Day 9 - White noise and random walks

Introduction

Last week, we introduced the Holt-Winters filter, a model that is capable of estimating a time series with trend and seasonal components. However, sometimes the Holt-Winters model is not able to capture the serial correlation present in the data, resulting in left-over serial correlation in the *residual error series*. In order to capture serial correlation in the residual error series, we need to build more complicated models by combining time series techniques used for the mean and error structure of the series. This week, we begin considering basic stochastic models for time series, including: white noise, random walks, and autoregressive models.

To guide our discussion, we will use a data set describing the number of visits to a website for a statistics course at Duke University, called Statistical Forecasting. The data set contains daily counts of the number of visits to the website between 2014 and 2020.

```
# packages
library(tidyverse)
library(lubridate)

# read table from the web
web <- read_csv("daily-website-visitors.csv")
```

Pros:

- Filled in notes
- Office hours

Cons:

- No printed notes :(
- Not finishing notes
- Balancing class, mid-terms, job search, etc

Changes:

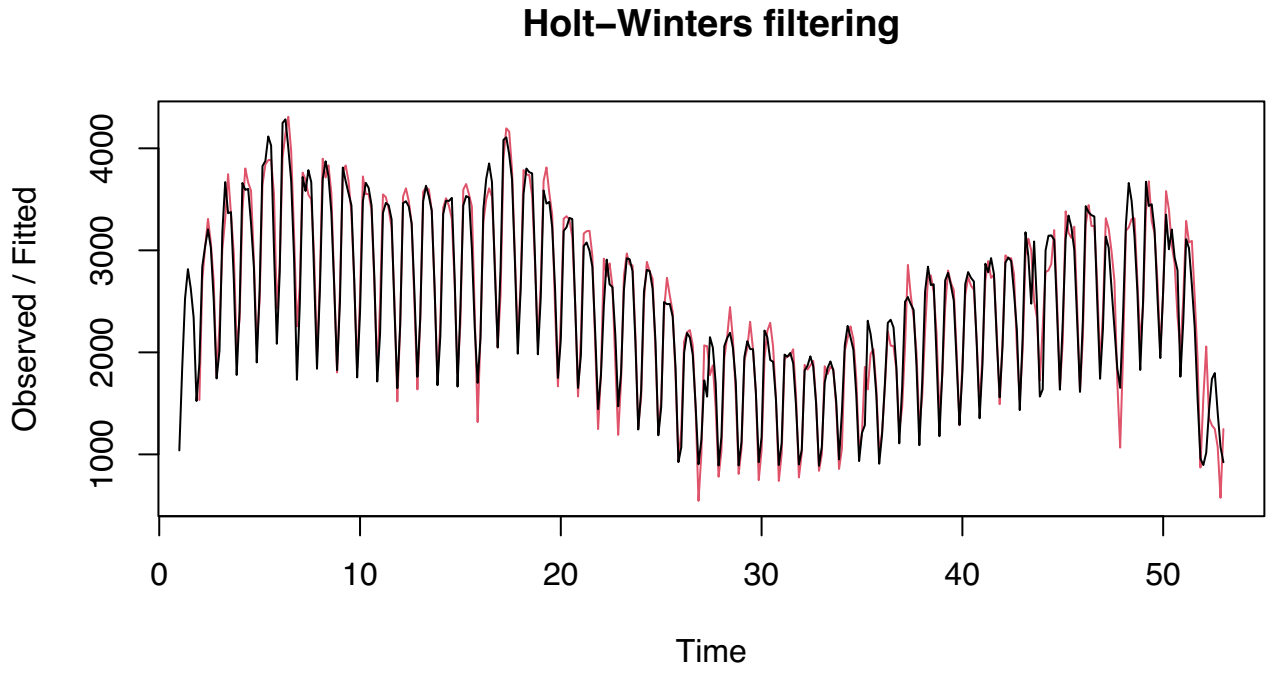
- More live coding, examples, and activities
- Randomized groups :)
- Slow it down Benson Boone style

Review

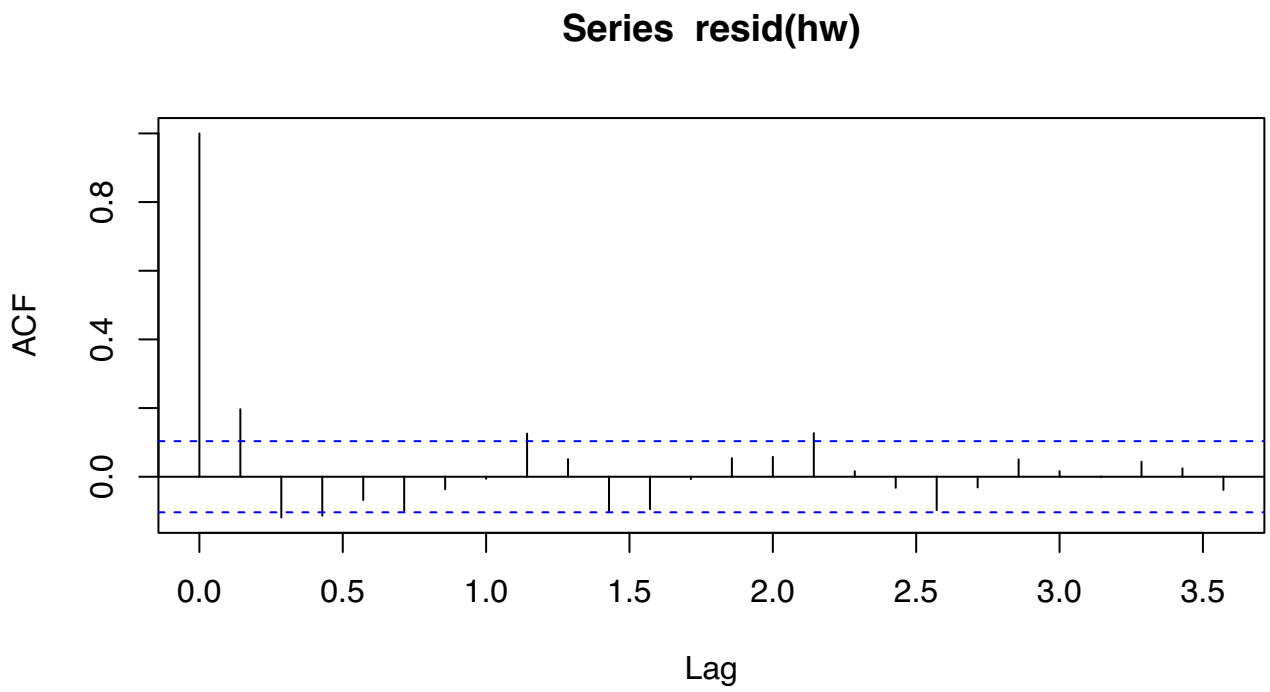
Construct a time series of the daily unique visits to the website in the year of 2017, then fit a Holt-Winters model to the data and plot the resulting model. Then create an ACF plot of the residual error series and comment on what you see.

> **Tip**
>
> You can use the `resid` function on a Holt-Winters model to obtain the residual error series.

```
web2017 <- web %>%
  mutate(dt = mdy(Date)) %>%
  filter(year(dt) == 2017) %>%
  arrange(dt)

web_ts <- ts(
  web2017$Unique.Visits,
  start = c(1,1),
  freq = 7
)

hw <- HoltWinters(web_ts)
plot(hw)
```



```
acf(resid(hw))
```



Note

A time series $\{w_t : t = 1, 2, \dots, n\}$ is called discrete white noise (DWN) if the variables $w_1, w_2, \dots, w_n$ are independent and identically distributed with a mean of 0.

If the variables also follow a normal distribution, the series is called Gaussian Noise.

$$w_t \sim N(0, \sigma^2)$$

(normal; variance $= \sigma^2$; mean $= 0$)

Second-order properties

$$\mu(t) = 0$$

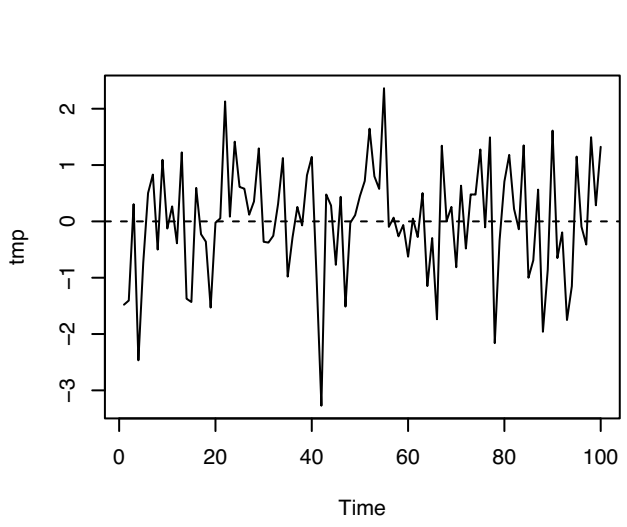
$$\gamma_k = \text{Cov}(w_t, w_{t+k}) = \begin{cases} \sigma^2 & k = 0 \\ 0 & k \neq 0 \end{cases}$$

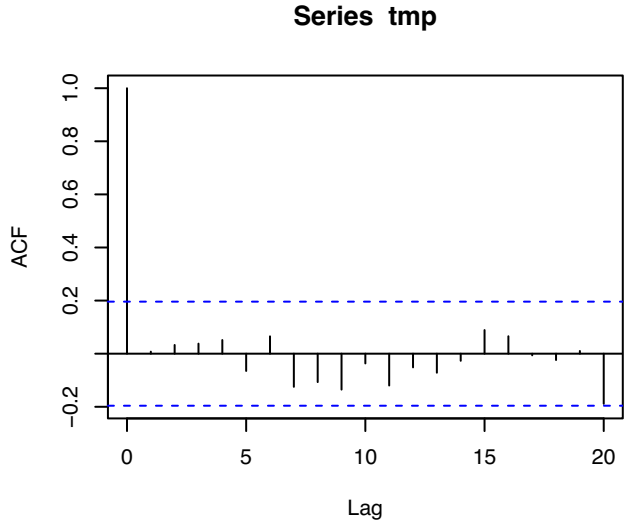
$$\rho_k = \begin{cases} 1 & k = 0 \\ 0 & k \neq 0 \end{cases}$$

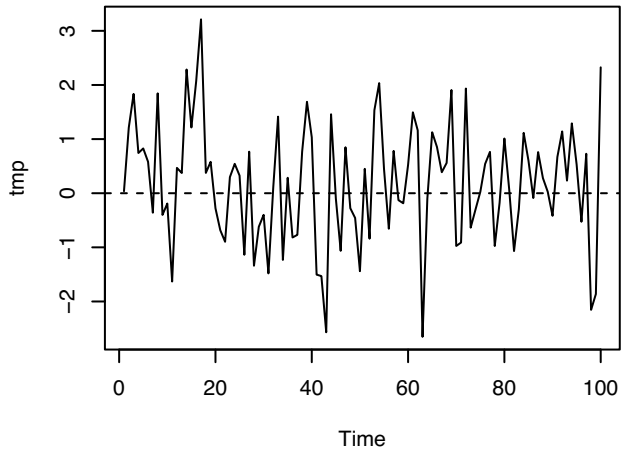
How can we simulate Gaussian white noise?

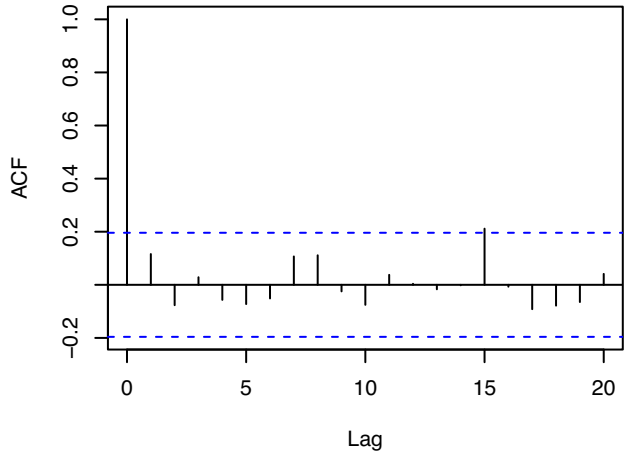
```
set.seed(10062024)
par(mfrow = c(3, 2))
for(i in 1:3) {
  tmp <- rnorm(100)
  plot(
    tmp, type = "l",
    xlab = "Time"
  )
  abline(h = 0, lty = 2)
  acf(tmp)
}
```

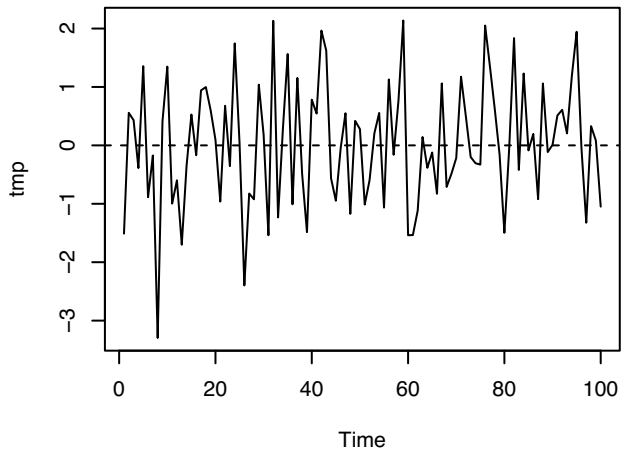
generate 100 random normal observations

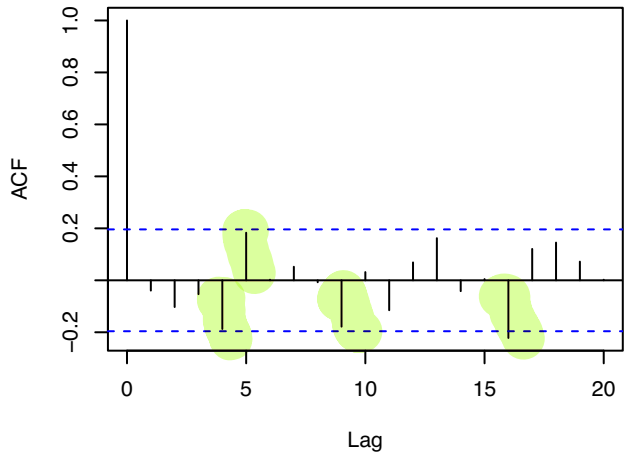

tmp
Time


Series tmp
ACF
Lag


tmp
Time


Series tmp
ACF
Lag


tmp
Time


Series tmp
ACF
Lag

Random walks

Note

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

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

where w_t is discrete white noise. It can be shown that

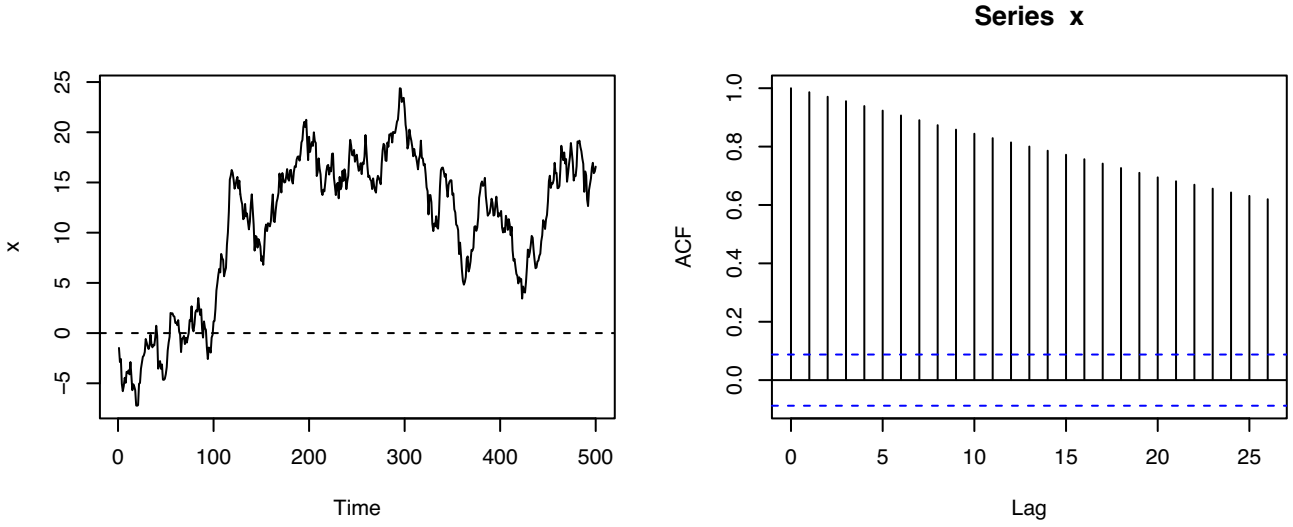
$$x_t = w_t + w_{t-1} + w_{t-2} + \ldots$$

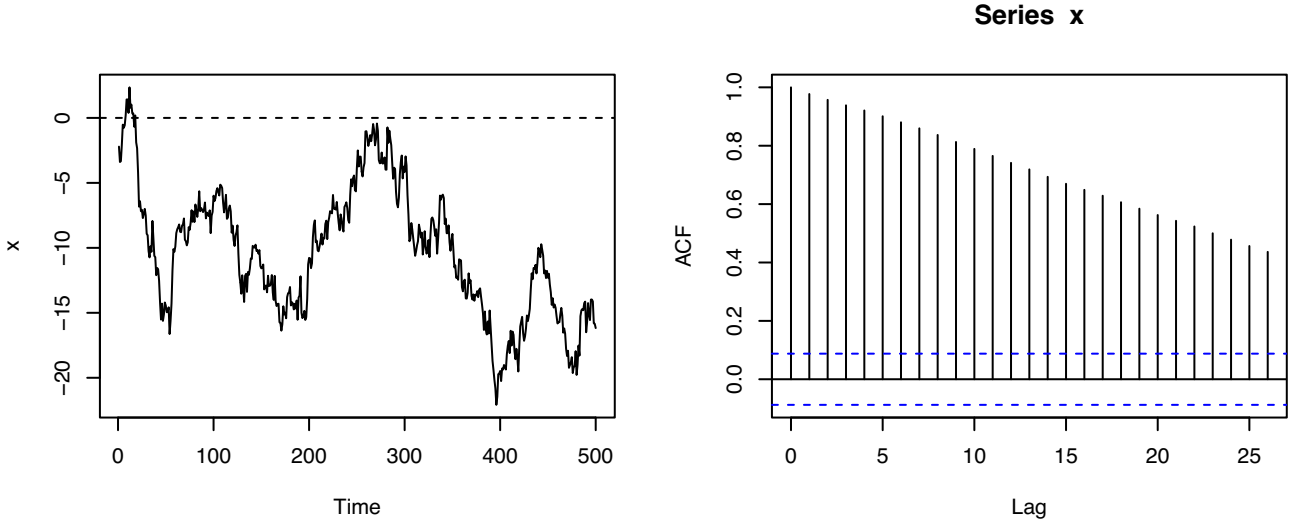
Second-order properties

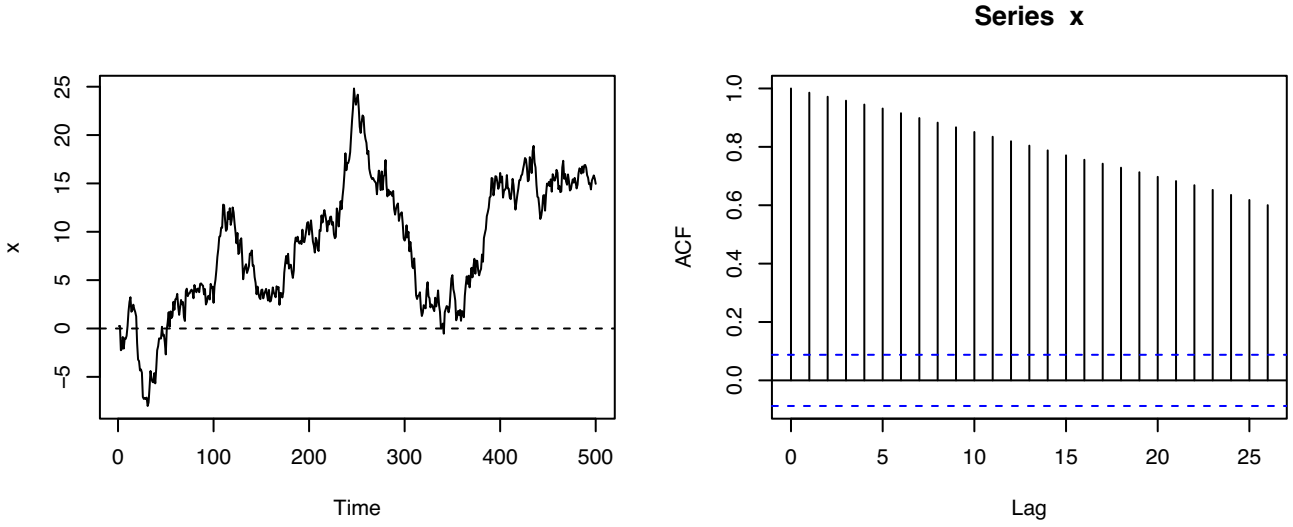
$$\mu(t) = 0$$
$$\gamma_k(t) = \text{Cov}(x_t, x_{t+k}) = t\sigma^2$$
$$\rho_k(t) = \frac{\text{Cov}(x_t, x_{t+k})}{\sqrt{\text{Var}(x_t)\text{Var}(x_{t+k})}} = \frac{t\sigma^2}{\sqrt{t\sigma^2(t+k)\sigma^2}} = \frac{1}{\sqrt{1+k/t}}$$

How can we simulate a random walk?

```
set.seed(10062024)
par(mfrow = c(3, 2))
for(i in 1:3) {
  x <- w <- rnorm(500)
  for(t in 2:500) x[t] <- x[t-1] + w[t]
  plot(x, type = "l", xlab = "Time")
  abline(h = 0, lty = 2)
  acf(x)
}
```


x
Time
Series x
ACF
Lag


x
Time
Series x
ACF
Lag

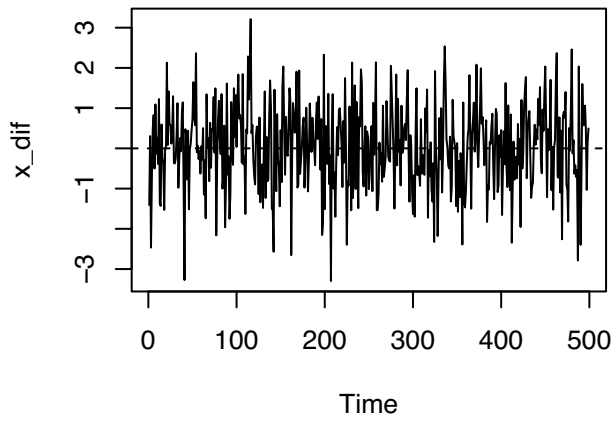

x
Time
Series x
ACF
Lag

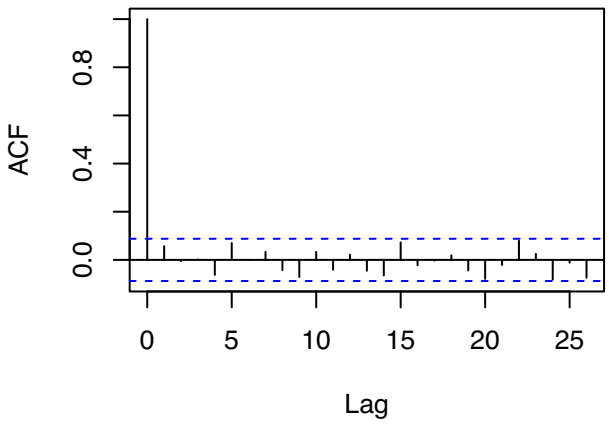
```r
set.seed(10062024)
par(mfrow = c(2, 2))
for(i in 1:2) {
  # generate random walk
  x <- w <- rnorm(500)
  for(t in 2:500) x[t] <- x[t-1] + w[t]

  # first order difference
  x_dif <- diff(x)
  plot(x_dif, type = "l", xlab = "Time")
  abline(h = 0, lty = 2)
  acf(x_dif)
}
```

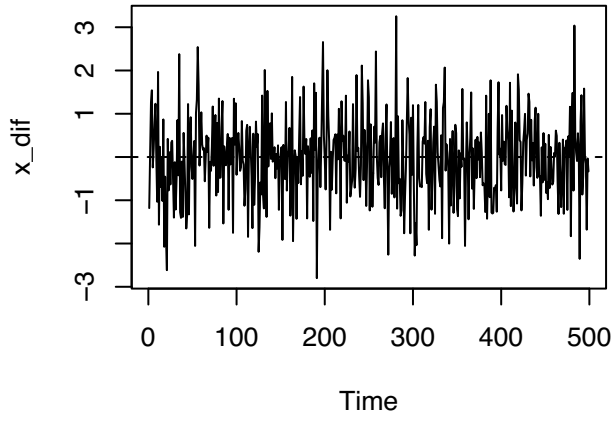
first order difference

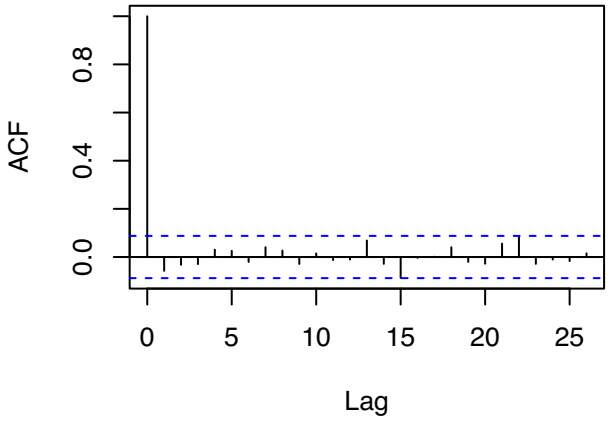
$x_t - x_{t-1}$

First order differences of a random walk are realizations of white noise.









Backshift operators

The backward shift operator B is defined by

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

and

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

which provides a convenient way to express autoregressive series.

Example: Express the random walk model in terms of the backshift operator.

$$X_t = X_{t-1} + \omega_t$$

$$X_t = BX_t + \omega_t$$

$$X_t - BX_t = \omega_t$$

$$(1 - B)X_t = \omega_t$$

↳ characteristic polynomial

Random walk with drift

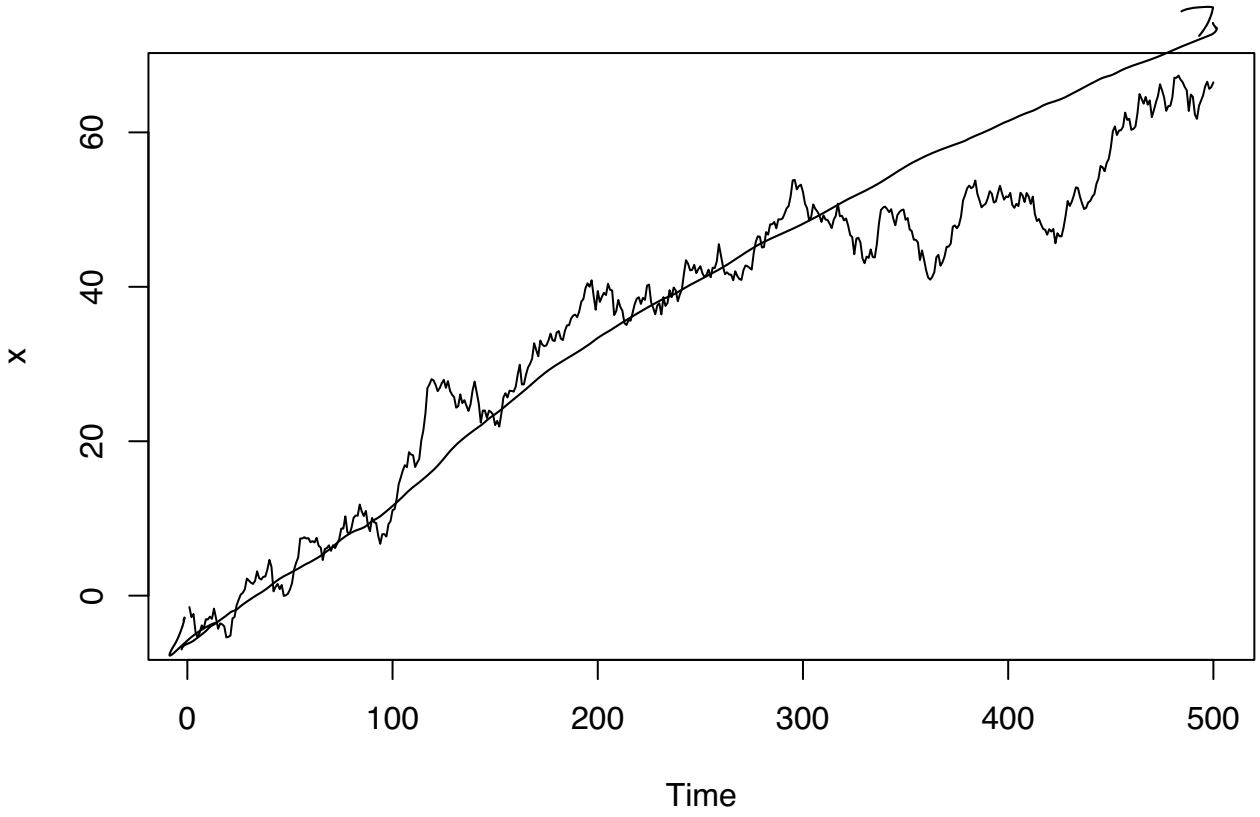
Note

A time series $\{x_t\}$ is a random walk w/ drift if

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

where w_t is white noise.

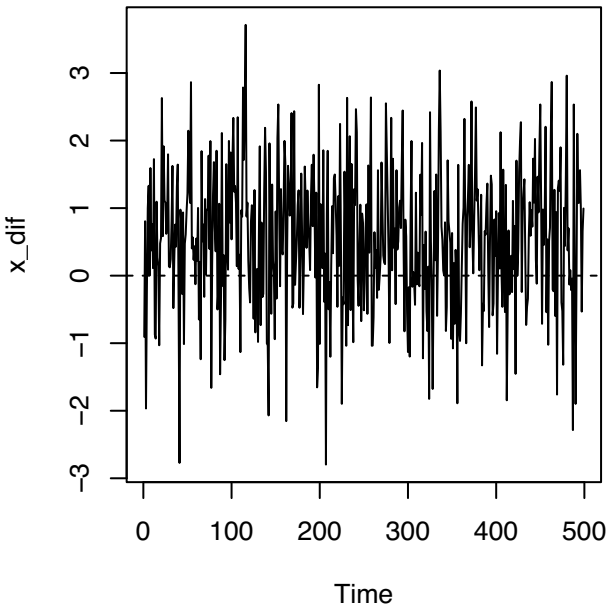
```
set.seed(10062024)
par(mfrow = c(1,1))
# generate random walk with drift
x <- w <- rnorm(500)
for(t in 2:500) x[t] <- x[t-1] + .10 + w[t]

plot(x, xlab = "Time", type = "l")
```

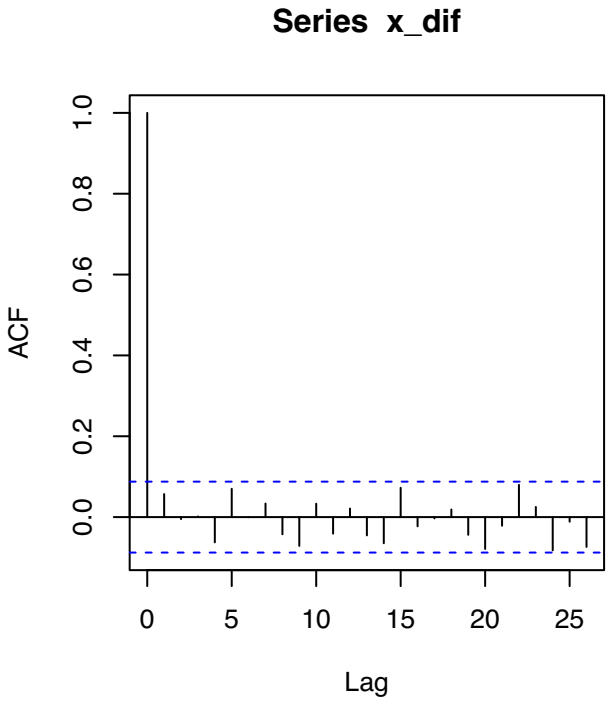

x
0
20
40
60
0
100
200
300
400
500
Time

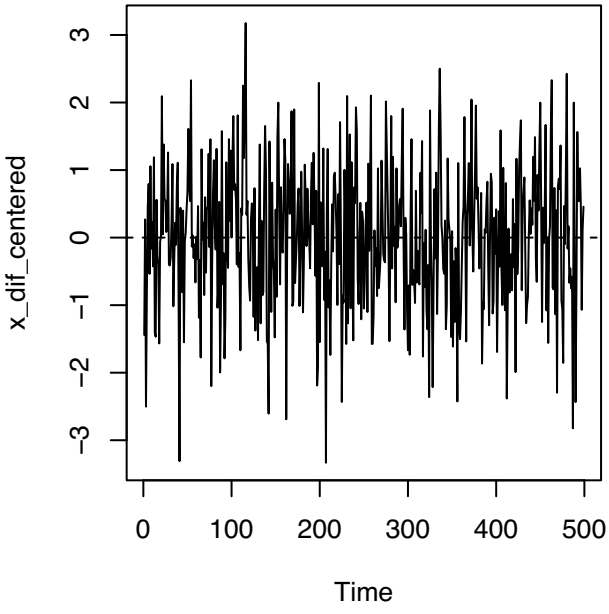
```r
set.seed(10062024)
par(mfrow = c(2, 2))
# generate random walk with drift
x <- w <- rnorm(500)
for(t in 2:500) x[t] <- x[t-1] + .5 + w[t]

# first order difference - notice the mean
x_dif <- diff(x)
mean(x_dif)
```

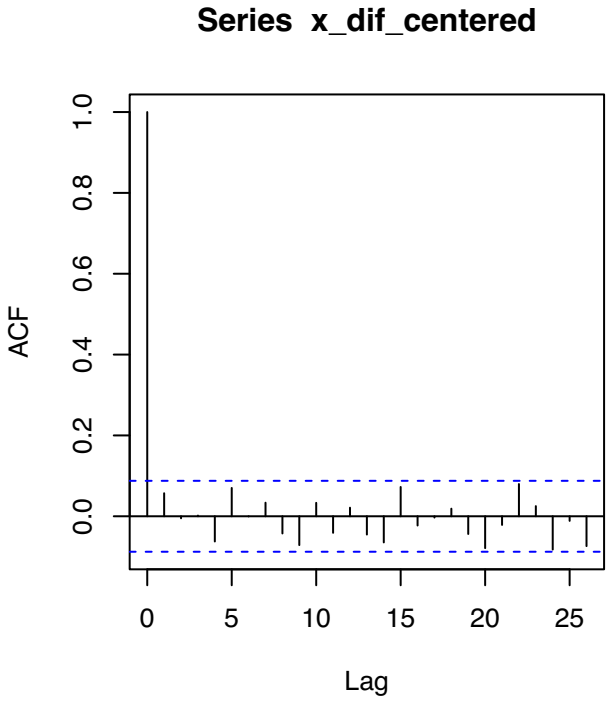
```
[1] 0.5361847
```

```r
plot(x_dif, type = "l", xlab = "Time")
abline(h = 0, lty = 2)
acf(x_dif)

# we can fix that
x_dif_centered <- x_dif - mean(x_dif)
plot(x_dif_centered, type = "l", xlab = "Time")
abline(h = 0, lty = 2)
acf(x_dif_centered)
```


x_dif
Time
−3
−2
−1
0
1
2
3
0
100
200
300
400
500


Series x_dif
ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25


x_dif_centered
Time
−3
−2
−1
0
1
2
3
0
100
200
300
400
500


Series x_dif_centered
ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25