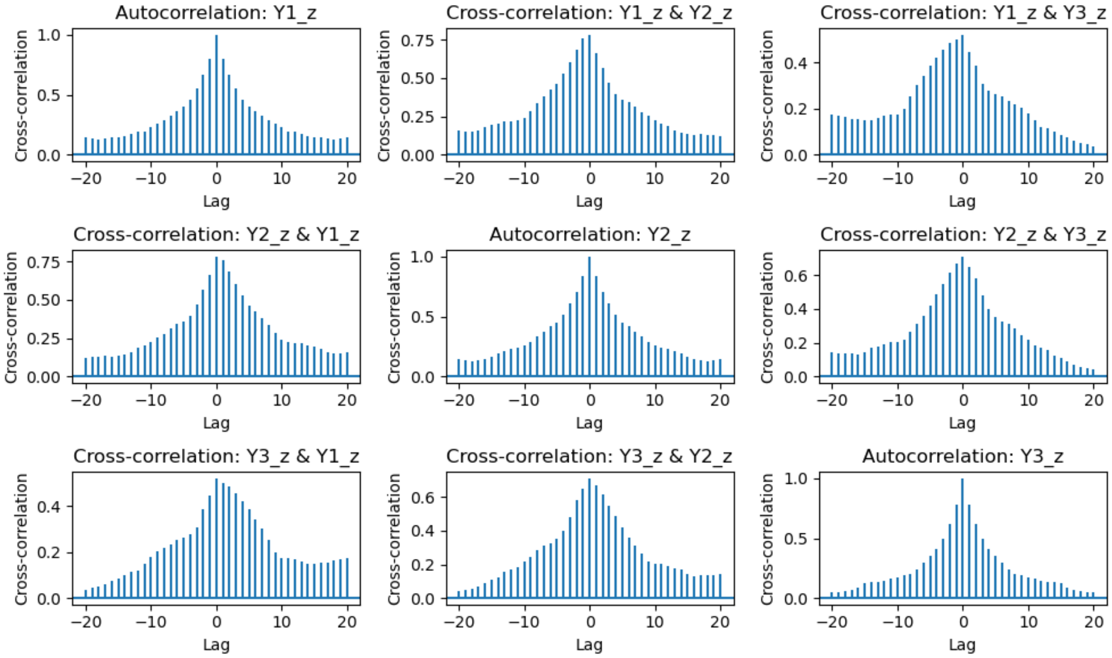

Figure 9: Matrix plot of the cross-correlations among Y_1, Y_2, and Y_3.

3.2 Stationarity in Multivariate Processes

We can extend the definition of stationarity from univariate to multivariate stochastic processes, as follows. A multivariate stochastic process $\mathcal{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_L\}$ is **strong-sense stationary** if, for any time indices $k_1, k_2, \ldots, k_n$ and for any integer h, the joint CDF satisfies:

$$F_{\mathcal{Y}}(\mathbf{y}_{k_1}, \mathbf{y}_{k_2}, \ldots, \mathbf{y}_{k_n}) = F_{\mathcal{Y}}(\mathbf{y}_{k_1+h}, \mathbf{y}_{k_2+h}, \ldots, \mathbf{y}_{k_n+h}),$$

for all $n \geq 1$ and for all possible time shifts h. This condition implies that the statistical properties of the process, as captured by the joint CDF, are invariant to shifts in time.

A multivariate stochastic process $\mathcal{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_n\}$ is said to be **weak-sense stationary** (or wide-sense stationary) if the mean vector is a time-invariant vector and the autocovariance matrix depend only on the time difference between the samples, not on absolute time. Formally, the process is weak-sense stationary if:

1. The mean vector $\mathbb{E}[\mathbf{Y}(k)] = \boldsymbol{\mu}_{\mathcal{Y}}$ is constant for all k.
2. The autocovariance matrix

   $$\Sigma_{\mathcal{Y}}(k_1, k_2) = \mathbb{E}[(\mathbf{Y}(k_1) - \boldsymbol{\mu}_{\mathcal{Y}})(\mathbf{Y}(k_2) - \boldsymbol{\mu}_{\mathcal{Y}})^{\intercal}]$$

   depends only on the time difference $k_1 - k_2$. Whenever clear from the context, we will simplify the notation of the covariance matrix to $\Sigma_{\mathcal{Y}}(h)$, where $h = |k_1 - k_2|$ denotes the lag between samples.

Thus, in a weak-sense stationary process, the first two moments (mean and covariance) are time-invariant, but higher-order moments may still vary with time. This weaker form of stationarity is often sufficient for many practical applications, especially in the context of linear modeling and time series analysis.

3.3 Multivariate Markov Processes

A **multivariate Markov process** is a generalization of the classical Markov process to systems involving multiple interrelated stochastic variables evolving over time. In this framework, the future evolution of the system depends only on the current state, not on the past history. Let $\{\boldsymbol{Y}_k \colon k \in \mathbb{N}\}$ be a vector-valued stochastic process, where each $\boldsymbol{Y}_k \in \mathbb{R}^p$ is a p-dimensional vector, with p being the number of interrelated components of the multivariate process. The process $\boldsymbol{Y}_k$ is said to be a multivariate Markov process if it satisfies the *Markov property*, defined as:

$$\Pr(\boldsymbol{Y}_{k+1} \leq \mathbf{y}_{k+1} \mid \boldsymbol{Y}_k = \mathbf{y}_k, \boldsymbol{Y}_{k-1} = \mathbf{y}_{k-1}, \ldots, \boldsymbol{Y}_1 = \mathbf{y}_1) = \Pr(\boldsymbol{Y}_{k+1} \leq \mathbf{y}_{k+1} \mid \boldsymbol{Y}_k = \mathbf{y}_k),$$

for all $k \in \mathbb{N}$, where the inequality $\boldsymbol{Y}_{k+1} \leq \mathbf{y}_{k+1}$ is understood component-wise. This means that the conditional distribution of the future state $\boldsymbol{Y}_{k+1}$ depends only on the current state $\boldsymbol{Y}_k$ and not on the entire past history $\boldsymbol{Y}_{k-1}, \boldsymbol{Y}_{k-2}, \ldots, \boldsymbol{Y}_1$. In a multivariate setting, the Markov process describes how the entire vector $\boldsymbol{Y}_k = [Y_k^1, Y_k^2, \ldots, Y_k^C]^{\intercal}$ evolves, where each component Y_k^i represents the value of the i-th component at time k.

A (scalar-valued) higher-order Markov process of **order** m is one in which the future state depends on the previous m states (see Subsection 2.3.1). Such a process can always be transformed into an equivalent **first-order vector-valued Markov process**. For simplicity, let us illustrate this transformation with a simple example.

Example 8: AR(2) as a VAR(1) Process

Consider a (scalar-valued) AR(2) process, defined as:

$$Y_{k+1} = \phi_0 Y_k + \phi_1 Y_{k-1} + \epsilon_k, \tag{1}$$

where $\epsilon_k \sim \mathcal{N}(0, \sigma^2)$ is a white noise process. This is a second-order Markov process, since the future value depends on the previous two observations; hence, the order parameter is $m = 2$.

We can transform this AR(2) process into a vector-valued (first-order) Markov process by defining a vector-valued process $\boldsymbol{\mathcal{X}} = \{\mathbf{X}_k \colon k \in \mathbb{N}\}$, where $\mathbf{X}_k = [Y_k, Y_{k-1}]^\intercal$, i.e., it contains all the previous two values of the time series. We also define the following first-order vector autoregressive process VAR(1), as follows:

$$\mathbf{X}_{k+1} = \mathbf{A}\mathbf{X}_k + \boldsymbol{\epsilon}_k, \tag{2}$$

where $\mathbf{A}$ is the following transition matrix:

$$\mathbf{A} = \begin{bmatrix} \phi_0 & \phi_1 \\ 1 & 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\epsilon}_k = \begin{bmatrix} \epsilon_k \\ 0 \end{bmatrix}.$$

The VAR(1) process is an example of a first-order multivariate Markov process, since $\mathbf{X}_{k+1}$ depends solely on the vector $\mathbf{X}_k$ through the transition matrix $\mathbf{A}$, plus a random noise component $\boldsymbol{\epsilon}_k$.

We now demonstrate the equivalence between second-order Markov process AR(2) and vector-valued first-order Markov model VAR(1). To achieve this, we expand the vector equation (2) into two scalar-valued recursions:

$$X_{k+1,1} = \phi_0 X_{k,1} + \phi_1 X_{k,2} + \epsilon_k, \tag{3}$$

$$X_{k+1,2} = X_{k,1}. \tag{4}$$

Substituting the second equation into the first one, one can verify that the obtained recursion in terms of $X_{k+1,1}$ is identical to the recursion defined in (1) by the simple change of variables $Y_k = X_{k,1}$.

More generally, any higher-order scalar process can be transformed into an equivalent first-order vector-valued Markov process by appropriately expanding the state space to include past observations. This transformation enables the use of powerful multivariate time series techniques, such as vector autoregressive models (VAR), to analyze the process. By recasting a higher-order dependence structure in terms of a first-order vector representation, we capture the same dynamics within a multivariate framework, providing a more flexible and comprehensive set of tools for studying interactions, forecasting, and understanding the behavior of complex time-dependent systems. This approach is particularly advantageous in fields where relationships between multiple variables evolve over time and higher-order dependencies need to be managed efficiently.

3.4 Causality in Multivariate Time Series

In multivariate time series analysis, the concept of **causality** is fundamental in understanding the dynamic interactions between multiple components. Specifically, we aim to determine whether the past values of certain components within a multivariate process can help predict the future values of other components. This is particularly important in systems where the components exhibit complex interdependencies over time, such as in climate data or energy consumption patterns.

To formalize the concept of causality, consider a multivariate time series $\boldsymbol{\mathcal{Y}}$ as a collection of C scalar-valued time series of equal length L, denoted as $\{\mathcal{Y}^1, \mathcal{Y}^2, \ldots, \mathcal{Y}^C\}$. Each component $\mathcal{Y}^c$, where $c \in \{1, 2, \dots, C\}$, represents a distinct time series. For example, $\mathcal{Y}^1$ might represent energy consumption, $\mathcal{Y}^2$ could correspond to temperature, and $\mathcal{Y}^3$ might track humidity.

Establishing causality poses significant challenges with conventional statistical methods, which typically excel at identifying correlations but fall short in discerning the direction or nature of influence. Correlation merely captures co-movement between variables, without revealing whether one variable drives changes in another. To establish a causal relationship, we must demonstrate that changes in one variable lead to changes in another. The gold standard for such an analysis is the randomized controlled trial (RCT), where an intervention is deliberately introduced to isolate the causal effect of one variable on another. However, RCTs are often costly, difficult to implement, or even infeasible in certain domains, both practically and ethically.

Granger Causality

In the absence of direct experimentation, **Granger causality** offers a tractable, alternative framework for inferring causal relationships from time series data. The central idea behind Granger causality is *predictive relevance*: if the past values of one time series $\mathcal{Y}^i$ enhance the prediction of another time series $\mathcal{Y}^j$, beyond what can be achieved using only the past values of $\mathcal{Y}^j$ itself, then $\mathcal{Y}^i$ is said to **Granger-cause** $\mathcal{Y}^j$.

To formalize this, let Y_k^i represent the k-th sample of the time series $\mathcal{Y}^i$, and define the **information set** (also known as a *filtration*[6]) as $\mathcal{F}_k^i = \{Y_1^i = y_1^i, Y_2^i = y_2^i, \dots, Y_k^i = y_k^i\}$, which represents the past values of $\mathcal{Y}^i$ up to time k. Granger causality between two stochastic processes $\mathcal{Y}^i$ and $\mathcal{Y}^j$ is formally defined as follows: $\mathcal{Y}^i$ Granger-causes $\mathcal{Y}^j$ if the conditional probability distribution of Y_{k+1}^j, given both information sets $\mathcal{F}_k^i$ and $\mathcal{F}_k^j$, differs from the conditional distribution of Y_{k+1}^j given only $\mathcal{F}_k^j$. In mathematical terms, this implies that there exists some $y \in \mathbb{R}$ such that:

$$\Pr\left(Y_{k+1}^j \leq y \mid \mathcal{F}_k^j, \mathcal{F}_k^i\right) \neq \Pr\left(Y_{k+1}^j \leq y \mid \mathcal{F}_k^j\right).$$

In simpler terms, $\mathcal{Y}^i$ *Granger-causes* $\mathcal{Y}^j$ if incorporating the information set $\mathcal{F}_k^i$, i.e., the past values of $\mathcal{Y}^i$, changes the conditional distribution of future values of $\mathcal{Y}^j$.

It is crucial to note that Granger causality does not imply *true* causality in the philosophical or scientific sense. Rather, it reflects the predictive power of one time series over another, indicating temporal precedence and predictability without accounting for possible confounding variables or underlying mechanisms that may affect both series. Establishing true causality often requires experimental intervention or deep insights into the system's underlying dynamics, where direct manipulation of variables is feasible. Granger causality, in contrast, is a statistical construct that captures the structure of the data without necessarily implying a direct cause-and-effect relationship.

Detecting Granger Causality

In the context of wide-sense stationary multivariate time series, detecting Granger causality is frequently accomplished using **Vector Autoregressive (VAR)** models. A VAR model captures the dependencies of each variable on both its own past values and the past values of other components

[6] A *filtration* is a sequence of growing information sets over time, where each set contains all information available up to that point. It embodies the idea that more information becomes available as time progresses. For a formal treatment of filtrations, see [2]

in the multivariate system. Formally, a VAR(m) model for a multivariate time series is given by the following recursion:

$$\mathbf{Y}_{k+1} = A_0\mathbf{Y}_k + A_1\mathbf{Y}_{k-1} + \cdots + A_m\mathbf{Y}_{k-m} + \boldsymbol{\epsilon}_k,$$

where m is the '*memory*' of the VAR process[7], $A_0, A_1, \ldots, A_m$ are $C \times C$ coefficient matrices, and $\boldsymbol{\epsilon}_k$ is a C-dimensional vector of white noise innovations, with $\boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \Sigma_\epsilon)$. To test whether the time series component $\mathcal{Y}^i$ Granger-causes $\mathcal{Y}^j$, two models are compared:

1. The **unrestricted model**, which predicts the values of Y_{k+1}^j using a linear combination of all available information, represented by the superset $\cup_{c=1}^{C}\mathcal{F}_k^c$, i.e., all the information from all components up to time k:

$$Y_{k+1}^j = \alpha_0 + \sum_{h=0}^{m}\sum_{c=1}^{C}\alpha_{hc}^j Y_{k-h}^c + \epsilon_k^j,$$

2. The **restricted model**, which excludes the information available in $\mathcal{F}_k^i$ from the recursion:

$$Y_{k+1}^j = \alpha_0 + \sum_{h=0}^{m}\sum_{c\neq i}\alpha_{hc}^j Y_{k-h}^c + \epsilon_k^j.$$

The null hypothesis H_0 is that $\mathcal{Y}^i$ does not Granger-cause $\mathcal{Y}^j$, meaning that the information in $\mathcal{F}_k^i$ does not provide any additional predictive power for Y_{k+1}^j. Formally, we test:

$$H_0 : \alpha_{hi}^j = 0 \text{ for all} h = 0, 1, \ldots, m \quad \text{(in the unrestricted model)}.$$

The statistical test for Granger causality is essentially a standard hypothesis test for a linear model. We employ an **F-test** to compare the residual sum of squares (RSS) between the restricted and unrestricted models. If the F-statistic is significant, we reject the null hypothesis H_0, leading to the conclusion that $\mathcal{Y}^i$ Granger-causes $\mathcal{Y}^j$. It is crucial to recognize that rejecting H_0 indicates that the past values of $\mathcal{Y}^i$ provide significant additional predictive information for $\mathcal{Y}^j$, beyond what can be predicted using the past values of $\mathcal{Y}^j$ alone. However, this result does not imply that $\mathcal{Y}^i$ directly causes $\mathcal{Y}^j$ in a causal or mechanistic sense. Granger causality merely identifies a temporal predictive relationship, and it is possible that other unobserved variables or confounding factors influence both series. Therefore, Granger causality should be understood as a statistical association reflecting predictive power, rather than definitive proof of a direct cause-and-effect relationship.

[7] In coming chapters, we will cover cross-validation techniques to find an appropriate value for this hyperparameter.

Appendix. Use of the Python Software

The Python software for general-purpose computing and data analysis has become a popular choice for time series analysis, statistical computing, and the development of new algorithms. Python is available as free and open-source software under the terms of the Python Software Foundation License. It runs on all major operating systems including Windows, macOS, and Linux. The main website for the Python project is `https://www.python.org`.

The Python environment consists of a base system, which includes the Python language interpreter, and a large ecosystem of user-contributed libraries. The base system provides the core functionality of Python, including built-in types and functions. Additionally, there are thousands of libraries available for download that extend the functionality of Python. These libraries cover various fields, such as numerical computing, data visualization, and machine learning.

For time series modeling and forecasting, some of the most useful Python libraries include:

- `NumPy`: A library for numerical computing, providing support for arrays, matrices, and a large collection of mathematical functions. `https://numpy.org/`
- `Pandas`: A powerful data manipulation and analysis library that includes support for time series data. `https://pandas.pydata.org/`
- `Statsmodels`: A library that provides classes and functions for the estimation of many different statistical models, including time series models like ARIMA. `https://www.statsmodels.org/`
- `SciPy`: A library used for scientific computing, providing tools for optimization, integration, interpolation, and statistics. `https://scipy.org/`
- `Matplotlib`: A plotting library used for creating static, interactive, and animated visualizations in Python. `https://matplotlib.org/`

Running Python Code in the Cloud (Google Colab)

Google Colab is a free cloud service that allows you to write and execute Python code in a web-based Jupyter notebook environment. One of the main benefits of Colab is that it requires no setup, and it provides free access to GPUs, making it an excellent option for heavy computations such as machine learning and time series forecasting.

To start using Google Colab:

1. Navigate to `https://colab.research.google.com`.
2. Sign in with your Google account (if not already signed in).
3. Create a new notebook by clicking `File > New Notebook`.
4. In the notebook, you can write Python code directly in cells and run it by clicking the ”Run” button or using the keyboard shortcut `Shift + Enter`.

To install and import libraries in Colab, use the following commands. For example, to install `pandas`, run:

```
!pip install pandas
```

Once installed, the library can be imported as usual:

```
import pandas as pd
import numpy as np
```

Colab also allows you to upload files directly into the environment or access them from Google Drive, making it highly flexible for data science tasks.

Running Python Code Locally on Your Computer

To run Python locally, you first need to install the Python interpreter and a suitable Integrated Development Environment (IDE), such as `Jupyter Notebook`, `VSCode`, or `PyCharm`.

1. Installing Python

The easiest way to install Python and manage packages is by using the `Anaconda` distribution, which comes with Python, `pip`, and most of the essential libraries for data analysis and scientific computing. To install Anaconda:

1. Go to `https://www.anaconda.com/products/individual`.
2. Download the installer for your operating system (Windows, macOS, or Linux).
3. Follow the installation instructions on the Anaconda website.

After installation, you can launch the `Anaconda Navigator` to open Jupyter Notebooks or use `Anaconda Prompt` to run Python code from the command line.

2. Installing Libraries

Once Python is installed, you can install additional libraries using `pip`, Python's package manager. For example, to install `pandas` and `statsmodels`, open a terminal or command prompt and run the following commands:

```
pip install pandas
pip install statsmodels
```

To verify that Python and the necessary libraries have been installed correctly, you can start a Python session and import the libraries:

```
import pandas as pd
import statsmodels.api as sm
```

3. Running Python Code in a Jupyter Notebook

Once Anaconda is installed, you can launch a `Jupyter Notebook` by opening the `Anaconda Navigator` and selecting `Jupyter Notebook`. This will open a web browser where you can create new notebooks and execute Python code in cells. Here's an example of loading a dataset in a notebook:

```python
import pandas as pd

# Load data from CSV
data = pd.read_csv('yourfile.csv')

# Display the first few rows
print(data.head())
```

For statistical analysis, the `statsmodels` library provides various time series models, including ARIMA, and functions to evaluate model performance. To fit an ARIMA model to a time series, you can use the following code:

```python
from statsmodels.tsa.arima.model import ARIMA

# Fit an ARIMA model
model = ARIMA(data, order=(1,1,1))
fit_model = model.fit()

# Output model summary
print(fit_model.summary())
```

Documentation and Learning Resources

The documentation for Python and its libraries is extensive and available online. Python's official documentation, tutorials, and FAQs can be accessed at `https://docs.python.org`. The documentation for major libraries such as `NumPy`, `Pandas`, and `Statsmodels` is also available on their respective websites.

Exercises

1. Consider a random process defined by the recursion: $Y_{k+1} = \alpha k + \phi Y_k + \epsilon_k$, where ϵ_k is a white noise with mean zero and constant variance σ_ϵ^2. Answer the following questions:

   (a) Compute the theoretical mean and variance of the process as a function of k.

   (b) Compute the autocorrelation function of the process for large k (i.e., the process is stationary).

   (c) Is the process wide-sense stationary? Explain your answer.

   (d) Is the process strong-sense stationary? Explain your answer.

   (e) Program a piece of Python code to generate and plot 10 sample paths of length 100 of the random process. Include a shaded area indicating the 95% confidence interval of the process.

2. Consider a random process Y_k defined by the recursion:

   $$Y_k = 1 + \epsilon_k + \theta_1 \epsilon_{k-1} \tag{5}$$

   where $\epsilon(k)$ are independent and identically distributed (i.i.d.) random variables, each following a standard normal distribution $\mathcal{N}(0, 1)$. Answer the following questions:

   (a) Compute the theoretical mean $\mu(k)$ and variance $\sigma^2(k)$ as a function of k.

   (b) Compute the theoretical autocovariance $\text{Cov}(Y_k, Y_{k-h})$.

   (c) Is the random process strong-sense stationary? Explain your answer.

   (d) Program a piece of Python code to generate and plot 10 sample paths of length 100 of the random process. Include a shaded area indicating the 95% confidence interval of the process.

3. Let $\alpha \sim \mathcal{N}(0, 1)$ and $\beta \sim \mathcal{N}(0, 2)$ be two independent random variables (independent of k). Define the random process $\mathcal{Y} = \{Y_k : k \in \mathbb{N}\}$ as the recursion:

   $$Y_k = \alpha + \beta k + k^2 \text{ for all } k \in \mathbb{N}.$$

   Find expressions for the mean, the variance, and the autocovariance of the random process as a function of k, as well as the lag h for the covariance.

4. Consider a fair coin toss game where you start with \$10. Every time you flip the coin:

   - If it lands heads, you win \$1.
   - If it lands tails, you lose \$1.

   Let Y_k represent the amount of money you have after k coin flips, with $Y_0 = 10$. For each coin flip, the outcome is independent of previous flips, and the probability of heads or tails is 1/2. Answer the following questions

   (a) Find the expected value of Y_k, i.e., the amount of money you will have after k coin flips.

   (b) Compute the variance of Y_k after k coin flips.

(c) Program a piece of Python code to generate and plot 10 sample paths of length 100 of the random process. Include a shaded area indicating the 95% confidence interval of the process (assume that the distribution of Y_k is approximately normal).

5. Consider the AR(2) process defined in Example 7. Derive an expression for the probability density function:

$$f_{Y_k|\mathcal{F}_{k-1}}(Y_k \mid Y_{k-1} = y_{k-1}, Y_{k-2} = y_{k-2}, Y_{k-3} = y_{k-3}, \dots).$$

6. Consider the AR(3) process defined by the recursion:

$$Y_{k+1} = \phi_0 Y_k + \phi_1 Y_{k-1} + \phi_2 Y_{k-2} + \epsilon_k,$$

where $\epsilon_k \sim \mathcal{N}(0, \sigma^2)$ is a white noise process. This process is autoregressive of order 3, meaning it depends on the previous three observations.

Answer the questions below:

(a) What is the order of this higher-order Markov process? Justify your answer.

(b) What is the conditional distribution of Y_{k+1} given all the previous observations?

(c) Rewrite the AR(3) process as a vector-valued first-order Markov process. Define the AR(3) process as a vector-valued recursion using a transition matrix and provide an explicit expression for the covariance matrix of the noise vector.

References

[1] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[2] Rick Durrett. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.