LW
Linlin WANG AUTHOR | TEACHER

Created Sep 11 11:41am | Posted Sep 11 11:41am | Last edited Sep 11 11:42am

Notice of Course Schedule Adjustment

Dear students,

We are making a temporary adjustment to the class schedule. **Next week's sessions will be moved to September 15th.**

We will have our classes on September 15th. I will set up a zoom section and upload course recording as well after the class in case you have time conflict.

Please note the following adjustment details:

Date: Sep. 15 (Mon) --2025/09/15
Time: 4:00pm to 6:50pm
Venue: M3017 （Run Run Shaw Creative Media Centre 邵逸夫創意媒體中心）

This adjustment applies only to next week.

Lecture 2
Stationarity and autoregressive models

- **Time series**

- **Stochastic process**

- **Measure of dependence**

  Mean function

  Autocovariance function

Review

- ***A time series*** **is a series of data points indexed in time order.**

- **A time series can be defined as a collection of random variables indexed according to the order they are obtained in time.**

- **A collection of random variables, $\{x_t\}$, indexed by t is referred to as a *stochastic process*.**

Review

- **Example:**

  White noise: $w_t \sim \mathrm{wn}(0, \sigma_w^2)$

  Moving averages and filtering: $v_t = \frac{1}{3}(w_{t-1} + w_t + w_{t+1})$

  Random walk with drift: $x_t = \delta + x_{t-1} + w_t$

  Signal in noise: $x_t = A\cos(2\pi\omega t + \phi) + w_t$

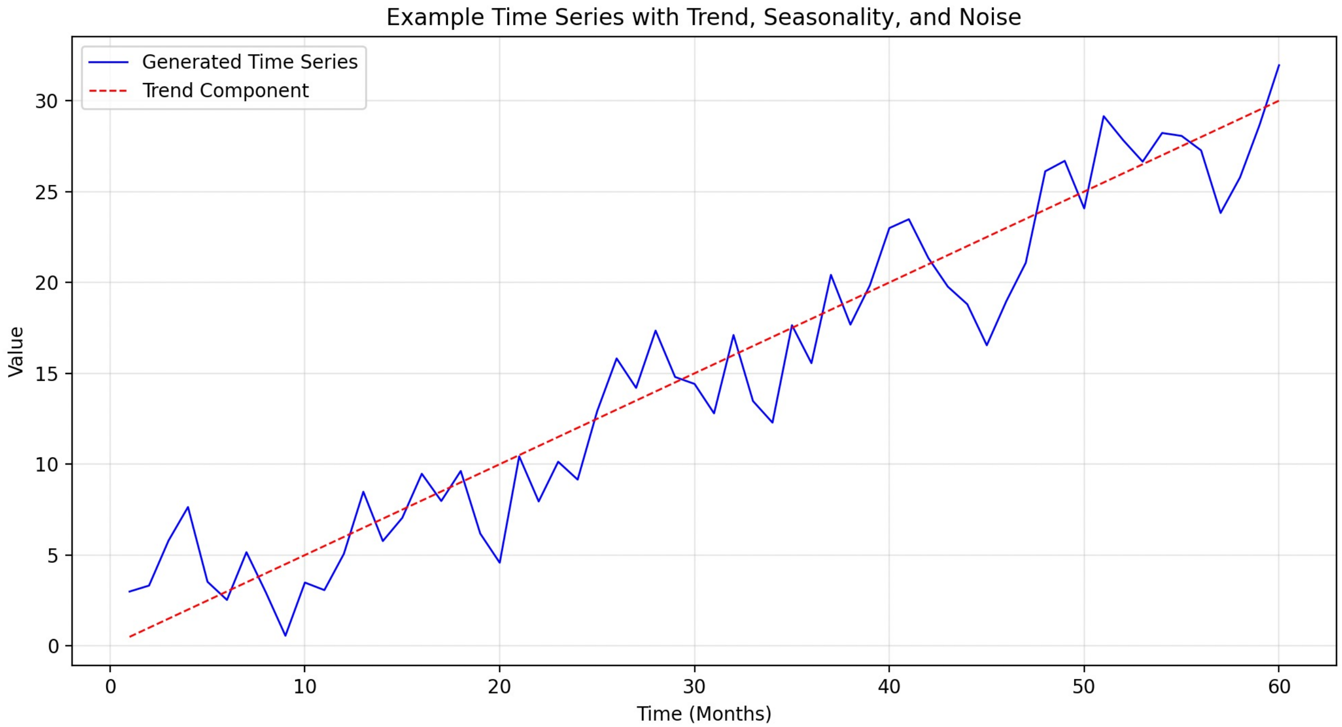
the process of detecting, extracting, or enhancing valuable information of genuine interest from background noise or interference.

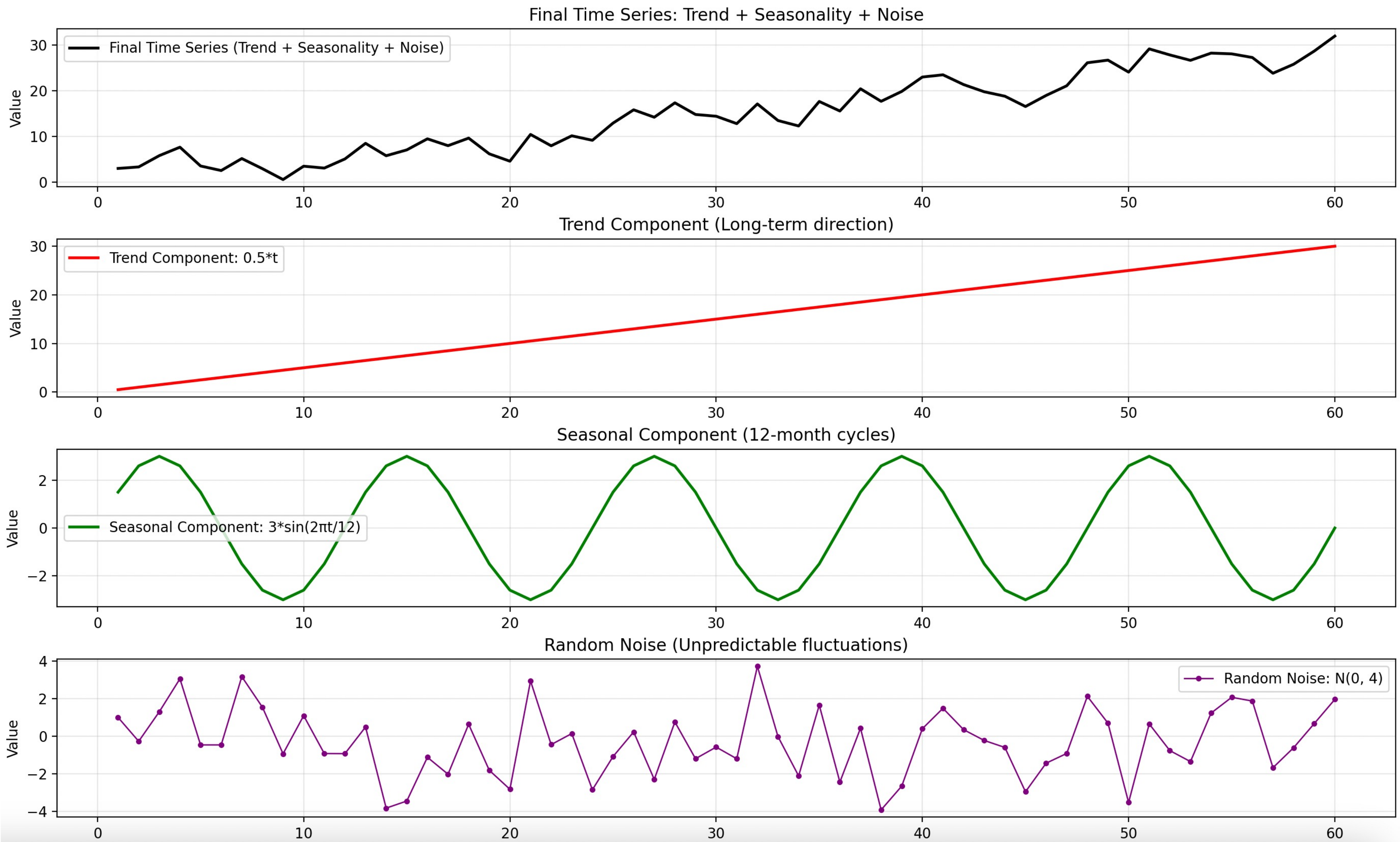
Review

- **Typically, a time series model can be described as**

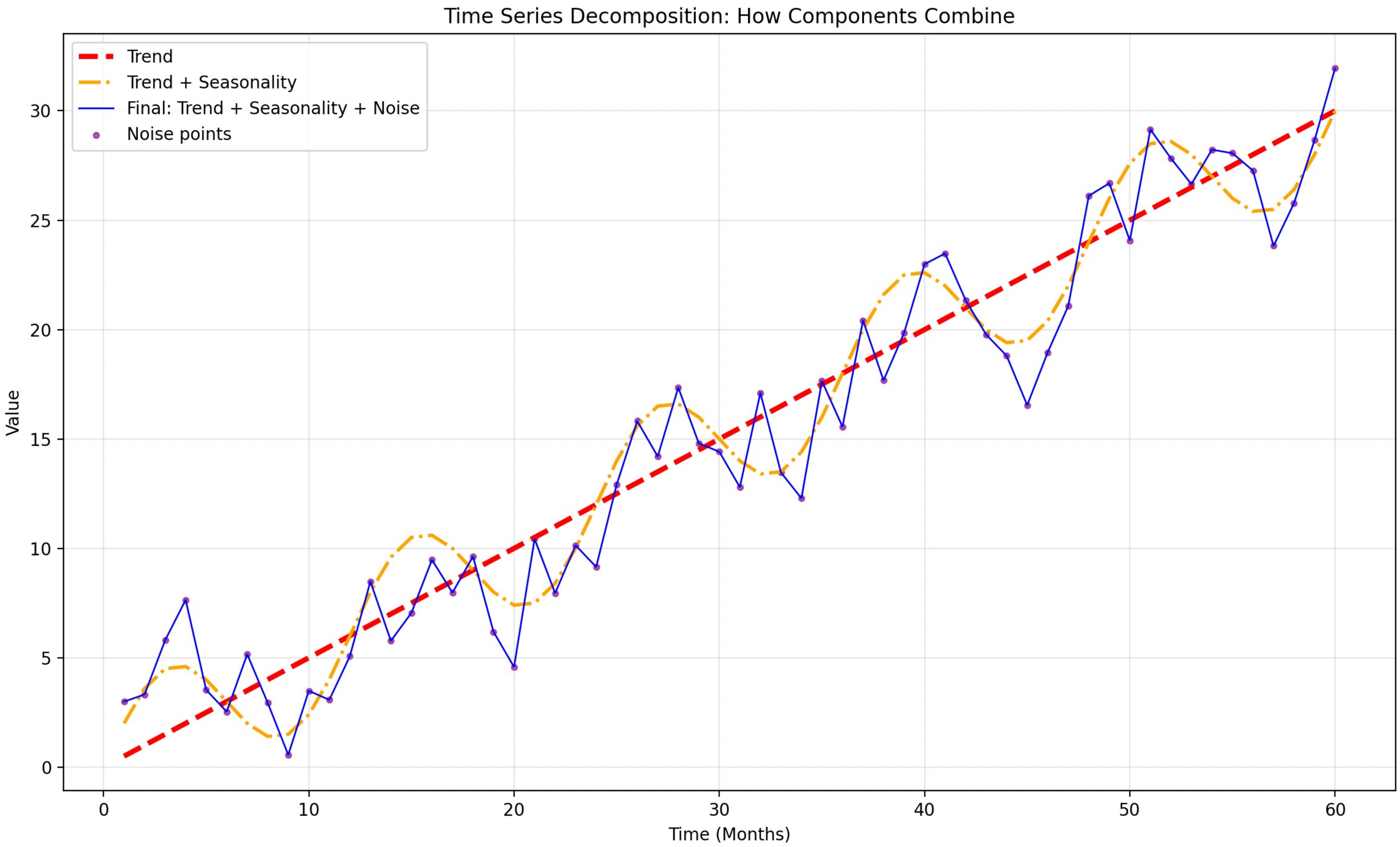
$$\underbrace{x_t = m_t}_{\text{trend component}} + \underbrace{s_t}_{\text{seasonal component}} + \underbrace{e_t}_{\text{residual}}$$

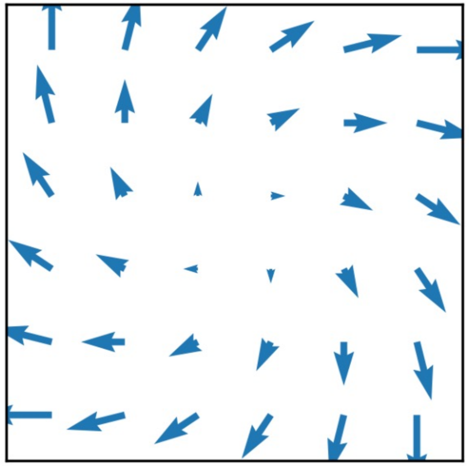
```python
import numpy as np
import matplotlib.pyplot as plt

# 设置随机种子以确保结果可重现
np.random.seed(42)

# 生成时间索引（60个月）
t = np.arange(1, 61)

# 1. 趋势成分：线性趋势
trend = 0.5 * t

# 2. 季节性成分：12个月周期
seasonality = 3 * np.sin(2 * np.pi * t / 12)

# 3. 随机噪声
noise = np.random.normal(0, 2, size=len(t))

# 组合所有成分
time_series = trend + seasonality + noise

# 绘制时间序列图
plt.figure(figsize=(12, 6))
plt.plot(t, time_series, label='Generated Time Series', color='blue', linewidth=1)
plt.plot(t, trend, label='Trend Component', color='red', linestyle='--', linewidth=1)
plt.title('Example Time Series with Trend, Seasonality, and Noise')
plt.xlabel('Time (Months)')
plt.ylabel('Value')
plt.legend()
plt.grid(True, alpha=0.3)
plt.show()
```





Final Time Series: Trend + Seasonality + Noise
Final Time Series (Trend + Seasonality + Noise)
Value
Trend Component (Long-term direction)
Trend Component: 0.5*t
Value
Seasonal Component (12-month cycles)
Seasonal Component: 3*sin(2πt/12)
Value
Random Noise (Unpredictable fluctuations)
Random Noise: N(0, 4)
Value


Time Series Decomposition: How Components Combine
Trend
Trend + Seasonality
Final: Trend + Seasonality + Noise
Noise points
Value
Time (Months)
0
5
10
15
20
25
30
0
10
20
30
40
50
60



quiver(X, Y, U, V)

Matplotlib: Visualization with Python

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

- Create publication quality plots.
- Make interactive figures that can zoom, pan, update.
- Customize visual style and layout.
- Export to many file formats.
- Embed in JupyterLab and Graphical User Interfaces.
- Use a rich array of third-party packages built on Matplotlib.

Try Matplotlib (on Binder) →

Getting Started

Examples

Reference

Cheat Sheets

Documentation

NumPy

The fundamental package for scientific computing with Python

LATEST RELEASE: NUMPY 2.3. VIEW ALL RELEASES

NumPy 2.3.0 released! 2025-06-07

Powerful N-dimensional arrays

Fast and versatile, the NumPy vectorization, indexing, and broadcasting concepts are the de-facto standards of array computing today.

Numerical computing tools

NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.

Open source

Distributed under a liberal BSD license, NumPy is developed and maintained publicly on GitHub by a vibrant, responsive, and diverse community.

Interoperable

NumPy supports a wide range of hardware and computing platforms, and plays well with distributed, GPU, and sparse array libraries.

Performant

The core of NumPy is well-optimized C code. Enjoy the flexibility of Python with the speed of compiled code.

Easy to use

NumPy's high level syntax makes it accessible and productive for programmers from any background or experience level.

SciPy

Fundamental algorithms for scientific computing in Python

GET STARTED

SciPy 1.16.1 released! 2025-07-27

Fundamental algorithms

SciPy provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics and many other classes of problems.

Broadly applicable

The algorithms and data structures provided by SciPy are broadly applicable across domains.

Foundational

Extends NumPy providing additional tools for array computing and provides specialized data structures, such as sparse matrices and k-dimensional trees.

Performant

SciPy wraps highly-optimized implementations written in low-level languages like Fortran, C, and C++. Enjoy the flexibility of Python with the speed of compiled code.

Easy to use

SciPy's high level syntax makes it accessible and productive for programmers from any background or experience level.

Open source

Distributed under a liberal BSD license, SciPy is developed and maintained publicly on GitHub by a vibrant, responsive, and diverse community.

Review

- **Mean function of** $\{x_t\}$

$$\mu_t = \mathrm{E}(x_t) = \int_{-\infty}^{\infty} x f_t(x) dx \text{ (provided it exist)}$$

- **Autocovariance function of** $\{x_t\}$

$$\gamma(s,t) = \mathrm{Cov}(x_s, x_t) = \mathrm{E}[(x_s - \mu_s)(x_t - \mu_t)]$$

How to calculate?

X: 1, 2, 3, 4, 5

Y: 2, 4, 6, 8, 10

Mean(X) = (1 + 2 + 3 + 4 + 5) / 5 = 3

Mean(Y) = (2 + 4 + 6 + 8 + 10) / 5 = 6

The discrete random variable X

$$E(X) = \Sigma\, [x_i * P(x_i)]$$

where $\mathbf{x_i}$ represents the **i-th possible outcome** that the random variable X can take.

P($\mathbf{x_i}$) denotes the **probability** that this outcome occurs.

Σ indicates the **summation** over all possible outcomes i.

How to calculate?

$E(X) = \Sigma [x_i * P(x_i)]$

Suppose you are playing a dice-rolling game with a fair six-sided die.

The number facing up is your score.

Step 1: List all possible outcomes (x_i) and their corresponding probabilities ($P(x_i)$)
Possible outcomes: 1, 2, 3, 4, 5, 6
Since the die is fair, the probability of each outcome is 1/6.

Step 2: Calculate the weighted value for each outcome [$x_i * P(x_i)$]
Multiply each outcome by its corresponding probability.

How to calculate?

$$E(X) = \Sigma [x_i * P(x_i)]$$

Outcome (x_i)	Probability ($P(x_i)$)	Weighted Value ($x_i * P(x_i)$)
1	1/6	1 * 1/6 = 1/6
2	1/6	2 * 1/6 = 2/6
3	1/6	3 * 1/6 = 3/6
4	1/6	4 * 1/6 = 4/6
5	1/6	5 * 1/6 = 5/6
6	1/6	6 * 1/6 = 6/6

Step 3: Sum all the weighted values (Σ)

Add all the weighted values together to get the expected value E(X):

$E(X) = (1/6) + (2/6) + (3/6) + (4/6) + (5/6) + (6/6)$

$= (1 + 2 + 3 + 4 + 5 + 6) / 6$

$= 21 / 6$

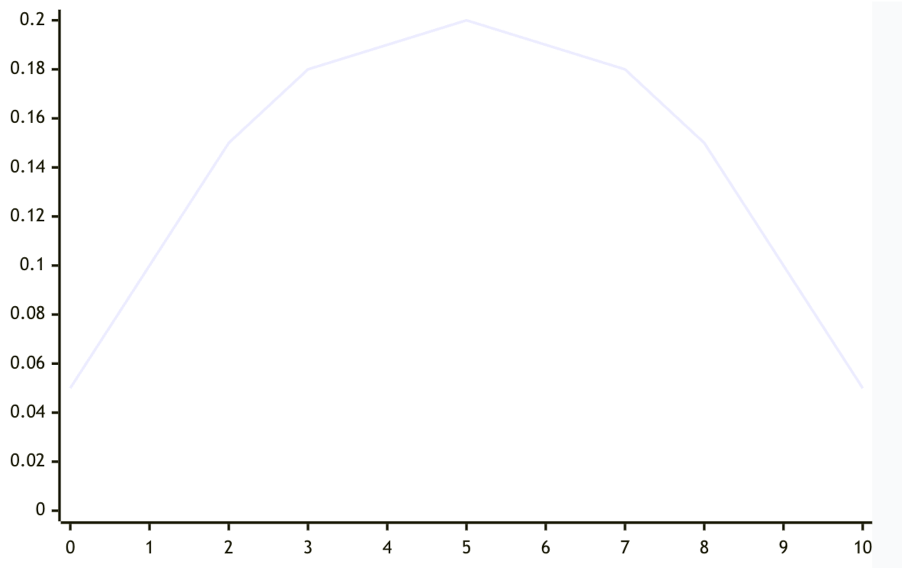
$= 3.5$

How to calculate?

Continuous random variable

$$E(X) = \int_{-\infty}^{\infty} x \cdot f(x)dx$$

Probability Density Function *f(x)*



The fundamental difference between a continuous random variable (e.g., a person's height, temperature, the time required to complete a task) and a discrete variable lies in the fact that its possible values are infinitely uncountable. Therefore, we cannot discuss the probability of a specific value (in fact, the probability of any single point is 0). Instead, we discuss the probability of the value falling within a certain interval.

The formula for calculating the expected value of a continuous random variable is a natural extension of the formula for discrete variables. It replaces the summation symbol Σ with the integral symbol ∫ and replaces the probability mass function $P(x_i)$ with the probability density function (PDF) f(x).

How to calculate? $E(X) = \int_{-\infty}^{\infty} x \cdot f(x)dx$

Calculation Steps (Illustrated with an Example)

Suppose a continuous random variable X represents the time you wait for a bus (in minutes). It follows a uniform distribution over the interval [0, 10] (meaning the probability of arriving at any time point is the same). Its probability density function is:

$$f(x) = \begin{cases} \frac{1}{10} & \text{for } 0 \leq x \leq 10 \\ 0 & \text{otherwise} \end{cases}$$

PDF f(x)=1/10 $E(X) = \frac{1}{10}\int_0^{10} x dx$

$$\int_0^{10} x dx = \left[\frac{1}{2}x^2\right]_0^{10}$$

$$E(X) = \int_0^{10} x \cdot \frac{1}{10} dx$$

$$\left[\frac{1}{2}(10)^2\right] - \left[\frac{1}{2}(0)^2\right] = \frac{1}{2} \times 100 - 0 = 50$$

$$E(X) = \frac{1}{10} \times 50 = 5$$

How to calculate?

- Covariance is a measure of how two random variables change together. It consists of

Positive Covariance

Negative Covariance

Near-Zero Covariance

$$\mathrm{Cov}(X, Y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu_X)(y_i - \mu_Y)$$

How to calculate?

X: 1, 2, 3, 4, 5

Y: 2, 4, 6, 8, 10

Mean(X) = (1 + 2 + 3 + 4 + 5) / 5 = 3

Mean(Y) = (2 + 4 + 6 + 8 + 10) / 5 = 6

$$\mathrm{Cov}(X, Y) = [(1-3)(2-6) + (2-3)(4-6) + (3-3)(6-6) + (4-3)(8-6) + (5-3)(10-6)] / 5$$

$$= [(-2)(-4) + (-1)(-2) + (0)(0) + (1)(2) + (2)(4)] / 5$$

$$= [8 + 2 + 0 + 2 + 8] / 5$$

$$= 20 / 5$$

$$= 4$$

- Covariance: Measures the relationship between two different random variables.
- Autocovariance Function: Measures the relationship between observations at different time points within the same random time series.

- **Covariance:**
  Example: Calculating the covariance between height X and weight Y.

- **Autocovariance** measures the linear relationship between observations at two different time points within the same time series (e.g., X_t and X_{t+k}).

- Example: Calculating the covariance between today's temperature X_t and tomorrow's temperature X_{t+1}.

Variance

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2$$

$$\operatorname{var}(X) = E\left[(X - \mu)^2\right]$$

Strict stationarity

- $\{x_t\}$ **is strictly (or strongly) stationary** if

  $(x_{t_1}, \ldots, x_{t_k})$ and $(x_{t_1+h}, \ldots, x_{t_k+h})$ have the same joint distribution

  for all $k, t_1, \ldots, t_k, h$

- **That is**

a **lag** of h days

$$\mathrm{P}\{x_{t_1} \leq c_1, \ldots, x_{t_k} \leq c_k\} = \mathrm{P}\{x_{t_1+h} \leq c_1, \ldots, x_{t_k+h} \leq c_k\}$$

Strict stationarity requires that the entire "probabilistic structure" or "statistical behavior" of a time series remains completely unchanged over time.

- **Strict stationarity means that all statistical properties of the process are invariant under a time shift. The joint distribution of any set of observations is exactly the same as that of a time-shifted set.**

Strict stationarity

$$\mathrm{P}\{x_{t_1} \leq c_1, \ldots, x_{t_k} \leq c_k\} = \mathrm{P}\{x_{t_1+h} \leq c_1, \ldots, x_{t_k+h} \leq c_k\}$$

$$\Downarrow$$

$$\mathrm{P}\{x_s \leq c\} = \mathrm{P}\{x_t \leq c\}$$

$$\Downarrow$$

$$\mu_t = \mu_s, \quad \forall s, t$$

the expectation (mean) is constant (the same for all time points)

- **Definition:**
  For a strictly stationary process $\{X_t\}$, its autocovariance function $\gamma(k)$ is defined as:
  $\gamma(k) = Cov(X_t, X_{t+k}) = E[(X_t - \mu)(X_{t+k} - \mu)]$
  where k is the lag order, and μ is the constant mean of the process.

What it represents:

- $\gamma(k)$ measures the degree of linear influence between an observation in the time series and another observation k periods later.

Strict stationarity

$$\mathrm{P}\{x_{t_1} \leq c_1, \ldots, x_{t_k} \leq c_k\} = \mathrm{P}\{x_{t_1+h} \leq c_1, \ldots, x_{t_k+h} \leq c_k\}$$

$$\Downarrow$$

$$\mathrm{P}\{x_s \leq c_1, x_t \leq c_2\} = \mathrm{P}\{x_{s+h} \leq c_1, x_{t+h} \leq c_2\}$$

$$\Downarrow$$

$$\gamma(s,t) = \gamma(s+h, t+h)$$

If a process is strictly stationary, then it must satisfy the condition $\gamma(s, t) = \gamma(s+h, t+h)$. This is because covariance is a property of the distribution - if the distribution remains unchanged, the covariance naturally remains unchanged as well.

Imagine an orchestra performing:

- Strict stationarity: Requires that every aspect of the orchestra's performance (melody, harmony, rhythm, volume, and even instrument timbre) remains exactly identical at all times. This is an extremely stringent requirement.

- Weak stationarity: Is a more relaxed and practical requirement. It only demands that:

1. The **average volume** remains constant (constant mean).
2. The **range of volume fluctuations** remains constant (constant variance). It won't suddenly shift from a whisper to a scream.
3. The **sense of rhythm** is stable. The time interval between adjacent drumbeats is fixed, regardless of which minute the song is in (stable covariance structure).

Weak stationarity

- $\{x_t\}$ **is weakly stationary** if

  μ_t is independent of t; and

  $\gamma(t+h,t)$ is independent of t for each h

$$\Downarrow$$

$$\mu_t = \mu$$

$$\gamma(t+h,t) = \gamma(h,0) = \gamma(h)$$

For a weakly stationary process, the covariance between any two time points depends solely on the interval (lag) *h* between them, and is independent of their absolute positions *t*. Consequently, we can use a function γ(*h*) that depends only on the lag *h* to completely describe the autocovariance structure of the entire process.

Weak stationarity

```
[X₁, X₂, X₃, X₄, ..., X_t, X_{t+1}, ...]
 |----h=1----| (Cov = γ(1) = 0.6)
          |----h=1----| (Cov = γ(1) = 0.6)
                |----h=1----| (Cov = γ(1) = 0.6)
```

Non-stationary process

```
[X₁, X₂, X₃, X₄, ..., X_t, X_{t+1}, ...]
 |----h=1----| (在t=1测量，Cov=0.9)
          |----h=1----| (在t=2测量，Cov=0.2)  <- 结果变了!
                |----h=1----| (在t=3测量，Cov=0.5)  <- 又变了!
```

Weak stationarity

This ruler is super simple—it can only measure the interval (*h*), and it automatically ignores the starting point (*t*).

How to measure:

1.You only need to tell the ruler an **interval *h*** (for example, *h* = 1 day).

2.The ruler will give you a unique value γ(*h*) (for example, γ(1) = 0.6).

3.The meaning of this value is: **In this process, the covariance between any two data points separated by one day is always 0.6.**

1. Cov(Jan 1, Jan 2) = 0.6
2. Cov(Jan 2, Jan 3) = 0.6
3. Cov(Jun 15, Jun 16) = 0.6
4. ...All pairs of points with a 1-day interval have a covariance of 0.6!

Strict stationarity

- The entire probability distribution is invariant with respect to time t.

Higher-order moments (skewness, kurtosis, and above)

Weak stationarity

First moment: Mean - describes the central location (average level) of a distribution.

Second central moment: Variance - describes the dispersion (magnitude of fluctuation) of a distribution.

Basic properties

- **Basic properties of $\gamma(\cdot)$**

  1. $\gamma(0) \geq 0$
  2. $|\gamma(h)| \leq \gamma(0), \ \forall h$
  3. $\gamma(h) = \gamma(-h), \ \forall h$

The variance of the time series is greater than zero.

Autocorrelation function

- **The autoccorelation function (ACF) of a stationary time series $\{x_t\}$ is defined as**

$$\rho(h) = \frac{\gamma(h)}{\gamma(0)} = \frac{\mathrm{Cov}(x_{t+h}, x_t)}{\mathrm{Cov}(x_t, x_t)} = \mathrm{Corr}(x_{t+h}, x_t)$$

$$\Downarrow$$

the variance of the time series.

$$-1 \leq \rho(h) \leq 1$$

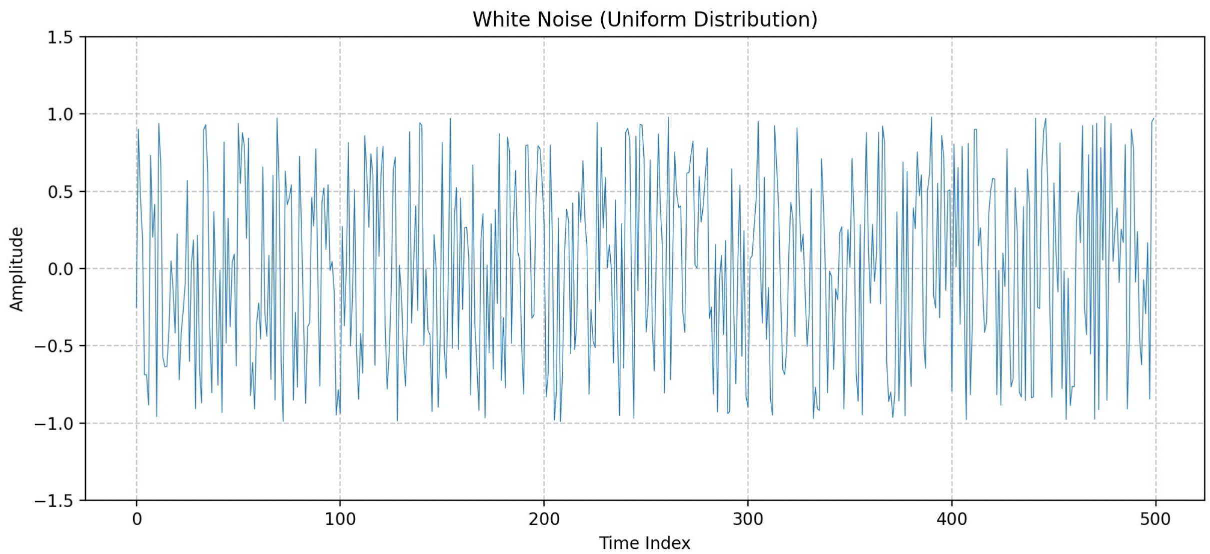
It answers the question: "To what extent is today's data correlated with data from yesterday, the day before yesterday, and so on?"

Stationarity

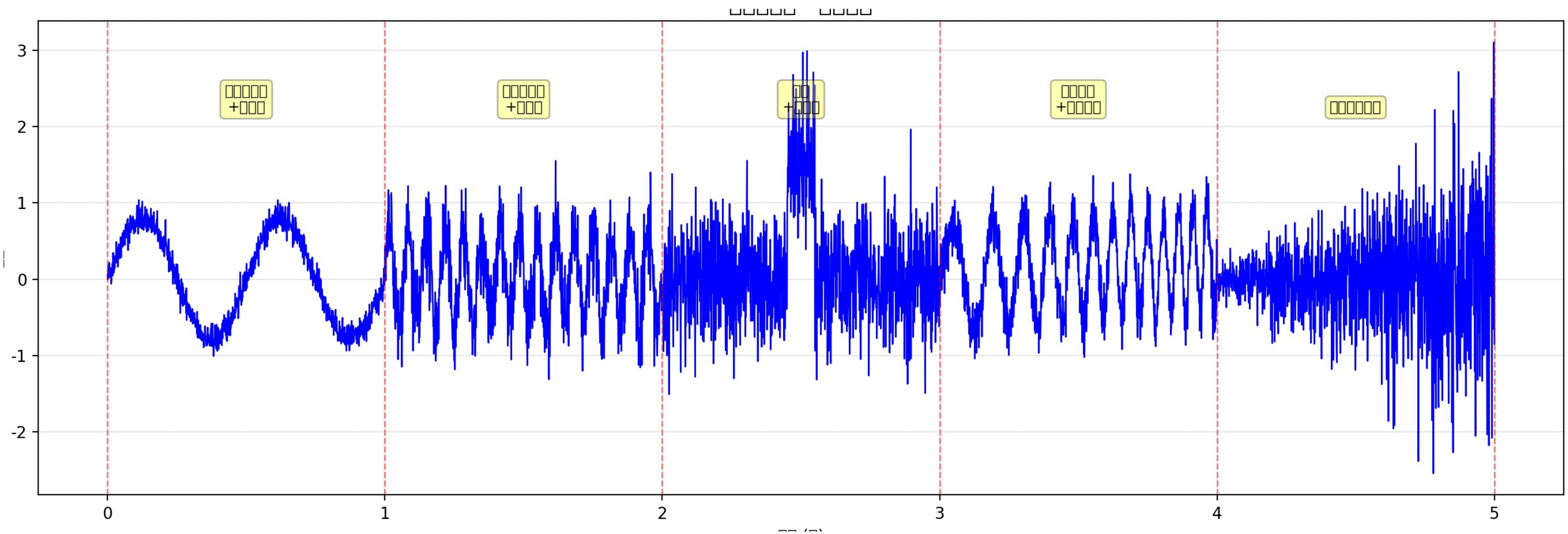
- **Examples:**

  white noise $\{w_t\}$

```python
import numpy as np
import matplotlib.pyplot as plt

# 设置随机种子以保证结果可重现
np.random.seed(42)

# 生成白噪声数据
# 生成长度为500的随机序列，值在[-1, 1]之间均匀分布
n_points = 500
white_noise = np.random.uniform(low=-1, high=1, size=n_points)
time = np.arange(n_points) # 时间轴

# 绘制图像
plt.figure(figsize=(12, 5))
plt.plot(time, white_noise, linewidth=0.5)
plt.title('White Noise (Uniform Distribution)')
plt.xlabel('Time Index')
plt.ylabel('Amplitude')
plt.grid(True, linestyle='--', alpha=0.7)
plt.ylim(-1.5, 1.5)
plt.show()
```



```python
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

# 设置中文字体
plt.rcParams['font.sans-serif'] = ['SimHei', 'DejaVu Sans']
plt.rcParams['axes.unicode_minus'] = False

# 设置随机种子以保证结果可重现
np.random.seed(42)

# 生成非平稳信号参数
sample_rate = 1000  # 采样率 1000 Hz
duration = 5        # 持续时间 5 秒
num_samples = sample_rate * duration  # 总样本数
time = np.linspace(0, duration, num_samples)

# 创建非平稳信号：由多个不同特性的片段组成
nonstationary_signal = np.zeros(num_samples)

# 分段创建信号（每段1秒）
segment_length = sample_rate  # 每段1000个样本

# 第1段：低频正弦波 + 低强度噪声
nonstationary_signal[0:segment_length] = (
    0.8 * np.sin(2 * np.pi * 2 * time[0:segment_length]) +  # 2Hz正弦波
    0.1 * np.random.normal(0, 1, segment_length)            # 低强度噪声
)

# 第2段：高频正弦波 + 中等强度噪声
nonstationary_signal[segment_length:2*segment_length] = (
    0.6 * np.sin(2 * np.pi * 15 * time[segment_length:2*segment_length]) +  # 15Hz正弦波
    0.3 * np.random.normal(0, 1, segment_length)                         # 中等噪声
)

# 第3段：脉冲信号 + 高强度噪声
nonstationary_signal[2*segment_length:3*segment_length] = (
    1.5 * (np.abs(time[2*segment_length:3*segment_length] - 2.5) < 0.05) +  # 窄脉冲
    0.5 * np.random.normal(0, 1, segment_length)                         # 高强度噪声
)

# 第4段：线性趋势 + 频率调制的正弦波
t_segment = time[3*segment_length:4*segment_length] - 3
nonstationary_signal[3*segment_length:4*segment_length] = (
    0.3 * t_segment +  # 线性趋势（均值随时间变化）
    0.7 * np.sin(2 * np.pi * (5 + 8 * t_segment) * t_segment) +  # 频率调制（5Hz增加到13Hz）
    0.2 * np.random.normal(0, 1, segment_length)
)

# 第5段：方差突变的随机噪声
nonstationary_signal[4*segment_length:5*segment_length] = np.random.normal(
    0,  # 均值保持为0
    0.1 + 0.9 * (time[4*segment_length:5*segment_length] - 4),  # 方差从0.1线性增加到1.0
    segment_length
)

# 创建时频分析（频谱图）
frequencies, times, spectrogram = signal.spectrogram(
    nonstationary_signal, sample_rate, nperseg=256, noverlap=128
)

# 创建图形
plt.figure(figsize=(14, 10))

# 1. 绘制时域波形图
plt.subplot(2, 1, 1)
plt.plot(time, nonstationary_signal, 'b-', linewidth=1.0)
plt.title('非平稳信号 - 时域波形', fontsize=14)
plt.xlabel('时间 (秒)')
plt.ylabel('幅度')
plt.grid(True, alpha=0.3)

# 添加分段标记
for i in range(6):
    plt.axvline(x=i, color='r', linestyle='--', alpha=0.5, linewidth=1)
plt.text(0.5, 2.2, '低频正弦波\n+弱噪声', ha='center', fontsize=9,
         bbox=dict(boxstyle="round,pad=0.3", facecolor="yellow", alpha=0.3))
plt.text(1.5, 2.2, '高频正弦波\n+中噪声', ha='center', fontsize=9,
         bbox=dict(boxstyle="round,pad=0.3", facecolor="yellow", alpha=0.3))
plt.text(2.5, 2.2, '脉冲\n+强噪声', ha='center', fontsize=9,
         bbox=dict(boxstyle="round,pad=0.3", facecolor="yellow", alpha=0.3))
plt.text(3.5, 2.2, '线性趋势\n+频率调制', ha='center', fontsize=9,
         bbox=dict(boxstyle="round,pad=0.3", facecolor="yellow", alpha=0.3))
plt.text(4.5, 2.2, '方差渐变噪声', ha='center', fontsize=9,
         bbox=dict(boxstyle="round,pad=0.3", facecolor="yellow", alpha=0.3))
```



=== 非平稳信号统计特性 ===

第1段 (0-1秒): 均值=0.002, 标准差=0.569, 方差=0.324

第2段 (1-2秒): 均值=0.021, 标准差=0.519, 方差=0.269

第3段 (2-3秒): 均值=0.153, 标准差=0.678, 方差=0.460

第4段 (3-4秒): 均值=0.163, 标准差=0.534, 方差=0.286

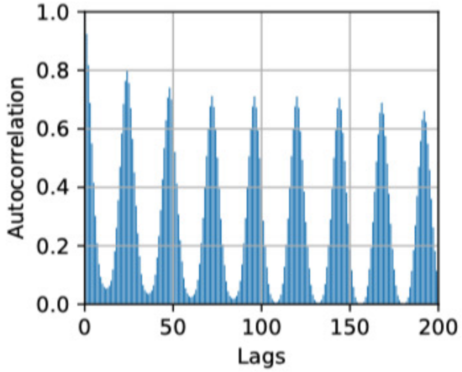
第5段 (4-5秒): 均值=-0.016, 标准差=0.610, 方差=0.372

Autocovariance Function

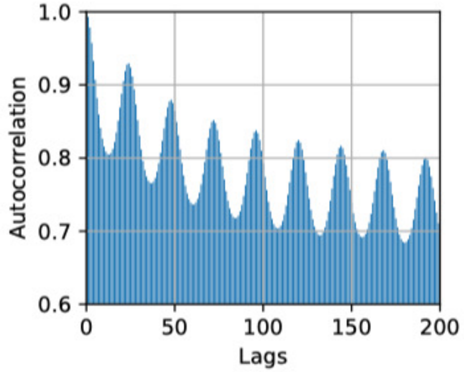
$$\gamma(k) = \mathrm{Cov}(X_t, X_{t+k}) = E[(X_t - \mu)(X_{t+k} - \mu)]$$

Autocorrelation Function - ACF

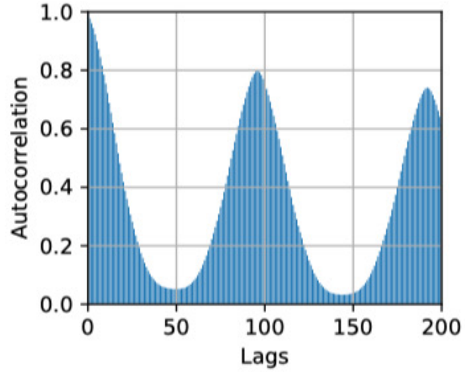
$$\rho(k) = \frac{\gamma(k)}{\gamma(0)} = \frac{\mathrm{Cov}(X_t, X_{t+k})}{\sqrt{\mathrm{Var}(X_t)\mathrm{Var}(X_{t+k})}} = \frac{\mathrm{Cov}(X_t, X_{t+k})}{\sigma^2}$$



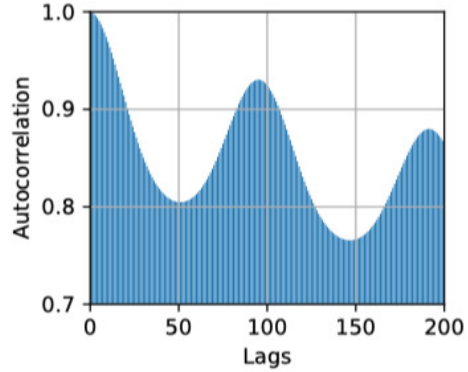
(a) ETTh1, $W = 24$



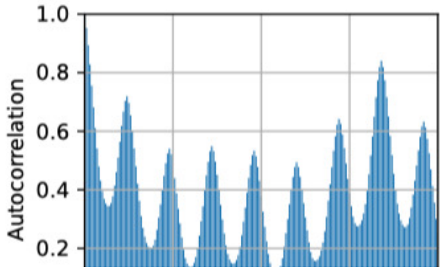
(b) ETTh2, $W = 24$

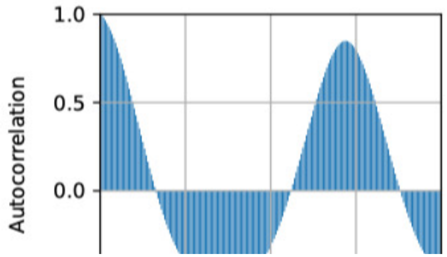


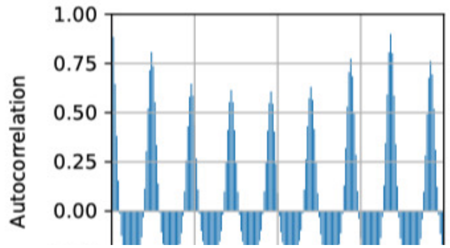
(c) ETTm1, $W = 96$

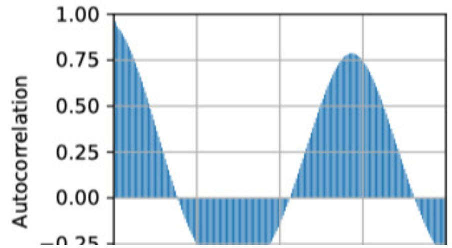


(d) ETTm2, $W = 96$









Stationarity

- **Examples:**

white noise $\{w_t\}$

$\mathrm{E}(w_t) = 0$ (independent of t) Zero Mean

$$\gamma(s,t) = \mathrm{cov}(w_s, w_t) = \begin{cases} \sigma_w^2 & s = t \\ 0 & s \neq t \end{cases}$$ (depends only on |s-t|)

Stationary!

Stationarity

- **Examples:**

$$\text{random walk } x_t = \sum_{j=1}^{t} w_j$$

Stationarity

- **Examples:**

random walk $x_t = \sum_{j=1}^{t} w_j$

$\gamma(s,t) = \min\{s,t\}\sigma_w^2$ (depends on both s and t)

NOT Stationary!

Stationarity

- **Examples:**

  MA(1) process (moving average)

  $x_t = w_t + \theta w_{t-1},$

  $\{w_t\} \sim \mathrm{wn}(0, \sigma^2)$

Stationarity

- **Examples:**

$\mathrm{E}(x_t) = 0$ (independent of t)

$$r(s,t) = \begin{cases} (1+\theta^2)\sigma^2, & s = t \\ \theta\sigma^2, & |s-t| = 1 \\ 0, & |s-t| \geq \end{cases}$$ (depends only on |s-t|)

Stationary!

Estimation of correlation

- **For observations $x_1, \ldots, x_n$ of a time series, the sample mean**

$$\bar{x} = \frac{1}{n}\sum_{t=1}^{n} x_t$$

- **The sample autocovariance function is**

$$\hat{\gamma}(h) = \frac{1}{n}\sum_{t=1}^{n-|h|}(x_{t+|h|} - \bar{x})(x_t - \bar{x}), \quad \text{for } -n < h < n$$

- **The sample autocorrelation function (sample ACF) is**

$$\hat{\rho}(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}, \quad \text{for } -n < h < n$$

Estimation of correlation

- **For observations $x_1, \ldots, x_n$ of a time series, the sample mean**

$$\bar{x} = \frac{1}{n}\sum_{t=1}^{n} x_t$$

$$\downarrow$$

$$\mathrm{E}(\bar{x}) = \mu, \quad \mathrm{Var}(\bar{x}) = \frac{1}{n}\sum_{h=-n}^{n}\left(1 - \frac{|h|}{n}\right)\gamma(h)$$

Estimation of correlation

- **For a white noise process *w_t*, if** $E(w_t^4) < \infty$,

$$\begin{pmatrix} \hat{\rho}(1) \\ \hat{\rho}(2) \\ \vdots \\ \hat{\rho}(K) \end{pmatrix} \sim AN\left(0, \frac{1}{n} I\right)$$

Asymptotically Normal

Estimation of correlation

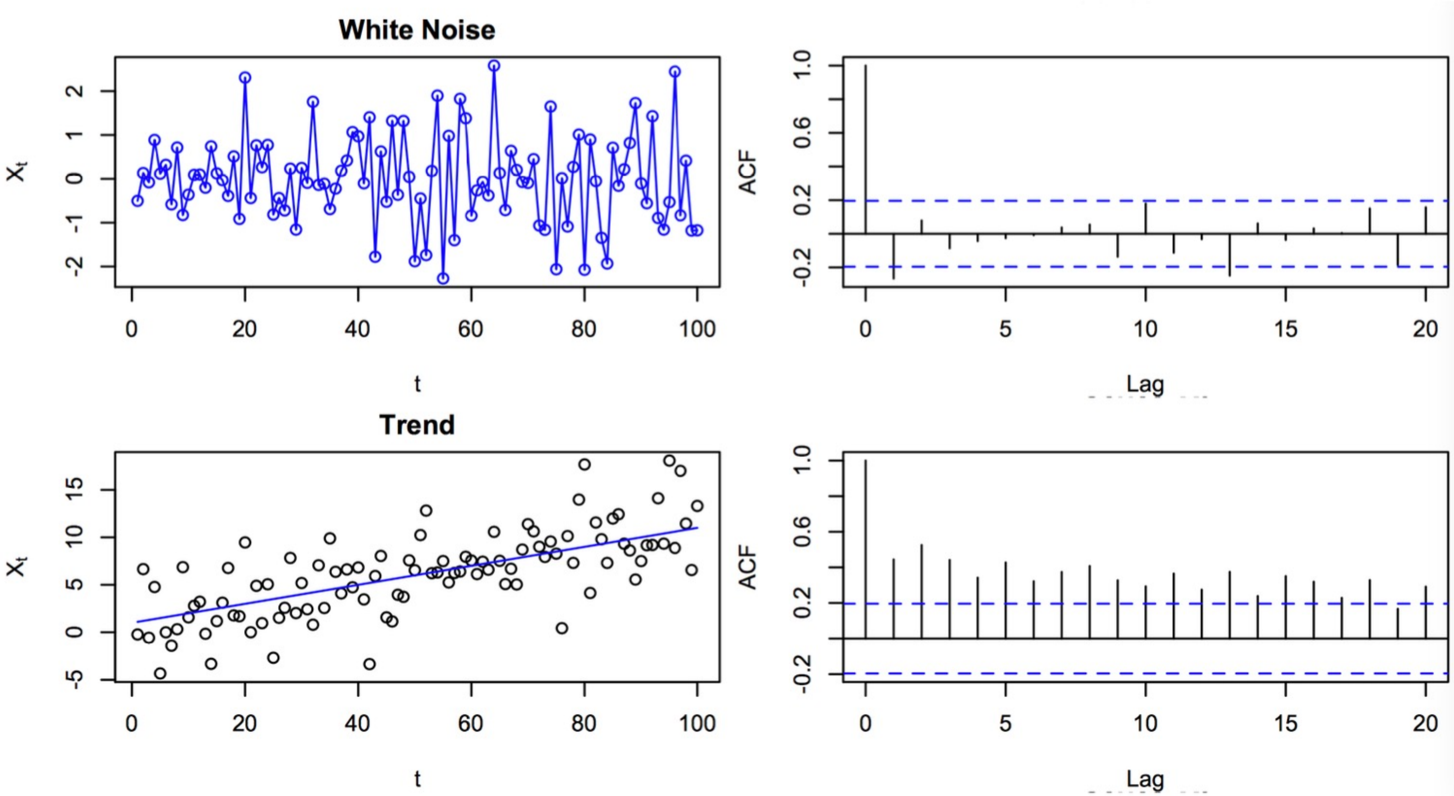
- **If $\{w_t\}$ is white noise, we expect no more than 5% of the peaks of the sample ACF to satisfy**

$$|\hat{\rho}(h)| > \frac{1.96}{\sqrt{n}}$$

- **Useful when we want to transform a time series to white noise.**

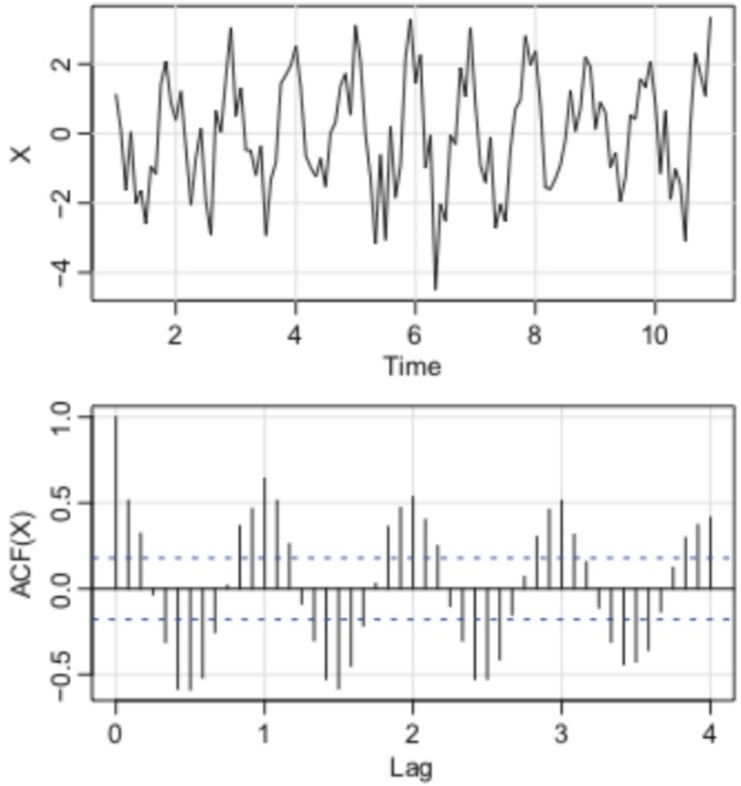
Estimation of correlation

- **The sample ACF can help us recognize many non-white (even non-stationary) time series.**



Estimation of correlation

- **The sample ACF can help us recognize many non-white (even non-stationary) time series.**



Backshift and forward-shift operator

- **Backshift operator:**

$$Bx_t = x_{t-1}$$

$$B^k x_t = x_{t-k}$$

- **Forward-shift operator**

$$x_t = B^{-1} x_{t-1}$$

Difference operator

- **First difference operator:**

$$\nabla x_t = x_t - x_{t-1}$$

$$\Downarrow$$

$$\nabla x_t = (1 - B)x_t$$

- **Differences with order d:**

$$\nabla^d = (1 - B)^d$$

Difference operator

- **The first difference eliminates a linear trend:**

  $$x_t = \beta_0 + \beta_1 t + y_t$$

- **The second order difference eliminates a quadratic trend:**

  $$x_t = \beta_0 + \beta_1 t + \beta_2 t^2 + y_t$$

Difference operator

- **The first difference eliminates a linear trend:**

$$x_t = \beta_0 + \beta_1 t + y_t$$

$$\begin{aligned}\nabla x_t &= x_t - x_{t-1} \\ &= \beta_0 + \beta_1 t + y_t - (\beta_0 + \beta_1(t-1) + y_{t-1}) \\ &= \beta_1 + y_t - y_{t-1}\end{aligned}$$

- **The second order difference eliminates a quadratic trend:**

$$x_t = \beta_0 + \beta_1 t + \beta_2 t^2 + y_t$$

$$\begin{aligned}\nabla x_t &= x_t - x_{t-1} \\ &= \beta_1 - \beta_2 + 2\beta_2 t + y_t - y_{t-1}\end{aligned}$$

$$\begin{aligned}\nabla^2 x_t &= \nabla(\nabla x_t) \\ &= 2\beta_2 + y_t - 2y_{t-1} + y_{t-2}\end{aligned}$$