Time Series Econometrics (ECO 402)

Lecture Notes

Feda Mohammadi

Professor: Dr. Nabila Rahman

Berea College

Spring 2026

These notes are for personal academic study. Notation and results follow standard time series econometrics conventions.

Contents

Chapter 1: Fundamental Concepts of Time-Series Econometrics

1.1 Time Series and White Noise

Many of the ideas we learned in cross-section econometrics still matter when data are collected over time. But time-series data create extra challenges because observations close in time often move together.

A key difference is this:

- In cross-section data, it is often reasonable to treat observations (across people, firms, counties, etc.) as independent.
- In time series, observations are **serially correlated**: what happens today is related to what happened yesterday.

1.1.1 Time-series processes

A **time series** is a sequence of observations on a variable taken at discrete time intervals.

We index time periods as

$$t = 1, 2, \ldots, T,$$

and we write the observed sample as

$$(y_1, y_2, \ldots, y_T).$$

Even though we only observe a finite sample, it is useful to think of these observations as coming from a larger object: a **time-series stochastic process** that extends into the past and the future.

Formally, imagine a full process:

$$\{\ldots, y_{-2}, y_{-1}, y_0, y_1, y_2, \ldots, y_T, y_{T+1}, y_{T+2}, \ldots\}.$$

To connect this picture to what we actually observe, we can mentally split the process into three parts:

$$\underbrace{\{\ldots, y_{-2}, y_{-1}, y_0\}}_{\text{pre-sample}} \quad \underbrace{\{y_1, y_2, \ldots, y_T\}}_{\text{sample}} \quad \underbrace{\{y_{T+1}, y_{T+2}, \ldots\}}_{\text{post-sample}}.$$

Why treat each y_t as random?

In time series econometrics, we treat each y_t as a **random variable** with a probability distribution.

That does *not* mean the numbers you see are random in a casual sense. It means we model the data as being generated by an underlying probabilistic mechanism, so we can:

- describe patterns (like persistence),
- quantify uncertainty,
- forecast future values,
- test hypotheses about the data-generating process.

Common parameters across time

A typical starting assumption is that the distribution of y_t has some parameters that are common across time. For example, we might assume:

- the variance is the same at every date:

$$\text{Var}(y_t) = \sigma_y^2 \quad \text{for all } t,$$

- and the covariance between adjacent observations is the same at every date:

$$\text{Cov}(y_t, y_{t-1}) = \text{constant for all } t.$$

If the distribution of y_t is the same for all t, we call the series **stationary**. We will define stationarity carefully later, but for now the intuition is: *A stationary series behaves the same way over time: its typical level, spread, and dependence pattern do not drift as t changes.*

Example: Sample vs process intuition

Suppose you have monthly inflation data for 10 years, so $T = 120$. You observe $(y_1, \ldots, y_{120})$, but you think of these values as one piece of a longer inflation-generating process that existed before your sample began and will continue after it ends. That viewpoint is what makes forecasting and inference possible.

1.1.2 White noise

The simplest time-series process is **white noise**. It is the time-series version of the classical "error term" idea from regression: pure randomness with no predictable structure.

Intuition

If a variable is white noise, then each new observation is a complete surprise.

- Knowing the past gives you **no clue** about whether the next value will be positive or negative.
- Values do not systematically persist or mean-revert in a predictable way.
- There is **no serial correlation**.

Formal definition:
A process $\{\varepsilon_t\}$ is a **white-noise process** if it satisfies all three properties:

$$\mathbb{E}(\varepsilon_t) = 0 \quad \text{for all } t,$$

$$\text{Var}(\varepsilon_t) = \sigma^2 \quad \text{for all } t,$$

$$\text{Cov}(\varepsilon_t, \varepsilon_{t-s}) = 0 \quad \text{for all } s \neq 0. \tag{1.1}$$

Let's unpack each one carefully.

Step 1: Mean zero

$$\mathbb{E}(\varepsilon_t) = 0 \ \ \forall t.$$

This means the series has no systematic upward or downward bias. On average, it is centered at zero.

Step 2: Constant variance

$$\mathrm{Var}(\varepsilon_t) = \sigma^2 \ \ \forall t.$$

This means the amount of randomness is stable over time. The process is not "calm" in some periods and "wild" in others.

Step 3: Zero covariance across time

$$\mathrm{Cov}(\varepsilon_t, \varepsilon_{t-s}) = 0 \quad \forall s \neq 0.$$

This is the key time-series feature.

To see what it means, recall the definition of covariance:

$$\mathrm{Cov}(\varepsilon_t, \varepsilon_{t-s}) = \mathbb{E}\big[(\varepsilon_t - \mathbb{E}[\varepsilon_t])(\varepsilon_{t-s} - \mathbb{E}[\varepsilon_{t-s}])\big].$$

Because $\mathbb{E}[\varepsilon_t] = 0$ for all t, this simplifies step-by-step to:

$$\mathrm{Cov}(\varepsilon_t, \varepsilon_{t-s}) = \mathbb{E}\big[(\varepsilon_t - 0)(\varepsilon_{t-s} - 0)\big] = \mathbb{E}(\varepsilon_t \varepsilon_{t-s}).$$

So the white-noise condition for $s \neq 0$ is equivalently:

$$\mathbb{E}(\varepsilon_t \varepsilon_{t-s}) = 0 \quad \text{for all } s \neq 0.$$

That means there is no linear relationship between ε_t and any past value $\varepsilon_{t-1}, \varepsilon_{t-2}, \ldots$.

Autocovariances

The covariances

$$\mathrm{Cov}(\varepsilon_t, \varepsilon_{t-s})$$

are called the **autocovariances** of the series. The number s is the **lag**.

For white noise:

$$\gamma_\varepsilon(0) = \mathrm{Var}(\varepsilon_t) = \sigma^2, \qquad \gamma_\varepsilon(s) = 0 \text{ for } s \neq 0.$$

Normality is not required

Some authors define white noise as i.i.d. normal. But in many econometrics settings, we do *not* include normality as part of the definition.

So in general:

$$\text{white noise} \neq \text{necessarily normal.}$$

Why most economic time series are not white noise

Economic time series often show persistence. Inflation, GDP, unemployment, interest rates, and exchange rates usually have values that "carry over" from one period to the next.

That implies:

$$\text{Cov}(y_t, y_{t-1}) \neq 0$$

for many economic series, so they are **serially correlated** and therefore **not** white noise.

Example: A white-noise model

A simple example is

$$\varepsilon_t \sim \text{i.i.d. } (0, \sigma^2).$$

Then:

$$\mathbb{E}[\varepsilon_t] = 0, \quad \text{Var}(\varepsilon_t) = \sigma^2,$$

and because the draws are independent,

$$\text{Cov}(\varepsilon_t, \varepsilon_{t-s}) = 0 \quad (s \neq 0).$$

So i.i.d. mean-zero draws with constant variance satisfy the white-noise definition.

1.1.3 White noise as a building block

Even though most economic time series are not white noise, white noise is still extremely useful because it can be used to build more realistic models.

Core idea: predictable part + surprise part

We model many time series by splitting y_t into two additive components:

- a part that can be predicted using past information, and
- a part that cannot be predicted (the new shock).

A very general way to write this is:

$$y_t = g(y_{t-1}, y_{t-2}, \dots, \varepsilon_{t-1}, \varepsilon_{t-2}, \dots) + \varepsilon_t. \tag{1.2}$$

Here:

- $g(\cdot)$ represents the **predictable component** of y_t based on the past.
- ε_t is the **innovation** (or **shock**): the part of y_t that is new at time t and not predictable from the past.

What information do we condition on?

Let $\mathcal{I}_{t-1}$ be the information available through time $t-1$. This typically means: all past observations and anything else known by $t-1$.

Because $g(\cdot)$ is built only from past values, it is measurable using $\mathcal{I}_{t-1}$. So the best possible forecast of y_t given $\mathcal{I}_{t-1}$ is:

$$\mathbb{E}(y_t \mid \mathcal{I}_{t-1}) = g(y_{t-1}, y_{t-2}, \dots, \varepsilon_{t-1}, \varepsilon_{t-2}, \dots).$$

Why is that the best forecast? (step-by-step)

Start from (1.2):

$$y_t = g(\cdot) + \varepsilon_t.$$

Take conditional expectation given $\mathcal{I}_{t-1}$ on both sides:

$$\mathbb{E}(y_t \mid \mathcal{I}_{t-1}) = \mathbb{E}(g(\cdot) + \varepsilon_t \mid \mathcal{I}_{t-1}).$$

Use linearity of conditional expectation:

$$\mathbb{E}(y_t \mid \mathcal{I}_{t-1}) = \mathbb{E}(g(\cdot) \mid \mathcal{I}_{t-1}) + \mathbb{E}(\varepsilon_t \mid \mathcal{I}_{t-1}).$$

Now:

- Since $g(\cdot)$ is entirely based on past information, it is already known at $t-1$, so

$$\mathbb{E}(g(\cdot) \mid \mathcal{I}_{t-1}) = g(\cdot).$$

- Since ε_t is white noise, it has mean zero and is uncorrelated with the past. The modeling meaning is:

$$\mathbb{E}(\varepsilon_t \mid \mathcal{I}_{t-1}) = 0.$$

Therefore:

$$\mathbb{E}(y_t \mid \mathcal{I}_{t-1}) = g(\cdot).$$

Forecast error

The one-step-ahead forecast error is:

$$y_t - \mathbb{E}(y_t \mid \mathcal{I}_{t-1}).$$

Substitute the expressions from above:

$$y_t - \mathbb{E}(y_t \mid \mathcal{I}_{t-1}) = \big(g(\cdot) + \varepsilon_t\big) - g(\cdot) = \varepsilon_t.$$

So the **unavoidable** forecast error is exactly the innovation:

$$y_t - \mathbb{E}(y_t \mid \mathcal{I}_{t-1}) = \varepsilon_t.$$

This is a big deal:

> The only reason our forecast is wrong is because of genuinely new information ε_t that no one could predict using the past.

Why equation (1.2) is powerful

Many stationary processes and many useful nonstationary processes can be described using a structure like (1.2).

To characterize a specific time series, we usually need two things:

1. The form of the predictable function $g(\cdot)$.
2. The variance of the innovations, $\text{Var}(\varepsilon_t) = \sigma^2$.

A very common choice is a **linear stochastic process**, where:

- $g(\cdot)$ is linear, and
- only finitely many lags appear.

That leads directly to AR, MA, and ARMA models later.

1.2 Linear Stochastic Processes and the Lag Operator

We already introduced the idea that a time series can be decomposed into two parts:

$$y_t = g(y_{t-1}, y_{t-2}, \dots, \varepsilon_{t-1}, \varepsilon_{t-2}, \dots) + \varepsilon_t$$

where the function $g(\cdot)$ represents the predictable part of the process and the term ε_t is the innovation or shock that cannot be predicted from past information.

In practice, it is very useful to work with a specific class of stochastic processes in which this predictable component is *linear*. These are called **linear stochastic processes**. Many of the models used in econometrics and macroeconomic analysis belong to this class.

The most important linear processes are called **autoregressive-moving-average processes**, or **ARMA processes**. These models combine two ideas:

- dependence on past values of the variable itself
- dependence on past shocks or innovations

These two sources of persistence help us describe the dynamic behavior observed in many economic time series.

1.2.1 ARMA processes

A linear stochastic process can be written in the following general form

$$y_t = \alpha + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} \tag{1.3}$$

This equation describes the current value of the series y_t as a function of three components.

1. A constant term: α

This constant allows the process to have a non-zero mean.

2. Autoregressive terms:

$$\phi_1 y_{t-1}, \quad \phi_2 y_{t-2}, \dots, \phi_p y_{t-p}$$

These terms capture the influence of past values of the variable itself. The name *autoregressive* comes from the idea that the variable is being "regressed" on its own past values. The number p tells us how many past values affect the present.

3. Moving-average terms

$$\varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

These terms represent the effect of current and past shocks. The parameter q tells us how many past innovations influence the current observation. Because the model contains both autoregressive and moving-average components, we call it an

$$\textbf{ARMA}(p, q)$$

process.

To see the connection with the general representation introduced earlier, we can write the predictable component as

$$g(y_{t-1}, y_{t-2}, \ldots, \varepsilon_{t-1}, \varepsilon_{t-2}, \ldots) = \alpha + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} \tag{1.4}$$

Thus the ARMA process simply specifies a particular linear form for the function $g(\cdot)$.

Two special cases are important.

Autoregressive process: If $q = 0$, there are no moving-average terms and the model becomes

$$y_t = \alpha + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

This is called an

$$AR(p)$$

process.

Moving-average process: If $p = 0$, there are no autoregressive terms and the model becomes

$$y_t = \alpha + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

This is called an

$$MA(q)$$

process.

Historical note: The systematic analysis of ARMA processes was developed by Box and Jenkins (1976), whose work became the foundation of modern time-series modeling.

1.2.2 Lag operator

When working with time-series models it is often convenient to use a mathematical operator called the **lag operator**.

The lag operator is denoted by (L), and it shifts a variable one period into the past.

Formally, applying the lag operator to a time-series variable produces

$$L(y_t) = y_{t-1}$$

This simply means that the operator replaces the variable with its previous value.

Because the lag operator can be applied repeatedly, we can generate longer lags.

For example,

$$L^2(y_t) = L(L(y_t)) = L(y_{t-1}) = y_{t-2}$$

More generally,

$$L^k(y_t) = y_{t-k}$$

This notation allows us to write time-series equations in a much more compact and readable way.

Using the lag operator, equation (1.3) can be rewritten as

$$y_t - \phi_1 L(y_t) - \phi_2 L^2(y_t) - \cdots - \phi_p L^p(y_t) = \alpha + \varepsilon_t + \theta_1 L(\varepsilon_t) + \theta_2 L^2(\varepsilon_t) + \cdots + \theta_q L^q(\varepsilon_t) \quad (1.5)$$

Now we simplify the left-hand side. Substituting the definition of the lag operator gives

$$y_t - \phi_1 y_{t-1} - \phi_2 y_{t-2} - \cdots - \phi_p y_{t-p}$$

Factoring y_t from the expression gives

$$(1 - \phi_1 L - \phi_2 L^2 - \cdots - \phi_p L^p) y_t \quad (1.6)$$

The expression inside the parentheses is called a **polynomial in the lag operator**.

We define

$$\phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \cdots - \phi_p L^p$$

Similarly, the moving-average part can be written as

$$(1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_q L^q)\varepsilon_t$$

and we define

$$\theta(L) = 1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_q L^q$$

Using these definitions we can express the ARMA model compactly as

$$\phi(L)y_t = \alpha + \theta(L)\varepsilon_t \tag{1.7}$$

The left side represents the autoregressive component, and the right side represents the moving-average component.

1.2.3 Infinite moving-average representation of an ARMA process

An important property of ARMA models is that they can often be written as an infinite moving-average process.

To understand the intuition, consider the simple ARMA(1,1) process

$$y_t = \phi_1 y_{t-1} + \varepsilon_t + \theta_1 \varepsilon_{t-1} \tag{1.8}$$

To see how past shocks affect the current value, we substitute the lagged equation into itself.

First, lag equation (1.8) by one period:

$$y_{t-1} = \phi_1 y_{t-2} + \varepsilon_{t-1} + \theta_1 \varepsilon_{t-2}$$

Substituting this into equation (1.8) gives

$$y_t = \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1} + \theta_1 \varepsilon_{t-2}) + \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

Expanding the expression yields

$$y_t = \phi_1^2 y_{t-2} + \varepsilon_t + (\phi_1 + \theta_1)\varepsilon_{t-1} + \phi_1 \theta_1 \varepsilon_{t-2}$$

If we continue substituting recursively, the lagged y term moves further into the past and the

expression accumulates additional lagged shocks. If the process is **stationary**, which requires

$$|\phi_1| < 1$$

then the coefficients on the lagged y terms shrink toward zero.

After infinitely many substitutions the lagged y terms vanish and we obtain

$$y_t = \varepsilon_t + \sum_{s=1}^{\infty} \phi_1^{s-1}(\phi_1 + \theta_1)\varepsilon_{t-s} \tag{1.9}$$

This representation shows that the current value of the series can be written entirely as a weighted sum of current and past shocks.

Thus an ARMA model can be interpreted as an **infinite moving-average process**.

Using lag-operator notation we can also derive this representation from

$$\phi(L)y_t = \alpha + \theta(L)\varepsilon_t$$

We are dividing both sides by the autoregressive polynomial which gives

$$y_t = \frac{\alpha}{\phi(L)} + \frac{\theta(L)}{\phi(L)}\varepsilon_t \tag{1.10}$$

The second term involves a ratio of two lag polynomials, which can be expanded as an infinite series when the stationarity condition holds.

Using the geometric-series result

$$\sum_{i=0}^{\infty} a^i = \frac{1}{1-a}, \qquad |a| < 1$$

we obtain

$$\sum_{i=0}^{\infty} (\phi_1 L)^i = \frac{1}{1-\phi_1 L} \tag{1.11}$$

Substituting this expansion into equation (1.10) produces the infinite moving-average representation shown in equation (1.9). This representation is extremely useful because it shows how shocks propagate through time in a dynamic system.

1.3 Stationary and Nonstationary Time-Series Processes

One of the most important questions in time-series econometrics is whether a process is **stationary** or **nonstationary**.

This matters because many of the methods used in econometrics work well only when the stochastic process has stable properties over time. If the behavior of the series keeps changing as time passes, then standard tools can become misleading or even completely invalid.

Intuitively, a process is **stationary** if it behaves the same way at every point in time. In other words, its probability structure does not drift over time.
A useful intuition is the following:

- In a stationary process, shocks affect the series, but their effects eventually die out.
- In a nonstationary process, shocks may persist for a very long time or even permanently.

A common example of nonstationarity is a series with a trend. If a variable grows or declines systematically over time, then its mean is changing, so its distribution is not the same at all dates.

This distinction is crucial before we study time-series regressions. Regressions involving nonstationary variables require special care, so we first need a clear definition of stationarity.

1.3.1 Stationarity and ergodicity

There are two related ideas here:

- **stationarity**, which says that the probabilistic behavior of the process does not change over time,
- **ergodicity**, which says that a long realization of the process contains enough information to learn about its population properties.

Strict stationarity

A time-series process y_t is **strictly stationary** if the joint probability distribution of any collection of observations depends only on the distances between them in time, and not on the calendar date at which they are observed.

For the pair (y_t, y_{t-s}), this means the joint distribution depends only on s, not on t.

So if a process is strictly stationary, then the joint distribution of (y_{1920}, y_{1925}) must be the same as the joint distribution of (y_{1980}, y_{1985}), because in both cases the observations are five periods apart.

Strict stationarity is the strongest notion of stationarity because it requires the *entire distribution* to be invariant over time.

Weak stationarity or covariance stationarity

In most econometrics applications, we use a weaker concept called **weak stationarity** or **covariance stationarity**.

A process y_t is covariance stationary if:

$$E(y_t) = \mu \qquad \forall t,$$

$$\text{var}(y_t) = \sigma_y^2 \qquad \forall t,$$

$$\text{cov}(y_t, y_{t-s}) = \sigma_s \qquad \forall t, s.$$

These three conditions mean:

- the mean does not depend on time,
- the variance does not depend on time,
- the covariance between two observations depends only on the lag s, not on the particular time t.

So the first two moments and the lag dependence structure stay constant over time.

If a process is covariance stationary, then:

- its average level stays the same,
- its spread stays the same,
- the relationship between current and lagged values depends only on how far apart they are, not on where you are in the sample.

Autocovariances and autocorrelations

The quantity

$$\sigma_s = \text{cov}(y_t, y_{t-s})$$

is called the ***s*-order autocovariance**. It measures how strongly the series is related to its own past s periods ago.

Because covariance depends on the units of measurement, we often work instead with **autocorrelation**, which is unit free.

The s-order autocorrelation is

$$\rho_s = \text{corr}(y_t, y_{t-s}) = \frac{\sigma_s}{\sigma_y^2}.$$

Since

$$\text{corr}(y_t, y_t) = 1,$$

we always have

$$\rho_0 = 1.$$

Persistence

Autocovariances and autocorrelations measure **persistence**.

If the autocorrelations are large and positive, then shocks tend to die out slowly, so the series is persistent. If the autocorrelations are small, then shocks disappear more quickly.

Ergodicity

A concept closely related to stationarity is **ergodicity**.

Roughly speaking, ergodicity means that one long sample path contains enough information to recover the population moments of the process. For our purposes, stationary normally distributed time-series processes are ergodic if the sum of the absolute autocorrelations is finite:

$$\sum_{s=0}^{\infty} |\rho_s| < \infty.$$

This condition is stronger than simply requiring

$$\lim_{s \to \infty} \rho_s = 0.$$

Why?

Because saying $\rho_s \to 0$ only tells us that autocorrelation eventually disappears. But ergodicity

requires the autocorrelations to go to zero *sufficiently fast*.

In these notes, we will usually assume that stationary processes are also ergodic.

1.3.2 Kinds of non-stationarity

There are several ways a time-series process can fail to be stationary.

1. Breakpoint nonstationarity

One possibility is that the parameters of the data-generating process change at some particular date. This is called **breakpoint nonstationarity**.

For example, a macroeconomic series might behave one way before a major oil shock or policy reform, and another way afterward. In that case, the process does not have the same distribution over the whole sample.

2. Trend nonstationarity

A second common case is **trend nonstationarity**. Suppose a series can be written as

$$y_t = \alpha + \beta t + x_t,$$

where β is a deterministic trend and x_t is stationary with

$$E(x_t) = 0.$$

Then the mean of y_t is

$$E(y_t) = E(\alpha + \beta t + x_t).$$

Using linearity of expectation,

$$E(y_t) = \alpha + \beta t + E(x_t).$$

Since $E(x_t) = 0$,

$$E(y_t) = \alpha + \beta t.$$

So the mean changes over time. Therefore the process is not stationary.

This type of nonstationarity comes from a deterministic trend.

3. Stochastic trend or integrated nonstationarity

A third important kind of nonstationarity is **stochastic trend** nonstationarity.

These processes do not just drift because of a deterministic time trend. Instead, they are highly persistent and accumulate shocks over time.

A standard feature of such processes is that they can often be made stationary by differencing.

That is, if y_t is nonstationary, the differenced series

$$\Delta y_t = y_t - y_{t-1}$$

may be stationary. Processes with this property are called **integrated processes**.

1.3.3 Stationarity in MA(q) models

We now study stationarity for specific classes of models. Start with the moving-average model of order q:

$$y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q},$$

where ε_t is white noise with variance σ^2.

For simplicity, we assume the mean is zero in our stationary examples. Adding a constant changes only the mean, not the variance or autocovariances, so it does not affect stationarity.

Mean of an MA(q) process

Take expectations:

$$E(y_t) = E(\varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}).$$

By linearity of expectation,

$$E(y_t) = E(\varepsilon_t) + \theta_1 E(\varepsilon_{t-1}) + \cdots + \theta_q E(\varepsilon_{t-q}).$$

Since white noise has mean zero at every date,

$$E(\varepsilon_t) = E(\varepsilon_{t-1}) = \cdots = E(\varepsilon_{t-q}) = 0.$$

Therefore,

$$E(y_t) = 0 \qquad \text{So the mean does not depend on } t.$$

Variance of an MA(q) process

Now we compute the variance of

$$y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}.$$

We write

$$\operatorname{var}(y_t) = \operatorname{var}\left(\varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}\right).$$

We expand variance term by term. Because white-noise innovations at different dates are uncorrelated, all cross-covariance terms are zero. So only the variance terms remain:

$$\operatorname{var}(y_t) = \operatorname{var}(\varepsilon_t) + \theta_1^2 \operatorname{var}(\varepsilon_{t-1}) + \cdots + \theta_q^2 \operatorname{var}(\varepsilon_{t-q}).$$

Since each white-noise term has variance σ^2,

$$\operatorname{var}(y_t) = \sigma^2 + \theta_1^2 \sigma^2 + \cdots + \theta_q^2 \sigma^2.$$

Factor out σ^2:

$$\operatorname{var}(y_t) = (1 + \theta_1^2 + \cdots + \theta_q^2)\sigma^2.$$

So the variance is finite and constant over time.

Autocovariances of an MA(q) process

Now consider the s-order autocovariance:

$$\operatorname{cov}(y_t, y_{t-s}).$$

Substitute the MA(q) form for both terms:

$$\operatorname{cov}(y_t, y_{t-s}) = E\left[(\varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q})(\varepsilon_{t-s} + \theta_1 \varepsilon_{t-s-1} + \cdots + \theta_q \varepsilon_{t-s-q})\right],$$

because the mean is zero.

The only nonzero terms are those in which the subscripts match, because white-noise terms at different dates are uncorrelated.

For $s = 1$, the overlapping terms are:

$$\varepsilon_{t-1}, \varepsilon_{t-2}, \ldots, \varepsilon_{t-q}.$$

So

$$\operatorname{cov}(y_t, y_{t-1}) = (\theta_1 + \theta_1\theta_2 + \theta_2\theta_3 + \cdots + \theta_{q-1}\theta_q)\sigma^2$$

is not the cleanest book-style way to derive it. Let us write it carefully from matching coefficients.

The coefficient on ε_{t-j} in y_t is:

$$\begin{cases} 1 & \text{if } j = 0, \\ \theta_j & \text{if } j = 1, \ldots, q. \end{cases}$$

The coefficient on the same shock ε_{t-j} in y_{t-s} exists only if that shock appears in both expressions.

So for $s = 1$,

$$\operatorname{cov}(y_t, y_{t-1}) = (\theta_1 + \theta_2\theta_1 + \theta_3\theta_2 + \cdots + \theta_q\theta_{q-1})\sigma^2.$$

More generally, for $s \leq q$,

$$\sigma_s = \operatorname{cov}(y_t, y_{t-s}) = (\theta_s + \theta_{s+1}\theta_1 + \theta_{s+2}\theta_2 + \cdots + \theta_q\theta_{q-s})\sigma^2.$$

And for $s > q$, there is no overlap between the sets of shocks appearing in y_t and y_{t-s}, so

$$\sigma_s = 0 \qquad \text{for } s > q.$$

Conclusion for MA(q)

The mean, variance, and autocovariances of an MA(q) process do not depend on t. Therefore every finite MA(q) process is stationary, regardless of the values of the θ parameters.
Also, because $\sigma_s = 0$ for all $s > q$, the autocorrelations die out completely after lag q. Hence all finite MA(q) processes are also ergodic.

1.3.4 Stationarity in AR(p) processes

Unlike MA processes, autoregressive processes are not automatically stationary. Consider the zero-mean AR(p) process:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t.$$

We begin with the AR(1) case.

AR(1) process:

$$y_t = \phi_1 y_{t-1} + \varepsilon_t. \tag{1.13}$$

From the infinite moving-average representation, we can write

$$y_t = \sum_{i=0}^{\infty} \phi_1^i \varepsilon_{t-i}, \tag{1.14}$$

provided the series converges.

Mean of an AR(1) process:

Take expectations of both sides:

$$E(y_t) = E\left(\sum_{i=0}^{\infty} \phi_1^i \varepsilon_{t-i}\right).$$

Using linearity of expectation,

$$E(y_t) = \sum_{i=0}^{\infty} \phi_1^i E(\varepsilon_{t-i}).$$

Since each white-noise innovation has mean zero,

$$E(\varepsilon_{t-i}) = 0 \qquad \forall i.$$

Therefore,

$$E(y_t) = 0.$$

Variance of an AR(1) process

Now we compute the variance:

$$\sigma_y^2 = \operatorname{var}(y_t) = \operatorname{var}\left(\sum_{i=0}^{\infty} \phi_1^i \varepsilon_{t-i}\right).$$

Because the white-noise innovations are uncorrelated across time, all cross terms vanish, so

$$\sigma_y^2 = \sum_{i=0}^{\infty} (\phi_1^i)^2 \operatorname{var}(\varepsilon_{t-i}).$$

Since $\operatorname{var}(\varepsilon_{t-i}) = \sigma^2$,

$$\sigma_y^2 = \sum_{i=0}^{\infty} \phi_1^{2i} \sigma^2 = \sigma^2 \sum_{i=0}^{\infty} \phi_1^{2i}. \tag{1.15}$$

This is a geometric series. It converges if and only if

$$|\phi_1| < 1.$$

When $|\phi_1| < 1$,

$$\sum_{i=0}^{\infty} \phi_1^{2i} = \frac{1}{1 - \phi_1^2},$$

so

$$\sigma_y^2 = \frac{\sigma^2}{1 - \phi_1^2}.$$

If $|\phi_1| \geq 1$, the variance is infinite, so the process is nonstationary.

Thus a necessary condition for stationarity of the AR(1) process is

$$|\phi_1| < 1.$$

Autocovariances of an AR(1) process

For the AR(1) process, the autocovariance at lag s is

$$\sigma_s = \text{cov}(y_t, y_{t-s}) = \phi_1^s \sigma_y^2,$$

and the autocorrelation is

$$\rho_s = \phi_1^s, \qquad s = 1, 2, \ldots$$

These are finite and depend only on the lag s, not on time t, provided $|\phi_1| < 1$.

So $|\phi_1| < 1$ is both necessary and sufficient for covariance stationarity of the AR(1) process.

Ergodicity of AR(1)

To show ergodicity, consider the sum of absolute autocorrelations:

$$\sum_{s=0}^{\infty} |\rho_s| = \sum_{s=0}^{\infty} |\phi_1|^s.$$

This is again a geometric series, which converges if $|\phi_1| < 1$. Therefore,

$$\sum_{s=0}^{\infty} |\rho_s| = \frac{1}{1 - |\phi_1|} < \infty.$$

Therefore the AR(1) process is ergodic whenever $|\phi_1| < 1$.

Intuition for AR(1)

The AR(1) process is stationary only when the effect of a shock shrinks over time.

If $|\phi_1| < 1$, then

$$\phi_1,\ \phi_1^2,\ \phi_1^3, \ldots$$

get smaller and smaller, so the effect of old shocks eventually dies out.

If $|\phi_1| \geq 1$, old shocks do not disappear, so the process does not settle into a stable distribution.

General AR(p) case and polynomial roots

For the general AR(p) process, the stationarity condition is easier to express using the lag polynomial

$$\phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \cdots - \phi_p L^p.$$

We study the roots of the polynomial equation

$$\phi(L) = 0.$$

For the AR(1) process, this becomes

$$1 - \phi_1 L = 0,$$

so the root is

$$L = \frac{1}{\phi_1}.$$

The condition $|\phi_1| < 1$ is equivalent to

$$\left|\frac{1}{\phi_1}\right| > 1.$$

So the AR(1) process is stationary if and only if the root of its lag polynomial lies outside the unit circle.

This generalizes to AR(p):

> An AR(p) process is stationary if and only if all roots of $\phi(L) = 0$ lie outside the unit circle.

That means if the roots are $r_1, \ldots, r_p$, then stationarity requires

$$|r_j| > 1 \qquad \text{for all } j = 1, \ldots, p.$$

Example 1: stationary AR(2)

Consider

$$y_t = 0.75y_{t-1} - 0.125y_{t-2} + \varepsilon_t.$$

The lag polynomial is

$$1 - 0.75L + 0.125L^2.$$

Factor it:

$$1 - 0.75L + 0.125L^2 = (1 - 0.5L)(1 - 0.25L).$$

So the roots are

$$r_1 = \frac{1}{0.5} = 2, \qquad r_2 = \frac{1}{0.25} = 4.$$

Both roots are greater than one in absolute value, so this AR(2) process is stationary.

Example 2: nonstationary AR(2)

Now consider

$$y_t = 1.25y_{t-1} - 0.25y_{t-2} + \varepsilon_t.$$

The lag polynomial is

$$1 - 1.25L + 0.25L^2.$$

Factor it:

$$1 - 1.25L + 0.25L^2 = (1 - L)(1 - 0.25L).$$

So the roots are

$$r_1 = 1, \qquad r_2 = 4.$$

One root equals 1, so it lies on the unit circle, not outside it. Therefore this process is nonstationary. Such a root is called a **unit root**.

1.3.5 Stationarity in ARMA(p,q) processes

We have established two main facts:

- all finite moving-average processes are stationary,
- autoregressive processes may or may not be stationary, depending on their parameters.

So for an ARMA(p,q) process,

$$\phi(L)y_t = \theta(L)\varepsilon_t,$$

the stationarity condition depends only on the autoregressive part, not on the moving-average part.

That is:

> The ARMA(p,q) process is stationary if and only if all roots of the autoregressive polynomial $\phi(L)$ lie outside the unit circle.

Equivalently, stationarity requires that the p roots of

$$\phi(L) = 0$$

all satisfy

$$|r_j| > 1 \qquad \text{for } j = 1, \ldots, p.$$

The moving-average polynomial $\theta(L)$ does not affect stationarity.

So the full conclusion is:

- MA(q): always stationary if q is finite,
- AR(p): stationary only if the roots of $\phi(L)$ are outside the unit circle,
- ARMA(p,q): same stationarity condition as the AR part alone.

1.4 Random walks and other "unit-root" processes

An important class of nonstationary processes are those that have roots on, but not inside, the unit circle. The simplest example of such a process is the autoregressive process AR(1) with parameter

$$\phi_1 = 1,$$

which is called a **random walk**.

Substituting $\phi_1 = 1$ into the AR(1) equation (1.13), we get the random-walk process

$$y_t = y_{t-1} + \varepsilon_t.$$

In words, the value of y in period t is equal to its value in the previous period plus a random "step" caused by the white-noise shock ε_t.

Rearranging the equation by subtracting y_{t-1} from both sides gives

$$\Delta y_t = y_t - y_{t-1} = \varepsilon_t.$$

Thus, the **first difference** of y, the change in y from one period to the next, is white noise.

We can also express the random walk using lag-operator notation. Writing the model in polynomial form gives

$$(1 - L)y_t = \varepsilon_t. \tag{1.16}$$

The expression $(1 - L)$ is called the **difference operator** (Δ). It represents the difference between the current value of the series and its value in the previous period.

The polynomial $(1 - L)$ has a root equal to 1, which means the process has a **unit root**. For this reason, the random walk is a nonstationary process.

First differencing:

Define a new time series z_t as the first difference of y_t:

$$z_t = \Delta y_t = (1 - L)y_t = y_t - y_{t-1}. \tag{1.17}$$

If y_t follows a random walk, then substituting equation (1.17) into equation (1.16) gives

$$z_t = \varepsilon_t.$$

Since ε_t is white noise, the differenced series z_t is stationary. Therefore, taking the first difference removes the nonstationarity of the random walk.

Differencing unit-root processes:

The random walk illustrates a more general idea: a process that contains unit roots can be made stationary by differencing the series.

Suppose that y_t follows an autoregressive process of order p:

$$\phi(L)y_t = \varepsilon_t,$$

and suppose that d of the p roots of $\phi(L)$ are equal to one, while the remaining $p - d$ roots lie outside the unit circle.

In this case, the lag polynomial can be written as

$$\phi(L) = \left(1 - \frac{1}{r_1}L\right)\left(1 - \frac{1}{r_2}L\right)\cdots\left(1 - \frac{1}{r_p}L\right),$$

where $r_1, r_2, \ldots, r_p$ are the roots of the polynomial.

If d of these roots are equal to one, then the polynomial can be factored as

$$\phi(L) = (1 - L)^d \left(1 - \frac{1}{r_{d+1}}L\right)\cdots\left(1 - \frac{1}{r_p}L\right). \tag{1.18}$$

To remove the unit roots, we difference the series d times.

Define

$$z_t = \Delta^d y_t = (1 - L)^d y_t. \tag{1.19}$$

Substituting this into equation (1.18) gives

$$\phi(L)y_t = \left(1 - \frac{1}{r_{d+1}}L\right) \cdots \left(1 - \frac{1}{r_p}L\right) z_t = \phi^*(L)z_t = \varepsilon_t. \tag{1.20}$$

Because the remaining roots $r_{d+1}, \ldots, r_p$ lie outside the unit circle, the differenced series z_t follows a stationary autoregressive process.

Thus, differencing the series d times removes the unit roots and produces a stationary process.

Integrated processes:

A nonstationary time series that becomes stationary after differencing d times is said to be **integrated of order** d, written as

$$I(d).$$

The term "integrated" comes from calculus, where integration and differentiation are inverse operations.

If z_t is stationary and

$$z_t = \Delta^d y_t,$$

then y_t can be viewed as the result of integrating the stationary series z_t d times.

ARIMA processes:

The ideas above can be extended to more general models that include both autoregressive and moving-average components.

The general integrated time-series model is written as

$$\phi(L)\Delta^d y_t = \theta(L)\varepsilon_t,$$

where

- $\phi(L)$ is an autoregressive polynomial of order p,
- $\theta(L)$ is a moving-average polynomial of order q,
- Δ^d represents differencing the series d times.

This model is called an

Autoregressive Integrated Moving Average process

or

$$\textbf{ARIMA}(p, d, q).$$

ARIMA models play a central role in time-series econometrics because they help us to model nonstationary data by transforming it into a stationary process through differencing.

Chapter 2: Correlation

2.1 What is Correlation?

After removing trend and seasonal effects from a series, we are left with the random component. But random does not mean independent. Consecutive observations are often correlated, and if we can measure and model that correlation, we can make much better forecasts.

The key tools are the autocovariance function, the autocorrelation function, and the correlogram.

2.2 Expectation and the Ensemble

2.2.1 Expected value

The **expected value** (or **expectation**) of a random variable is its long-run average across the whole population. We write it as $E(x)$, and it equals the population mean μ.

If we have two variables x and y, we can extend the idea of variance to the **covariance**, which measures the linear association between them.

Population covariance:

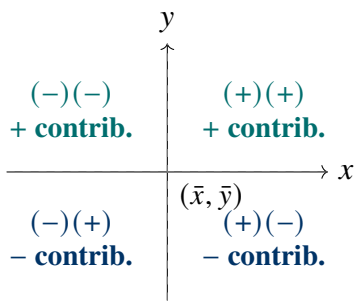
$$\gamma(x, y) = E\big[(x - \mu_x)(y - \mu_y)\big] \tag{2.1}$$

Sample covariance (unbiased estimator, denominator $n - 1$):

$$\mathrm{Cov}(x, y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n - 1} \tag{2.2}$$

Intuition for covariance

Plot the pairs (x_i, y_i) and draw the lines $x = \bar{x}$ and $y = \bar{y}$. These lines divide the plot into four quadrants.



- If y tends to increase with x: most points land in the top-right and bottom-left quadrants, so the covariance is **positive**.
- If y tends to decrease with x: most points land in the top-left and bottom-right quadrants, so the covariance is **negative**.
- If there is no linear association: positive and negative contributions roughly cancel, and the covariance is **near zero**.

Important: covariance depends on the units of measurement. Two completely unrelated series can have a large covariance just because the numbers are large. This is why we normalise to get **correlation**.

Population correlation:

$$\rho(x, y) = \frac{E\left[(x - \mu_x)(y - \mu_y)\right]}{\sigma_x \sigma_y} = \frac{\gamma(x, y)}{\sigma_x \sigma_y} \tag{2.4}$$

Sample correlation:

$$\text{Cor}(x, y) = \frac{\text{Cov}(x, y)}{\text{sd}(x) \cdot \text{sd}(y)} \tag{2.5}$$

Correlation always satisfies $-1 \leq \rho \leq +1$ and has no units.

R: computing covariance and correlation

```
# Sample covariance (denominator n-1)
cov(x, y)
```

```
# From the definition (divides by n, not n-1)
mean((x - mean(x)) * (y - mean(y)))

# Sample correlation
cor(x, y)

# Equivalent from definition
cov(x, y) / (sd(x) * sd(y))
```

Remark. `cov()` uses denominator $n - 1$ (unbiased). The expression `mean(...)`, by contrast, divides by n. For large n the difference vanishes, but they give slightly different numbers in small samples.

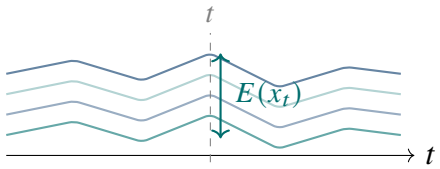
2.2.2 The ensemble and stationarity

The **mean function** of a time series model is

$$\mu(t) = E(x_t), \tag{2.6}$$

which is, in general, a different value at every time t.

The expectation here is taken across the **ensemble**: the entire population of all possible time series that the model could have produced. Think of it as running the same random process many times independently and averaging across all those runs at a fixed time t.



With real data, we only ever observe *one* time series. So we have two choices:

1. Estimate $\mu(t)$ at each time t separately (this requires modeling the trend explicitly).
2. Assume the mean is **constant** across time: $\mu(t) = \mu$ for all t.

If the mean function is constant, the model is said to be **stationary in the mean**, and the single sample mean

$$\bar{x} = \frac{1}{n} \sum_{t=1}^{n} x_t \tag{2.7}$$

is the natural estimate of μ.

In practice, we remove any trend and seasonal effects first (using, for example, `decompose` in R), and then the residual series can reasonably be treated as stationary in the mean.

2.2.3 Ergodic series

A time series model that is stationary in the mean is **ergodic in the mean** if the time average converges to the ensemble mean as the series length grows:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} x_t = \mu. \tag{2.8}$$

Ergodicity is the property that justifies using a single long time series to estimate population parameters. Without it, no matter how much data you collected, you could not recover μ.

All time series models in this course are ergodic. The assumption matters conceptually but rarely requires explicit verification in standard econometric applications.

2.2.4 Variance function

The **variance function** of a model that is stationary in the mean is

$$\sigma^2(t) = E\left[(x_t - \mu)^2\right]. \tag{2.9}$$

This can, in principle, take a different value at every t. But since we only observe one realisation, we cannot estimate a separate variance at every date.

So we add the assumption that the model is also **stationary in the variance**:

$$\sigma^2(t) = \sigma^2 \quad \text{for all } t.$$

Under this assumption, the single estimate

$$\text{Var}(x) = \frac{\sum(x_t - \bar{x})^2}{n-1} \tag{2.10}$$

estimates the common population variance σ^2.

Remark. If consecutive observations are positively correlated (as is common in time series), then

$\text{Var}(x)$ tends to *underestimate* the true variance in short samples, because nearby observations look more similar than independent draws would. In large samples this bias disappears quickly.

2.2.5 Autocorrelation

The mean and variance describe the distribution at a single point in time. To describe how observations at *different* times relate to each other, we need the autocovariance and autocorrelation functions.

A time series model is **second-order stationary** (also called **weakly stationary**) if it is stationary in the mean and variance, *and* the correlation between any two observations depends only on how far apart they are in time, not on when in the series they occur.
The gap in time between two observations is called the **lag**, denoted k.

Under second-order stationarity, we define the **autocovariance function (acvf)**:

$$\gamma_k = E[(x_t - \mu)(x_{t+k} - \mu)] \tag{2.11}$$

Note that γ_k does not depend on t — only on the lag k. This follows directly from the stationarity assumption.

The **autocorrelation function (acf)**, ρ_k, standardises the autocovariance:

$$\rho_k = \frac{\gamma_k}{\sigma^2} \tag{2.12}$$

Since $\gamma_0 = \sigma^2$ (the variance of x_t with itself at lag zero), we always have $\rho_0 = 1$.

Sample estimates

From a time series of length n, the **sample autocovariance** at lag k is:

$$c_k = \frac{1}{n}\sum_{t=1}^{n-k}(x_t - \bar{x})(x_{t+k} - \bar{x}) \tag{2.13}$$

Note two things about this formula. First, the denominator is n (not $n - k$), even though only $n - k$ terms appear in the sum. This convention ensures that all sample autocorrelations lie in $[-1, 1]$. Second, c_0 is the variance with denominator n.

The **sample autocorrelation** at lag k is:

$$r_k = \frac{c_k}{c_0} \tag{2.14}$$

R: computing autocorrelations

```
# The full acf (also plots the correlogram by default)
acf(x)

# Extract the lag-k autocorrelation
# (lag 0 is stored at index 1, so lag k is at index k+1)
acf(x)$acf[k + 1]

# Example: lag 1 autocorrelation
acf(x)$acf[2]

# Autocovariance instead of autocorrelation
acf(x, type = "covariance")$acf[2]
```

2.3 The Correlogram

2.3.1 What it is

The **correlogram** is a plot of the sample autocorrelation r_k on the y-axis against the lag k on the x-axis. In R, `acf(x)` produces this plot by default.

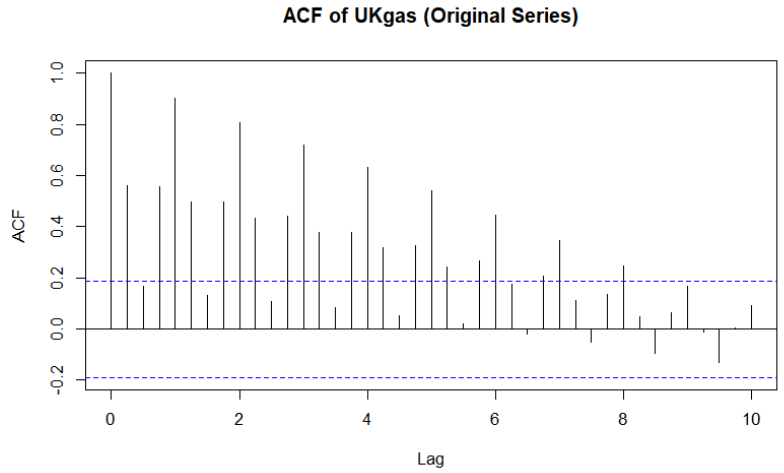


Figure 2.1: Correlogram of the UKgas series (original, untransformed). The slow near-linear decay at low lags is the signature of a trend. The spikes repeating every 4 lags reflect the quarterly seasonal pattern. Neither feature means the series is useful for modeling yet — both the trend and seasonality need to be removed first.

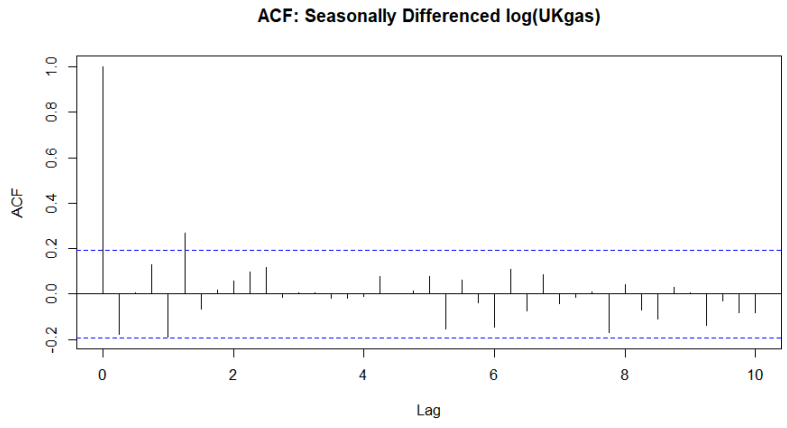


Figure 2.2: Correlogram of the seasonally differenced log(UKgas). The slow decay and seasonal spikes are gone. Most autocorrelations fall inside the significance bands, indicating the transformation has brought the series close to stationarity.

Reading a correlogram: key features

1. **Lag 0 is always 1.** It is shown on the plot as a reference point.
2. **Significance bands.** Under the null hypothesis $H_0 : \rho_k = 0$, the sampling distribution of r_k is approximately normal with mean $-1/n$ and variance $1/n$. The dashed lines on the correlogram are drawn at
$$-\frac{1}{n} \pm \frac{2}{\sqrt{n}}.$$
If r_k falls outside these bands, we have evidence against H_0 at the 5% level.
3. **Multiple testing caution.** If all $\rho_k = 0$, we still expect roughly 5% of the r_k values to fall outside the bands by chance alone. Do not treat every crossing as strong evidence of autocorrelation.
4. **Practical versus statistical significance.** For long series, even very small autocorrelations can be statistically significant. Squaring the autocorrelation gives the percentage of variance in x_t explained by a linear relationship with x_{t-k}. A value of $r_k = 0.1$ explains only 1% of the variance, which is practically negligible.

2.3.2 What different patterns mean

The shape of the correlogram tells you about the structure of the series.

Pattern	What it suggests
Slow, near-linear decay	A trend is present in the data. Autocorrelations are large and positive because observations close in time are very similar.
Damped cosine shape	Consistent with an AR(2) model.
Exponential decay	Consistent with an AR(1) model.
Single dominant spike at lag k	The series repeats with period k (e.g., a seasonal pattern).
Seasonal spikes on top of a trend decay	Both trend and seasonality are present.
All bars inside the bands	The series looks like white noise. No significant linear dependence.

2.3.3 Example: air passenger series

The air passenger data has a clear upward trend and seasonal variation. Here is the R workflow to deseasonalise the series, remove the trend, and inspect the residuals.

```
data(AirPassengers)
AP <- AirPassengers

# Decompose multiplicatively (variance grows with the trend)
AP.decom <- decompose(AP, "multiplicative")

# The centred moving average uses 6 points on each end,
# so the first 6 and last 6 random values are NA
plot(ts(AP.decom$random[7:138]))
acf(AP.decom$random[7:138])

# Check how much the decomposition reduced variability
sd(AP[7:138])                              # original
sd(AP[7:138] - AP.decom$trend[7:138])      # after trend removal
sd(AP.decom$random[7:138])                 # after full decomposition
```

A typical result:

Series	Standard deviation
Original (July to June only)	109
After trend removal	41.1
After seasonal adjustment	0.034

The dramatic drop in standard deviation shows that the decomposition is very effective. The correlogram of the residuals tends to show a damped cosine pattern, suggesting an AR(2) model may be appropriate for the remaining structure.

2.3.4 Example: Font Reservoir inflows

Monthly inflows to the Font Reservoir (Northumberland, UK, 1909–1980) after removing a linear trend and seasonal effects by regression show a clear lag-1 autocorrelation.

```
# Data are the regression residuals stored as 'adflow'
plot(ts(adflow), ylab = "adflow")
acf(adflow, xlab = "lag (months)", main = "")
```

The physical interpretation is simple: a month with above-average inflow is likely followed by another above-average month, because groundwater drains slowly. The correlogram shows exponential decay, which is consistent with an AR(1) model.

2.4 Covariance of Sums of Random Variables

This result is used repeatedly in the derivations of second-order properties for time series models in later chapters. It is worth knowing it well.

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ be random variables. Then

$$\mathrm{Cov}\left(\sum_{i=1}^{n} x_i,\ \sum_{j=1}^{m} y_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathrm{Cov}(x_i,\ y_j). \tag{2.15}$$

The covariance of two sums equals the sum of all pairwise covariances.

Proof

We use three basic facts:

$$\left(\sum_{i=1}^{n} x_i\right)\left(\sum_{j=1}^{m} y_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} x_i y_j$$

$$E\left[\sum_{i=1}^{n} x_i\right] = \sum_{i=1}^{n} E(x_i)$$

$$\mathrm{Cov}(x, y) = E(xy) - E(x)\,E(y)$$

Now expand:

$$\begin{aligned}
\mathrm{Cov}\left(\sum_i x_i,\ \sum_j y_j\right) &= E\left[\sum_i x_i \cdot \sum_j y_j\right] - E\left[\sum_i x_i\right] E\left[\sum_j y_j\right] \\
&= \sum_i \sum_j E(x_i y_j) - \sum_i \sum_j E(x_i)\, E(y_j) \\
&= \sum_i \sum_j \big[E(x_i y_j) - E(x_i)\, E(y_j)\big] \\
&= \sum_i \sum_j \mathrm{Cov}(x_i,\ y_j).
\end{aligned}$$

Useful special case

Set $n = m = 2$ and let $x_i = y_i$. Then equation (2.15) gives:

$$\begin{aligned}
\mathrm{Var}(x_1 + x_2) &= \mathrm{Cov}(x_1, x_1) + \mathrm{Cov}(x_1, x_2) + \mathrm{Cov}(x_2, x_1) + \mathrm{Cov}(x_2, x_2) \\
&= \mathrm{Var}(x_1) + 2\,\mathrm{Cov}(x_1, x_2) + \mathrm{Var}(x_2).
\end{aligned}$$

The familiar result from statistics:

$$\mathrm{Var}(x + y) = \mathrm{Var}(x) + \mathrm{Var}(y) + 2\,\mathrm{Cov}(x, y).$$

R: verify with Herald Square data

```
# x = CO concentration, y = benzoapyrene
var(x + y)
var(x) + var(y) + 2 * cov(x, y)   # should match
```

2.5 R Commands Summary

Command	What it does
`mean(x)`	Sample mean (divides by n)
`var(x)`	Sample variance (denominator $n - 1$)
`sd(x)`	Sample standard deviation
`cov(x, y)`	Sample covariance (denominator $n - 1$)
`cor(x, y)`	Sample correlation
`acf(x)`	Plot the correlogram; returns r_k values
`acf(x, type = "covariance")`	Autocovariance instead of autocorrelation
`acf(x)$acf[k+1]`	Extract lag-k autocorrelation
`decompose(x, type)`	Decompose into trend, seasonal, random

Chapter 3: Forecasting Strategies

Three broad strategies exist for forecasting a time series. The first is to find a related variable that moves *ahead* of the one you care about, and use it as a leading indicator. The second is to model the adoption pattern of a new product using a mathematical diffusion model. The third is to extrapolate current trends and seasonal patterns forward using adaptive smoothing.

3.1 Cross-Correlation and Leading Variables

3.1.1 The idea

Sometimes one variable x consistently moves before another variable y by a fixed number of time steps. If that lead-lag relationship is stable, then x is a useful predictor of future values of y.

To quantify this, we extend the autocorrelation function from Chapter 2 to the **cross-correlation function (ccf)**.

3.1.2 Cross-covariance and cross-correlation

Suppose x_t and y_t are both second-order stationary. The **cross-covariance function (ccvf)** at lag k is:

$$\gamma_k(x, y) = E\left[(x_{t+k} - \mu_x)(y_t - \mu_y)\right] \tag{3.1}$$

The **cross-correlation function (ccf)** standardises this:

$$\rho_k(x, y) = \frac{\gamma_k(x, y)}{\sigma_x \sigma_y} \tag{3.3}$$

Note the direction carefully: at lag $k > 0$, the variable x is shifted k steps *forward* relative to y. So a spike at positive k means x *leads* y (equivalently, y lags x).

This is not symmetric:

$$\gamma_k(x, y) = \gamma_{-k}(y, x). \tag{3.2}$$

Swapping the order of the variables flips the sign of the lag.

Sample estimates

Sample cross-covariance:

$$c_k(x, y) = \frac{1}{n} \sum_{t=1}^{n-k} (x_{t+k} - \bar{x})(y_t - \bar{y}) \tag{3.4}$$

Sample cross-correlation:

$$r_k(x, y) = \frac{c_k(x, y)}{\sqrt{c_0(x, x)\, c_0(y, y)}} \tag{3.5}$$

R: cross-correlation

```
# Plot auto- and cross-correlograms together
acf(ts.union(x.ts, y.ts))

# Single cross-correlogram
ccf(x.ts, y.ts)
```

Remark. The ccf should only be interpreted meaningfully after removing trends and seasonal effects from both series. If both variables share a common trend, the ccf will show large correlations at all lags that have nothing to do with a genuine lead-lag relationship. Use `decompose` on each series first.

Example: building approvals and activity

Australian building approvals are published monthly; building activity is published quarterly and tends to lag approvals by about one quarter. After removing trend and seasonal components from both series with `decompose`, the ccf confirms the lead-lag structure.

```
# Read and construct time series
```

```
App.ts <- ts(Approvals, start = c(1996,1), freq = 4)
Act.ts <- ts(Activity,  start = c(1996,1), freq = 4)

# Remove trend and seasonal effects
app.ran <- decompose(App.ts)$random
act.ran <- decompose(Act.ts)$random

# Trim NAs left by the centred moving average
app.ran.ts <- window(app.ran, start = c(1996,3))
act.ran.ts <- window(act.ran, start = c(1996,3))

# Cross-correlogram
ccf(app.ran.ts, act.ran.ts)
```

The cross-correlogram shows significant values at negative lags, meaning approvals lead activity. Numerical ccf values at lags 0, 1, 2, 3 are approximately 0.43, 0.49, 0.50, 0.46.

3.2 The Bass Model

3.2.1 Background

When a genuinely new product enters the market, there is no past sales history to extrapolate from. Frank Bass (1969) proposed a model that captures two types of buyers:

- **Innovators:** people who buy early out of interest in the novelty, independent of what others do.
- **Imitators:** people who buy later after seeing friends and acquaintances use the product.

The model has three parameters: m (total market size), p (coefficient of innovation), and q (coefficient of imitation).

3.2.2 Model definition

Let N_t be the cumulative number of people who have bought the product by time t. The Bass difference equation is:

$$N_{t+1} = N_t + \underbrace{p(m - N_t)}_{\text{innovators}} + \underbrace{\frac{qN_t}{m}(m - N_t)}_{\text{imitators}} \tag{3.6}$$

The term $(m - N_t)$ is the number of potential buyers who have not yet purchased. The fraction N_t/m is the proportion of the market already reached — this scales the imitation effect.

The closed-form solution for cumulative sales is:

$$N_t = m \cdot \frac{1 - e^{-(p+q)t}}{1 + (q/p)\, e^{-(p+q)t}} \tag{3.7}$$

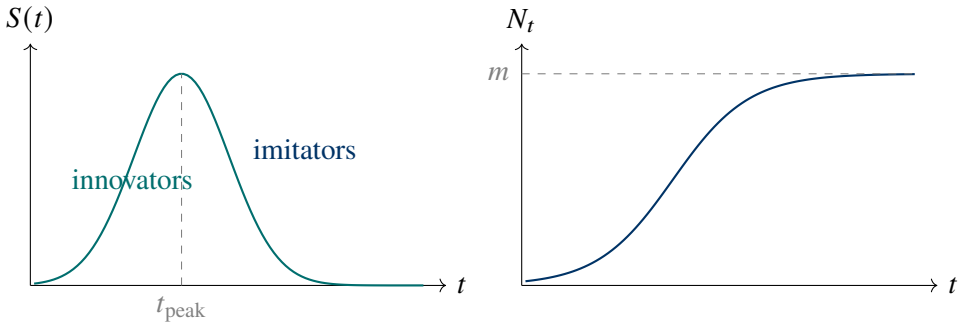
Taking the derivative gives the **sales rate** (sales per unit time):

$$S(t) = m \cdot \frac{(p+q)^2}{p} \cdot \frac{e^{-(p+q)t}}{\left(1 + (q/p)\, e^{-(p+q)t}\right)^2} \tag{3.12}$$

The sales curve rises to a peak, then falls. The time to peak is:

$$t_{\text{peak}} = \frac{\ln(q) - \ln(p)}{p + q} \tag{3.13}$$

3.2.3 Shape of the Bass curve



Sales per period **Cumulative sales**

The left panel is the sales rate: a bell-shaped curve that rises as imitators join innovators, peaks at t_{peak}, then falls as the market saturates. The right panel is cumulative sales: an S-curve that flattens at m.

3.2.4 Parameter values

Product	m	p	q	Source
Typical product	—	0.030	0.380	VBM
35mm projectors (1965–1986)	3.37M	0.009	0.173	Bass
Overhead projectors (1960–1970)	0.96M	0.028	0.311	Bass
PCs (1981–2010)	3.38B	0.001	0.195	Bass

Notice that $q > p$ in every row. Imitation always dominates innovation in these cases, which is typical for consumer products. The ratio q/p controls how steeply the sales curve rises before the peak.

3.2.5 Fitting the Bass model in R

The Bass model is nonlinear in its parameters, so we use `nls` (nonlinear least squares). The example below fits VCR sales in the US from 1980 to 1989.

```
T79   <- 1:10
Tdelt <- (1:100) / 10          # finer grid for smooth curve
Sales <- c(840, 1470, 2110, 4000, 7590, 10950, 10530, 9470, 7790, 5890)
Cusales <- cumsum(Sales)

Bass.nls <- nls(
  Sales ~ M * ((P+Q)^2 / P) * exp(-(P+Q)*T79) /
          (1 + (Q/P)*exp(-(P+Q)*T79))^2,
  start = list(M = 60630, P = 0.03, Q = 0.38)
)
summary(Bass.nls)

# Extract coefficients
Bcoef <- coef(Bass.nls)
m <- Bcoef[1]; p <- Bcoef[2]; q <- Bcoef[3]

# Evaluate the fitted curves on the fine grid
ngete <- exp(-(p+q) * Tdelt)
Bpdf  <- m * ((p+q)^2 / p) * ngete / (1 + (q/p)*ngete)^2  # sales rate
Bcdf  <- m * (1 - ngete) / (1 + (q/p)*ngete)               # cumulative
```

```
# Plot
plot(Tdelt, Bpdf, type = "l", xlab = "Year from 1979", ylab = "Sales/year")
points(T79, Sales)

plot(Tdelt, Bcdf, type = "l", xlab = "Year from 1979", ylab = "Cumulative sales")
points(T79, Cusales)
```

Fitted estimates: $\hat{m} = 68000$, $\hat{p} = 0.0066$, $\hat{q} = 0.64$. Starting values for p and q are taken from the typical-product row in the table above. The starting value for M is set to the observed total sales.

Remark. The Bass model is most valuable when you *do not* have past sales data for the exact product, because you can borrow parameter estimates from analogous products. The forecasts will be uncertain, but they are the best available information for planning and investment decisions.

3.3 Exponential Smoothing and Holt-Winters

3.3.1 Exponential smoothing

Exponential smoothing handles the case where there is no clear trend or seasonality, but the mean level of the series can drift slowly over time. The model is:

$$x_t = \mu_t + w_t \tag{3.14}$$

where μ_t is the current (unknown) mean level and w_t is white noise with standard deviation σ.

We estimate μ_t by a_t, a weighted average of the latest observation and the previous estimate:

EWMA updating equation:

$$a_t = \alpha x_t + (1 - \alpha) a_{t-1}, \qquad 0 < \alpha < 1 \tag{3.15}$$

Forecast for any lead time k:

$$\hat{x}_{n+k|n} = a_n \qquad k = 1, 2, \ldots \tag{3.16}$$

Since there is no trend, every forecast from time n is just the current estimated mean.

Three equivalent ways to write the EWMA

These three forms are all identical; each one reveals something different.

Form 1 – error correction:

$$a_t = a_{t-1} + \alpha(x_t - a_{t-1}) \tag{3.17}$$

Each step, we update our estimate by a fraction α of the one-step forecast error. If α is small, we barely react to new data. If α is large, we react aggressively.

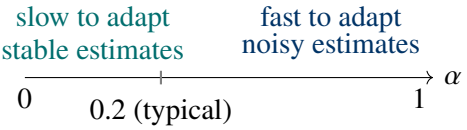
Form 2 – weighted history: By substituting Form 1 repeatedly back into itself:

$$a_t = \alpha x_t + \alpha(1-\alpha)x_{t-1} + \alpha(1-\alpha)^2 x_{t-2} + \cdots \tag{3.18}$$

This shows that a_t is a weighted average of all past observations, with weights that decay geometrically. The most recent observation gets weight α, the one before gets $\alpha(1-\alpha)$, and so on. The weights sum to 1 (geometric series with ratio $1-\alpha$).

Form 3 – original definition (3.15): the compact recursive form used in computation.

Choosing α



R estimates α by minimising the sum of squared one-step prediction errors:

$$\text{SS1PE} = \sum_{t=2}^{n} e_t^2, \qquad e_t = x_t - a_{t-1} \tag{3.19, 3.20}$$

Remark. If the series is long and the mean has been stable, R may return a very small α. This is optimal for the historical data but makes the smoother slow to react if the mean suddenly shifts. This is a real practical risk.

R: exponential smoothing

```
# Exponential smoothing: set beta = 0 and gamma = 0
# Let R choose alpha
```

```
hw1 <- HoltWinters(x.ts, beta = 0, gamma = 0)
hw1           # see estimated alpha and final level
hw1$SSE       # sum of squared one-step errors

# Specify alpha manually
hw2 <- HoltWinters(x.ts, alpha = 0.2, beta = 0, gamma = 0)
hw2$SSE

# Plot fitted values against observations
plot(hw1)
```

3.3.2 Holt-Winters method

Exponential smoothing has no mechanism to track a trend or seasonal pattern. The Holt-Winters method extends it by maintaining three adaptive estimates at each time step: the level a_t, the slope b_t, and the seasonal effect s_t.

Additive seasonal form

Use this when the amplitude of the seasonal swings is roughly constant regardless of the level of the series.

Updating equations (period p):

$$a_t = \alpha(x_t - s_{t-p}) + (1-\alpha)(a_{t-1} + b_{t-1}) \tag{3.21a}$$

$$b_t = \beta(a_t - a_{t-1}) + (1-\beta)\, b_{t-1} \tag{3.21b}$$

$$s_t = \gamma(x_t - a_t) + (1-\gamma)\, s_{t-p} \tag{3.21c}$$

Forecast ($k \le p$):

$$\hat{x}_{n+k|n} = a_n + k\, b_n + s_{n+k-p} \tag{3.22}$$

Reading the three equations:

- Equation (3.21a): The level estimate is a weighted average of the seasonally adjusted observation $(x_t - s_{t-p})$ and the one-step-ahead level forecast $(a_{t-1} + b_{t-1})$.
- Equation (3.21b): The slope is updated by comparing the current and previous level estimates.

- Equation (3.21c): The seasonal factor is updated from the most recent estimate of that season's deviation.

The forecast adds the expected level ($a_n + k\,b_n$) to the most recent estimate of the appropriate seasonal effect (s_{n+k-p}).

Multiplicative seasonal form

Use this when the amplitude of the seasonal swings grows proportionally with the level, as in the UKgas data (Figure 3.1).

Updating equations:

$$a_n = \alpha\left(\frac{x_n}{s_{n-p}}\right) + (1-\alpha)(a_{n-1} + b_{n-1}) \tag{3.23a}$$

$$b_n = \beta(a_n - a_{n-1}) + (1-\beta)\,b_{n-1} \tag{3.23b}$$

$$s_n = \gamma\left(\frac{x_n}{a_n}\right) + (1-\gamma)\,s_{n-p} \tag{3.23c}$$

Forecast:

$$\hat{x}_{n+k|n} = (a_n + k\,b_n)\,s_{n+k-p} \tag{3.24}$$

The difference is that seasonal effects now multiply the level rather than shift it. The forecast is the expected level times the appropriate seasonal factor.

Comparison: additive vs. multiplicative

	Additive	Multiplicative
Seasonal amplitude	Constant	Grows with level
Seasonal update	$x_t - a_t$	x_t/a_t
Forecast	$a_n + kb_n + s_{n+k-p}$	$(a_n + kb_n) \cdot s_{n+k-p}$
When to use	Stable amplitude	Proportional amplitude

R: Holt-Winters

```
# Multiplicative Holt-Winters (trend + multiplicative seasonal)
hw <- HoltWinters(x.ts, seasonal = "mult")
hw              # estimated alpha, beta, gamma and coefficients
hw$SSE          # sum of squared one-step errors

# Compare to naive benchmark: predict next = current
# (useful sanity check)
sqrt(hw$SSE / length(x.ts))   # RMS prediction error
sd(x.ts)                       # baseline: predict the mean

# Fitted values vs. observed
plot(hw)

# Forecast k steps ahead
pred <- predict(hw, n.ahead = k)
ts.plot(x.ts, pred, lty = 1:2)
```

3.3.3 Application to UKgas

The UKgas series (Figure **??**) shows both a strong upward trend and seasonal variation whose amplitude grows with the level. A multiplicative Holt-Winters model is the natural choice.

```
ukgas.hw <- HoltWinters(UKgas, seasonal = "mult")
plot(ukgas.hw)                                  # Figure: hw_fit
pred <- predict(ukgas.hw, n.ahead = 8)
ts.plot(UKgas, pred, lty = 1:2)            # Figure: hw_forecast
```

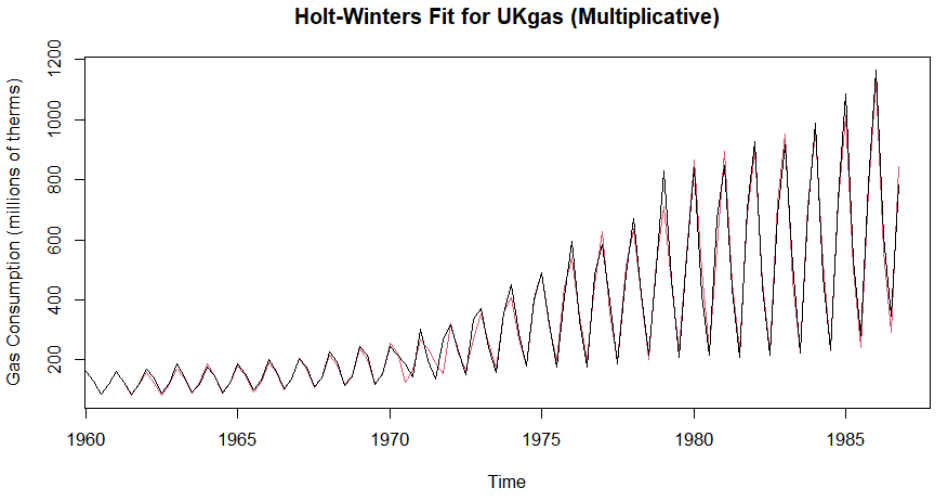


Figure 3.1: Holt-Winters multiplicative fit for UKgas. The fitted values (red) track the observed series (black) closely even as the seasonal amplitude grows.

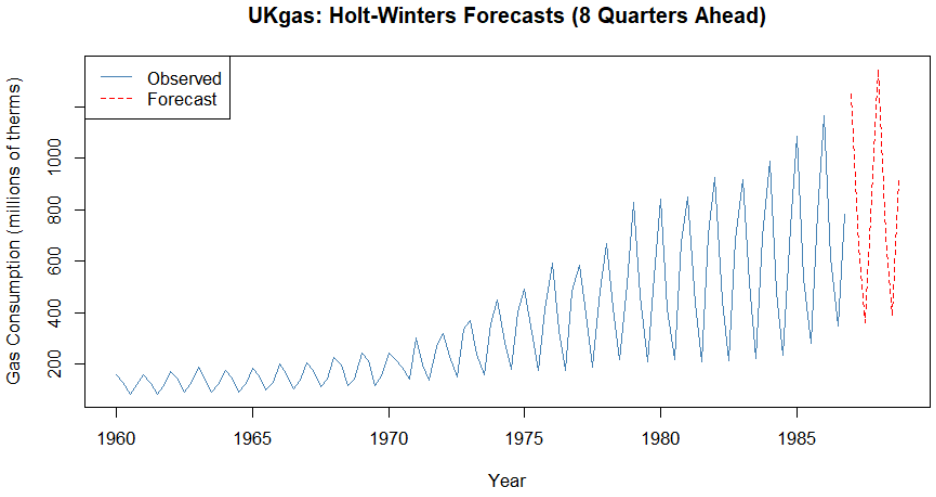


Figure 3.2: Holt-Winters 8-quarter-ahead forecasts for UKgas (dashed). The forecast carries forward the trend and seasonal structure estimated from the data.

The forecast extrapolates under the assumption that the current trend and seasonal pattern continue. This is a reasonable working assumption over a short horizon, but unforeseen structural breaks (a policy change, an energy price shock) would invalidate it.

3.4 R Commands Summary

Command	What it does
`ccf(x, y)`	Cross-correlogram of x and y
`acf(ts.union(x, y))`	ACFs and CCFs in one figure
`ts.union(x, y)`	Bind two series with common frequency
`nls(...)`	Nonlinear least squares
`coef(fit)`	Extract fitted coefficients
`cumsum(x)`	Cumulative sum
`HoltWinters(x)`	Fit exponential smoothing / Holt-Winters
`beta = 0, gamma = 0`	Pure exponential smoothing
`seasonal = "mult"`	Multiplicative seasonal form
`predict(hw, n.ahead = k)`	Forecast k steps ahead
`ts.plot(x, pred, lty = 1:2)`	Plot observed + forecast on same axes

Chapter 4: Basic Stochastic Models

We already described and decomposed time series: find the trend, find the seasonal pattern, smooth it, forecast it. That is useful, but it is not modeling in the deeper sense. We were describing what the data look like, not explaining the mechanism that produced them.

No we learn what kind of stochastic process could have generated a series. Once we have a stochastic model, we can derive its statistical properties mathematically, simulate it, fit it to data, check whether it fits, and compare it to competing models.

The three fundamental building blocks are white noise, random walks, and autoregressive processes. Everything in the rest of the course (MA, ARMA, ARIMA) is built on top of these.

4.1 White Noise

4.1.1 Definition

After we fit a model to a time series, the residuals are what the model could not explain. If the model has captured all the systematic structure in the data, the residuals should be pure randomness with no pattern left over. The formal name for that ideal residual series is **discrete white noise**.

A time series $\{w_t : t = 1, 2, \ldots, n\}$ is **discrete white noise (DWN)** if:

$$w_1, w_2, \ldots, w_n \quad \text{are independent and identically distributed with } E(w_t) = 0.$$

This implies:

$$\text{Var}(w_t) = \sigma^2 \quad \text{for all } t, \qquad \text{Cov}(w_i, w_j) = 0 \quad \text{for all } i \neq j. \tag{4.2}$$

If the w_t are also normally distributed, the series is called **Gaussian white noise**: $w_t \sim N(0, \sigma^2)$.

The autocorrelation function of a white noise process is:

$$\rho_k = \begin{cases} 1 & k = 0 \\ 0 & k \neq 0 \end{cases} \tag{4.3}$$

In words: a white noise series has no memory at all. The past carries no information about the future.

4.1.2 Why white noise?

White noise plays two roles.

Role 1: The innovation. Every stochastic model we build contains a white noise term w_t. It represents the genuinely new, unpredictable information that arrives at each time step. The model says how past values influence the present, and w_t is whatever the past cannot explain.

Role 2: The diagnostic target. A well-fitted model should leave residuals that look like white noise. If the residual ACF still shows patterns, the model has missed structure. This idea runs through every model-fitting exercise in the rest of the course.

4.1.3 Simulation and the sample ACF

```
set.seed(1)
w <- rnorm(100)
plot(w, type = "l", main = "Simulated Gaussian White Noise")
acf(w)
```

The time plot looks irregular and patternless. The ACF shows all bars near zero — but not exactly zero. This is expected.

Key point about the sample ACF:
Even when the true process is white noise ($\rho_k = 0$ for all $k \neq 0$), the sample autocorrelations r_k will not be exactly zero because of sampling variation.
Under $H_0 : \rho_k = 0$, each r_k is approximately $N(-1/n, 1/n)$. So we expect about 5% of r_k values to fall outside the significance bands purely by chance, even in a true white noise series.
Implication: One or two bars crossing the dashed lines in an ACF plot is not alarming. Look for *patterns* — many consecutive significant values, or spikes at theoretically meaningful lags.

4.2 Random Walks

4.2.1 Definition and back substitution

A time series $\{x_t\}$ is a **random walk** if

$$x_t = x_{t-1} + w_t \tag{4.4}$$

where $\{w_t\}$ is white noise.

Each new value equals the previous value plus a random shock. To understand what this implies about the whole history, substitute backwards repeatedly:

$$\begin{aligned} x_t &= x_{t-1} + w_t \\ &= (x_{t-2} + w_{t-1}) + w_t \\ &= (x_{t-3} + w_{t-2}) + w_{t-1} + w_t \\ &\vdots \\ &= w_1 + w_2 + \cdots + w_t \end{aligned}$$

$$x_t = \sum_{i=1}^{t} w_i \tag{4.6}$$

The current value of a random walk is the *cumulative sum* of all past shocks. Every shock is permanently embedded in all future values.

This is the essential difference from white noise: in white noise, shocks are one-and-done. In a random walk, they accumulate forever.

4.2.2 The backward shift operator

The backward shift operator B provides compact notation for lag operations.

$$Bx_t = x_{t-1}, \qquad B^n x_t = x_{t-n} \tag{4.7–4.8}$$

Using B, the random walk equation becomes:

$$x_t = Bx_t + w_t \implies (1 - B)x_t = w_t$$

The operator $(1 - B)$ is the **difference operator** ∇:

$$\nabla x_t = x_t - x_{t-1} = (1 - B)x_t \tag{4.11}$$

Higher-order differencing:

$$\nabla^n = (1 - B)^n \tag{4.12}$$

The random walk says $(1 - B)x_t = w_t$. Dividing both sides by $(1 - B)$ recovers equation (4.6) via the geometric series $(1 - B)^{-1} = 1 + B + B^2 + \cdots$.

4.2.3 Second-order properties: why the random walk is nonstationary

From equation (4.6), the random walk is a sum of t independent white noise terms. Using the covariance-of-sums result from Chapter 2:

$$\mu_x = 0, \qquad \gamma_k(t) = \text{Cov}(x_t, x_{t+k}) = t\sigma^2 \tag{4.9}$$

The variance is $\text{Var}(x_t) = t\sigma^2$, which grows without bound as t increases.

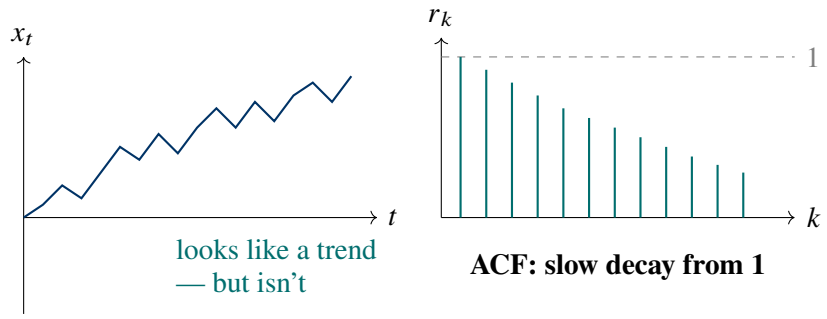
This is why the random walk is nonstationary: its variance depends on t. There is no stable distribution that the process settles into. The further out we are in time, the more uncertain the series is.

The time-varying autocorrelation at lag $k > 0$ follows from (4.9):

$$\rho_k(t) = \frac{t\sigma^2}{\sqrt{t\sigma^2 \cdot (t + k)\sigma^2}} = \frac{1}{\sqrt{1 + k/t}} \tag{4.10}$$

For large t with $k \ll t$, this is close to 1. So the ACF of a random walk starts near 1 and decays very slowly — a completely different shape from the ACF of a stationary process.

4.2.4 What a random walk looks like



Time plot

The left panel shows a typical random walk path. It looks like it has an upward trend, but this is entirely the result of accumulated random shocks — there is no deterministic trend function in the model. This is a critical lesson for data analysis:

Lesson: A series can visually look like it is trending even when no deterministic trend is present. The apparent drift may be the result of cumulative random shocks. Do not confuse visual pattern with model mechanism.

4.2.5 First differencing removes nonstationarity

Since $(1 - B)x_t = w_t$, the first difference of a random walk is white noise:

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

```
# Simulate a random walk
set.seed(1)
w <- rnorm(1000)
x <- cumsum(w)          # cumulative sum = random walk

# Inspect the original
plot(x, type = "l")
acf(x)                  # slow decay from 1

# Difference it
dx <- diff(x)
plot(dx, type = "l")
```

```
acf(dx)                    # should look like white noise
```

Diagnostic workflow for a suspected random walk:

1. Plot the series. Does it wander without a fixed level?
2. Check the ACF. Does it decay very slowly from nearly 1?
3. Take first differences. Does the differenced ACF look like white noise?
4. If yes: the random walk is a reasonable model.

4.2.6 Random walk with drift

A plain random walk has no directional tendency on average: $E(\nabla x_t) = 0$. We can add a constant drift δ:

$$x_t = x_{t-1} + \delta + w_t$$

Now $E(\nabla x_t) = \delta$. The series has a systematic upward (if $\delta > 0$) or downward (if $\delta < 0$) movement on average, on top of the cumulative random variation.

In practice, δ is estimated as the sample mean of the first differences:

$$\hat{\delta} = \frac{1}{n-1} \sum_{t=2}^{n} (x_t - x_{t-1}) = \overline{\nabla x}.$$

A 95% confidence interval is $\hat{\delta} \pm 2 \cdot \frac{s_{\nabla x}}{\sqrt{n-1}}$, where $s_{\nabla x}$ is the standard deviation of the differences. If this interval excludes 0, we have evidence of genuine drift.

```
DP <- diff(Price)
mean(DP) + c(-2, 2) * sd(DP) / sqrt(length(DP))
```

Forecasting implications:

Model	Forecast at lead k	Shape
Plain random walk	x_n (flat)	horizontal line
Random walk with drift	$x_n + k\hat{\delta}$	straight line with slope $\hat{\delta}$

The drift model gives a forecast that moves in the estimated direction, but the uncertainty still grows like $\sqrt{t}$ because of the cumulative shocks.

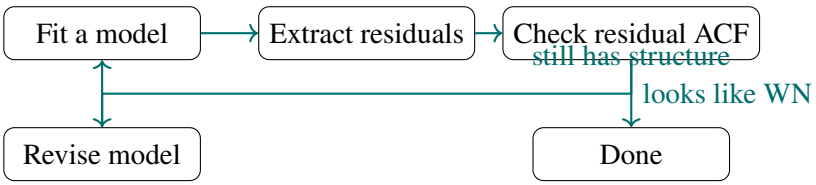
4.3 Fitted Models and Diagnostic Plots

4.3.1 The logic of residual diagnostics

This is the core idea that runs through the rest of the course.

Model-checking principle:
A fitted model is adequate if and only if the residuals look approximately like white noise. If the residual ACF still shows structure, the model has not extracted all the predictable information from the data.

The workflow is always:



4.3.2 Example: exchange rate series

The GBP to NZD quarterly exchange rate (1991–2000) was introduced in Chapter 1 as a series with a stochastic trend. We now try to model it.

Step 1: Difference and check ACF.

```
acf(diff(Z.ts))
```

The ACF of the differenced series shows one significant spike at lag 1. That means the plain random walk is not quite adequate — the differenced series is not fully white noise. But it is close.

Step 2: Extend with Holt-Winters.

```
Z.hw <- HoltWinters(Z.ts, alpha = 1, gamma = 0)
acf(resid(Z.hw))
```

Setting $\alpha = 1$ means the level is updated entirely from the most recent observation. Setting $\gamma = 0$ means no seasonal component. The slope β is left free to be estimated. The residual ACF is now consistent with white noise.

Step 3: Write out the fitted model.

Using the Holt-Winters updating equations (3.21) with the estimated parameter values:

$$\begin{aligned} x_t &= x_{t-1} + b_{t-1} + w_t \\ b_{t-1} &= 0.167(x_{t-1} - x_{t-2}) + 0.833\, b_{t-2} \end{aligned} \tag{4.13}$$

Reading these two equations:

- The first says today's rate equals yesterday's rate plus yesterday's estimated drift plus a shock.
- The second says the drift estimate is updated by blending 16.7% of the most recent observed change with 83.3% of the previous drift estimate. The drift is *not* a fixed constant; it is a slowly evolving, exponentially smoothed estimate of local trend.

Step 4: Rewrite using the backward shift operator.

After algebra (Exercise 8 in the book), equations (4.13) collapse into a single equation:

$$(1 - 0.167B + 0.167B^2)(1 - B)\, x_t = w_t \tag{4.14}$$

This is a big conceptual moment. The Holt-Winters smoothing model has been rewritten as an ARIMA-type model: a differenced series driven by an autoregressive polynomial. This shows that smoothing methods and stochastic time series models are not separate worlds — they can often be expressed in equivalent forms.

4.4 Autoregressive Models

4.4.1 Definition

The series $\{x_t\}$ is an **autoregressive process of order** p, written AR(p), if

$$x_t = \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \cdots + \alpha_p x_{t-p} + w_t \tag{4.15}$$

where $\{w_t\}$ is white noise and $\alpha_p \neq 0$.
In backward-shift notation:

$$\theta_p(B)\, x_t = (1 - \alpha_1 B - \alpha_2 B^2 - \cdots - \alpha_p B^p)\, x_t = w_t \tag{4.16}$$

Special cases worth noting:

- AR(1) with $\alpha_1 = 1$ is the random walk.
- AR(∞) with $\alpha_i = \alpha(1-\alpha)^i$ is exponential smoothing.
- The AR model is literally a regression of x_t on its own past values — hence *auto*regressive.

4.4.2 Stationarity condition

The AR(1) model $x_t = \alpha x_{t-1} + w_t$ can be written as a weighted sum of all past shocks (via back substitution):

$$x_t = \sum_{i=0}^{\infty} \alpha^i w_{t-i}$$

This infinite sum converges if and only if $|\alpha| < 1$, which is the stationarity condition.

For a general AR(p) model, stationarity requires that all roots of the characteristic equation $\theta_p(B) = 0$ lie *outside* the unit circle.

Stationarity condition for AR(p):
Let $r_1, r_2, \ldots, r_p$ be the roots of $1 - \alpha_1 B - \cdots - \alpha_p B^p = 0$. The process is stationary if and only if $|r_j| > 1$ for all j.
In R: `polyroot(c(1, -alpha1, -alpha2, ...))` finds these roots.

Four worked examples

Example 1: AR(1), stationary

$x_t = \frac{1}{2}x_{t-1} + w_t$. The characteristic equation is $1 - \frac{1}{2}B = 0$, giving root $B = 2$. Since $|2| > 1$, the process is **stationary**.

Example 2: AR(2), stationary

$x_t = x_{t-1} - \frac{1}{4}x_{t-2} + w_t$. The characteristic polynomial is $\frac{1}{4}(B-2)^2$, giving a repeated root $B = 2$. Since $|2| > 1$, **stationary**.

Example 3: AR(2), nonstationary — unit root

$x_t = \frac{1}{2}x_{t-1} + \frac{1}{2}x_{t-2} + w_t$. Factoring: $-\frac{1}{2}(B-1)(B+2) = 0$, so roots are $B = 1$ and $B = -2$. One root is exactly 1 (a **unit root**), so the process is **nonstationary**.

Example 4: AR(2), stationary — complex roots

$x_t = -\frac{1}{4}x_{t-2} + w_t$. The equation $1 + \frac{1}{4}B^2 = 0$ gives $B = \pm 2i$. These are complex, with $|\pm 2i| = 2 > 1$. **Stationary**.

The three regimes, clearly:

$\lvert\alpha\rvert$ (AR(1))	Behavior	Variance	Stationarity
$\lvert\alpha\rvert < 1$	Shocks die out	Finite, constant	Stationary
$\lvert\alpha\rvert = 1$	Shocks persist forever	Grows as $t\sigma^2$	Nonstationary (random walk)
$\lvert\alpha\rvert > 1$	Shocks explode	Grows exponentially	Nonstationary, unstable

4.4.3 Second-order properties of AR(1)

For the AR(1) process $x_t = \alpha x_{t-1} + w_t$ with $|\alpha| < 1$:

$$\mu_x = 0, \qquad \gamma_k = \frac{\alpha^k \sigma^2}{1 - \alpha^2} \tag{4.19}$$

The autocorrelation function is:

$$\rho_k = \alpha^k \qquad (k \geq 0) \tag{4.21}$$

Derivation

Write the AR(1) as an infinite MA using back substitution:

$$(1 - \alpha B)x_t = w_t \implies x_t = \sum_{i=0}^{\infty} \alpha^i w_{t-i} \tag{4.20}$$

Mean: $E(x_t) = \sum_{i=0}^{\infty} \alpha^i E(w_{t-i}) = 0$.

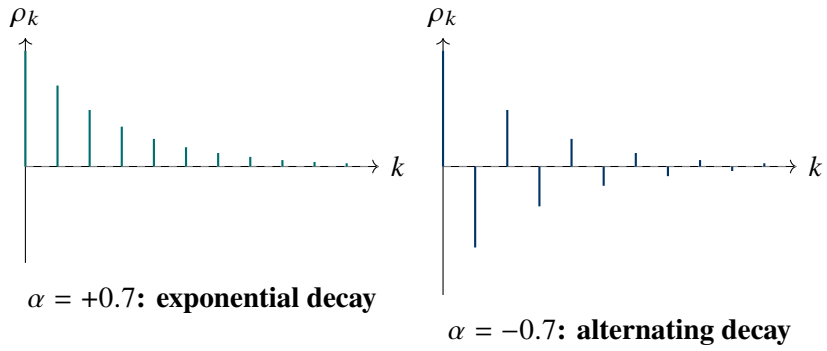
Autocovariance: apply the covariance-of-sums result (equation 2.15):

$$\gamma_k = \text{Cov}\left(\sum_{i=0}^{\infty} \alpha^i w_{t-i},\ \sum_{j=0}^{\infty} \alpha^j w_{t+k-j} \right) = \alpha^k \sigma^2 \sum_{i=0}^{\infty} \alpha^{2i} = \frac{\alpha^k \sigma^2}{1 - \alpha^2}.$$

Only the terms where subscripts match ($j = k + i$) contribute, because white noise terms at different times are uncorrelated.

Shape of the ACF for AR(1)

Since $\rho_k = \alpha^k$:



$\alpha = +0.7$**: exponential decay** $\alpha = -0.7$**: alternating decay**

Positive α: ACF decays exponentially and stays positive — the series is positively persistent. Negative α: ACF alternates in sign and decays — the series oscillates.

4.4.4 Partial autocorrelation (PACF)

The ACF of an AR(1) process is nonzero at all lags (because $\alpha^k \neq 0$ for any finite k), even though x_t only directly depends on x_{t-1}. The correlation at lag 2 is nonzero simply because x_t depends on x_{t-1} which depends on x_{t-2} — not because there is a direct link from x_{t-2} to x_t.

The partial autocorrelation removes these indirect effects.

> The **partial autocorrelation at lag** k (PACF) is the correlation between x_t and x_{t-k} after removing the linear effects of $x_{t-1}, x_{t-2}, \ldots, x_{t-k+1}$.
> Equivalently, it is the coefficient α_k in a fitted AR(k) model. If the true process is AR(p), then the PACF is zero for all lags $k > p$.

This gives us a powerful identification tool:

Process	ACF	PACF
AR(p)	Decays gradually (exponential or oscillating)	Cuts off after lag p
MA(q)	Cuts off after lag q	Decays gradually
White noise	All bars near zero	All bars near zero
Random walk	Slow decay from 1	Large spike at lag 1 only

In R, `acf()` produces the ACF and `pacf()` produces the PACF. Note that `pacf()` starts at lag 1, not lag 0 (there is no partial autocorrelation at lag 0).

```
# Simulate AR(1) with alpha = 0.7
set.seed(1)
x <- w <- rnorm(100)
for (t in 2:100) x[t] <- 0.7 * x[t-1] + w[t]

plot(x, type = "l")
acf(x)     # decays exponentially
pacf(x)    # only lag 1 is significant
```

4.5 Fitting AR Models

4.5.1 Estimation and AIC

```
x.ar <- ar(x, method = "mle")
x.ar$order          # selected order
x.ar$ar             # coefficient estimates
x.ar$asy.var        # asymptotic variance of estimates

# Approximate 95% confidence interval
x.ar$ar + c(-2, 2) * sqrt(x.ar$asy.var)
```

The `ar()` function fits AR models of order $0, 1, 2, \ldots$ and selects the best using the **Akaike Information Criterion (AIC)**:

$$\text{AIC} = -2 \times \log\text{-likelihood} + 2 \times (\text{number of parameters}) \tag{4.22}$$

- The first term rewards fit: higher likelihood means smaller $-2 \log L$.
- The second term penalizes complexity: more parameters cost more.

The model with the **lowest AIC** is selected.

AIC is not interpreted in isolation — only comparatively. If AIC(AR(1)) = 120, AIC(AR(2)) = 115, AIC(AR(3)) = 117, then AR(2) wins.

The problem that AIC solves is overfitting. Adding more lags always improves the in-sample fit, but eventually we are just fitting noise. AIC penalizes extra parameters to prevent this.

4.5.2 Residuals from a fitted AR(p) model

For a fitted AR(p) model, the first p residuals are undefined because there are no observations before $t = 1$ to form the prediction. So we always strip the first p values before inspecting the residual ACF:

```
# AR(1): remove first residual
acf(x.ar$res[-1])

# AR(p) in general: remove first p residuals
acf(x.ar$res[-(1:x.ar$order)])
```

4.5.3 Example: exchange rate series

```
Z.ar <- ar(Z.ts)
mean(Z.ts)                                    # 2.823
Z.ar$order                                    # 1
Z.ar$ar                                       # 0.8903
Z.ar$ar + c(-2, 2) * sqrt(Z.ar$asy.var)       # [0.74, 1.04]
```

The fitted AR(1) coefficient is $\hat{\alpha} = 0.89$. The 95% confidence interval $[0.74, 1.04]$ includes 1, meaning we cannot reject the hypothesis $\alpha = 1$ (the random walk). The prediction equation (subtracting the estimated mean):

$$\hat{z}_t = 2.8 + 0.89(z_{t-1} - 2.8) \tag{4.23}$$

This illustrates an important point: fitting a model does not settle the interpretation. The AR(1) fit is consistent with a random walk. We must reason about the coefficient's meaning and what values remain plausible.

4.5.4 Example: global temperature

```
Global.ar <- ar(aggregate(Global.ts, FUN = mean), method = "mle")
Global.ar$order    # 4
Global.ar$ar       # 0.588  0.013  0.111  0.268
acf(Global.ar$res[-(1:Global.ar$order)], lag = 50)
```

The fitted model is AR(4):

$$\hat{x}_t = -0.14 + 0.59(x_{t-1} + 0.14) + 0.013(x_{t-2} + 0.14) + 0.11(x_{t-3} + 0.14) + 0.27(x_{t-4} + 0.14) \quad (4.24)$$

The residual ACF is consistent with white noise. Since the AR(4) model has no deterministic trend component, the apparent upward trend arises entirely from serial correlation and accumulated shocks. This does not rule out a physical cause, it simply means the data alone cannot distinguish between a stochastic and a deterministic explanation.

4.6 R Commands Summary

Command	What it does
`set.seed(n)`	Fix random seed for reproducibility
`rnorm(n)`	Simulate n Gaussian white noise values
`cumsum(w)`	Cumulative sum (simulates random walk)
`diff(x)`	First differences: $x_t - x_{t-1}$
`diff(x, lag = s)`	Seasonal differences: $x_t - x_{t-s}$
`acf(x)`	ACF plot
`pacf(x)`	PACF plot (starts at lag 1)
`ar(x, method = "mle")`	Fit best AR(p) by AIC
`$order`	Selected order p
`$ar`	Estimated coefficients
`$asy.var`	Asymptotic variance of estimates
`$res`	Residuals (first p are NA)
`polyroot(c(1,-a1,-a2,...))`	Roots of AR characteristic polynomial
`resid(fit)`	Extract residuals from any fitted model

Chapter 5: Regression

Trends and seasonal patterns can be modelled deterministically using regression. The complication is that time series residuals are almost never independent, so standard OLS standard errors are wrong and need to be corrected.

5.1 Linear Models

A **linear model** for a time series $\{x_t\}$ is:

$$x_t = \alpha_0 + \alpha_1 u_{1,t} + \alpha_2 u_{2,t} + \cdots + \alpha_m u_{m,t} + z_t \tag{5.1}$$

where $u_{i,t}$ are predictor variables (time, seasonal dummies, harmonics, etc.), α_i are unknown parameters estimated by least squares, and $\{z_t\}$ is the residual series with mean zero — which does *not* need to be white noise.

The simplest case is the straight-line trend ($p = 1$ in the polynomial):

$$x_t = \alpha_0 + \alpha_1 t + z_t.$$

Many nonlinear models can be linearized by transformation. Taking logs of a multiplicative model:

$$x_t = m'_t \cdot s'_t \cdot z'_t \implies \log x_t = \underbrace{\log m'_t}_{m_t} + \underbrace{\log s'_t}_{s_t} + \underbrace{\log z'_t}_{z_t} \tag{5.16}$$

converts it to an additive linear form that OLS can handle directly.

Simulating a series with trend and autocorrelated errors

```
set.seed(1)
```

```
z <- w <- rnorm(100, sd = 20)
for (t in 2:100) z[t] <- 0.8 * z[t-1] + w[t]   # AR(1) errors
Time <- 1:100
x <- 50 + 3 * Time + z
plot(x, type = "l")
```

The model is $x_t = 50 + 3t + z_t$ where $\{z_t\}$ is AR(1) with $\alpha = 0.8$. This is the standard setup: a deterministic trend plus serially correlated errors.

5.2 Fitting Linear Models (OLS)

```
x.lm <- lm(x ~ Time)
coef(x.lm)              # estimated alpha_0, alpha_1
sqrt(diag(vcov(x.lm))) # standard errors of coefficients
acf(resid(x.lm))        # ALWAYS check the residual ACF
pacf(resid(x.lm))
```

In this example, OLS gives estimates close to the true values (intercept ≈ 58.6, slope ≈ 3.06). But the residual ACF shows strong positive autocorrelation, which means the standard errors reported by OLS are *underestimates*. The t-statistics and p-values are therefore misleading.

Why autocorrelation inflates significance:
For a stationary series with autocorrelation function ρ_k, the true variance of the sample mean is:

$$\text{Var}(\bar{x}) = \frac{\sigma^2}{n}\left[1 + 2\sum_{k=1}^{n-1}\left(1 - \frac{k}{n}\right)\rho_k\right] \tag{5.5}$$

If $\rho_k > 0$, then $\text{Var}(\bar{x}) > \sigma^2/n$. Positive autocorrelation makes consecutive observations redundant — the effective sample size is smaller than n. OLS ignores this and produces standard errors that are too small, making the trend appear more significant than it really is.

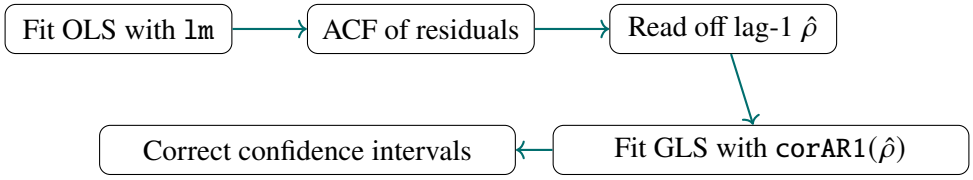
5.3 Generalised Least Squares (GLS)

GLS fixes the standard error problem by explicitly modelling the autocorrelation structure of the residuals.

```
library(nlme)
```

```
x.gls <- gls(x ~ Time, cor = corAR1(0.8))
coef(x.gls)
sqrt(diag(vcov(x.gls)))   # standard errors are now larger (and correct)
```

The workflow for GLS on real data:



Example: global temperature trend (1970–2005)

```
temp.lm  <- lm(temp ~ time(temp))
acf(resid(temp.lm))           # strong positive autocorrelation at short lags

# GLS with lag-1 autocorrelation ~0.7 read from the ACF plot
temp.gls <- gls(temp ~ time(temp), cor = corAR1(0.7))
confint(temp.gls)
```

Method	95% CI for slope	Conclusion
OLS (lm)	(0.0165, 0.0188)	Significant (but CI too narrow)
GLS (gls)	(0.0144, 0.0201)	Significant (wider, more honest)

Both exclude zero, so there is genuine statistical evidence of a warming trend. But the GLS interval is wider - it is the one to trust.

5.4 Linear Models with Seasonal Variables

5.4.1 Seasonal indicator (dummy) variables

For a series with s seasons, we assign a separate intercept to each season. The model is:

$$x_t = m_t + s_t + z_t \tag{5.6}$$

where $s_t = \beta_i$ when t falls in season i, and m_t is the trend. This gives each season its own baseline level on top of a common trend.

In R, the season index for each observation is extracted with `cycle()`, and passed to `factor()` so R creates the dummy variables automatically:

```
Seas <- cycle(temp)
Time <- time(temp)
temp.lm <- lm(temp ~ 0 + Time + factor(Seas))   # 0 suppresses the intercept
coef(temp.lm)
```

The `0 +` in the formula removes the overall intercept, allowing all $s = 12$ seasonal coefficients to be estimated freely. If you include an intercept, R drops one seasonal dummy to avoid collinearity and uses it as the reference level.

5.4.2 Harmonic (sine/cosine) seasonal models

Dummy variables use one parameter per season. Harmonic models use smooth sine and cosine waves instead, which can describe seasonal variation with fewer parameters.

A sine wave with frequency f, amplitude A, and phase ϕ can be linearized:

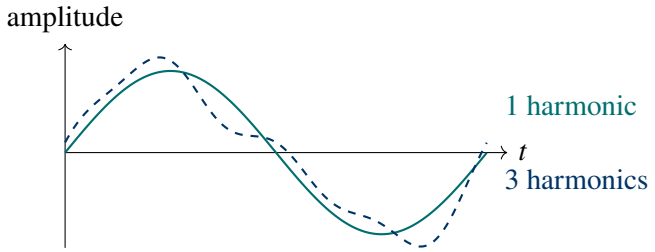
$$A\sin(2\pi ft + \phi) = \alpha_s \sin(2\pi ft) + \alpha_c \cos(2\pi ft) \tag{5.9}$$

The right-hand side is linear in α_s and α_c, so OLS applies directly.

The full harmonic model for a series with s seasons is:

$$x_t = m_t + \sum_{i=1}^{[s/2]} \left\{ s_i \sin\left(\frac{2\pi it}{s}\right) + c_i \cos\left(\frac{2\pi it}{s}\right) \right\} + z_t \tag{5.10}$$

Each term at frequency i/s adds one "harmonic" that perturbs the underlying seasonal pattern. Using fewer harmonics than the maximum gives a smoother, more parsimonious model.



Adding harmonics produces a less regular but more realistic seasonal shape.

Fitting a harmonic model in R

```
SIN <- COS <- matrix(nr = length(TIME), nc = 6)
for (i in 1:6) {
    SIN[, i] <- sin(2 * pi * i * TIME / 12)
    COS[, i] <- cos(2 * pi * i * TIME / 12)
}

# Fit all harmonics plus quadratic trend
x.lm1 <- lm(x ~ TIME + I(TIME^2) + SIN[,1] + COS[,1] + SIN[,2] +
             COS[,2] + SIN[,3] + COS[,3] + SIN[,4] + COS[,4] +
             SIN[,5] + COS[,5] + SIN[,6] + COS[,6])

# Check t-ratios to drop insignificant terms
coef(x.lm1) / sqrt(diag(vcov(x.lm1)))

# Refit with only significant terms, compare AIC
x.lm2 <- lm(x ~ I(TIME^2) + SIN[,1] + SIN[,2])
AIC(x.lm1); AIC(x.lm2)        # lower AIC wins

# Automate selection
step(x.lm1)
```

The selection rule: keep a term if its $|t\text{-ratio}| \gtrsim 2$ (approximately 5% level). Then use AIC to confirm the final model.

5.5 Logarithmic Transformations

Take logs when:

- The seasonal amplitude grows proportionally with the trend (multiplicative series).
- The series is strictly positive and variance increases over time.

```
plot(AP)        # variance grows with trend -- needs log
plot(log(AP))   # variance now approximately constant
```

After fitting a log-regression model and forecasting:

```
AP.lm2 <- lm(log(AP) ~ TIME + I(TIME^2) + SIN[,1] + COS[,1] + ...)
AP.pred <- exp(predict(AP.lm2, newdata = new.dat))   # back-transform
```

Bias correction

Taking the anti-log of a prediction for $\log x_t$ gives the median of x_t, not the mean. For Gaussian white noise residuals with standard deviation σ, the corrected forecast of the mean is:

$$\hat{x}_t' = e^{\widehat{\log x_t}} \cdot e^{\frac{1}{2}\sigma^2}$$

More generally, use the empirical correction factor:

$$\hat{x}_t' = e^{\widehat{\log x_t}} \cdot \frac{1}{n}\sum_{t=1}^{n} e^{z_t} \tag{5.19}$$

where $\{z_t\}$ are the residuals from the log-regression model.

If R^2 is high and σ is small, the correction factor is close to 1 and often negligible in practice (for the airline data it is only 0.1%). But there is no reason to assume it will always be small, so always check.

```
sigma <- summary(AP.lm2)$sigma
exp(sigma^2 / 2)                  # log-normal correction
mean(exp(resid(AP.lm2)))          # empirical correction (more robust)
```

5.6 Forecasting from Regression Models

A regression forecast is just the model evaluated at future time points. The key assumption: the fitted relationship continues to hold.

```
# Create future time values as a data.frame
new.t <- seq(2006, len = 2*12, by = 1/12)
new.dat <- data.frame(Time = new.t, Seas = rep(1:12, 2))

# Forecast
pred <- predict(temp.lm, newdata = new.dat)

# Plot observed + forecast
plot(temp, type = "l")
lines(new.t, pred, lty = 2)
```

For a log model, back-transform the forecasts:

```
AP.pred.ts <- exp(ts(predict(AP.lm2, new.dat), st = 1961, fr = 12))
ts.plot(AP, AP.pred.ts, lty = 1:2)
```

5.7 Full Modeling Workflow for Chapter 5

Step	Action	What to look for
1	Plot the series	Does variance grow? (suggests log transform) Is there a trend? Seasonality?
2	Log-transform if needed	Stabilizes variance, linearizes multiplicative structure
3	Fit OLS (`lm`)	Start with trend + seasonal terms
4	Check residual ACF/PACF	Are residuals autocorrelated? What order AR?
5	Use AIC / t-ratios	Drop insignificant terms, compare models
6	Refit with GLS (`gls`)	Get honest standard errors for inference
7	Fit AR to residuals	`ar(resid(fit))` to capture remaining autocorrelation
8	Check final residuals	Should look like white noise
9	Forecast with `predict`	Back-transform and apply bias correction if log was used

Applied to UKgas

The UKgas series has multiplicative seasonality (amplitude grows with trend), so we work on log(UKgas). The residuals from a log-linear regression with seasonal dummies show strong autocorrelation (Figure 5.1), confirming that OLS alone is not sufficient.

```
lx <- log(UKgas)
Time <- time(UKgas)
Seas <- cycle(UKgas)
lx.lm <- lm(lx ~ 0 + Time + factor(Seas))
acf(resid(lx.lm))          # Figure: acf_resid_lm
plot(resid(lx.lm), type="l")  # Figure: resid_plot
```

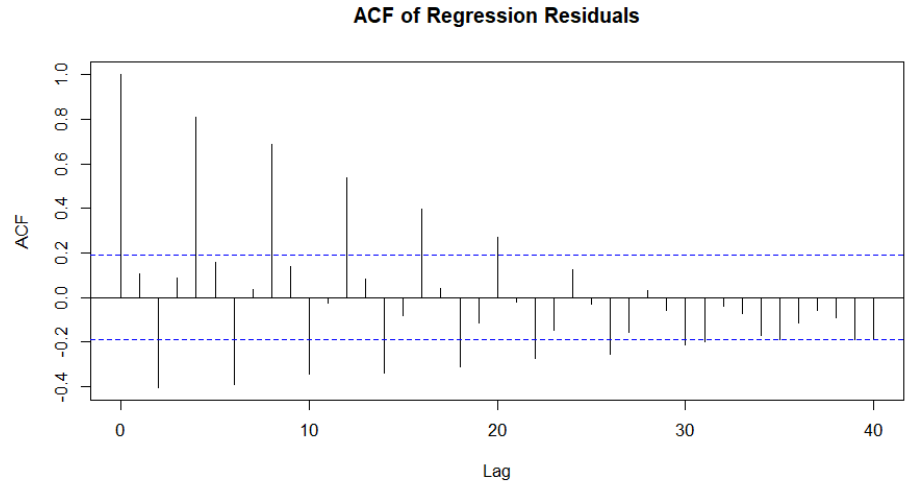


Figure 5.1: ACF of residuals from the log-linear regression with seasonal dummies fitted to UKgas. The strong autocorrelation at low lags shows that a regression model alone does not capture all the structure — an AR model for the residuals is needed.

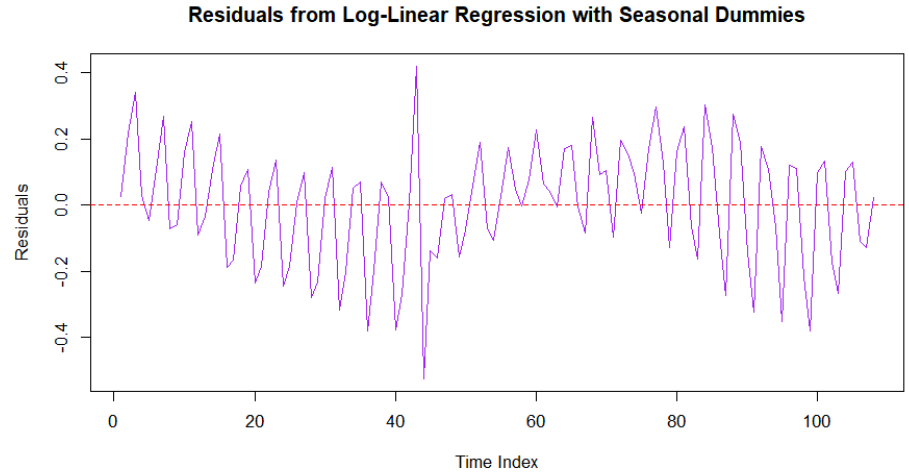


Figure 5.2: Residuals from the log-linear regression with seasonal dummies. The oscillating pattern and persistence above/below zero confirm positive serial correlation in the residuals.

5.8 R Commands Summary

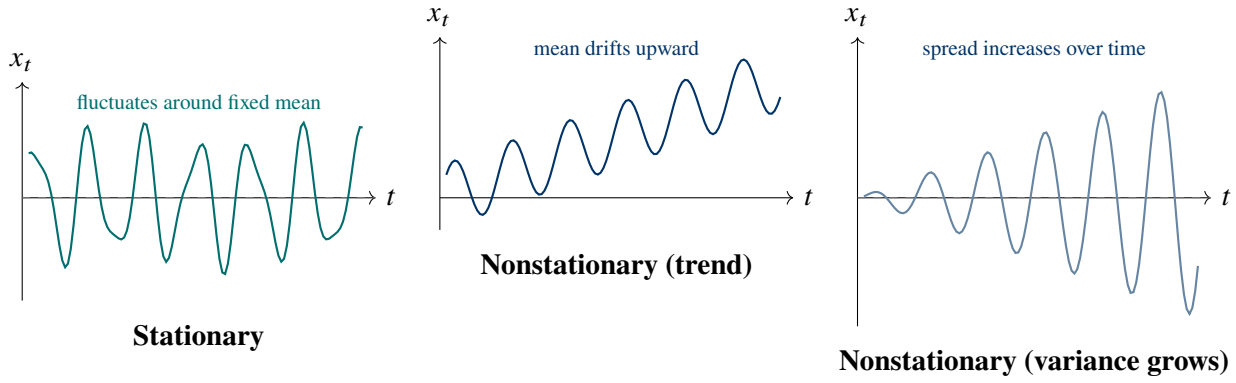
Command	What it does
`lm(x ~ Time)`	Fit OLS linear regression
`coef(fit)`	Extract coefficient estimates
`confint(fit)`	95% confidence intervals for parameters
`sqrt(diag(vcov(fit)))`	Standard errors of estimates
`resid(fit)`	Extract residuals
`acf(resid(fit))`	Residual ACF — always run this
`AIC(fit)`	AIC for model comparison
`step(fit)`	Stepwise AIC-based variable selection
`cycle(x)`	Extract season index from ts object
`factor(cycle(x))`	Create seasonal dummy variables
`library(nlme)`	Load GLS package
`gls(x ~ Time, cor=corAR1(`ρ`))`	GLS with AR(1) errors
`predict(fit, newdata=df)`	Forecast at new time points
`exp(predict(...))`	Back-transform log forecasts
`mean(exp(resid(fit)))`	Empirical bias correction factor

Chapter 6: Stationary Models

After removing trend and seasonality through regression, the residual series still tends to be serially correlated. We study the models used to capture that remaining dependence: MA, AR, and ARMA processes. These are all stationary models, so the first step is always making sure the series we are working with actually is stationary.

6.1 Stationarity: A Quick Check

Formally, a series $\{x_t\}$ is **strictly stationary** if the joint distribution of any collection of observations is unchanged by an arbitrary time shift. In practice we work with the weaker **second-order stationarity**: constant mean, constant variance, and autocovariance depending only on lag k.



Before fitting any model, ask:

- Is the mean stable? (no trend)
- Is the variance stable? (no growing spread - log-transform if needed)
- Have seasonal effects been removed?

6.2 Moving Average Models

6.2.1 Definition and properties

An MA(q) process is a weighted sum of the current and q most recent white noise shocks:

$$x_t = w_t + \beta_1 w_{t-1} + \cdots + \beta_q w_{t-q} \tag{6.1}$$

In backward-shift notation: $x_t = \phi_q(B)\, w_t$, where $\phi_q(B) = 1 + \beta_1 B + \cdots + \beta_q B^q$.
Mean: $E(x_t) = 0$ (sum of zero-mean terms)
Variance: $\sigma_x^2 = \sigma_w^2(1 + \beta_1^2 + \cdots + \beta_q^2)$
ACF:

$$\rho(k) = \begin{cases} 1 & k = 0 \\ \dfrac{\sum_{i=0}^{q-k} \beta_i \beta_{i+k}}{\sum_{i=0}^{q} \beta_i^2} & k = 1, \ldots, q \\ 0 & k > q \end{cases} \tag{6.3}$$

where $\beta_0 = 1$.

The ACF is *exactly zero* after lag q. This is the defining signature of an MA model, and the cleanest identification tool we have.

Invertibility

An MA(q) model is **invertible** if it can be written as an AR(∞) process. For MA(1):

$$x_t = (1 - \beta B) w_t \implies w_t = x_t + \beta x_{t-1} + \beta^2 x_{t-2} + \cdots$$

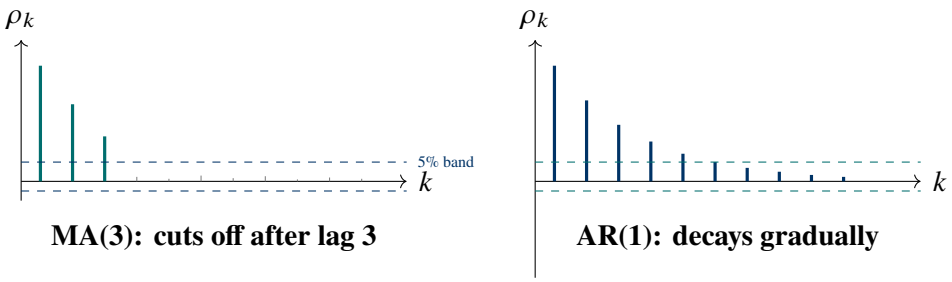
This converges only if $|\beta| < 1$. In general, invertibility requires all roots of $\phi_q(B) = 0$ to lie outside the unit circle. The `arima` function in R automatically returns invertible solutions.

6.2.2 Simulation and the ACF cutoff

```
# Simulate MA(3) with beta = (0.8, 0.6, 0.4)
set.seed(1)
b <- c(0.8, 0.6, 0.4)
x <- w <- rnorm(1000)
for (t in 4:1000) {
```

```
    for (j in 1:3) x[t] <- x[t] + b[j] * w[t - j]
}
plot(x, type = "l")
acf(x)              # should cut off after lag 3
```

The ACF for an MA(q) process cuts off sharply after lag q:



MA(3): cuts off after lag 3 **AR(1): decays gradually**

6.2.3 Fitting an MA model

```
# MA(3) fitted to simulated data
x.ma <- arima(x, order = c(0, 0, 3))
x.ma                    # coefficients, s.e., AIC
acf(x.ma$residuals)     # check for remaining structure
```

The `order = c(p, d, q)` argument: p = AR order, d = differencing, q = MA order.

Model	`order`	AR lags	MA lags
MA(1)	`c(0,0,1)`	none	1
MA(3)	`c(0,0,3)`	none	1, 2, 3
AR(2)	`c(2,0,0)`	1, 2	none
ARMA(1,1)	`c(1,0,1)`	1	1

Fitted output for the simulated MA(3):

```
Coefficients:
    ma1    ma2    ma3  intercept
  0.790  0.566  0.396     -0.032
s.e. 0.031  0.035  0.032      0.090
```

The estimates are close to the true values (0.8, 0.6, 0.4) and the 95% intervals ($\hat{\beta} \pm 2$ s.e.) contain them.

6.3 ARMA Models

6.3.1 Definition

An ARMA(p, q) process combines AR and MA terms:

$$x_t = \underbrace{\alpha_1 x_{t-1} + \cdots + \alpha_p x_{t-p}}_{\text{AR part}} + w_t + \underbrace{\beta_1 w_{t-1} + \cdots + \beta_q w_{t-q}}_{\text{MA part}} \tag{6.7}$$

In polynomial form: $\theta_p(B)\, x_t = \phi_q(B)\, w_t$

Stationarity: roots of $\theta_p(B) = 0$ all outside the unit circle.

Invertibility: roots of $\phi_q(B) = 0$ all outside the unit circle.

Why use ARMA instead of AR or MA alone?

Parameter parsimony. An ARMA(1,1) can often describe dependence that would require a high-order AR or MA alone. Fewer parameters means less estimation error and better forecasts.

Parameter redundancy warning. If θ_p and ϕ_q share a common factor, the model simplifies. For example:

$$(1 - \tfrac{1}{2}B)(1 - \tfrac{1}{3}B)x_t = (1 - \tfrac{1}{2}B)w_t \quad \Rightarrow \quad (1 - \tfrac{1}{3}B)x_t = w_t.$$

Always cancel common factors before fitting.

6.3.2 Second-order properties of ARMA(1,1)

The ARMA(1,1) process $x_t = \alpha x_{t-1} + w_t + \beta w_{t-1}$ can be written in terms of white noise by expanding $(1 - \alpha B)^{-1}(1 + \beta B)w_t$:

$$x_t = w_t + (\alpha + \beta) \sum_{i=1}^{\infty} \alpha^{i-1} w_{t-i} \tag{6.10}$$

From this the variance and autocovariance follow:

$$\mathrm{Var}(x_t) = \sigma_w^2 + \frac{\sigma_w^2(\alpha + \beta)^2}{1 - \alpha^2} \tag{6.11}$$

$$\rho_k = \frac{\alpha^{k-1}(\alpha+\beta)(1+\alpha\beta)}{1+\alpha\beta+\beta^2}, \qquad k \geq 1 \tag{6.13}$$

Note that $\rho_k = \alpha\,\rho_{k-1}$ for $k \geq 2$, so after the first lag the ACF decays at rate α — just like an AR(1).

6.3.3 Identification: ACF and PACF together

Process	ACF	PACF
White noise	All near zero	All near zero
AR(p)	Decays gradually	Cuts off after lag p
MA(q)	Cuts off after lag q	Decays gradually
ARMA(p,q)	Decays gradually	Decays gradually

When both ACF and PACF decay without a clean cutoff, ARMA is the right class of model. The question then is which orders p and q to use — answered by AIC.

6.3.4 Simulation and fitting

```
# Simulate ARMA(1,1) with alpha = -0.6, beta = 0.5
set.seed(1)
x <- arima.sim(n = 10000, list(ar = -0.6, ma = 0.5))

# Fit and recover parameters
coef(arima(x, order = c(1, 0, 1)))
#   ar1     ma1  intercept
# -0.597   0.503    -0.007
```

6.4 Model Selection and Diagnostics

6.4.1 Comparing models with AIC

Fit several candidate models and compare:

```
x.ar   <- arima(x, order = c(1, 0, 0))
x.ma   <- arima(x, order = c(0, 0, 1))
x.arma <- arima(x, order = c(1, 0, 1))
```

```
AIC(x.ar); AIC(x.ma); AIC(x.arma)
```

For the exchange rate series: AIC(MA(1)) = −3.5, AIC(AR(1)) = −37.4, AIC(ARMA(1,1)) = −42.3. ARMA(1,1) wins. The fitted model:

```
Coefficients:
   ar1    ma1  intercept
 0.892  0.532      2.960
s.e. 0.076  0.202      0.244
```

6.4.2 Grid search over ARMA orders

When the best order is unknown, search over a grid of (p, q) combinations:

```
best.order <- c(0, 0, 0)
best.aic   <- Inf
for (i in 0:2) for (j in 0:2) {
    fit.aic <- AIC(arima(x, order = c(i, 0, j)))
    if (fit.aic < best.aic) {
        best.order <- c(i, 0, j)
        best.arma  <- arima(x, order = best.order)
        best.aic   <- fit.aic
    }
}
best.order    # reports the winning (p, 0, q)
```

6.4.3 Residual diagnostics

After selecting a model, always inspect the residuals:

```
acf(resid(x.arma))     # should be white noise
pacf(resid(x.arma))
```

If significant structure remains, the model is incomplete. Try increasing p or q.

For the UKgas project, after fitting AR(1) and ARMA(1,1) to the stationary series, the residual ACFs are shown in Figures 6.1 and 6.2.

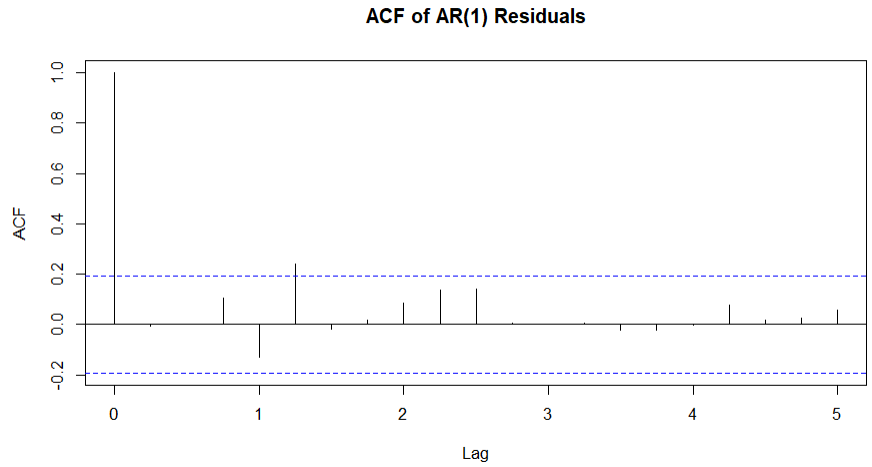


Figure 6.1: Residual ACF after fitting AR(1) to the stationary UKgas series. A few spikes near the significance boundary — marginally adequate.

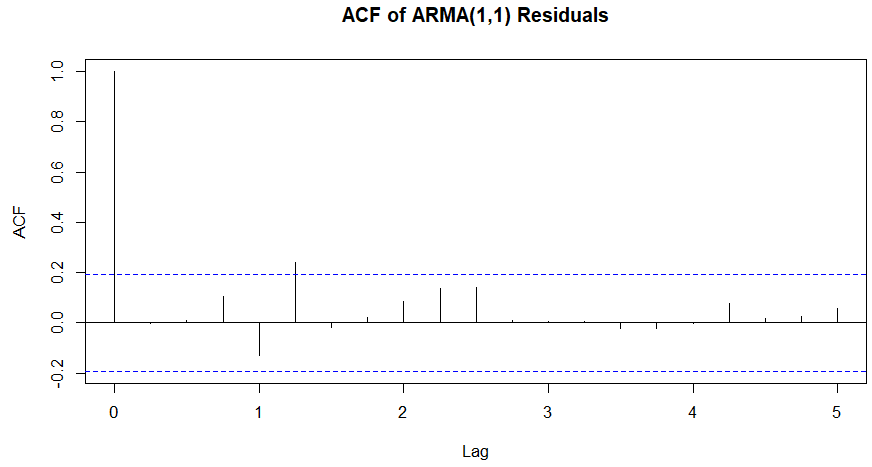


Figure 6.2: Residual ACF after fitting ARMA(1,1). The residuals are closer to white noise than the AR(1) fit, supporting ARMA(1,1) as the better model.

6.5 Forecasting from ARMA Models

```
pred <- predict(x.arma, n.ahead = 12)
pred$pred    # point forecasts
pred$se      # forecast standard errors
```

To combine a regression forecast with an ARMA forecast on the residuals:

```
# Regression forecast (trend + seasonal)
```

```
predict.lm <- predict(Elec.lm, new.data)

# ARMA forecast on residuals
predict.arma <- predict(best.arma, n.ahead = 36)

# Combined forecast (anti-log to return to original scale)
elec.pred <- ts(exp(predict.lm + predict.arma$pred),
                start = 1991, freq = 12)
ts.plot(cbind(Elec.ts, elec.pred), lty = 1:2)
```

The total forecast = regression component + ARMA component. This is the payoff of the full Chapter 5–6 workflow.

6.6 Full Chapter 6 Workflow

Step	Action	What to check
1	Confirm stationarity	No trend, stable variance, seasonality removed
2	Plot ACF and PACF	Cutoff → AR or MA; both decay → ARMA
3	Fit candidate models	`arima(x, order = c(p,0,q))`
4	Compare AIC	Lower is better; differences < 2 are not meaningful
5	Check residual ACF	Must look like white noise
6	Forecast	`predict(fit, n.ahead = k)`

6.7 R Commands Summary

Command	What it does
`arima.sim(n, list(ar=..., ma=...))`	Simulate ARMA data
`arima(x, order = c(p,0,q))`	Fit ARMA(p,q) model
`coef(fit)`	Estimated AR and MA coefficients
`AIC(fit)`	AIC for model comparison
`resid(fit)`	Residuals
`acf(resid(fit))`	Residual ACF — always check
`pacf(resid(fit))`	Residual PACF
`predict(fit, n.ahead = k)`	Forecast k steps ahead
`  $pred`	Point forecasts
`  $se`	Forecast standard errors

Chapter 7: Non-stationary Models

These cover the most common forms of nonstationarity encountered in real data. ARIMA models (differencing + ARMA), seasonal ARIMA (SARIMA) which extends ARIMA to handle stochastic seasonality, and ARCH/GARCH models for series where the *variance* itself changes over time.

7.1 Non-seasonal ARIMA

7.1.1 From differencing to ARIMA

We already know that differencing can remove a stochastic trend. The key insight of ARIMA is that after differencing, the remaining series can be modelled as ARMA(p,q). This gives the full ARIMA(p,d,q) class.

A series $\{x_t\}$ follows an **ARIMA(p,d,q)** process if its dth difference is ARMA(p,q):

$$\theta_p(B)(1-B)^d x_t = \phi_q(B)\, w_t \tag{7.2}$$

where θ_p and ϕ_q are polynomials of orders p and q. The integer d is the order of integration. A series requiring d differences to reach stationarity is called **integrated of order** d, written $I(d)$.

Shorthand	ARIMA form	Equation	Description
IMA(1,1)	ARIMA(0,1,1)	$(1-B)x_t = (1+\beta B)w_t$	Random walk + MA shock
ARI(1,1)	ARIMA(1,1,0)	$(1-\alpha B)(1-B)x_t = w_t$	Differenced AR(1)
RW	ARIMA(0,1,0)	$(1-B)x_t = w_t$	Pure random walk

Differencing in R

```
diff(x)           # first difference:  x_t - x_{t-1}
diff(x, d = 2)    # second difference: apply diff twice
diff(x, lag = 12) # seasonal difference: x_t - x_{t-12}
```

For the UKgas series: the log transformation stabilizes variance, then seasonal differencing (lag 4) removes the quarterly pattern, then first differencing removes the remaining trend. The ACF and PACF of the resulting series guide model selection (Figures 7.3, 7.4, 7.6, **??**).

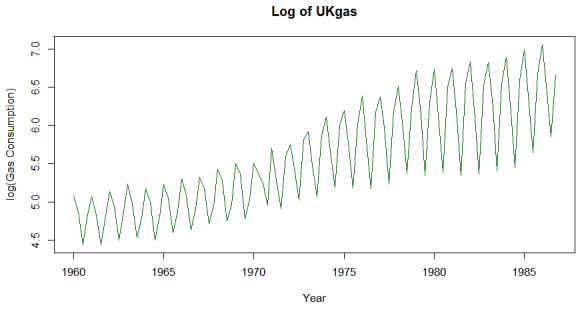


Figure 7.1: Log of UKgas. Trend remains; seasonal amplitude is now stable.

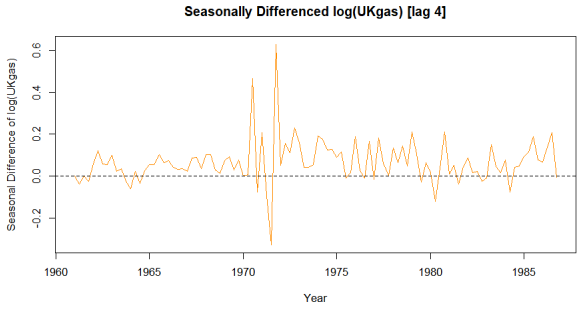


Figure 7.2: Seasonally differenced log(UKgas) at lag 4. Trend mostly removed.

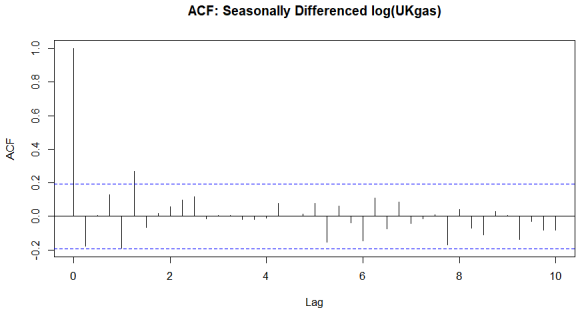


Figure 7.3: ACF of seasonally differenced log(UKgas). Significant spike at lag 1 suggests an MA(1) or AR term is still needed.

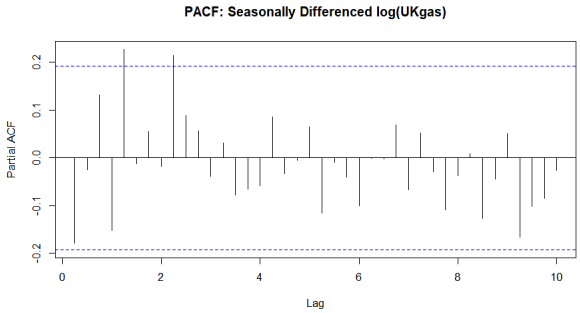


Figure 7.4: PACF of seasonally differenced log(UKgas).

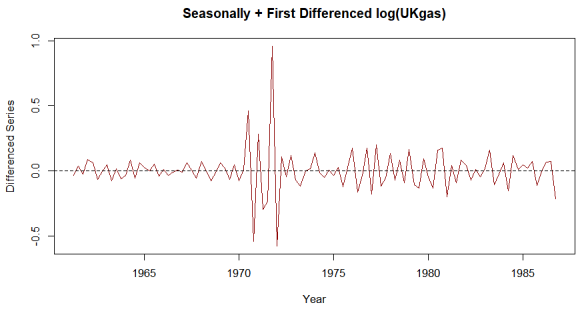


Figure 7.5: Seasonally + first differenced log(UKgas). Series now looks stationary.

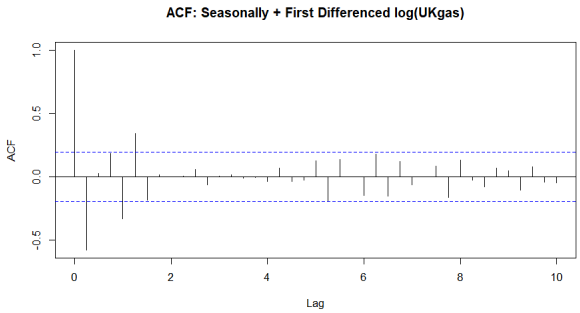


Figure 7.6: ACF after both differences. Most bars inside bands.

7.1.2 Simulation and fitting

```
# Simulate ARIMA(1,1,1)
x <- arima.sim(model = list(order = c(1,1,1), ar = 0.5, ma = 0.3),
               n = 1000)

# Fit
arima(x, order = c(1, 1, 1))
```

The `arima` function handles differencing internally — you pass the original series, not the pre-differenced one.

7.1.3 Example: IMA(1,1) fitted to beer production

The Australian beer series has a stochastic trend. Fitting ARIMA(0,1,1):

```
Beer.ima <- arima(Beer.ts, order = c(0, 1, 1))
# Coefficients: ma1 = -0.333

# Forecast next 12 months
Beer.1991 <- predict(Beer.ima, n.ahead = 12)
sum(Beer.1991$pred)   # total annual production forecast
```

Fitted model: $x_t = x_{t-1} + w_t - 0.33\, w_{t-1}$. The residual ACF shows spikes at yearly lags, signalling that seasonal structure remains — a seasonal ARIMA is needed.

7.2 Seasonal ARIMA (SARIMA)

7.2.1 Definition

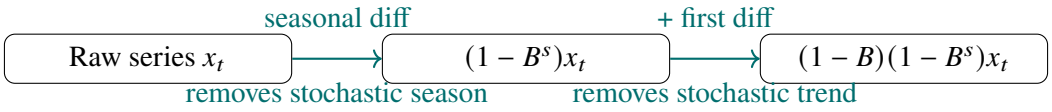
Seasonal ARIMA extends the ARIMA framework by adding AR, differencing, and MA terms at the seasonal lag s.

The **SARIMA**$(p, d, q)(P, D, Q)_s$ model is:

$$\Theta_P(B^s)\,\theta_p(B)\,(1-B^s)^D(1-B)^d\,x_t = \Phi_Q(B^s)\,\phi_q(B)\,w_t \tag{7.3}$$

Term	Meaning
p, d, q	Non-seasonal AR order, differencing, MA order
P, D, Q	Seasonal AR order, differencing, MA order
s	Seasonal period (12 = monthly, 4 = quarterly)
$(1-B)^d$	Non-seasonal differencing
$(1-B^s)^D$	Seasonal differencing

What each differencing operator removes



Three illustrative SARIMA examples

Example 1: Seasonal AR only — ARIMA(0,0,0)(1,0,0)$_{12}$

$$x_t = \alpha x_{t-12} + w_t$$

Current value depends only on the same month last year. Stationary when $|\alpha| < 1$.

Example 2: Stochastic trend + seasonal AR — ARIMA(0,1,0)(1,0,0)$_{12}$

$$(1-\alpha B^{12})(1-B)x_t = w_t$$

Equivalently: $\nabla x_t = \alpha \nabla x_{t-12} + w_t$. The change this period depends on the change in the same period last year. Nonstationary because of the $(1 - B)$ factor.

Example 3: Seasonal MA with trend — ARIMA(0,1,0)(0,0,1)$_4$

$$x_t = x_{t-1} + w_t - \beta w_{t-4}$$

For quarterly data: random walk plus a seasonal moving average shock.

7.2.2 What differencing choice does to a deterministic trend

An important subtlety: if the series is $x_t = a + bt + s_{[t]} + w_t$ (linear trend, additive seasonal, white noise), then:

- Seasonal differencing only $(1 - B^4)$: gives $4b + w_t - w_{t-4}$, an ARIMA(0,0,0)(0,1,1)$_4$ with a constant.
- Seasonal + first differencing $(1-B^4)(1-B)$: gives $w_t - w_{t-1} - w_{t-4} + w_{t-5}$, an ARIMA(0,1,1)(0,1,1)$_4$ with no constant. Over-differencing introduces extra MA terms.

This means differencing more than necessary is wasteful. Only difference as many times as needed to reach stationarity.

7.2.3 Fitting in R

```
# Fit SARIMA(1,1,0)(1,0,0)_12
arima(log(Elec.ts),
      order    = c(1, 1, 0),
      seasonal = list(order = c(1, 0, 0), period = 12))

# Compare two models by AIC
AIC(arima(log(Elec.ts), order = c(1,1,0),
          seas = list(order = c(1,0,0), 12)))   # -1765
AIC(arima(log(Elec.ts), order = c(0,1,1),
          seas = list(order = c(0,0,1), 12)))   # -1362
```

The first model wins. When the search space is large, use the automated grid search:

```
get.best.arima <- function(x.ts, maxord = c(1,1,1,1,1,1)) {
```

```
    best.aic <- 1e8
    n <- length(x.ts)
    for (p in 0:maxord[1]) for (d in 0:maxord[2]) for (q in 0:maxord[3])
    for (P in 0:maxord[4]) for (D in 0:maxord[5]) for (Q in 0:maxord[6]) {
        fit <- arima(x.ts, order = c(p,d,q),
                     seas = list(order = c(P,D,Q),
                     frequency(x.ts)), method = "CSS")
        fit.aic <- -2 * fit$loglik + (log(n) + 1) * length(fit$coef)
        if (fit.aic < best.aic) {
            best.aic   <- fit.aic
            best.fit   <- fit
            best.model <- c(p,d,q,P,D,Q)
        }
    }
    list(best.aic, best.fit, best.model)
}

best <- get.best.arima(log(Elec.ts), maxord = c(2,2,2,2,2,2))
best[[3]]    # winning order: 0 1 1 2 0 2
```

The function uses CAIC (consistent AIC) rather than standard AIC to penalise complexity more heavily, which helps avoid overfitting when p, d, q, P, D, Q are all free.

Forecasting from SARIMA

```
best.fit <- best[[2]]
acf(resid(best.fit))        # confirm white noise residuals

# Forecast 12 months ahead
pred <- predict(best.fit, n.ahead = 12)
ts.plot(cbind(window(Elec.ts, start = 1981),
              exp(pred$pred)), lty = 1:2)
```

Applied to UKgas

For the UKgas project, the AR(1) model on the stationary (doubly-differenced) series gives reasonable forecasts. Figures 7.7 shows the AR(1) forecast on the transformed scale.

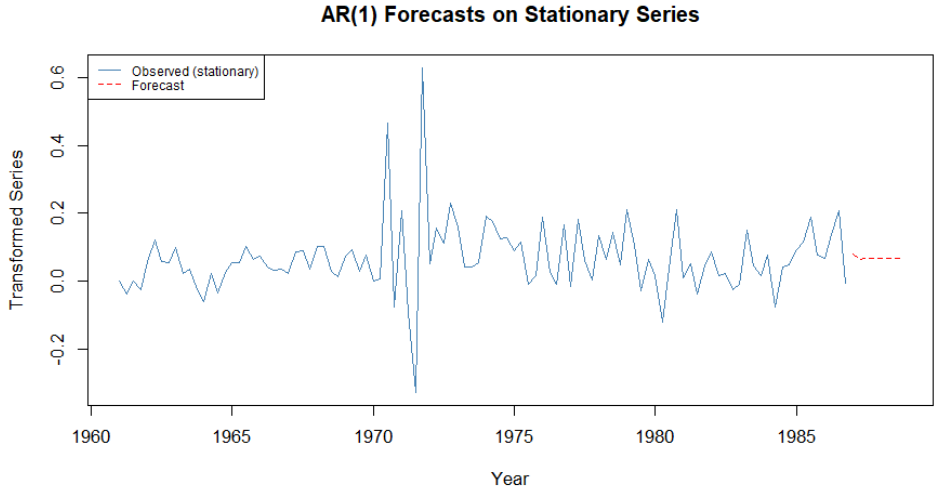


Figure 7.7: AR(1) forecasts on the stationary (seasonally + first differenced log) UKgas series. The forecast reverts toward the mean of the stationary series, as expected from a stationary AR model.

7.3 ARCH and GARCH Models

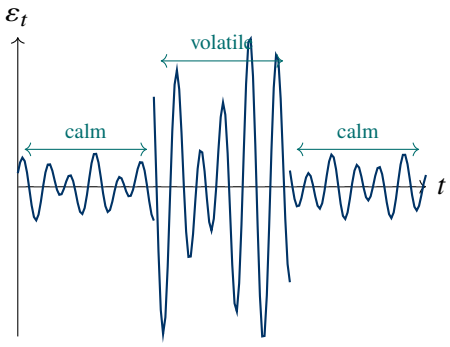
7.3.1 Volatility: a different kind of nonstationarity

All models so far assume the variance is constant. Some series — especially financial ones — have a constant mean but a time-varying variance: calm periods interrupted by bursts of volatility. This is called **conditional heteroskedasticity**.

How to detect volatility:

1. Fit an appropriate ARMA/ARIMA model to remove serial correlation.
2. Check ACF of residuals — should look like white noise.
3. Check ACF of *squared* residuals. If this shows significant autocorrelation, volatility is present.

A series can appear white noise in the ACF of levels but show strong autocorrelation in the ACF of squares. This is the signature of ARCH/GARCH behaviour.



Conditional heteroskedasticity

7.3.2 ARCH(1) definition

A series $\{\varepsilon_t\}$ is **ARCH(1)** if:

$$\varepsilon_t = w_t\sqrt{\alpha_0 + \alpha_1\varepsilon_{t-1}^2} \tag{7.4}$$

where $\{w_t\}$ is white noise with mean 0 and variance 1.
The conditional variance is:

$$\text{Var}(\varepsilon_t \mid \varepsilon_{t-1}) = \alpha_0 + \alpha_1\varepsilon_{t-1}^2 \tag{7.5}$$

This behaves exactly like an AR(1) model but for the *variance*. A large shock at $t-1$ increases the variance at t, producing volatility clustering.

The general ARCH(p) replaces the single lag with p lags:

$$\varepsilon_t = w_t\sqrt{\alpha_0 + \sum_{i=1}^{p}\alpha_i\varepsilon_{t-i}^2} \tag{7.6}$$

7.3.3 GARCH(q,p) model

ARCH requires many parameters to capture long memory in volatility. GARCH (Generalized ARCH) adds lagged variance terms, making it much more parsimonious.

A series $\{\varepsilon_t\}$ is **GARCH(q,p)** if $\varepsilon_t = w_t\sqrt{h_t}$ where:

$$h_t = \alpha_0 + \sum_{i=1}^{p}\alpha_i\varepsilon_{t-i}^2 + \sum_{j=1}^{q}\beta_j h_{t-j} \tag{7.7–7.8}$$

h_t is the conditional variance. It depends on past squared shocks (ARCH part) and past variances (GARCH part). GARCH(1,1) is almost always sufficient in practice.

Model	Conditional variance
ARCH(1)	$h_t = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2$
GARCH(1,1)	$h_t = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 h_{t-1}$
GARCH(0,p)	Reduces to ARCH(p)

Stability requires $\alpha_1 + \beta_1 < 1$ for GARCH(1,1).

7.3.4 Simulation and fitting

```
# Simulate GARCH(1,1)
set.seed(1)
alpha0 <- 0.1; alpha1 <- 0.4; beta1 <- 0.2
w <- rnorm(10000); a <- h <- rep(0, 10000)
for (i in 2:10000) {
    h[i] <- alpha0 + alpha1 * a[i-1]^2 + beta1 * h[i-1]
    a[i] <- w[i] * sqrt(h[i])
}

acf(a)     # looks like white noise
acf(a^2)   # strong autocorrelation -- volatility confirmed

# Fit GARCH(1,1)
library(tseries)
a.garch <- garch(a, grad = "numerical", trace = FALSE)
confint(a.garch)
```

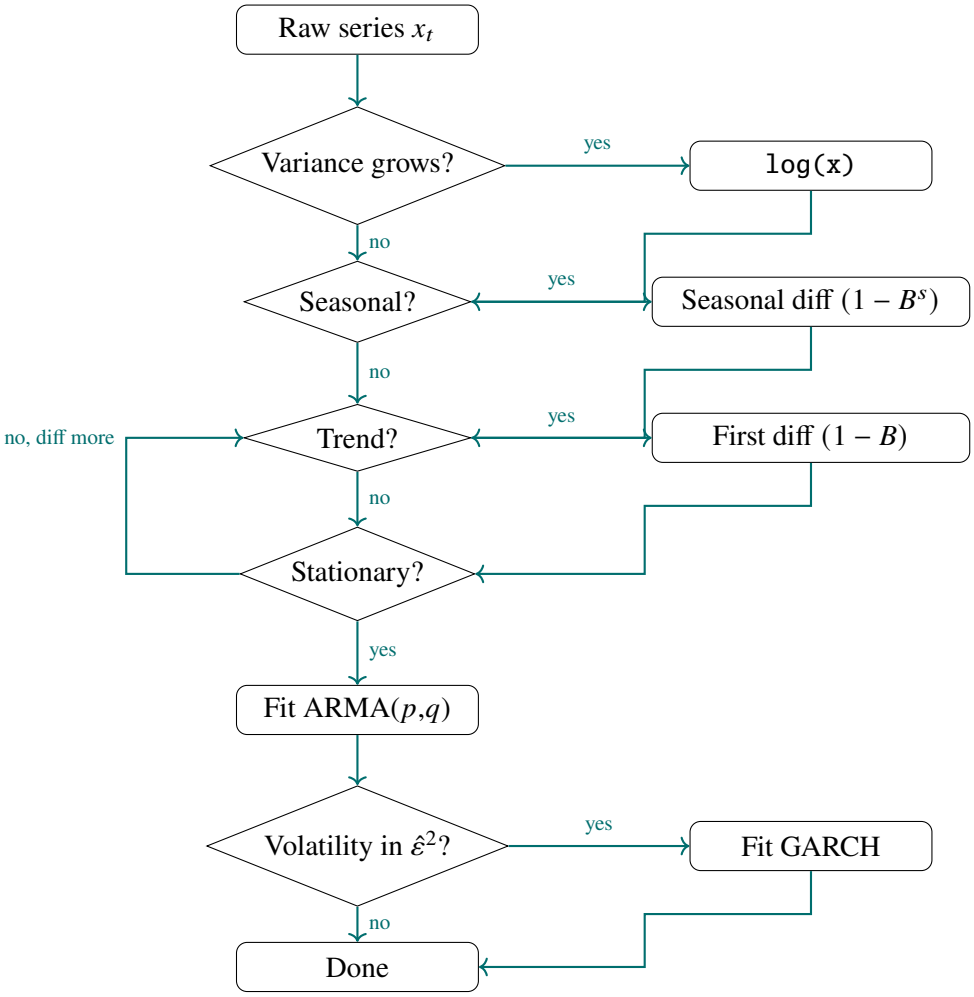
Checking GARCH residuals

Define standardised residuals $\hat{w}_t = \varepsilon_t / \sqrt{\hat{h}_t}$. If the GARCH model fits well, both $\hat{w}_t$ and $\hat{w}_t^2$ should look like white noise.

```
sp.garch <- garch(SP500, trace = FALSE)
```

```
sp.res   <- sp.garch$res[-1]   # remove first NA

acf(sp.res)     # should be white noise
acf(sp.res^2)   # should also be white noise if GARCH fit is adequate
```



7.4 R Commands Summary

Command	What it does
`diff(x)`	First difference
`diff(x, lag = s)`	Seasonal difference at lag s
`diff(x, d = 2)`	Second difference
`arima(x, order = c(p,d,q))`	Fit ARIMA(p,d,q)
`arima(x, order = c(p,d,q),` `  seasonal = list(order = c(P,D,Q), s))`	Fit SARIMA
`arima.sim(model = list(order=..., ar=..., ma=...), n)`	Simulate ARIMA
`predict(fit, n.ahead = k)`	Forecast k steps
`library(tseries)`	Load for GARCH
`garch(x, trace = FALSE)`	Fit GARCH(1,1)
`garch(x, order = c(q,p))`	Fit GARCH(q,p)
`confint(fit)`	Parameter confidence intervals
`resid(fit)`	Residuals
`acf(resid(fit)^2)`	Check for volatility in residuals

Acknowledgment

Some examples, simulations, and chapter structure in these notes are taken from Paul S.P. Cowpertwait & Andrew V. Metcalfe, *Introductory Time Series with R*, Springer, 2009.