

Applied Time Series Analysis

SS 2016 – Introduction

Marcel Dettling

Institute for Data Analysis and Process Design

Zurich University of Applied Sciences

marcel.dettling@zhaw.ch

<http://stat.ethz.ch/~dettling>

ETH Zürich, February-May 2016

Applied Time Series Analysis

SS 2016 – Introduction

Your Lecturer



Name: **Marcel Dettling**

Age: 41 Years

Civil Status: Married, 2 children

Education: Dr. Math. ETH

Position:

Lecturer @ ETH Zürich and @ ZHAW

Researcher in Applied Statistics @ ZHAW

Connection:

Research with industry: *hedge funds, insurance, ...*

Academic research: *high-frequency financial data*

Applied Time Series Analysis

SS 2016 – Introduction

A First Example

In 2006, Singapore Airlines decided to place an order for new aircraft. It contained the following jets:

- 20 Boeing 787
- 20 Airbus A350
- 9 Airbus A380

How was this decision taken?

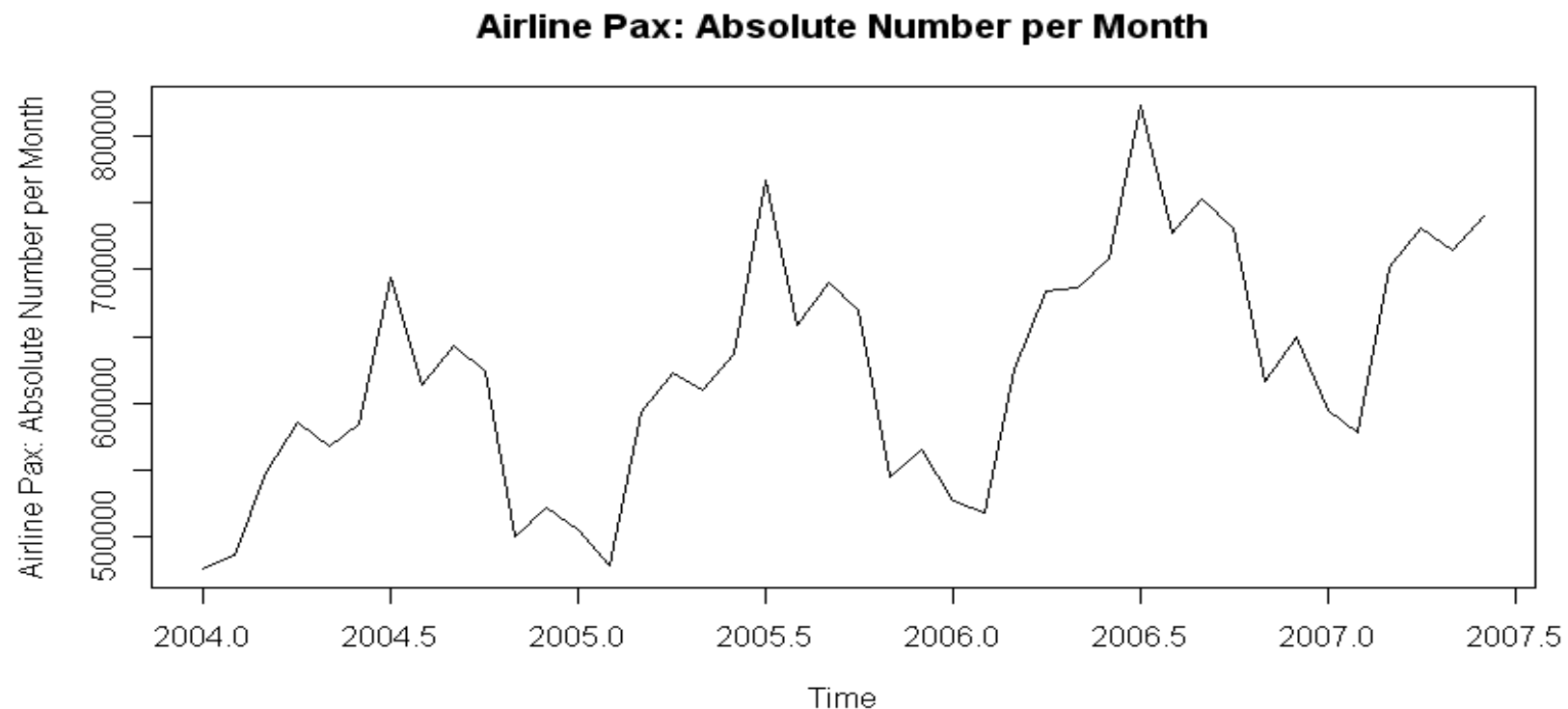
It was based on a combination of time series analysis on airline passenger trends, plus knowing the corporate plans for maintaining or increasing the market share.

Applied Time Series Analysis

SS 2016 – Introduction

A Second Example

- Taken from a former research project @ ZHAW
- Airline business: # of checked-in passengers per month



Applied Time Series Analysis

SS 2016 – Introduction

Some Properties of the Series

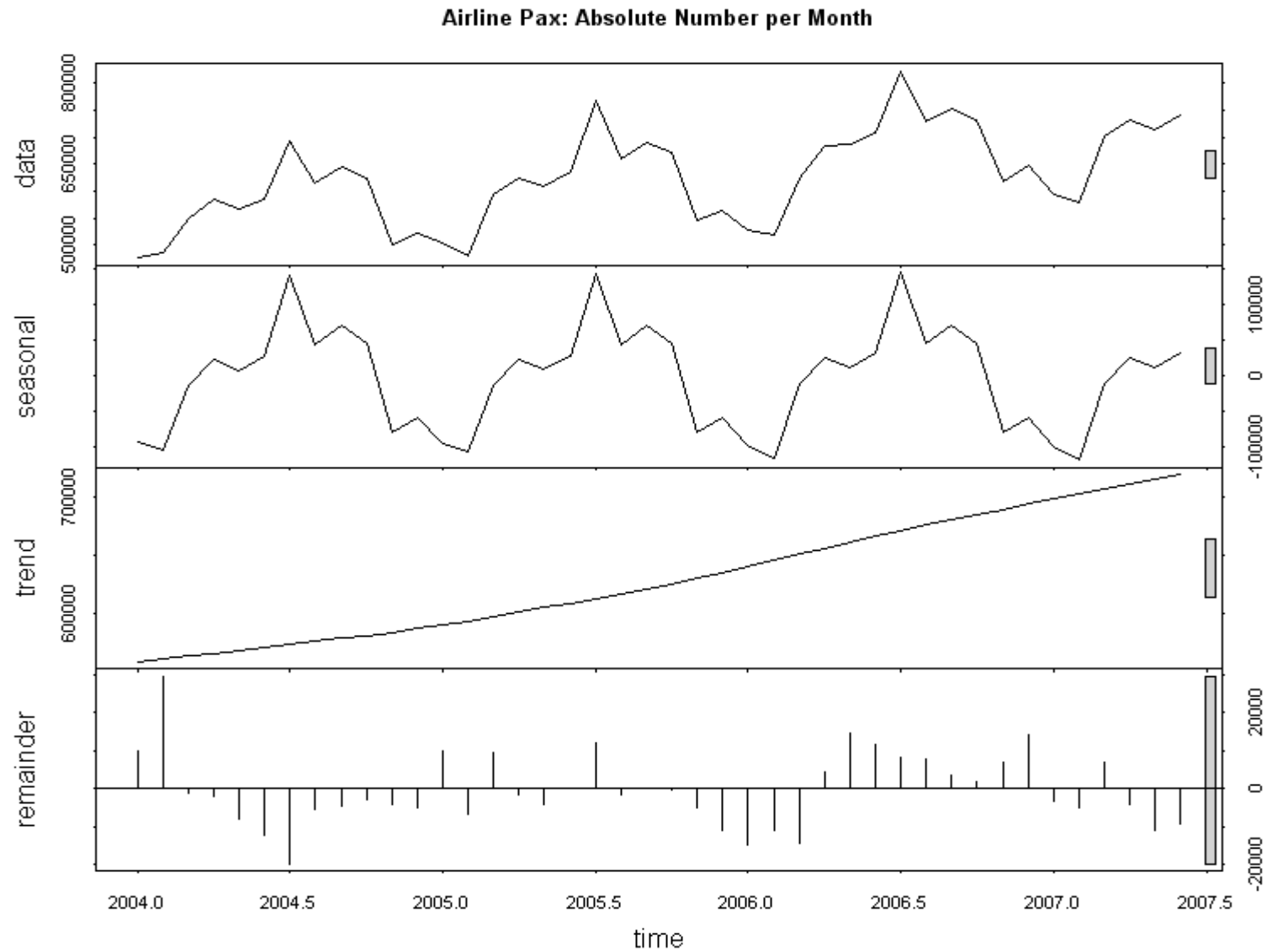
- Increasing trend (i.e. generally more passengers)
- Very prominent seasonal pattern (i.e. peaks/valleys)
- Hard to see details beyond the obvious

Goals of the Project

- Visualize, or better, extract trend and seasonal pattern
- Quantify the amount of random variation/uncertainty
- Provide the basis for a man-made forecast after mid-2007
- Forecast (extrapolation) from mid-2007 until end of 2008
- How can we better organize/collect data?

Applied Time Series Analysis

SS 2016 – Introduction



Applied Time Series Analysis

SS 2016 – Introduction

Organization of the Course

Contents:

- Basics, Mathematical Concepts, Time Series in R
- Descriptive Analysis (Plots, Decomposition, Correlation)
- Models for Stationary Series (AR(p), MA(q), ARMA(p,q))
- Non-Stationary Models (SARIMA, GARCH, Long-Memory)
- Forecasting (Regression, Exponential Smoothing, ARMA)
- Miscellaneous (Multivariate, Spectral Analysis, State Space)

Goal:

The students acquire experience in analyzing time series problems, are able to work with the software package R, and can perform time series analyses correctly on their own.

Applied Time Series Analysis

SS 2016 – Introduction

Organization of the Course

Applied Time Series Analysis – SS 2016

People:

Lecturer: Dr. Marcel Dettling (marcel.dettling@stat.math.ethz.ch)
Assistants: Sonja Gassner (WBL) (sonja.gassner@stat.math.ethz.ch)
Ruben Dezeure (Other) (ruben.dezeure@stat.math.ethz.ch)

Organization:

This course on Applied Time Series Analysis has a heterogeneous audience. It will be visited by students of the "Weiterbildungs-Lehrgang in Angewandter Statistik" (WBL, i.e. "Advanced Studies in Applied Statistics"), from the Master of Science in Statistics, as well as from several bachelor, masters and doctoral programs of other faculties. Rules and organization will differ somewhat between the WBL and other students, see below.

Lectures:

Lectures for all students will be held on Mondays from 10.15-11.55 at ETH Zentrum, room HG E1.2. Theory and examples will be shown on power point slides and the blackboard. Also, a script is available. The tentative schedule is as follows:

Week	Date	L/L	Topics
01	22.02.2016	L/L	Introduction; Stationarity; Visualization
02	29.02.2016	L/L	Transformation; Decomposition
03	07.03.2016	L/L	Autocorrelation; Partial Autocorrelation
04	14.03.2016	L/L	White Noise, Autoregressive Models
05	21.03.2016	L/L	Autoregressive and Moving Average Models
--	28.03.2016	-/-	Easter Break
06	04.04.2016	L/L	Autoregressive and Moving Average Models
07	11.04.2016	L/L	Time Series Regression
08	18.04.2016	L/L	ARIMA and SARIMA Models
09	25.04.2016	L/L	Forecasting 1
10	02.05.2016	L/L	Forecasting 2
11	09.05.2016	L/L	Multivariate Time Series Analysis
12	16.05.2016	-/-	Whitmonday
13	23.05.2016	L/L	Spectral Analysis
14	30.05.2016	L/L	Miscellaneous, Outlook

Exercises:

Exercises for the WBL students are held weekly on Monday 08.15-10.00 at HG E19 and HG D11 in the usual WBL exercise mode. They start on Monday, February 29, 2016. For all other students, there are only bi-weekly exercises, which take place on Monday 15.15-17.00 in HG E1.2. All exercises will be guided tutorials: you are expected to solve the problems using the statistical software package R, an assistant will be there to give instructions and support. You need to bring your own laptop with R installed on it. If you wish, you can hand in your solution to obtain a feedback on it. Please send code and/or comments by e-mail with "[ATSAA]" in the subject line to the assistant Ruben Dezeure until at last a week after the exercise. Feedback will be given until about another week later also via e-mail.

→ more details are given on the additional organization sheet

Applied Time Series Analysis

SS 2016 – Introduction

What is a Time Series?

A time series is a set of observations $x = (x_1, \dots, x_n)$, where each of the observations was made at a specific time t .

- In contrast to multivariate statistics, the data in a time series are usually not iid, but are serially correlated.
- We assume that the observations were made at fixed time intervals and do not treat continuous or irregular series.

Rationale behind time series analysis:

The rationale in time series analysis is to understand the pattern of the dependencies in the past of the series, and exploit them to be able to predict the future well.

Applied Time Series Analysis

SS 2016 – Introduction

Example 1: Air Passenger Bookings

```
> data(AirPassengers)
```

```
> AirPassengers
```

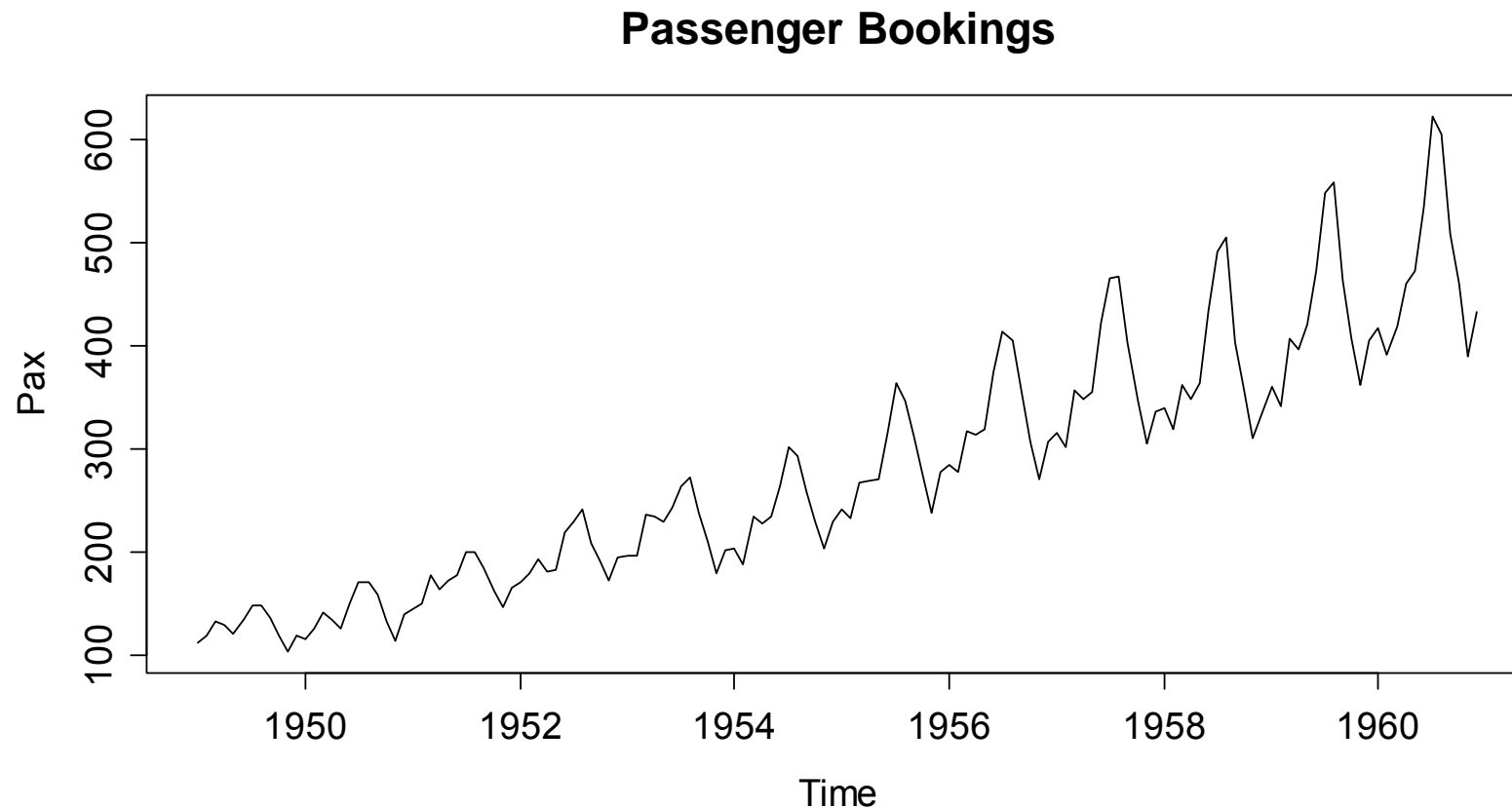
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949	112	118	132	129	121	135	148	148	136	119	104	118
1950	115	126	141	135	125	149	170	170	158	133	114	140
1951	145	150	178	163	172	178	199	199	184	162	146	166
1952	171	180	193	181	183	218	230	242	209	191	172	194
1953	196	196	236	235	229	243	264	272	237	211	180	201
1954	204	188	235	227	234	264	302	293	259	229	203	229
1955	242	233	267	269	270	315	364	347	312	274	237	278
1956	284	277	317	313	318	374	413	405	355	306	271	306
1957	315	301	356	348	355	422	465	467	404	347	305	336
1958	340	318	362	348	363	435	491	505	404	359	310	337
1959	360	342	406	396	420	472	548	559	463	407	362	405
1960	417	391	419	461	472	535	622	606	508	461	390	432

Applied Time Series Analysis

SS 2016 – Introduction

Example 1: Air Passenger Bookings

```
> plot(AirPassengers, ylab="Pax", main="Pax Bookings")
```

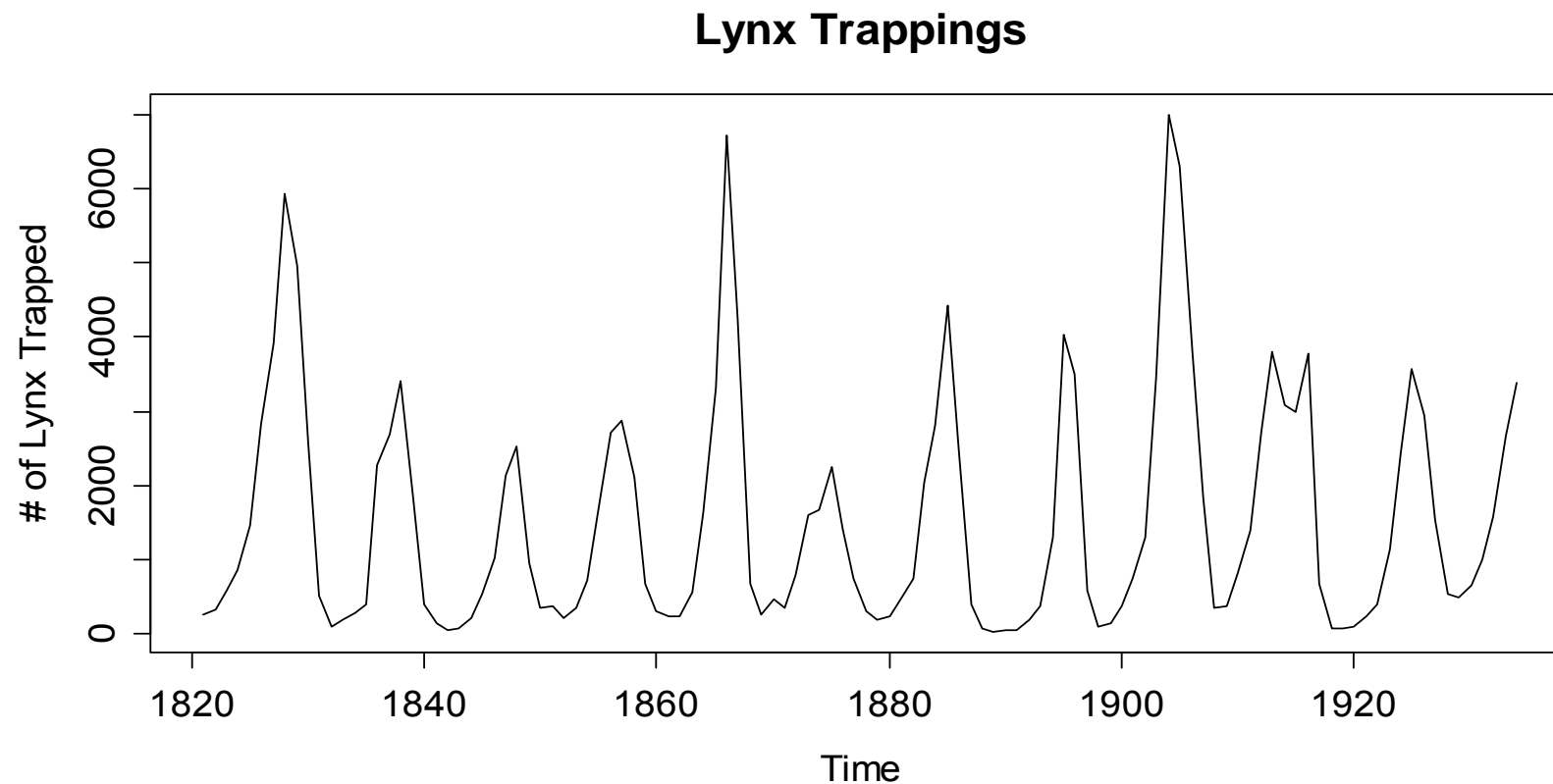


Applied Time Series Analysis

SS 2016 – Introduction

Example 2: Lynx Trappings

```
> data(lynx)  
> plot(lynx, ylab="# of Lynx", main="Lynx Trappings")
```

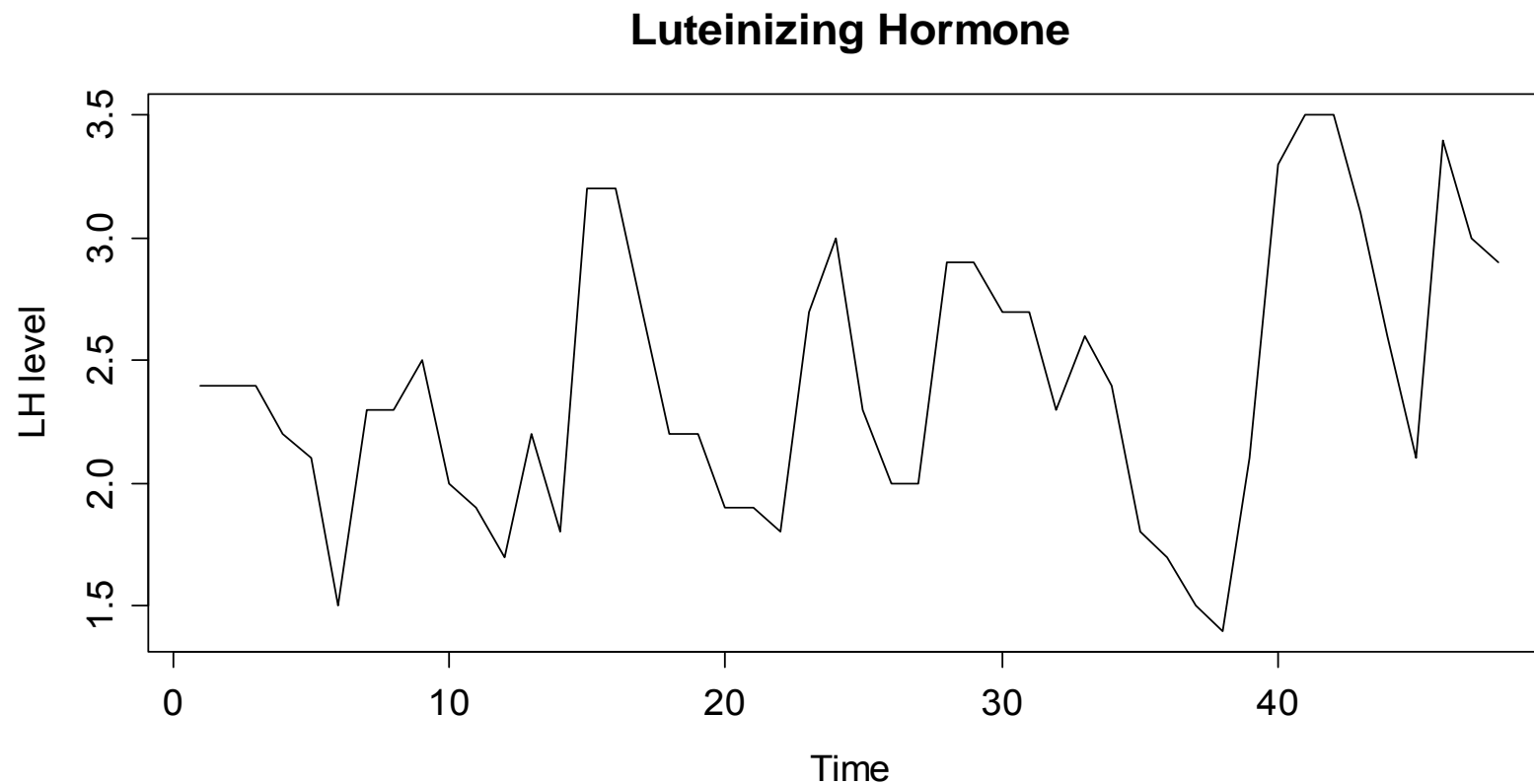


Applied Time Series Analysis

SS 2016 – Introduction

Example 3: Luteinizing Hormone

```
> data(lh)  
> plot(lh, ylab="LH level", main="Luteinizing Hormone")
```

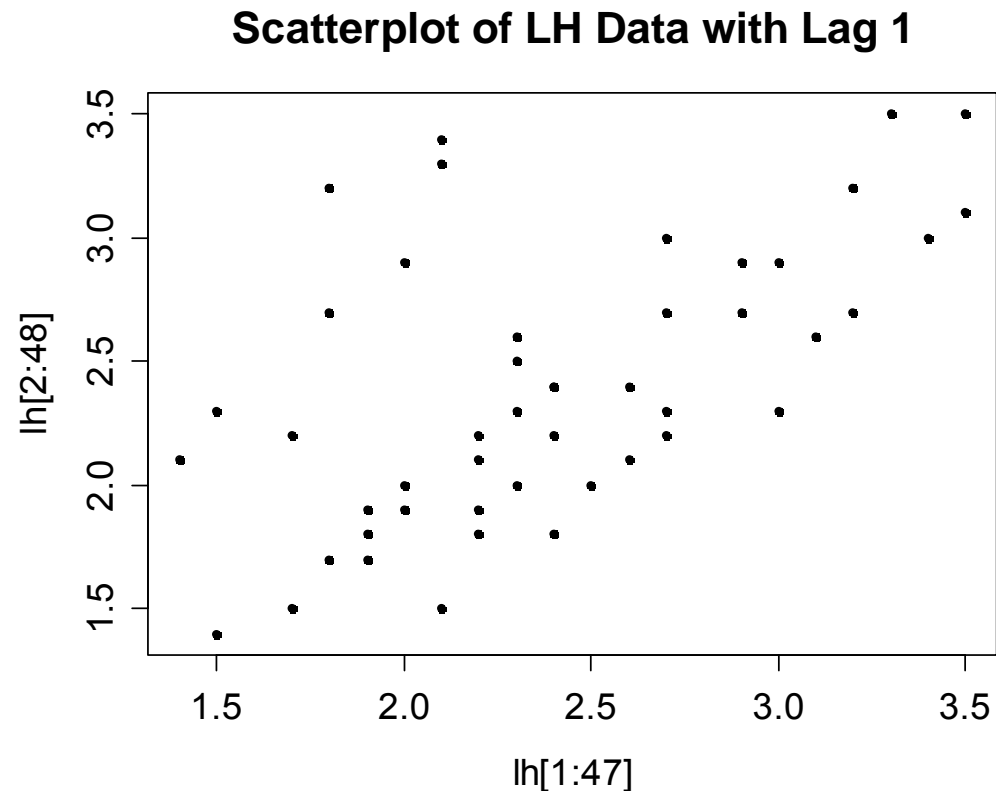


Applied Time Series Analysis

SS 2016 – Introduction

Example 3: Lagged Scatterplot

```
> plot(lh[1:47], lh[2:48], pch=20)  
> title("Scatterplot of LH Data with Lag 1")
```



Applied Time Series Analysis

SS 2016 – Introduction

Example 4: Swiss Market Index

We have a multiple time series object:

```
> data(EuStockMarkets)
> EuStockMarkets
Time Series:
Start = c(1991, 130)
End = c(1998, 169)
Frequency = 260
```

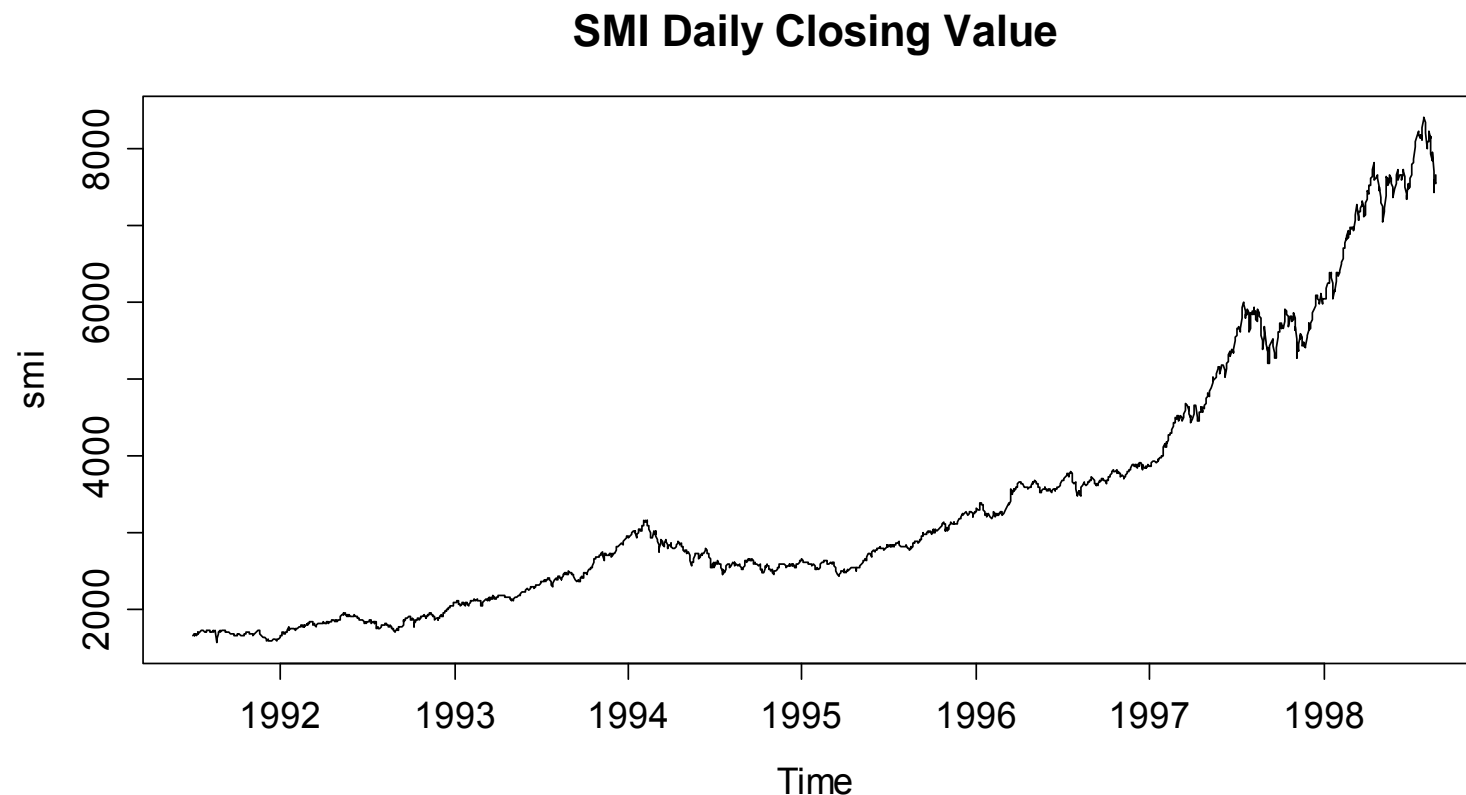
	DAX	SMI	CAC	FTSE
1991.496	1628.75	1678.1	1772.8	2443.6
1991.500	1613.63	1688.5	1750.5	2460.2
1991.504	1606.51	1678.6	1718.0	2448.2
1991.508	1621.04	1684.1	1708.1	2470.4
1991.512	1618.16	1686.6	1723.1	2484.7
1991.515	1610.61	1671.6	1714.3	2466.8

Applied Time Series Analysis

SS 2016 – Introduction

Example 4: Swiss Market Index

```
> smi <- ts(tmp, start=start(esm), freq=frequency(esm))  
> plot(smi, main="SMI Daily Closing Value")
```

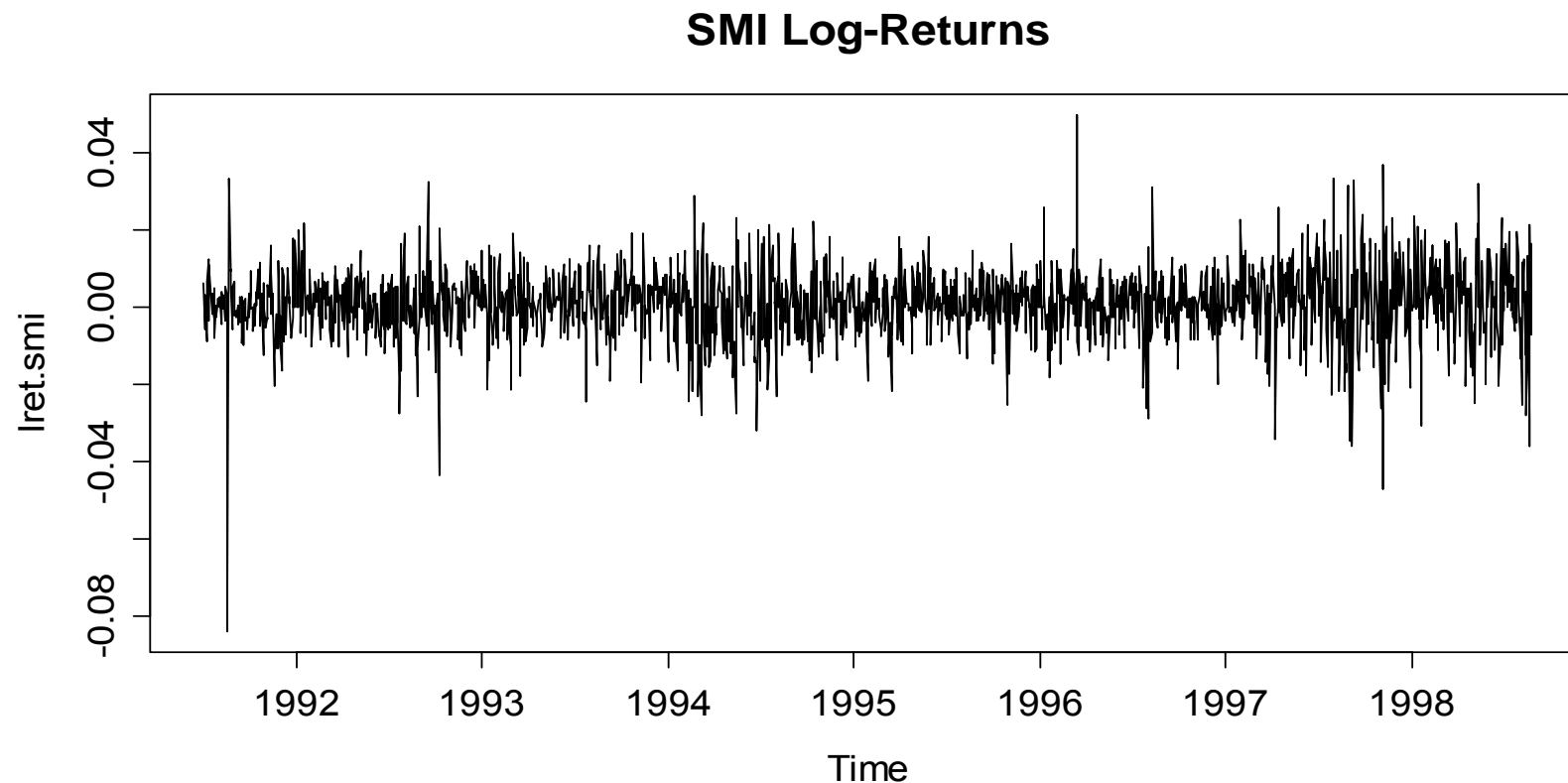


Applied Time Series Analysis

SS 2016 – Introduction

Example 4: Swiss Market Index

```
> lret.smi <- log(smi[2:1860]/smi[1:1859])  
> plot(lret.smi, main="SMI Log>Returns")
```



Applied Time Series Analysis

SS 2016 – Introduction

Goals in Time Series Analysis

1) Exploratory Analysis

Visualization of the properties of the series

- *time series plot*
- *decomposition into trend/seasonal pattern/random error*
- *correlogram for understanding the dependency structure*

2) Modeling

Fitting a stochastic model to the data that represents and reflects the most important properties of the series

- *done exploratory or with previous knowledge*
- *model choice and parameter estimation is crucial*
- *inference: how well does the model fit the data?*

Applied Time Series Analysis

SS 2016 – Introduction

Goals in Time Series Analysis

3) Forecasting

Prediction of future observations with measure of uncertainty

- *mostly model based, uses dependency and past data*
- *is an extrapolation, thus often to take with a grain of salt*
- *similar to driving a car by looking in the rear window mirror*

4) Process Control

The output of a (physical) process defines a time series

- *a stochastic model is fitted to observed data*
- *this allows understanding both signal and noise*
- *it is feasible to monitor normal/abnormal fluctuations*

Applied Time Series Analysis

SS 2016 – Introduction

Goals in Time Series Analysis

5) Time Series Regression

Modeling response time series using 1 or more input series

$$Y_t = \beta_0 + \beta_1 u_t + \beta_2 v_t + E_t$$

where E_t is independent of u_t and v_t , but not i.i.d.

Example: $(\text{Ozone})_t = (\text{Wind})_t + (\text{Temperature})_t + E_t$

Fitting this model under i.i.d error assumption:

- leads to unbiased estimates, but...
- often grossly wrong standard errors
- thus, confidence intervals and tests are misleading

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Stochastic Model for Time Series

Def: A *time series process* is a set $\{X_t, t \in T\}$ of random variables, where T is the set of times. Each of the random variables $X_t, t \in T$ has a univariate probability distribution F_t .

- If we exclusively consider time series processes with equidistant time intervals, we can enumerate $\{T = 1, 2, 3, \dots\}$
- An observed time series is a realization of $X = (X_1, \dots, X_n)$, and is denoted with small letters as $x = (x_1, \dots, x_n)$.
- We have a multivariate distribution, but only 1 observation (i.e. 1 realization from this distribution) is available. In order to perform “statistics”, we require some additional structure.

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Stationarity

For being able to do statistics with time series, we require that the series “doesn’t change its probabilistic character” over time. This is mathematically formulated by **strict stationarity**.

Def: A time series $\{X_t, t \in T\}$ is strictly stationary, if the joint distribution of the random vector (X_t, \dots, X_{t+k}) is equal to the one of (X_s, \dots, X_{s+k}) for all combinations of t, s and k .

→

$X_t \sim F$	all X_t are identically distributed
$E[X_t] = \mu$	all X_t have identical expected value
$Var(X_t) = \sigma^2$	all X_t have identical variance
$Cov(X_t, X_{t+h}) = \gamma_h$	autocovariance depends only on lag h

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Stationarity

It is impossible to „prove“ the theoretical concept of stationarity from data. We can only search for evidence in favor or against it.

However, with strict stationarity, even finding evidence only is too difficult. We thus resort to the concept of *weak stationarity*.

Def: A time series $\{X_t, t \in T\}$ is said to be *weakly stationary*, if

$$E[X_t] = \mu$$

$$\text{Cov}(X_t, X_{t+h}) = \gamma_h \quad \text{for all lags } h$$

$$\text{and thus also: } \text{Var}(X_t) = \sigma^2$$

Note that weak stationarity is sufficient for „practical purposes“.

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Testing Stationarity

- In time series analysis, we need to verify whether the series has arisen from a stationary process or not. Be careful: stationarity is a property of the process, and not of the data.
- Treat stationarity as a hypothesis! We may be able to reject it when the data strongly speak against it. However, we can never prove stationarity with data. At best, it is plausible.
- Formal tests for stationarity do exist (→ see script). We discourage their use due to their low power for detecting general non-stationarity, as well as their complexity.

→ **Use the time series plot for deciding on stationarity!**

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Evidence for Non-Stationarity

- **Trend**, i.e. non-constant expected value
- **Seasonality**, i.e. deterministic, periodical oscillations
- **Non-constant variance**, i.e. multiplicative error
- **Non-constant dependency structure**

Remark:

Note that some periodical oscillations, as for example in the lynx trappings data, can be stochastic and thus, the underlying process is assumed to be stationary. However, the boundary between the two is fuzzy.

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Strategies for Detecting Non-Stationarity

1) Time series plot

- non-constant expected value (trend/seasonal effect)
- changes in the dependency structure
- non-constant variance

2) Correlogram (presented later...)

- non-constant expected value (trend/seasonal effect)
- changes in the dependency structure

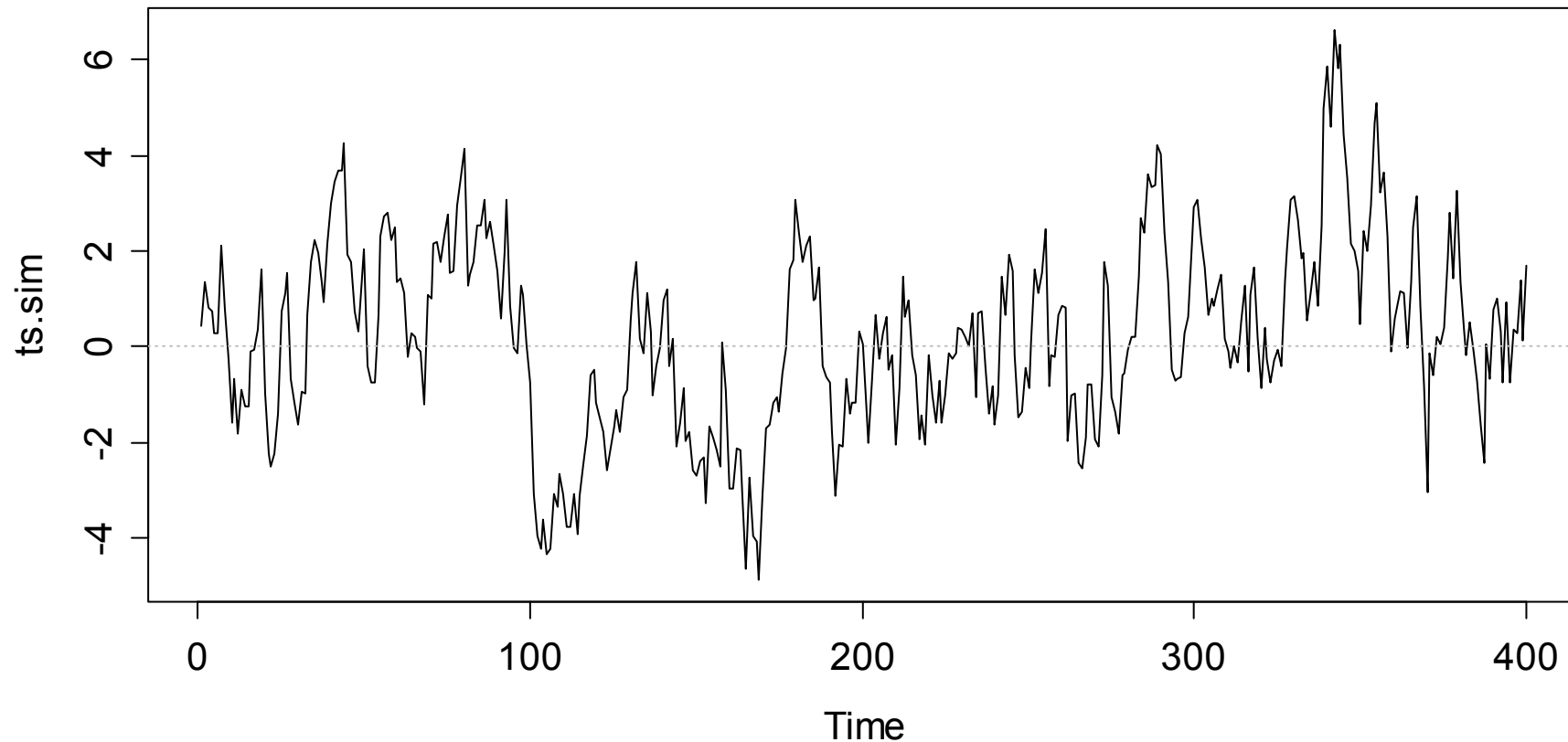
A (sometimes) useful trick, especially when working with the correlogram, is to split up the series in two or more parts, and producing plots for each of the pieces separately.

Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Example: Simulated Time Series 1

Simulated Time Series Example

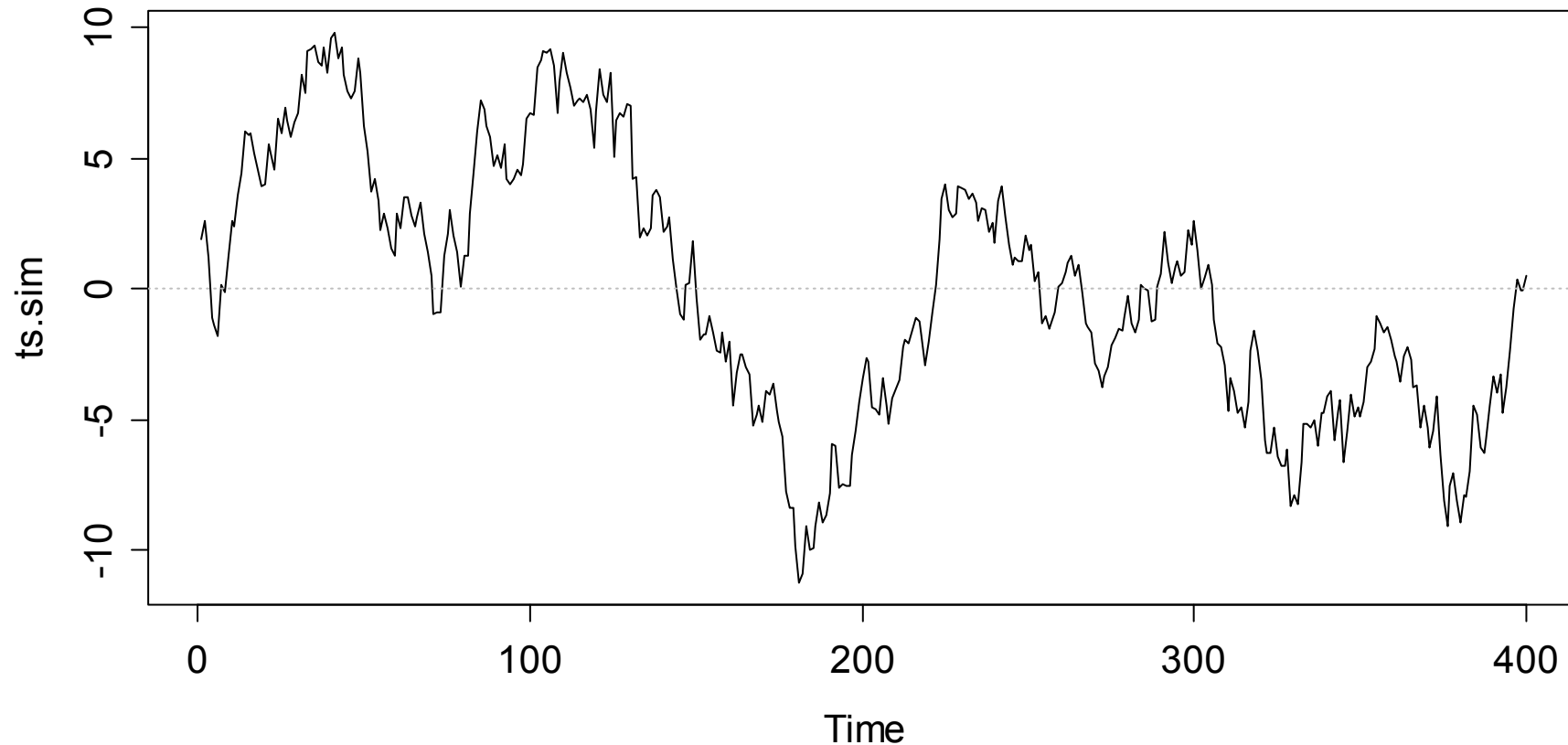


Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Example: Simulated Time Series 2

Simulated Time Series Example

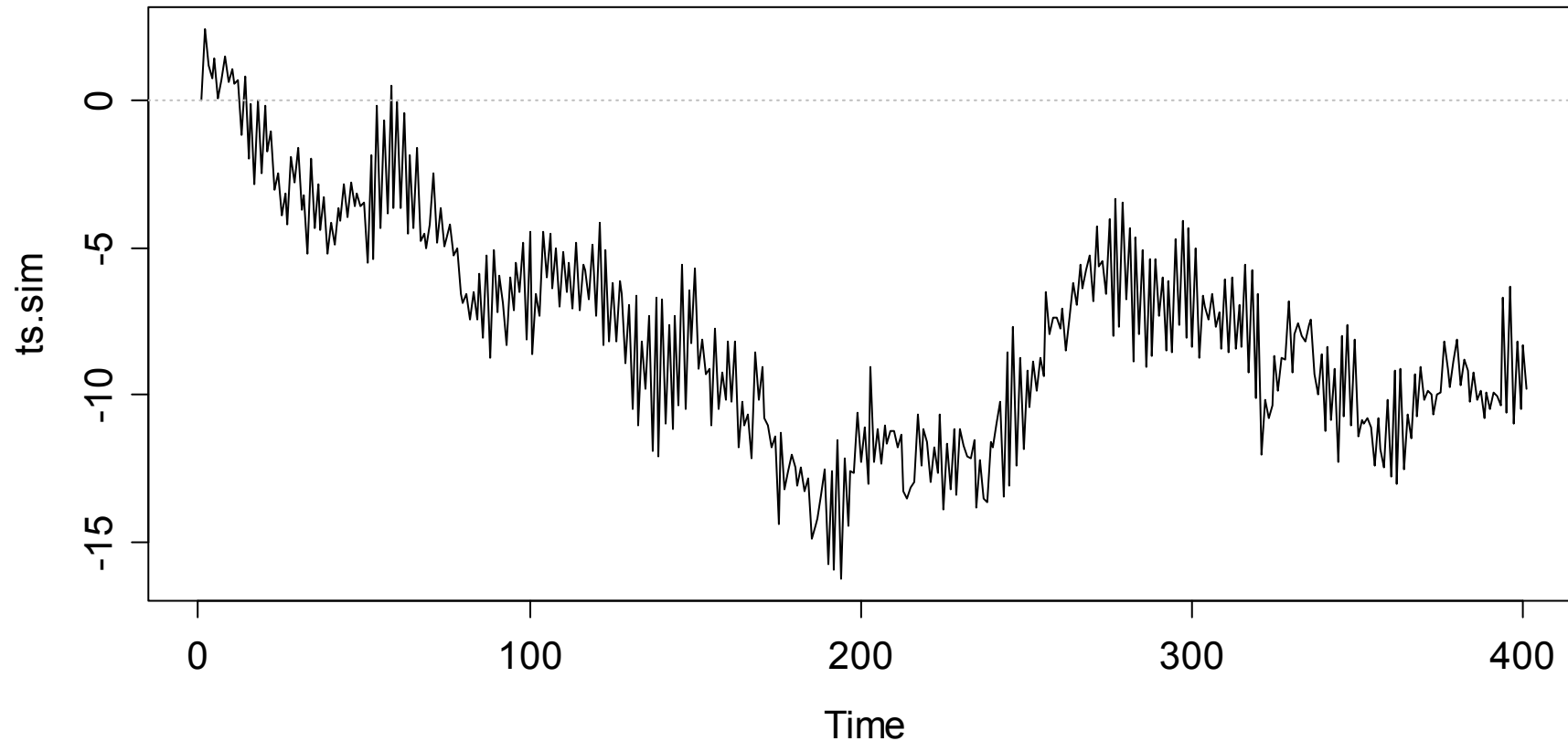


Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Example: Simulated Time Series 3

Simulated Time Series Example

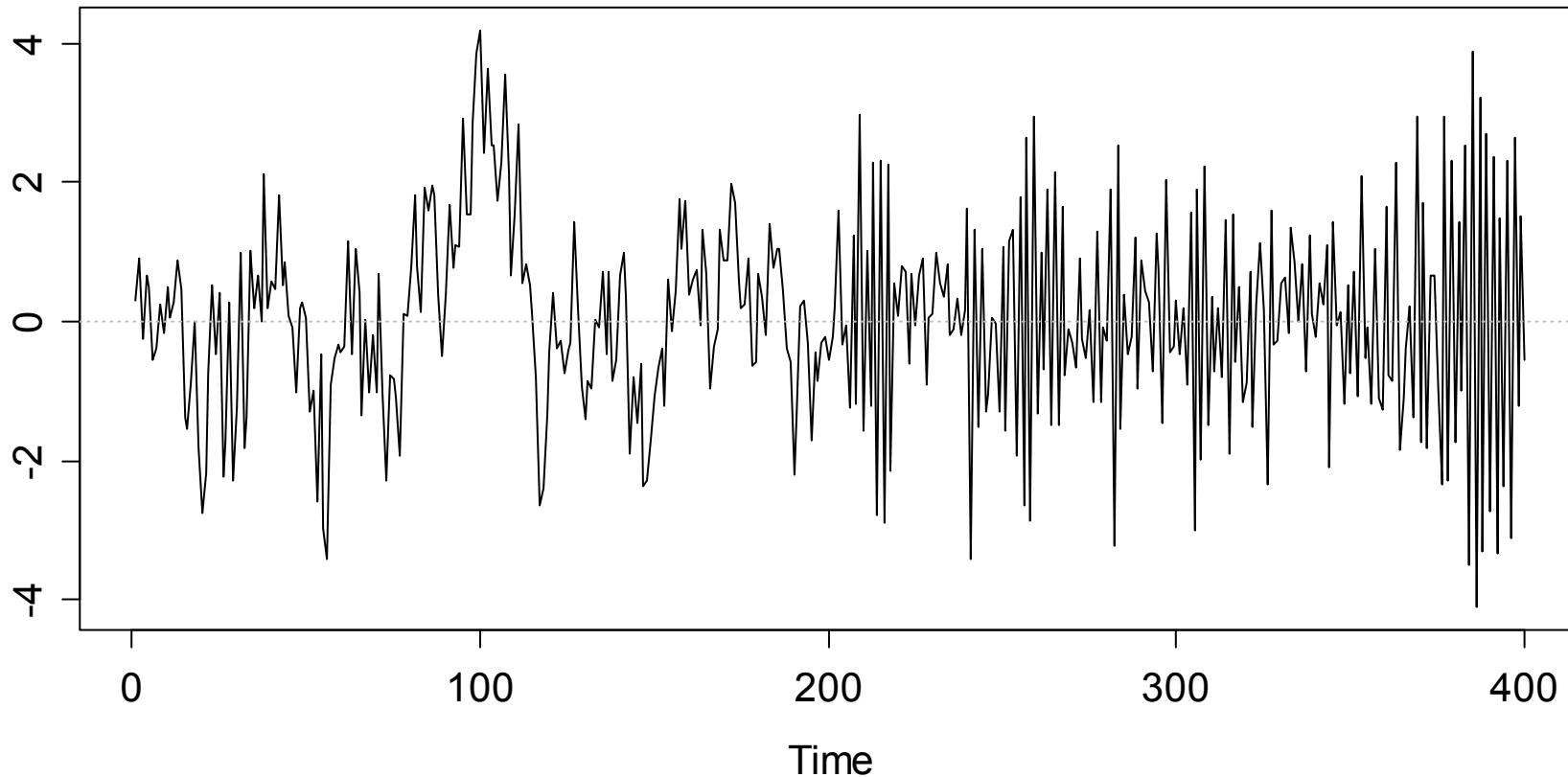


Applied Time Series Analysis

SS 2016 – Mathematical Concepts

Example: Simulated Time Series 4

Simulated Time Series Example



Applied Time Series Analysis

SS 2016 – Time Series in R

Time Series in R

- In **R**, there are *objects*, which are organized in a large number of *classes*. These classes e.g. include *vectors*, *data frames*, *model output*, *functions*, and many more. Not surprisingly, there are also *several classes for time series*.
- We focus on **ts**, the basic class for regularly spaced time series in **R**. This class is comparably simple, as it can only represent time series with *fixed interval records*, and *only uses numeric time stamps*, i.e. enumerates the index set.
- For defining a **ts** object, we have to supply the *data*, but also the *starting time* (as argument *start*), and the *frequency* of measurements as argument *frequency*.

Applied Time Series Analysis

SS 2016 – Time Series in R

Time Series in R: Example

Data: number of days per year with traffic holdups in front of the Gotthard road tunnel north entrance in Switzerland.

2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
88	76	112	109	91	98	139	150	168	149

```
> rawdat <- c(88, 76, 112, 109, 91, 98, 139, 150, 168, 149)
> ts.dat <- ts(rawdat, start=2004, freq=1)
```

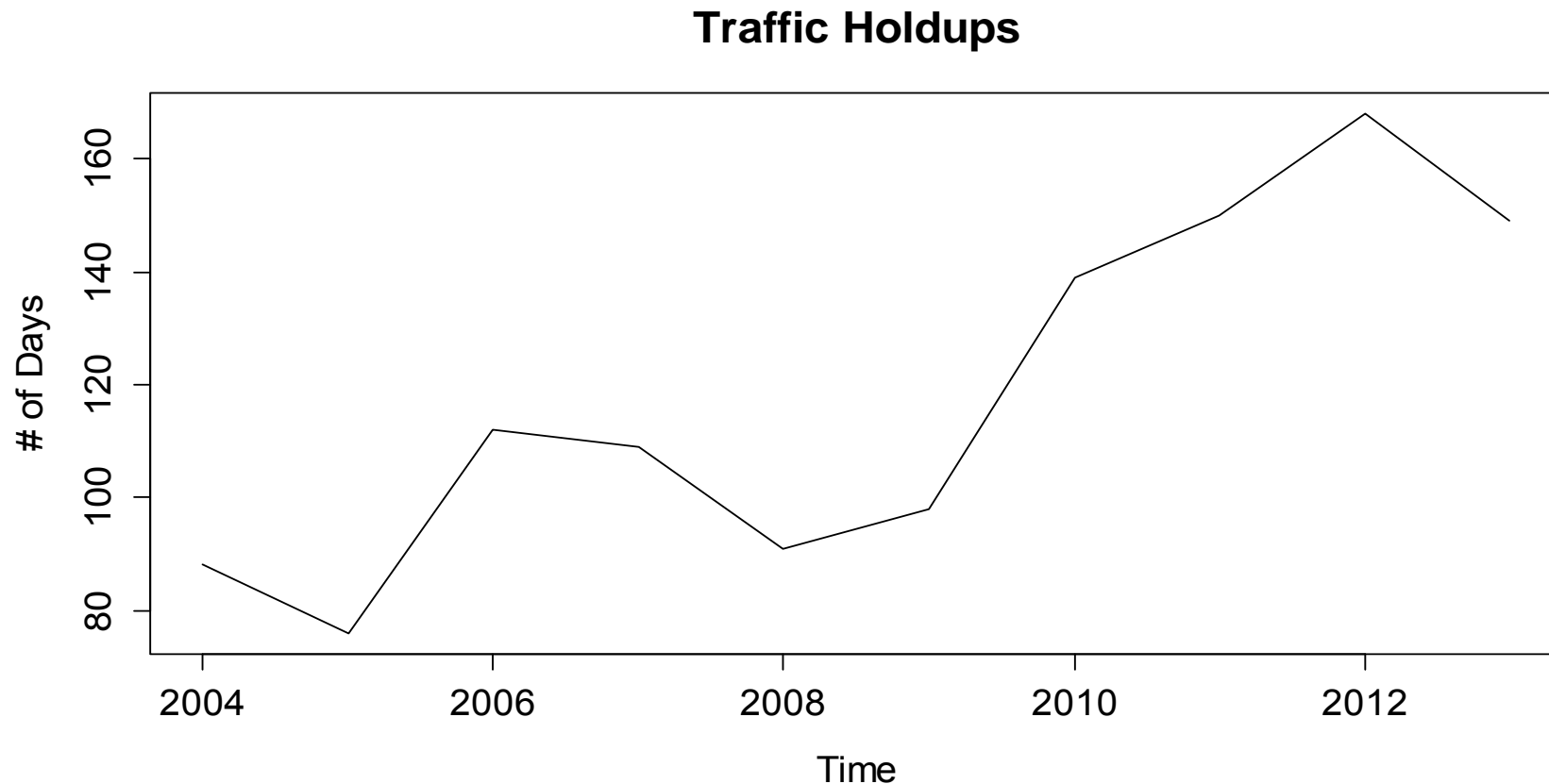
```
> ts.dat
Time Series: Start = 2004
End = 2013; Frequency = 1
[1] 88 76 112 109 91 98 139 150 168 149
```


Applied Time Series Analysis

SS 2016 – Time Series in R

Time Series in R: Example

```
> plot(ts.dat, ylab="# of Days", main="Traffic Holdups")
```



Applied Time Series Analysis

SS 2016 – Time Series in R

Further Topics in R

The scriptum discusses some further topics which are of interest when doing time series analysis in R:

- *Handling of dates and times in R*
 - *Reading/Importing data into R*
- **Please thoroughly read and study these chapters.
Examples will be shown/discussed in the exercises.**

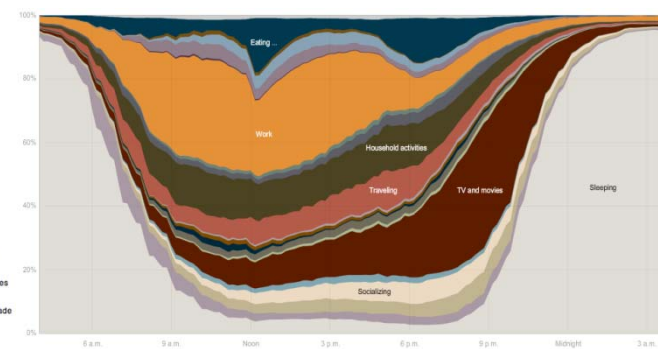
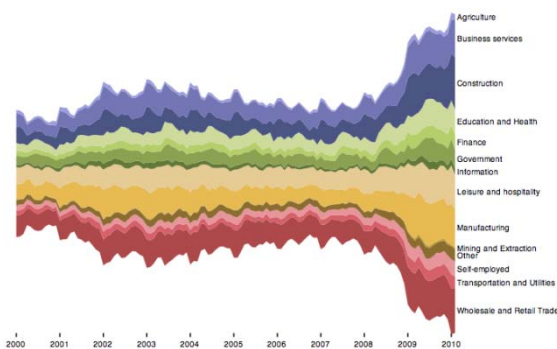
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Descriptive Analysis

As always, when working with data, it is important to first gain an overview. In time series analysis, the following is required:

- *Understanding the context of the data and the data source*
- *Making suitable plots, looking for structure and outliers*
- *Thinking about transformations, e.g. to reduce skewness*
- *Judging stationarity and achieve it by decomposition*
- *For stationary series, the analysis of autocorrelations*

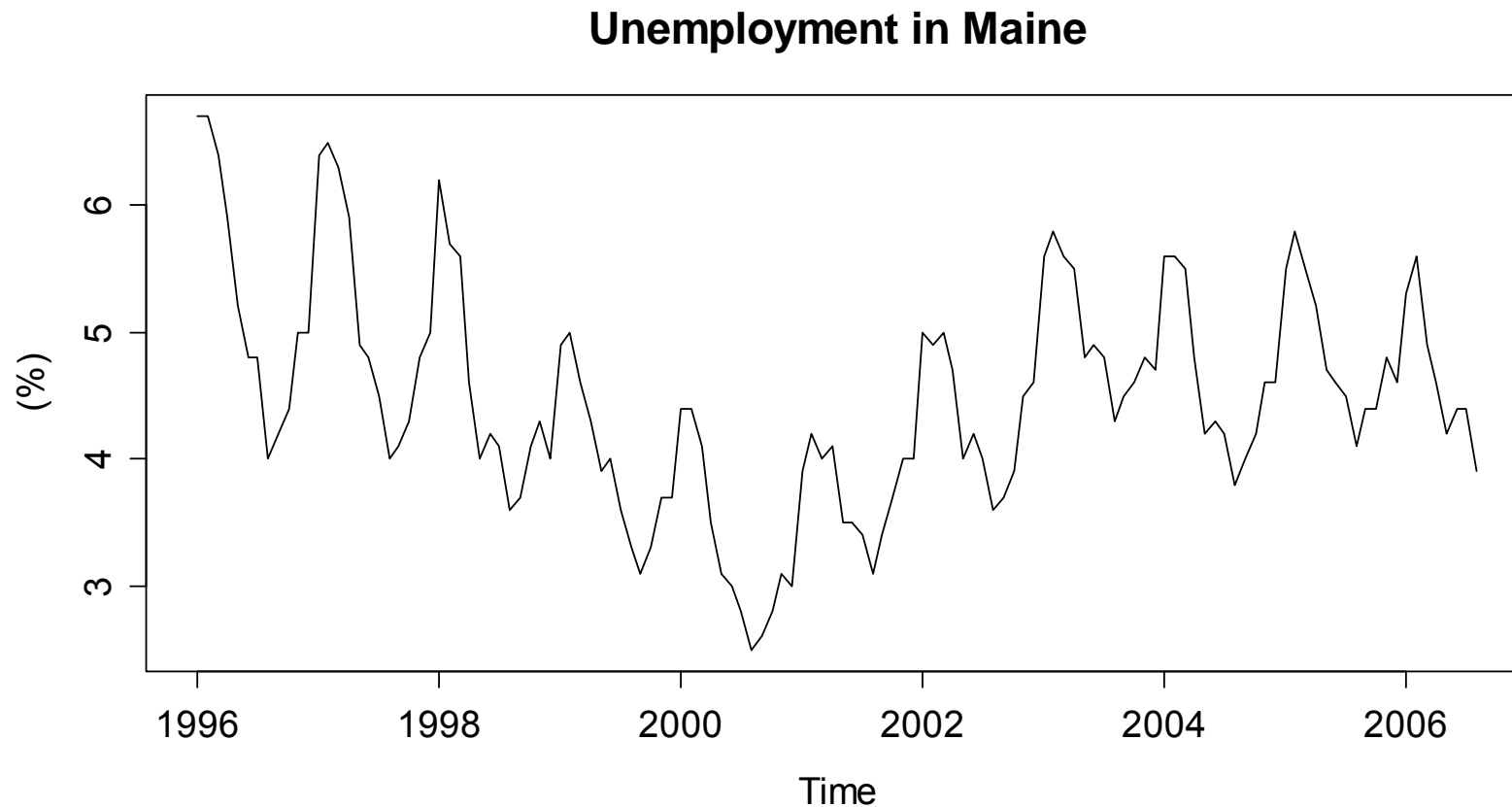


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Visualization: Time Series Plot

```
> plot(tsd, ylab="(%)", main="Unemployment in Maine")
```



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Do's and Dont's with Time Series Plots

- For easier reading, the data points are always joined by lines. This is despite the data are discrete, not continuous!
- An exception to this rule is made when data are missing. Then, the data points shall not be joined by lines.
- Choosing the correct aspect ratio for a time series plot is an art. As a rule of the thumb, use the paradigm of “banking to 45 degrees”.
- For time series with very many observations, it may be required to split them up to several frames. One can for example use `longtsPlot()` from `library(IDPmisc)` or other R functions.

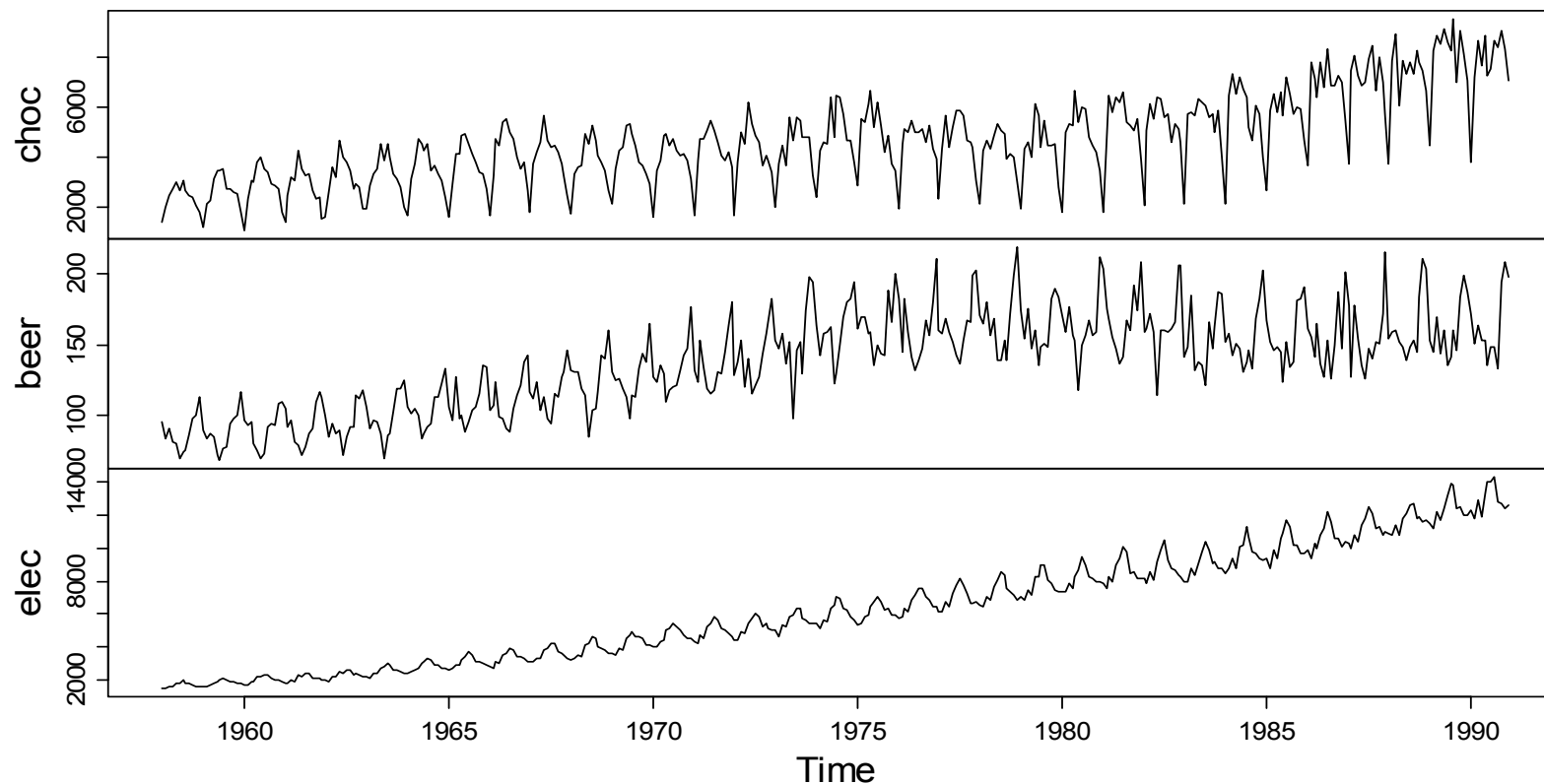
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Multiple Time Series Plots

```
> plot(tsd, main="Chocolate, Beer & Electricity")
```

Chocolate, Beer & Electricity



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Only One or Multiple Frames?

- Due to different scale/units it is often impossible to directly plot multiple time series in one single frame. Also, multiple frames are convenient for visualizing the series.
- If the relative development of multiple series is of interest, then we can (manually) index the series and (manually) plot them into one single frame.
- This clearly shows the magnitudes for both the trend and the seasonality. However, the original units are lost.
- For details on how indexing is done, see the script. One basically needs to divide every observation by the first.

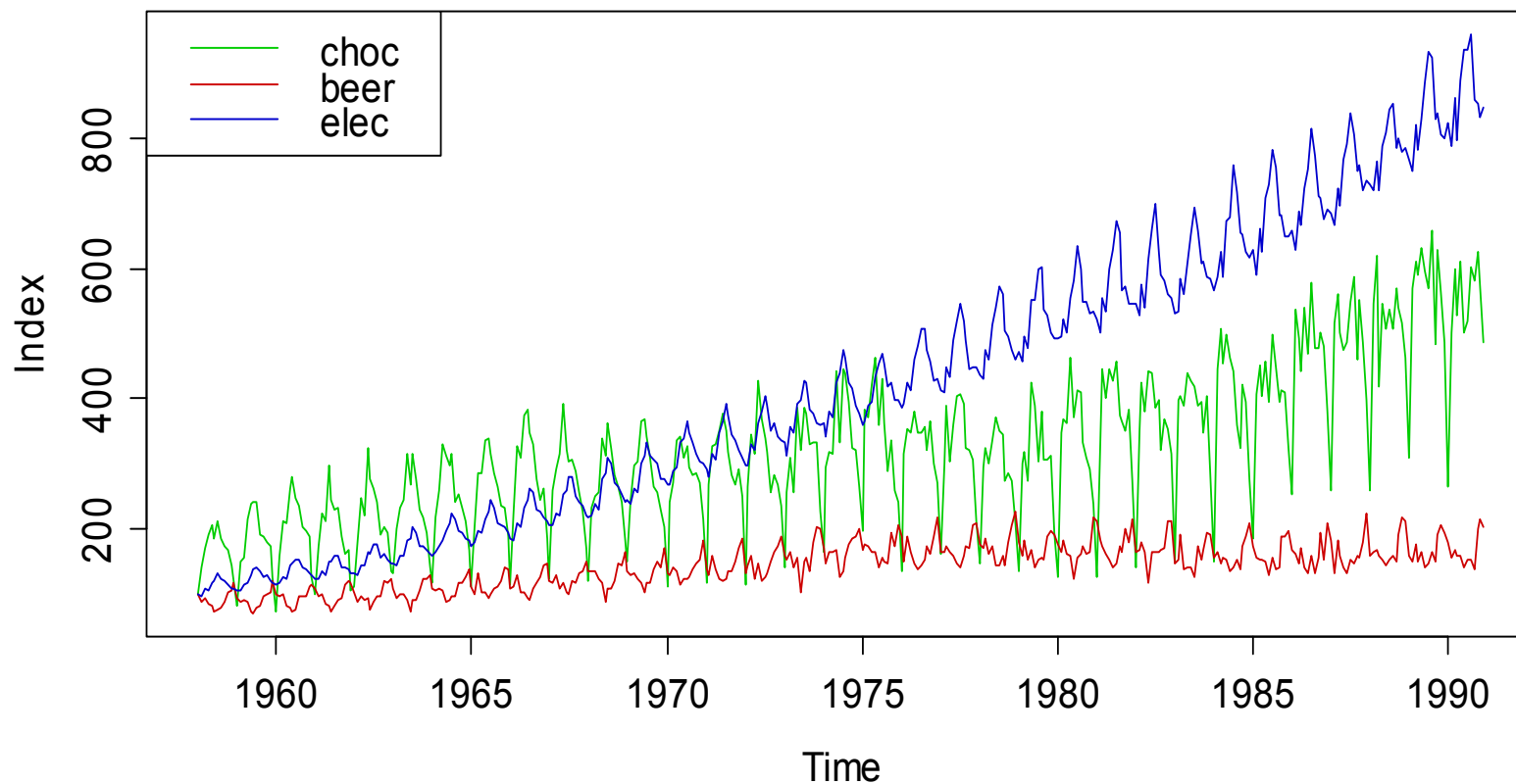
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Multiple Time Series Plots

```
> plot(tsd.ind, plot.type="single", col=2:4, ...)
```

Indexed Chocolate, Beer & Electricity



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Transformations

For strictly stationary time series, we have: $X_t \sim F$

We did not specify the distribution F and there is no restriction to it. However, many popular time series models are based on:

- 1) *Gaussian distribution*
- 2) *linear relations between the variables*

If the data show different behaviour, we can often improve the situation by transforming x_1, \dots, x_n to $g(x_1), \dots, g(x_n)$. The most popular and practically relevant transformation is:

$$g(\cdot) = \log(\cdot)$$

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

When to Apply the Log-Transformation?

As we argued above, a log-transformation of the data often facilitates estimation, fitting and interpretation. When is it indicated to log-transform the data?

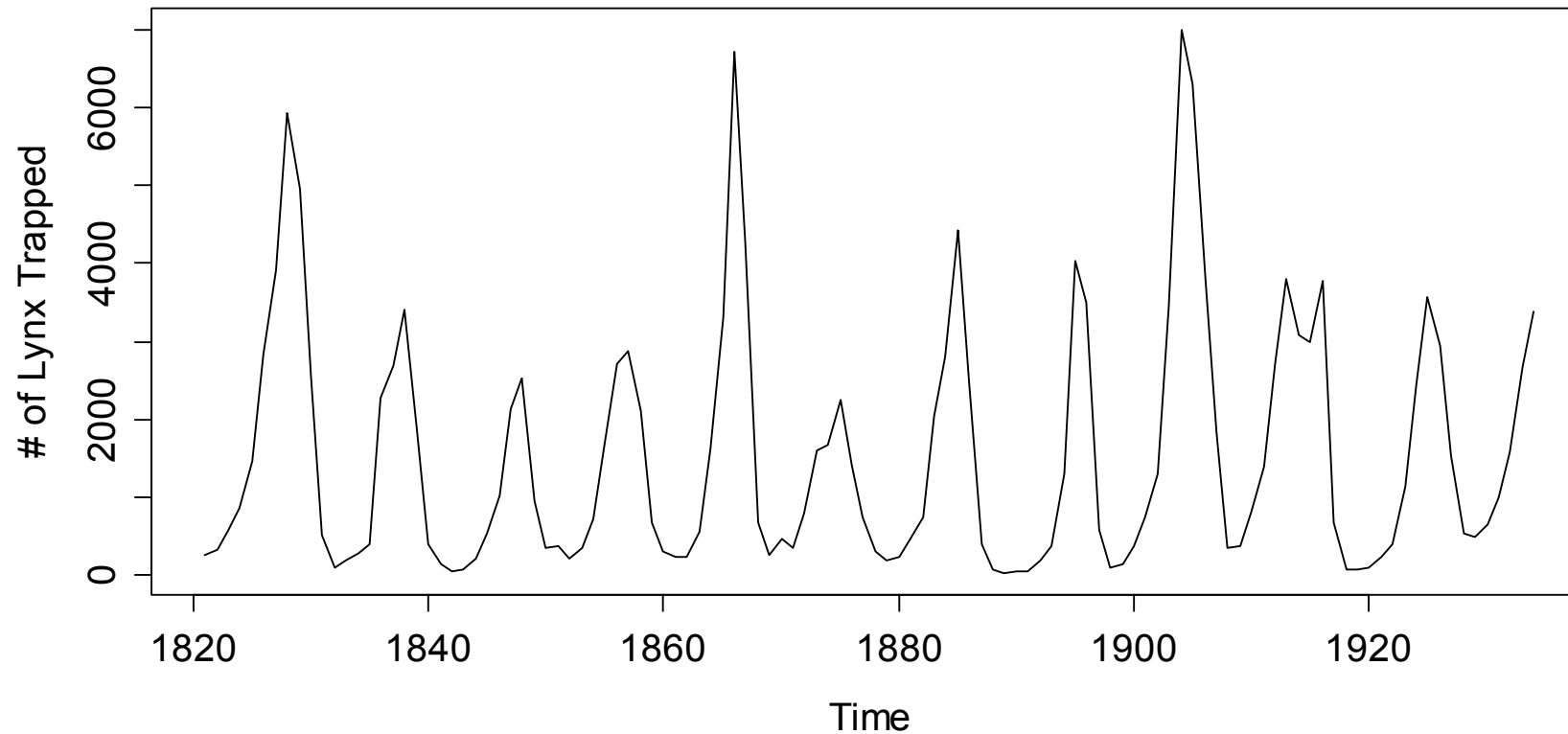
- If the time series is on a scale which is left closed (with value 0) and right open.
- If the marginal distribution of the time series (i.e. when analyzed with a histogram) is right-skewed.
- If the time series is on a relative scale, i.e. where an absolute increment changes its meaning with the level of the series.

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Transformations: Lynx Data

Lynx Trappings

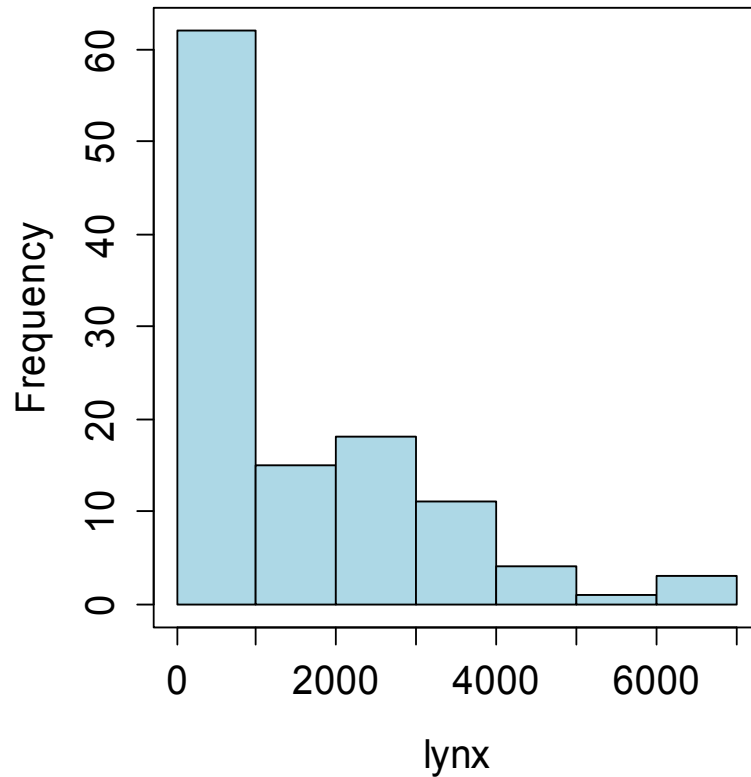


Applied Time Series Analysis

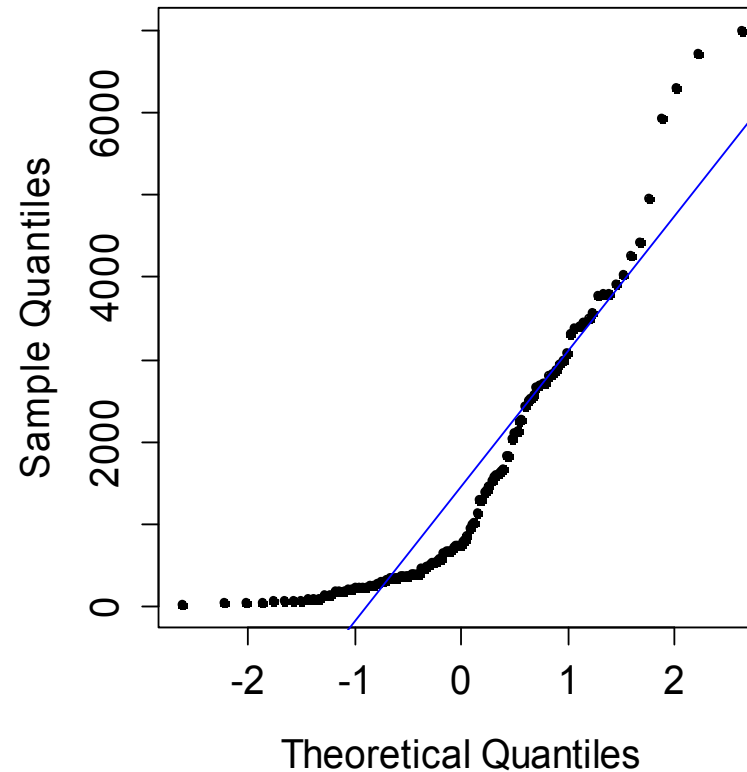
SS 2016 – Descriptive Analysis

Transformations: Lynx Data

Histogram of lynx



Normal Q-Q Plot

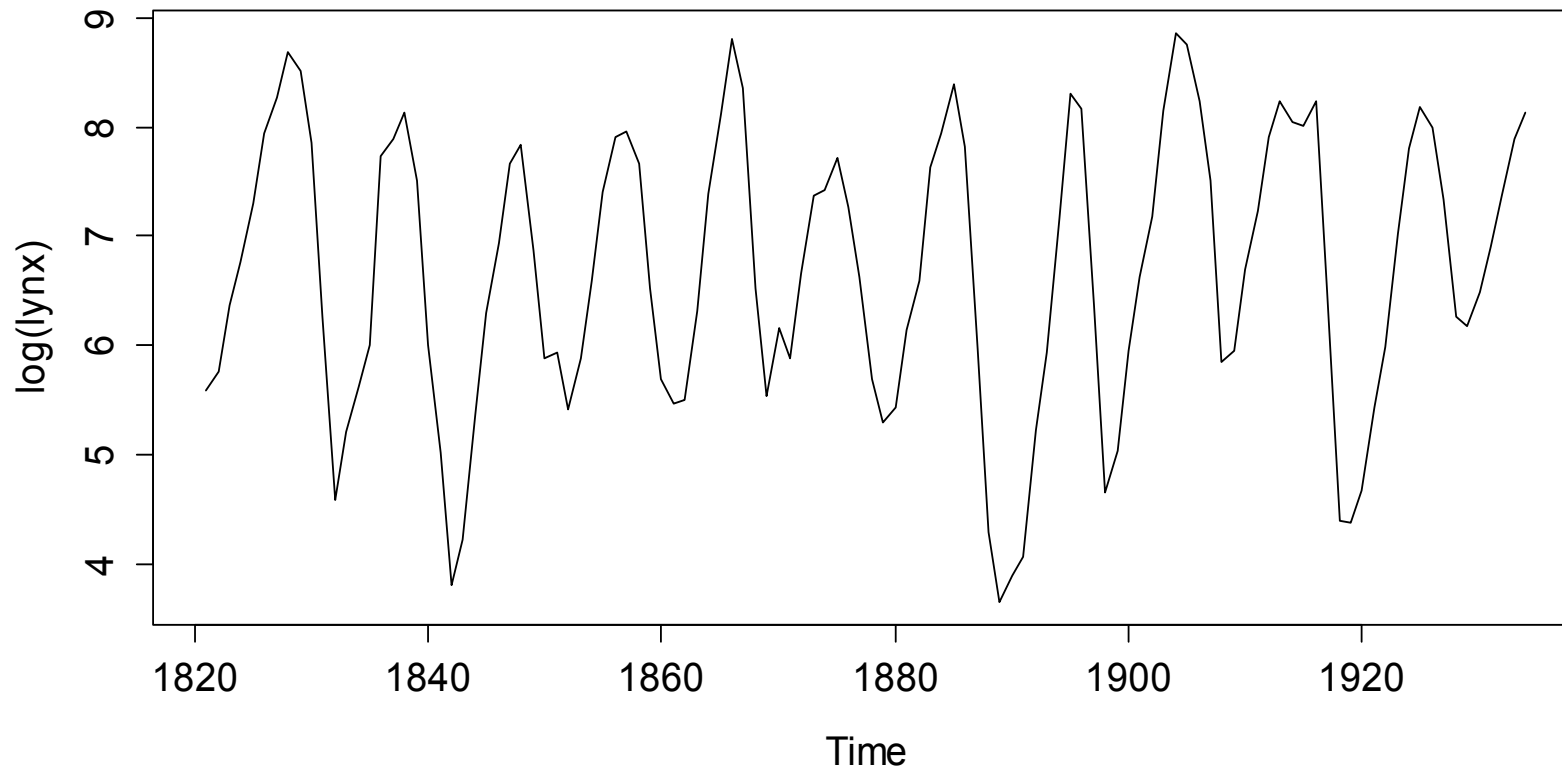


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Transformations: Lynx Data

Logged Lynx Trappings



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Decomposition

Stationarity is key for statistical learning, but real data often have trend/seasonality, and are non-stationary. We can (often) deal with that using the simple additive decomposition model:

$$X_t = m_t + s_t + R_t$$

= trend + seasonal effect + stationary remainder

The goal is to find a remainder term R_t , as a sequence of correlated random variables with mean zero, i.e. a stationary ts.

We can employ:

- 1) *taking differences (=differencing)*
- 2) *smoothing approaches (= filtering)*
- 3) *parametric models (= curve fitting)*

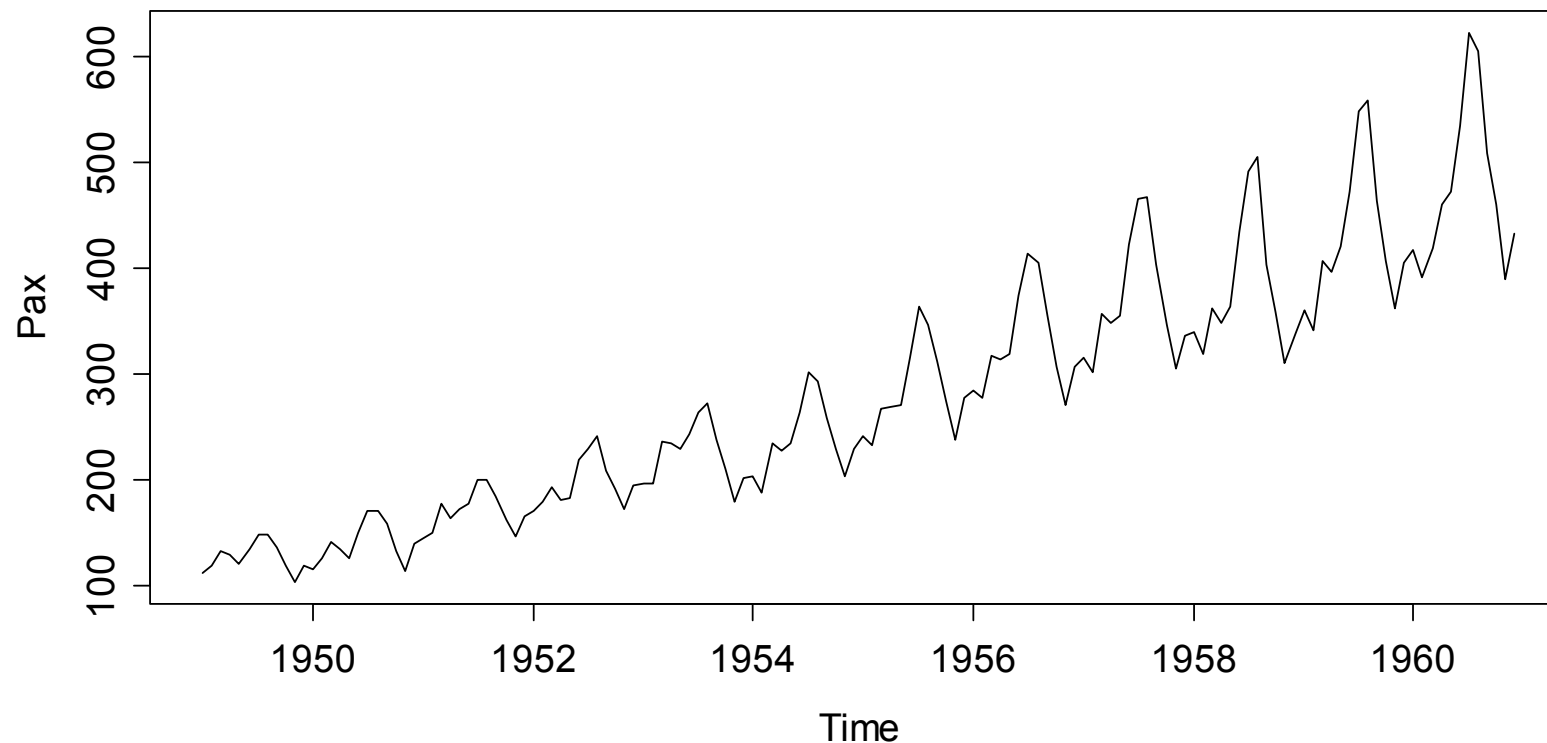
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Multiplicative Decomposition

$X_t = m_t + s_t + R_t$ is not always a good model:

Passenger Bookings



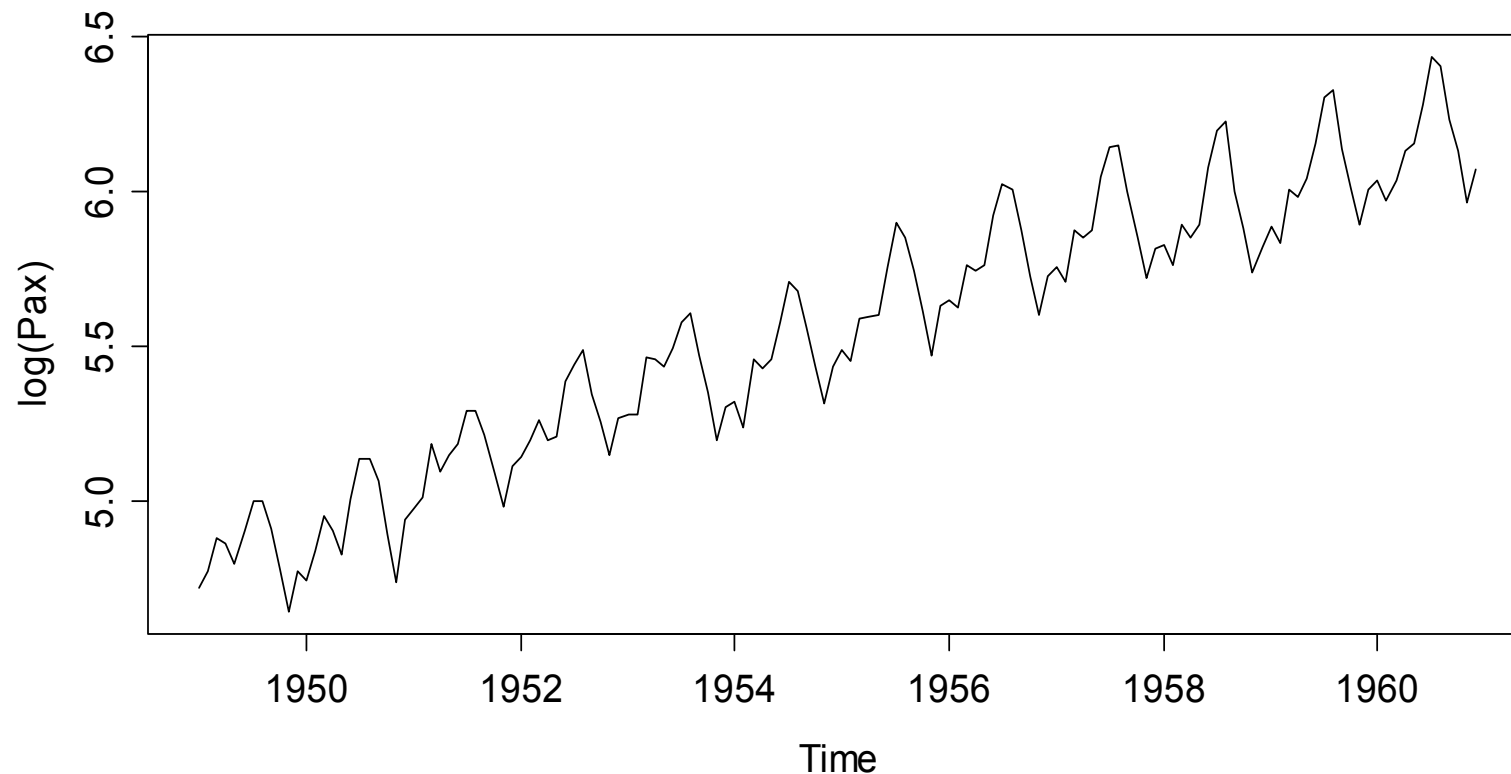
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Multiplicative Decomposition

Better: $X_t = m_t \cdot s_t \cdot R_t$, respectively $\log(X_t) = m'_t + s'_t + R'_t$

Logged Passenger Bookings



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing: Removing a Trend

We assume a series with an additive trend, but no seasonal variation. We can write: $X_t = m_t + R_t$. If we perform differencing and assume a slowly-varying trend with $m_t \approx m_{t-1}$, we obtain:

$$Y_t = X_t - X_{t-1} \approx R_t - R_{t-1}$$

- Note that Y_t are the observation-to-observation changes in the series, but no longer the observations or the remainder.
- This may (or may not) remove trend/seasonality, but does not yield estimates for m_t and s_t , and not even for R_t .
- For a slow, curvy trend, the mean is zero: $E[Y_t] = 0$

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Peculiarities in Differencing

In the (practically rare) case where a series with a perfect linear trend $m_t = \alpha + \beta t$ is differenced, the trend will be reduced to an additive constant:

$$Y_t = X_t - X_{t-1} = \beta + R_t - R_{t-1}$$

It is important to know that differencing creates artificial new dependencies that are different from the original ones. For illustration, consider a stochastically independent remainder:

$$\begin{aligned} \text{Cov}(Y_t, Y_{t-1}) &\approx \text{Cov}(R_t - R_{t-1}, R_{t-1} - R_{t-2}) \\ &= -\text{Cov}(R_{t-1}, R_{t-1}) \\ &\neq 0 \end{aligned}$$

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing: Example

Swiss Traffic Index



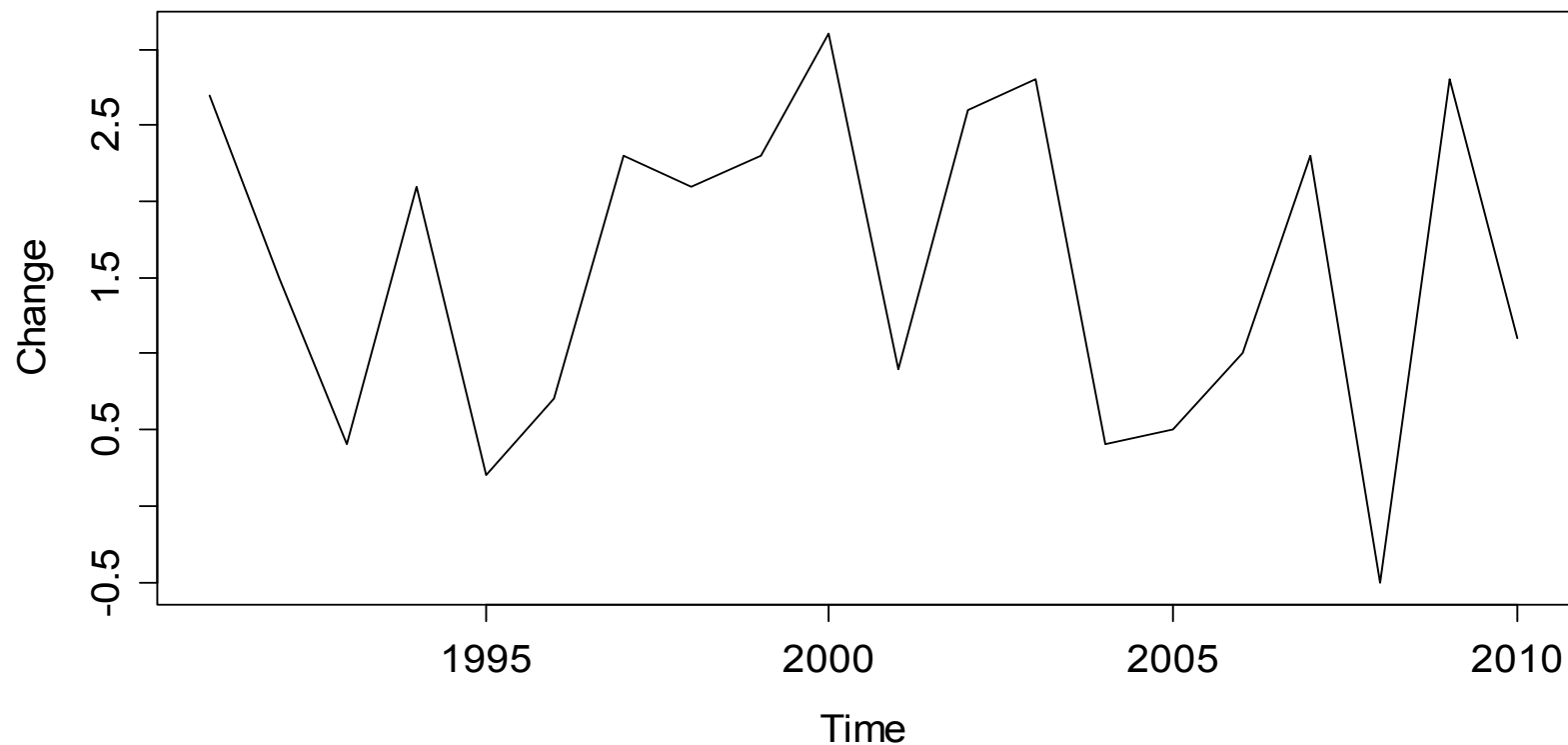
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing: Example

```
> plot(diff(SwissTraffic), main=...)
```

Differenced Swiss Traffic Index



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing: Further Remarks

- If log-transformed series are difference (i.e. the SMI series), we are considering (an approximation to) the relative changes:

$$Y_t = \log(X_t) - \log(X_{t-1}) = \log\left(\frac{X_t}{X_{t-1}}\right) = \log\left(\frac{X_t - X_{t-1}}{X_{t-1}} + 1\right) \approx \frac{X_t - X_{t-1}}{X_{t-1}}$$

- The backshift operator “go back 1 step” allows for convenient notation with all differencing operations:

Backshift operator: $B(X_t) = X_{t-1}$

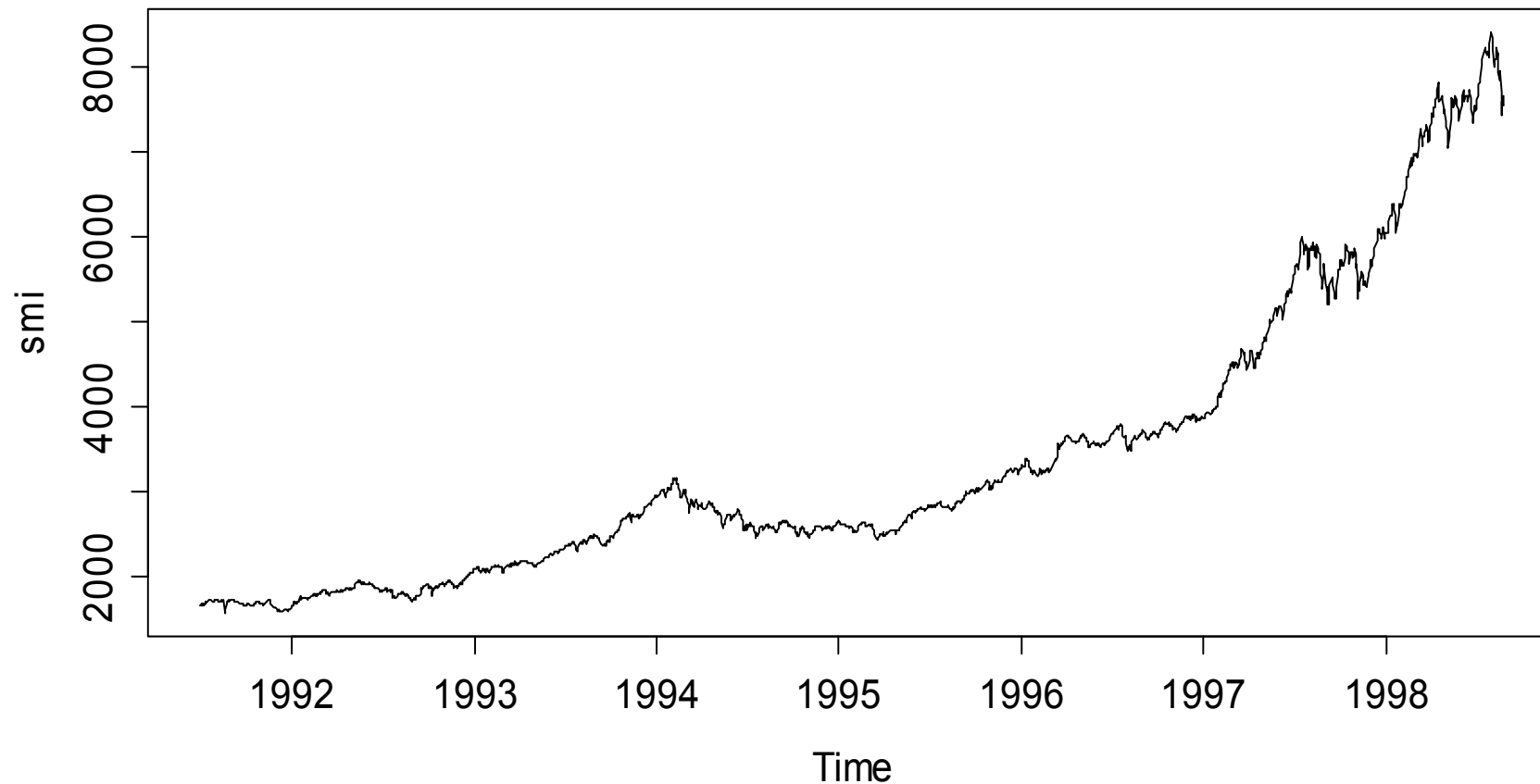
Differencing: $Y_t = (1 - B)X_t = X_t - X_{t-1}$

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing Series with Transformation

SMI Daily Closing Value

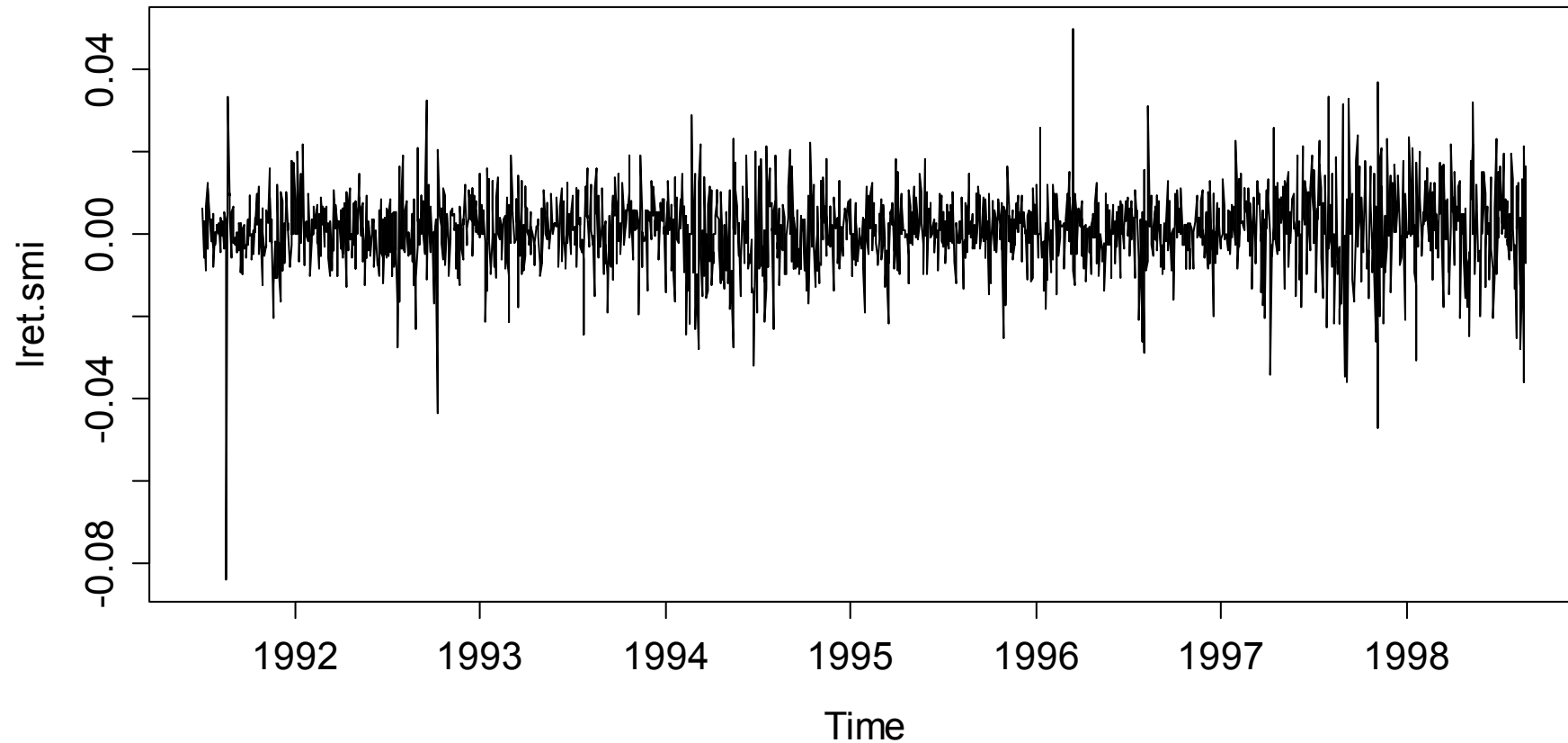


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing Series with Transformation

SMI Log>Returns



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Higher-Order Differencing

The “normal” differencing from above managed to remove any linear trend from the data. In case of polynomial trend, that is no longer true. But we can take higher-order differences:

$$\begin{aligned}X_t &= \alpha + \beta_1 t + \beta_2 t^2 + R_t, \quad R_t \text{ stationary} \\Y_t &= (1-B)^2 X_t \\&= (X_t - X_{t-1}) - (X_{t-1} - X_{t-2}) \\&= R_t - 2R_{t-1} + R_{t-2} + 2\beta_2\end{aligned}$$

A quadratic trend can be removed by taking second-order differences. However, what we obtain is not an estimate of the remainder term R_t , but something that is much more complicated.

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Removing Seasonal Effects

Time series with seasonal effects can be made stationary through differencing by comparing to the previous periods' value.

$$Y_t = (1 - B^p)X_t = X_t - X_{t-p}$$

- Here, p is the frequency of the series.
- A potential trend which is exactly linear will be removed by the above form of seasonal differencing.
- In practice, trends are rarely linear but slowly varying: $m_t \approx m_{t-1}$
However, here we compare m_t with m_{t-p} , which means that seasonal differencing often fails to remove trends completely.

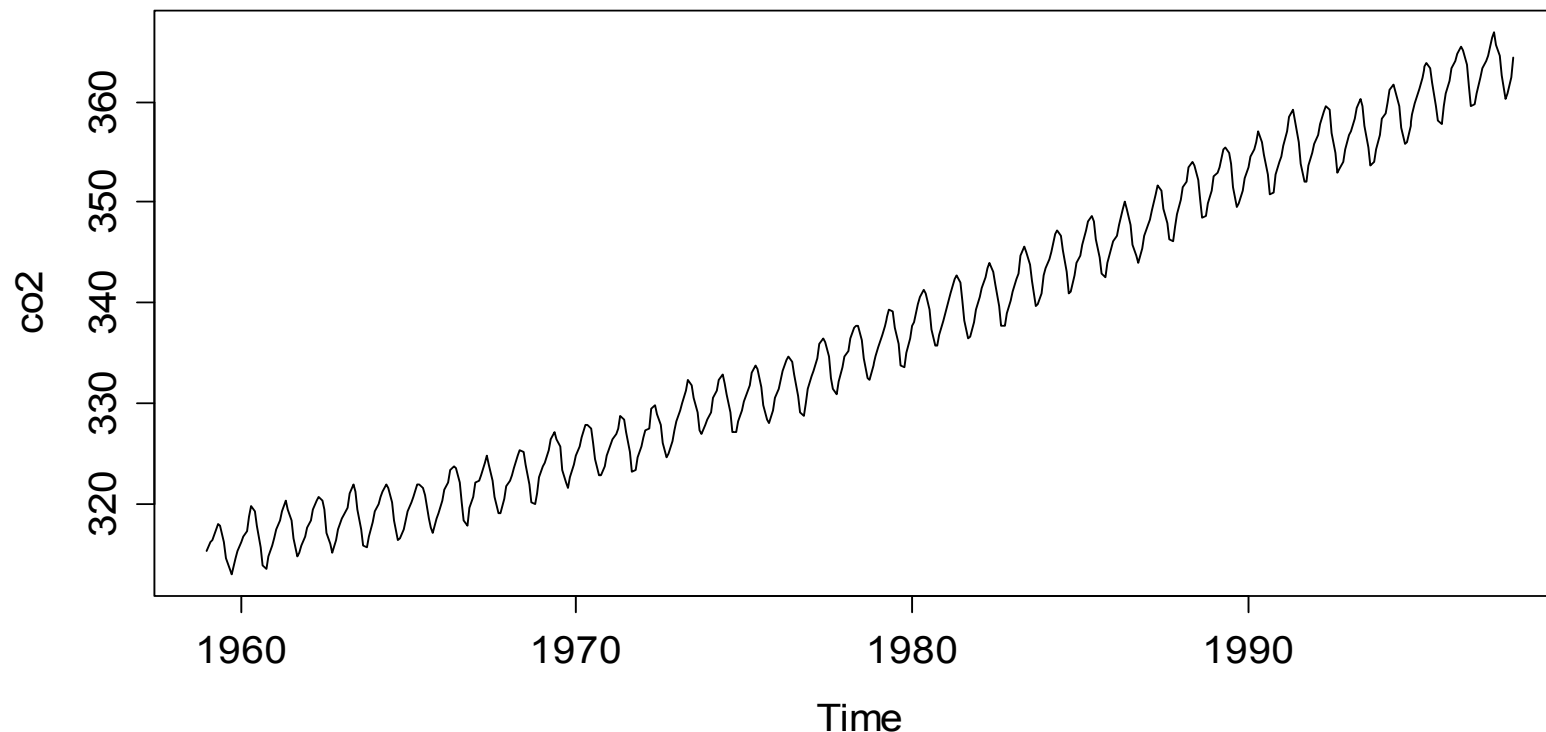
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Seasonal Differencing: Example

```
> data(co2); plot(co2, main=...)
```

Mauna Loa CO2 Concentrations



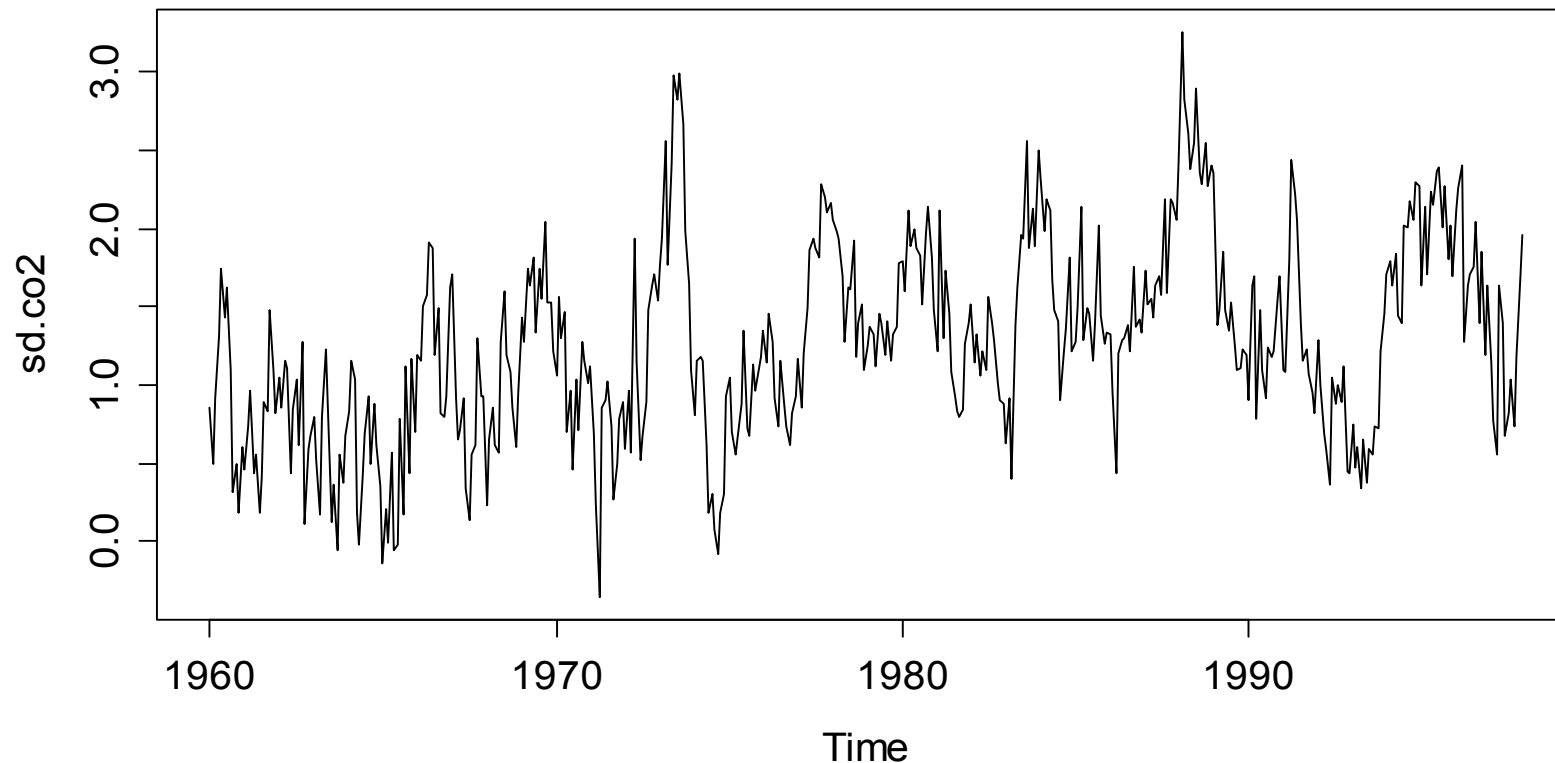
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Seasonal Differencing: Example

```
> sd.co2 <- diff(co2, lag=12)
```

Differenced Mauna Loa Data (p=12)



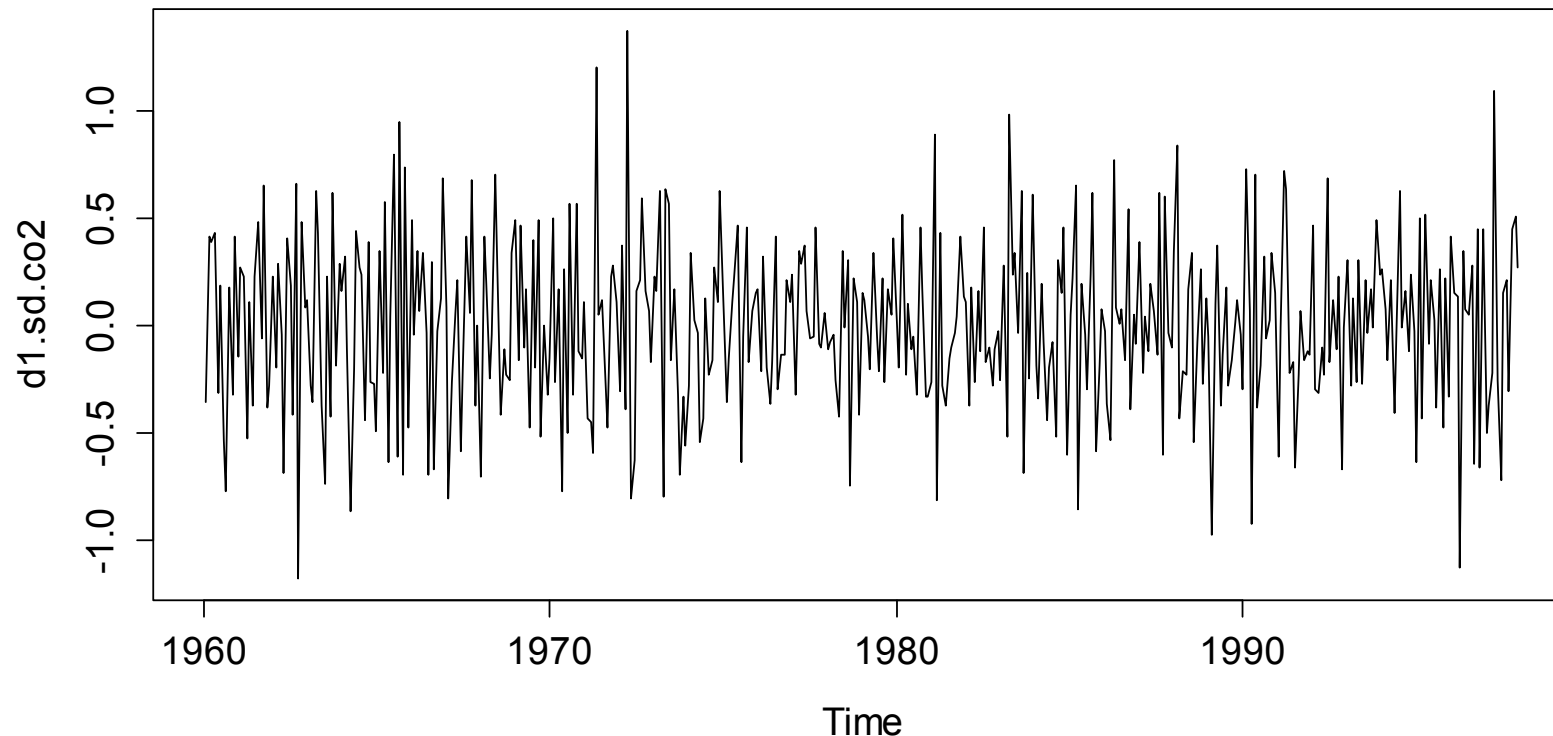
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Seasonal Differencing: Example

This is: $Z_t = (1 - B)Y_t = (1 - B)(1 - B^{12})X_t$

Twice Differenced Mauna Loa Data (p=12, p=1)



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Differencing: Summary

Some advantages and disadvantages:

- + trend and seasonal effect can be removed
- + procedure is very quick and very simple to implement

- \hat{m}_t , \hat{s}_t and \hat{R}_t are not known, and cannot be visualised
- resulting time series will be shorter than the original
- differencing leads to strong artificial dependencies
- extrapolation of \hat{m}_t , \hat{s}_t is not easily possible

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Smoothing, Filtering: Part 1

In the **absence of a seasonal effect**, the trend of a non-stationary time series can be determined by applying any **additive, linear filter**. We obtain a new time series \hat{m}_t , representing the trend:

$$\hat{m}_t = \sum_{i=-p}^q a_i X_{t+i}$$

- the window, defined by p and q , can or can't be symmetric.
- the weights, given by a_i , can or can't be uniformly distributed.
- most popular is to rely on $p = q$ and $a_i = 1 / (2p + 1)$.
- other smoothing procedures can be applied, too.

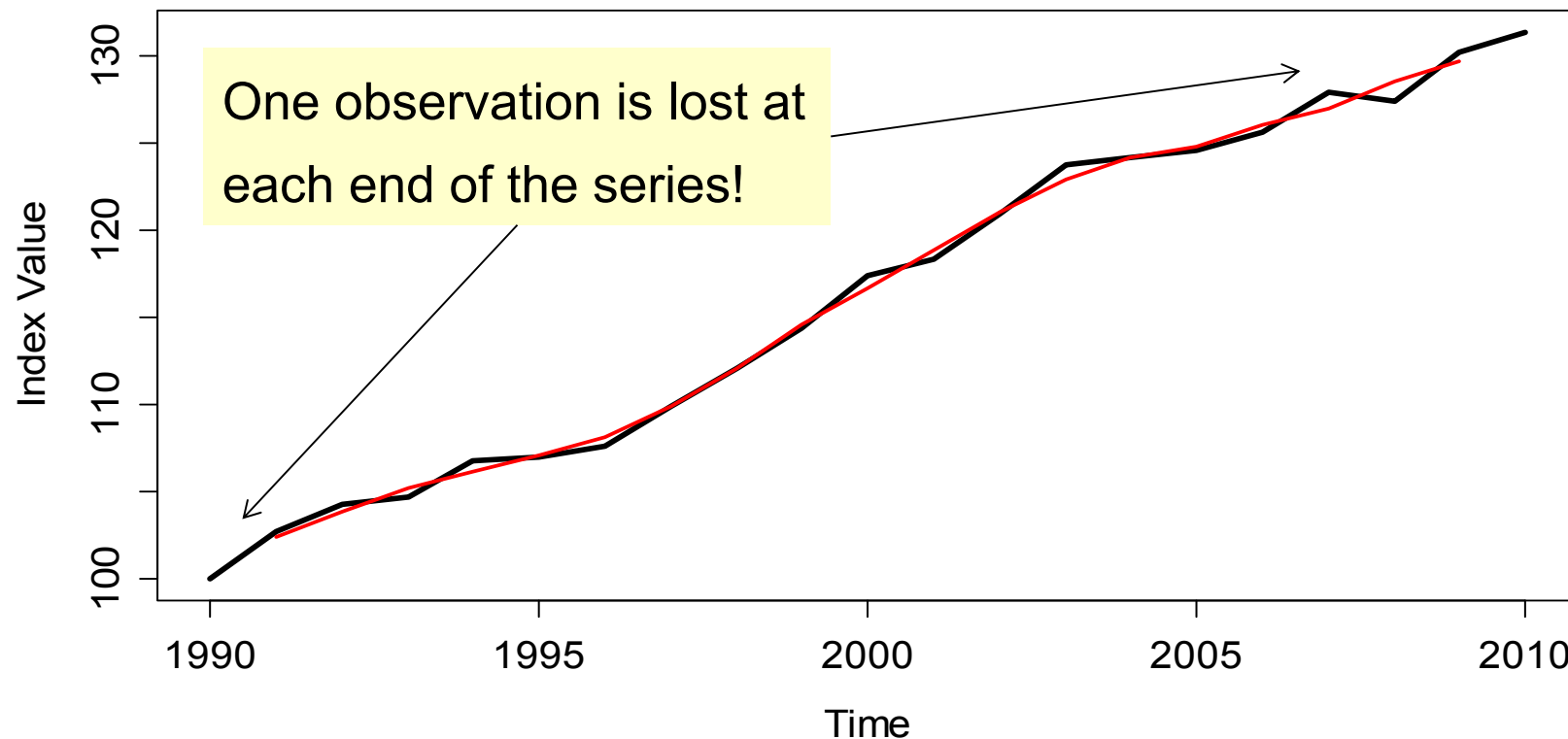
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Trend Estimation with the Running Mean

```
> trd <- filter(SwissTraffic, filter=c(1,1,1)/3)
```

Swiss Traffic Index with Running Mean

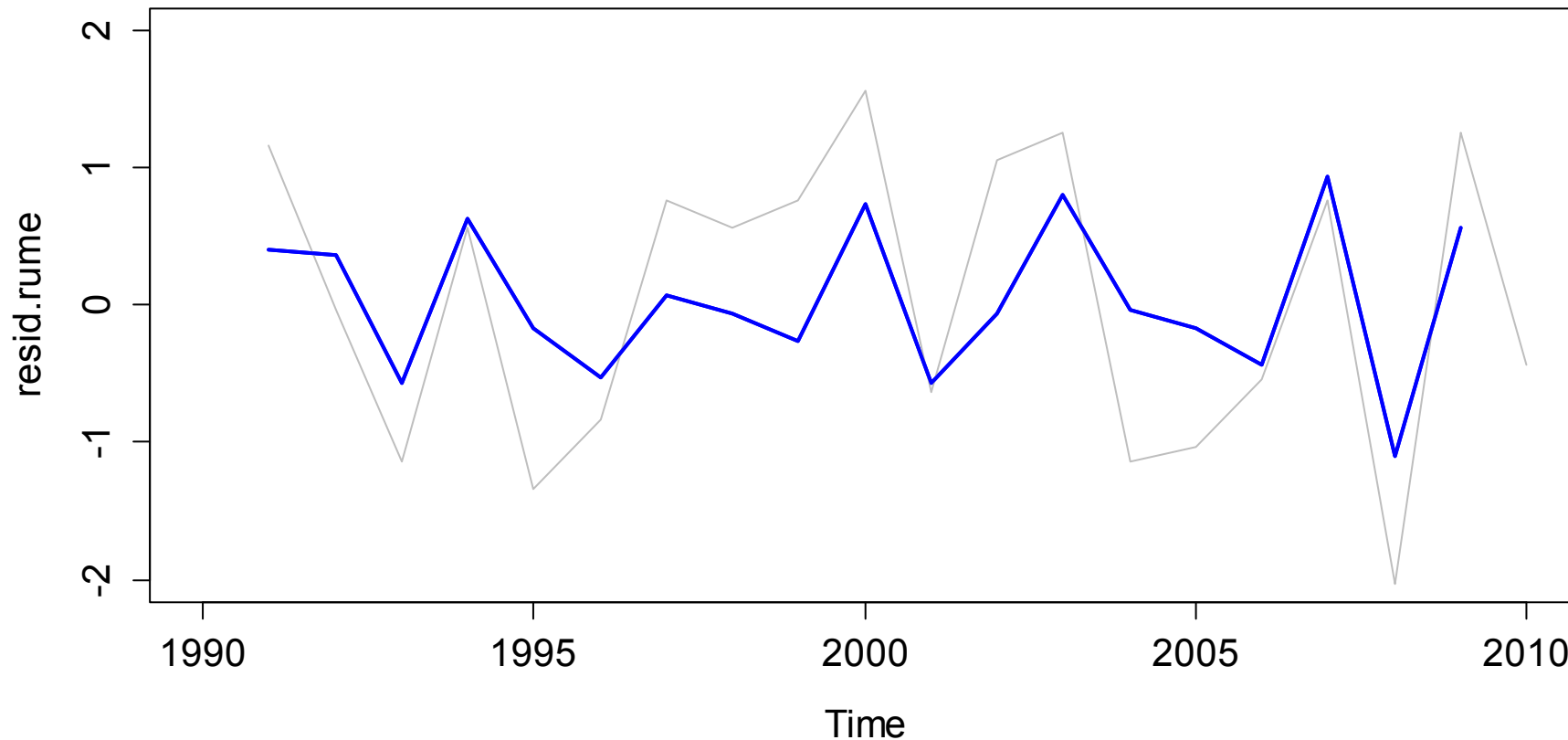


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Filtering and Differencing: Not the Same!

Estimated Stochastic Remainder Term



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Smoothing, Filtering: Part 2

In the presence a seasonal effect, smoothing approaches are still valid for estimating the trend. We have to make sure that the sum is taken over an entire season, i.e. for monthly data:

$$\hat{m}_t = \frac{1}{12} \left(\frac{1}{2} X_{t-6} + X_{t-5} + \dots + X_{t+5} + \frac{1}{2} X_{t+6} \right) \text{ for } t = 7, \dots, n-6$$

An estimate of the seasonal effect s_t at time t can be obtained by:

$$\hat{s}_t = x_t - \hat{m}_t$$

By averaging these estimates of the effects for each month, we obtain a single estimate of the effect for each month.

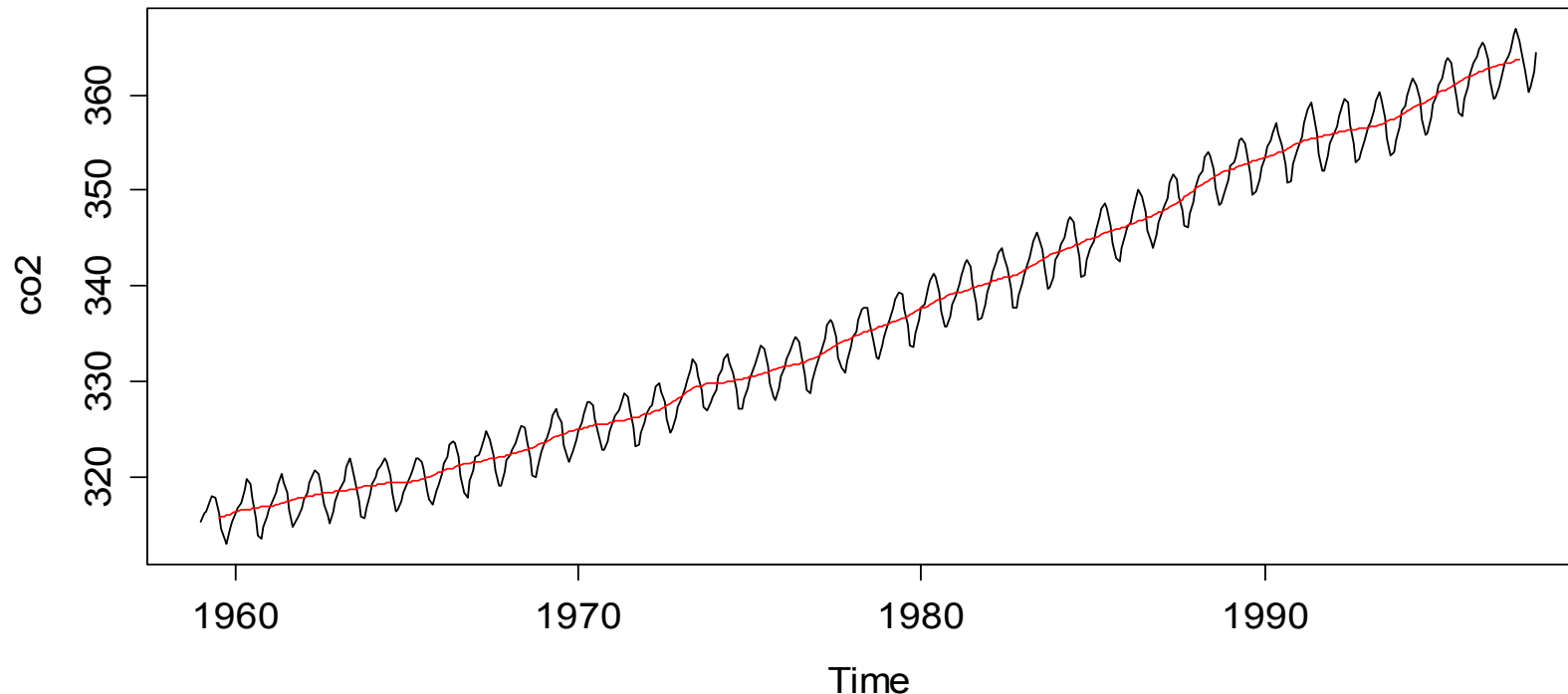
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Trend Estimation for Mauna Loa Data

```
> wghts <- c(.5,rep(1,11),.5)/12  
> trd <- filter(co2, filter=wghts, sides=2)
```

Mauna Loa CO2 Concentrations



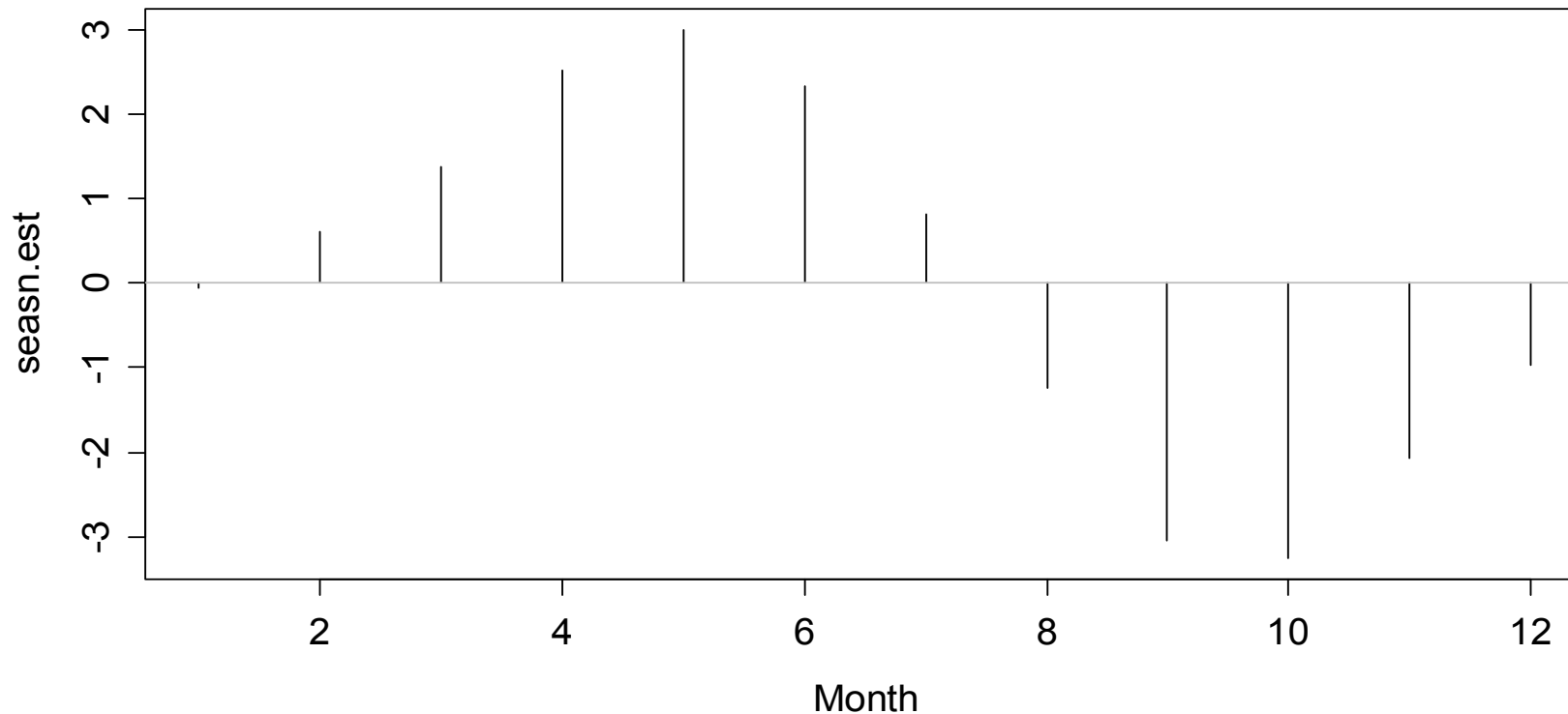
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Estimating the Seasonal Effects

$$\hat{s}_{Jan} = \hat{s}_1 = \hat{s}_{13} = \dots = \frac{1}{39} \cdot \sum_{j=0}^{38} (x_{12j+1} - \hat{m}_{12j+1})$$

Seasonal Effects for Mauna Loa Data



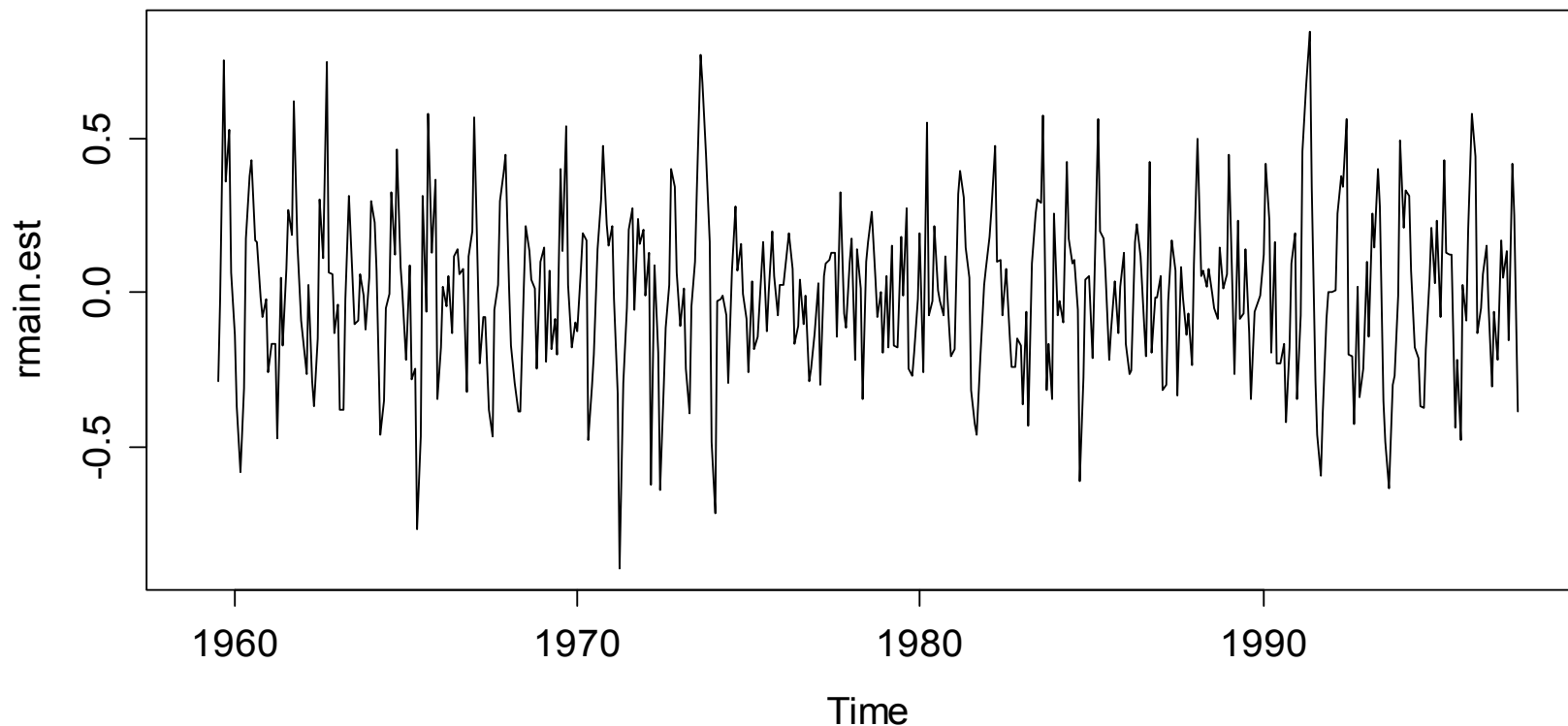
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Estimating the Remainder Term

$$\hat{R}_t = x_t - \hat{m}_t - \hat{s}_t$$

Estimated Stochastic Remainder Term



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Smoothing, Filtering: Part 3

- The smoothing approach is based on estimating the trend first, and then the seasonality after removal of the trend.
- The generalization to other periods than $p = 12$, i.e. monthly data is straightforward. Just choose a symmetric window and use uniformly distributed coefficients that sum up to 1.
- The sum over all seasonal effects will often be close to zero. Usually, one centers the seasonal effects to mean zero.
- This procedure is implemented in R with `decompose()`. Note that it only works for seasonal series where at least two full periods were observed!

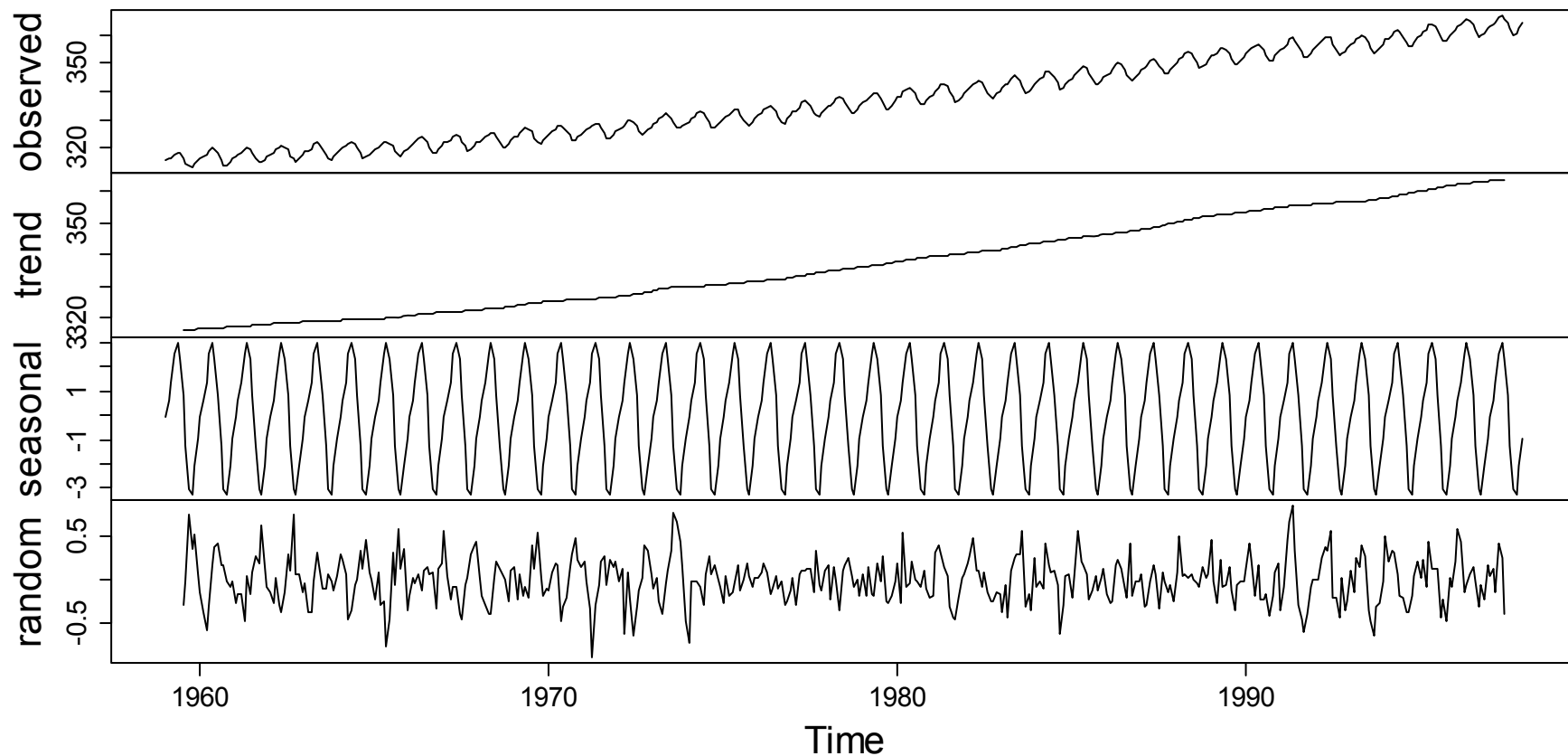
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Estimating the Remainder Term

```
> plot(decompose(co2))
```

Decomposition of additive time series



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Smoothing, Filtering: Remarks

Some advantages and disadvantages:

- + trend and seasonal effect can be estimated
 - + \hat{m}_t , \hat{s}_t and \hat{R}_t are explicitly known & can be visualised
 - + procedure is transparent, and simple to implement
-
- resulting time series will be shorter than the original
 - the running mean is not the very best smoother
 - extrapolation of \hat{m}_t , \hat{s}_t are not entirely obvious
 - seasonal effect is constant over time

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Trend Estimation with Loess

The running mean has some poor properties as a smoother. One prefers to use the the so-called Loess Smoother instead. For a time series with trend, but without seasonality, we recommend:

```
> fit    <- loess(SwissTraffic~time(SwissTraffic))  
> trend <- predict(fit)
```

Main advantages:

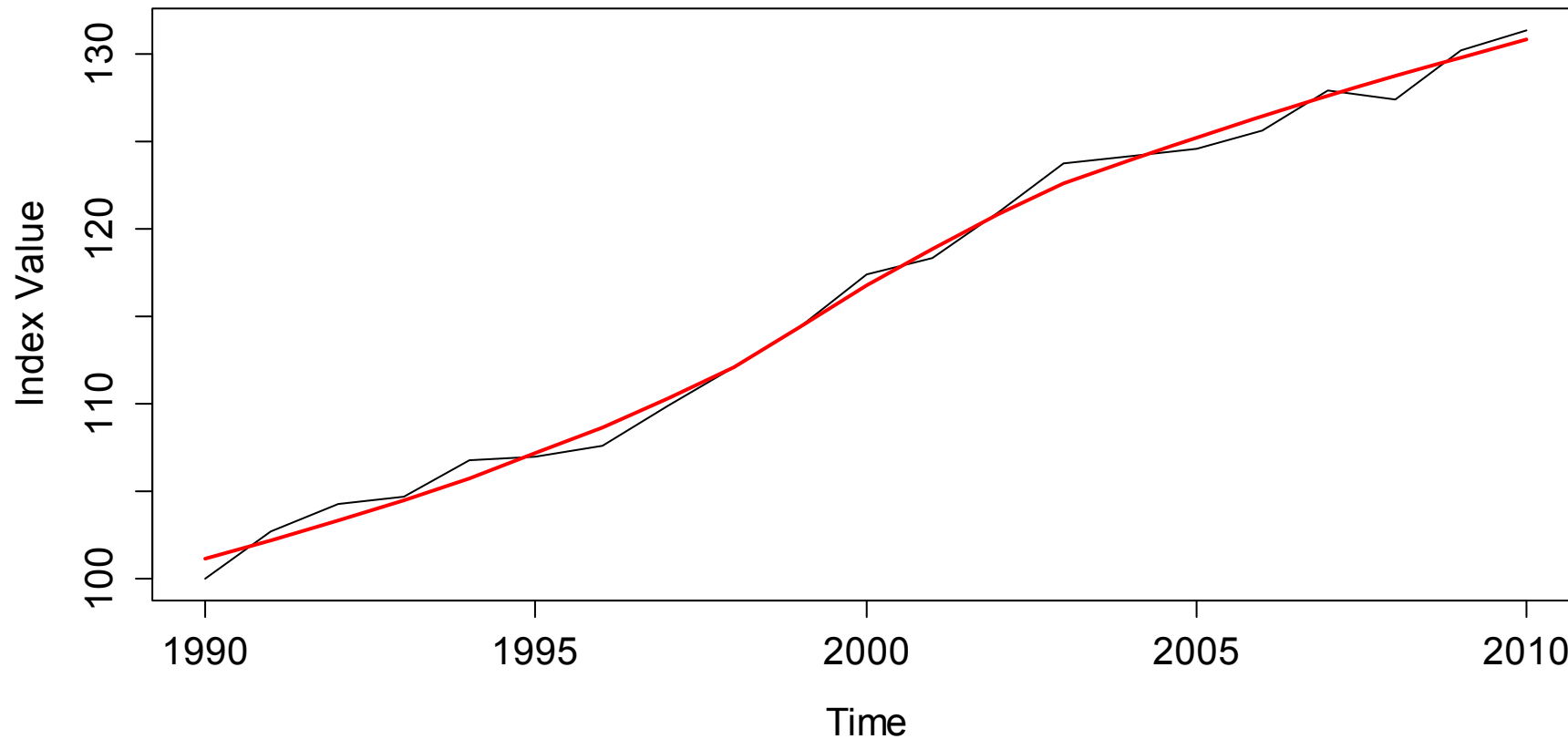
- the smoother has nicer theoretical properties.
- the algorithm is far more robust against outliers.
- it will also produce trend estimates at the boundaries.

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Trend Estimation with Loess

Swiss Traffic Index with Running Mean



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Smoothing, Filtering: STL-Decomposition

The **S**easonal-**T**rend Decomposition Procedure by **L**oess

- is an iterative, non-parametric smoothing algorithm
- yields a simultaneous estimation of trend and seasonal effect
- similar to what was presented above, but more **robust!**

+ very simple to apply

+ very illustrative and quick

+ seasonal effect can be constant or smoothly varying

- model free, extrapolation and forecasting is difficult

→ **Good method for „having a quick look at the data“**

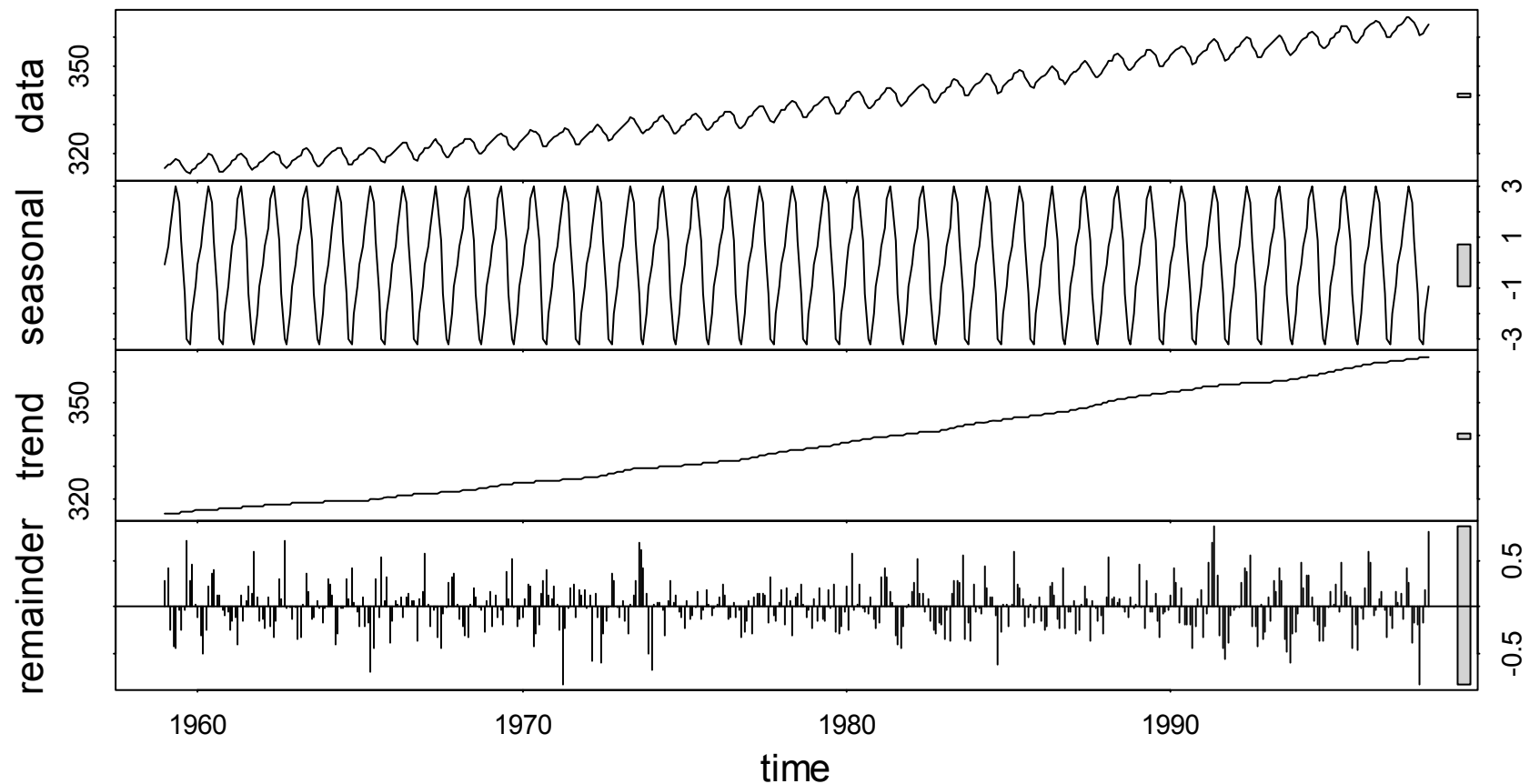
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL-Decomposition for Periodic Series

```
> co2.stl <- stl(co2, s.window="periodic")  
> plot(co2.stl, main="STL-Decomposition of CO2 Data")
```

STL-Decomposition of CO2 Data



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Using the `stl()` *Function in R*

`stl {stats}`

R Documentation

Seasonal Decomposition of Time Series by Loess

Description

Decompose a time series into seasonal, trend and irregular components using `loess`, acronym STL.

Usage

```
stl(x, s.window, s.degree = 0,
    t.window = NULL, t.degree = 1,
    l.window = nextodd(period), l.degree = t.degree,
    s.jump = ceiling(s.window/10),
    t.jump = ceiling(t.window/10),
    l.jump = ceiling(l.window/10),
    robust = FALSE,
    inner = if(robust) 1 else 2,
    outer = if(robust) 15 else 0,
    na.action = na.fail)
```

We need to supply argument `x` (i.e. the data) and `s.window` (for seasonal smoothing), either by setting it to `"periodic"` or to a numerical value. We can adjust `t.window` to a numerical value for altering the trend smoothing. Leave the rest alone!

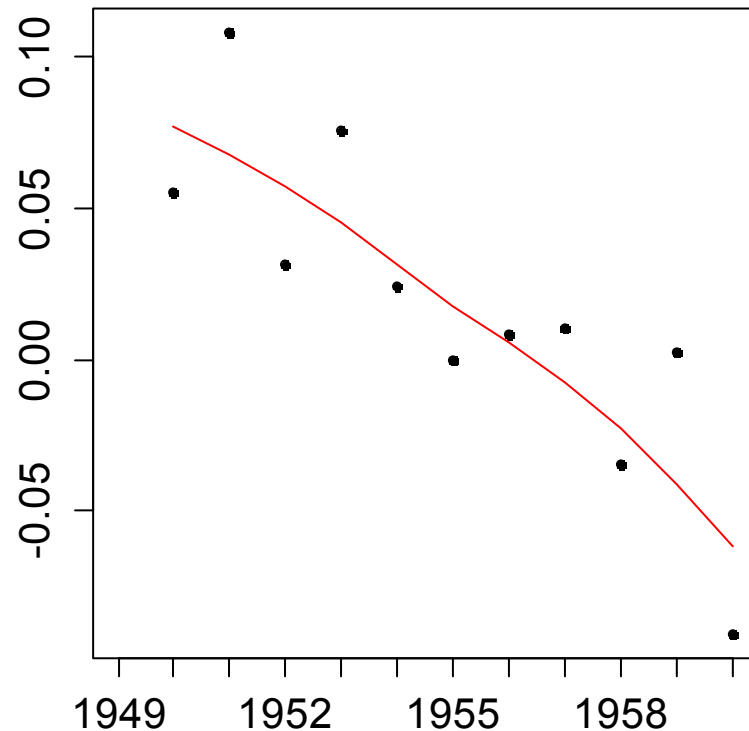
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

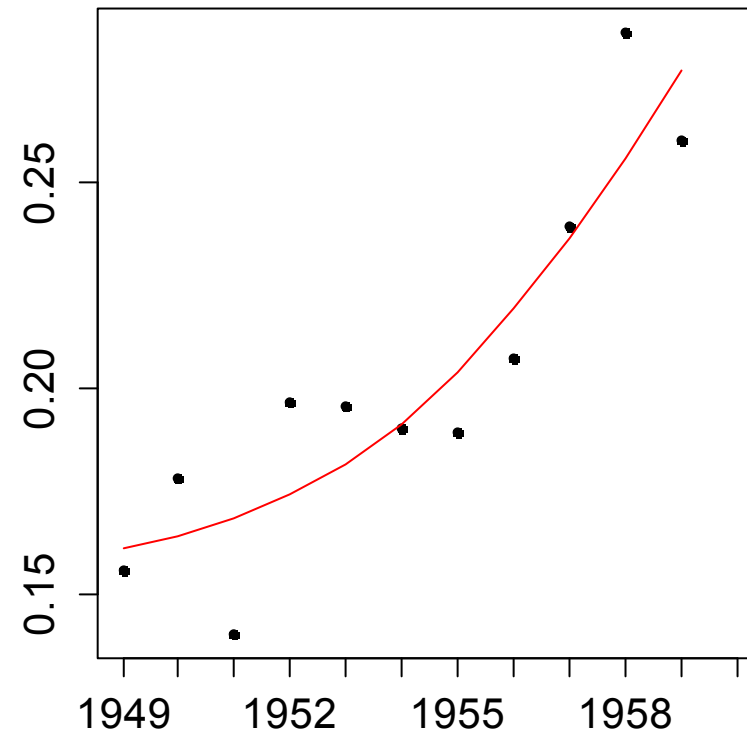
Idea for Evolving Seasonality

March & August observation after trend removal are as follows:

Effect of March



Effect of August

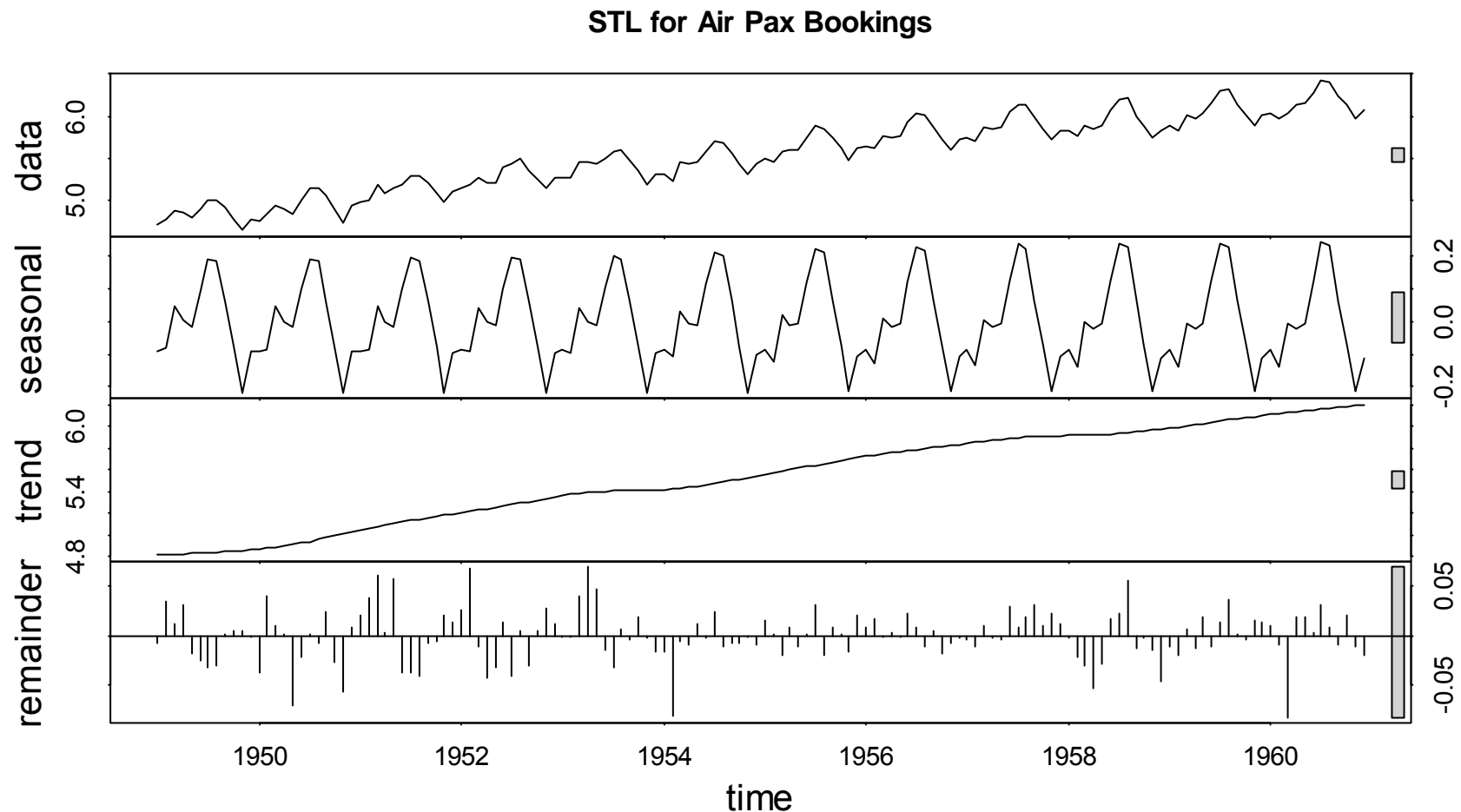


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL for Series with Evolving Seasonality

```
> lap.stl <- stl(lap, s.window=13)  
> plot(lap.stl, main="STL for Air Pax Bookings")
```



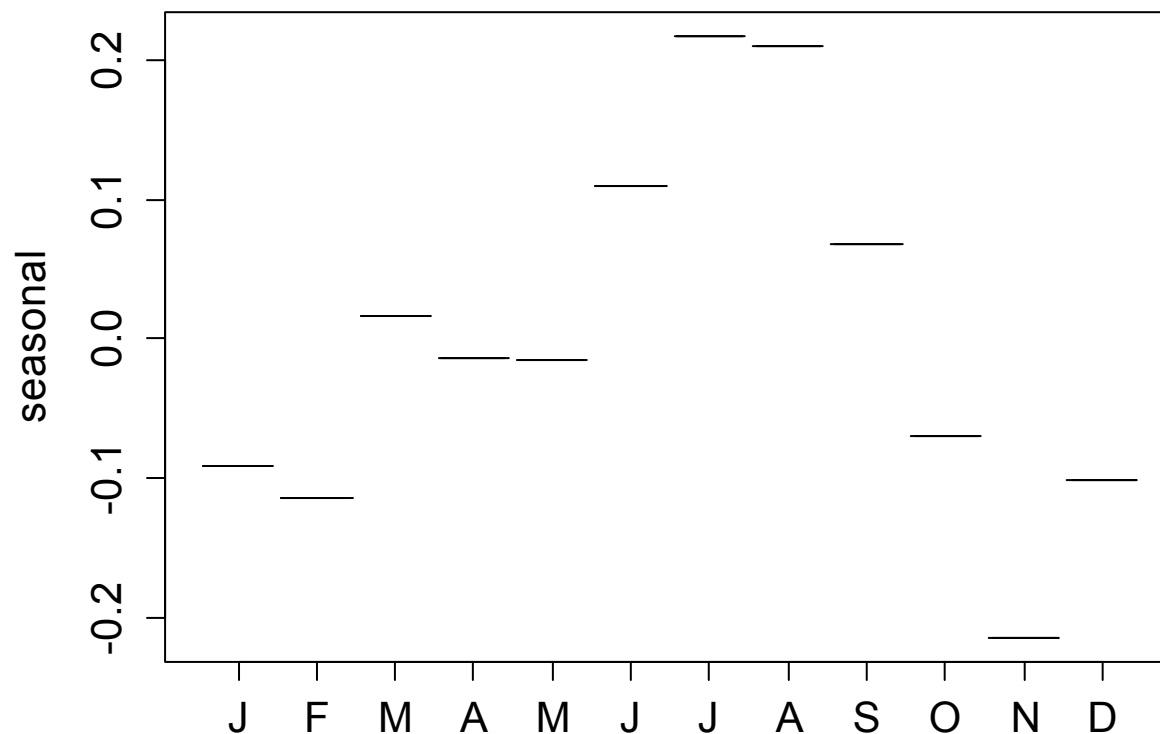
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL for Series with Evolving Seasonality

```
> monthplot(stl(lap, s.window="periodic"))
```

Monthplot, s.window="periodic"



Constant Seasonality:
Check the STL plot on the previous slide for assessing whether this is reasonable or not!

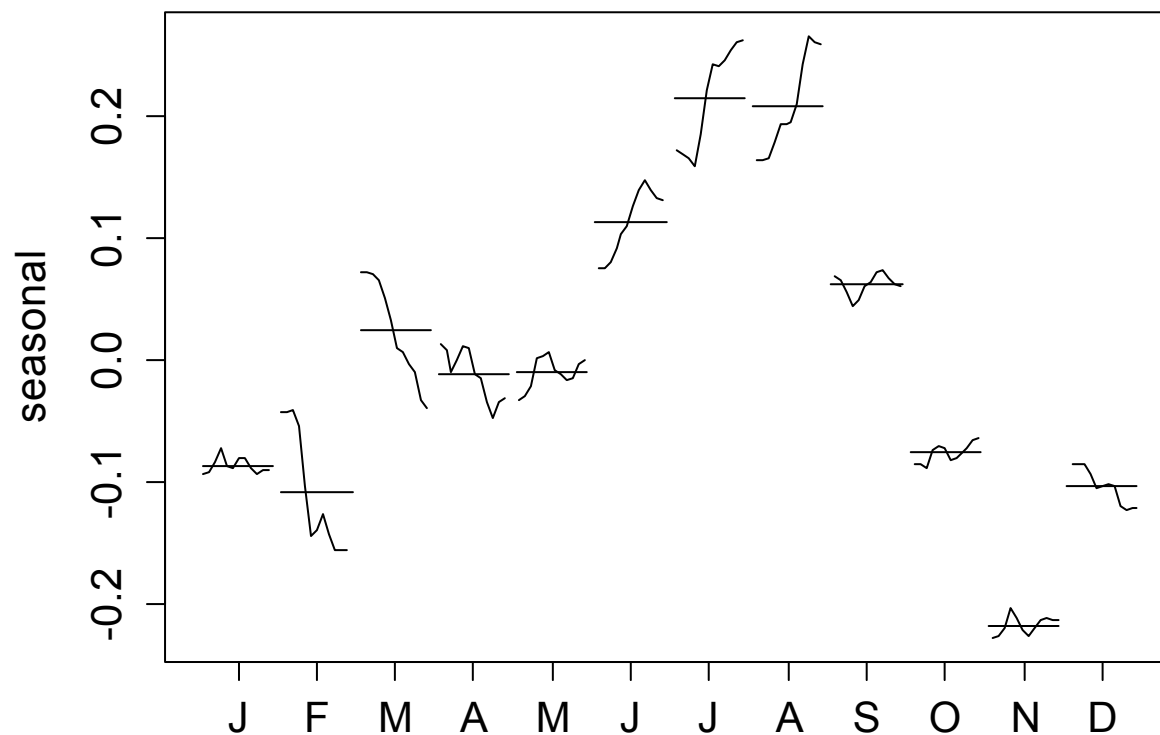
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL for Series with Evolving Seasonality

```
> monthplot(stl(lap, s.window=5))
```

Monthplot, s.window=5



Evolving Seasonality:

Too little smoothing in the seasonal effect, the changes are irregular.

As a remedy, increase parameter `s.window`

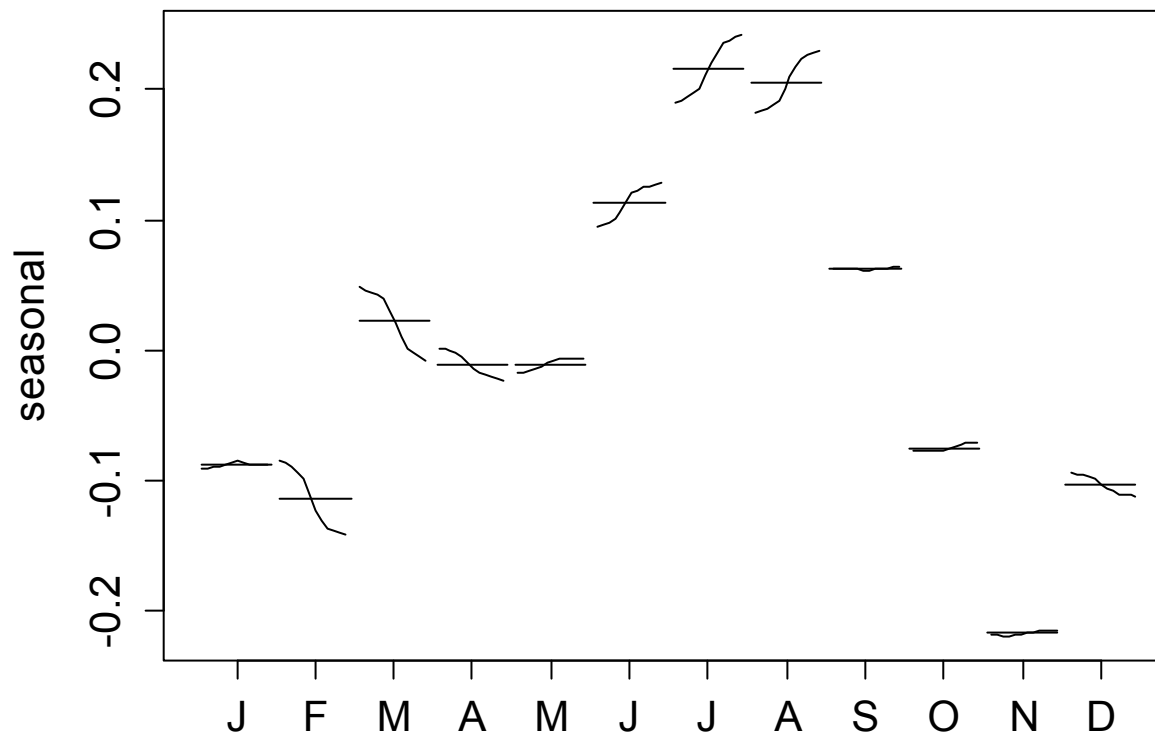
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL for Series with Evolving Seasonality

```
> monthplot(stl(lap, s.window=13))
```

Monthplot, s.window=13



Evolving Seasonality:

Adequate amount of smoothing will well chosen `s.window`

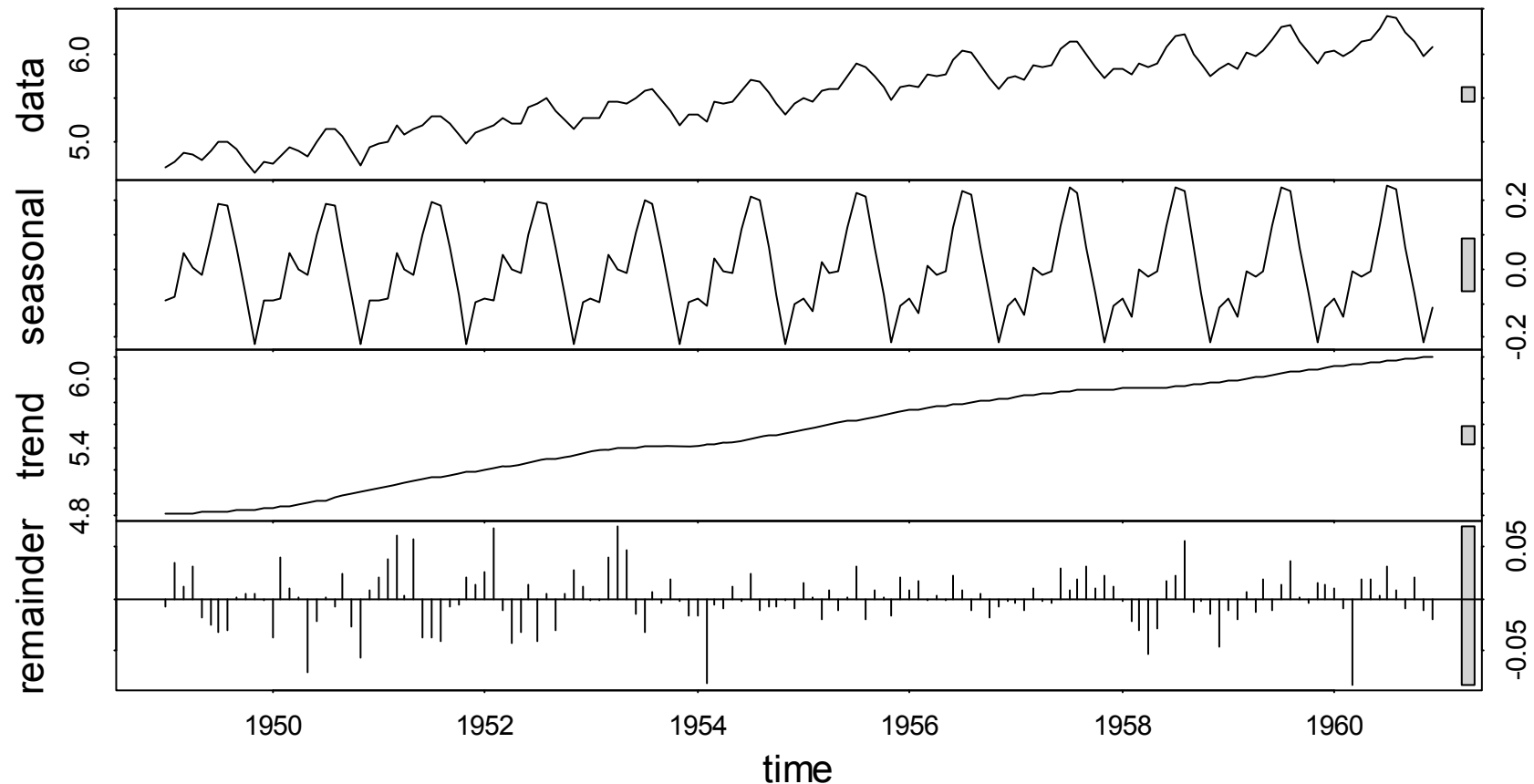
Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL for Series with Evolving Seasonality

```
> plot(stl(lap, s.window=13))
```

STL Decomposition with Time Varying Seasonal Effect



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

STL Decomposition: Remarks

Some advantages and disadvantages:

- + trend and seasonal effect can be estimated
 - + \hat{m}_t , \hat{s}_t and \hat{R}_t are explicitly known & can be visualised
 - + resulting time series has the same length as original
 - + relatively complicated, but good R implementation
 - + `stl()` even allows for evolving seasonality
 - + good estimators, robust against outliers
-
- extrapolation of \hat{m}_t , \hat{s}_t are not entirely obvious

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Parametric Modelling

When to use?

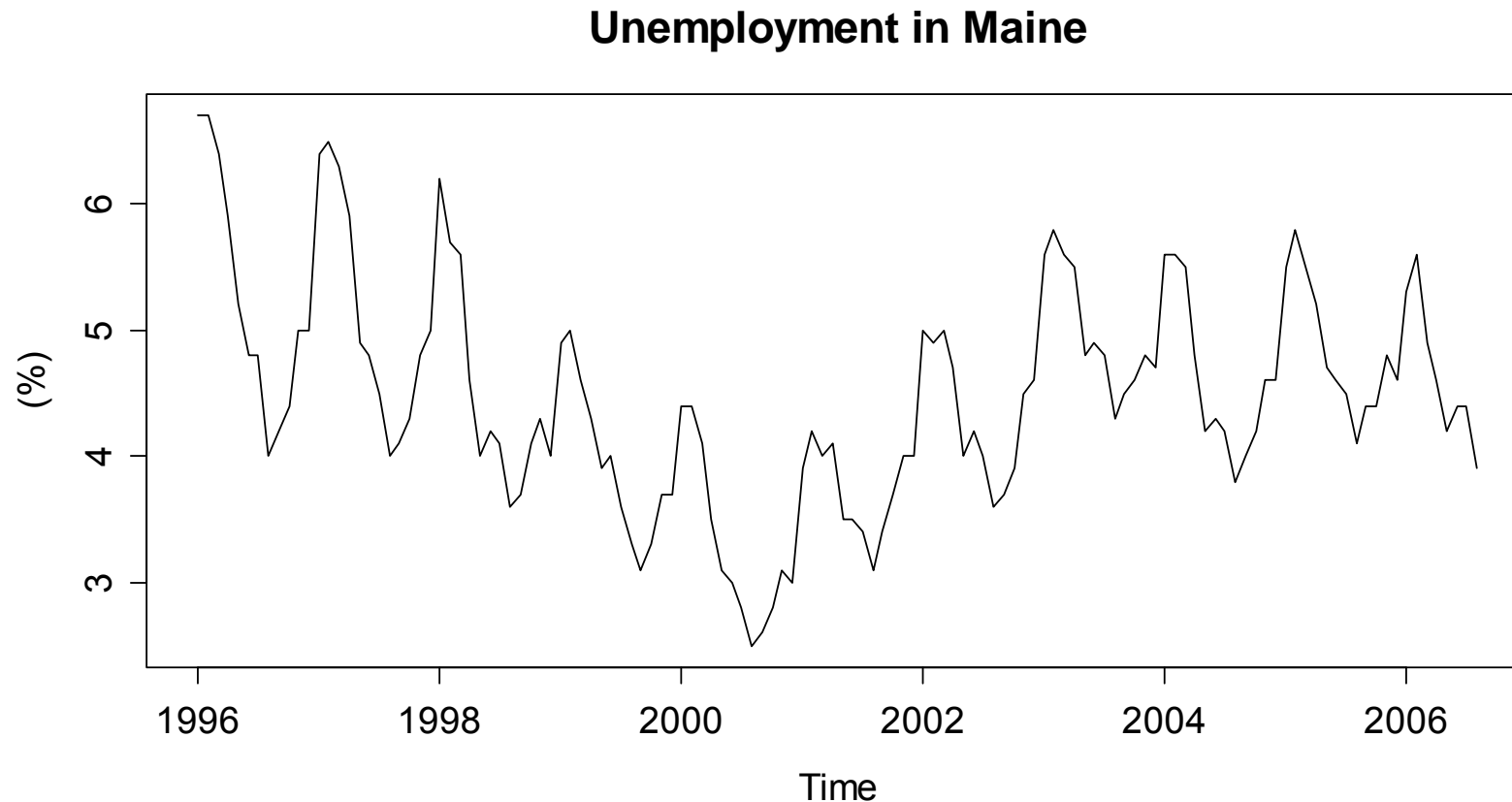
- Parametric modelling can be used if we have previous knowledge about the trend and/or seasonality following a functional form.
- If the main goal of the analysis is forecasting, a trend in functional form may allow for easier extrapolation than a trend obtained via smoothing. But it can be treacherous!
- It can also be useful if we have a specific model in mind and want to infer it. Typical example: testing whether a linear trend exists. **Caution: beware of correlated errors!**

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Parametric Modelling: Example

Maine unemployment data: Jan/1996 – Aug/2006



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Modeling the Unemployment Data

Most often, time series are parametrically decomposed by using regression models. For the trend, polynomial functions are widely used, whereas the seasonal effect is modelled with dummy variables (= a factor).

$$X_t = \beta_0 + \beta_1 \cdot t + \beta_2 \cdot t^2 + \beta_3 \cdot t^3 + \beta_4 \cdot t^4 + \alpha_{i(t)} + E_t$$

where $t \in \{1, 2, \dots, 128\}$

$i(t) \in \{1, 2, \dots, 12\}$

Remark: choice of the polynomial degree is crucial!

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Polynomial Order / OLS Fitting

Estimation of the coefficients will be done in a regression context. We can use the ordinary least squares algorithm, but:

- we have violated assumptions, E_t is not uncorrelated
- the estimated coefficients are still unbiased
- standard errors (tests, CIs) can be wrong

Which polynomial order is required?

Eyeballing allows to determine the minimum grade that is required for the polynomial. It is at least the number of maxima the hypothesized trend has, plus one.

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Important Hints for Fitting

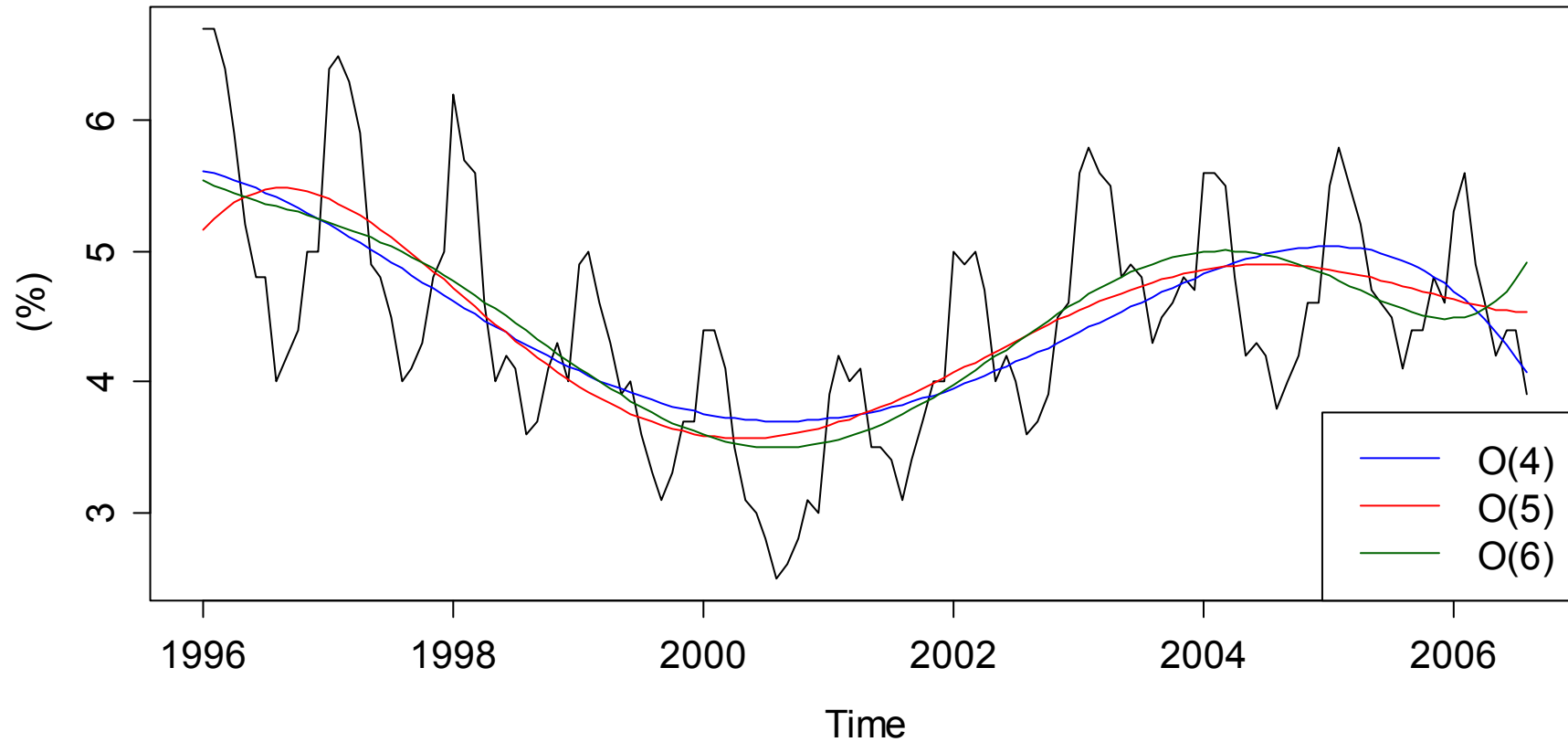
- The main predictor used in polynomial parametric modeling is the time of the observations. It can be obtained by typing `xx <- as.numeric(time(maine))`.
- For avoiding numerical and collinearity problems, it is key to center the time/predictors: `xc <- xx - mean(xx)`.
- R by default sets the first factor level value to 0, seasonality is thus expressed as the surplus to the January value.
- For visualization: when the trend must fit the data, we have to adjust, because the mean for the seasonal effect is usually different from zero!

Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Trend of O(4), O(5) and O(6)

Unemployment in Maine

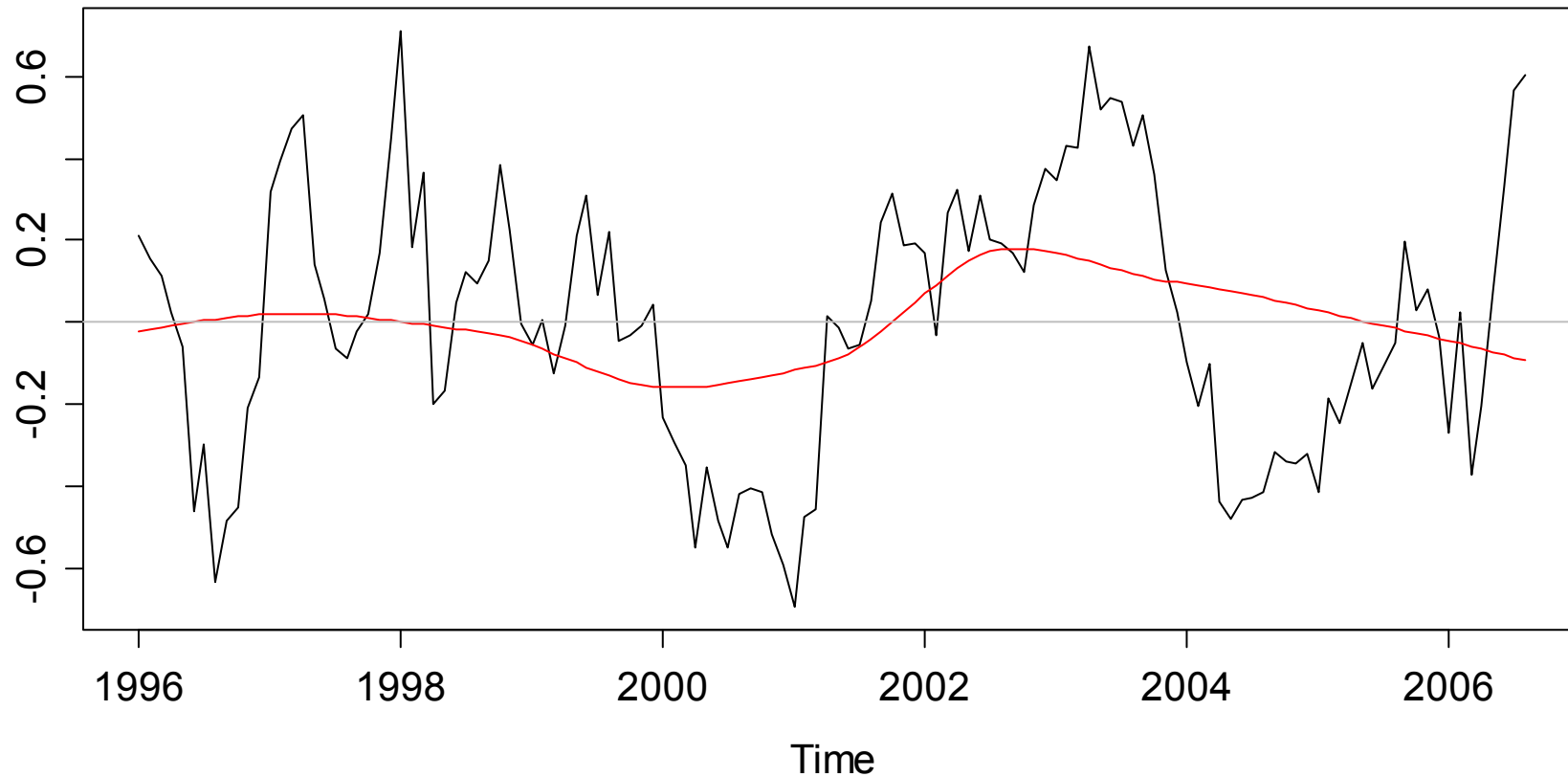


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Residual Analysis: O(4)

Residuals vs. Time, O(4)

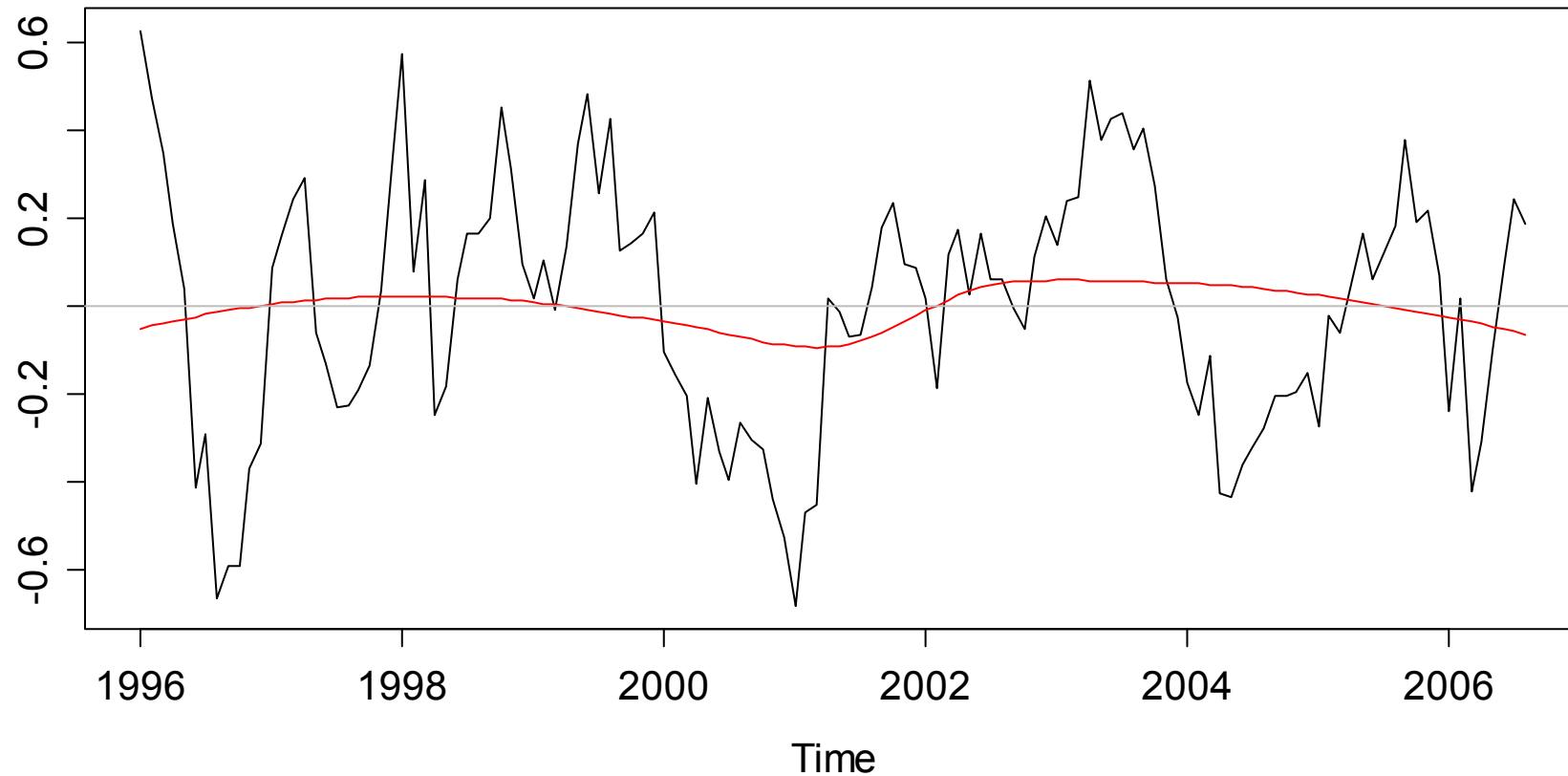


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Residual Analysis: O(5)

Residuals vs. Time, O(5)

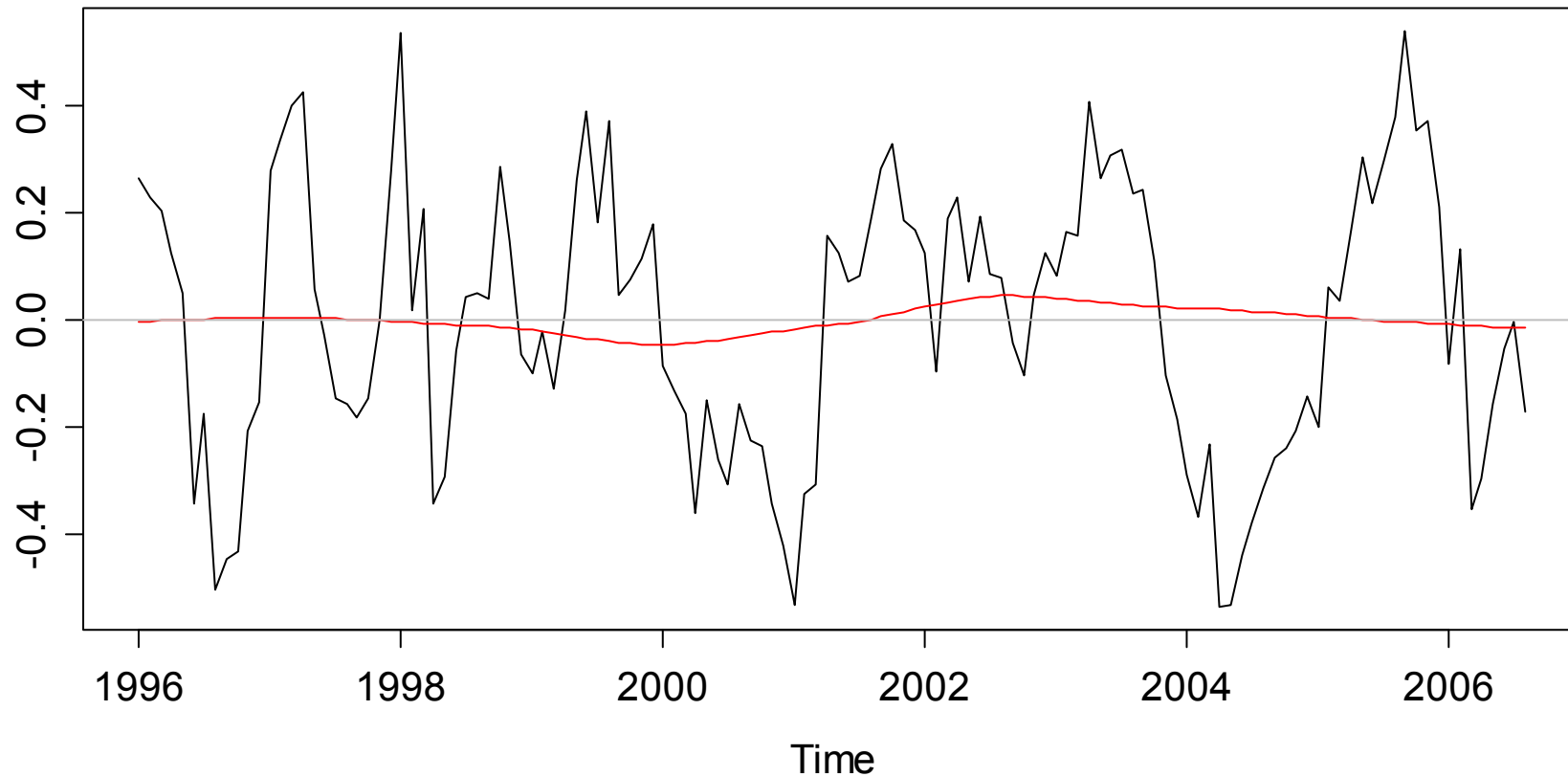


Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Residual Analysis: O(6)

Residuals vs. Time, O(6)



Applied Time Series Analysis

SS 2016 – Descriptive Analysis

Parametric Modeling: Remarks

Some advantages and disadvantages:

- + functional trend and seasonal effect can be estimated
- + \hat{m}_t and \hat{s}_t are explicitly known, can be visualised
- + even some inference on trend/season is possible
- + time series keeps the original length
- choice of a/the correct model is necessary/difficult
- residuals are correlated: this is a model violation!
- extrapolation of \hat{m}_t, \hat{s}_t easy, but maybe treacherous!
- evolving seasonality possible, but rather difficult

Applied Time Series Analysis

SS 2016 – Autocorrelation

Where are we?

For most of the rest of this course, we will deal with (weakly) stationary time series. They have the following properties:

- $E[X_t] = \mu$
- $Var(X_t) = \sigma^2$
- $Cov(X_t, X_{t+h}) = \gamma_h$

If a time series is non-stationary, we know how to decompose into deterministic trend/season and stationary, random part.

Our forthcoming goals are:

- understanding the dependency in a stationary series
- modeling this dependency and generate forecasts

Applied Time Series Analysis

SS 2016 – Autocorrelation

Autocorrelation

The aim of this section is to estimate, explore and understand the dependency structure within a stationary time series.

Def: **Autocorrelation**

$$Cor(X_{t+k}, X_t) = \frac{Cov(X_{t+k}, X_t)}{\sqrt{Var(X_{t+k}) \cdot Var(X_t)}} = \rho(k)$$

Autocorrelation is a dimensionless measure for the strength of the linear association between the random variables X_{t+k} and X_t .

There are 2 estimators, i.e. the lagged sample and the plug-in.

→ *see slides & blackboard for a sketch of the two approaches...*

Applied Time Series Analysis

SS 2016 – Autocorrelation

Practical Interpretation of Autocorrelation

We e.g. assume $\rho(k) = 0.7$

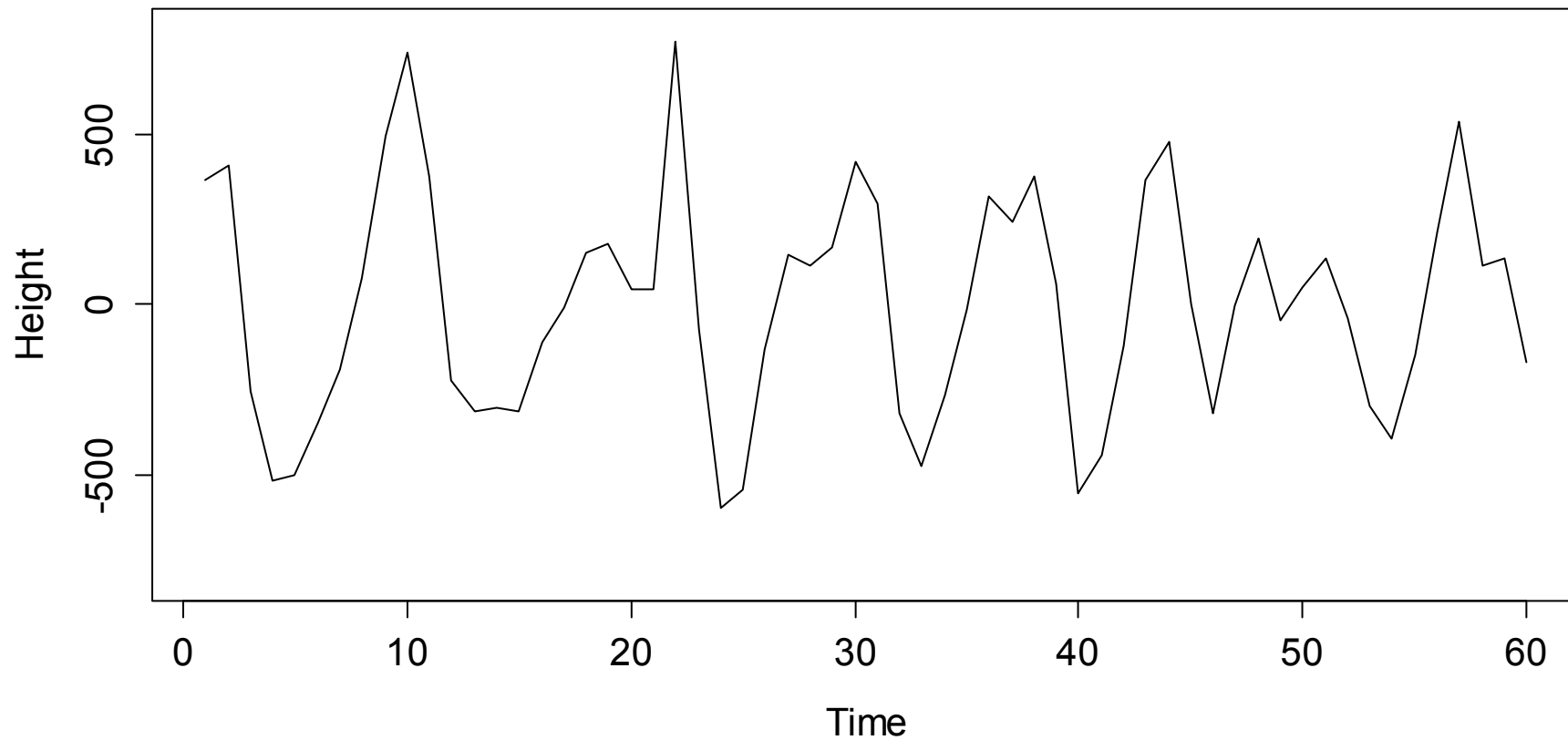
- The square of the autocorrelation, i.e. $\rho(k)^2 = 0.49$, is the percentage of variability explained by the linear association between X_t and its predecessor X_{t-k} .
- Thus, in our example, X_{t-k} accounts for roughly 49% of the variability observed in random variable X_t . Only roughly because the world is seldom exactly linear.
- From this we can also conclude that any $\rho(k) < 0.4$ is not a strong association, i.e. has a small effect on the next observation only.

Applied Time Series Analysis

SS 2016 – Autocorrelation

Example: Wave Tank Data

Wave Tank Data



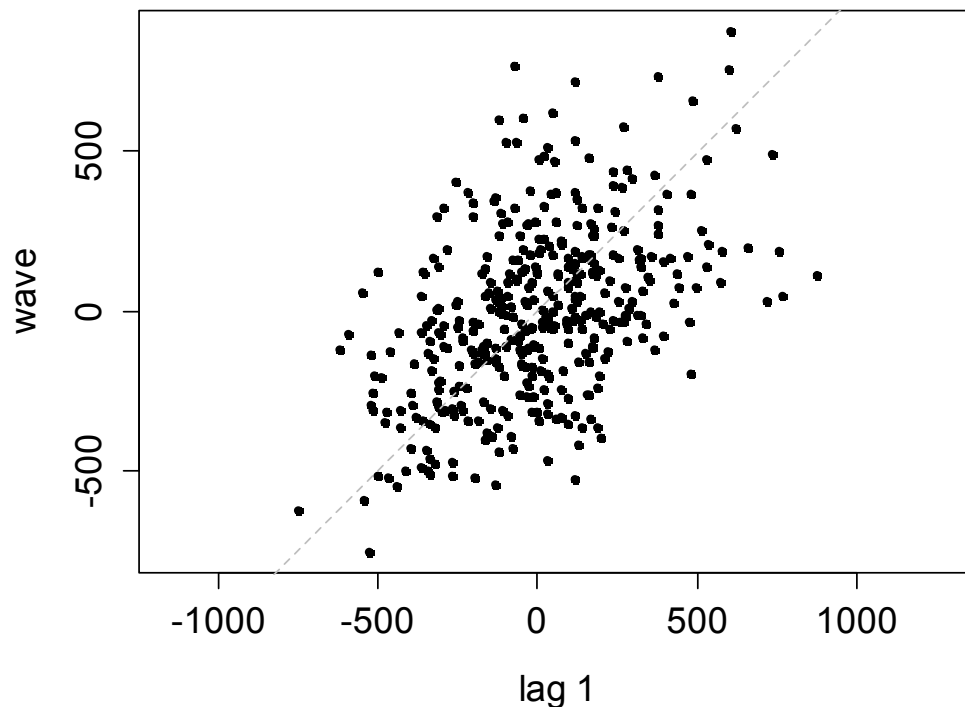
Applied Time Series Analysis

SS 2016 – Autocorrelation

Lagged Scatterplot Approach

Generate a plot of (x_t, x_{t+k}) for all $t = 1, \dots, n - k$ and compute the canonical Pearson correlation coefficient from these data pairs.

Lagged Scatterplot, k=1, cor=0.47



```
> lag.plot(wave, do.lines=FALSE, pch=20)
> title("Lagged Scatter, k=1, cor=0.47")
```

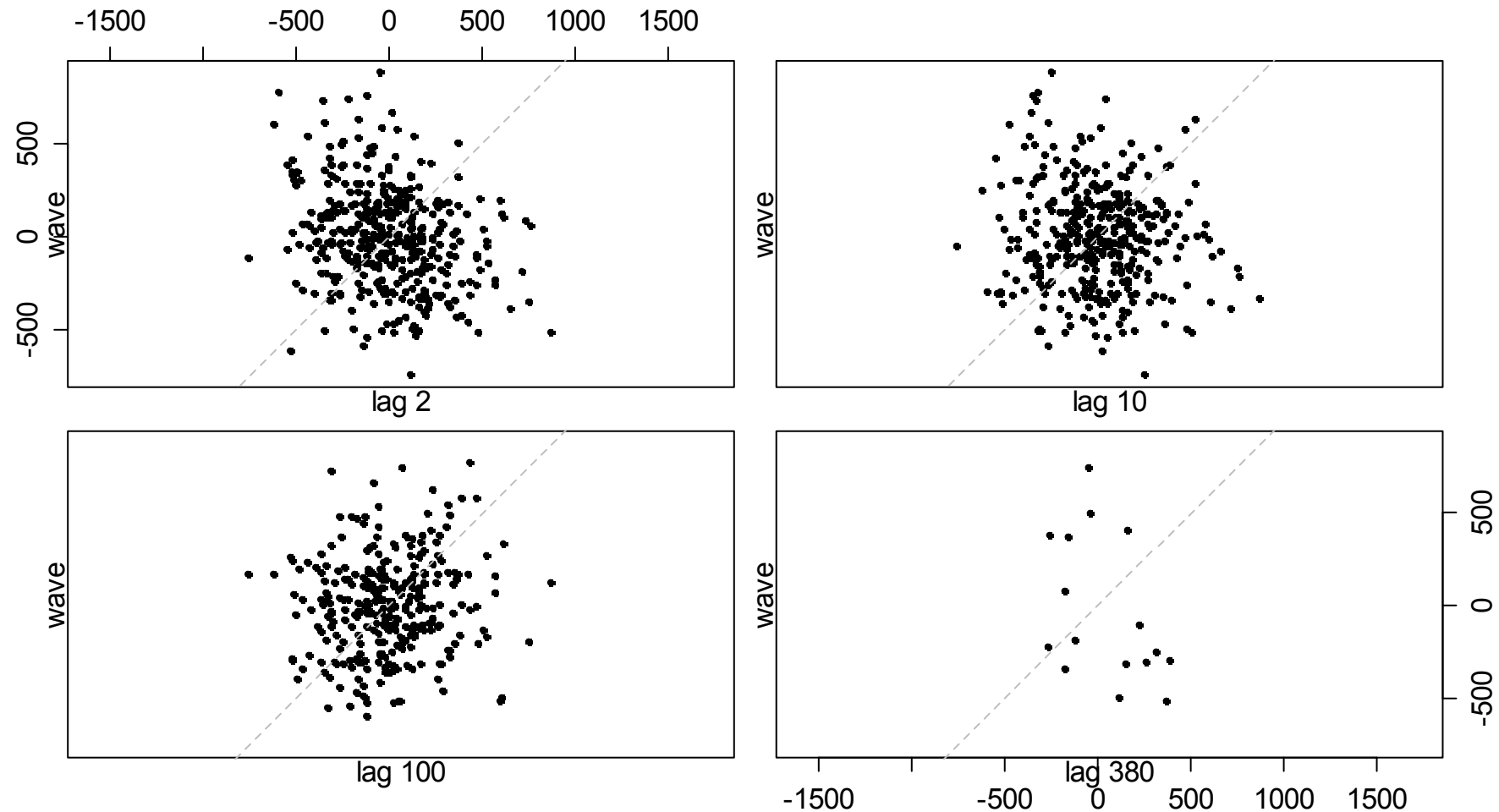
$$\tilde{\rho}(k) = \frac{\sum_{s=1}^{n-k} (x_{s+k} - \bar{x}_{(k)})(x_s - \bar{x}_{(1)})}{\sqrt{\sum_{s=k+1}^n (x_s - \bar{x}_{(k)})^2 \cdot \sum_{t=1}^{n-k} (x_t - \bar{x}_{(1)})^2}}$$

Applied Time Series Analysis

SS 2016 – Autocorrelation

Lagged Scatterplot for Higher Lags

Situation for lags $k = 2, 10, 100, 380$



Applied Time Series Analysis

SS 2016 – Autocorrelation

Plug-In Estimation

For obtaining an estimate of $\hat{\rho}(k)$, determine the sample covariance at lag k and divide by the sample variance.

$$\hat{\rho}(k) = \frac{\hat{\gamma}(k)}{\hat{\gamma}(0)} = \frac{\text{Cov}(X_t, X_{t+k})}{\text{Var}(X_t)}$$

$$\hat{\gamma}(k) = \frac{1}{n} \sum_{s=1}^{n-k} (x_{s+k} - \bar{x})(x_s - \bar{x})$$

where

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

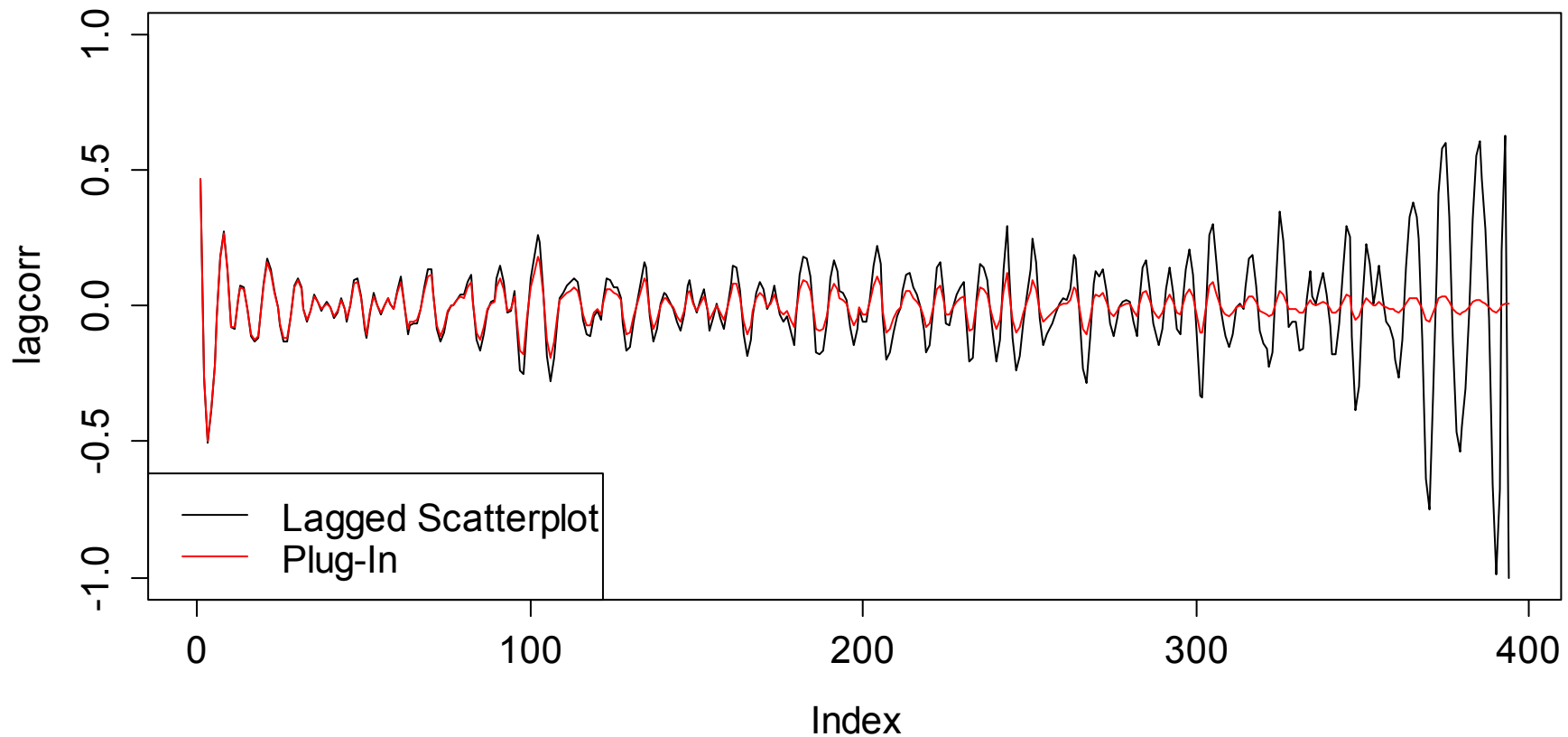
This is the standard approach for computing autocorrelations in time series analysis. It is better than the lagged scatterplot idea.

Applied Time Series Analysis

SS 2016 – Autocorrelation

Comparison: Lagged Scatterplot vs. Plug-In

ACF Estimation: Lagged Scatterplot vs. Plug-In

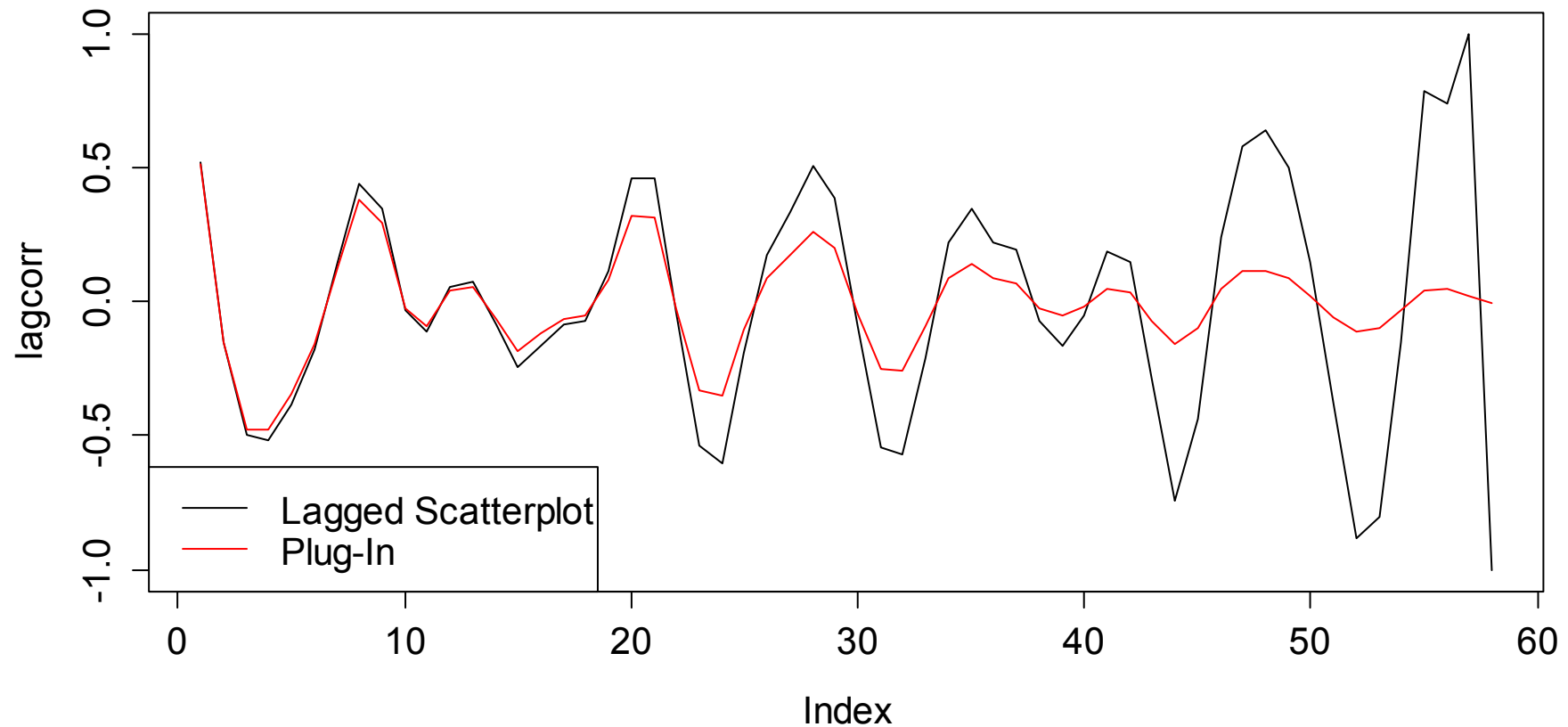


Applied Time Series Analysis

SS 2016 – Autocorrelation

Comparison: Lagged Scatterplot vs. Plug-In

ACF Estimation: Lagged Scatterplot vs. Plug-In



Applied Time Series Analysis

SS 2016 – Autocorrelation

What is important about ACF estimation?

- Correlations measure linear association and usually fail if there are non-linear associations between the variables.
- The bigger the lag k for which $\rho(k)$ is estimated, the fewer data pairs remain. Hence the higher the lag, the bigger the variability in $\hat{\rho}(k)$.
- To avoid spurious autocorrelation, the plug-in approach shrinks $\hat{\rho}(k)$ for large k towards zero. This creates a bias, but pays off in terms of *mean squared error*.
- Autocorrelations are only computed and inspected for lags up to $10 \cdot \log_{10}(n)$, where they have less bias/variance

Applied Time Series Analysis

SS 2016 – Autocorrelation

Correlogram

```
> acf(wave, plot=FALSE)
Autocorrelations of series 'wave', by lag

      0      1      2      3      4      5      6      7
1.000  0.470 -0.263 -0.499 -0.379 -0.215 -0.038  0.178

      8      9     10     11     12     13     14     15
0.269  0.130 -0.074 -0.079  0.029  0.070  0.063 -0.010

     16     17     18     19     20     21     22     23
-0.102 -0.125 -0.109 -0.048  0.077  0.165  0.124  0.049

     24     25
-0.005 -0.066
```

As always, numerical values are precise, but provide little overview over the characteristics of the data...

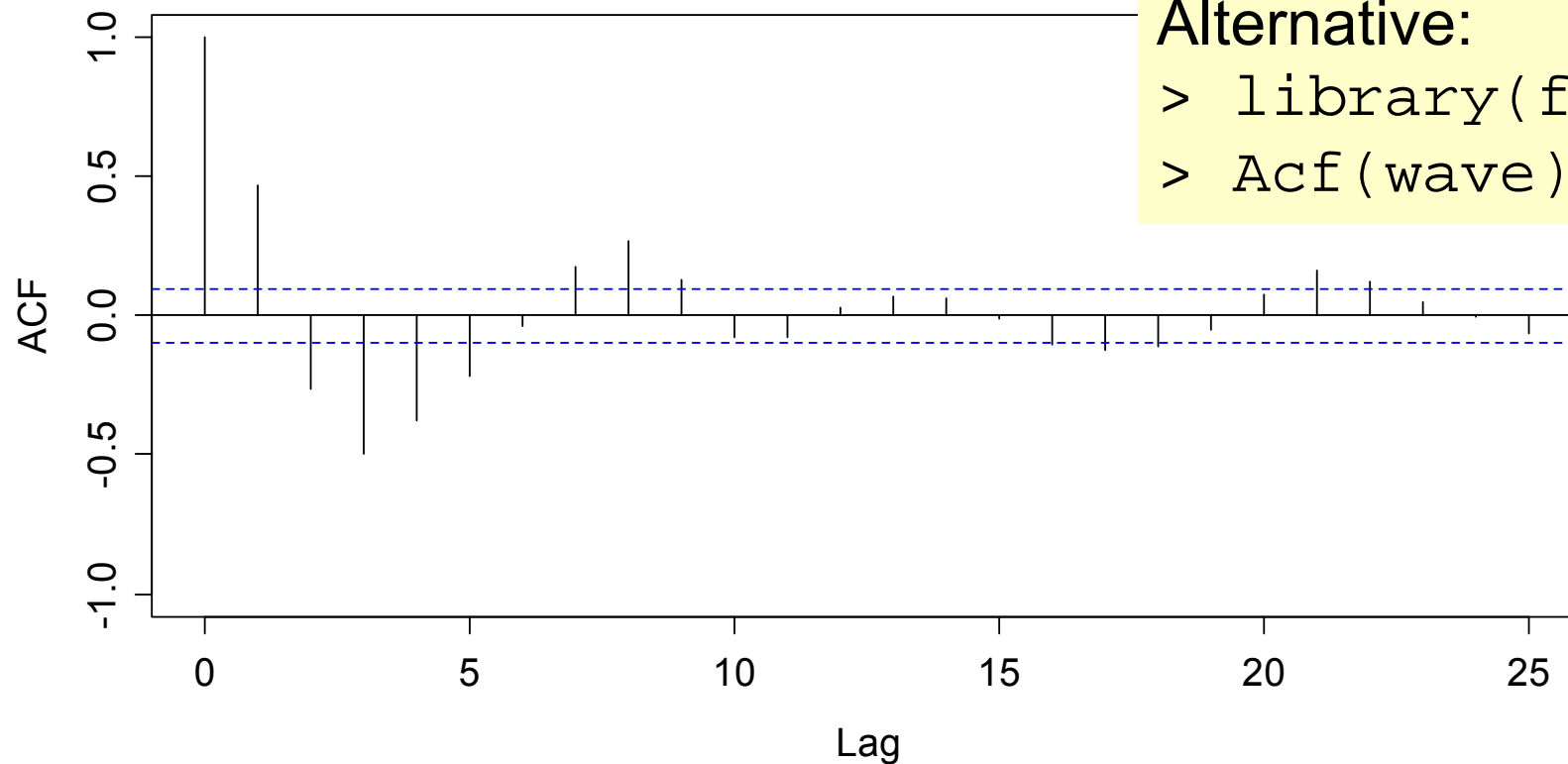
Applied Time Series Analysis

SS 2016 – Autocorrelation

Correlogram

```
> acf(wave, ylim=c(-1,1))
```

Correlogram of Wave Tank Data



Alternative:

```
> library(forecast)  
> Acf(wave)
```

Applied Time Series Analysis

SS 2016 – Autocorrelation

Confidence Bands

- Even for an *iid* series X_t without autocorrelation, i.e. $\rho(k) = 0$ for all k , the estimates will be different from zero: $\hat{\rho}(k) \neq 0$
- Question: which $\hat{\rho}(k)$ are significantly different from zero?

Mathematics:

Asymptotic result: $\hat{\rho}(k) \sim N(0, 1/n)$ for "large" n in *iid* series.

→ Under the null hypothesis of an *iid* series, a 95% acceptance region for the null is given by the interval $\pm 1.96 / \sqrt{n}$.

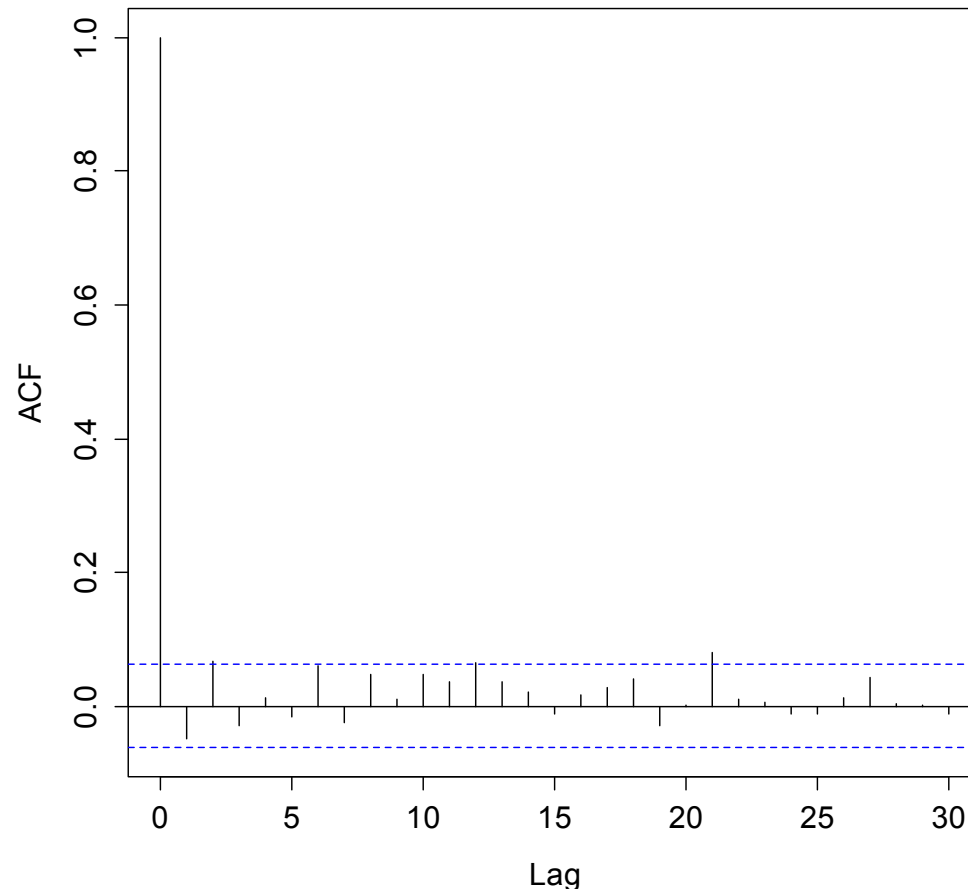
→ For any stationary series, $\hat{\rho}(k)$ within the confidence bands are considered to be different from 0 only by chance, while those outside are considered to be truly different from zero.

Applied Time Series Analysis

SS 2016 – Autocorrelation

Confidence Bands

ACF of iid Series, n=1000



Type I Errors:

For iid series, we need to expect 5% of type I errors, i.e. $\hat{\rho}(k)$ that go beyond the confidence bands by chance.

Non iid Series:

The confidence bands are asymptotic for iid series. Real finite length non-*iid* series have different (unknown) properties.

Applied Time Series Analysis

SS 2016 – Autocorrelation

Ljung-Box Test

The Ljung-Box approach tests the null hypothesis that a number of autocorrelation coefficients are simultaneously equal to zero. Thus, it tests for significant autocorrelation in a series. The test statistic is:

$$Q(h) = n \cdot (n + 2) \cdot \sum_{k=1}^h \frac{\hat{\rho}_k^2}{n - k} \sim \chi_h^2$$

In R:

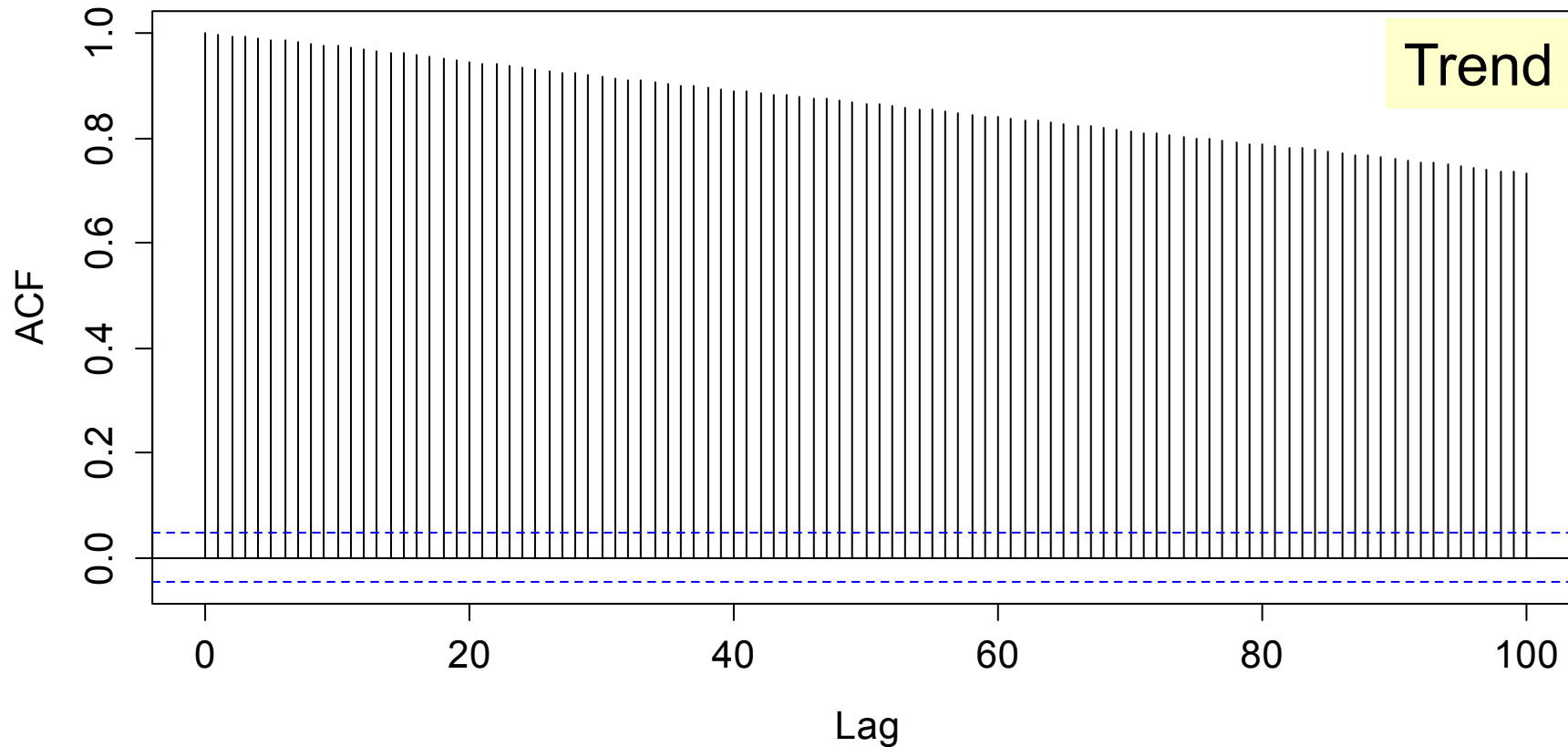
```
> Box.test(wave, lag=10, type="Ljung-Box")
Box-Ljung test
data: wave
X-squared = 344.0155, df = 10, p-value < 2.2e-16
```

Applied Time Series Analysis

SS 2016 – Autocorrelation

ACF of Non-Stationary Series

Correlogram of SMI Daily Closing Values

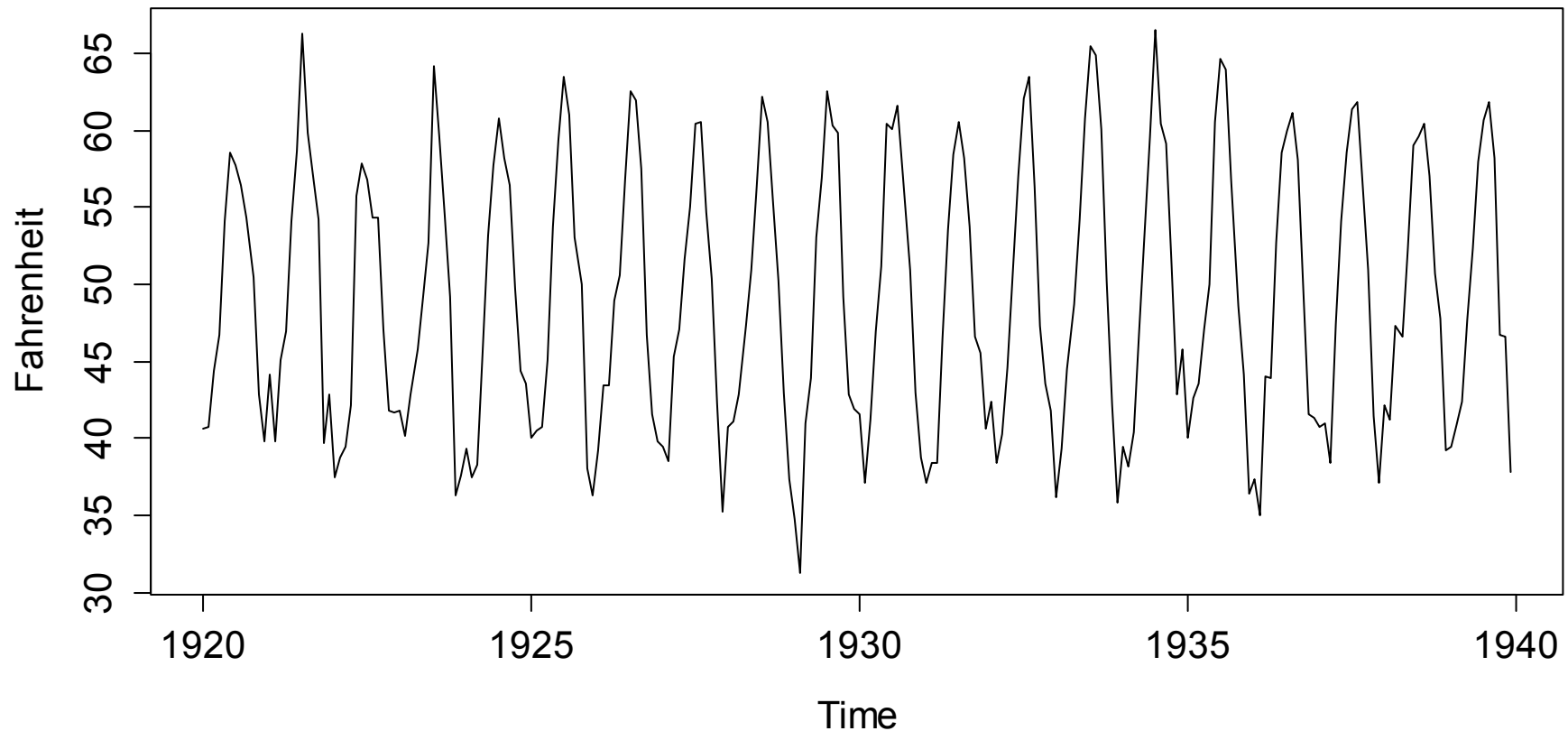


Applied Time Series Analysis

SS 2016 – Autocorrelation

ACF of Non-Stationary Series

Nottingham Monthly Average Temperature Data

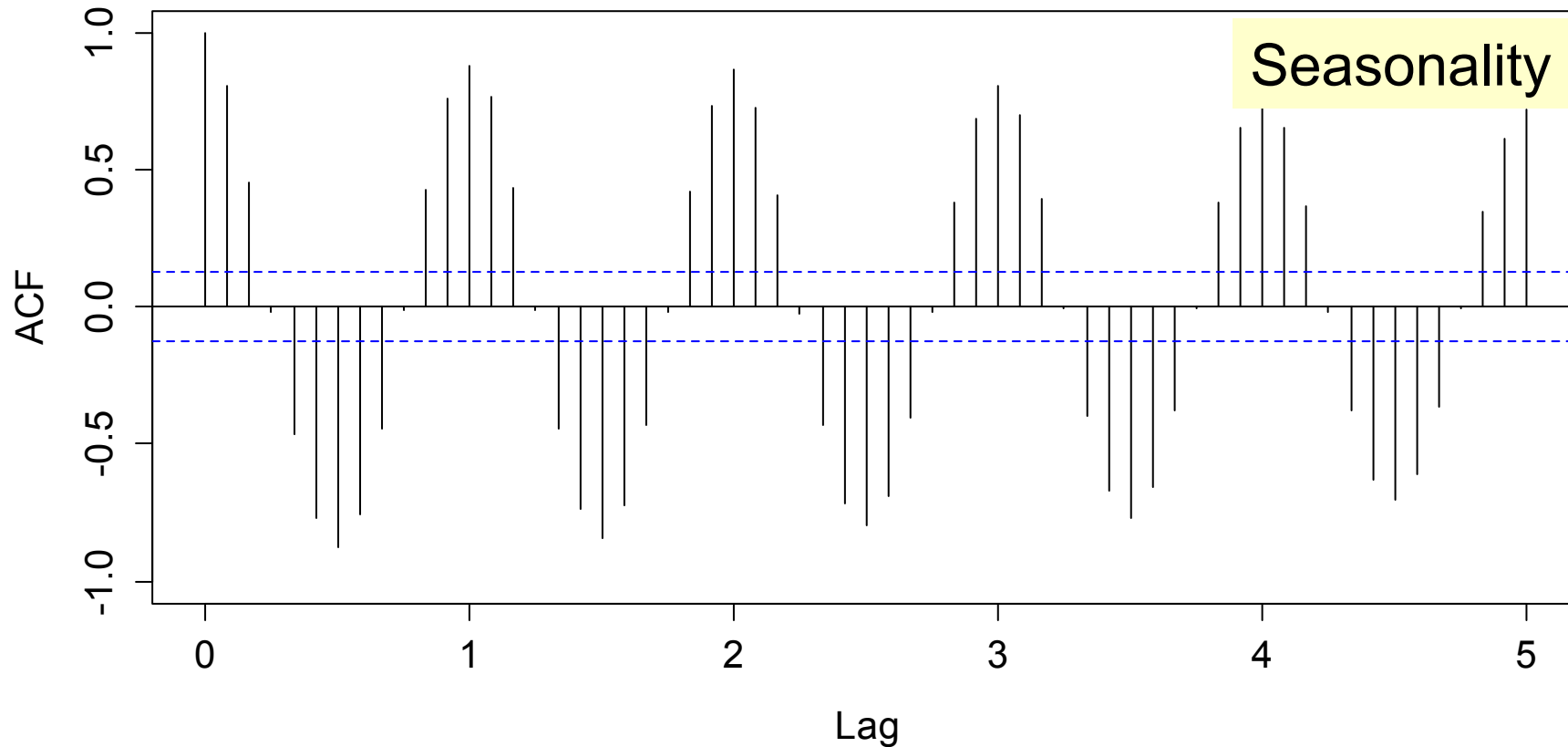


Applied Time Series Analysis

SS 2016 – Autocorrelation

ACF of Non-Stationary Series

Correlogram of Nottingham Temperature Data

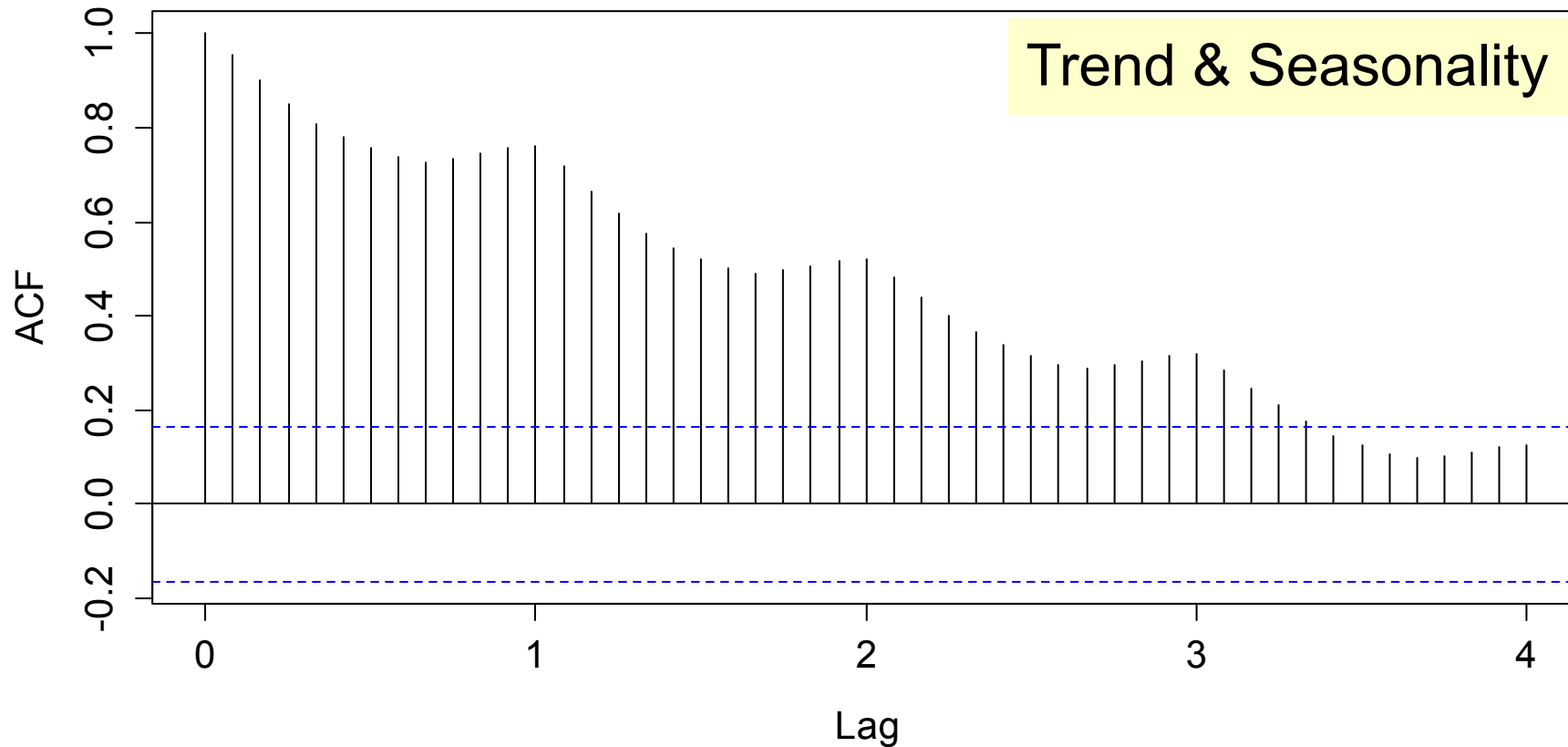


Applied Time Series Analysis

SS 2016 – Autocorrelation

ACF of Non-Stationary Series

Correlogram of Logged Air Passenger Bookings

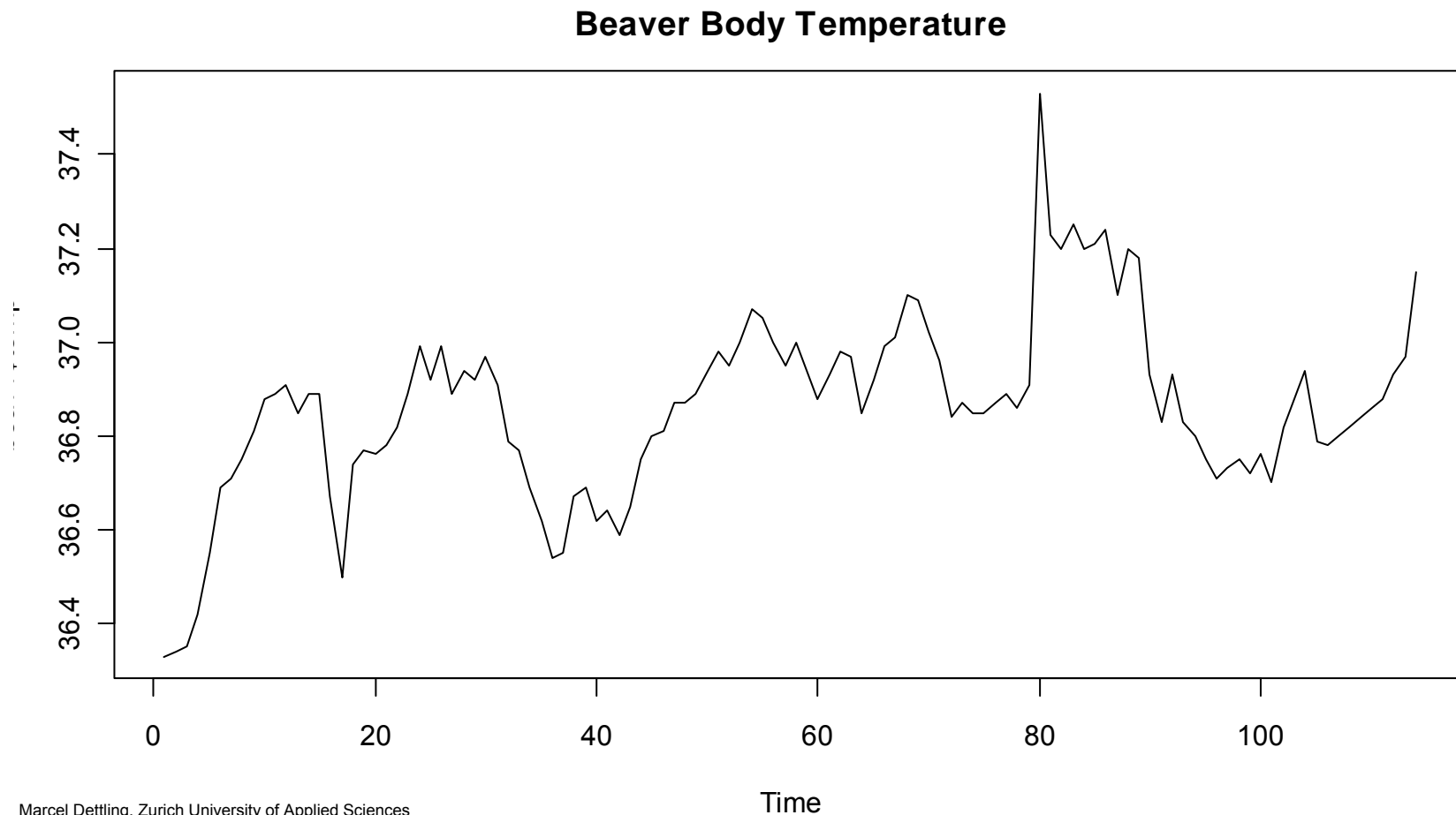


Applied Time Series Analysis

SS 2016 – Autocorrelation

The ACF and Outliers

Outliers in a time series may strongly affect the ACF estimation!



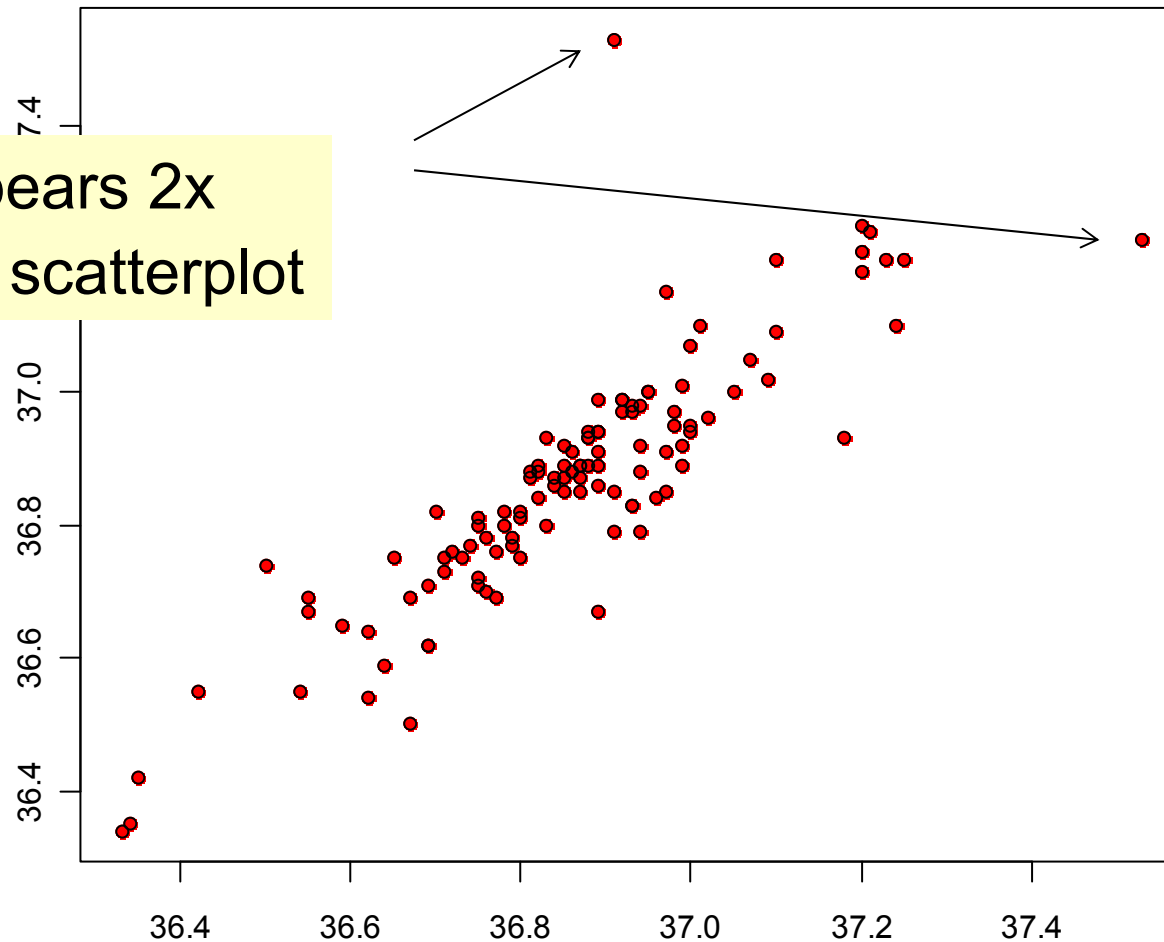
Applied Time Series Analysis

SS 2016 – Autocorrelation

The ACF and Outliers

Lagged Scatterplot with $k=1$ for Beaver Data

1 Outlier, appears 2x
in the lagged scatterplot

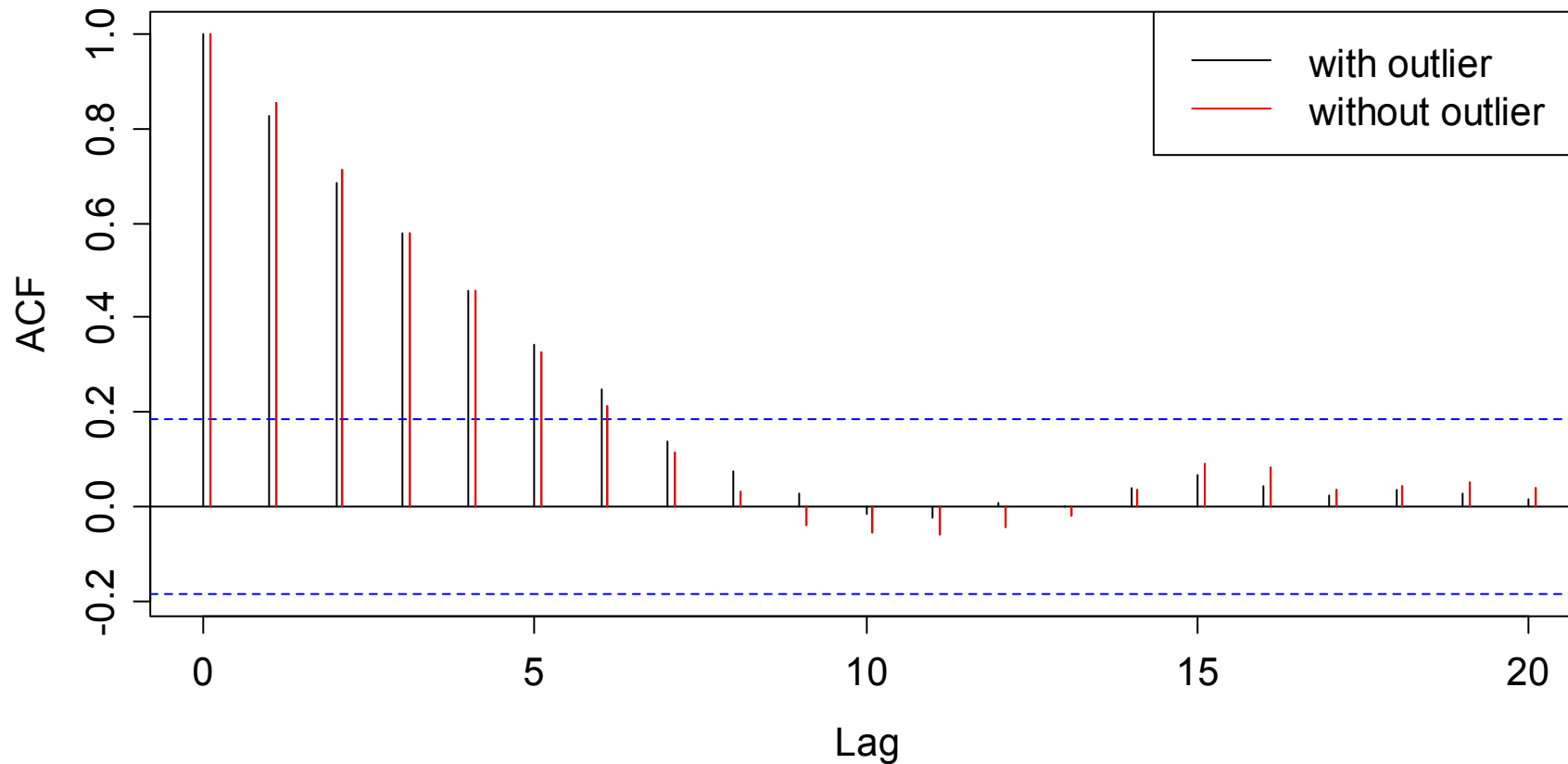


Applied Time Series Analysis

SS 2016 – Autocorrelation

The ACF and Outliers

Correlogram of Beaver Temperature Data



Applied Time Series Analysis

SS 2016 – Autocorrelation

The ACF and Outliers

The estimates $\hat{\rho}(k)$ are sensitive to outliers. They can be diagnosed using the lagged scatterplot, where every single outlier appears twice.

Some basic strategies for dealing with outliers:

- if it is bad data point: delete the observation
- most (if not all) R functions can deal with missing data
- if complete data are required, replace missing values with:
 - a) global mean of the series
 - b) local mean of the series, e.g. +/- 3 observations
 - c) fit a time series model and predict the missing value

Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates

- a) Appearance of the series \Rightarrow Appearance of the ACF
Appearance of the series ~~\Leftarrow~~ Appearance of the ACF

- b) The compensation issue

$$\sum_{k=1}^{n-1} \hat{\rho}(k) = -\frac{1}{2}$$

All estimable autocorrelation coefficients sum up to $-1/2$.

- c) For large lags k , there are only few data pairs for estimating $\rho(k)$. This leads to higher variability and hence the plug-in estimates are shrunken towards zero.

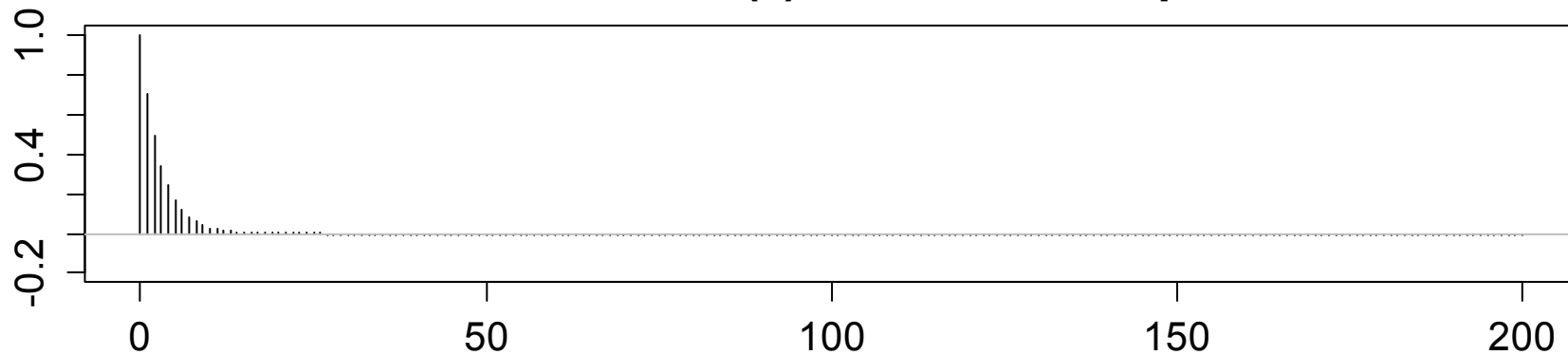
\rightarrow *We run some simulation studies for further insight.*

Applied Time Series Analysis

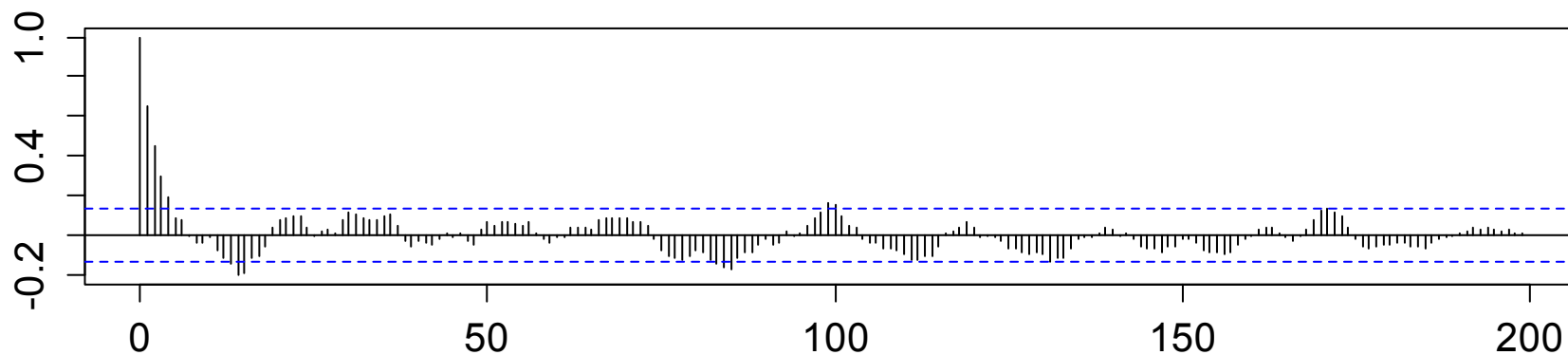
SS 2016 – Autocorrelation

Theoretical vs. Estimated ACF

True ACF of an AR(1) Process with $\alpha=0.7$



Correlogram for a Realization from an AR(1) Process



Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates

We run a simulation study for showing the distribution of $\hat{\rho}(k)$.
The procedure is as follows:

A) For an AR(1)-process, we know the theoretical ACF

B) Repeat for $i = 1, \dots, 1000$

 Simulate a **length** n AR(1)-process

 Estimate the ACF from that realization

End for

C) Boxplot the (bootstrap) sample distribution of ACF-estimates

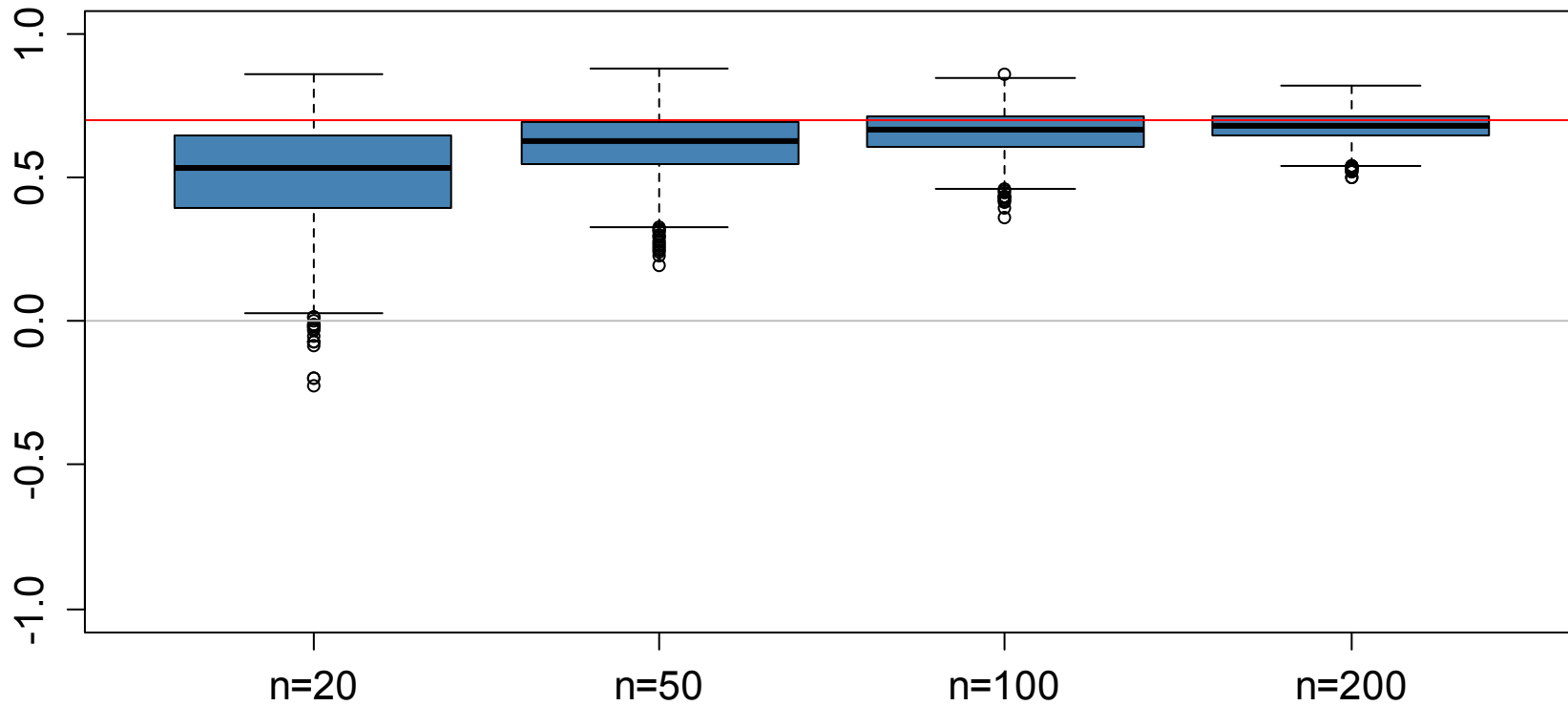
Do so for different **lags** k and different series **length** n .

Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates

Variation in ACF(1) estimation

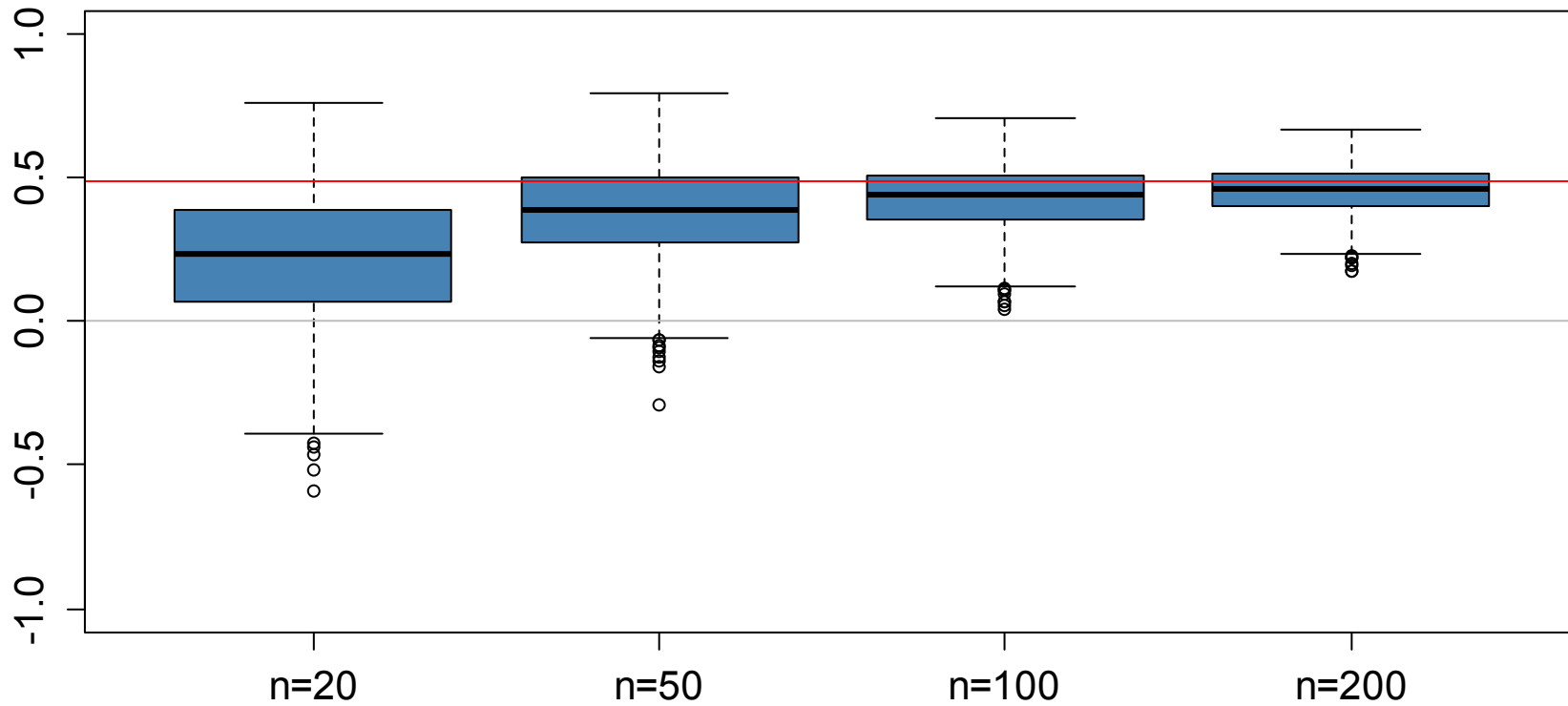


Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates

Variation in ACF(2) estimation

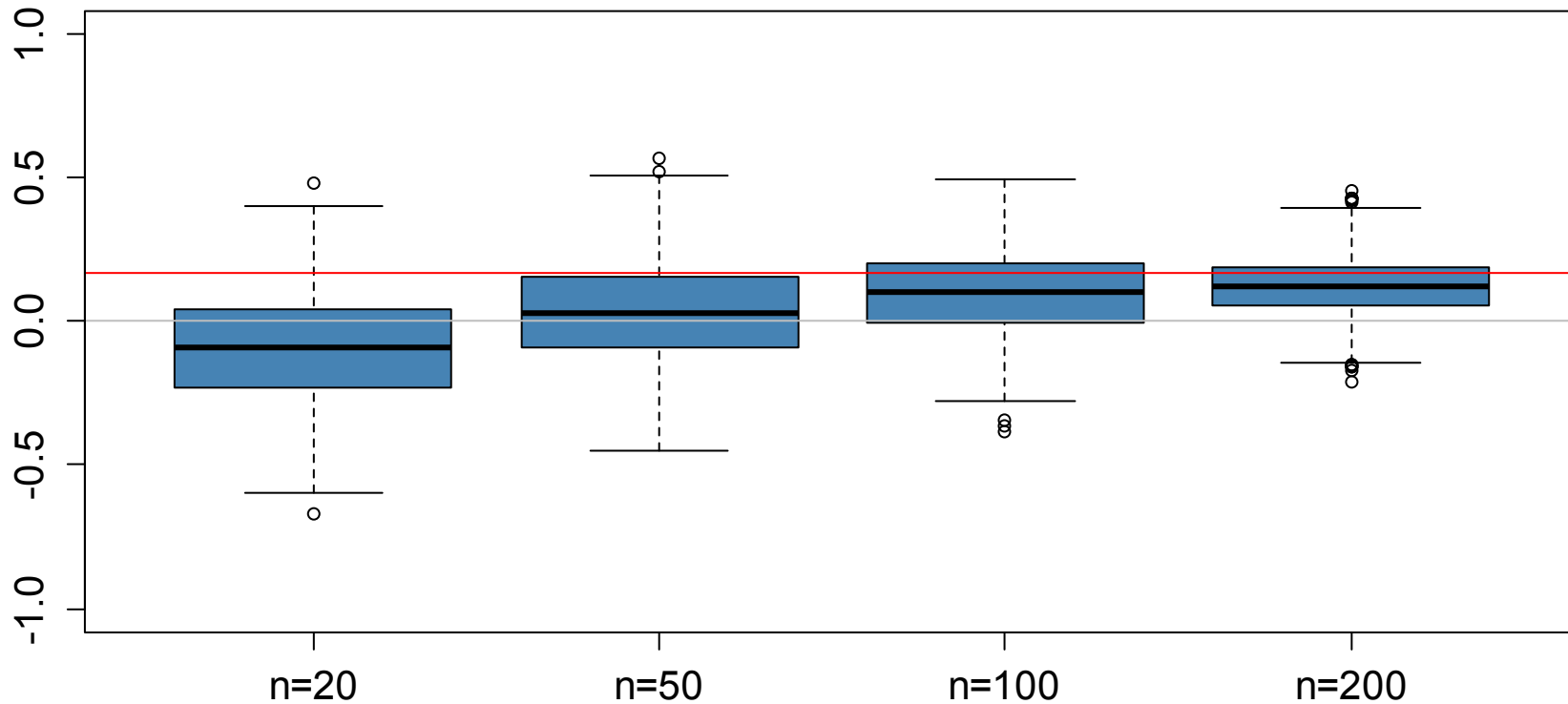


Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates

Variation in ACF(5) estimation

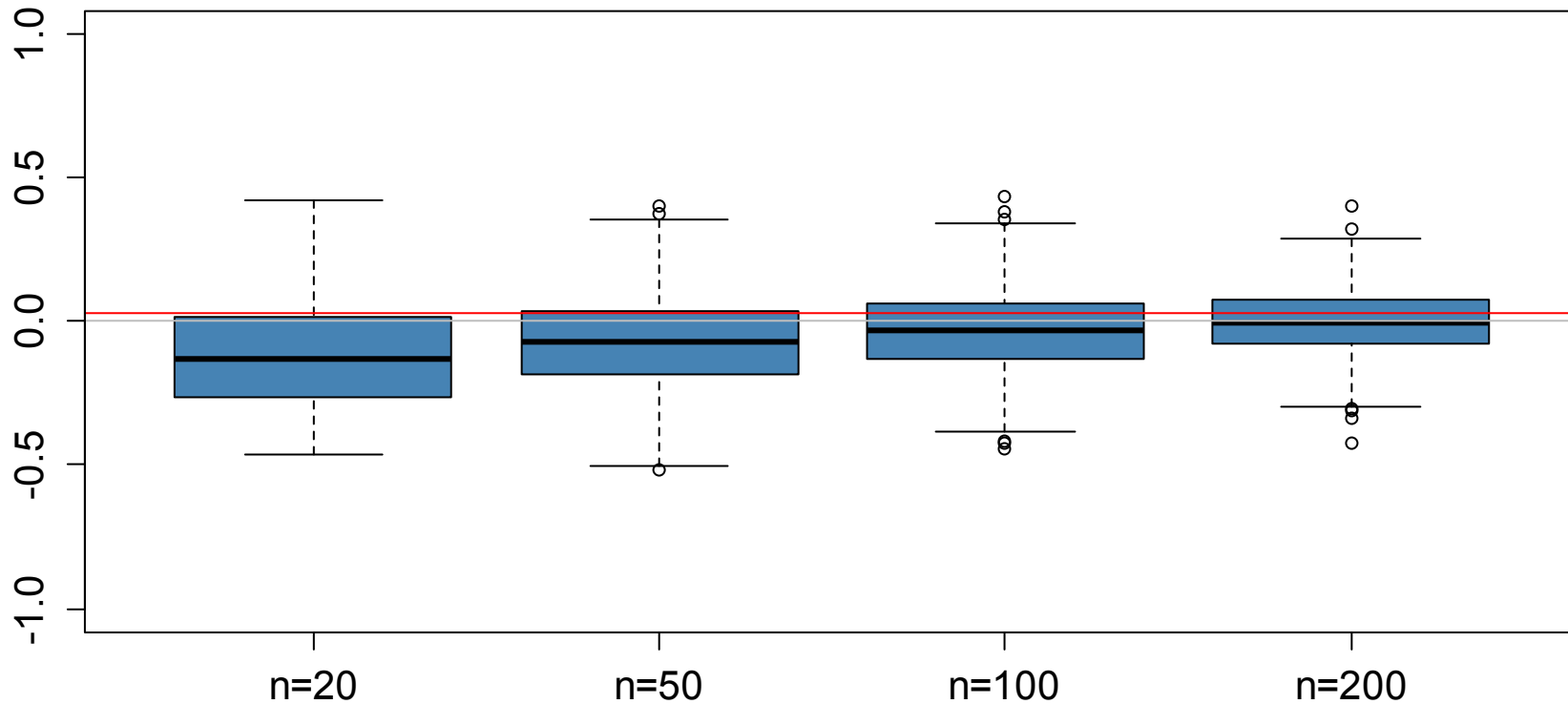


Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates

Variation in ACF(10) estimation



Applied Time Series Analysis

SS 2016 – Autocorrelation

Properties of ACF Estimates: Summary

- In short series, the $\hat{\rho}(k)$ are strongly biased. The consistency kicks in and kills the bias only after ~ 100 observations.
- The variability in the estimation of $\hat{\rho}(k)$ is considerable. We observe that we need at least 50, or better, 100 observations.
- For higher lags k , the bias seems a little less problematic, but the variability remains large even with many observations n .
- The confidence bounds, derived under independence, do not seem very accurate for (dependent) finite length time series.
- **Interpreting the ACF is tricky business! But there is no other alternative for inferring the serial correlation in a time series.**

Applied Time Series Analysis

SS 2016 – Autocorrelation

Application: Variance of the Arithmetic Mean

Practical problem: we need to estimate the mean of a realized/observed time series. We would like to attach a standard error.

- If we estimate the mean of a time series without taking into account the dependency, the standard error will be flawed.
- This leads to misinterpretation of tests and confidence intervals and therefore needs to be corrected.
- The standard error of the mean can both be over-, but also underestimated. This depends on the ACF of the series.

→ **For the derivation, see the blackboard...**

Applied Time Series Analysis

SS 2016 – Autocorrelation

Confidence Interval for Time Series Mean

An approximate 95% confidence interval for the mean of a time series that features autocorrelation is given by:

$$\hat{\mu} \pm 1.96 \cdot \sqrt{\frac{\gamma(0)}{n^2} \cdot \left(n + 2 \cdot \sum_{k=1}^{10 \cdot \log_{10}(n)} (n-k) \rho(k) \right)}$$

Note that it is usually not a good idea to plug-in all estimable autocorrelations, but to limit the contribution to the default max lag that is used in R: $10 \cdot \log_{10}(n)$.

Another alternative that will only be presented later is to fit a time series model and obtain a CI for the time series mean from it.

Applied Time Series Analysis

SS 2016 – Autocorrelation

Computation of CI for Time Series Mean

Under (falsely) assuming *iid* properties of the observations:

```
> mean(b) + c(-1.96, 1.96) * sd(b) / sqrt(length(b))  
[1] 36.827 36.898
```

When adjusting for the sequential correlation of the observations, the confidence interval becomes around **2.7x longer**, which can make a big difference!

```
> n <- length(b)  
> var.ts <- 1/n^2 * acf(b, lag=0, type="cov")$acf[1] *  
  (n + 2 * sum((n-1):(n-10)) * acf(b, 10)$acf[-1])  
> mean(b) + c(-1.96, 1.96) * sqrt(var.ts)  
[1] 36.765 36.959
```

Applied Time Series Analysis

SS 2016 – Autocorrelation

Partial Autocorrelation Function (PACF)

The k^{th} partial autocorrelation π_k is defined as the correlation between X_{t+k} and X_t , given all the values in between.

$$\pi_k = \text{Cor}(X_{t+k}, X_t \mid X_{t+1} = x_{t+1}, \dots, X_{t+k-1} = x_{t+k-1})$$

Interpretation:

- Given a time series X_t , the partial autocorrelation of lag k , is the autocorrelation between X_t and X_{t+k} with the linear dependence of X_{t+1} through to X_{t+k-1} removed.
- One can draw an analogy to regression. The ACF measures the „simple“ dependence between X_t and X_{t+k} , whereas the PACF measures that dependence in a „multiple“ fashion.

Applied Time Series Analysis

SS 2016 – Autocorrelation

Facts About the PACF and Estimation

We have:

- $\pi_1 = \rho_1$
- $\pi_2 = \frac{\rho_2 - \rho_1^2}{1 - \rho_1^2}$ for AR(1) models, we have $\pi_2 = 0$, because $\rho_2 = \rho_1^2$, i.e. there is no conditional relation between $(X_t, X_{t+2} | X_{t+1})$
- For estimating the PACF, we utilize the fact that for any AR(p) model, we have: $\pi_p = \alpha_p$ and $\pi_k = 0$ for all $k > p$. Thus, for finding $\hat{\pi}_p$, we fit an AR(p) model to the series for various orders p and set $\hat{\pi}_p = \hat{\alpha}_p$.

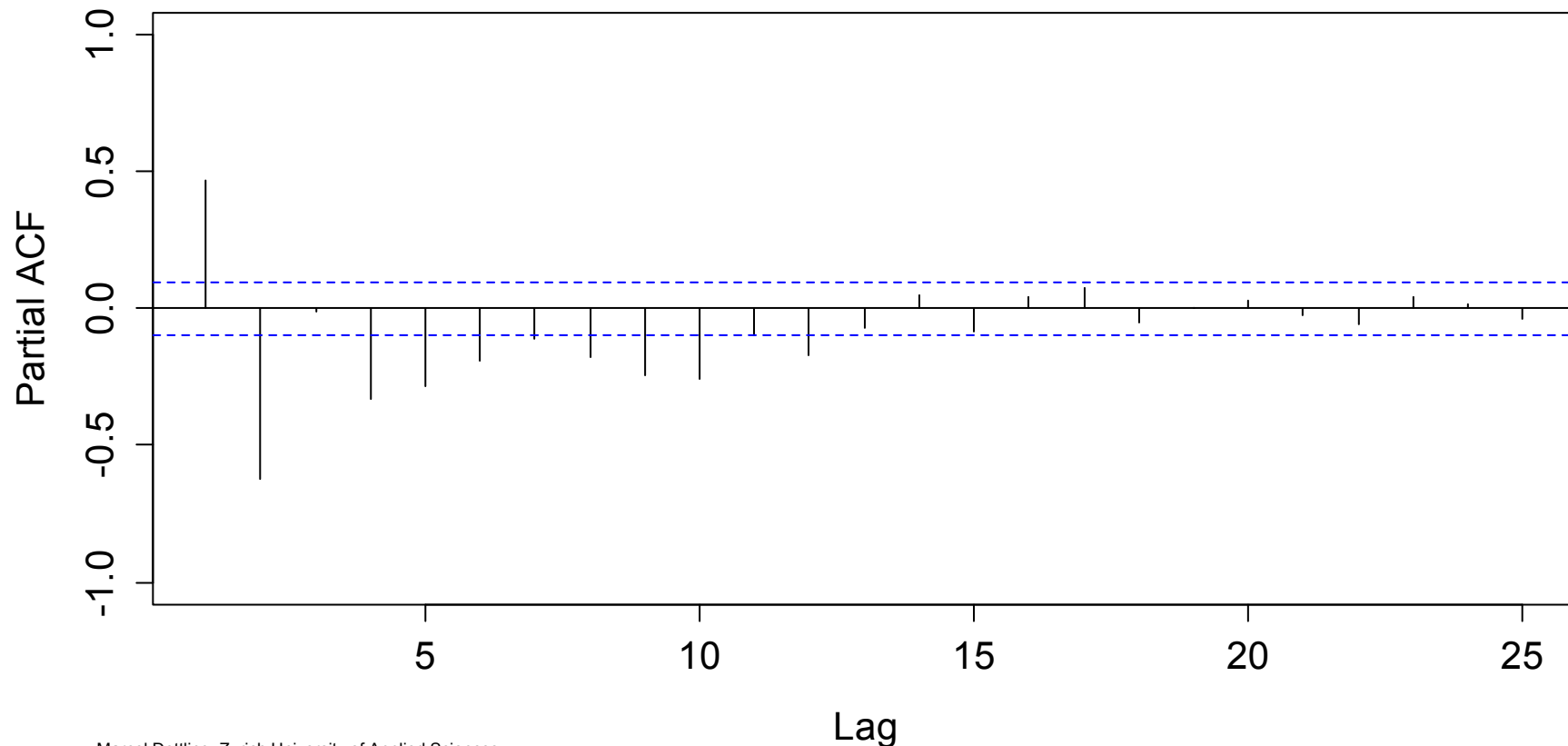
Applied Time Series Analysis

SS 2016 – Autocorrelation

PACF of Wave Tank Data

```
> pacf(wave, ylim=c(-1,1), main="PACF of ...")
```

PACF of Wave Tank Data

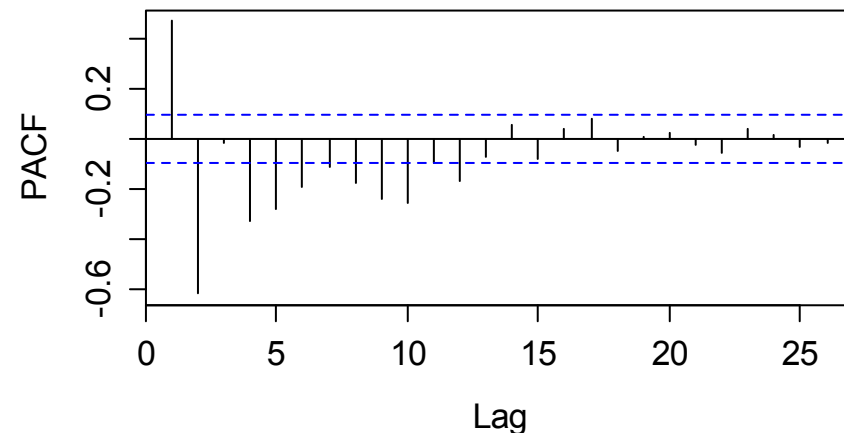
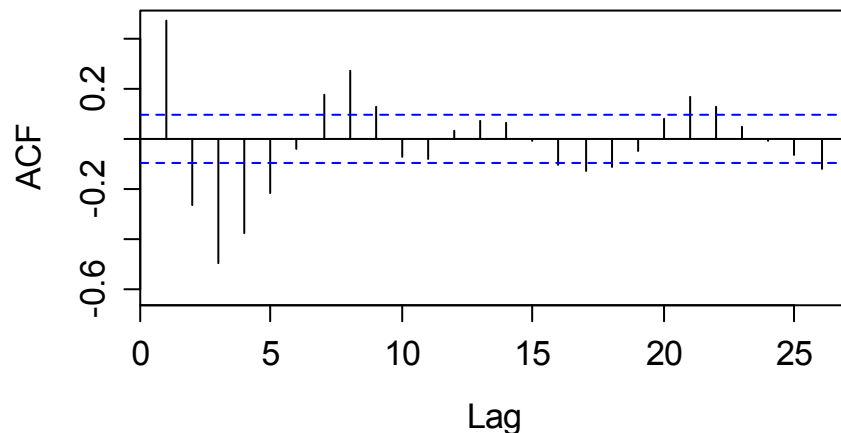
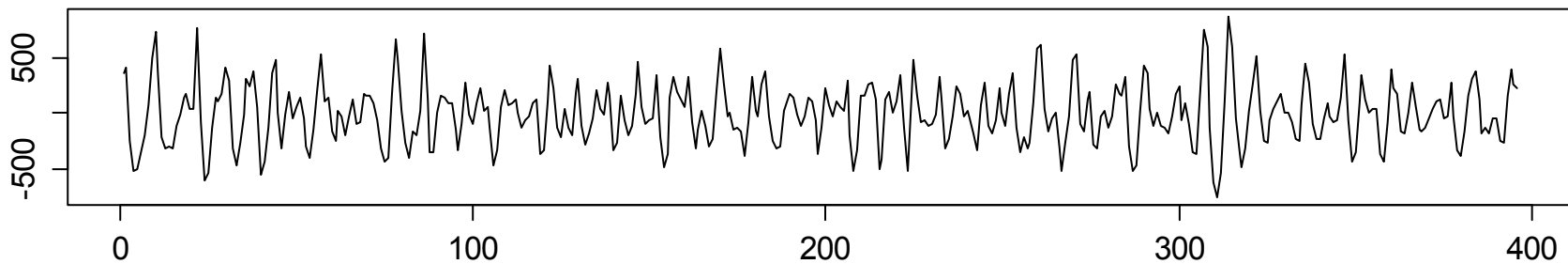


Applied Time Series Analysis

SS 2016 – Autocorrelation

Plotting the Series, ACF & PACF

```
> library(forecast); tdisplay(wave, points=F)  
wave
```



Applied Time Series Analysis

SS 2016 – Basics of Modeling

Basics of Modeling

Simulation & Generation

(Time Series) Model → Data

Estimation, Inference & Residual Analysis

Data → (Time Series) Model

We will first discuss the theoretical properties of the most important time series processes and then mainly focus on how to successfully fit models to data.

Applied Time Series Analysis

SS 2016 – Basics of Modeling

A Simple Model: White Noise

A time series (W_1, W_2, \dots, W_n) is a **White Noise series** if the random variables W_1, W_2, \dots are *independent and identically distributed* with *mean zero*.

This implies that all W_t have the same variance σ_W^2 , and

$$\text{Cov}(W_i, W_j) = 0 \quad \text{for all } i \neq j.$$

Thus, there is no autocorrelation either: $\rho_k = 0$ for all $k \neq 0$.

If in addition, the variables also follow a *Gaussian distribution*, i.e. $W_t \sim N(0, \sigma_W^2)$, the series is called **Gaussian White Noise**. The term White Noise is due to the analogy to white light.

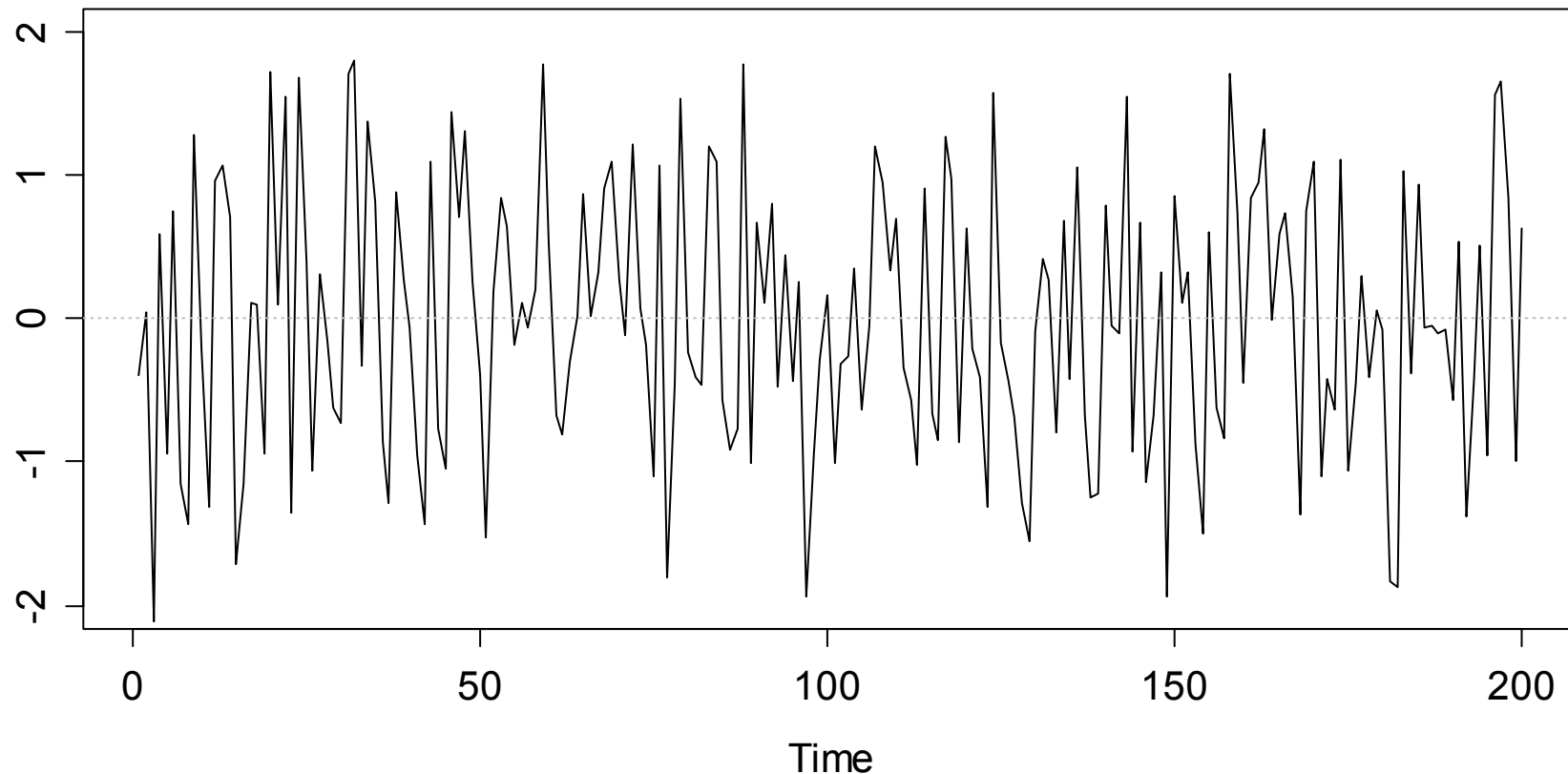
Applied Time Series Analysis

SS 2016 – Basics of Modeling

Example: Gaussian White Noise

```
> plot(ts(rnorm(200, mean=0, sd=1)))
```

Gaussian White Noise

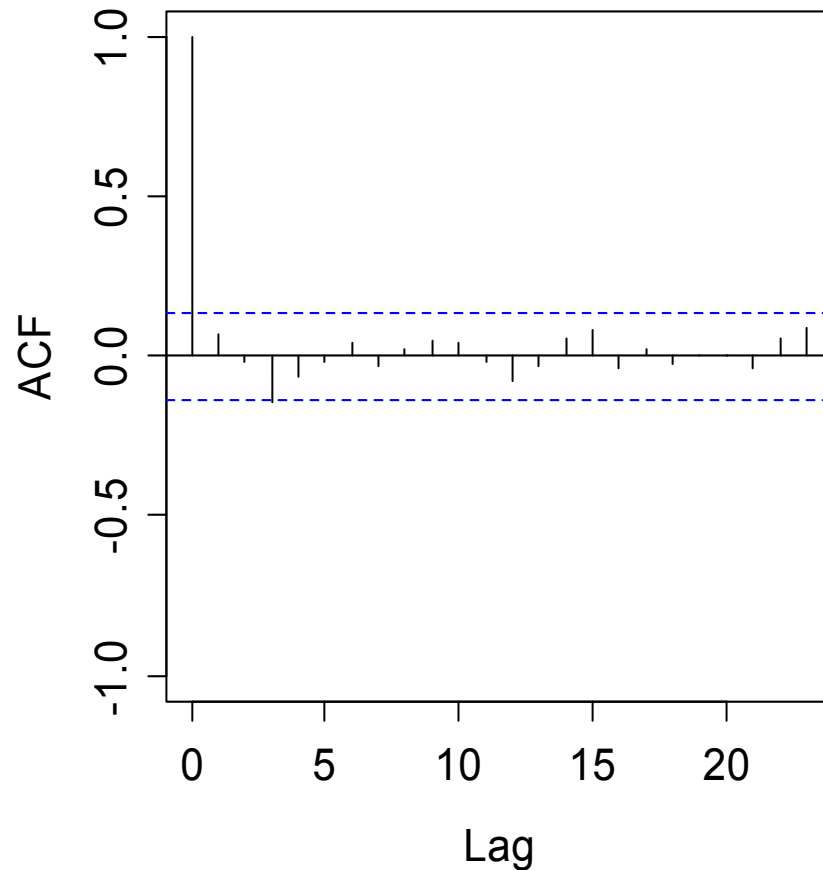


Applied Time Series Analysis

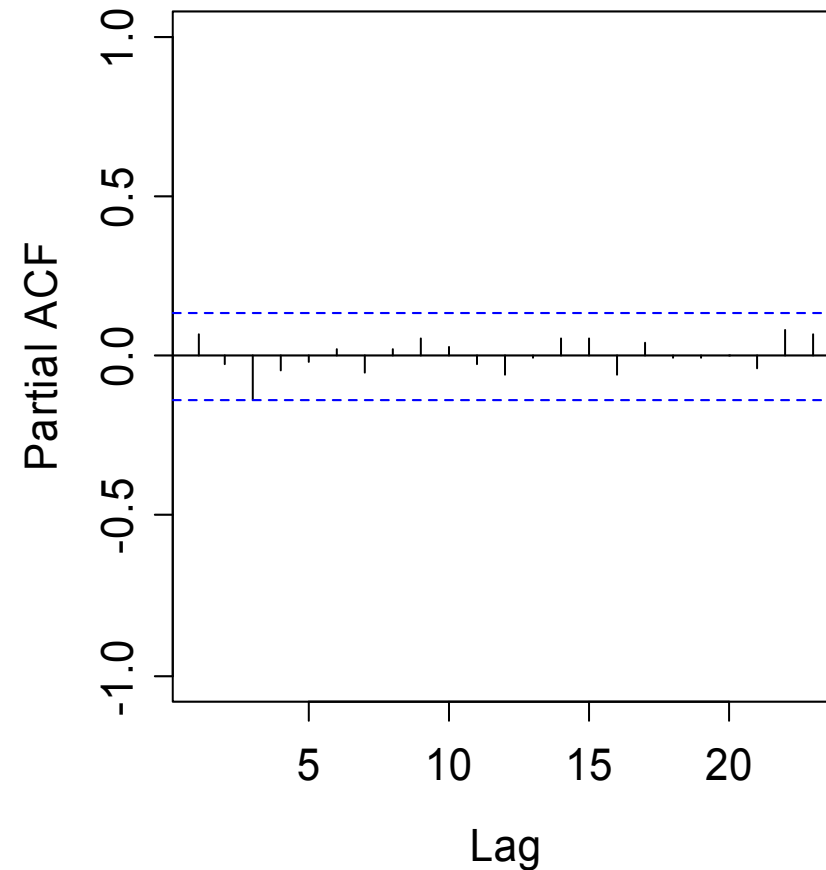
SS 2016 – Basics of Modeling

Example: Gaussian White Noise

ACF



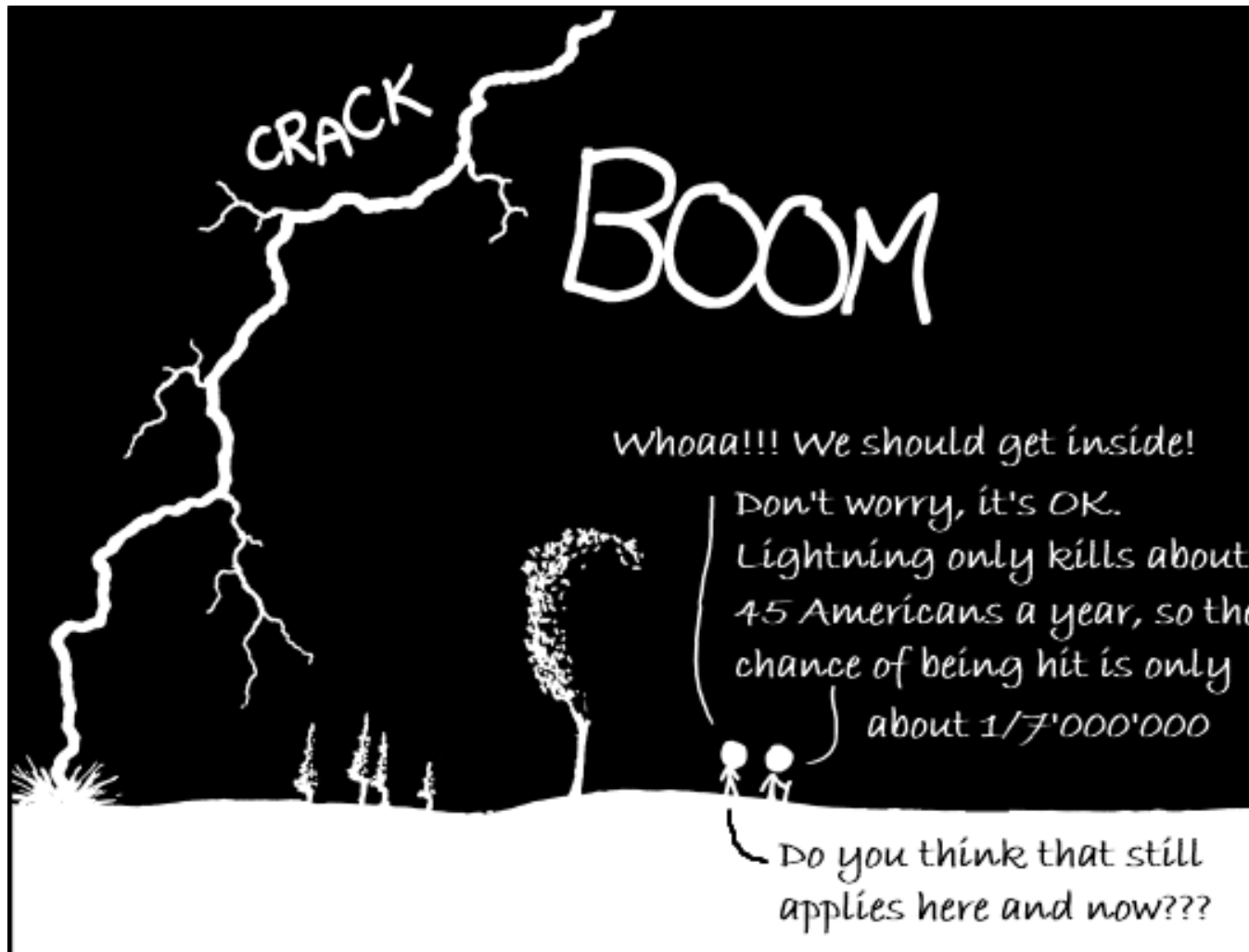
PACF



Applied Time Series Analysis

SS 2016 – Basics of Modeling

Estimating the Conditional Mean



→ see blackboard...

Applied Time Series Analysis

SS 2016 – Basics of Modeling

Time Series Modeling

There is a wealth of time series models

- AR autoregressive model
- MA moving average model
- ARMA combination of AR & MA
- ARIMA non-stationary ARMAs
- SARIMA seasonal ARIMAs
- ...

We start by discussing autoregressive models. They are perhaps the simplest and most intuitive time series models that exist.

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Basic Idea for AR(p)-Models

In an AR(p) process, the random variable X_t depends on an autoregressive linear combination of the preceding X_{t-1}, \dots, X_{t-p} , plus a „completely independent“ term called *innovation* E_t .

$$X_t = \alpha_1 X_{t-1} + \dots + \alpha_p X_{t-p} + E_t$$

Here, p is called the order of the AR model. Hence, we abbreviate by AR(p). An alternative notation is with the *backshift operator* B :

$$(1 - \alpha_1 B - \alpha_2 B^2 - \dots - \alpha_p B^p) X_t = E_t \quad \text{or short, } \Phi(B) X_t = E_t$$

Here, $\Phi(B)$ is called the *characteristic polynomial* of the AR(p). It determines most of the relevant properties of the process.

Applied Time Series Analysis

SS 2016 – Autoregressive Models

AR(1)-Model

The simplest autoregressive model is the AR(1) process:

$$X_t = \alpha_1 X_{t-1} + E_t$$

where

E_t is *iid* with $E[E_t] = 0$ and $Var(E_t) = \sigma_E^2$

We also require that E_t is independent of $X_s, s < t$

Under these conditions, E_t is a **causal White Noise** process, or an **innovation**. Be aware that this is stronger than the iid requirement: not every *iid* process is an innovation and that property is absolutely central to AR(p)-modelling.

Applied Time Series Analysis

SS 2016 – Autoregressive Models

AR(p)-Models and Stationarity

The following is absolutely essential:

AR(p) models must only be fitted to stationary time series. Any potential trends and/or seasonal effects need to be removed first. We will also make sure that the processes are stationary.

Under which circumstances is an AR(p) stationary?

→ **see blackboard...**

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Stationarity of AR(p)-Processes

As we have seen, any stationary AR(p) process meets:

1) $E[X_t] = \mu = 0$

2) The condition on $(\alpha_1, \dots, \alpha_p)$:

All (complex) roots of the characteristic polynomial

$$1 - \alpha_1 z - \alpha_2 z^2 - \dots - \alpha_p z^p = 0$$

must exceed 1 in absolute value (verifiable with `polyroot()`)

We can always shift a stationary AR(p) process: $Y_t = m + X_t$

The resulting process is still stationary and allows for greater flexibility in modelling. It is a **shifted AR(p) process**.

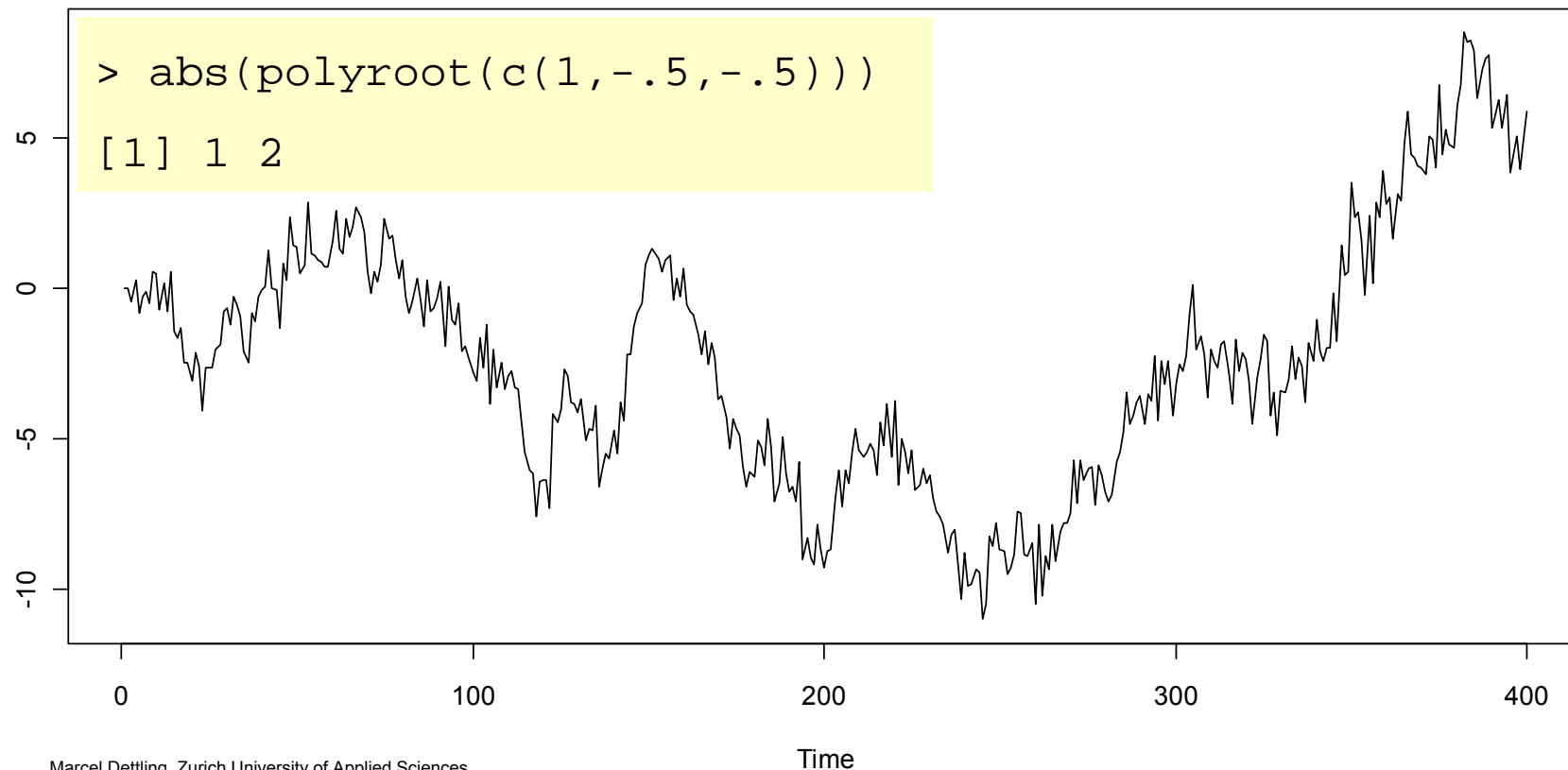
Applied Time Series Analysis

SS 2016 – Autoregressive Models

A Non-Stationary AR(2)-Process

$$X_t = \frac{1}{2} X_{t-1} + \frac{1}{2} X_{t-2} + E_t \text{ is not stationary...}$$

Non-Stationary AR(2)

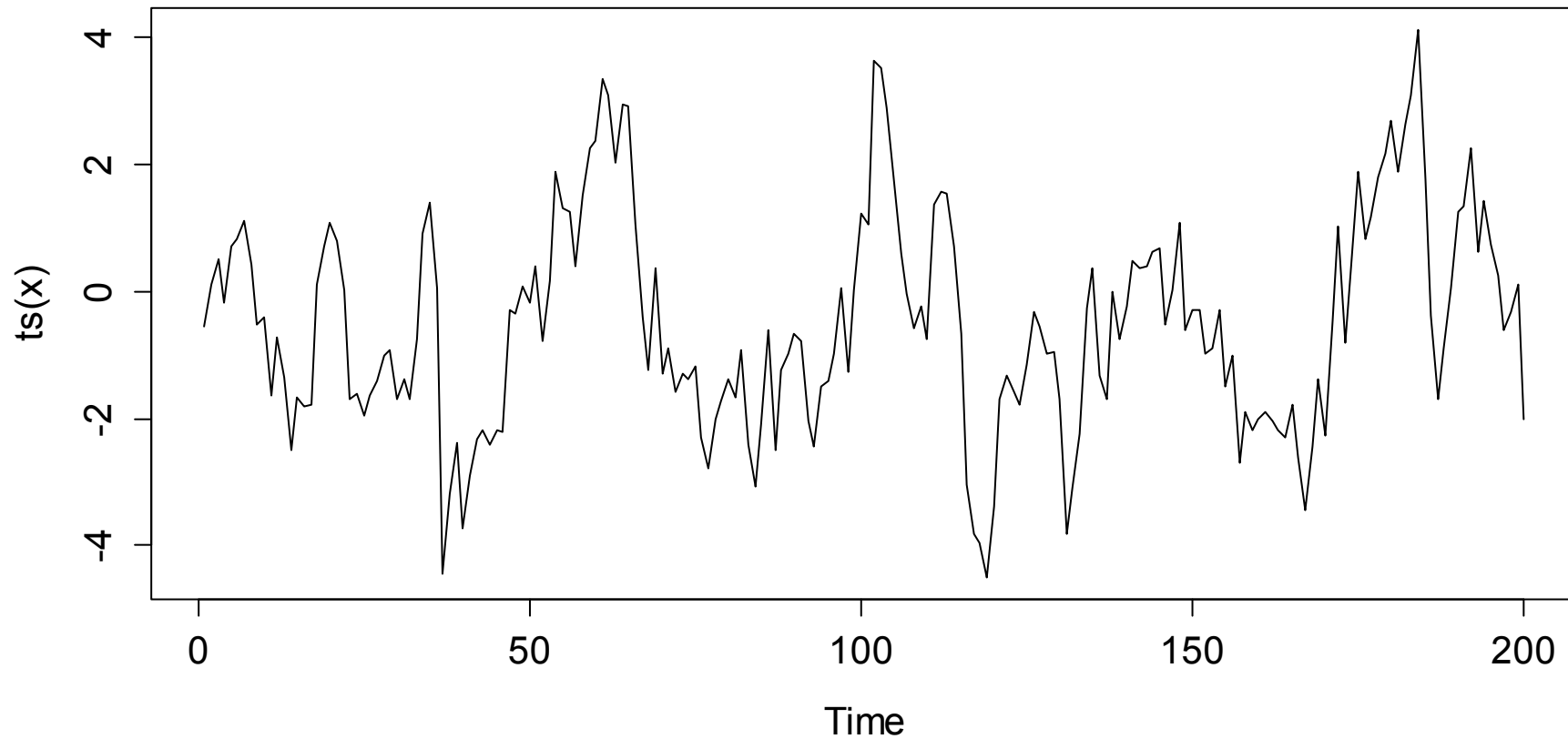


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Simulated AR(1)-Series

AR(1) with $\alpha_1=0.8$

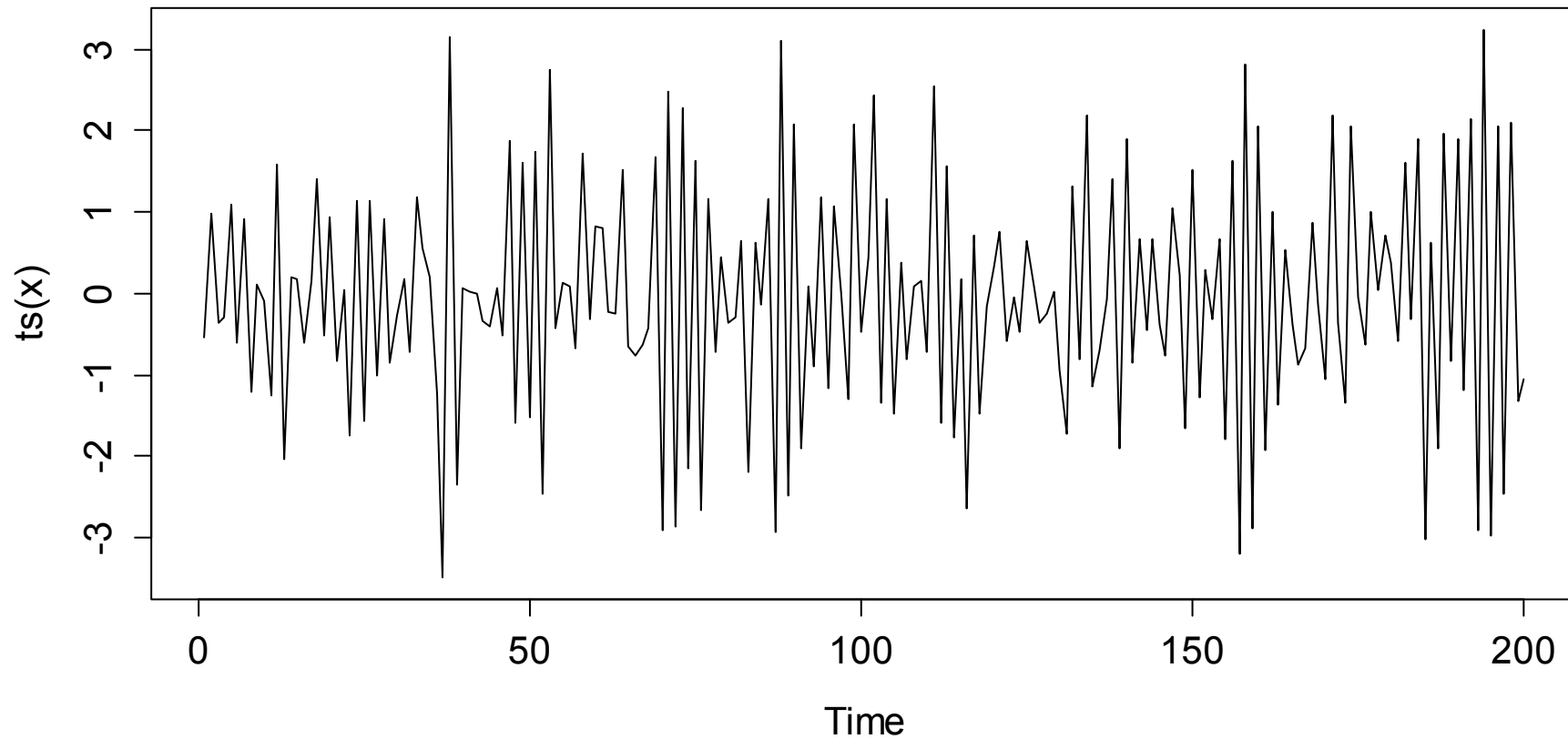


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Simulated AR(1)-Series

AR(1) with $\alpha_1 = -0.8$



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Autocorrelation of AR(p) Processes

→ See the blackboard...

Yule-Walker Equations

We observe that there exists a linear equation system built up from the AR(p)-coefficients and the ACF-coefficients of up to lag p . These are called Yule-Walker-Equations.

We can use these equations for fitting an AR(p)-model:

- 1) *Estimate the ACF from a time series*
- 2) *Plug-in the estimates into the Yule-Walker-Equations*
- 3) *The solution are the AR(p)-coefficients*

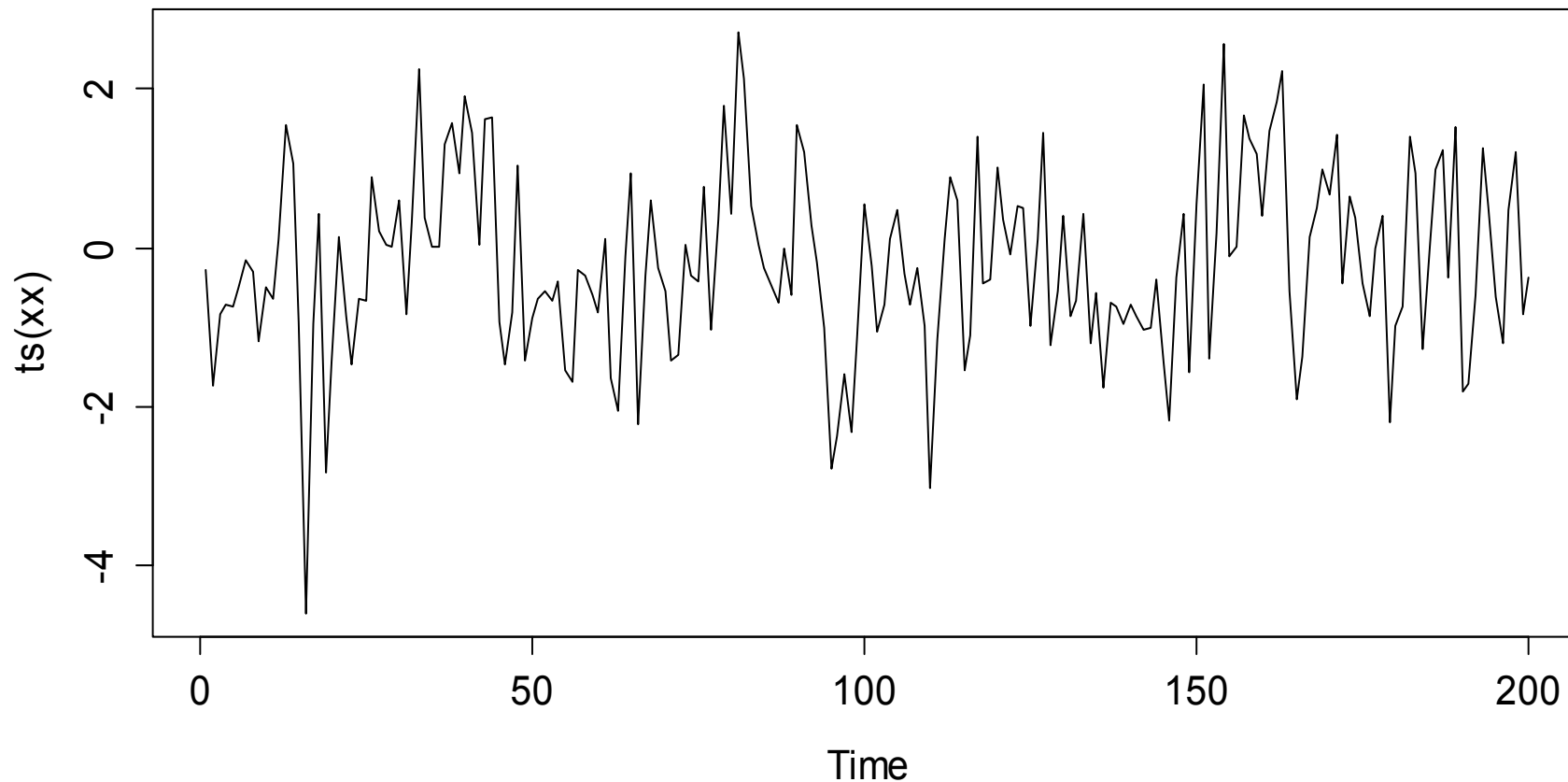
Applied Time Series Analysis

SS 2016 – Autoregressive Models

AR(3): Simulation and Properties

```
> xx <- arima.sim(list(ar=c(0.4, -0.2, 0.3)))
```

AR(3) with $\alpha_1=-0.4$, $\alpha_2=-0.2$, $\alpha_3=0.3$



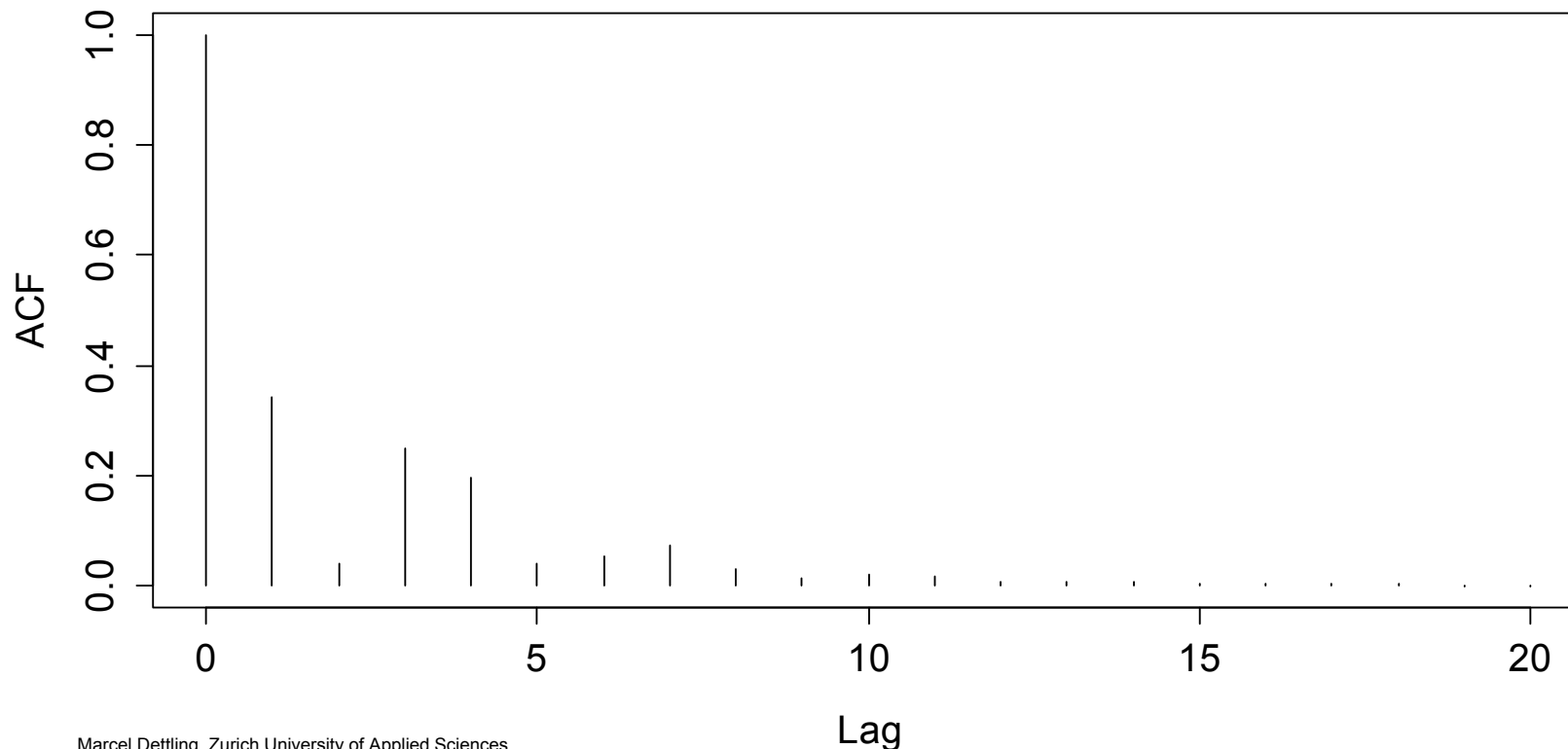
Applied Time Series Analysis

SS 2016 – Autoregressive Models

AR(3): Simulation and Properties

```
> autocorr <- ARMAacf(ar=c(0.4, -0.2, 0.3),...)  
> plot(0:20, autocorr, type="h", xlab="Lag")
```

Theoretical Autocorrelation for an AR(3)



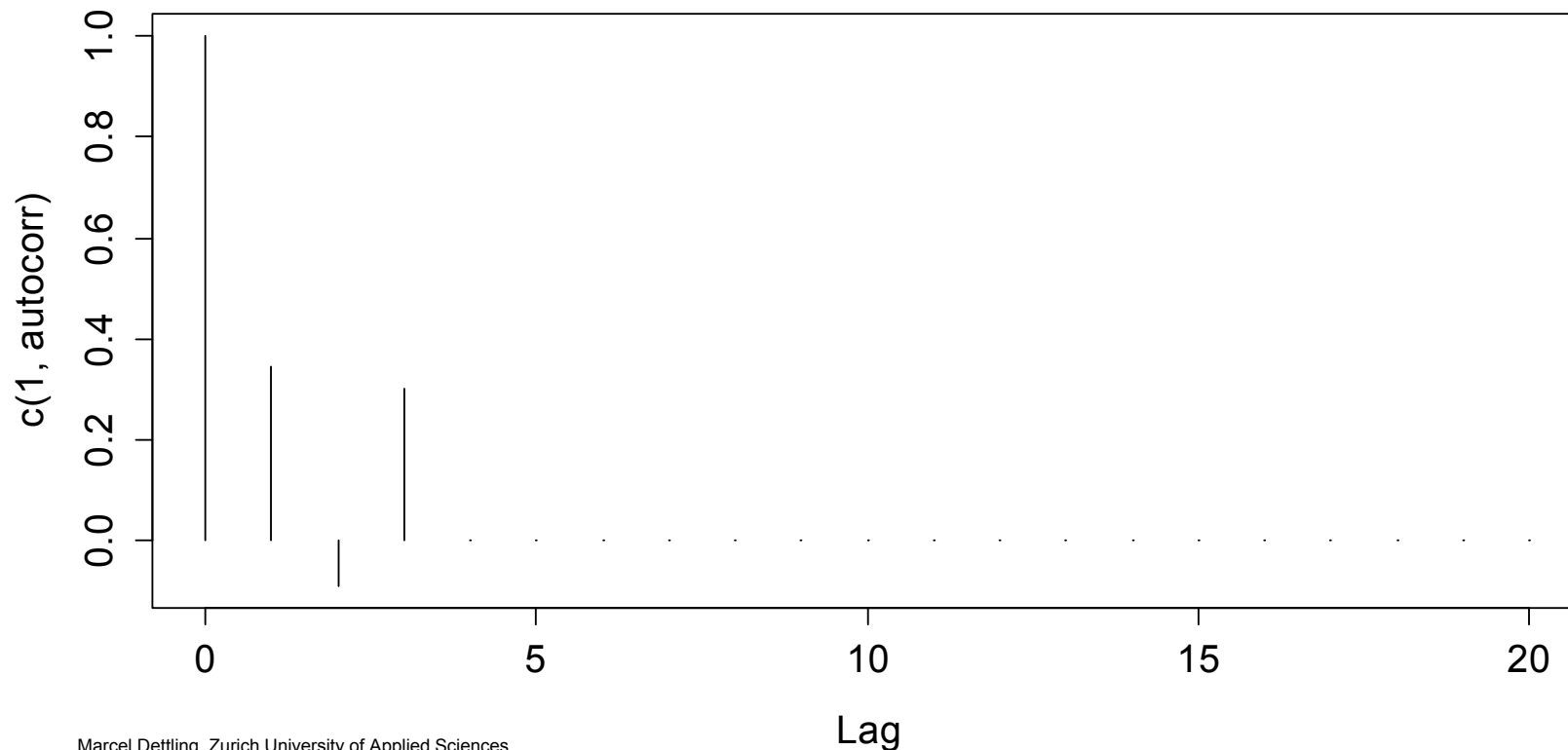
Applied Time Series Analysis

SS 2016 – Autoregressive Models

AR(3): Simulation and Properties

```
> autocorr <- ARMAacf(ar=..., pacf=TRUE, ...)  
> plot(0:20, autocorr, type="h", xlab="Lag")
```

Theoretical Partial Autocorrelation for an AR(3)



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Fitting AR(p)-Models

This involves 3 crucial steps:

1) **Model Identification**

- is an AR process suitable, and what is p ?
- will be based on ACF/PACF-Analysis

2) **Parameter Estimation**

- Regression approach
- Yule-Walker-Equations
- and more (MLE, Burg-Algorithm)

3) **Residual Analysis**

- to be discussed

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Model Identification

- AR(p) processes are stationary
- For all AR(p) processes, the **ACF** decays exponentially quickly, or is an exponentially damped sinusoid.
- For all AR(p) processes, the **PACF** is equal to zero for all lags $k > p$. The behavior before lag p can be arbitrary.

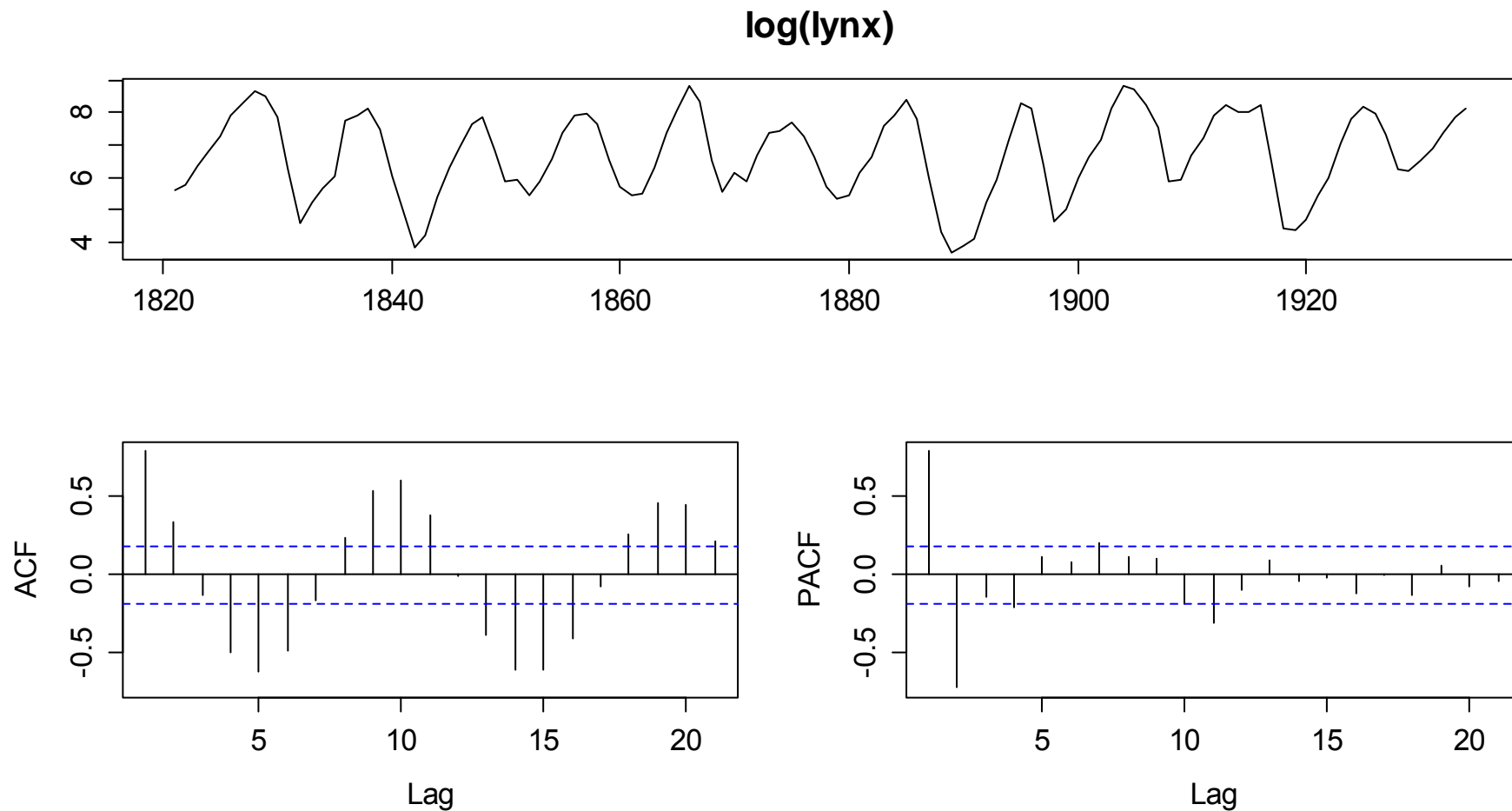
If what we observe is fundamentally different from the above, it is unlikely that the series was generated from an AR(p)-process. We thus need other models, maybe more sophisticated ones.

Remember that the sample ACF has a few peculiarities (bias, variability, compensation issue) and is tricky to interpret!!!

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Model Order for $\log(\text{lynx})$



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Parameter Estimation for AR(p)

Observed time series are rarely centered. Then, it is inappropriate to fit a pure AR(p) process. All R routines by default assume the shifted process $Y_t = m + X_t$. Thus, we face the problem:

$$(Y_t - m) = \alpha_1(Y_{t-1} - m) + \dots + \alpha_p(Y_{t-p} - m) + E_t$$

The goal is to estimate the *global mean* m , the *AR-coefficients* $\alpha_1, \dots, \alpha_p$, and some parameters defining the distribution of the innovation E_t . We usually assume a Gaussian, hence this is σ_E^2 .

We will discuss 4 methods for estimating the parameters:

OLS, Burg's algorithm, Yule-Walker, MLE

Applied Time Series Analysis

SS 2016 – Autoregressive Models

OLS Estimation

If we rethink the previously stated problem:

$$(Y_t - m) = \alpha_1 (Y_{t-1} - m) + \dots + \alpha_p (Y_{t-p} - m) + E_t$$

we recognize a multiple linear regression problem without intercept on the centered observations. What we need to do is:

- 1) Estimate $\hat{m} = \bar{y} = \frac{1}{n} \sum_{t=1}^n y_t$ and determine $x_t = y_t - \hat{m}$
- 2) Run a regression w/o intercept on x_t to obtain $\hat{\alpha}_1, \dots, \hat{\alpha}_p$
- 3) For $\hat{\sigma}_E^2$, take the residual standard error from the output.

This all works without any time series software, but is a bit cumbersome to implement. Dedicated procedures exist...

Applied Time Series Analysis

SS 2016 – Autoregressive Models

OLS Estimation

```
> f.ols <- ar.ols(llynx, aic=F, inter=F, order=2)
```

```
> f.ols
```

```
Coefficients:
```

```
          1          2  
1.3844 -0.7479
```

```
Order selected 2  sigma^2 estimated as  0.2738
```

```
> f.ols$x.mean
```

```
[1] 6.685933
```

```
> sum(na.omit(f.ols$resid)^2)/112
```

```
[1] 0.2737594
```

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Burg's Algorithm

While OLS works, the first p instances are never evaluated as responses. This is cured by Burg's algorithm, which uses the property of time-reversal in stochastic processes. We thus evaluate the RSS of forward and backward prediction errors:

$$\sum_{t=p+1}^n \left\{ \left(X_t - \sum_{k=1}^p \alpha_k X_{t-k} \right)^2 + \left(X_{t-p} - \sum_{k=1}^p \alpha_k X_{t-p+k} \right)^2 \right\}$$

In contrast to OLS, there is no explicit solution and numerical optimization is required. This is done with a recursive method called the Durbin-Levison algorithm (implemented in R).

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Burg's Algorithm

```
> f.burg <- ar.burg(llynx, aic=F, order.max=2)
```

```
> f.burg
```

```
Coefficients:
```

```
          1          2  
1.3831  -0.7461
```

```
Order selected 2  sigma^2 estimated as  0.2707
```

```
> f.ar.burg$x.mean
```

```
[1] 6.685933
```

Note: The innovation variance is estimated from the Durbin-Levinson updates and not from the residuals using the MLE!

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Yule-Walker Equations

The Yule-Walker-Equations yield a LES that connects the true ACF with the true AR-model parameters. We plug-in the estimated ACF coefficients:

$$\hat{\rho}(k) = \hat{\alpha}_1 \hat{\rho}(k-1) + \dots + \hat{\alpha}_p \hat{\rho}(k-p) \text{ for } k = 1, \dots, p$$

and can solve the LES to obtain the AR-parameter estimates.

- \hat{m} is the arithmetic mean of the time series
- $\hat{\sigma}_E^2$ is obtained from the fitted coefficients via the autocovariance of the series and takes a different value than before!

There is an implementation in R with function `ar.yw()`.

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Yule-Walker Equations

```
> f.ar.yw
```

```
Call: ar.yw.default(x = log(lynx), aic = FALSE,  
order.max = 2)
```

```
Coefficients:
```

```
      1      2  
1.3504 -0.7200
```

```
Order selected 2  sigma^2 estimated as  0.3109
```

While the Yule-Walker method is asymptotically equivalent to OLS and Burg's algorithm, it generally yields a solution with worse Gaussian likelihood on finite samples

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Maximum-Likelihood-Estimation

Idea: Determine the parameters such that, given the observed time series (y_1, \dots, y_n) , the resulting model is the most plausible (i.e. the most likely) one.

This requires the choice of a probability model for the time series. By assuming Gaussian innovations, $E_t \sim N(0, \sigma_E^2)$, any AR(p) process has a multivariate normal distribution:

$$Y = (Y_1, \dots, Y_n) \sim N(m \cdot \underline{1}, V), \text{ with } V \text{ depending on } \underline{\alpha}, \sigma_E^2$$

MLE then provides simultaneous estimates by optimizing:

$$L(\alpha, m, \sigma_E^2) \propto \exp\left(\sum_{t=1}^n (x_t - \hat{x}_t)^2\right)$$

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Maximum-Likelihood-Estimation

```
> f.ar.mle
```

```
Call: arima(x = log(lynx), order = c(2, 0, 0))
```

```
Coefficients:
```

	ar1	ar2	intercept
	1.3776	-0.7399	6.6863
s.e.	0.0614	0.0612	0.1349

```
sigma^2=0.271; log likelihood=-88.58; aic=185.15
```

While MLE by default assumes Gaussian innovations, it still performs reasonably for other distributions as long as they are not extremely skewed or have very precarious outliers.

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Practical Aspects

- All 4 estimation methods are asymptotically equivalent and even on finite samples, the differences are usually small.
- All 4 estimation methods are non-robust against outliers and perform best on data that are approximately Gaussian.
- Function `arima()` provides standard errors for \hat{m} ; $\hat{\alpha}_1, \dots, \hat{\alpha}_p$ so that statements about significance become feasible and confidence intervals for the parameters can be built.
- `ar.ols()`, `ar.yw()` & `ar.burg()` allow for convenient choice of the optimal model order p using the AIC criterion. Among these methods, `ar.burg()` is usually preferred.

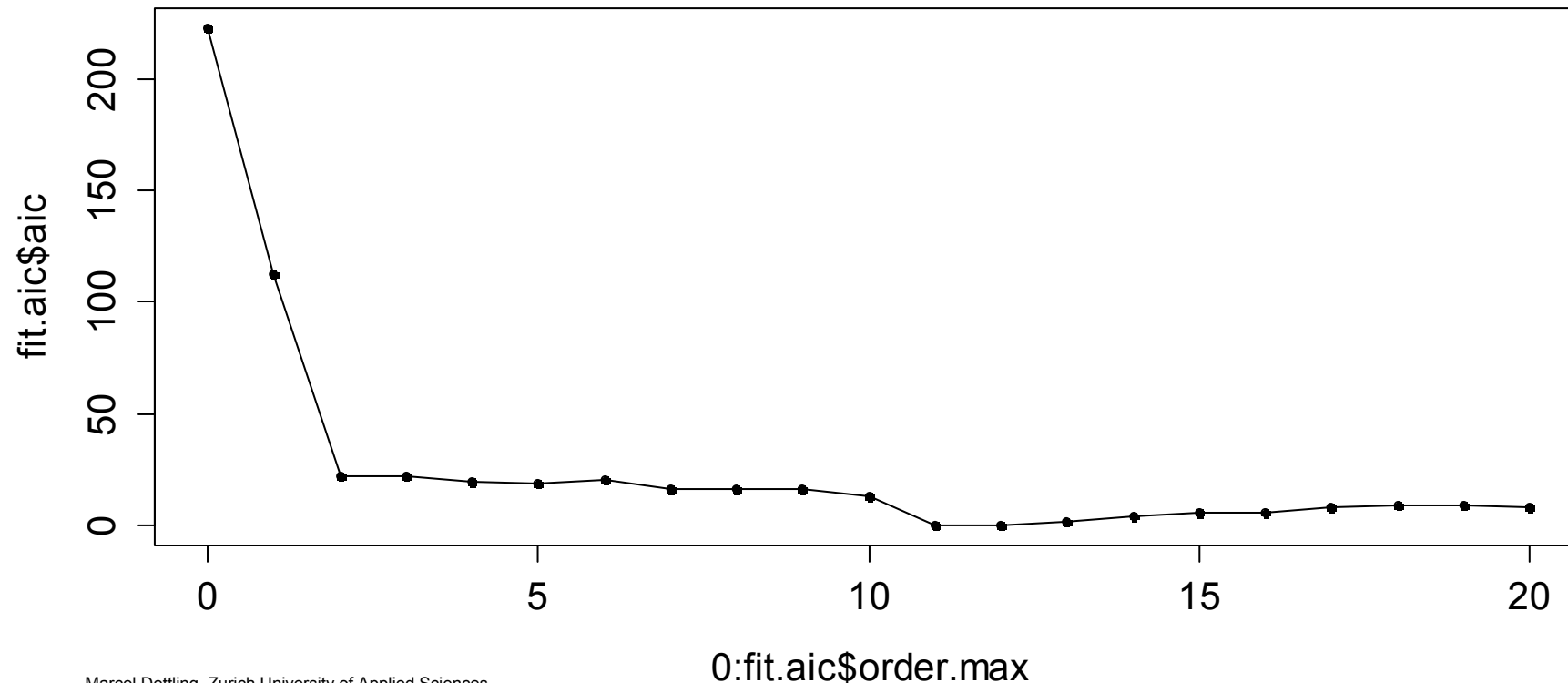
Applied Time Series Analysis

SS 2016 – Autoregressive Models

AIC-Based Order Choice

```
> fit.aic <- ar.burg(log(lynx))  
> plot(0:fit.aic$order.max, fit.aic$aic)
```

AIC-Values for AR(p)-Models on the Logged Lynx Data

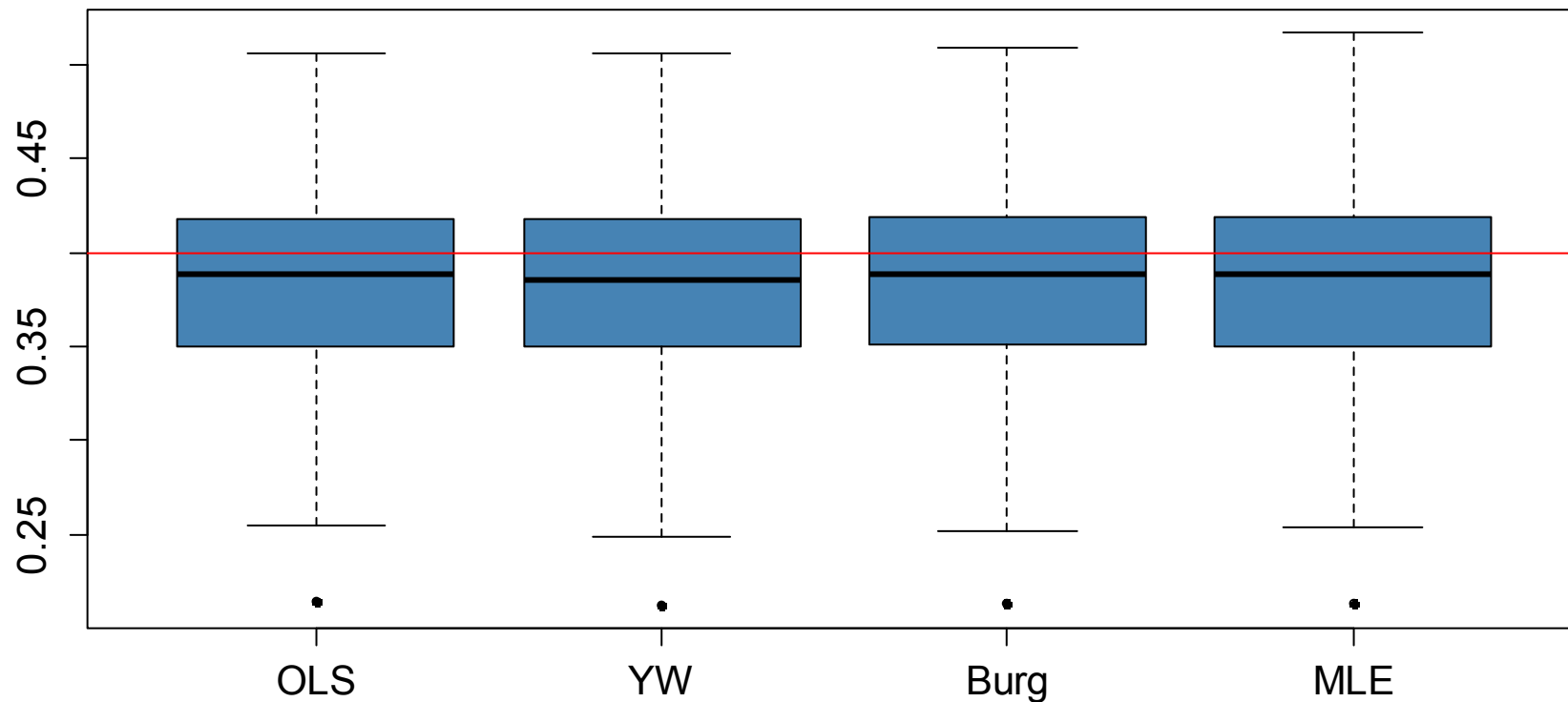


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Comparison: Alpha Estimation vs. Method

Estimates $\hat{\alpha}_1$ for an AR(1) with $n=200$ and $\alpha_1=0.4$

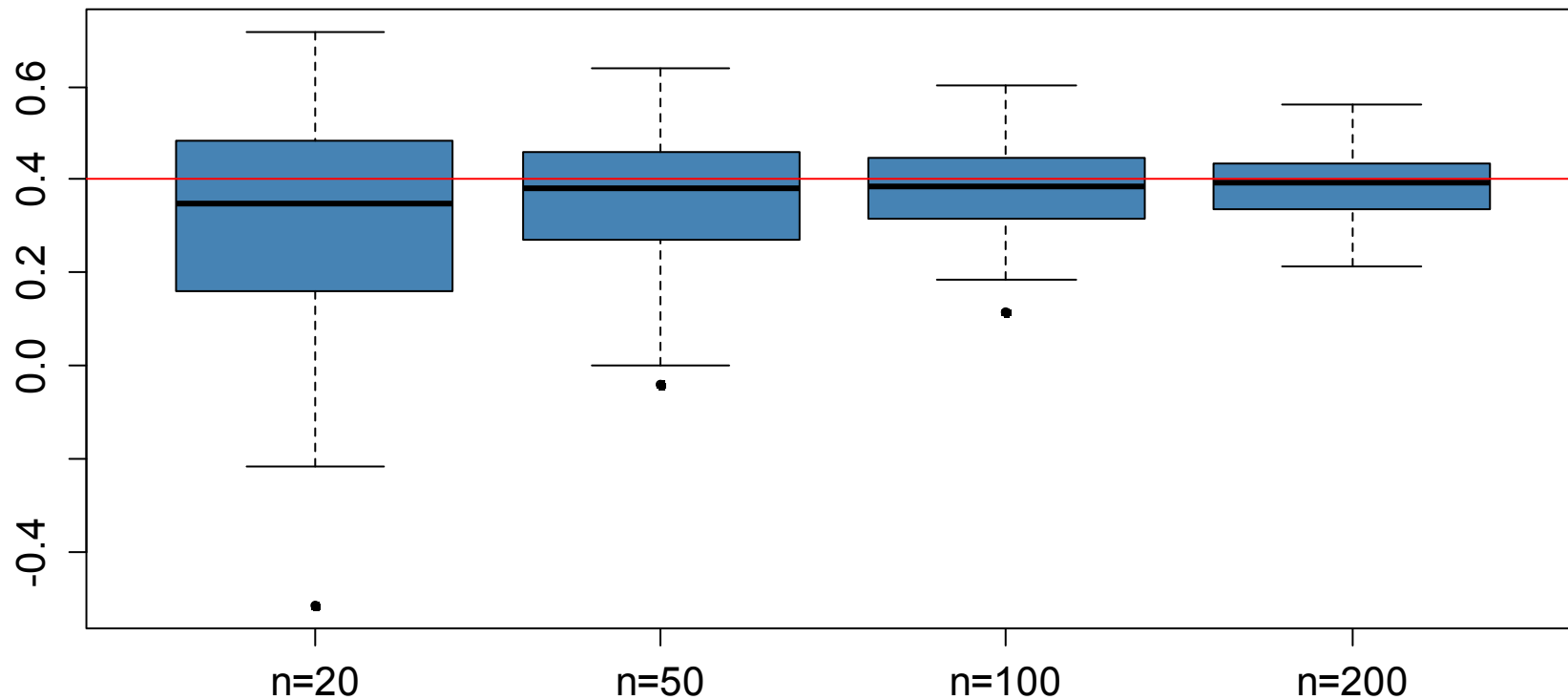


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Comparison: Alpha Estimation vs. n

Estimates $\hat{\alpha}_1$ for an AR(1) with Burg and $\alpha_1=0.4$

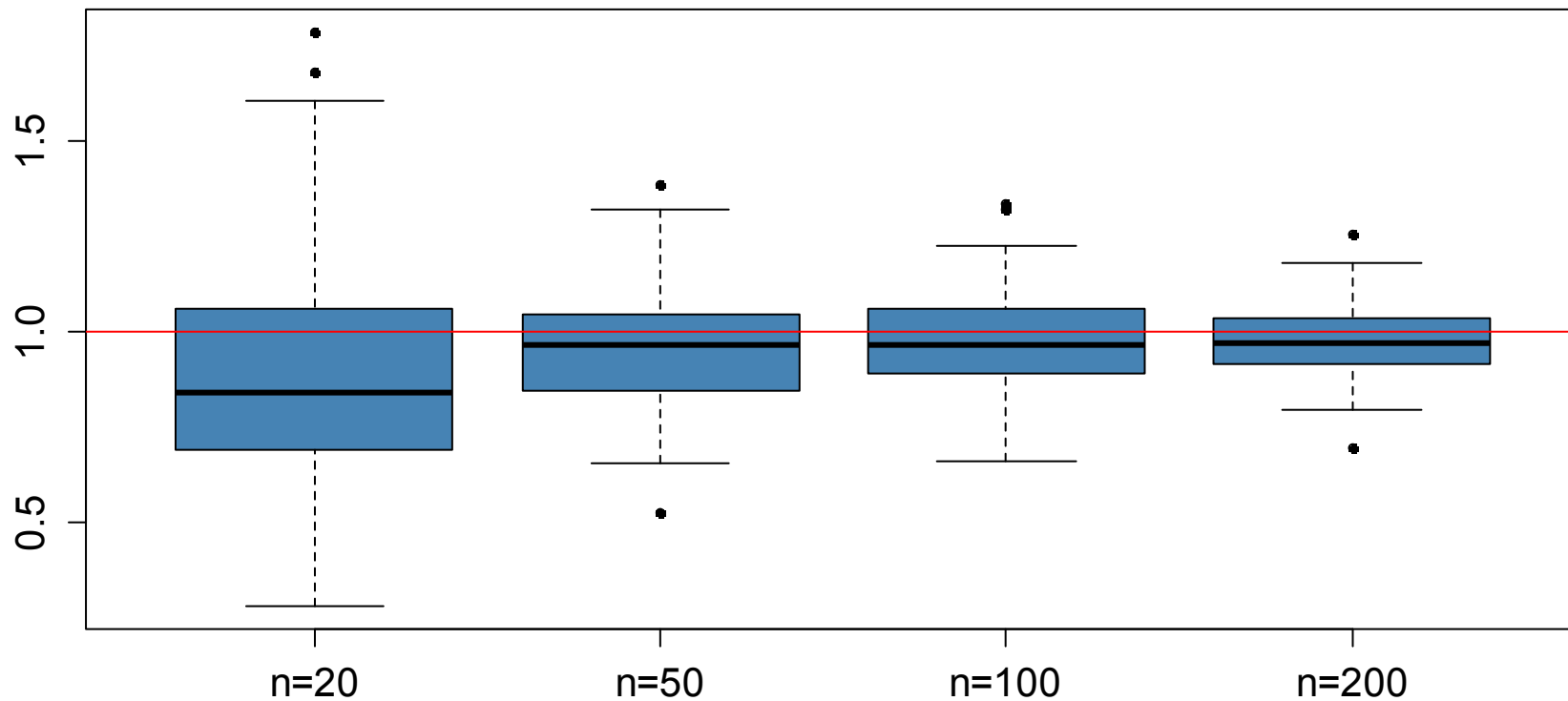


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Comparison: Sigma Estimation vs. n

Estimates $\hat{\sigma}_E^2$ for an AR(1) with Burg and $\sigma_E^2=1$



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Model Diagnostics

What we do here is Residual Analysis:

$$\begin{aligned}\text{„residuals“} &= \text{„estimated innovations“} \\ &= \hat{E}_t \\ &= (y_t - \hat{m}) - \left(\hat{\alpha}_1 (y_{t-1} - \hat{m}) - \dots - \hat{\alpha}_p (y_{t-p} - \hat{m}) \right)\end{aligned}$$

Remember the assumptions we made:

$$E_t \text{ i.i.d, } E[E_t] = 0, \text{ Var}(E_t) = \sigma_E^2$$

and probably

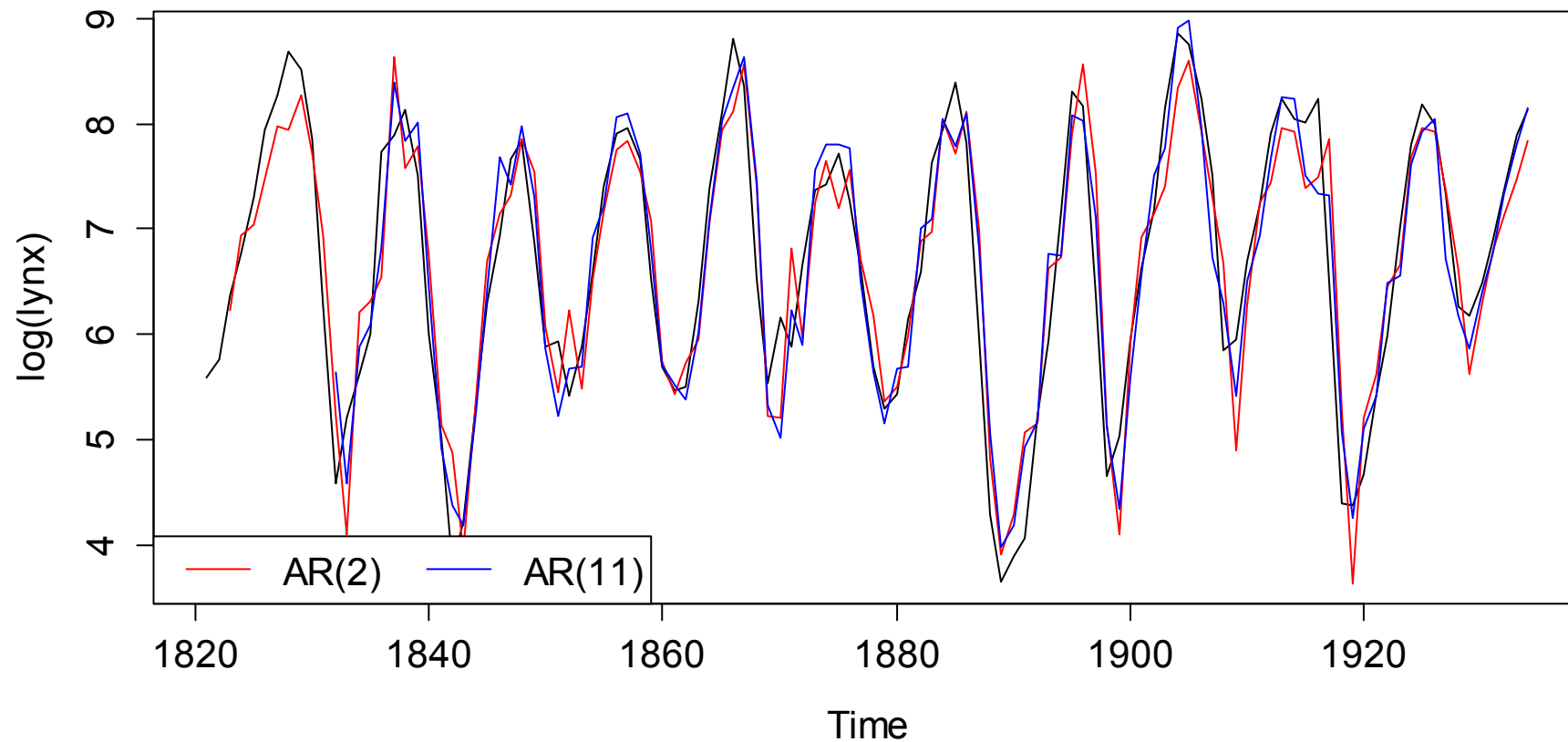
$$E_t \sim N(0, \sigma_E^2)$$

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Plotting the Fit

Logged Lynx Data with AR(2) and AR(11)



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Residual Analysis

We check the assumptions we made with the following plots:

a) Time series plot of \hat{E}_t

b) ACF/PACF plot of \hat{E}_t

c) QQ-plot of \hat{E}_t

→ **The innovation time series \hat{E}_t should look like White Noise**

Lynx example:

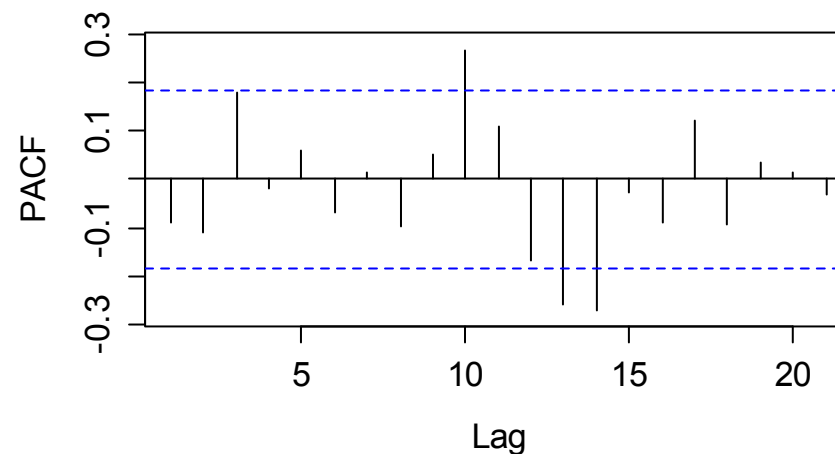
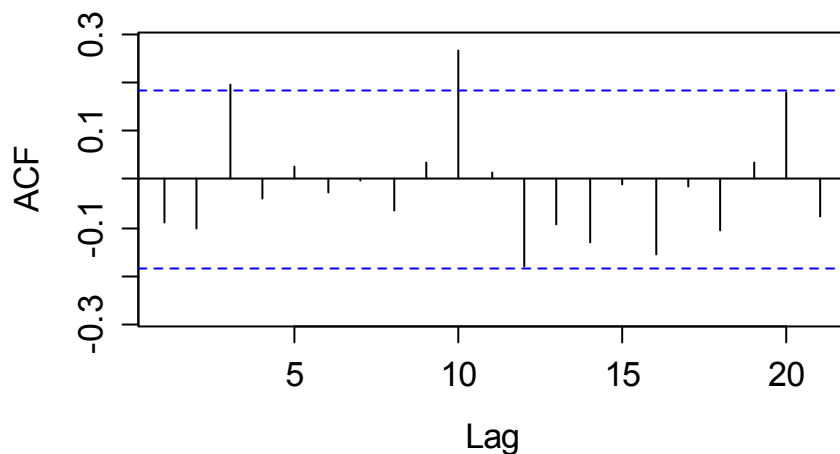
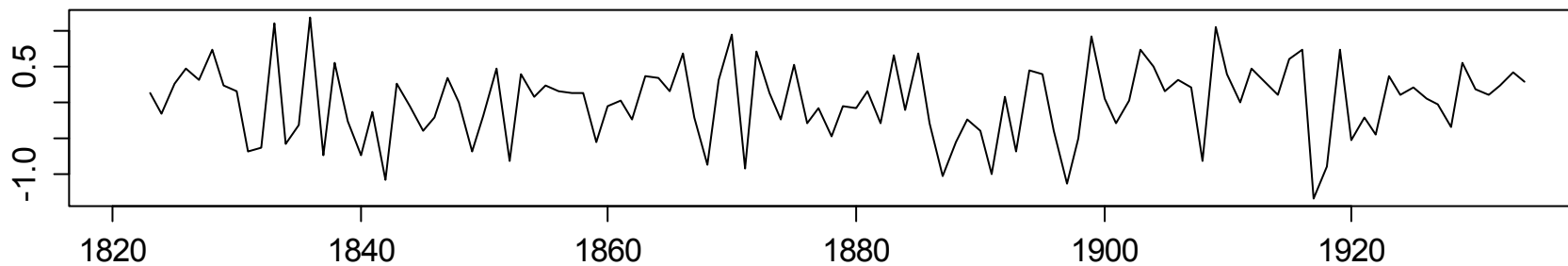
```
fit <- arima(log(lynx), order=c(2,0,0))  
tsdisplay(resid(fit))
```


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Residual Analysis for AR(2) Model

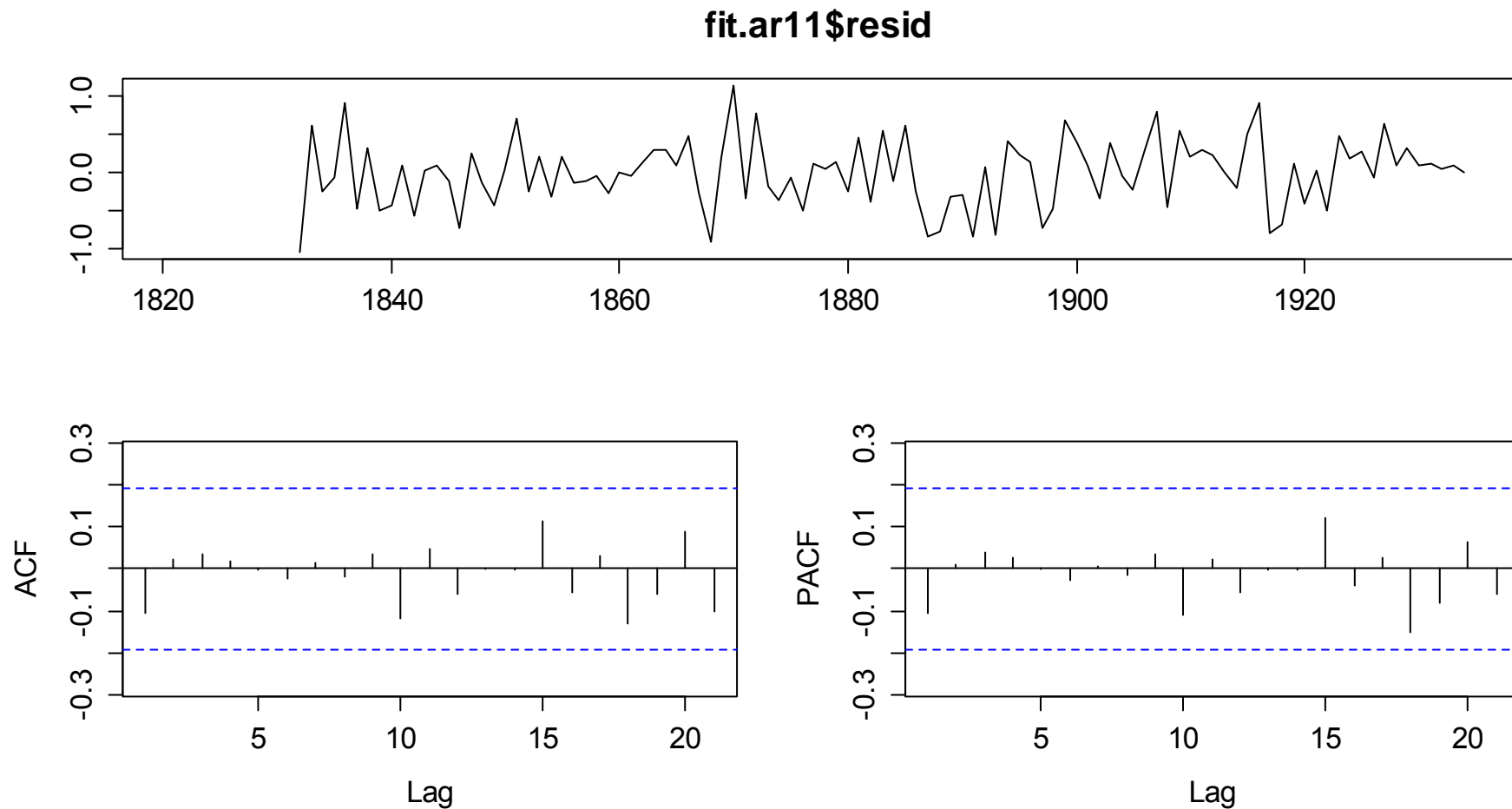
fit.ar02\$resid



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Residual Analysis for AR(11) Model

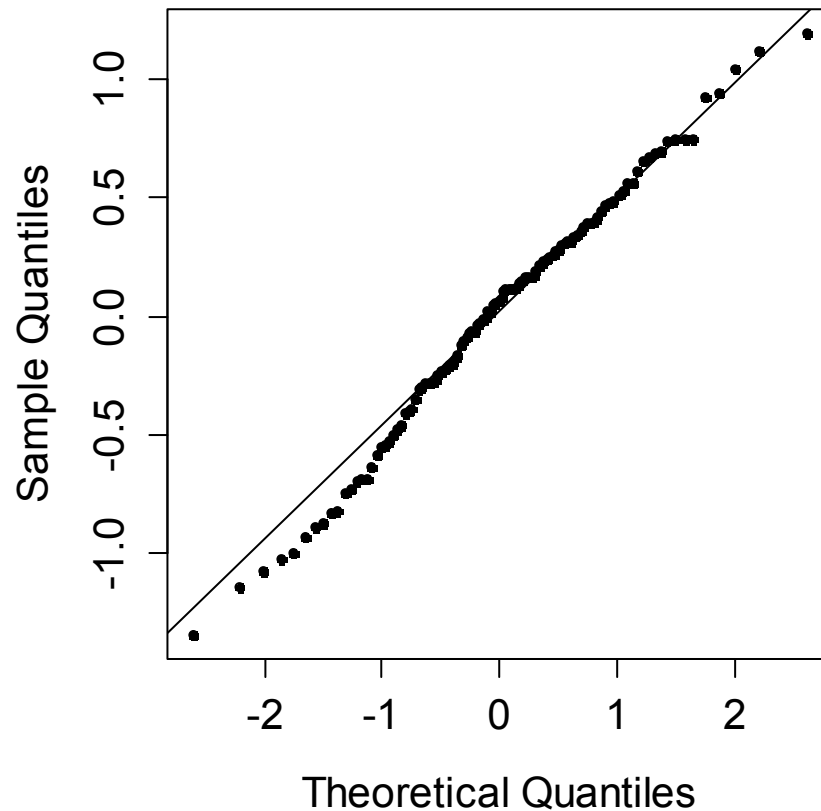


Applied Time Series Analysis

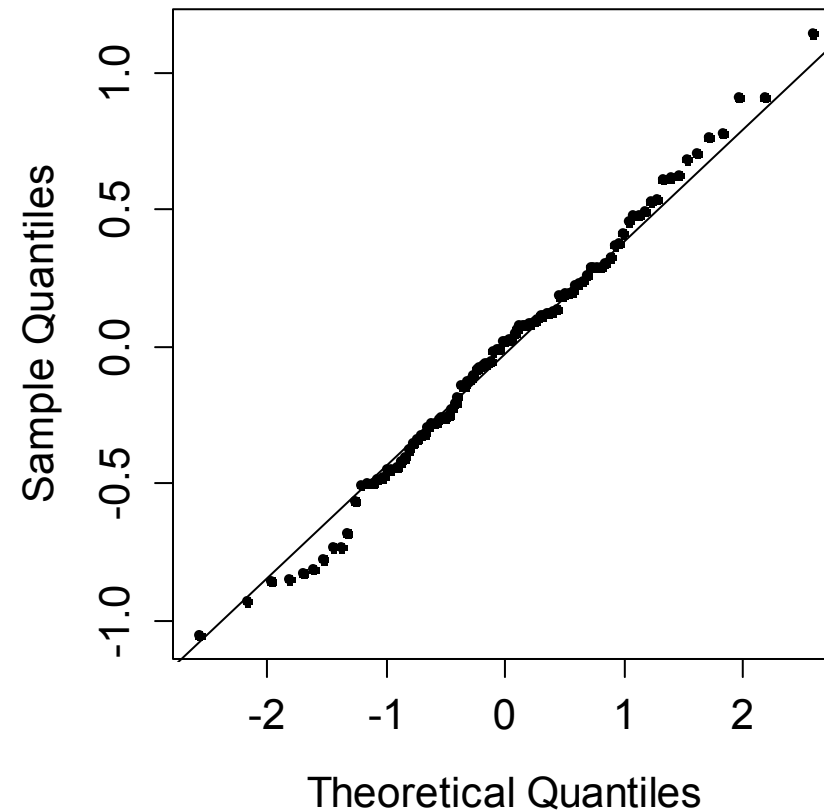
SS 2016 – Autoregressive Models

Normal Plots for AR(2) & AR(11)

Normal QQ-Plot: AR(2)



Normal QQ-Plot: AR(11)



Applied Time Series Analysis

SS 2016 – Autoregressive Models

Diagnostics by Simulation

As a last check before a model is called appropriate, simulating from the estimated coefficients and visually inspecting the resulting series (without any prejudices) to the original one can be beneficial.

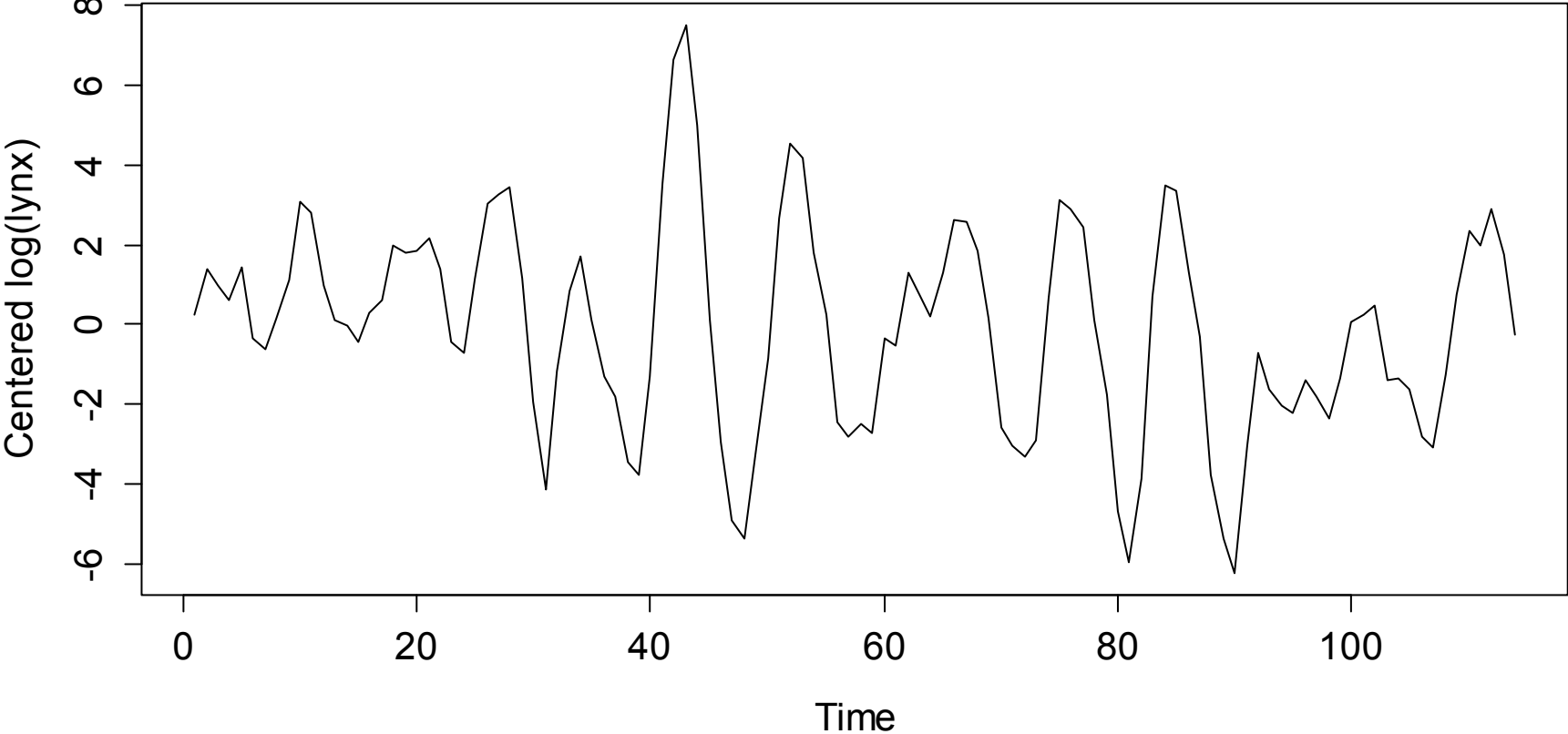
- **The simulated series should „look like“ the original. If this is not the case, the model failed to capture (some of) the properties in the original data.**
- **A larger or more sophisticated model may be necessary in cases where simulation does not recapture the features in the original data.**

Applied Time Series Analysis

SS 2016 – Autoregressive Models

Diagnostics by Simulation: AR(2)

Simulated Series from the Fitted AR(2)

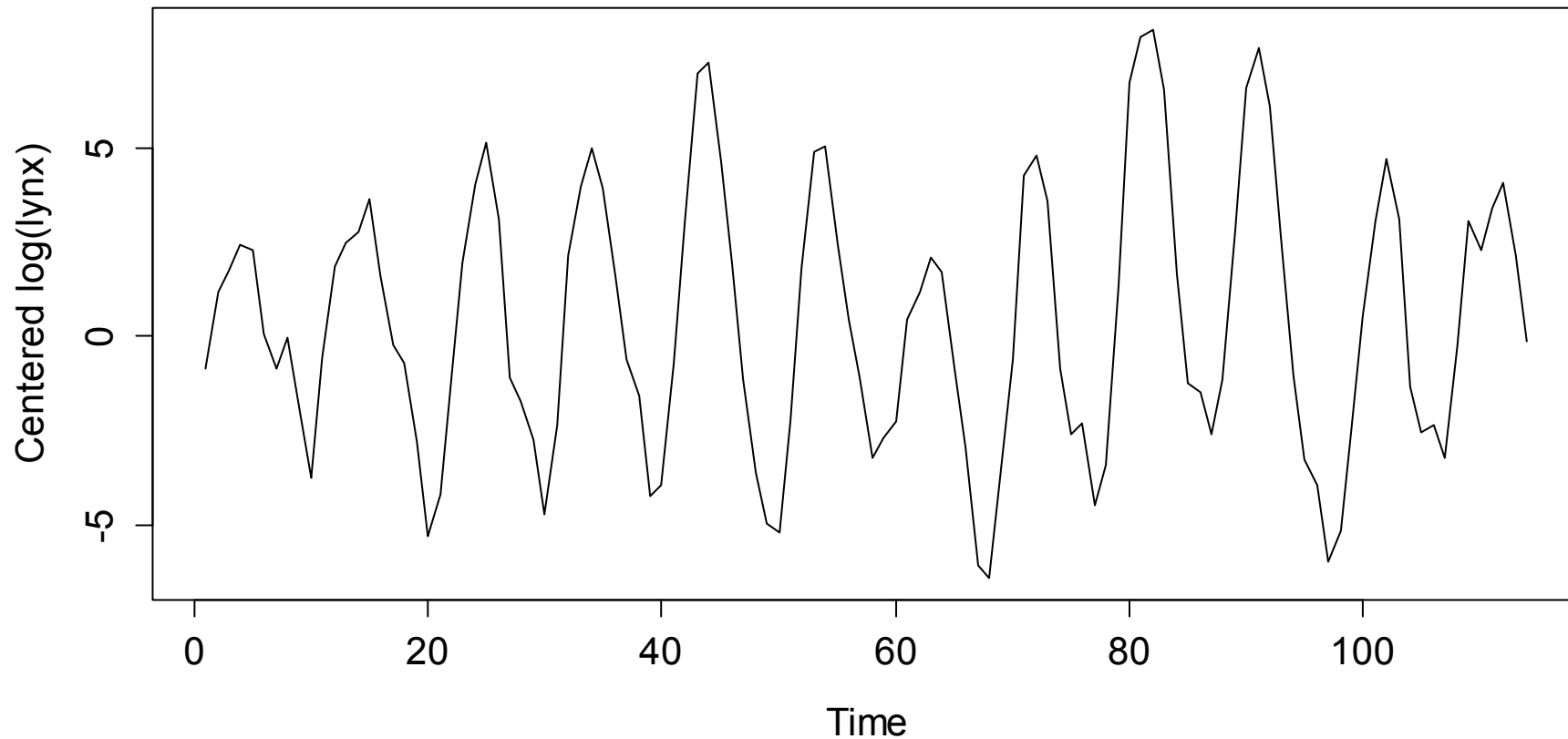


Applied Time Series Analysis

SS 2016 – Autoregressive Models

Diagnostics by Simulation: AR(11)

Simulated Series from the Fitted AR(11)



Applied Time Series Analysis

SS 2016 – Moving Average Models

Looking Back & Outlook

We did consider shifted **AR(p)-models** $Y_t = m + X_t$ with:

$$X_t = \alpha_1 X_{t-1} + \dots + \alpha_p X_{t-p} + E_t$$

where the correlation structure was as follows:

ACF: „exponential decay“

PACF: = 0 for all lags $k > p$

Now, in practice we could well observe a time series whose autocorrelation differs from the above structure.

We will thus discuss **ARMA(p,q) models**, a class that is suitable for modeling a wider spectrum of dependency structures.

Applied Time Series Analysis

SS 2016 – Moving Average Models

Moving Average Models

Whereas for AR(p) models, the current observation of a time series is written as a linear combination of its own past, **MA(q) models** can be seen as an extension of the „pure“ process

$$X_t = E_t, \text{ where } E_t \text{ is a white noise process,}$$

in the sense that past innovation terms E_{t-1}, E_{t-2}, \dots are included, too. We call this a **moving average** model:

$$X_t = E_t + \beta_1 E_{t-1} + \beta_2 E_{t-2} + \dots + \beta_q E_{t-q}$$

This is a time series process that is stationary, but not *iid*. In many respects, MA(q) models are complementary to AR(p).

Applied Time Series Analysis

SS 2016 – Moving Average Models

Notation for MA(q)-models

The backshift operator, and the characteristic polynomial, allow for convenient notation:

MA(q):
$$X_t = E_t + \beta_1 E_{t-1} + \beta_2 E_{t-2} + \dots + \beta_q E_{t-q}$$

MA(q) with BS:
$$X_t = \left(1 + \beta_1 B + \beta_2 B^2 + \dots + \beta_q B^q\right) E_t$$

MA(q) with BS+CP:
$$X_t = \Theta(B) E_t$$

where

$$\Theta(z) = 1 + \beta_1 z + \beta_2 z^2 + \dots + \beta_q z^q$$

is the characteristic polynomial

Applied Time Series Analysis

SS 2016 – Moving Average Models

Stationarity of MA(q)-Models

We first restrict ourselves to the simple MA(1)-model:

$$X_t = E_t + \beta_1 E_{t-1}, \text{ where } E_t \text{ is a White Noise innovation.}$$

The series X_t is always weakly stationary, no matter what the choice of the parameter β_1 is. Remember that for proving weak stationarity, we have to show that:

- the expected value is constant
- the variance is constant and finite
- the autocovariance only depends on the lag k

→ **see the blackboard for the proof**

Applied Time Series Analysis

SS 2016 – Moving Average Models

ACF of MA(q)-Processes

We can deduct the ACF for the MA(1)-process:

$$\rho(1) = \frac{\gamma(1)}{\gamma(0)} = \frac{\beta_1}{(1 + \beta_1^2)} < 0.5$$

and

$$\rho(k) = 0 \text{ for all } k > 1.$$

Thus, we have a “cut-off” situation, i.e. a similar behavior to the one of the PACF in an AR(1) process. This is why and how AR(1) and MA(1) are complementary.

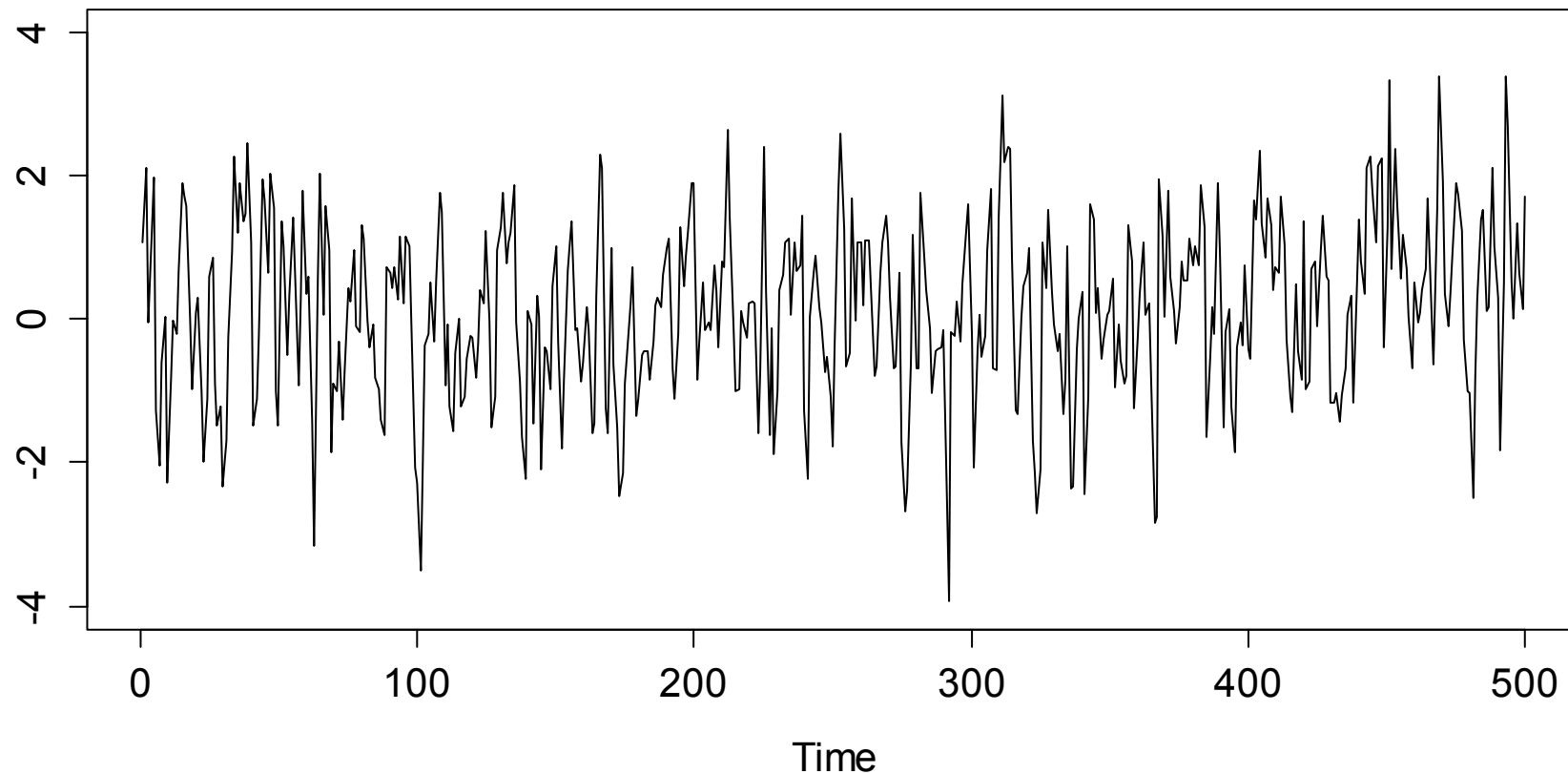
Applied Time Series Analysis

SS 2016 – Moving Average Models

Simulation of MA(1)

```
> ts.ma1 <- arima.sim(list(ma=0.7), n=500)
```

Simulation from a MA(1) Process



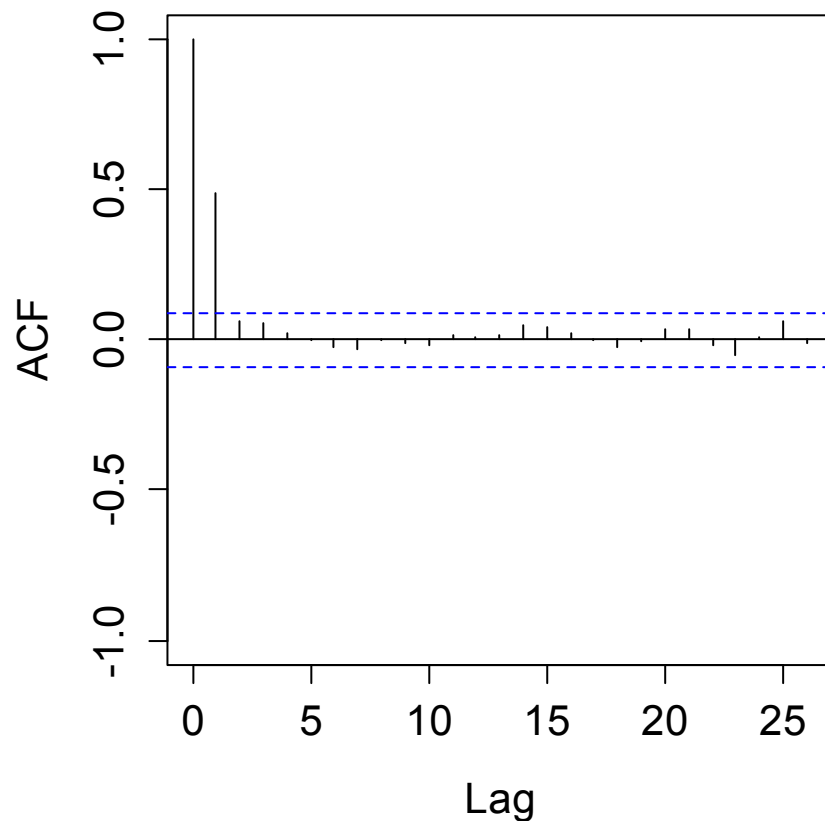
Applied Time Series Analysis

SS 2016 – Moving Average Models

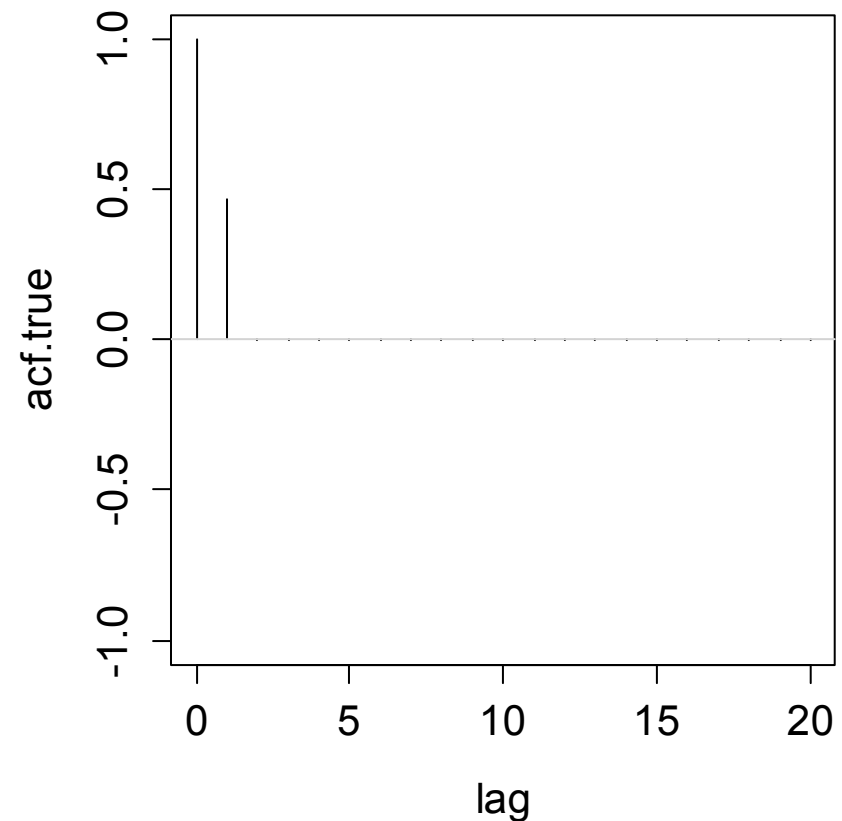
Estimated and True ACF of MA(1)

```
> acf.true <- ARMAacf(ma=0.7, lag.max=20)
```

Estimated ACF



True ACF



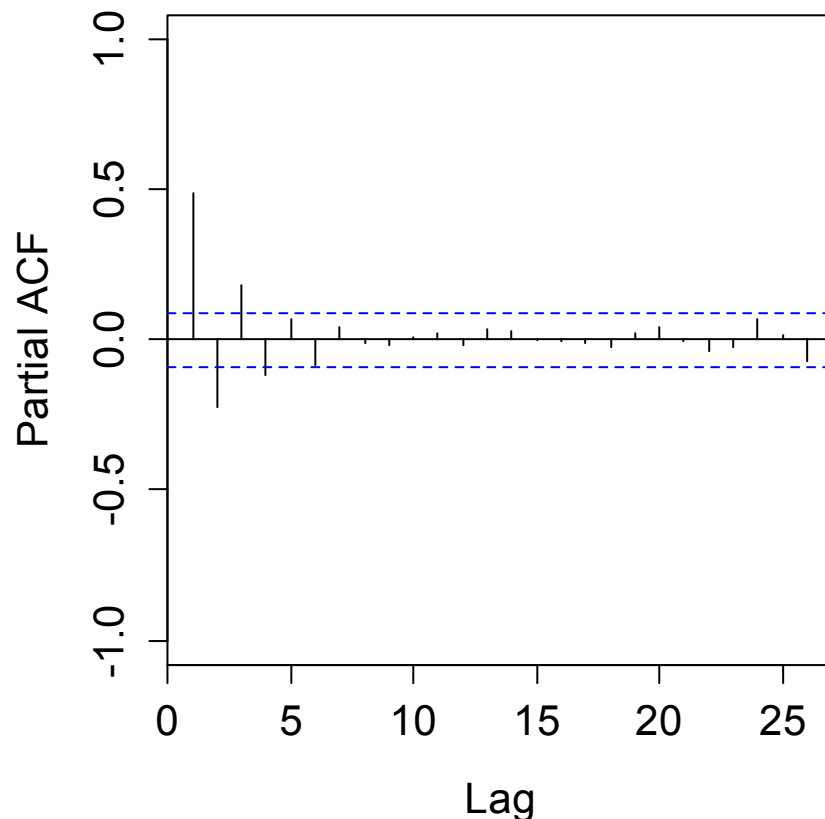
Applied Time Series Analysis

SS 2016 – Moving Average Models

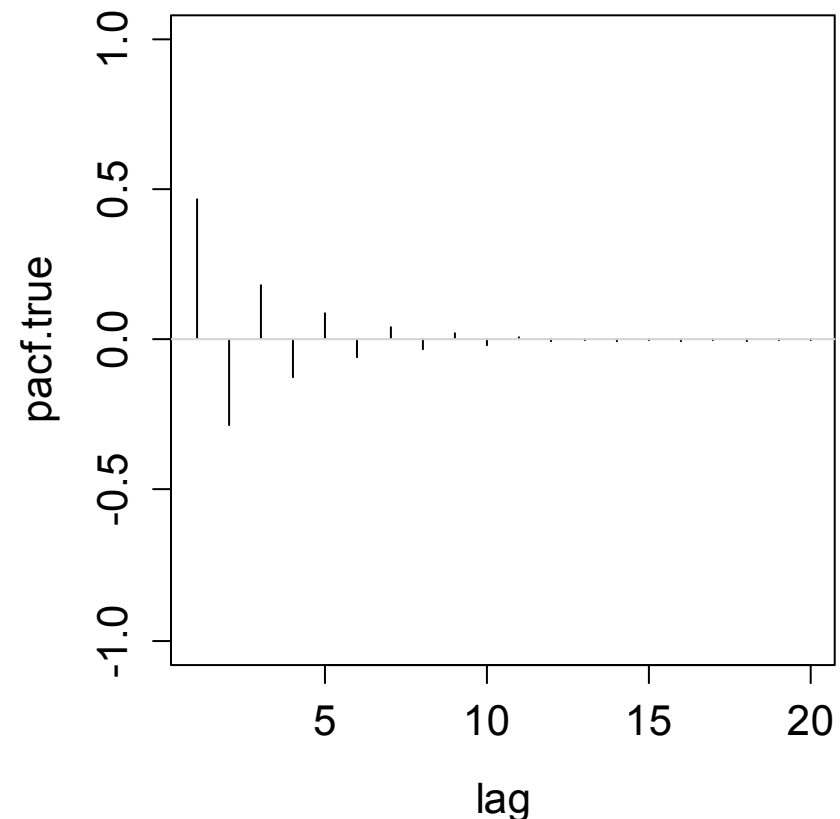
Estimated and True PACF of MA(1)

```
> pacf.true <- ARMAacf(ma=0.7, pacf=T, lag.m=20)
```

Estimated PACF



True PACF



Applied Time Series Analysis

SS 2016 – Moving Average Models

MA(1): Invertibility

Without additional assumptions, the ACF of an MA(1) doesn't allow identification of the generating model.

In particular, the two processes

$$X_t = E_t + 0.5 \cdot E_{t-1}$$

$$U_t = E_t + 2 \cdot E_{t-1}$$

have identical ACF:

$$\rho(1) = \frac{\beta_1}{1 + \beta_1^2} = \frac{1 / \beta_1}{1 + (1 / \beta_1^2)}$$

Applied Time Series Analysis

SS 2016 – Moving Average Models

MA(1): Invertibility

- An MA(1)-, or in general an MA(q)-process is said to be invertible if the roots of the characteristic polynomial $\Theta(B)$ exceed one in absolute value.
- Under this condition, there exists only one MA(q)-process for any given ACF. But please note that any MA(q) is stationary, no matter if it is invertible or not.
- The condition on the characteristic polynomial translates to restrictions on the coefficients. For any MA(1)-model, $|\beta_1| < 1$ is required.
- R function `polyroot()` can be used for finding the roots.

Applied Time Series Analysis

SS 2016 – Moving Average Models

Practical Importance of Invertibility

The condition of invertibility is not only a technical issue, but has important practical meaning. All invertible MA(q) processes can be expressed in terms of an AR(∞), e.g. for an MA(1):

$$\begin{aligned} X_t &= E_t + \beta_1 E_{t-1} \\ &= E_t + \beta_1 (X_{t-1} - \beta_1 E_{t-2}) \\ &= \dots \\ &= E_t + \beta_1 X_{t-1} - \beta_1^2 X_{t-2} + \beta_1^3 X_{t-3} + \dots \\ &= E_t + \sum_{i=1}^{\infty} \psi_i X_{t-i} \end{aligned}$$

Invertibility is practically relevant for model fitting!

Applied Time Series Analysis

SS 2016 – Moving Average Models

Fitting MA(q) Models to Data

As with AR(p) models, there are three main steps:

1) Model Identification

Is the series stationary?

Do the properties of ACF/PACF match?

Derive order q from the cut-off in the ACF

2) Parameter Estimation

How to determine estimates for $m, \beta_1, \dots, \beta_q, \sigma_E^2$?

Conditional Sum of Squares or *MLE*

3) Model Diagnostics

With the same tools/techniques as for AR(p) models

Applied Time Series Analysis

SS 2016 – Moving Average Models

Parameter Estimation for MA(q)

The simplest idea is to exploit the relation between model parameters and autocorrelation coefficients (“Yule-Walker”) after the global mean m has been estimated and subtracted:

$$\rho(k) = \begin{cases} \sum_{j=0}^{q-k} \beta_j \beta_{j+k} / \sum_{j=0}^q \beta_j^2 & \text{for } k = 1, \dots, q \\ 0 & \text{for } k > q \end{cases}$$

In contrast to the Yule-Walker method for AR(p) models, this yields an inefficient estimator that generally generates poor results and hence should not be used in practice.

→ Use CSS or MLE instead!

Applied Time Series Analysis

SS 2016 – Moving Average Models

Conditional Sum of Squares

This is based on the fundamental idea of expressing $\sum E_t^2$ in terms of X_1, \dots, X_n and β_1, \dots, β_q , as the innovations themselves are unobservable.

This is possible for any invertible MA(q), e.g. the MA(1):

$$E_t = X_t - \beta_1 X_{t-1} + \beta_1^2 X_{t-2} + \dots + (-\beta_1)^{t-1} X_1 + \beta_1^t E_0$$

Conditional on the assumption of $E_0 = 0$, it is possible to rewrite $\sum E_t^2$ for any MA(1) using X_1, \dots, X_n and β_1 .

Numerical optimization is required for finding the optimal parameter β_1 , but is available in R function `arima()` with:

```
> arima(..., order=c(...), method="CSS")
```

Applied Time Series Analysis

SS 2016 – Moving Average Models

Maximum-Likelihood Estimation

```
> arima(..., order=c(...), method="CSS-ML")
```

This is the default methods in R, which is based on finding starting values for MLE using the CSS approach. If assuming Gaussian innovations, then:

$$X_t = E_t + \beta_1 E_{t-1} + \dots + \beta_q E_{t-q}$$

will follow a Gaussian distribution as well, and we have:

$$X = (X_1, \dots, X_n) \sim N(0, V) \text{ resp. } Y = (Y_1, \dots, Y_n) \sim N(m \cdot \underline{1}, V)$$

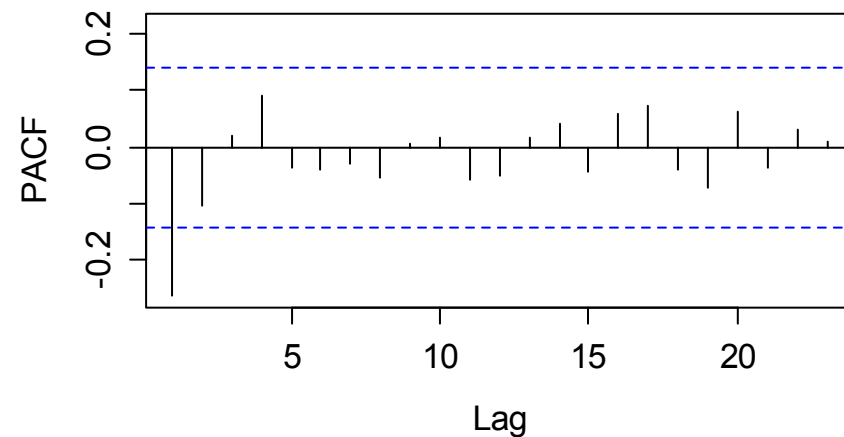
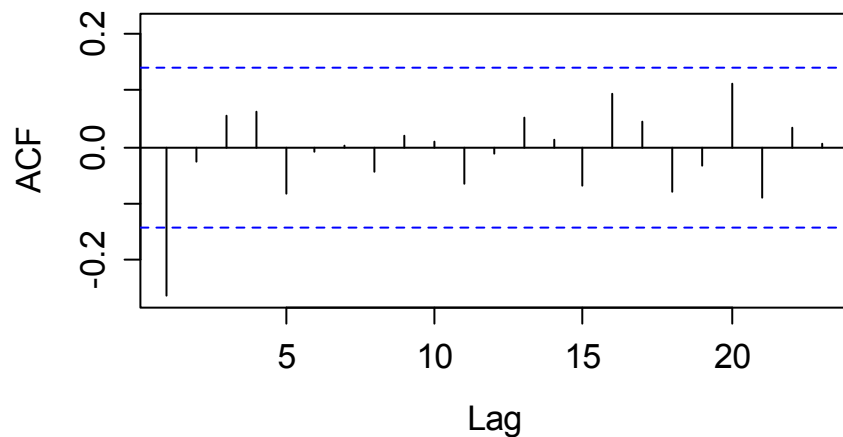
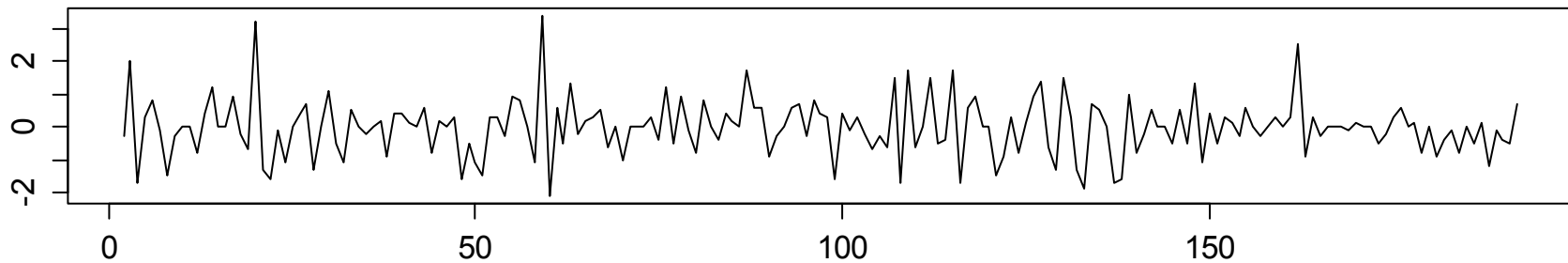
Hence it is possible to derive the likelihood function and simultaneously estimate the parameters $m; \beta_1, \dots, \beta_q; \sigma_E^2$.

Applied Time Series Analysis

SS 2016 – Moving Average Models

Example: Return of an AT&T Bond

Daily Changes in the Return of an AT&T Bond



Applied Time Series Analysis

SS 2016 – Moving Average Models

Example: Return of an AT&T Bond

Fitting an MA(1) model seems a reasonable choice...

```
> arima(diff(attbond), order=c(0,0,1))
```

Call:

```
arima(x = diff(attbond), order = c(0, 0, 1))
```

Coefficients:

	ma1	intercept
	-0.2865	-0.0247
s.e.	0.0671	0.0426

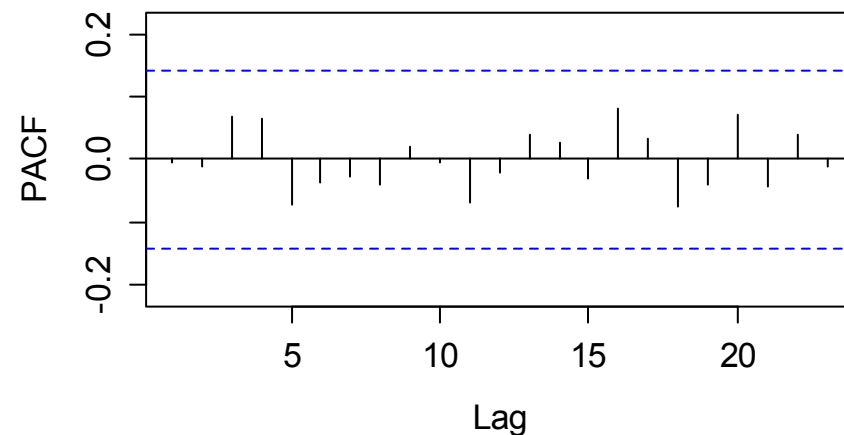
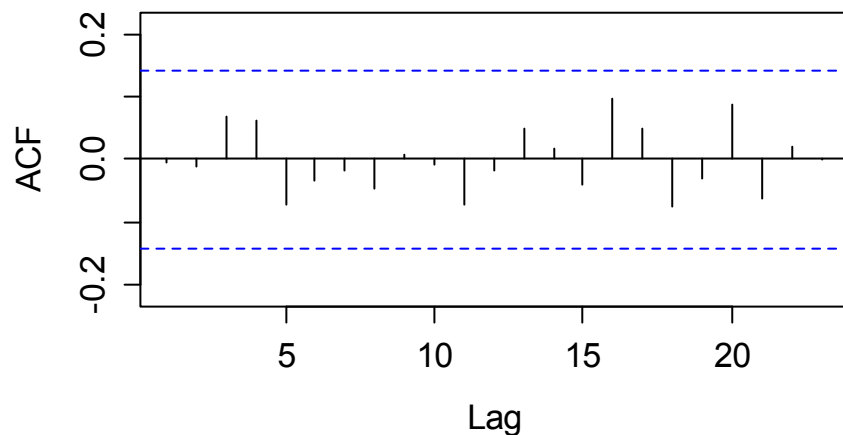
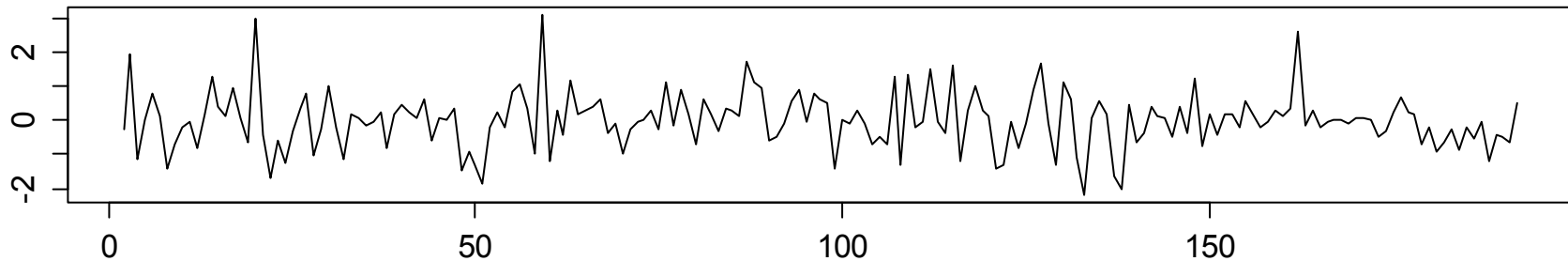
```
sigma^2=0.680, loglikelihood=-234.16, aic=474.3
```

Applied Time Series Analysis

SS 2016 – Moving Average Models

Example: Residual Analysis

Residuals of MA(1)

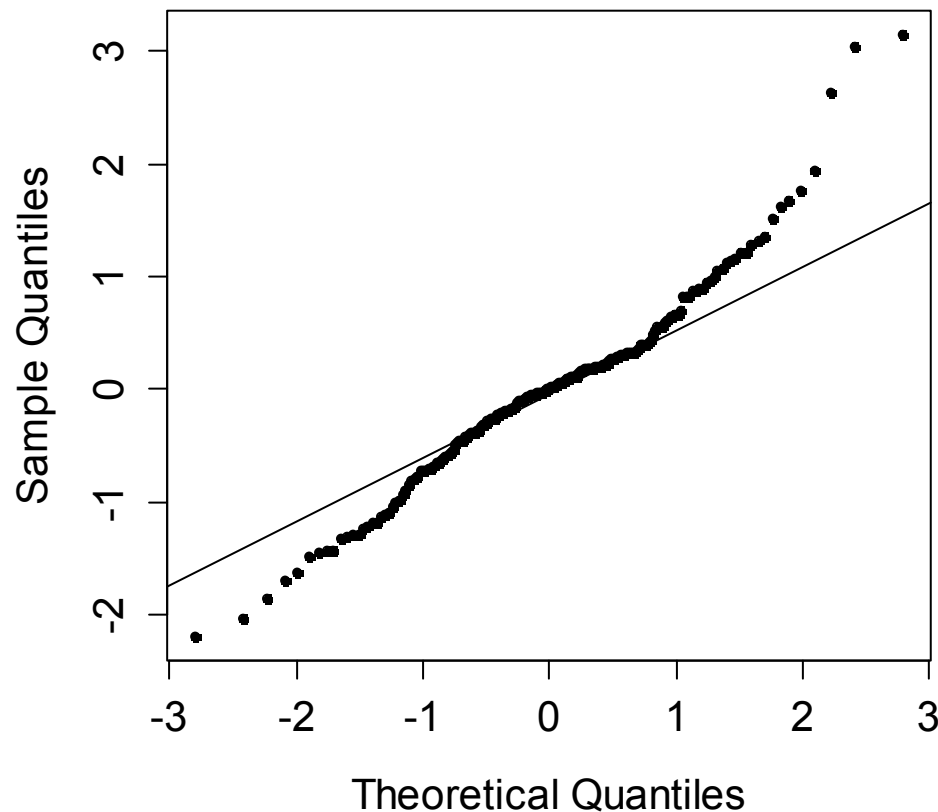


Applied Time Series Analysis

SS 2016 – Moving Average Models

Example: Residual Analysis

Normal QQ-Plot



The residuals are distinctly long tailed. This provides some loss of efficiency in the MLE estimator.

It might as well give a hint that the *iid* property of the residuals is violated, i.e. that there is volatility.

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

ARMA(p,q)-Models

An ARMA(p,q)-model combines AR(p) and MA(q):

$$X_t = \alpha_1 X_{t-1} + \dots + \alpha_p X_{t-p} + E_t + \beta_1 E_{t-1} + \dots + \beta_q E_{t-q}$$

where E_t are i.i.d. innovations (=a white noise process).

It's easier to write ARMA(p,q)'s with the characteristic polynomials:

$$\Phi(B)X_t = \Theta(B)E_t, \text{ where}$$

$$\Phi(z) = 1 - \alpha_1 z - \dots - \alpha_p z^p \quad \text{is the cP of the AR-part, and}$$

$$\Theta(z) = 1 + \beta_1 z + \dots + \beta_q z^q \quad \text{is the cP of the MA-part}$$

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Properties of ARMA(p,q)-Models

The *stationarity* is determined by the AR(p)-part of the model:

If the roots of the characteristic polynomial $\Phi(B)$ exceed one in absolute value, the process is stationary.

The *invertibility* is determined by the MA(q)-part of the model:

If the roots of the characteristic polynomial $\Theta(B)$ exceed one in absolute value, the process is invertible.

Any stationary and invertible ARMA(p,q) can either be rewritten in the form of a non-parsimonious $AR(\infty)$ or an $MA(\infty)$.

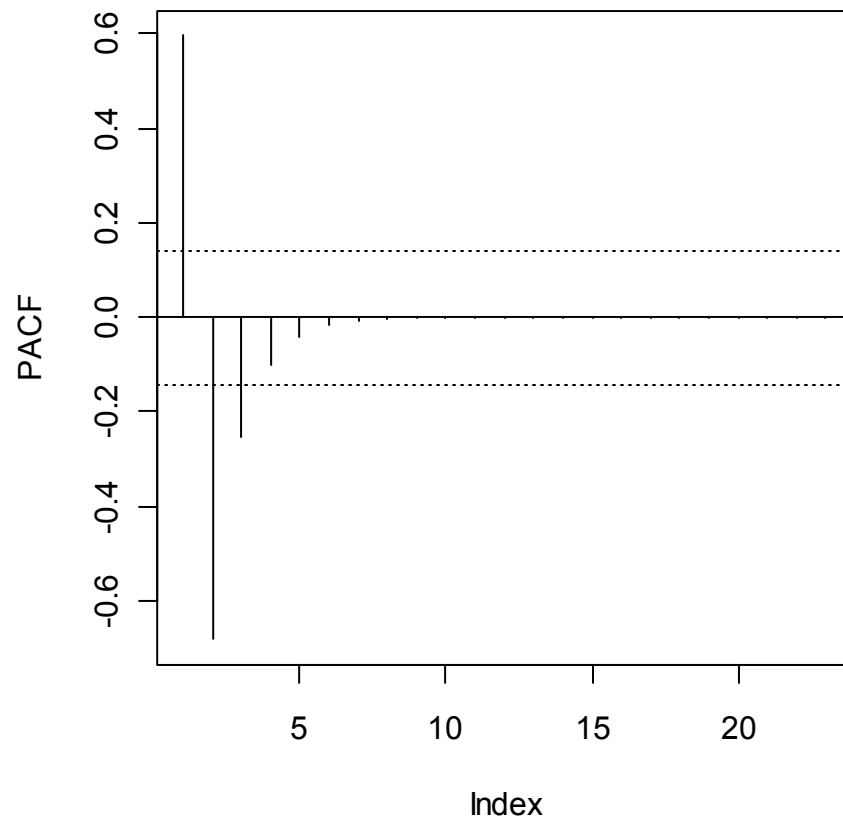
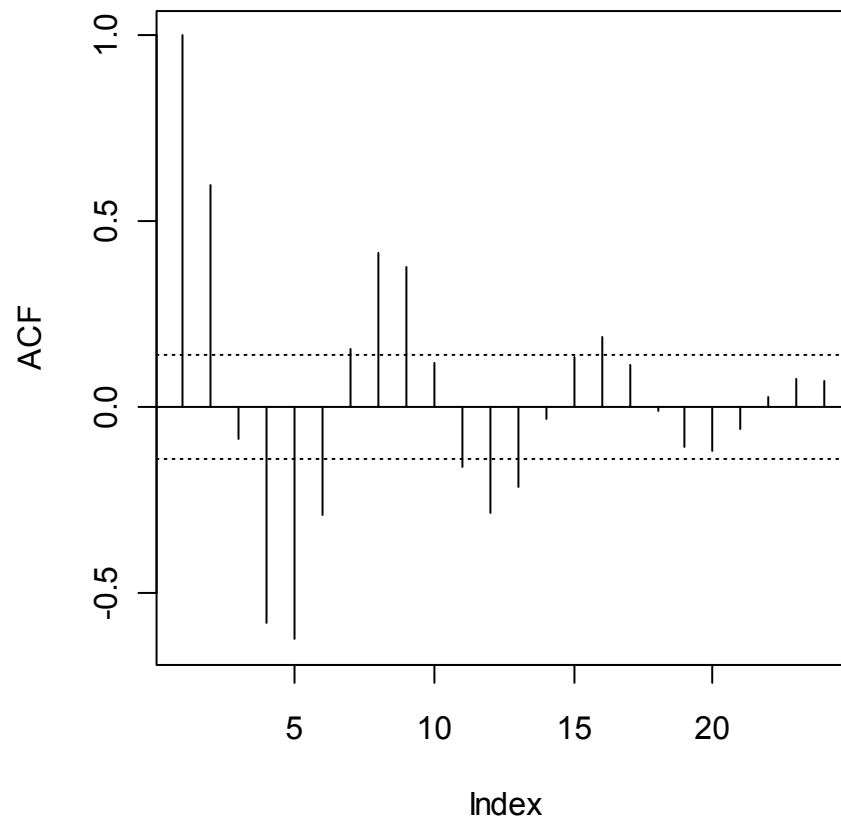
In practice, we mostly consider shifted ARMA(p,q): $Y_t = m + X_t$

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

True ACF/PACF of an ARMA(2,1)

$$X_t = 1.2 \cdot X_{t-1} - 0.8 \cdot X_{t-2} + E_t + 0.4 \cdot E_{t-1}, \quad E_t \sim N(0,1)$$

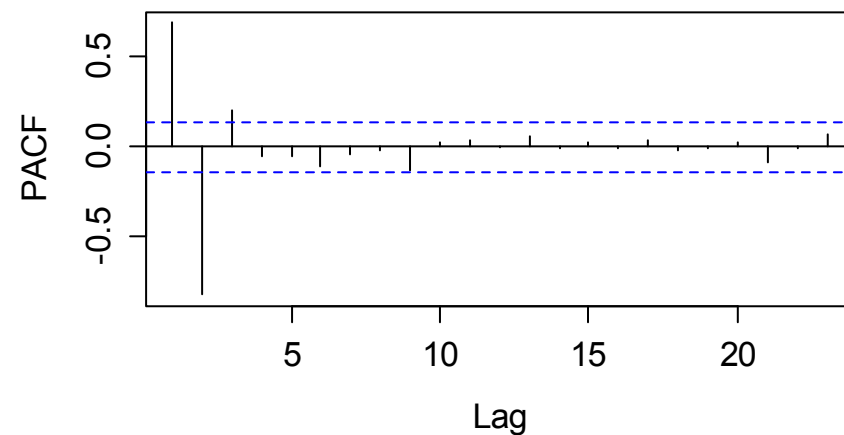
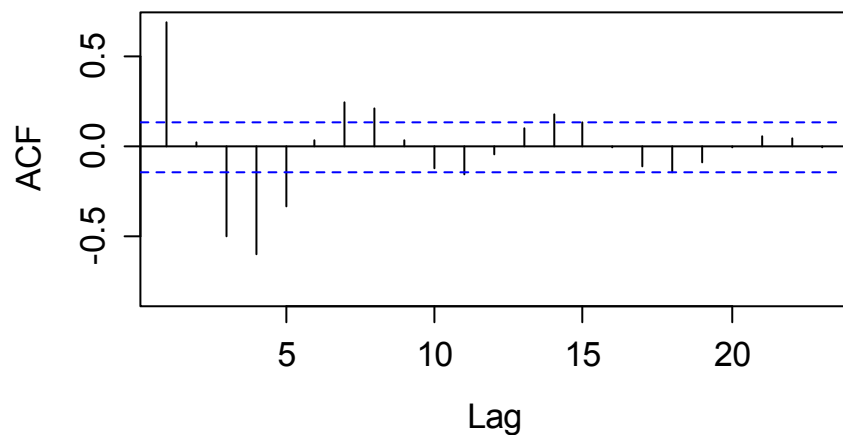
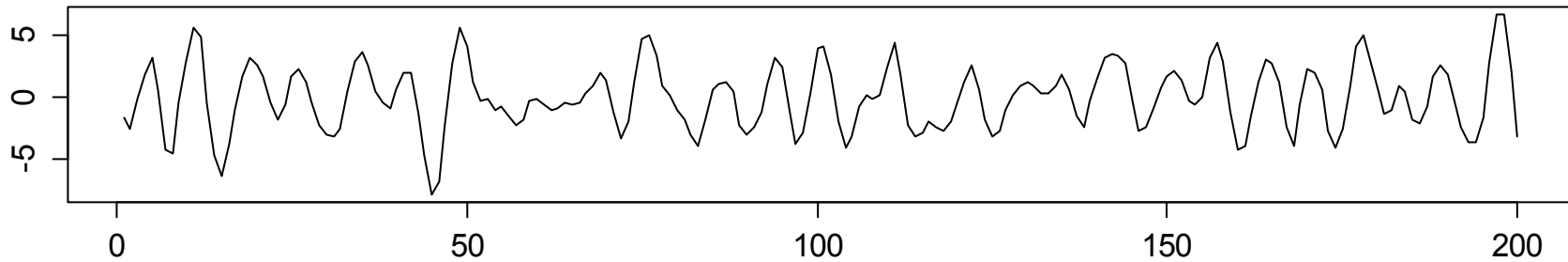


Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Estimated ACF/PACF of an ARMA(2,1)

$$X_t = 1.2 \cdot X_{t-1} - 0.8 \cdot X_{t-2} + E_t + 0.4 \cdot E_{t-1}, \quad E_t \sim N(0,1)$$



Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Properties of ACF/PACF in ARMA(p,q)

	ACF	PACF
AR(p)	exponential decay	cut-off at lag p
MA(q)	cut-off at lag q	exponential decay
ARMA(p,q)	mix decay/cut-off	mix decay/cut-off

- In an ARMA(p,q), depending on the coefficients of the model, either the AR(p) or the MA(q) part can dominate the ACF/PACF characteristics.
- All linear time series processes can be approximated by an ARMA(p,q) with possibly large orders p, q .

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Fitting ARMA(p,q) Models to Data

As with AR(p) or MA(q) models, there are three main steps:

1) Model Identification

Is the series stationary?

Do the properties of ACF/PACF match?

Derive orders p, q from cut-offs in ACF and PACF

2) Parameter Estimation

How to determine estimates for $m, \alpha_1, \dots, \alpha_p, \beta_1, \dots, \beta_q, \sigma_E^2$?

Conditional Sum of Squares or *MLE*

3) Model Diagnostics

With the same tools/techniques as for AR & MA models

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Identification of the Order (p,q)

May be more difficult in reality than in theory:

- We only have one single realization of the time series with finite length. The ACF/PACF plots are not „facts“, but are estimates with uncertainty. The superimposed cut-offs may be difficult to identify from the ACF/PACF plots.
- ARMA(p,q) models are parsimonious, but can usually be replaced by high-order pure AR(p) or MA(q) models. This is not a good idea in practice, however!
- In many cases, an AIC grid search over all ARMA(p,q) with $p + q < 5$ may help to identify promising models.

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Parameter Estimation for ARMA(p,q): CSS

This is based on the fundamental idea of expressing $\sum E_t^2$ in terms of X_1, \dots, X_n and $\alpha_1, \dots, \alpha_p, \beta_1, \dots, \beta_q$ as the innovations themselves are unobservable.

This is possible for any invertible ARMA(p,q). Conditional on the assumption of:

$$E_0 = E_{-1} = E_{-2} = \dots = 0$$

it is possible to rewrite $\sum E_t^2$ for any invertible ARMA(p,q) using X_1, \dots, X_n and $\alpha_1, \dots, \alpha_p, \beta_1, \dots, \beta_q$. Numerical optimization is required for finding the optimal parameters, but is available in R function `arima()` with: `> arima(..., ..., method="CSS")`

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Parameter Estimation for ARMA(p,q): MLE

```
> arima(..., order=c(...), method="CSS-ML")
```

This is the default methods in R, which is based on finding starting values for MLE using the CSS approach. If assuming Gaussian innovations, then:

$$X_t = \alpha_1 X_{t-1} + \dots + \alpha_p X_{t-p} + E_t + \beta_1 E_{t-1} + \dots + \beta_q E_{t-q}$$

will follow a Gaussian distribution as well, and we have:

$$X = (X_1, \dots, X_n) \sim N(0, V) \text{ resp. } Y = (Y_1, \dots, Y_n) \sim N(m \cdot \underline{1}, V)$$

Hence it is possible to derive the likelihood function and simultaneously estimate $m; \alpha_1, \dots, \alpha_p; \beta_1, \dots, \beta_q; \sigma_E^2$.

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

MLE: Remarks

- The MLE approach would work for any distribution. However, for innovation distributions other than the Gaussian, the joint distribution might be „difficult“.
- For „reasonable“ deviations from the normality assumption, MLE mostly still yields „good“ results. Critical are situations with extreme outliers or precarious skewness.
- Besides the parameter estimates, we also obtain their standard errors, assessing the precision of the estimates.
- Other software packages than R often do not rely on MLE, but only use the theoretically less precise CSS approach.

Applied Time Series Analysis

SS 2016 – ARMA(p,q)

AIC-Based Model Choice

In R, finding the AIC-minimizing ARMA(p,q) model is convenient with the use of `auto.arima()` from `library(forecast)`.

→ Handle this function with care! It will always identify a «best fitting» ARMA(p,q), but there is no guarantee that this model provides an adequate fit.

Using `auto.arima()` should always be complemented by visual inspection of the time series for assessing stationarity, verifying the ACF/PACF plots for a second thought on suitable models. Finally, model diagnostics with the usual residual plots will decide whether the model is useful in practice.

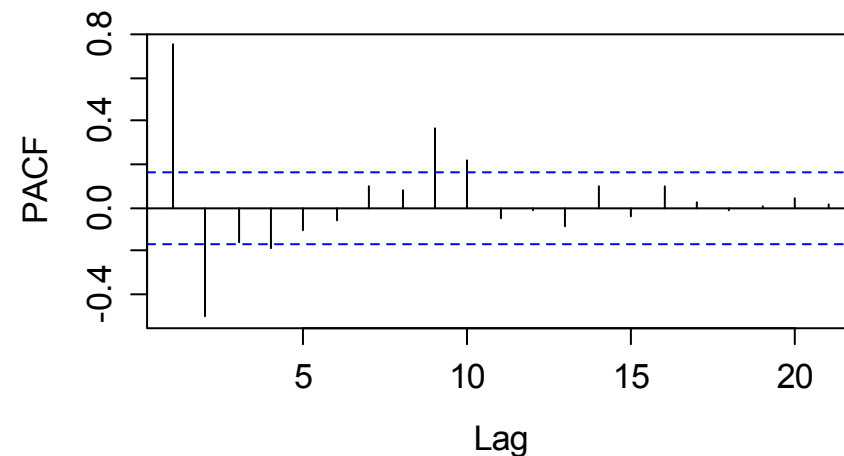
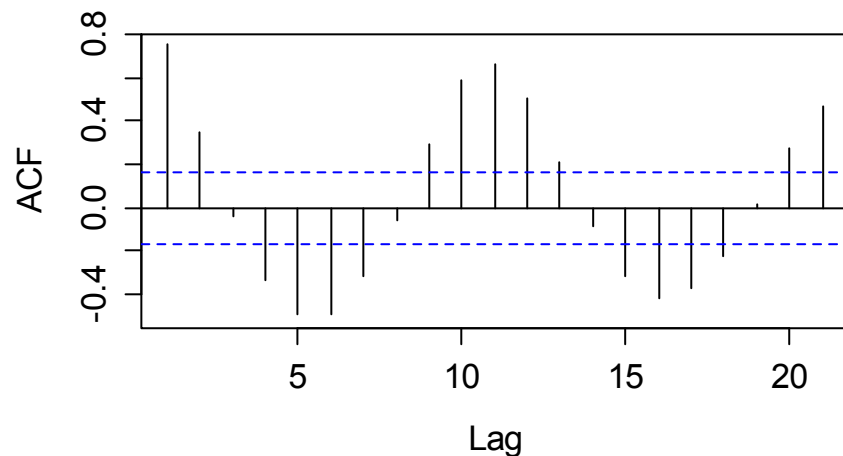
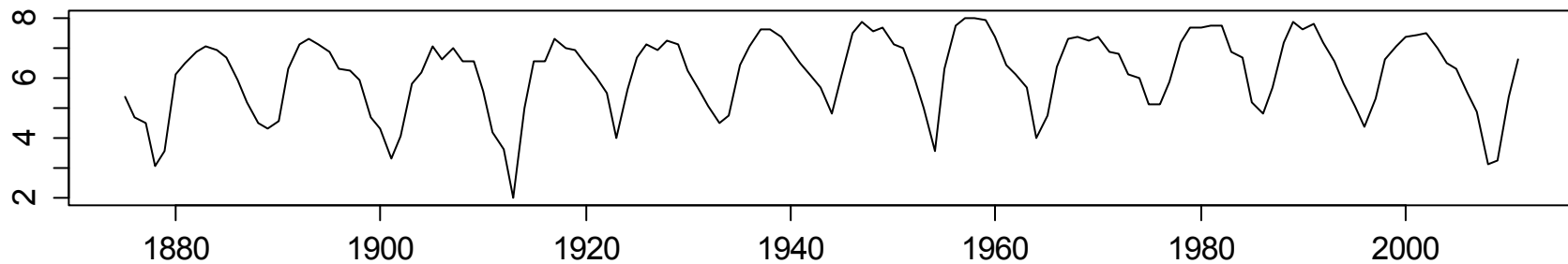
Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Example: Sunspot Areas

```
> tsdisplay(log(sunspotarea), points=FALSE)
```

log(sunspotarea)



Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Example: Sunspot Areas

```
> auto.arima(log(sunspotarea), max.p=10,  
  max.q=10, stationary=TRUE, seasonal=FALSE,  
  allowdrift=FALSE)
```

Series: log(sunspotarea)

ARIMA(2,0,1) with non-zero mean

Coefficients:

	ar1	ar2	ma1	intercept
	1.5001	-0.789	-0.5246	6.2065
s.e.	0.0720	0.061	0.0924	0.0977

$\sigma^2 = 0.4702$, log likelihood = -143.6

AIC=297.2, AICc=297.66, BIC=311.8

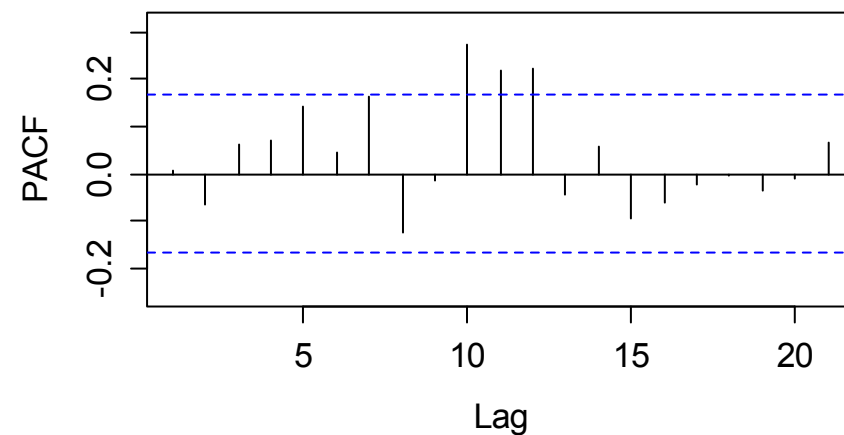
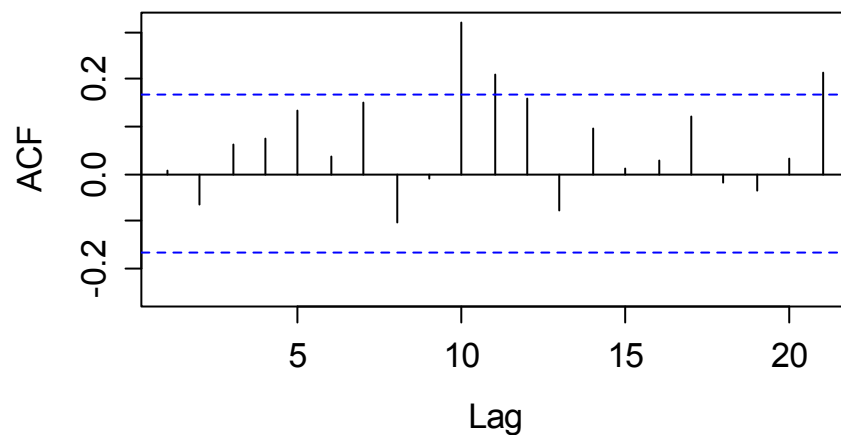
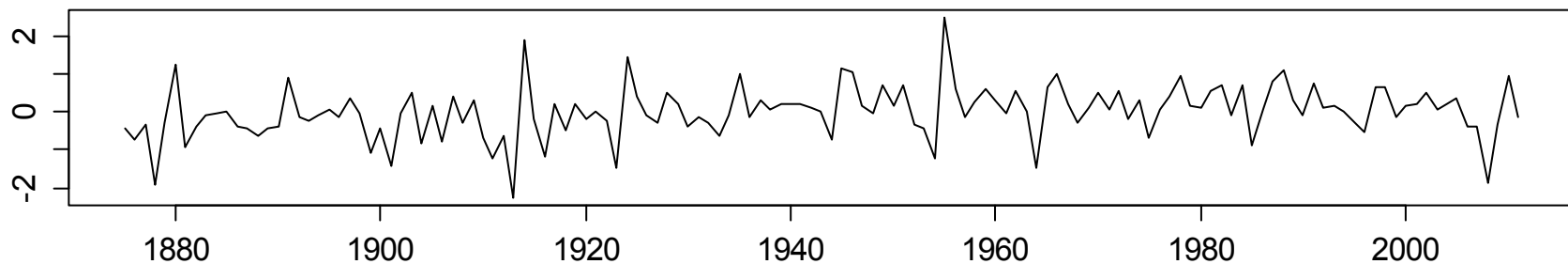
Applied Time Series Analysis

SS 2016 – ARMA(p,q)

Example: Sunspot Areas

```
> tsdisplay(resid(fit), points=FALSE)
```

Residuals from auto.arima() Fit

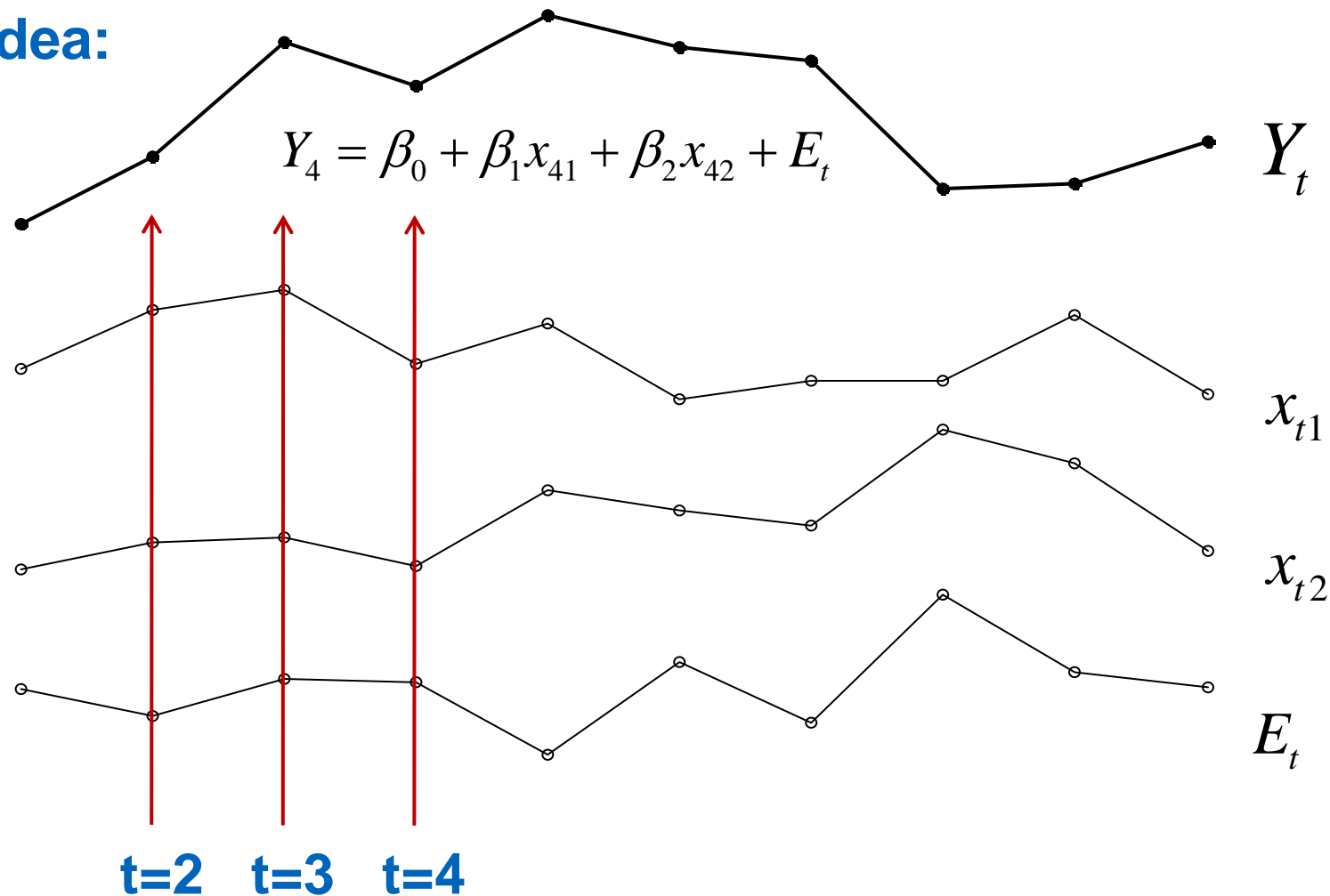


Applied Time Series Analysis

SS 2016 – Time Series Regression

Time Series Regression

Idea:



Applied Time Series Analysis

SS 2016 – Time Series Regression

Time Series Regression

- We speak of time series regression if response and predictors are time series, i.e. if they were observed in a sequence.
- In principle, it is perfectly fine to apply the usual OLS setup

$$Y_t = \beta_0 + \beta_1 x_{t1} + \dots + \beta_q x_{tp} + E_t$$

Be careful: this assumes that the errors E_t are uncorrelated.

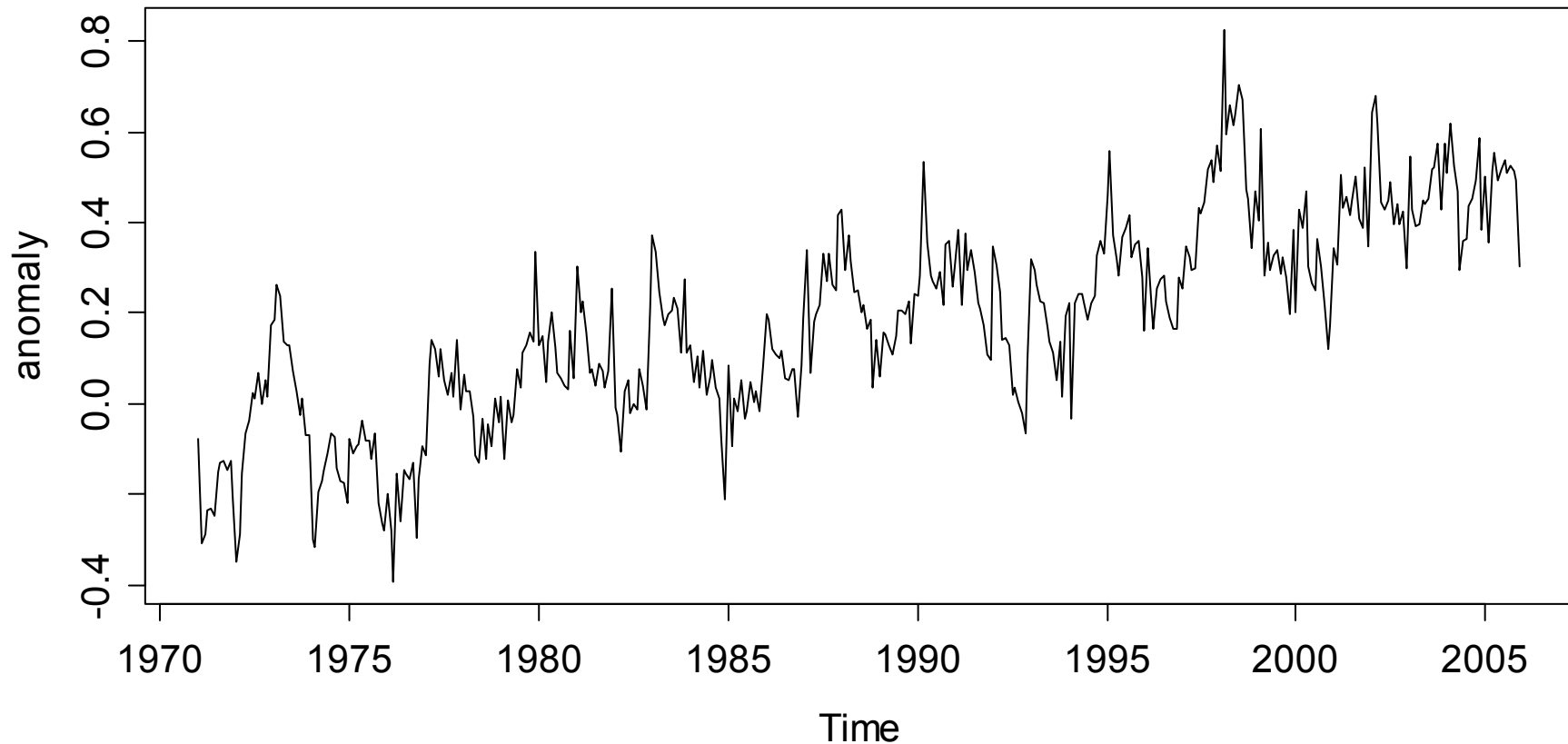
- With correlated errors, the estimates $\hat{\beta}_j$ are still unbiased, but more efficient estimators than OLS exist. The standard errors are wrong, often underestimated, causing spurious significance.
- The **Generalized Least Squares** procedure solves the issue!

Applied Time Series Analysis

SS 2016 – Time Series Regression

Example 1: Global Temperature

Global Temperature Anomalies



Applied Time Series Analysis

SS 2016 – Time Series Regression

Example 1: Global Temperature

Temperature = Linear Trend + Seasonality + Remainder

$$Y_t = \beta_0 + \beta_1 \cdot t + \beta_2 \cdot 1_{[month="Feb"]} + \dots + \beta_{12} \cdot 1_{[month="Dec"]} + E_t,$$

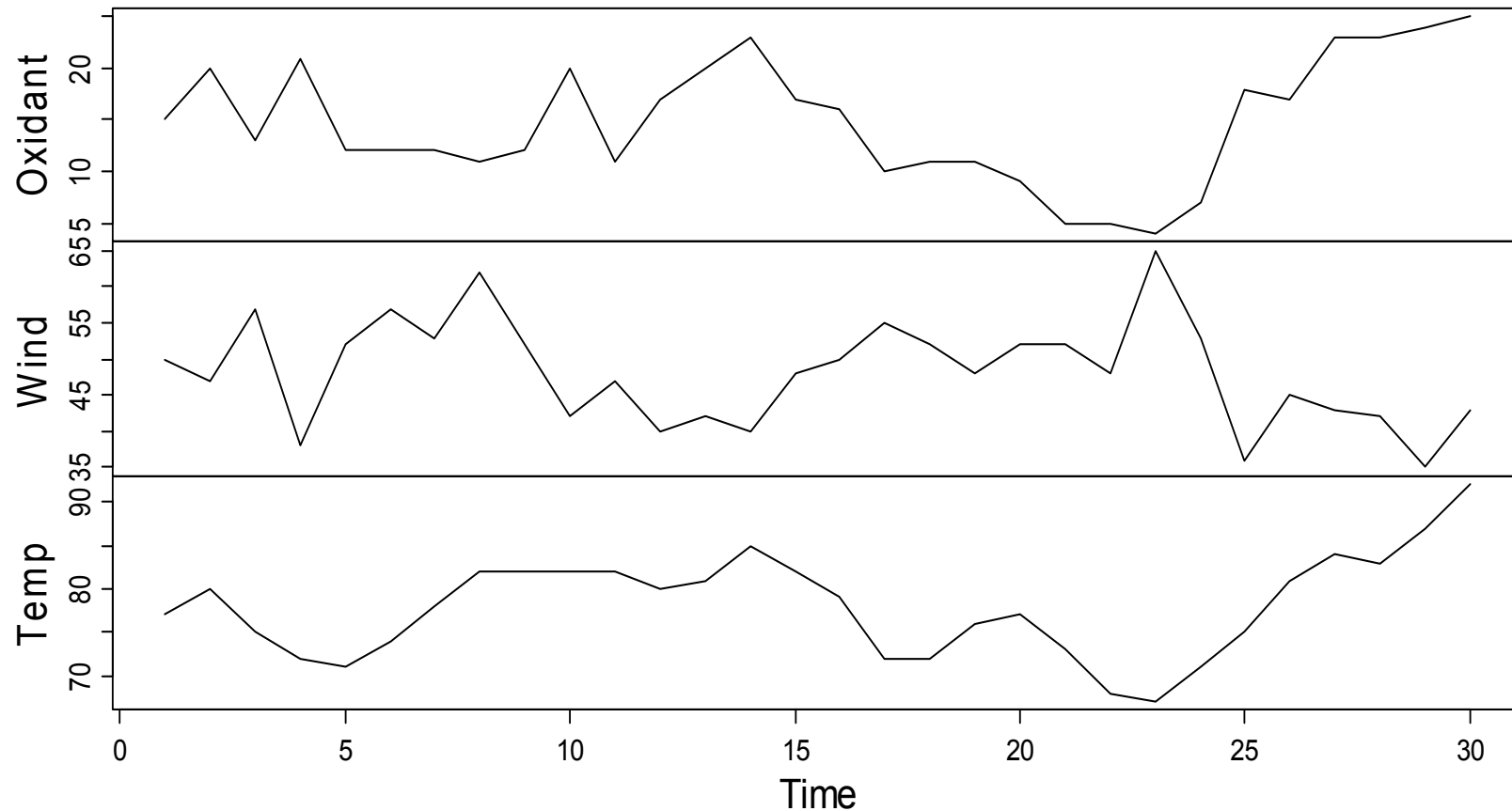
- Recordings from 1971 to 2005, $n = 420$
- The remainder term is usually a stationary time series, thus it would not be surprising if the regression model features correlated errors.
- The applied question which is of importance here is whether there is a significant trend, and a significant seasonal variation

Applied Time Series Analysis

SS 2016 – Time Series Regression

Example 2: Air Pollution

Air Pollution Data



Applied Time Series Analysis

SS 2016 – Time Series Regression

Example 2: Air Pollution

Oxidant = Wind + Temperature + Error

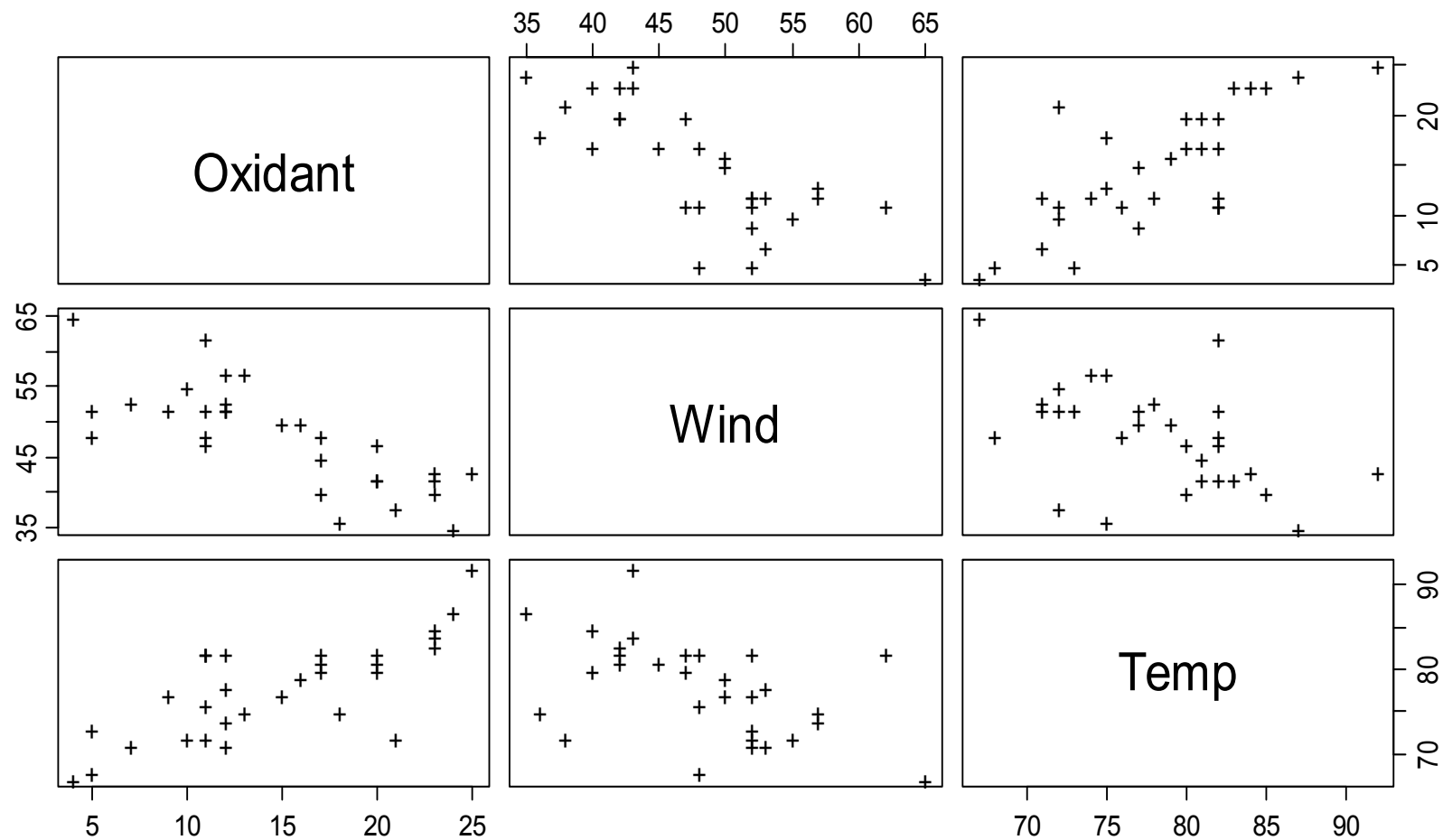
$$Y_t = \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + E_t$$

- Recordings from 30 consecutive days, $n = 30$
- The data are from the Los Angeles basin, USA
- The pollutant level is influenced by both wind and temperature, plus some more, unobserved variables.
- It is well conceivable that there is "day-to-day memory" in the pollutant levels, i.e. there are correlated errors.

Applied Time Series Analysis

SS 2016 – Time Series Regression

Example 2: Air Pollution



Applied Time Series Analysis

SS 2016 – Time Series Regression

Time Series Regression Model

The two examples show that time series regression can appear when decomposing series parametrically, or when working with response/predictors that were recorded sequentially.

$$Y_t = \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + \dots + \beta_p x_{tp} + E_t$$

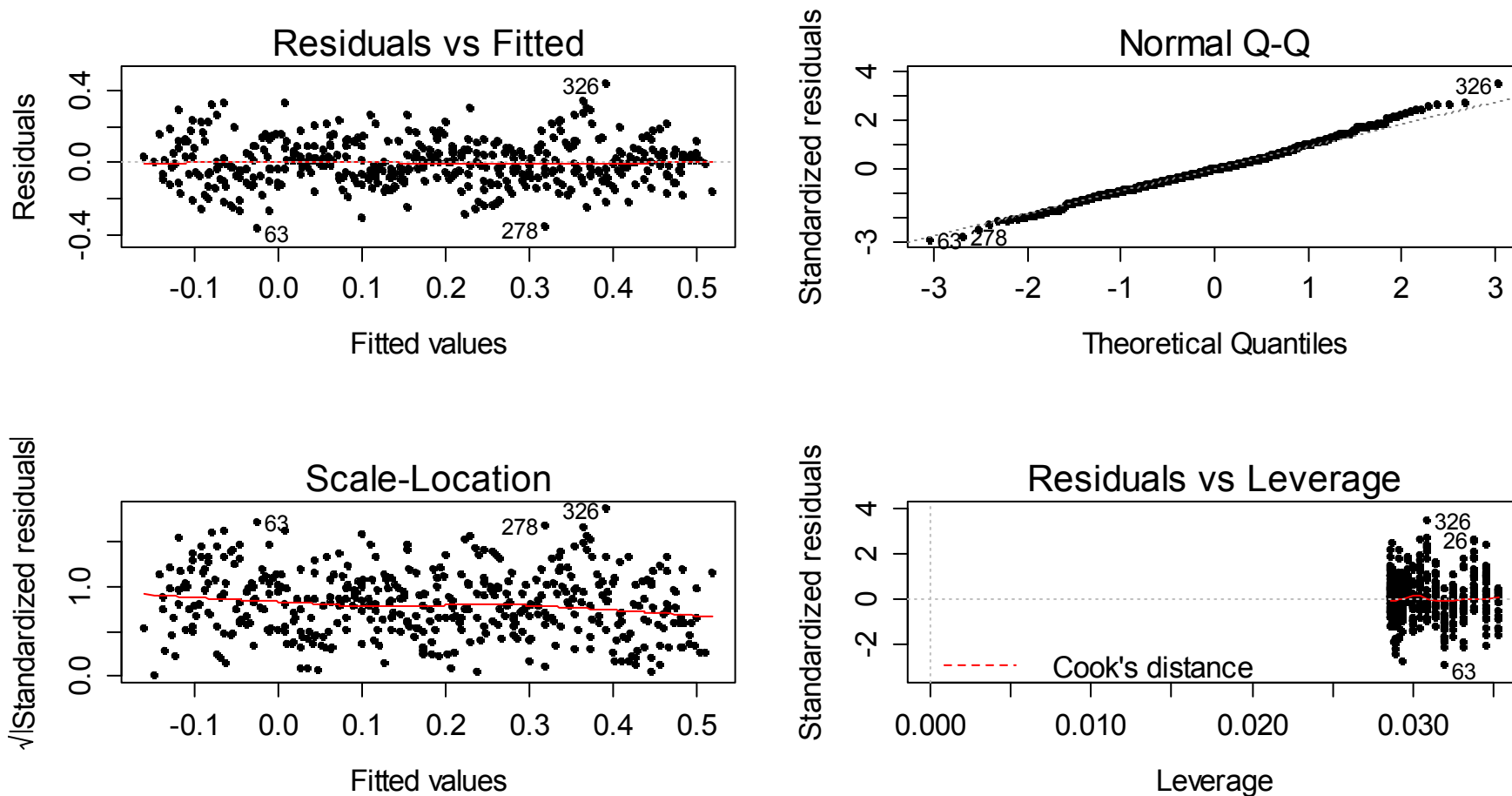
- The series $Y_t, x_{t1}, \dots, x_{tp}$ can be stationary or non-stationary.
- It is crucial that there is no feedback from the response Y_t to the predictor variables x_{t1}, \dots, x_{tp} , i.e. we require an input/output system.
- E_t must be stationary and independent of x_{t1}, \dots, x_{tp} , but may be Non-White-Noise with some serial correlation.

Applied Time Series Analysis

SS 2016 – Time Series Regression

Finding Correlated Errors

1) Start by fitting an OLS regression and analyze residuals



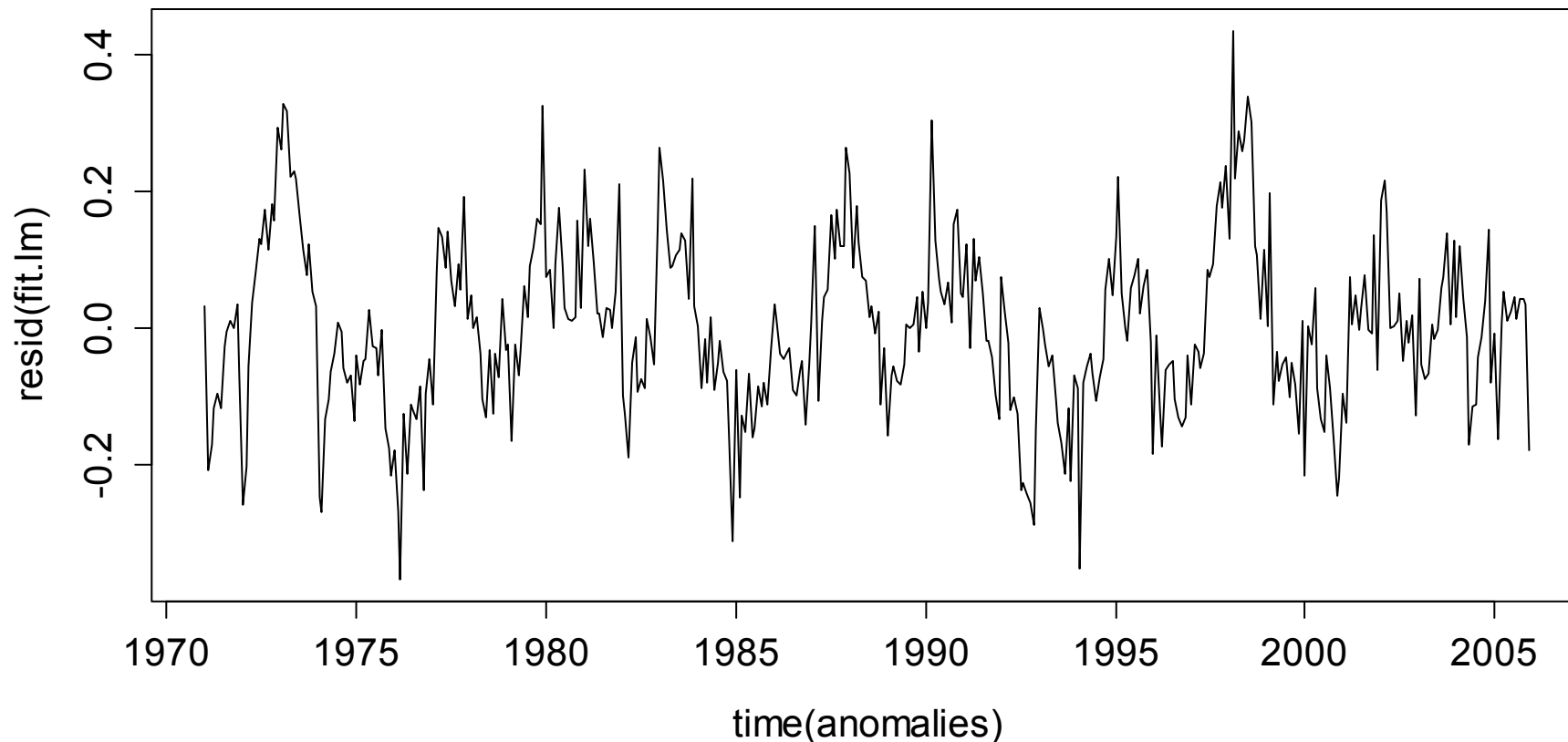
Applied Time Series Analysis

SS 2016 – Time Series Regression

Finding Correlated Errors

2) Continue with a time series plot of OLS residuals

Residuals of the lm() Function



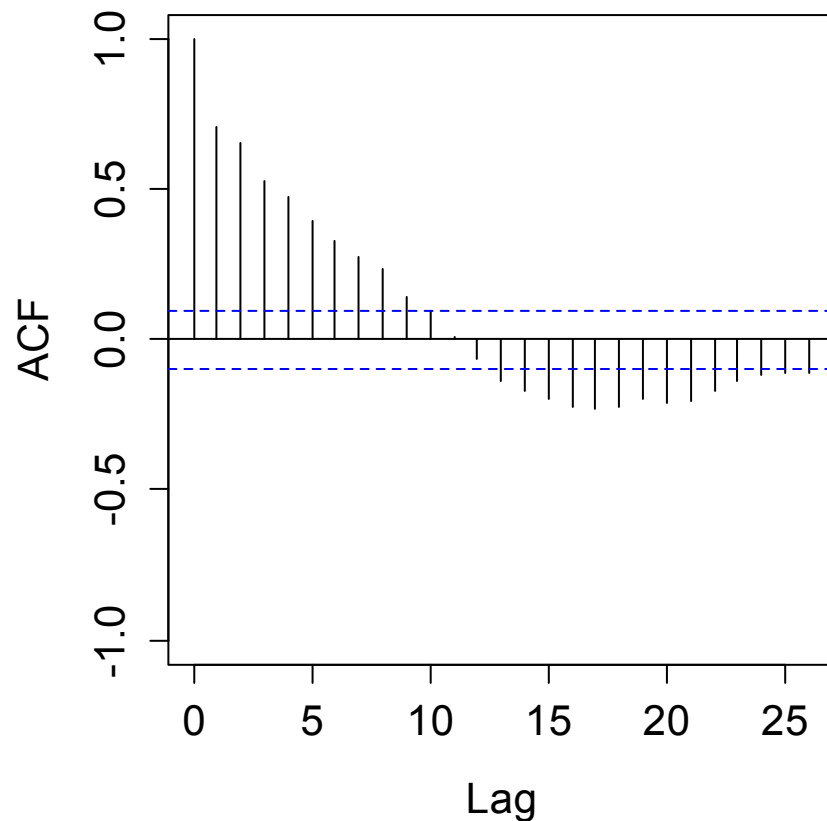
Applied Time Series Analysis

SS 2016 – Time Series Regression

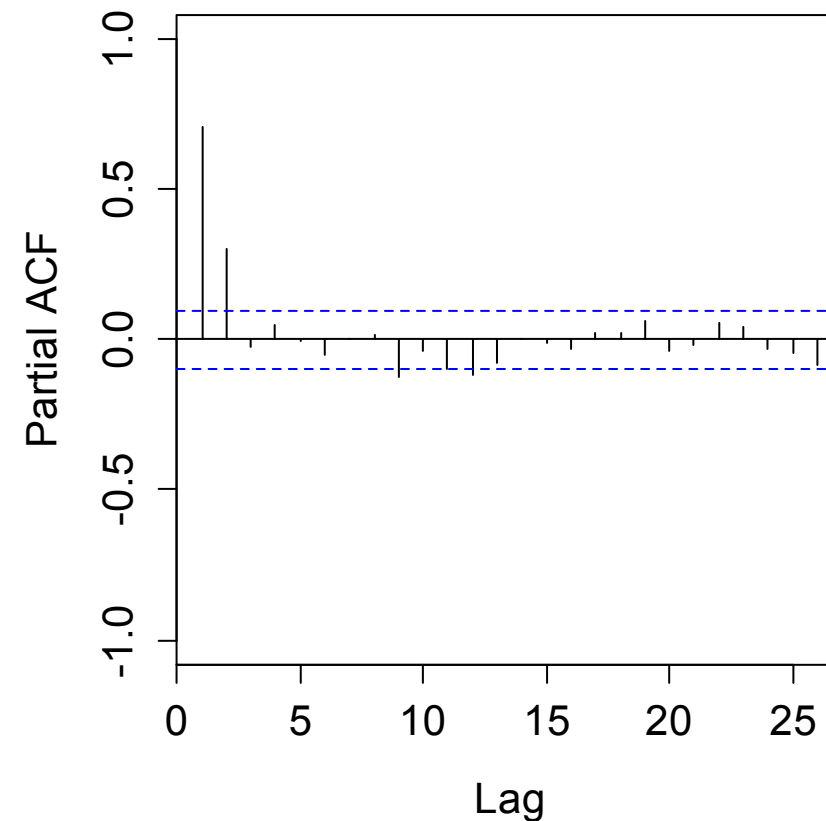
Finding Correlated Errors

3) Also analyze ACF and PACF of OLS residuals

ACF of Residuals



PACF of Residuals



Applied Time Series Analysis

SS 2016 – Time Series Regression

Model for Correlated Errors

→ It seems as if an AR(2) model provides an adequate model for the correlation structure observed in the residuals of the OLS regression model.

```
> fit.ar2 <- ar.burg(resid(fit.lm)); fit.ar2
```

```
Call: ar.burg.default(x = resid(fit.lm))
```

```
Coefficients:
```

```
      1      2  
0.4945  0.3036
```

```
Order selected 2  sigma^2 estimated as  0.00693
```

→ Residuals of this AR(2) model must look like white noise!

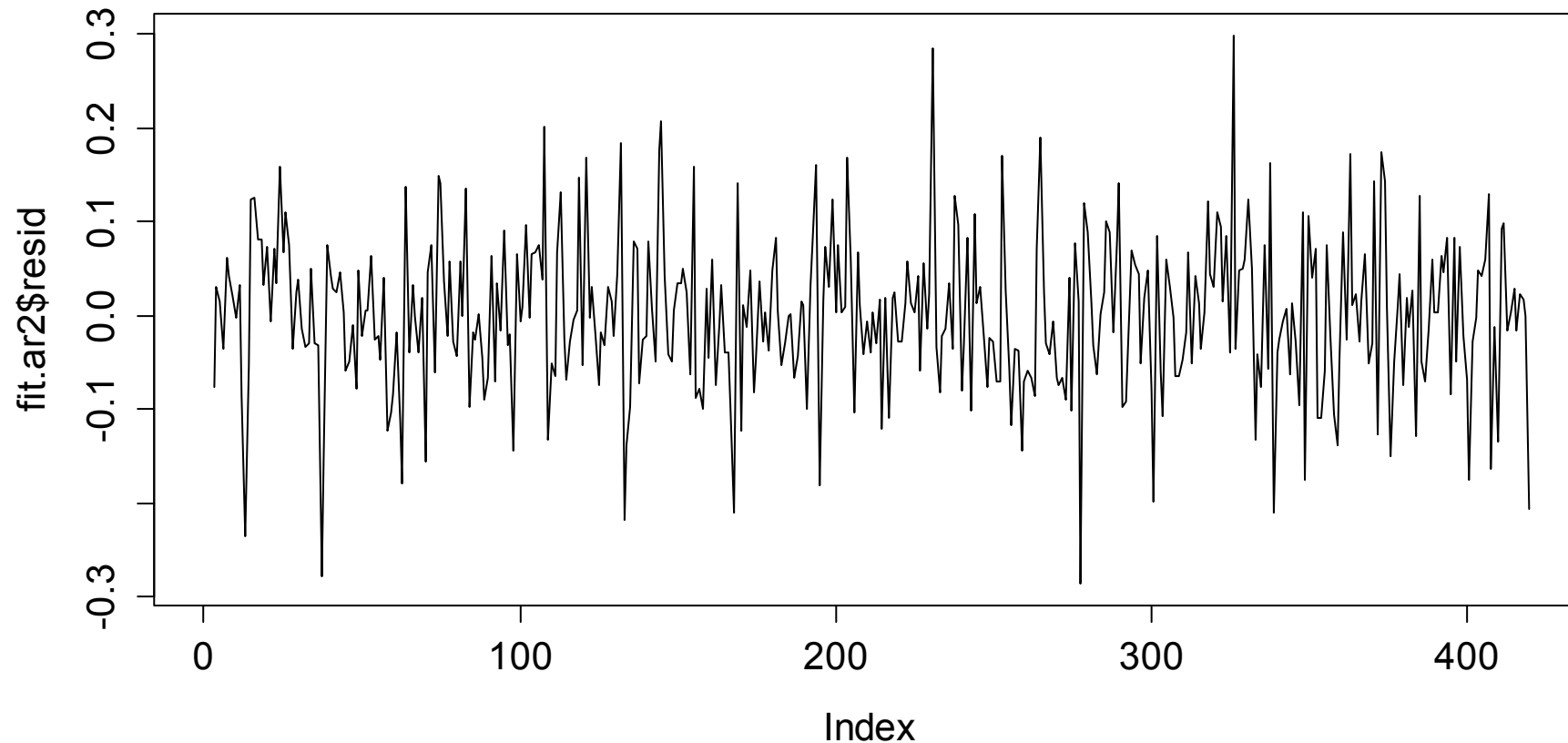
Applied Time Series Analysis

SS 2016 – Time Series Regression

Does the Model Fit?

5) Visualize a time series plot of the AR(2) residuals

Residuals of AR(2)



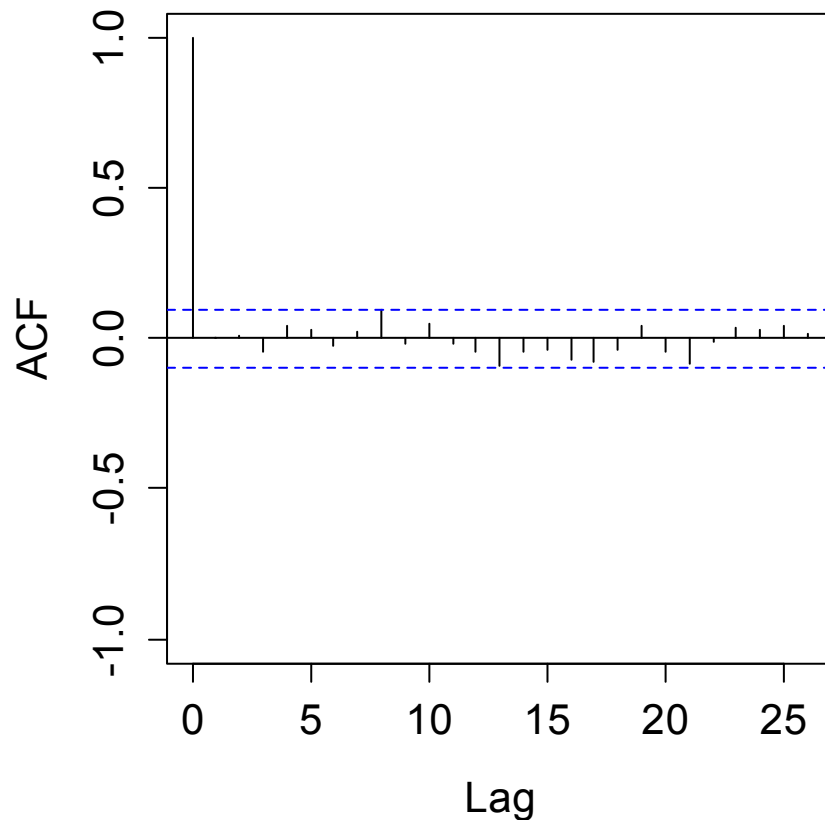
Applied Time Series Analysis

SS 2016 – Time Series Regression

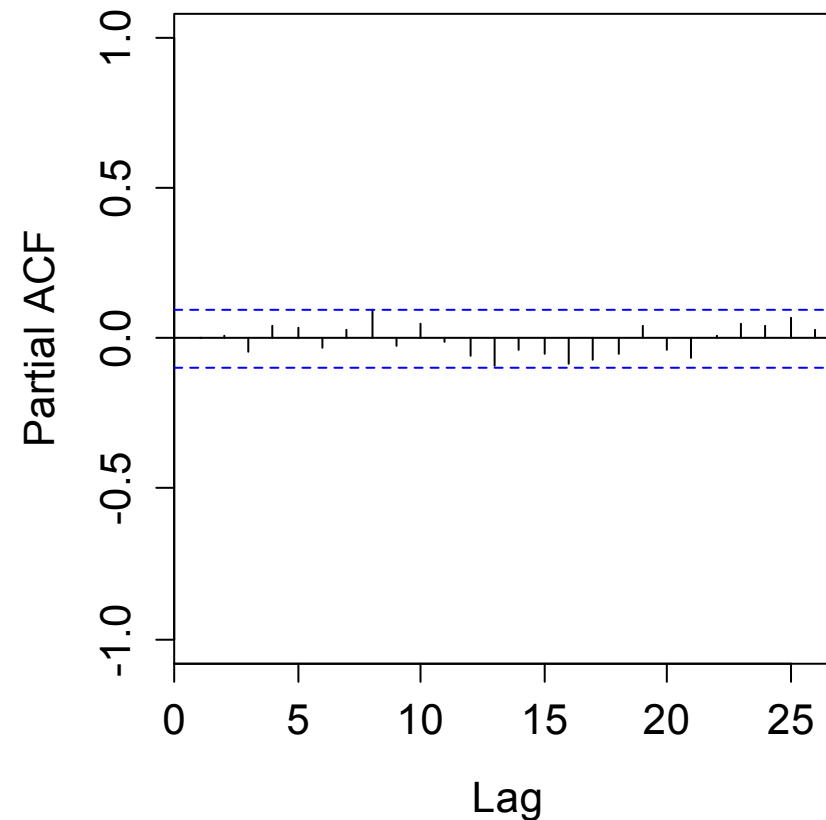
Does the Model Fit?

6) ACF and PACF plots of AR(2) residuals

ACF of AR(2) Residuals



ACF of AR(2) Residuals



Applied Time Series Analysis

SS 2016 – Time Series Regression

Global Temperature: Conclusions

- The residuals from OLS regression are visibly correlated.
- An AR(2) model seems appropriate for this dependency.
- The AR(2) yields a good fit, because its residuals have White Noise properties. We have thus understood the dependency of the regression model errors.

→ We need to account for the correlated errors, else the coefficient estimates will be unbiased but inefficient, and the standard errors are wrong, preventing successful inference for trend and seasonality

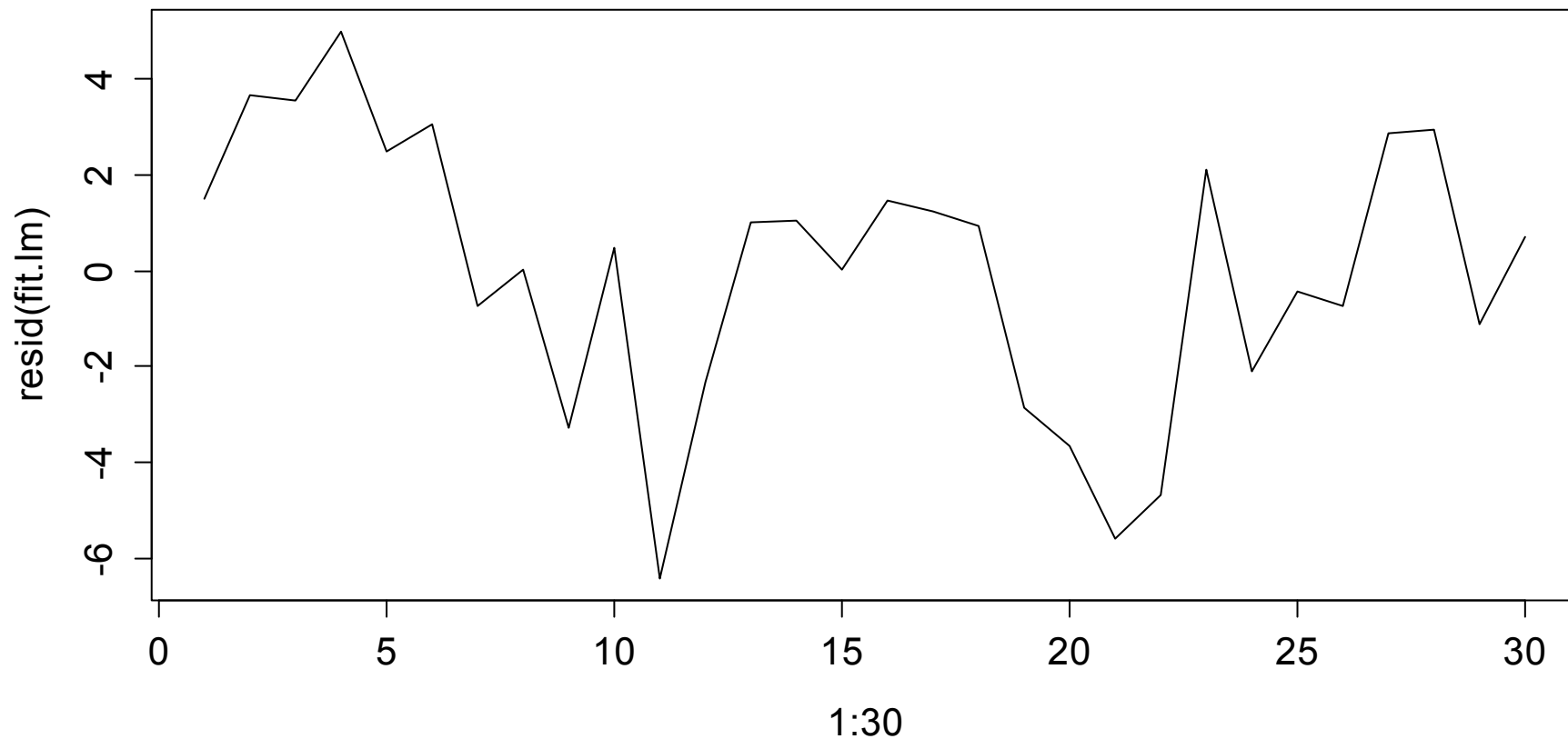
Applied Time Series Analysis

SS 2016 – Time Series Regression

Air Pollution: OLS Residuals

Time series plot: dependence present or not?

Residuals of the lm() Function



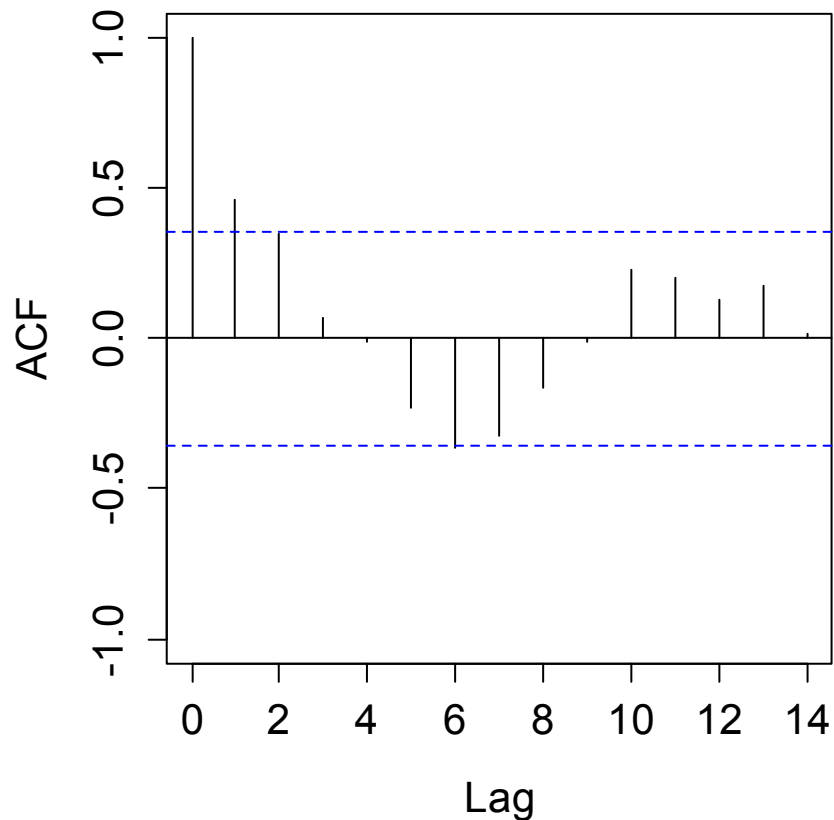
Applied Time Series Analysis

SS 2016 – Time Series Regression

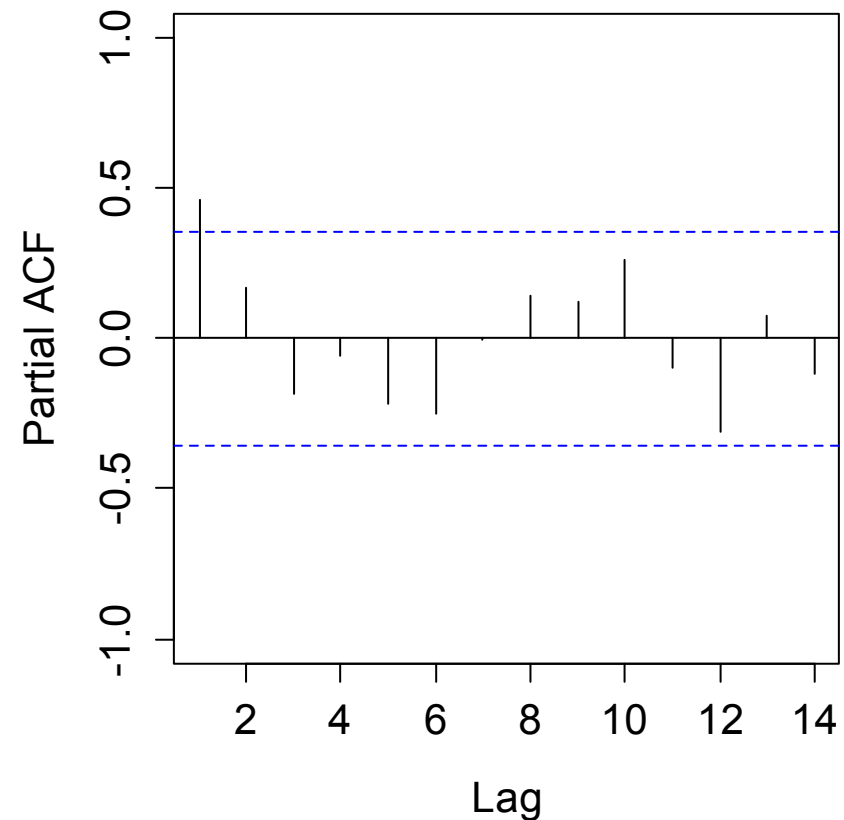
Air Pollution: OLS Residuals

ACF and PACF suggest: *there is AR(1) dependence*

ACF of Residuals



PACF of Residuals



Applied Time Series Analysis

SS 2016 – Time Series Regression

Air Pollution Example

```
> summary(fit.lm)
```

```
Call: lm(formula = Oxidant ~ Wind + Temp, data = dat)
```

```
Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-5.20334	11.11810	-0.468	0.644	
Wind	-0.42706	0.08645	4.940	3.58e-05	***
Temp	0.52035	0.10813	4.812	5.05e-05	***

```
---
```

```
Residual standard error: 2.95 on 27 degrees of freedom
```

```
Multiple R-squared: 0.7773, Adjusted R-squared: 0.7608
```

```
F-statistic: 47.12 on 2 and 27 DF, p-value: 1.563e-09
```

→ Inference results are not valid if residuals are correlated

Applied Time Series Analysis

SS 2016 – Time Series Regression

Durbin-Watson Test

- The Durbin-Watson approach is a test for autocorrelated errors in regression modeling based on the test statistic:

$$D = \frac{\sum_{t=2}^N (r_t - r_{t-1})^2}{\sum_{t=1}^N r_t^2}$$

- This is implemented in R: `dwtest()` in `library(lmtest)`. A p-value for the null of no autocorrelation is computed.
- This test does not detect all autocorrelation structures. If the null is not rejected, the residuals may still be autocorrelated.

→ **Never forget to check ACF/PACF of the residuals!**

Applied Time Series Analysis

SS 2016 – Time Series Regression

Durbin-Watson Test

Example 1: Global Temperature

```
> library(lmtest)
> dwtest(fit.lm)
data:  fit.lm
DW = 0.5785, p-value < 2.2e-16
alt. hypothesis: true autocorrelation is greater than 0
```

Example 2: Air Pollution

```
> dwtest(fit.lm)
data:  fit.lm
DW = 1.0619, p-value = 0.001675
alt. hypothesis: true autocorrelation is greater than 0
```

Applied Time Series Analysis

SS 2016 – Time Series Regression

Cochrane-Orcutt Method

This is a simple, iterative approach for correctly dealing with time series regression. We consider the pollutant example:

$$Y_t = \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + E_t \quad \text{with} \quad E_t = \alpha E_{t-1} + U_t; \quad U_t \sim N(0, \sigma_U^2) \quad iid$$

The fundamental trick is using the transformation:

$$Y'_t = Y_t - \alpha Y_{t-1}$$

This will lead to a regression problem with *iid* errors:

$$Y'_t = \beta'_0 + \beta_1 x'_{t1} + \beta_2 x'_{t2} + U_t$$

See the blackboard for full details. The idea is to run an OLS regression first, determine the transformation from the residuals and finally obtaining corrected estimates.

Applied Time Series Analysis

SS 2016 – Time Series Regression

Generalized Least Squares

→ See the blackboard for full explanation

- OLS regression assumes a diagonal error covariance matrix, but there is a generalization to $Var(E) = \sigma^2 \Sigma$.

- If we find $\Sigma = SS^T$, the regression model can be rewritten as:

$$y = X\beta + E$$

$$S^{-1}y = S^{-1}X\beta + S^{-1}E$$

$$y^* = X^*\beta + E^* \quad \text{with } Var(E^*) = \sigma^2 I$$

- One obtains the generalized least square estimates:

$$\hat{\beta} = (X^T \Sigma^{-1} X)^{-1} X^T \Sigma^{-1} y \quad \text{with } Var(\hat{\beta}) = (X^T \Sigma^{-1} X)^{-1} \sigma^2$$

Applied Time Series Analysis

SS 2016 – Time Series Regression

Generalized Least Squares

For using the GLS approach, i.e. for correcting the dependent errors, we need an estimate of the error covariance matrix Σ .

The two major options for obtaining it are:

- 1) **Cochrane-Orcutt (for AR(p) correlation structure only)**
iterative approach: i) β , ii) α , iii) β
 - 2) **GLS (Generalized Least Squares, for ARMA(p,q))**
simultaneous estimation of β and α
- **Full explanation of the two different approaches was provided on the blackboard!**

Applied Time Series Analysis

SS 2016 – Time Series Regression

GLS: Syntax

Package `nlme` has function `gls()`. It does only work if the correlation structure of the errors is provided. This has to be determined from the residuals of an OLS regression first.

```
> library(nlme)
> corStruct <- corARMA(form=~time, p=2)
> fit.gls <- gls(temp~time+season, data=dat,
                 correlation=corStruct)
```

The output contains the **regression coefficients** and their **standard errors**, as well as the **AR-coefficients** plus some further information about the model (Log-Likelihood, AIC, ...).

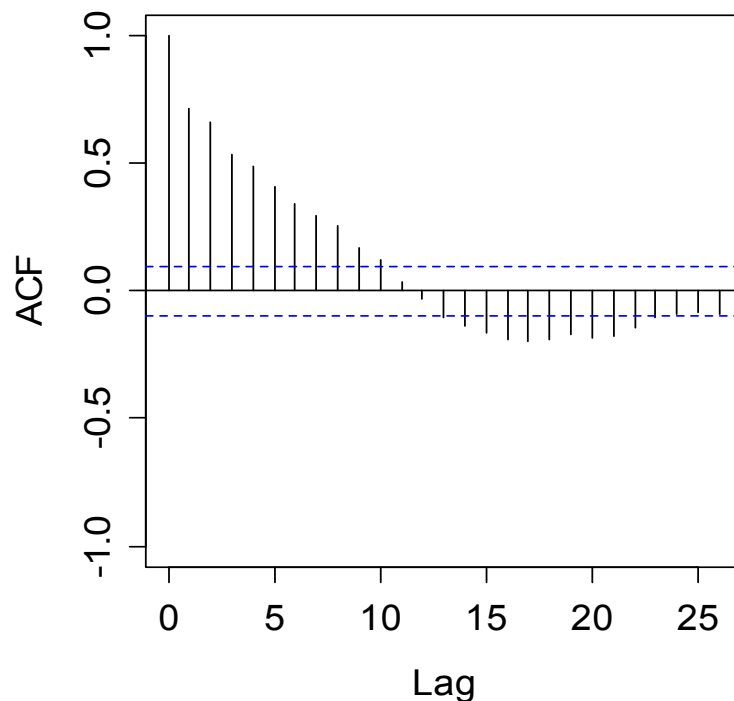
Applied Time Series Analysis

SS 2016 – Time Series Regression

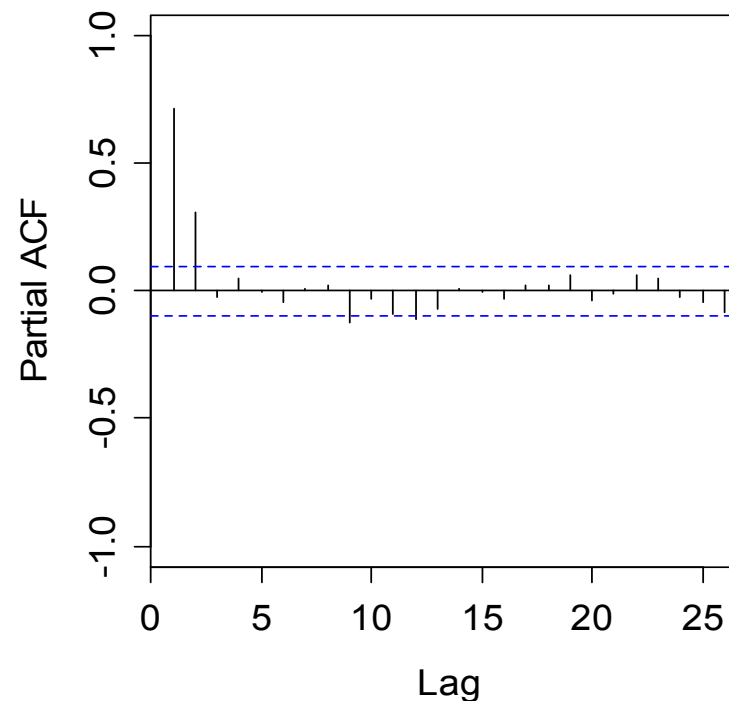
GLS: Residual Analysis

The residuals from a GLS must look like coming from a time series process with the respective structure:

ACF of GLS-Residuals



PACF of GLS-Residuals



Applied Time Series Analysis

SS 2016 – Time Series Regression

GLS/OLS: Comparison of Results

→ The trend in the global temperature is significant!

```
> coef(fit.lm)["time"]  
      time  
0.01822374  
> confint(fit.lm, "time")  
              2.5 %      97.5 %  
time 0.01702668 0.0194208
```

OLS

```
> coef(fit.gls)["time"]  
      time  
0.02017553  
> confint(fit.gls, "time")  
              2.5 %      97.5 %  
time 0.01562994 0.02472112
```

GLS

Applied Time Series Analysis

SS 2016 – Time Series Regression

GLS/OLS: Comparison of Results

→ The seasonal effect is not significant!

```
> drop1(fit.lm, test="F")
```

OLS

```
temp ~ time + season
```

	Df	Sum of Sq	RSS	AIC	F value	Pr(F)	
<none>			6.4654	-1727.0			
time	1	14.2274	20.6928	-1240.4	895.6210	<2e-16	***
season	11	0.1744	6.6398	-1737.8	0.9982	0.4472	

```
> anova(fit.gls)
```

GLS

```
Denom. DF: 407
```

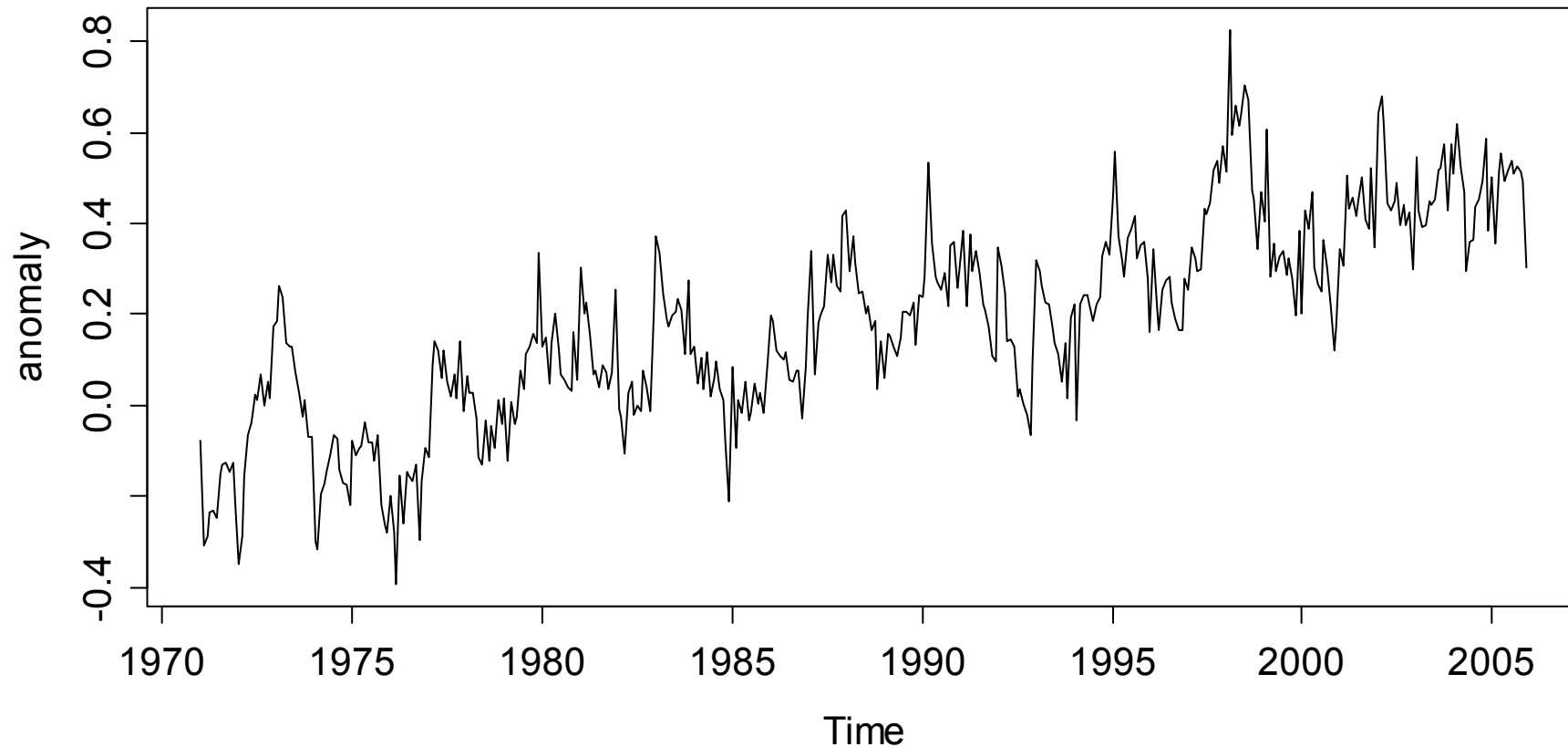
	numDF	F-value	p-value
(Intercept)	1	78.40801	<.0001
time	1	76.48005	<.0001
season	11	0.64371	0.7912

Applied Time Series Analysis

SS 2016 – Time Series Regression

GLS Example 1: Global Temperature

Global Temperature Anomalies



Applied Time Series Analysis

SS 2016 – Time Series Regression

Air Pollution: Results

Both predictors are significant with both approaches...

```
> confint(fit.lm, c("Wind", "Temp"))  
                2.5 %      97.5 %  
Wind -0.6044311 -0.2496841  
Temp  0.2984794  0.7422260
```

OLS

```
> confint(fit.gls, c("Wind", "Temp"))  
                2.5 %      97.5 %  
Wind -0.5447329 -0.2701709  
Temp  0.2420436  0.7382426
```

GLS

→ But still, it is important to use GLS with correlated errors!

Applied Time Series Analysis

SS 2016 – Time Series Regression

Simulation Study: Model

We want to study the effect of correlated errors on the quality of estimates when using the least squares approach:

$$x_t = t / 50$$

$$y_t = x_t + 2x_t^2 + E_t$$

where E_t is from an AR(1)-process with $\alpha = -0.65$ and $\sigma = 0.1$.

We generate 100 realizations from this model and estimate the regression coefficient and its standard error by:

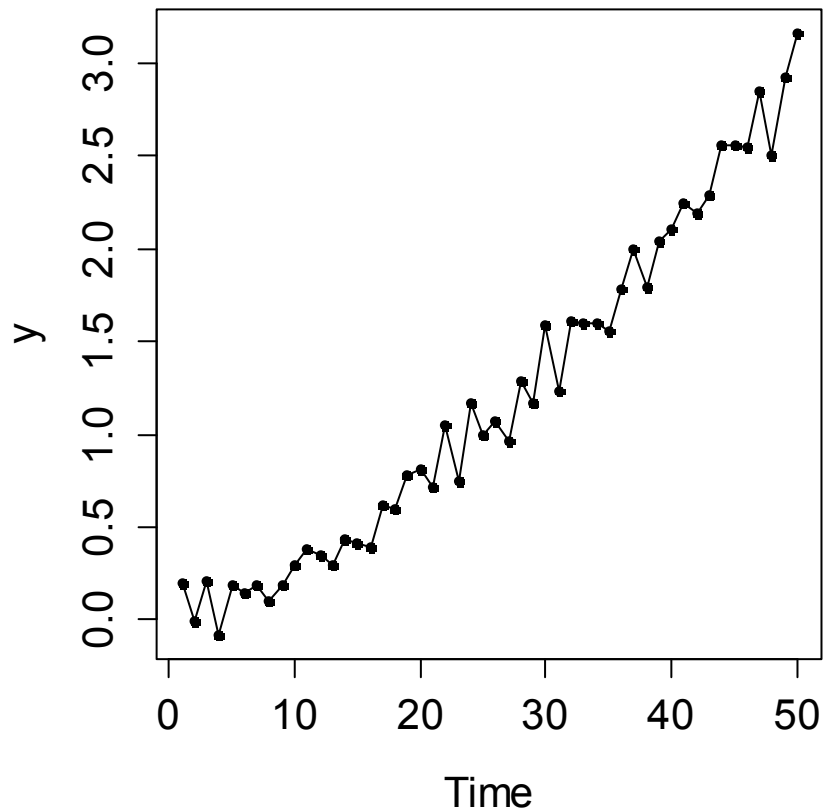
- 1) OLS
- 2) GLS

Applied Time Series Analysis

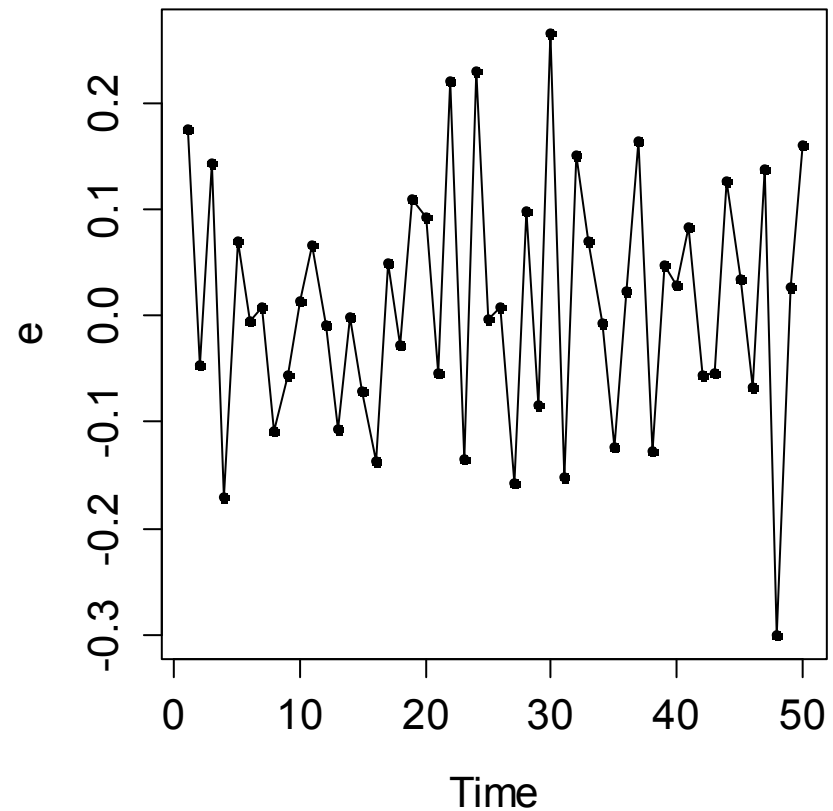
SS 2016 – Time Series Regression

Simulation Study: Series

Response Time Series



Error Time Series

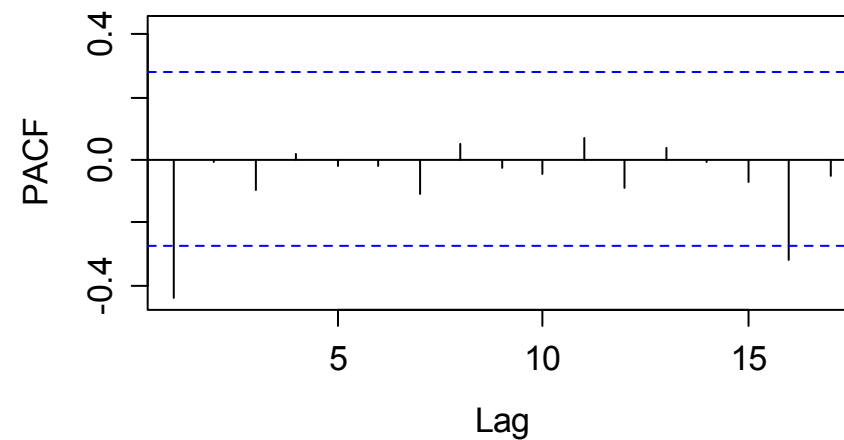
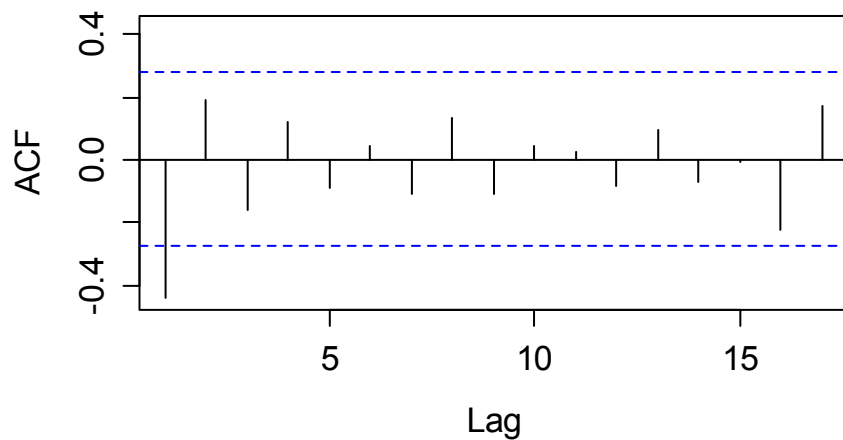
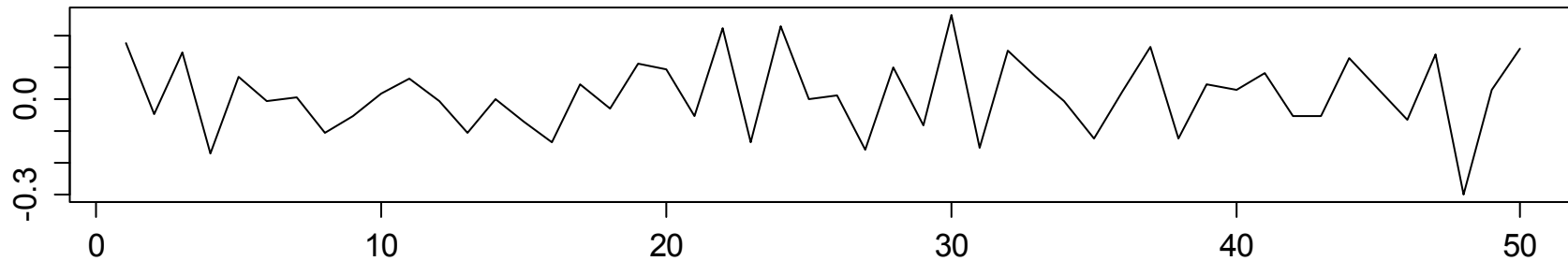


Applied Time Series Analysis

SS 2016 – Time Series Regression

Simulation Study: ACF/PACF of Errors

Error Time Series

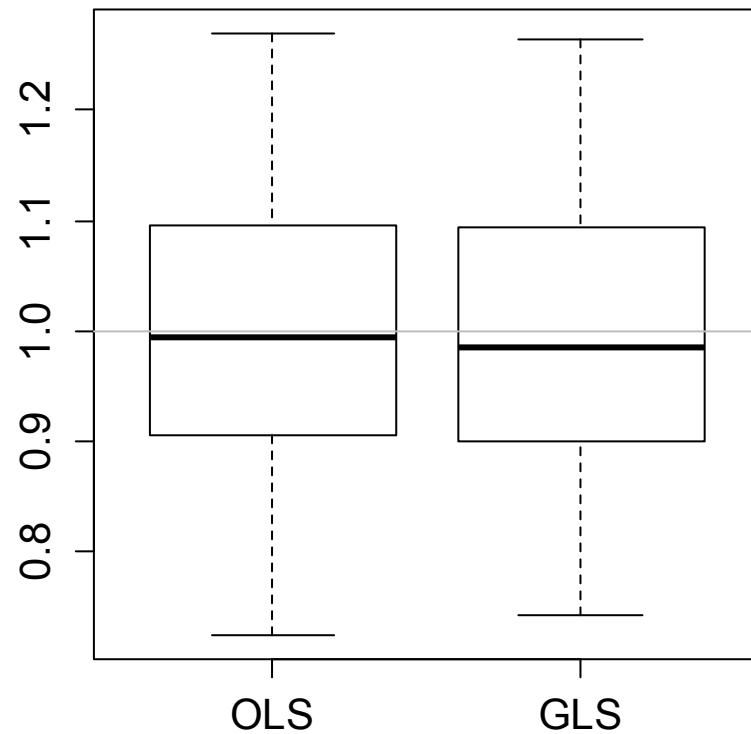


Applied Time Series Analysis

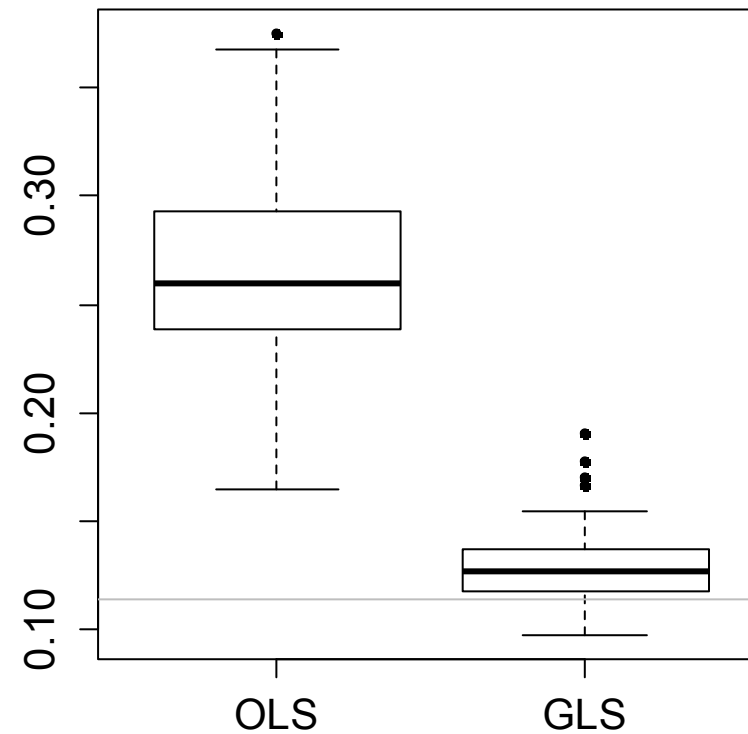
SS 2016 – Time Series Regression

Simulation Study: Results

Coefficient



Standard Error



Applied Time Series Analysis

SS 2016 – Time Series Regression

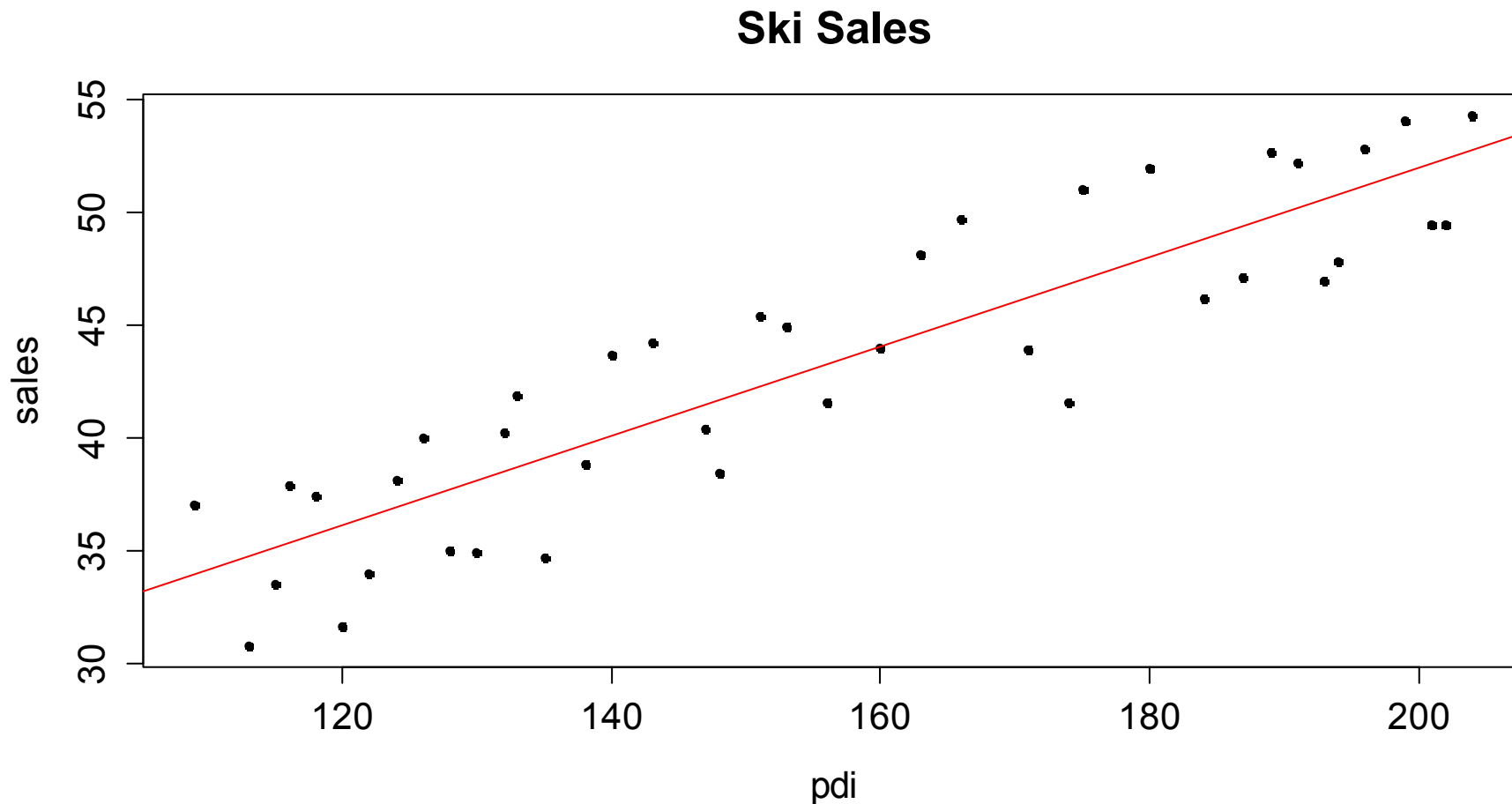
Missing Input Variables

- Correlated errors in (time series) regression problems are often caused by the absence of crucial input variables (time series).
 - In all these cases, it is much better to identify the not-yet-present variables and include them into the regression model.
 - However, in practice this isn't always possible, because these crucial variables may be non-available.
- **Time series regression methods for correlated errors such as GLS can be seen as a sort of emergency kit for the case where the non-present variables cannot be added. If you can do without them, even better!**

Applied Time Series Analysis

SS 2016 – Time Series Regression

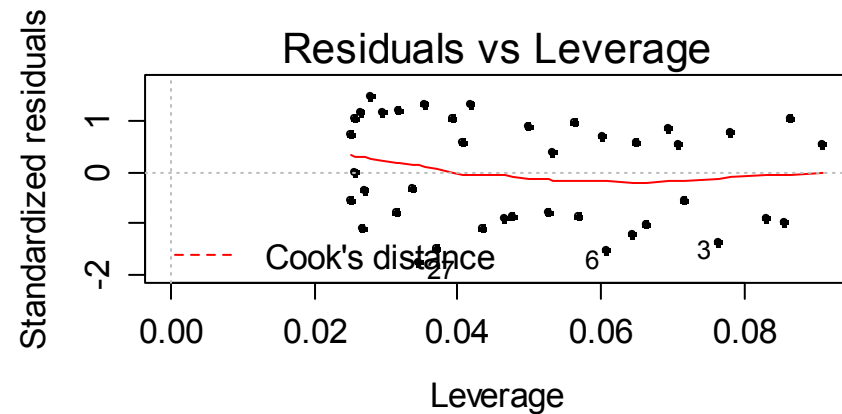
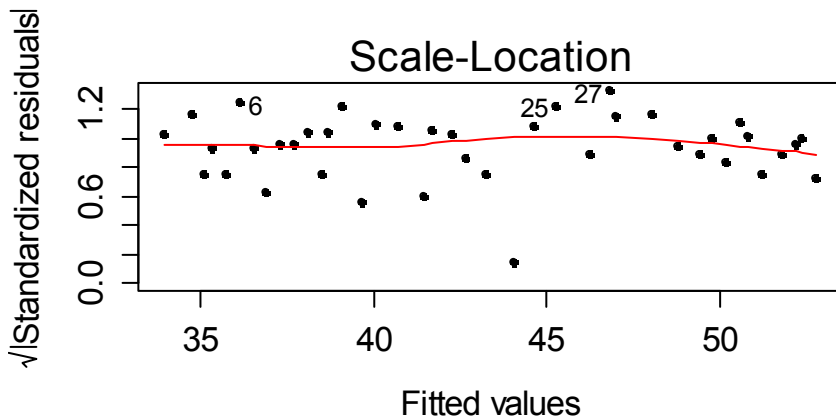
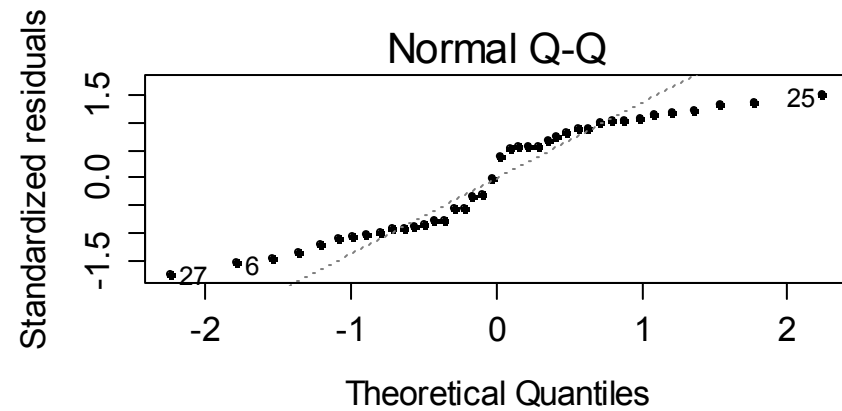
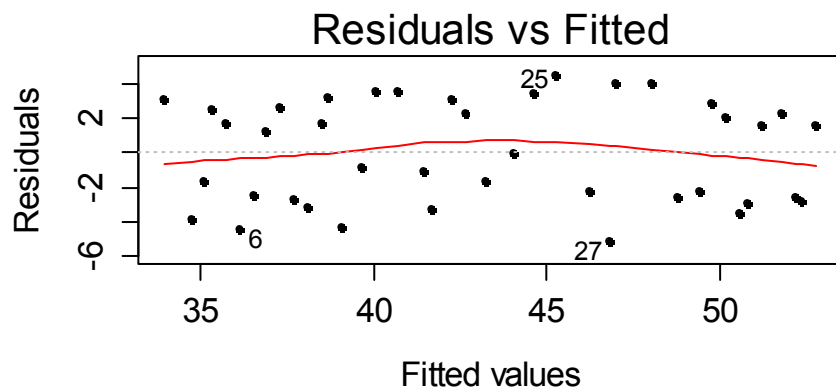
Example: Ski Sales



Applied Time Series Analysis

SS 2016 – Time Series Regression

Ski Sales: Residual Diagnostics

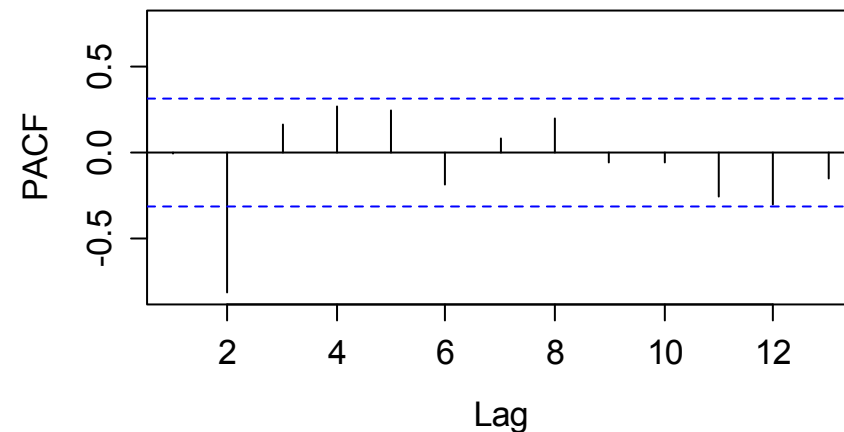
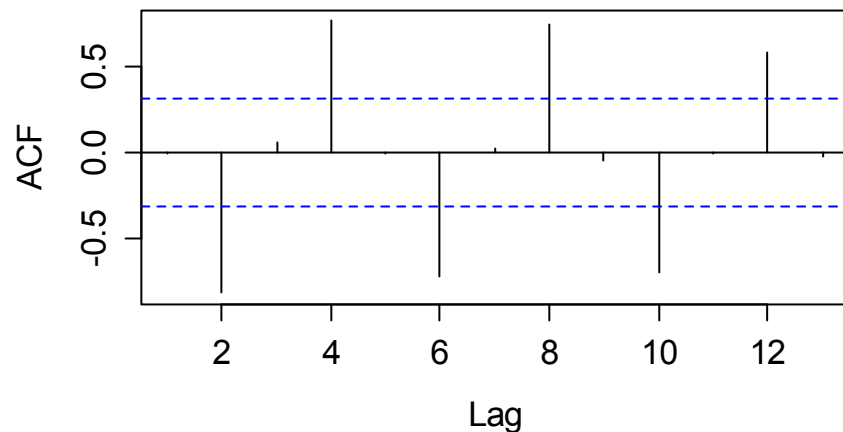
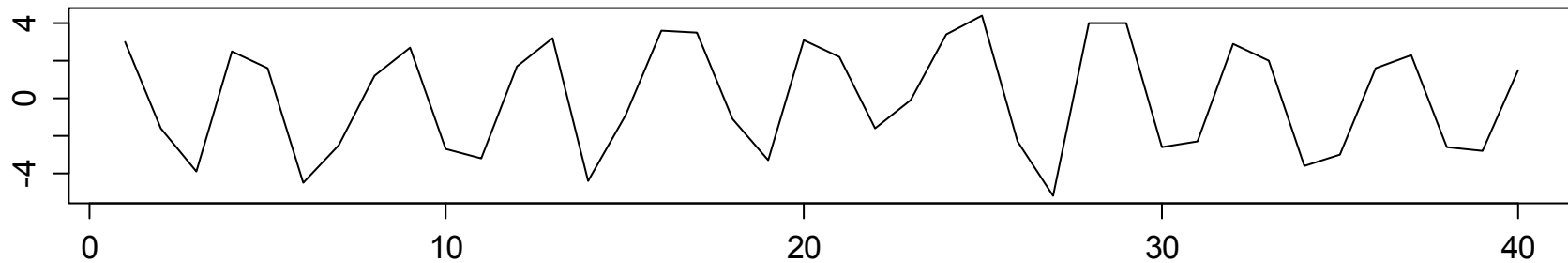


Applied Time Series Analysis

SS 2016 – Time Series Regression

Ski Sales: ACF/PACF of Residuals

Analysis of OLS Residuals



Applied Time Series Analysis

SS 2016 – Time Series Regression

Ski Sales: Durbin-Watson-Test

We perform a Durbin-Watson-Test for supporting our impression about the correlation of the OLS residuals:

```
> dwtest(fit)
data: fit
DW = 1.9684, p-value = 0.3933
alt. hypothesis: true autocorrelation > 0
```

The null hypothesis of “no autocorrelation” cannot be rejected in this case. This contradicts our findings from ACF/PACF analysis.

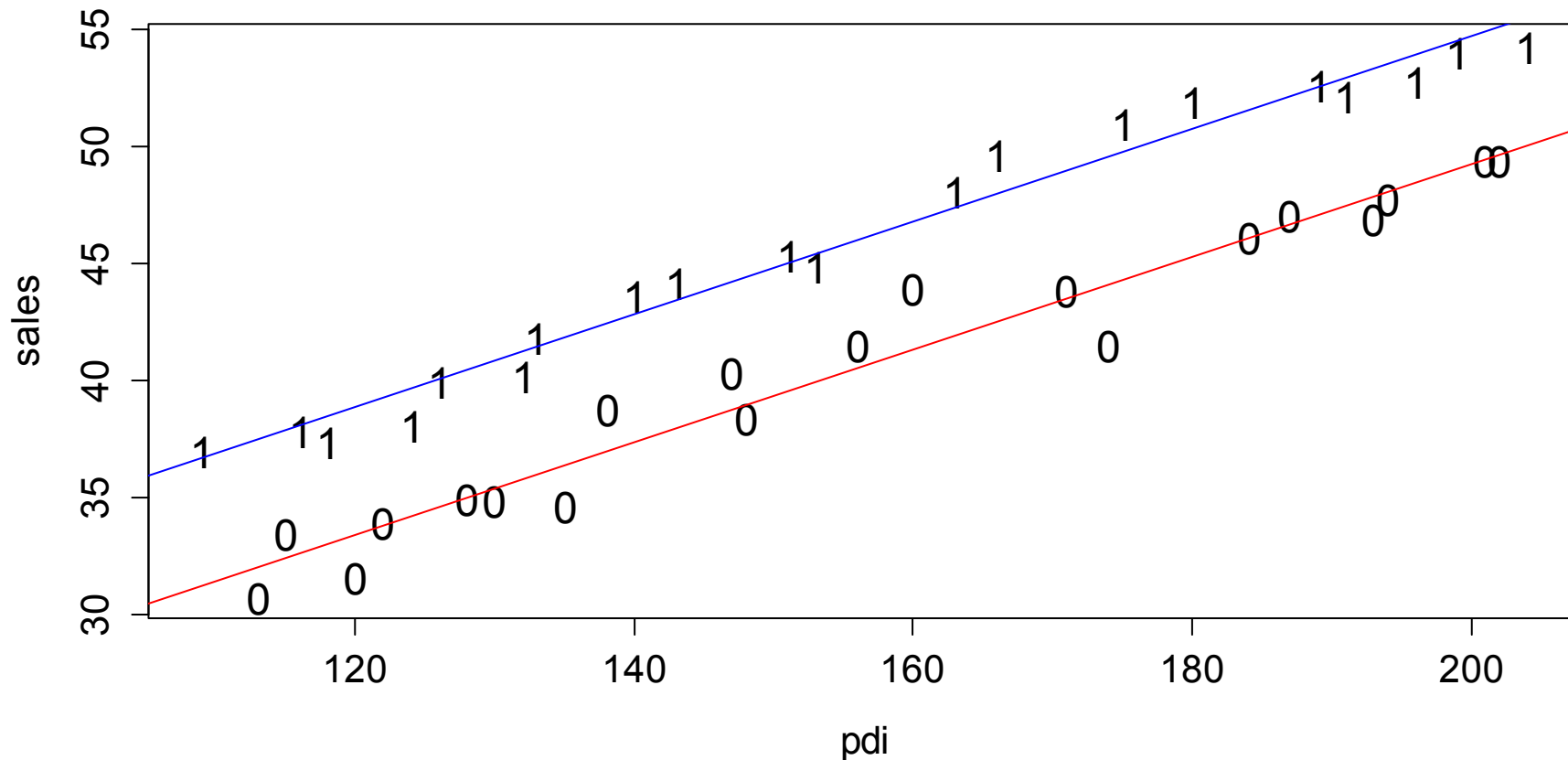
This is a case, where the Durbin-Watson-Test fails due to low power for situations, where $\rho(1)$ is small, but autocorrelation of significant magnitude exists at other lags.

Applied Time Series Analysis

SS 2016 – Time Series Regression

Ski Sales: Model with Seasonal Factor

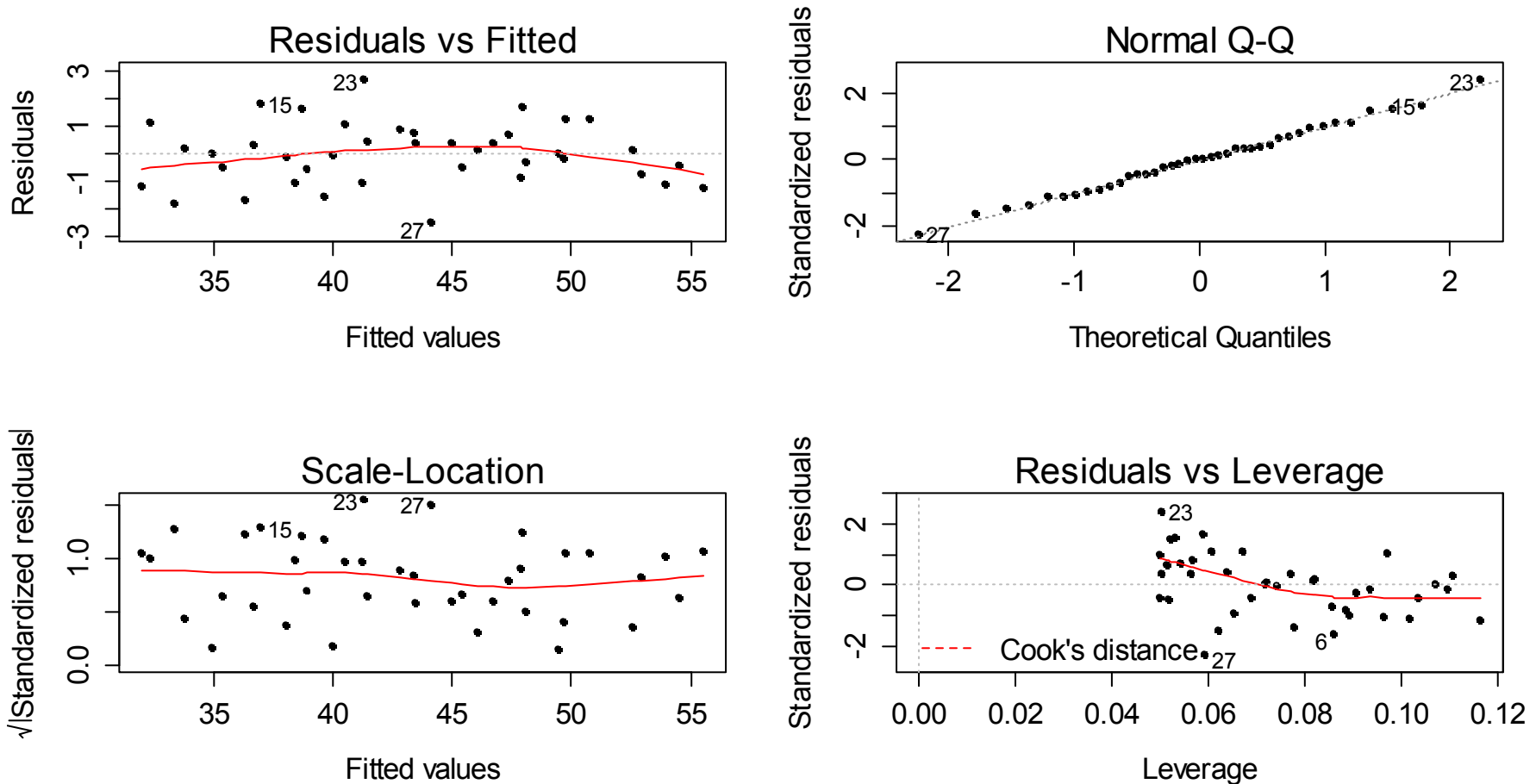
Ski Sales - Winter=1, Summer=0



Applied Time Series Analysis

SS 2016 – Time Series Regression

Residuals from Seasonal Factor Model

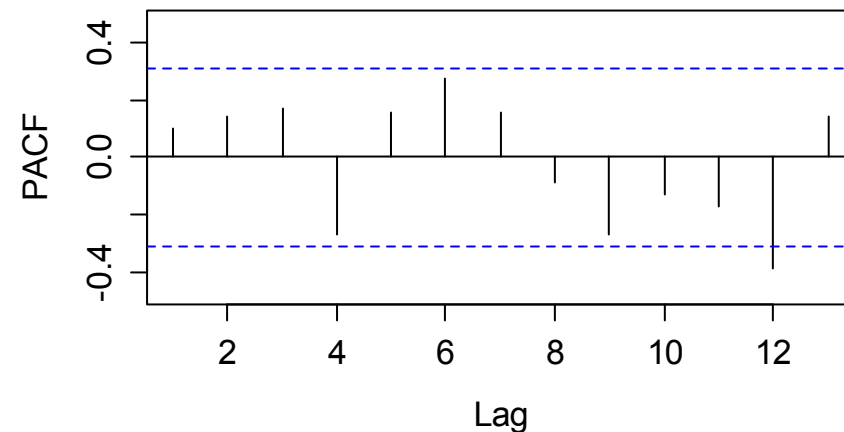
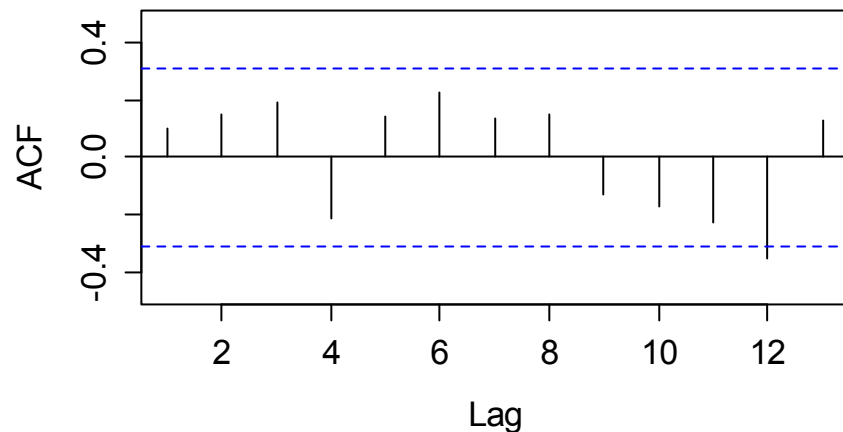
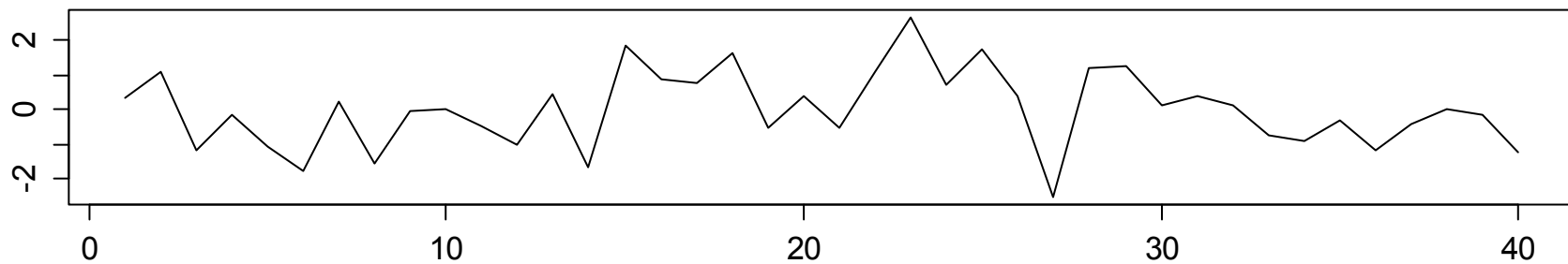


Applied Time Series Analysis

SS 2016 – Time Series Regression

Residuals from Seasonal Factor Model

Residuals of Extended Model



Applied Time Series Analysis

SS 2016 – Time Series Regression

Ski Sales: Summary

- The first model time series regression model (Sales vs. PDI) showed strongly correlated residuals, so that all inference results are to be considered as invalid.
 - The Durbin-Watson test failed to indicate this correlation, as $\rho(1)$ was very small, but significant autocorrelation at higher lags exists.
 - The correlated residuals are caused by omitting the season of the observation. Including this into the model cures the problem.
- The emergency kit of GLS is, after careful modeling, not even necessary in this example. This is often the case!**

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

ARIMA & SARIMA

Why?

Many time series in practice show trends and/or seasonality. While we can decompose them and describe the stationary part, it might be attractive to directly model them.

Advantages

Forecasting is convenient and AIC-based decisions for the presence of trend/seasonality become feasible.

Disadvantage

Lack of transparency for the decomposition and forecasting has a bit the flavor of a black-box-method.

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

ARIMA(p,d,q)-Models

ARIMA models are aimed at describing series that have a trend which can be removed by differencing, and where the differences can be described with an ARMA(p,q) model.

Definition: If $Y_t = X_t - X_{t-1} = (1-B)^d X_t \sim ARMA(p, q)$, then $X_t \sim ARIMA(p, d, q)$. In most practical cases, using $d = 1$ will be enough!

Notation: Very compact with the backshift operator $B()$:
 $\Phi(B)(1-B)^d X_t = \Theta(B)E_t$ is an $ARIMA(p, d, q)$

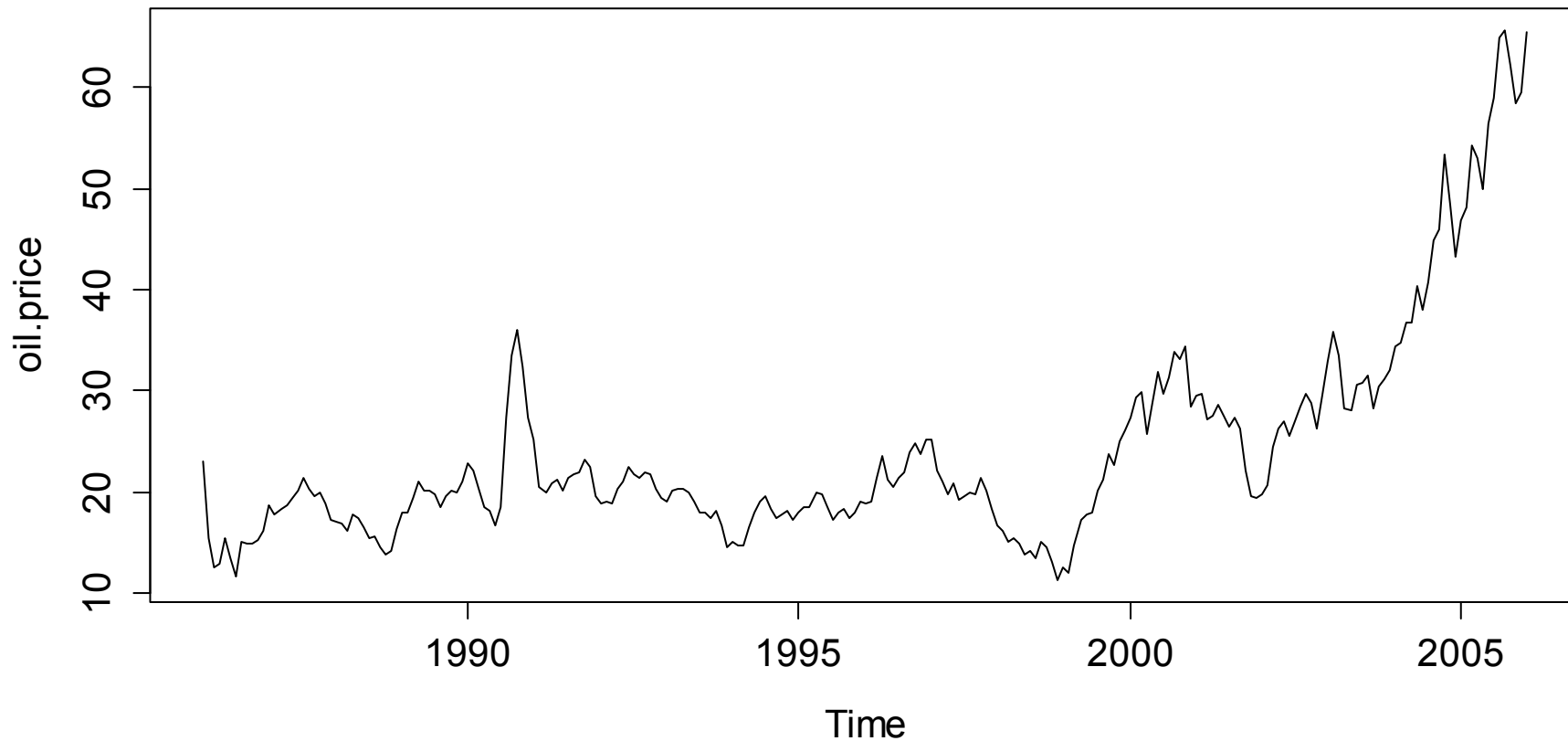
Stationarity: ARIMA processes are non-stationary if $d > 0$, option to rewrite as non-stationary ARMA(p,q).

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example: Monthly Crude Oil Prices

Monthly Price for a Crude Oil Barrel

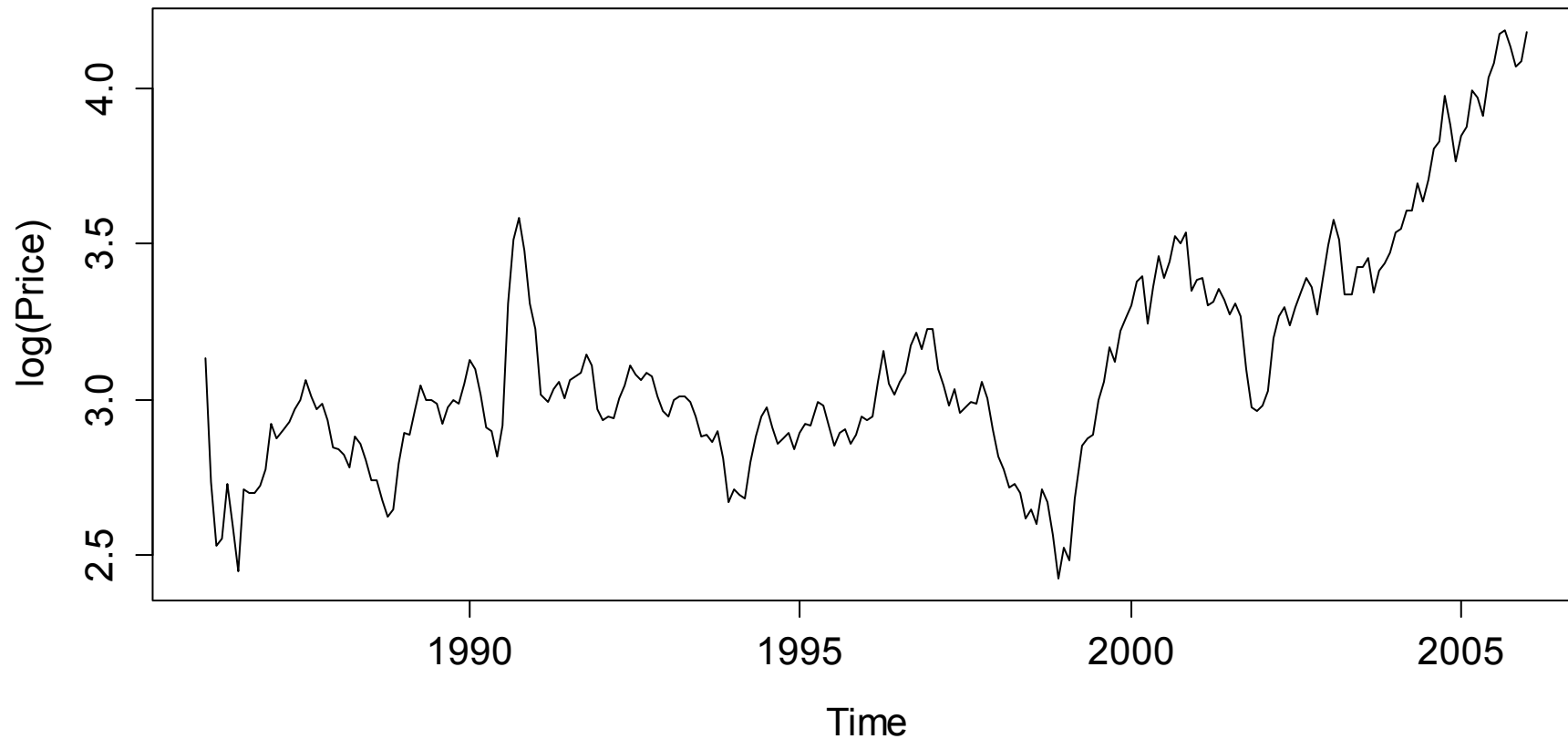


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example: Taking the Logarithm is Key

Logged Monthly Price for a Crude Oil Barrel

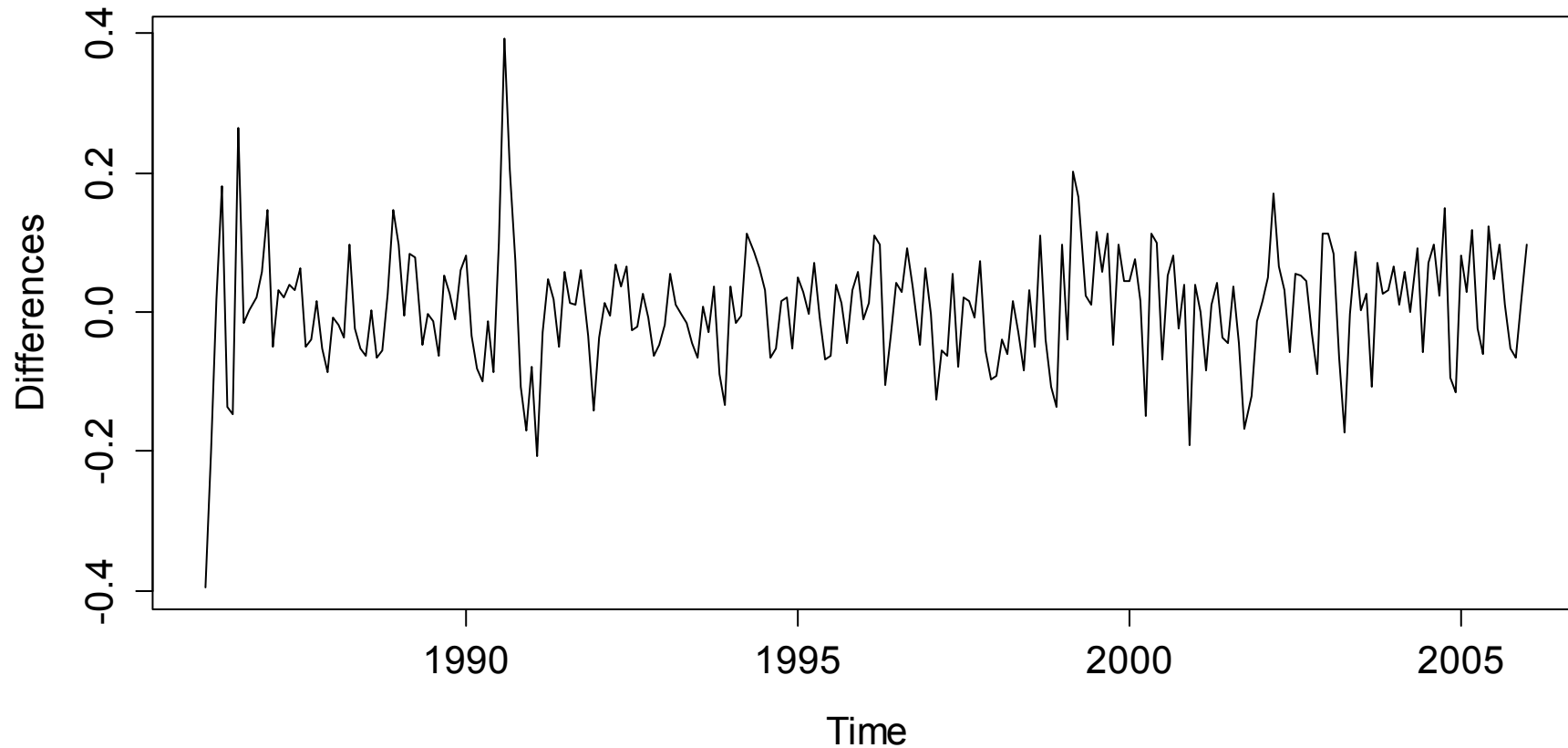


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example: Differencing for Stationarity

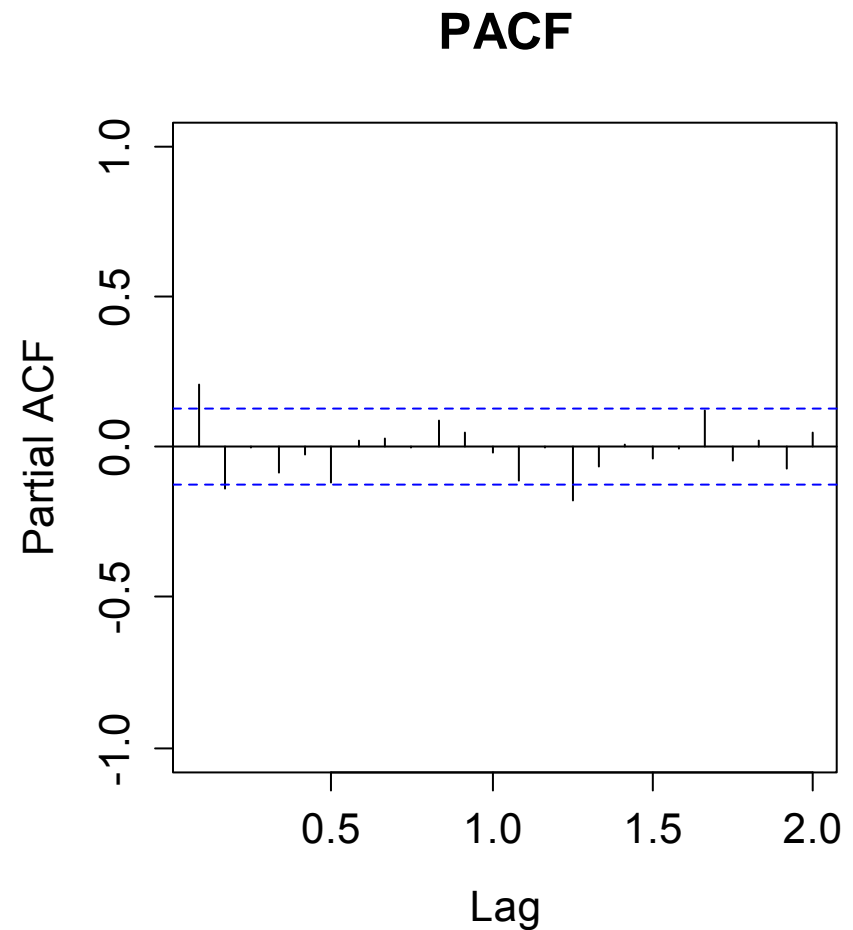
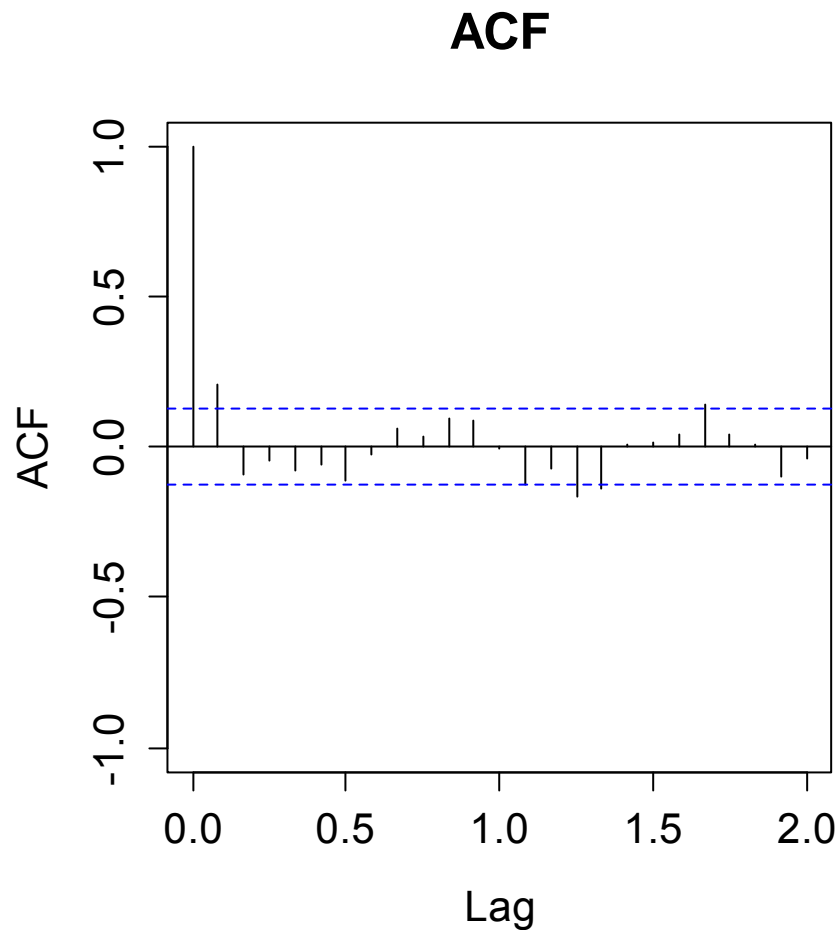
Differences of Logged Monthly Crude Oil Prices



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example: ACF/PACF of Differenced Series



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Fitting an ARIMA in R

Plausible models for the logged oil prices after inspection of ACF/PACF of the differenced series (that seems stationary):

→ **ARIMA(1,1,1)** or **ARIMA(2,1,1)** or **ARIMA(1,1,2)**

```
> arima(lop, order=c(1,1,2))
```

Coefficients:

	ar1	ma1	ma2
	0.8429	-0.5730	-0.3104
s.e.	0.1548	0.1594	0.0675

sigma² = 0.0066: ll = 261.88, aic = -515.75

Alternative R command with equivalent result:

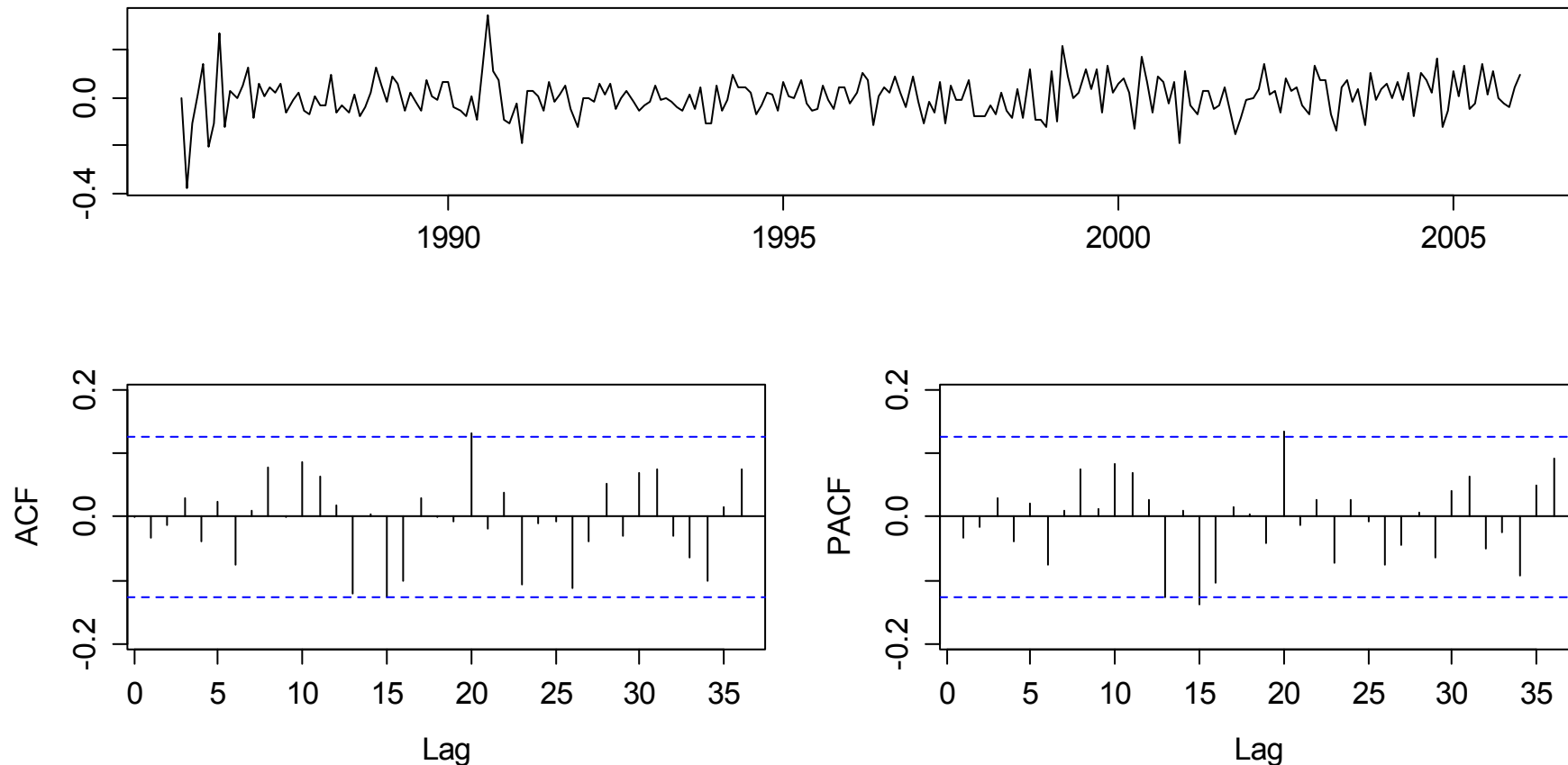
```
> arima(dlop, order=c(1,0,2), include.mean=FALSE)
```


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example: Residuals for ARIMA(1,1,2)

Residuals from ARIMA(1,1,2)



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Rewriting ARIMA as Non-Stationary ARMA

Any ARIMA(p,d,q) model can be rewritten in the form of a non-stationary ARMA((p+1),q) process. This provides some deeper insight, especially for the task of forecasting:

$$Y_t = 0.84 \cdot Y_{t-1} + E_t - 0.57 \cdot E_{t-1} - 0.31 \cdot E_{t-2}$$

$$X_t - X_{t-1} = 0.84 \cdot (X_{t-1} - X_{t-2}) + E_t - 0.57 \cdot E_{t-1} - 0.31 \cdot E_{t-2}$$

$$X_t = 1.84 \cdot X_{t-1} - 0.84 \cdot X_{t-2} + E_t - 0.57 \cdot E_{t-1} - 0.31 \cdot E_{t-2}$$

Verification with the `polyroot()` command in R shows:

```
> abs(polyroot(c(1, -1.84, 0.84)))  
[1] 1.000000 1.190476
```

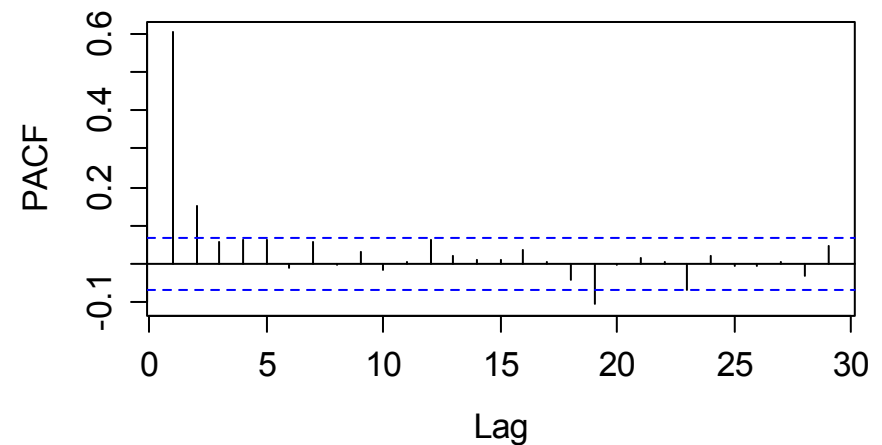
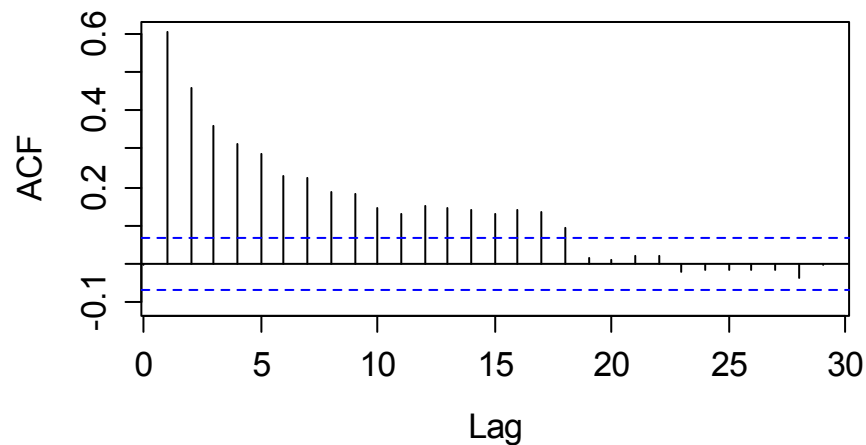
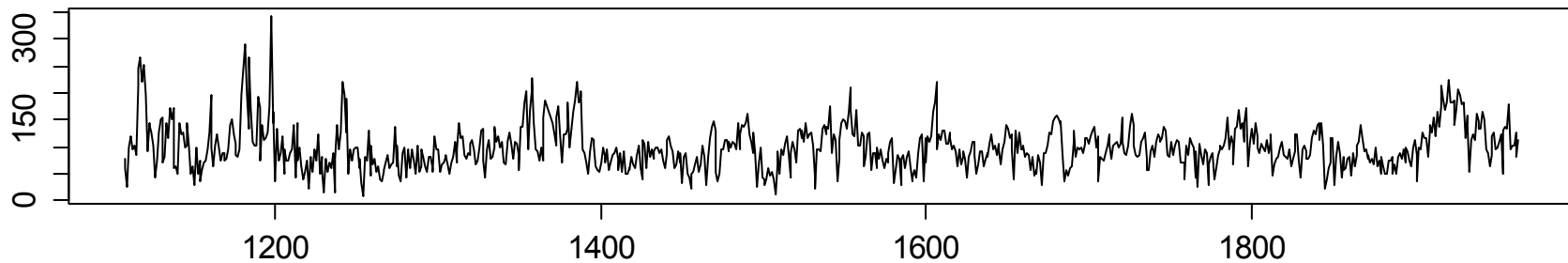
→ *There will always be a unit root!*

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example 2: Douglas Fir Data

Douglas Fir Tree Ring Width from 1107-1964

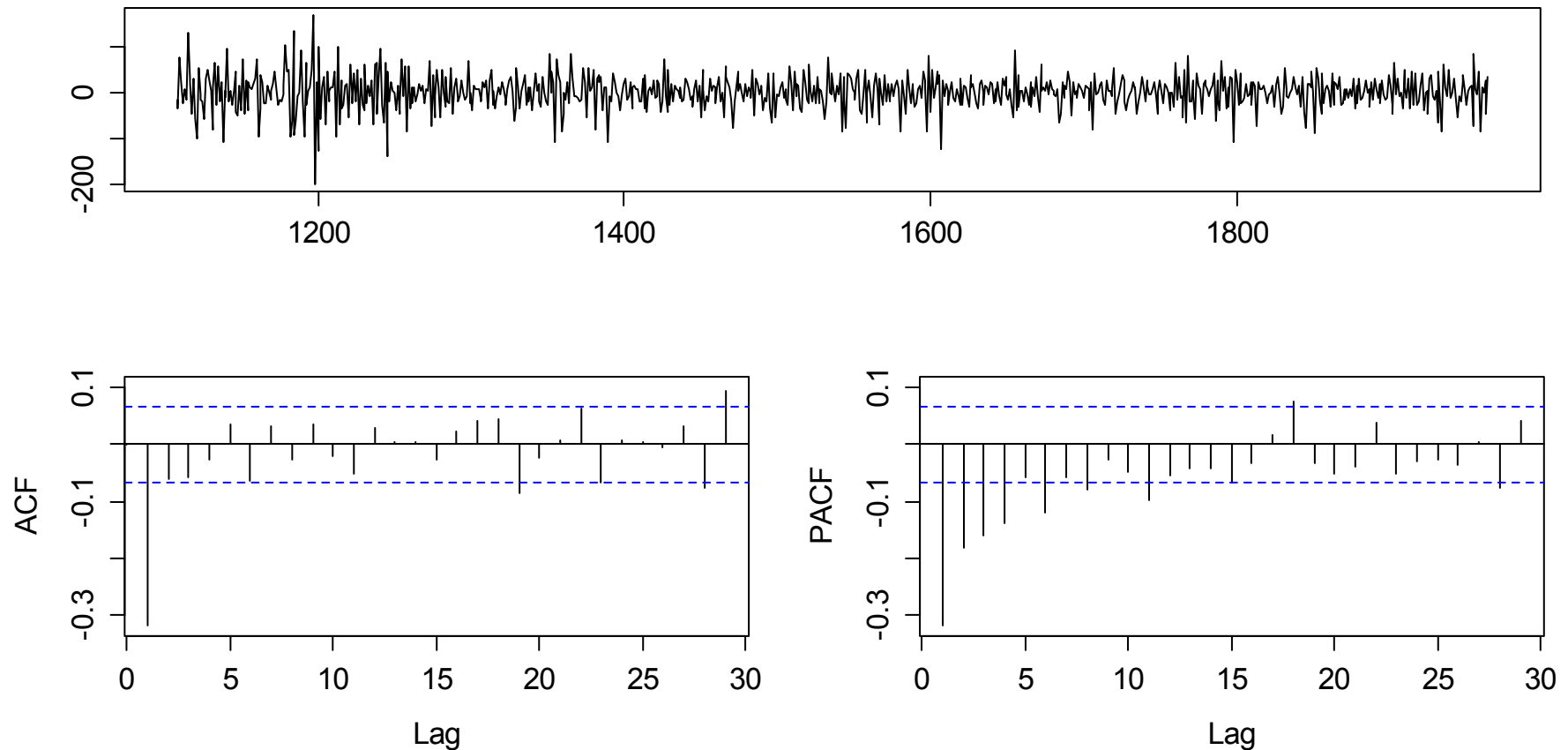


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example 2: After Differencing

Differenced Douglas Fir Tree Ring Width from 1107-1964



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example 2: Differencing or Not?

Some facts and statements:

- The ACF of the original data shows a relatively slow decay. However, the series still fits within what could have been produced by a stationary time series process.
- On the other hand, the differenced series looks “much more stationary” than the original one. It is also conceivable that data-generating process is an ARIMA.
- Stationary ARMA modelling would focus more on long term, i.e. climatic changes over time. Non-stationary modelling focuses more on short-term changes from year to year, i.e. local effects.

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example 2: Model Choice Original

- Analysis of ACF/PACF of the original series suggests using a ARIMA(2,0,0) or ARIMA(1,0,1) as parsimonious models.
- The residuals of these models look similar. ARIMA(1,0,1) has the lower AIC-value. According to `auto.arima()`, the model with minimal AIC is an ARIMA(1,0,3) though. That model is not suggestive according to ACF/PACF.

```
> fit <- auto.arima(douglasfir, max.p=5,  
  max.q=5, stationary=FALSE, seasonal=FALSE,  
  allowdrift=TRUE, allowmean=TRUE, ic="aic")
```

ARIMA(1,0,3) with non-zero mean

AIC=8410.22 AICc=8410.32 BIC=8438.75

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example 2: Model Choice Differenced

- Analysis of ACF/PACF of the differenced series suggests using a ARIMA(0,1,1) or ARIMA(1,1,1) as simple models.
- The ARIMA(0,1,1) model cannot capture the dependencies in a reasonable way, the residuals are not White Noise.
- The ARIMA(1,1,1) is much better and produces OK residuals. Its AIC values is slightly worse than the ARIMA(1,0,1) though.

```
> fit <- auto.arima(diff(douglasfir), max.p=5,
  max.q=5, stationary=FALSE, seasonal=FALSE,
  allowdrift=TRUE, allowmean=TRUE, ic="aic")
```

```
Series: diff(douglasfir)
```

```
ARIMA(1,0,2) with zero mean
```

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Guidelines for Fitting ARIMA Models

- 1) If you recognize a non-stationary series with a trend and without seasonal effect, choose the appropriate order of differencing at lag 1, usually $d = 1$.
- 2) Analyze ACF/PACF of the differenced series. If the stylized facts of an ARMA process are present, decide for p, q .
- 3) Fit the model using the `arima()` procedure. This is best done with the original series as the input and setting d .
- 4) Analyze the residuals, these must look like White Noise. If several competing models are appropriate, use AIC for a second opinion. Function `auto.arima()` may be handy.

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

SARIMA(p,d,q)(P,D,Q)^s

We have learned that it is also possible to use differencing for obtaining a stationary series out of one that features both trend and seasonal effect.

- 1) Removing the seasonal effect by differencing at lag 12

$$Y_t = X_t - X_{t-12} = (1 - B^{12})X_t$$

- 2) Usually, further differencing at lag 1 is required to obtain a series that has constant global mean and is stationary

$$Z_t = Y_t - Y_{t-1} = (1 - B)Y_t = (1 - B)(1 - B^{12})X_t = X_t - X_{t-1} - X_{t-12} + X_{t-13}$$

The stationary series Z_t is then modelled with some special kind of ARMA(p,q) models, see the forthcoming slides.

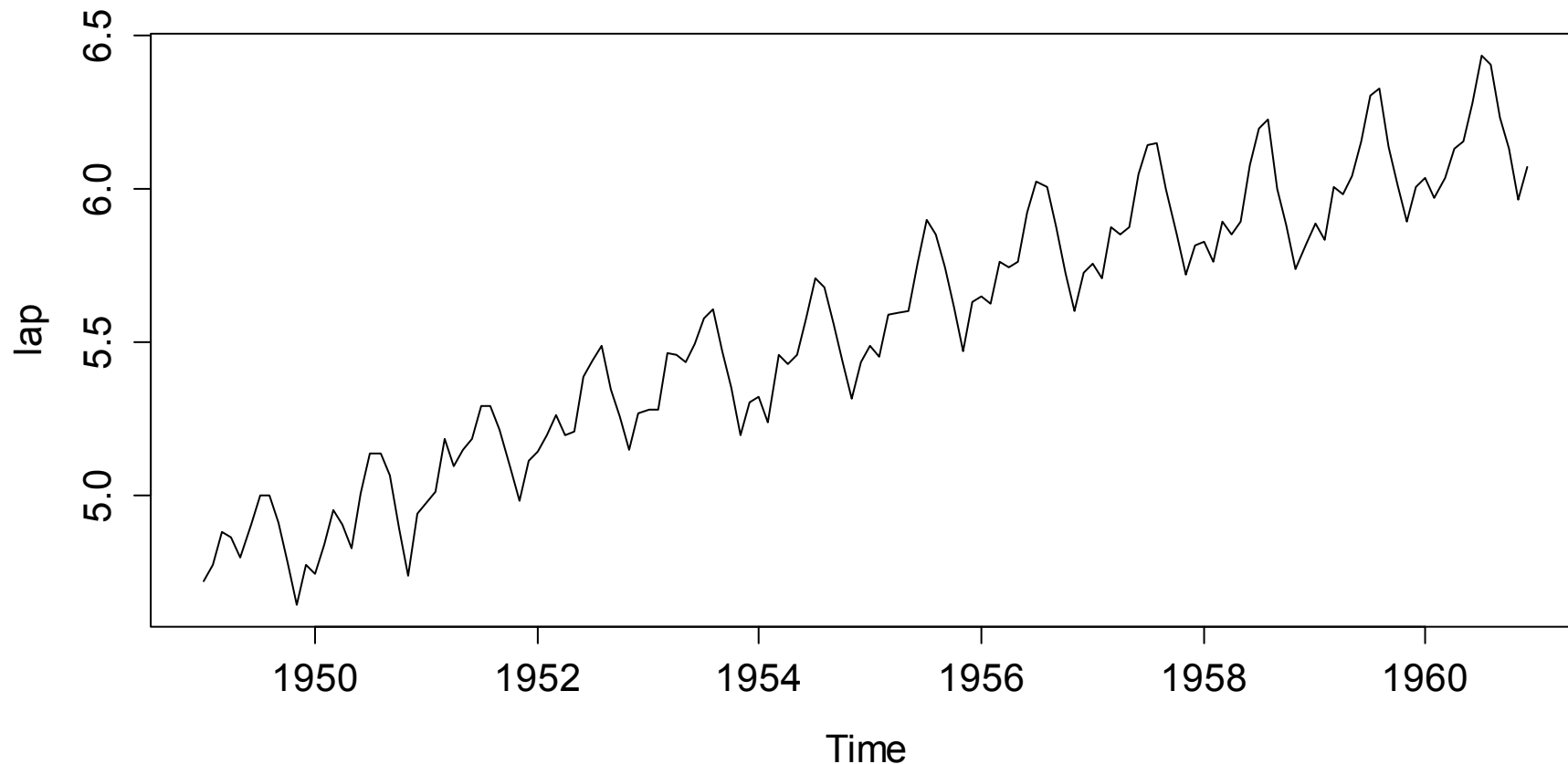
Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

SARIMA(p,d,q)(P,D,Q)^s

= a.k.a. *Airline Model*. We are looking at the log-trsf. airline data

Logged Air Passengers

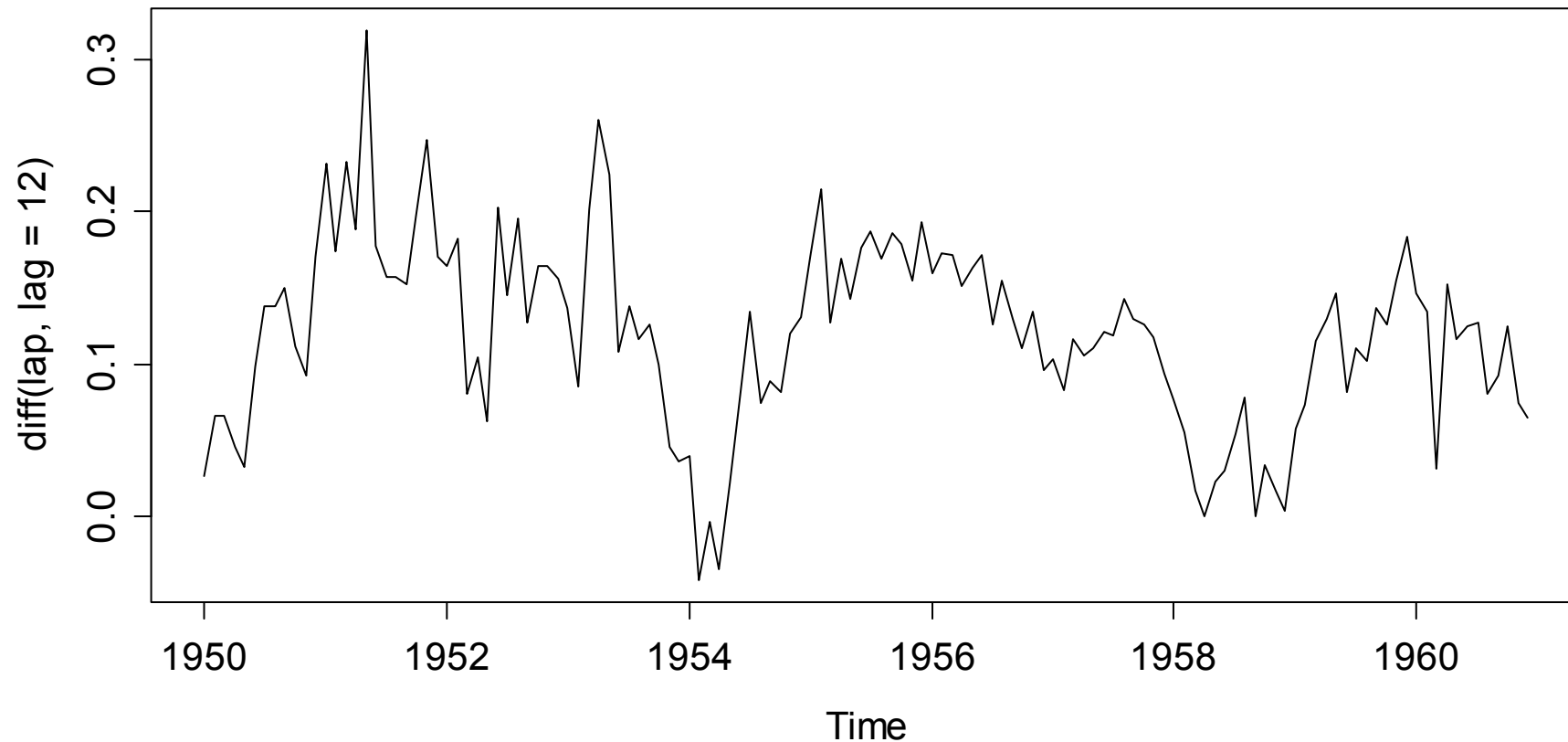


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Seasonal Differencing Helps...

Seasonally Differenced Airline Passenger Series

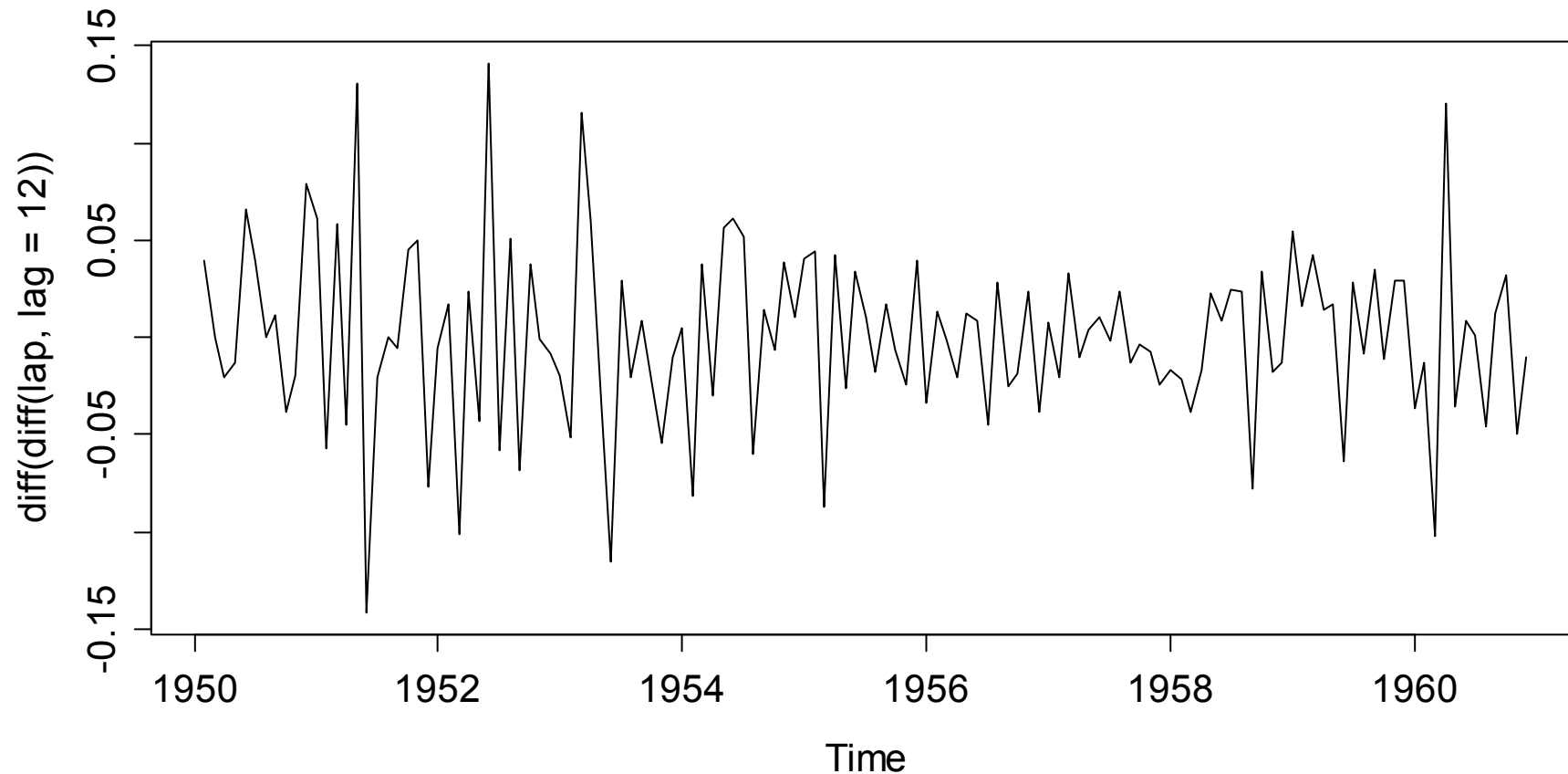


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

... But More Is Needed!

Differenced Seasonally Differenced Airline Passenger Series

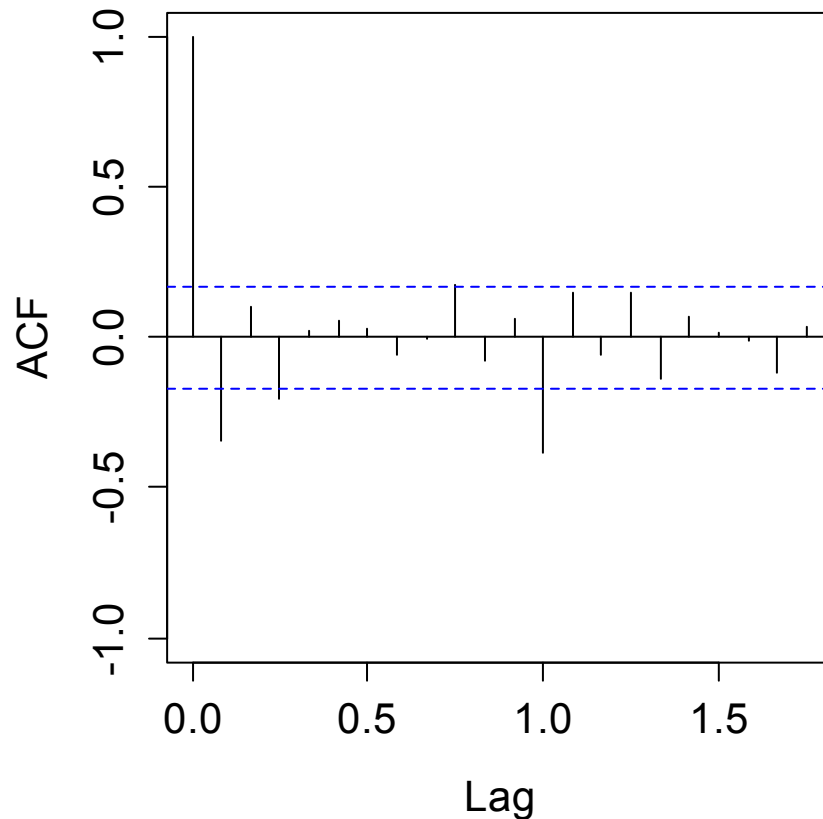


Applied Time Series Analysis

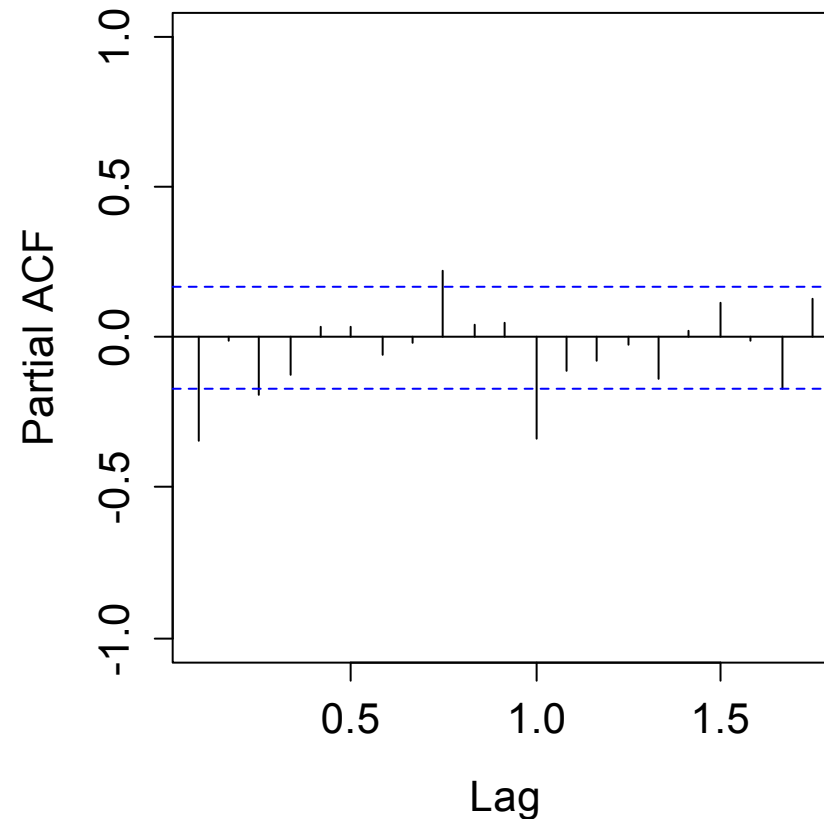
SS 2016 – ARIMA, SARIMA & GARCH

Analysis of ACF & PACF

ACF of Twice Differenced Series



PACF of Twice Differenced Series



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Stylized Facts in ACF & PACF

Twice differenced data show typical behavior in ACF/PACF: there is usually some significant short term dependence over the first couple of lags, as well as significant autocorrelation at multiples of the period s .

- This suggests that large p, q are required for describing the data with ARMA models, making them non-parsimonious.
- We may overcome this problem by using the *airline model*

$$\begin{aligned} Z_t &= (1 + \beta_1 B)(1 + \xi_1 B^{12})E_t \\ &= E_t + \beta_1 E_{t-1} + \xi_1 E_{t-12} + \beta_1 \xi_1 E_{t-13} \end{aligned}$$

This is an MA(13) where most coefficients are zero.

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

***SARIMA*(p, d, q)(P, D, Q)^s**

The *airline model* $Z_t = (1 + \beta_1 B)(1 + \xi_1 B^{12})E_t$ from the previous slide is a *SARIMA*(0,1,1)(0,1,1)¹² for the logged air passenger data X_t .

Definition: A series X_t follows a *SARIMA*(p, d, q)(P, D, Q)^s process if the following equation holds.

$$\Phi(B)\Phi_s(B^s)Z_t = \Theta(B)\Theta_s(B^s)E_t$$

Here, series Z_t originated from X_t after appropriate seasonal and trend differencing: $Z_t = (1 - B)^d (1 - B^s)^D X_t$

In most practical cases, using differencing order $d = D = 1$ will be sufficient. Choosing of p, q, P, Q happens via ACF/PACF or via AIC-based decisions.

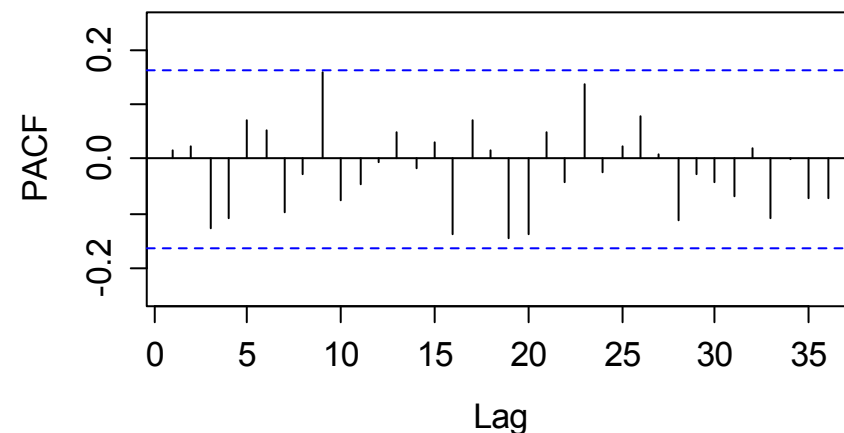
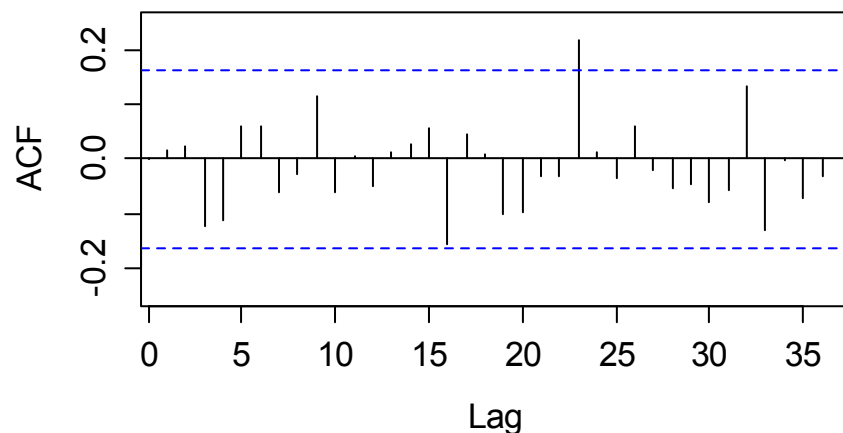
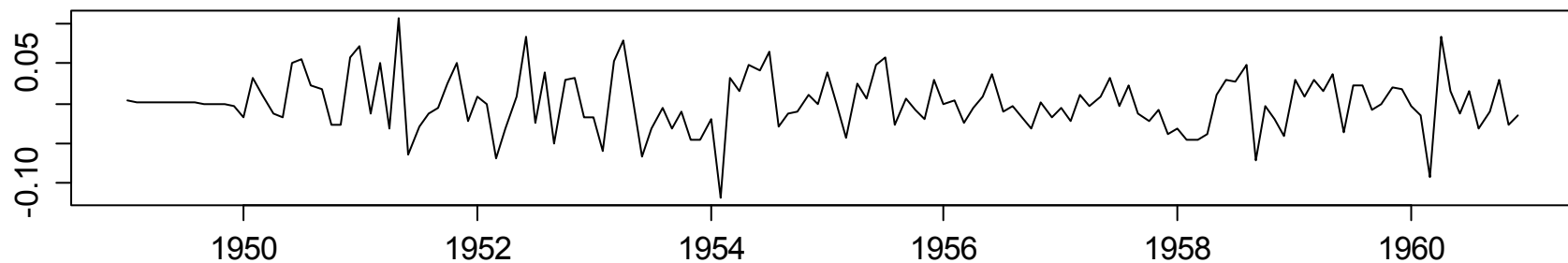
Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Fit & Residuals from the Airline Model

```
> fit <- arima(lap, order=c(0,1,1), seasonal=c(0,1,1))
```

Residuals from SARIMA(0,1,1)(0,1,1)



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Using auto.arima() for Model Selection

The results is a $SARIMA(0,1,1)(2,1,2)^{12}$...

```
> fit <- auto.arima(lap, ic = "aic"); fit
```

```
Series: lap
```

```
ARIMA(0,1,1)(2,1,2)[12]
```

```
Coefficients:
```

	ma1	sar1	sar2	sma1	sma2
	-0.3632	-0.4852	-0.0933	-0.0204	-0.1803
s.e.	0.0949	0.1561	0.1107	0.1700	0.1117

```
sigma^2 = 0.001258; log likelihood = 251.52
```

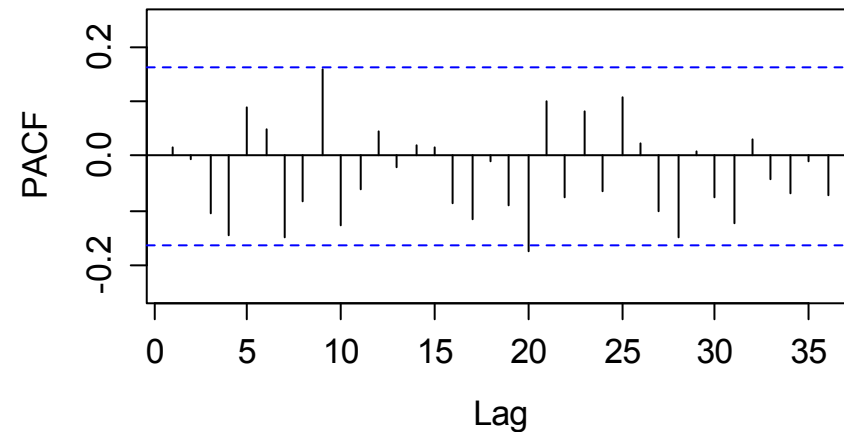
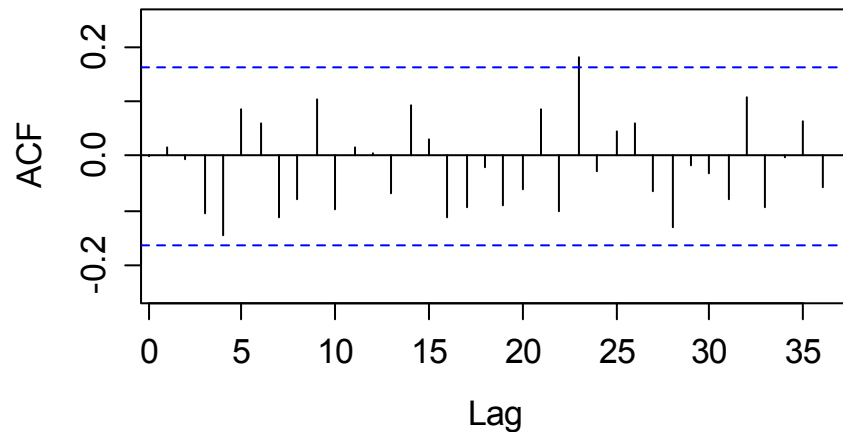
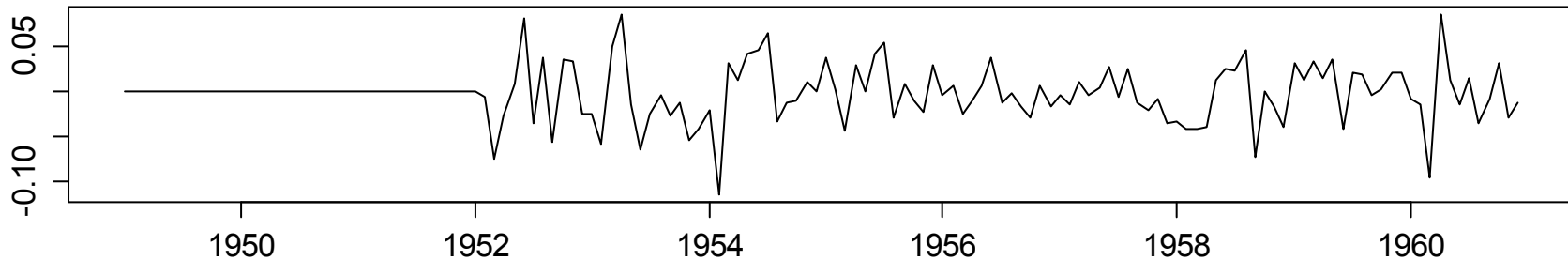
```
AIC=-409.06      AICc=-408.39      BIC=-391.81
```

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Residuals from auto.arima() Fit

Residuals from SARIMA(0,1,1)(2,1,2)

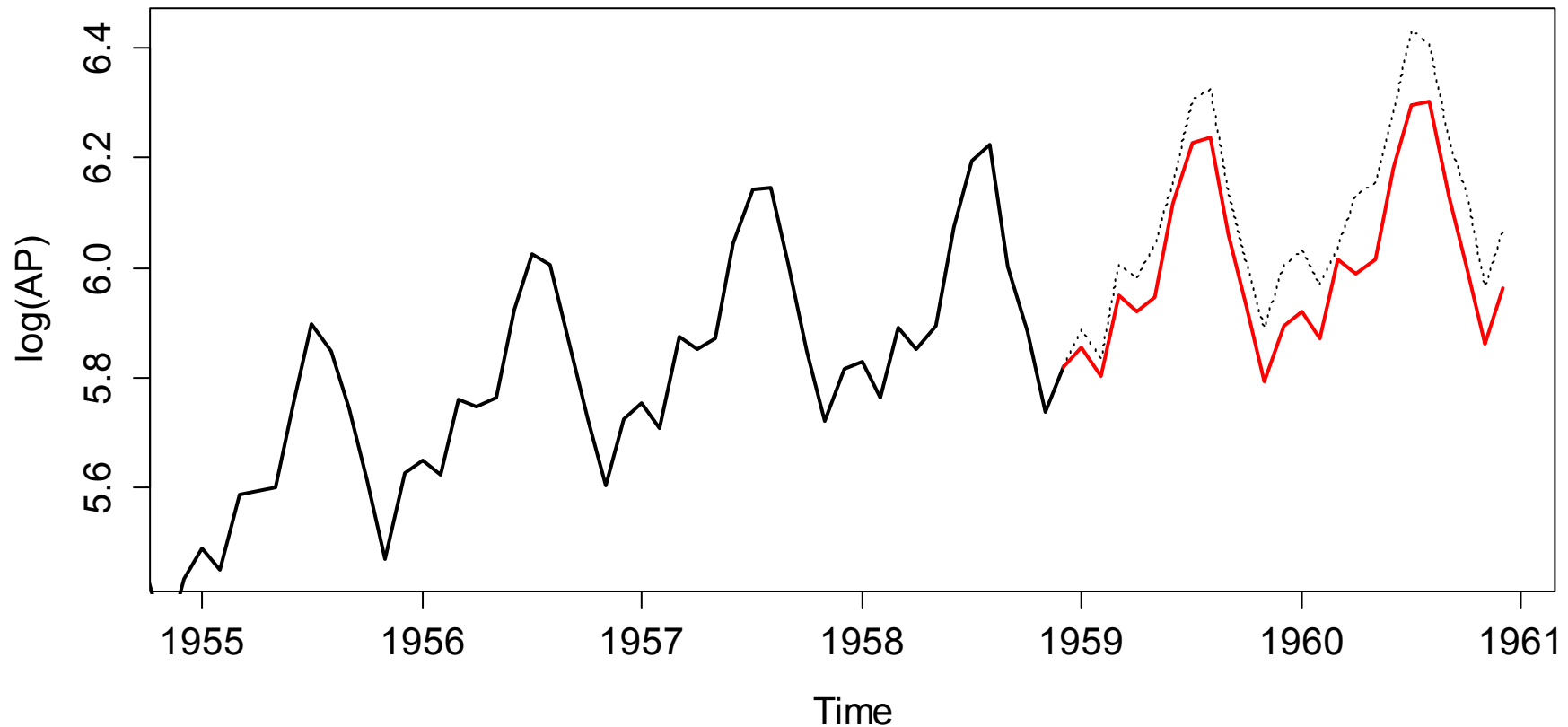


Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Outlook: Forecasting with SARIMA

Forecast of $\log(\text{AP})$ with SARIMA(0,1,1)(0,1,1)



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Guidelines for Fitting SARIMA Models

- 1) Perform seasonal differencing of the data. The lag s is determined by the period. Order $D = 1$ is mostly enough.
- 2) Decide if additional differencing at lag 1 is required for stationarity. If not, then $d = 0$. If yes, then try $d = 1$.
- 3) Analyze ACF/PACF of Z_t to determine p, q for the short term and P, Q at multiple-of-the-period dependency.
- 4) Fit the model using `arima()` by setting `order=c(p, d, q)` and `seasonal=c(P, D, Q)` accordingly to your choices.
- 5) Check the accuracy of the model by residual analysis. The residuals must look like White Noise and +/- Gaussian.

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Outlook to Non-Linear Models

What are linear models?

Models which can be written as a linear combination of X_t .
This includes all invertible AR-, MA- and ARMA-models.

What are non-linear models?

Everything else, e.g. non-linear combinations of X_t ,
terms like X_t^2 in the linear combination, and much more!

Motivation for non-linear models?

- cyclic behavior with quicker increase than decrease
- volatility models with conditional heteroskedasticity

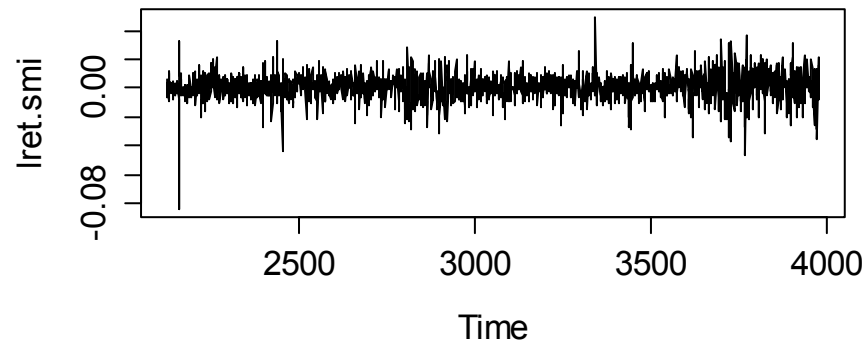
We here only present a brief outlook to volatility models.

Applied Time Series Analysis

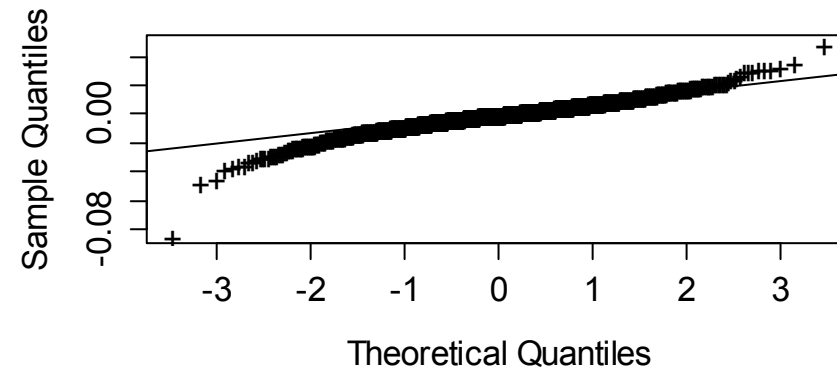
SS 2016 – ARIMA, SARIMA & GARCH

Volatility Models: Example & Motivation

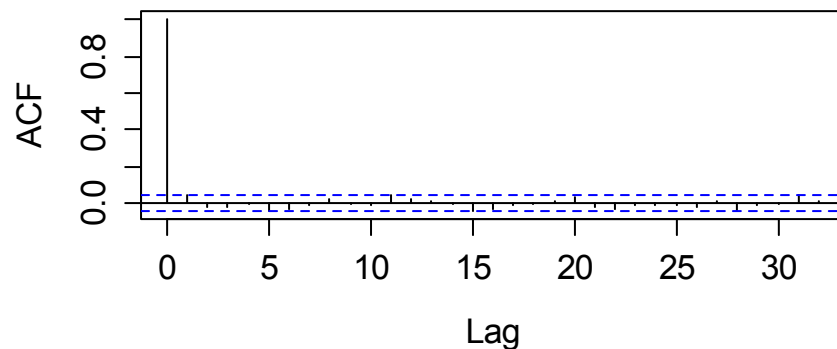
SMI Log-Returns



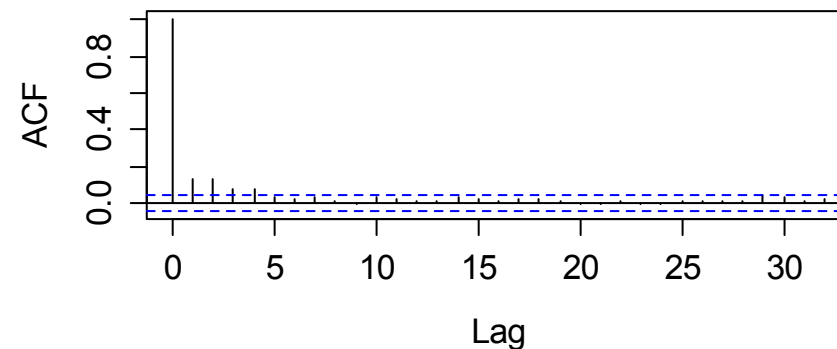
Normal Plot



ACF of SMI Log-Returns



ACF of Squared Log-Returns



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Example: Stylized Facts

Many real-world time series, especially the log-returns of financial series exhibit a phenomenon called conditional heteroskedasticity.

- *periods of higher and lower volatility*
- *long-tailed appearance in the Normal Plot*
- *no significant autocorrelation in the series*
- *significant autocorrelation if the series is squared*

Despite not showing direct autocorrelation, these series are not White Noise. There is dependency which can be exploited by using conditional heteroskedasticity (ARCH/GARCH) models.

Note: Time series with conditional heteroskedasticity can be stationary, i.e. ARCH/GARCH models are stationary.

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

ARCH/GARCH Models

The basic assumption for ARCH/GARCH models is as follows:

$$X_t = \mu_t + E_t, \text{ where } E_t = \sigma_t W_t \text{ and } W_t \text{ is White Noise}$$

Here, both the conditional mean and variance are non-trivial

$$\mu_t = E[X_t | X_{t-1}, X_{t-2}, \dots] \text{ and } \sigma_t = \text{Var}[X_t | X_{t-1}, X_{t-2}, \dots]$$

and can be modelled using a mixture of ARMA & GARCH.

For simplicity, we here assume that both the conditional and the global mean are zero $\mu = \mu_t = 0$ and consider pure ARCH processes only where:

$$X_t = \sigma_t W_t \text{ with } \sigma_t = f(X_{t-1}^2, X_{t-2}^2, \dots, X_{t-p}^2)$$

Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Definition of the ARCH(p) Model

A time series X_t is autoregressive conditional heteroskedastic of order p , abbreviated ARCH(p), if:

$$X_t = \sigma_t W_t \quad \text{with} \quad \sigma_t = \sqrt{\alpha_0 + \sum_{i=1}^p \alpha_i X_{t-i}^2}.$$

It is obvious that an ARCH(p) process shows volatility, as:

$$\text{Var}(X_t) = \alpha_0 + \alpha_1 \cdot \text{Var}(X_{t-1}) + \dots + \alpha_p \cdot \text{Var}(X_{t-p})$$

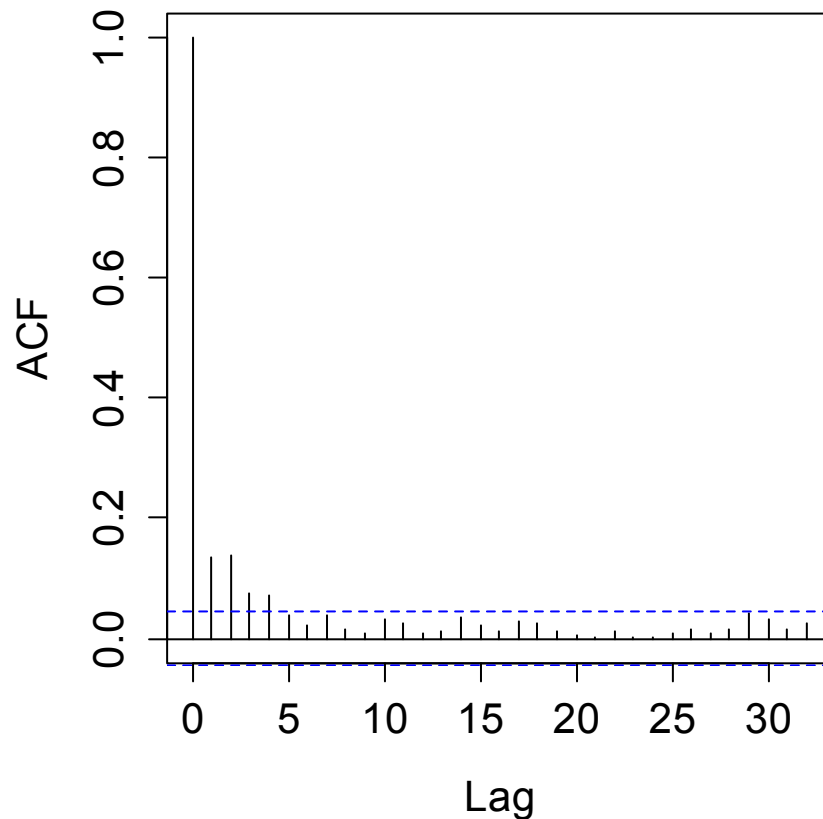
We can determine the order of an ARCH(p) process in practice by analyzing ACF and PACF of the squared time series data. We then again search for an exponential decay in the ACF and a cut-off in the PACF.

Applied Time Series Analysis

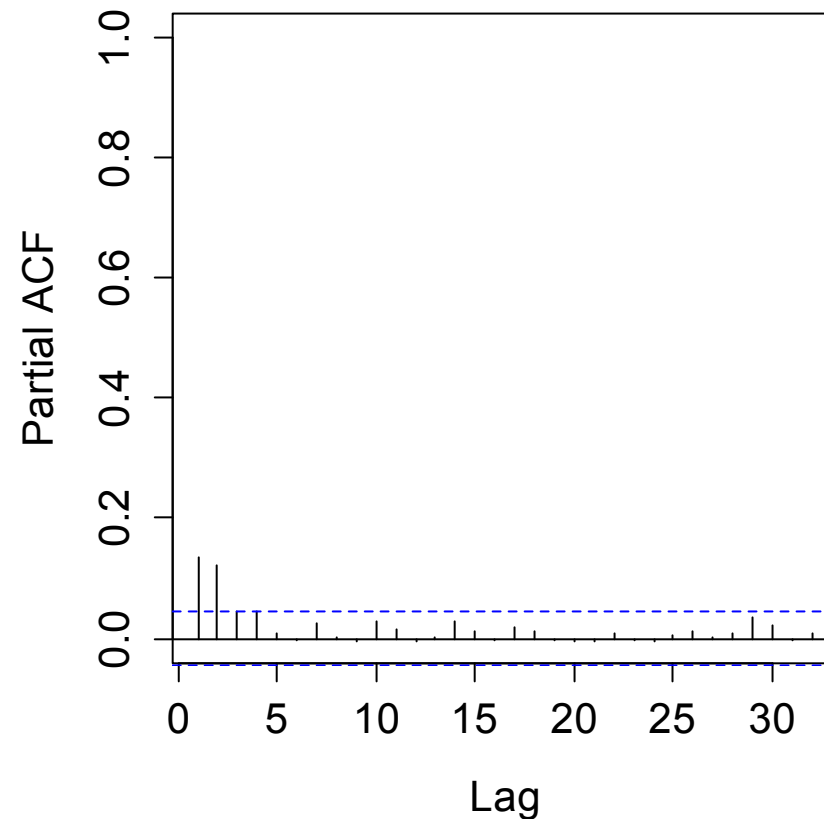
SS 2016 – ARIMA, SARIMA & GARCH

Model Order for SMI Log-Returns

ACF of Squared Log-Returns



PACF of Squared Log-Returns



Applied Time Series Analysis

SS 2016 – ARIMA, SARIMA & GARCH

Fitting an ARCH(2) Model in R

The simplest option for fitting an ARCH(p) in R is to use function `garch()` from `library(tseries)`. Be careful, because the `order=c(q,p)` argument differs from most of the literature.

```
> fit <- garch(lret.smi, order = c(0,2))  
> fit
```

```
Call: garch(x = lret.smi, order = c(0, 2))
```

```
Coefficient(s):
```

a0	a1	a2
6.568e-05	1.309e-01	1.074e-01

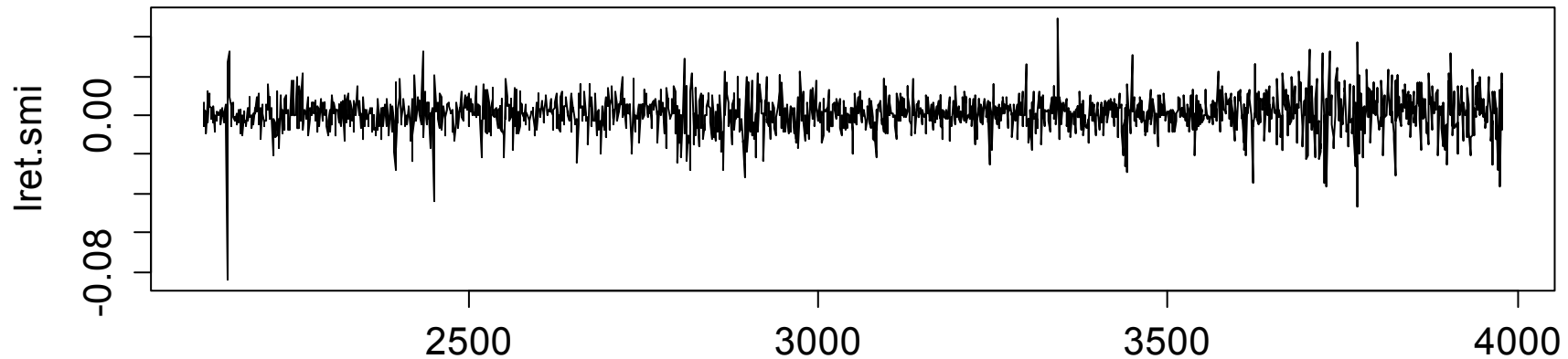
We recommend to run residual analysis afterwards.

Applied Time Series Analysis

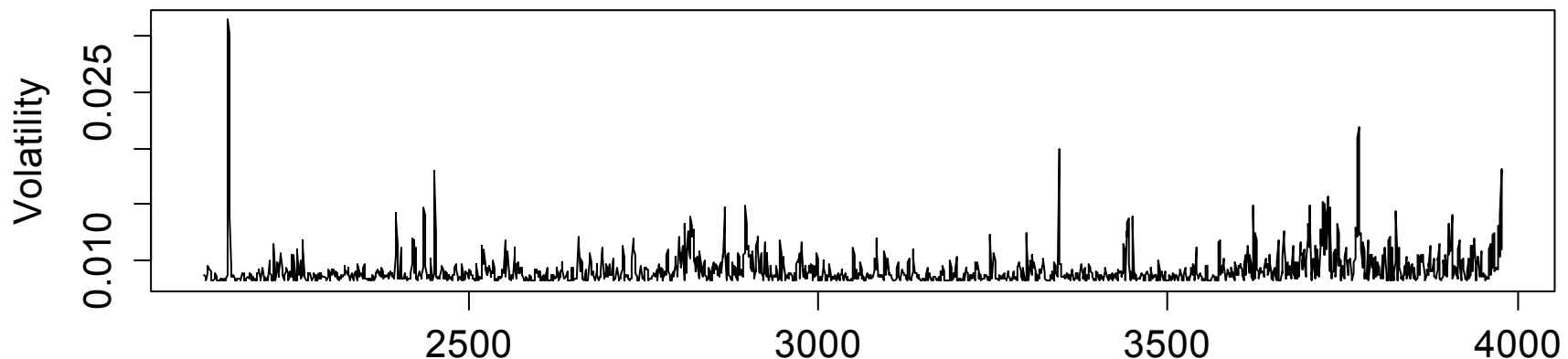
SS 2016 – ARIMA, SARIMA & GARCH

Showing the ARCH(2) Fit

SMI Log-Returns



Estimated Volatility from ARCH(2)

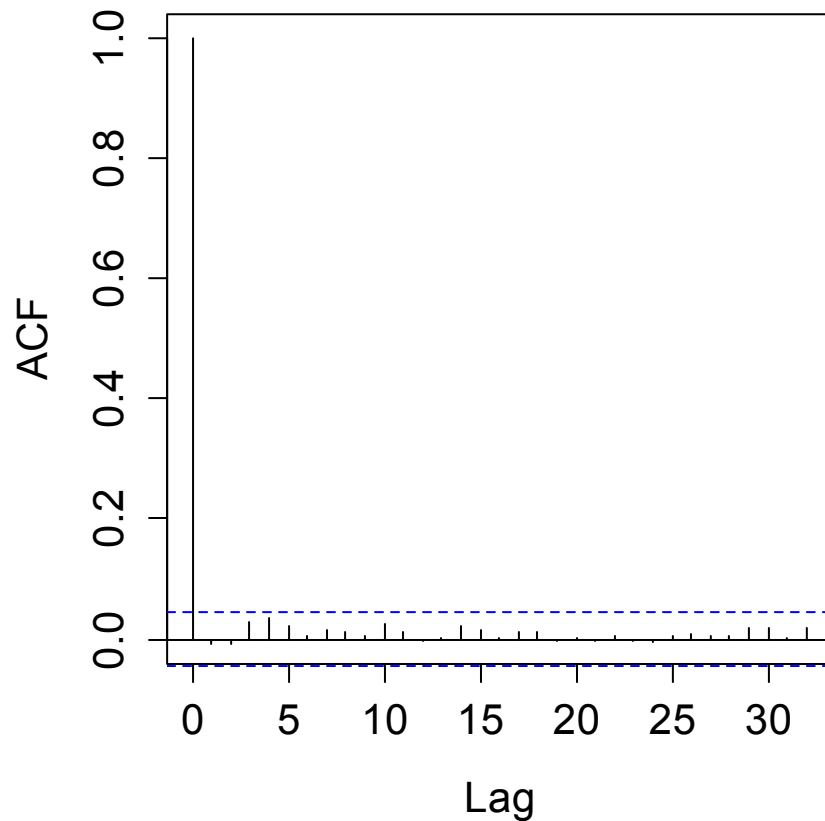


Applied Time Series Analysis

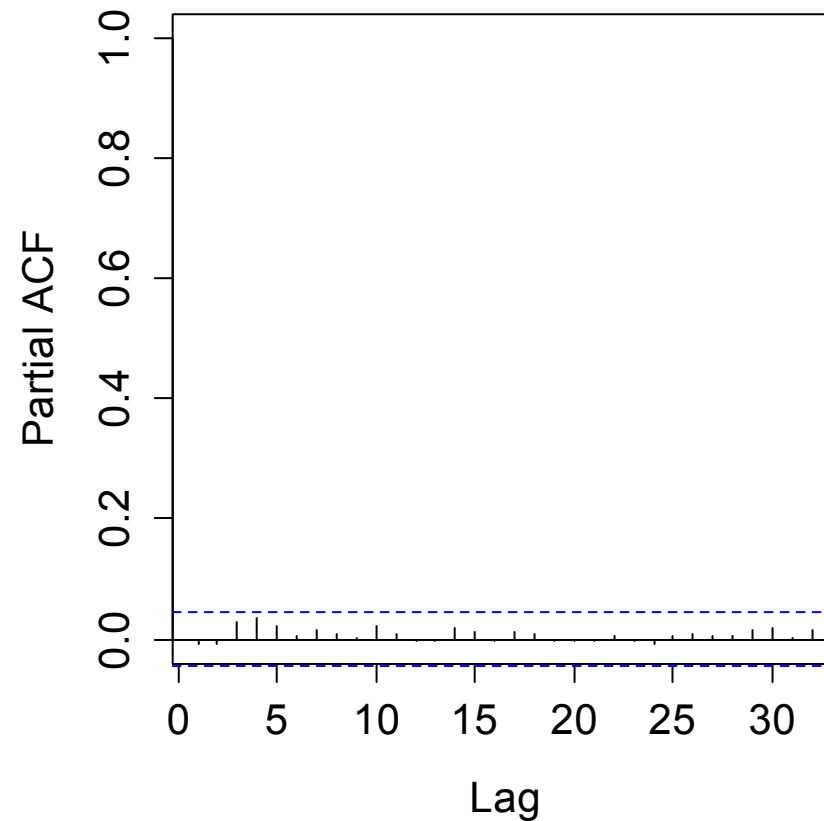
SS 2016 – ARIMA, SARIMA & GARCH

Residual Analysis for the ARCH(2)

ACF of Squared Residuals



PACF of Squared Residuals



Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting with Time Series

Goal: Point predictions for future observations with a measure of uncertainty, i.e. a 95% prediction interval.

Note:

- will be based on a stochastic model
- builds on the dependency structure and past data
- is an extrapolation, thus to take with a grain of salt
- similar to driving a car by using the rear window mirror



Applied Time Series Analysis

SS 2016 – Forecasting

Sources of Uncertainty in Forecasting

There are 4 principal sources of uncertainty:

- 1) Does the data generating model from the past also apply in the future? Or are there any breaks?
 - 2) Is the ARMA(p,q)-model we fitted to the data correctly chosen? What is the “true” order?
 - 3) Are the parameters $\alpha_1, \dots, \alpha_p, \sigma_E^2$ and m accurately estimated? How much do they differ from the “truth”?
 - 4) The stochastic variability coming from the innovation E_t
- **we will here restrict to short-term forecasting!**

Applied Time Series Analysis

SS 2016 – Forecasting

How to Forecast?

Probabilistic principle for deriving point forecasts:

$$\hat{X}_{n+k;1:n} = E \left[X_{n+k} \mid X_1, \dots, X_n \right]$$

→ The point forecast will be based on the conditional mean.

Probabilistic principle for deriving prediction intervals:

$$\sigma_{\hat{X}_{n+k;1:n}}^2 = \text{Var} \left(X_{n+k} \mid X_1, \dots, X_n \right)$$

→ An (approximate) 95% prediction interval will be obtained via:

$$\hat{X}_{n+k;1:n} \pm 1.96 \cdot \hat{\sigma}_{\hat{X}_{n+k;1:n}}$$

Applied Time Series Analysis

SS 2016 – Forecasting

How to Apply the Principles?

- The principles provide a generic setup, but are only useful and practicable under additional assumptions and have to be operationalized for every time series model/process.
 - For stationary AR(1)-processes with normally distributed innovations, we can apply the generic principles with relative ease and derive formulae for the point forecast and the prediction interval.
- **see blackboard for the derivation in the AR(1) case...**

Applied Time Series Analysis

SS 2016 – Forecasting

AR(1): 1-Step Forecast Summary

The 1-step forecast for a mean-zero AR(1) process is:

$$\hat{X}_{n+1;1:n} = \alpha_1 x_n$$

with a 95% prognosis interval given by

$$\alpha_1 x_n \pm 1.96 \cdot \sigma_E$$

Notice that in practice, we need to plug-in the estimated parameters $\hat{\alpha}_1, \hat{\sigma}_E$ plus potentially the global mean \hat{m} that was subtracted first. This adds additional uncertainty to the forecast which is not accounted for by the 95% prognosis interval. Its true coverage hence is probably less than 95%.

Applied Time Series Analysis

SS 2016 – Forecasting

AR(1): Simulation Study

We have seen that the 95% prognosis interval does not account for the uncertainty that is introduced by parameter estimation. A simulation study yields some insight. **Repeat 10'000x:**

Simulate from an AR(1) process with $\alpha_1 = 0.5$ and length n . Estimate parameters $\hat{\alpha}_1, \hat{\sigma}_E$ by MLE and plug-in for producing a 1-step forecast with 95% prognosis interval.

Finally, it was checked whether the prognosis interval contained the true next value of the time series process. The empirically obtained confidence levels were:

n=20	n=50	n=100	n=200
91.01%	93.18%	94.48%	94.73%

Applied Time Series Analysis

SS 2016 – Forecasting

AR(1): k-Step Forecast Summary

The k-step forecast for an AR(1) process is:

$$\hat{X}_{n+k;1:n} = \alpha_1^k x_n$$

with prognosis interval based on

$$\text{Var}(X_{n+k;1:n} | X_1, \dots, X_n) = \left(1 + \sum_{j=1}^{k-1} \alpha_1^{2j} \right) \cdot \sigma_E^2$$

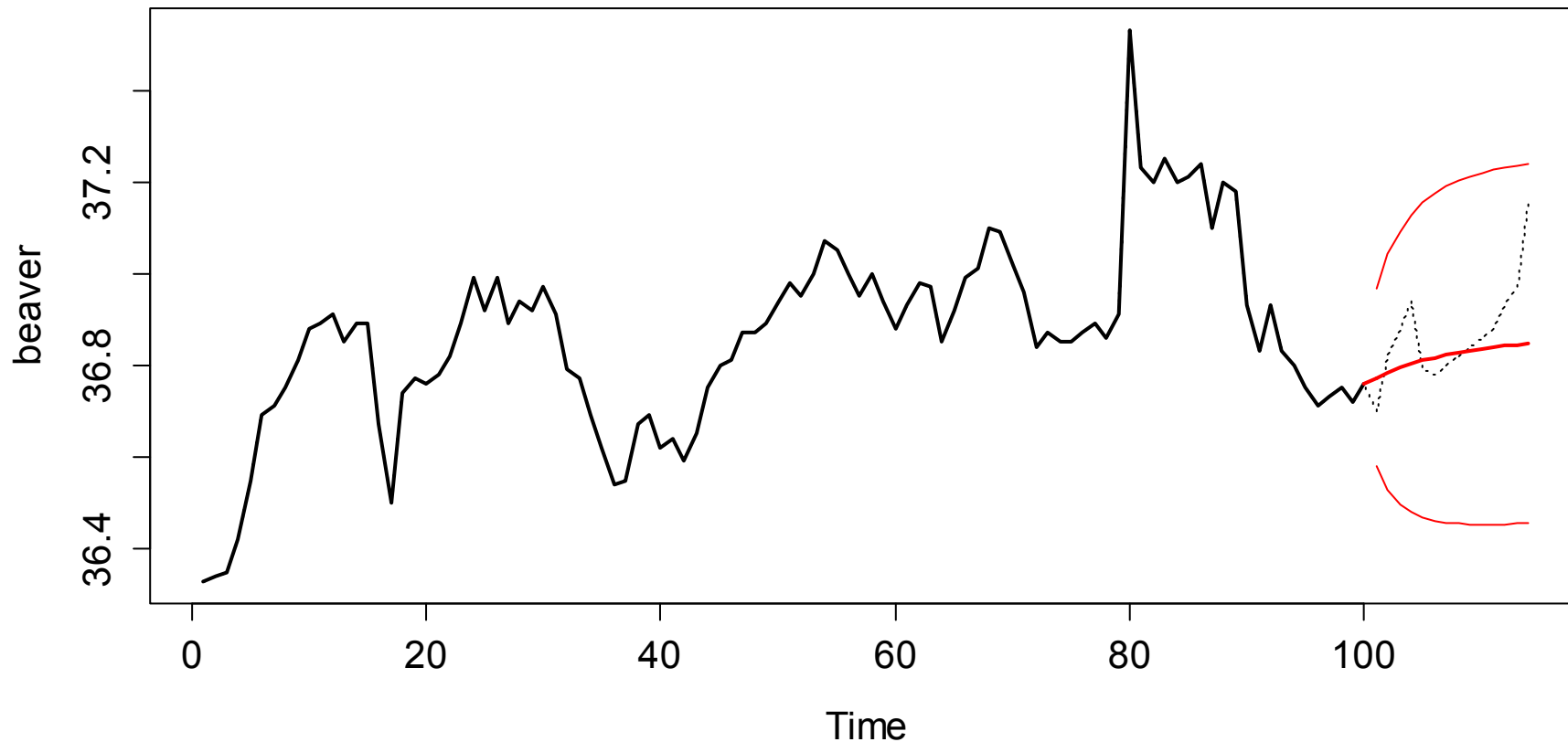
It is important to note that for increasing forecasting horizon $k \rightarrow \infty$, the point forecast $\hat{X}_{n+k;1:n}$ converges to 0 and the conditional variance to $\sigma_E^2 / (1 - \alpha_1^2) = \sigma_X^2$, reflecting the general parameters of the process.

Applied Time Series Analysis

SS 2016 – Forecasting

AR(1): Forecasting the Beaver Data

Beaver Data: 14-Step Prediction Based on AR(1)



Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting AR(p)

The principles are the same, forecast and prognosis interval are:

$$E[X_{n+k} | X_1, \dots, X_n] \text{ and } Var(X_{n+k} | X_1, \dots, X_n)$$

The computations are a bit more complicated, but do not yield major further insight. We are thus doing without and present:

$$\text{1-step-forecast: } \hat{X}_{n+1;1:n} = \alpha_1 x_n + \dots + \alpha_p x_{n+1-p}$$

$$\text{k-step-forecast: } \hat{X}_{n+k;1:n} = \alpha_1 \hat{X}_{n+k-1;1:n} + \dots + \alpha_p \hat{X}_{n+k-p;1:n}$$

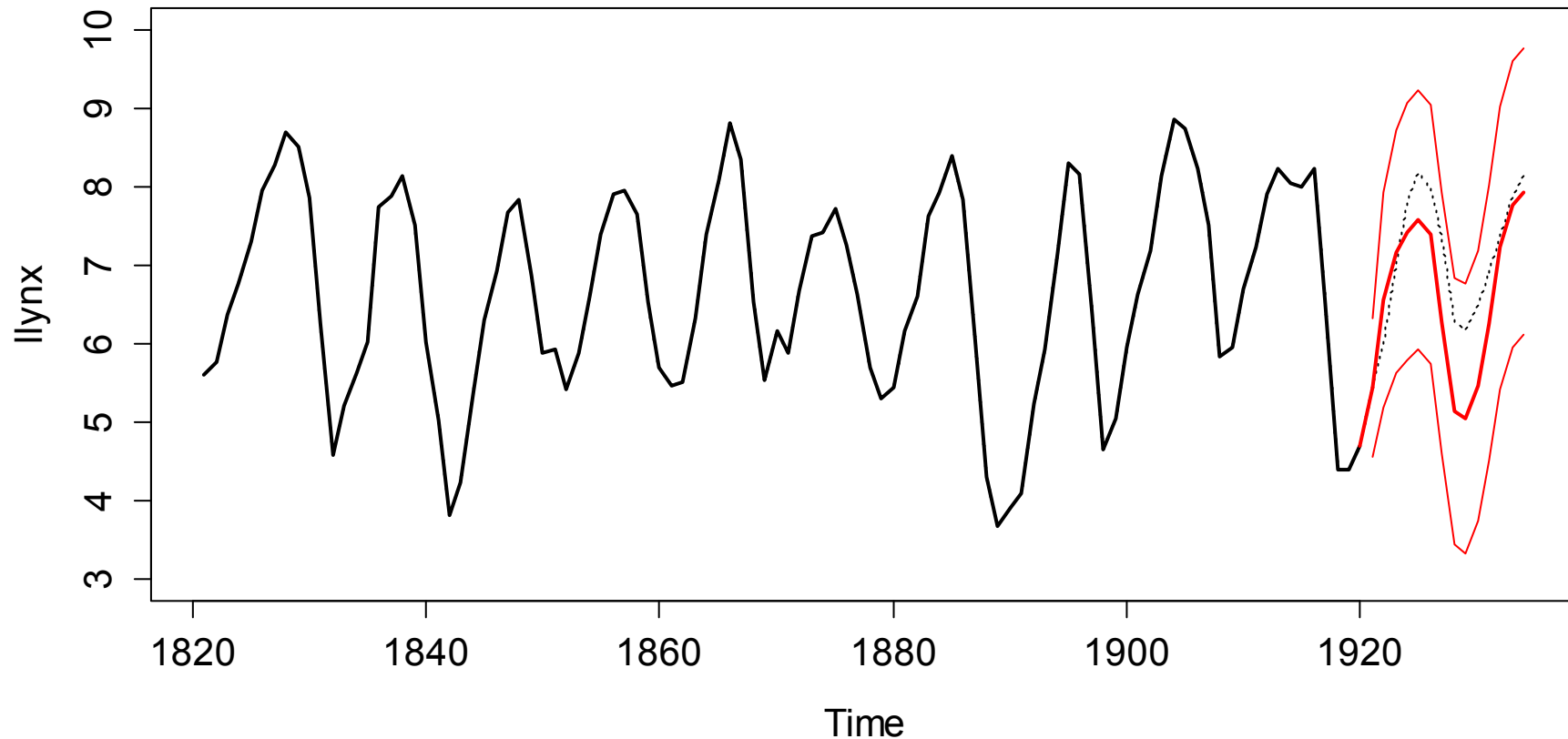
If an observed value for \hat{X}_{n+k-t} is available, we plug it in. Else, the forecasted value is used. Hence, the forecasts for horizons $k > 1$ are determined in a recursive manner.

Applied Time Series Analysis

SS 2016 – Forecasting

AR(p): Forecasting the Lynx Data

Logged Lynx Data: 14-Step Prediction Based on AR(11)



Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting AR(p): Remarks

- AR(p) processes have a Markov property. Given the model parameters, we only need to know the last p observations in the series to compute the forecast & prognosis interval.
- The prognosis intervals are only valid on a pointwise basis, and they generally only cover the uncertainty coming from innovation, but not from other sources. Hence, they are generally too small.
- Retaining the final part of the series, and predicting it with several competing models may give hints which one yields the best forecasts. This can be an alternative approach for choosing the model order p .

Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting MA(q) & ARMA(p,q)

- Point and interval forecasts will again, as for AR(p), be derived from the conditional mean and variance.
 - The derivation is more complicated, as it involves the latent innovations terms $e_n, e_{n-1}, e_{n-2}, \dots$.
 - If invertibility is required, the issues can be solved. However, the forecast requires knowledge about all past instances of a time series.
 - In practice, initial values need to be chosen.
- **See blackboard for the derivation in the MA(1) case...**

Applied Time Series Analysis

SS 2016 – Forecasting

MA(1) Forecasting: Summary

- We have seen that for any MA(1)-process, the k step forecast for all $k > 1$ is trivial and equal to 0.
- In case of $k = 1$, we obtain for the MA(1)-forecast:

$$\hat{X}_{n+1;1:n} = \beta_1 \cdot E[E_n | X_1, \dots, X_n]$$

This conditional expectation is (too) difficult to compute, but we can get out by conditioning on the infinite past:

$$e_n := E[E_n | X_{-\infty}, \dots, X_n]$$

- We then express the MA(1) as an AR(∞) and obtain:

$$\hat{X}_{n+1;1:n} = \sum_{j=0}^{n-1} \hat{\beta}_1 (-\hat{\beta}_1)^j x_{n-j} = \sum_{j=0}^{n-1} \hat{\psi}_j x_{n-j}$$

Applied Time Series Analysis

SS 2016 – Forecasting

MA(q) Forecasting

- With MA(q) models, all forecasts for horizons $k > q$ will be trivial and equal to zero. This is not the case for $k \leq q$.
- We encounter the same difficulties as with MA(1) processes. By conditioning on the infinite past, rewriting the MA(q) as an AR(∞) and the choice of initial values for times $t \leq 0$, the forecasts can be computed.
- We do without giving precise details about the involved formulae here, but refer to the general results for ARMA(p,q), from where the solution for pure MA(q) can be obtained.
- In R, function `predict()` implements all this!!!

Applied Time Series Analysis

SS 2016 – Forecasting

ARMA(p,q) Forecasting

We here face identical problems for forecasting as with MA(q):

$$\hat{X}_{n+k;1:n} = \sum_{i=1}^p \alpha_i x_{n+k-i}^* + \sum_{j=1}^q \beta_j e_{n+k-j}^*$$

For practical implementation, we use the following scheme:

$$x_t^* = \begin{cases} x_t, & \text{if } t \leq n \\ \hat{X}_{t;1:n}, & \text{if } t > n \end{cases}$$

$$e_t^* = \begin{cases} e_t, & \text{if } 0 < t \leq n \\ 0, & \text{if } t > n \end{cases}$$

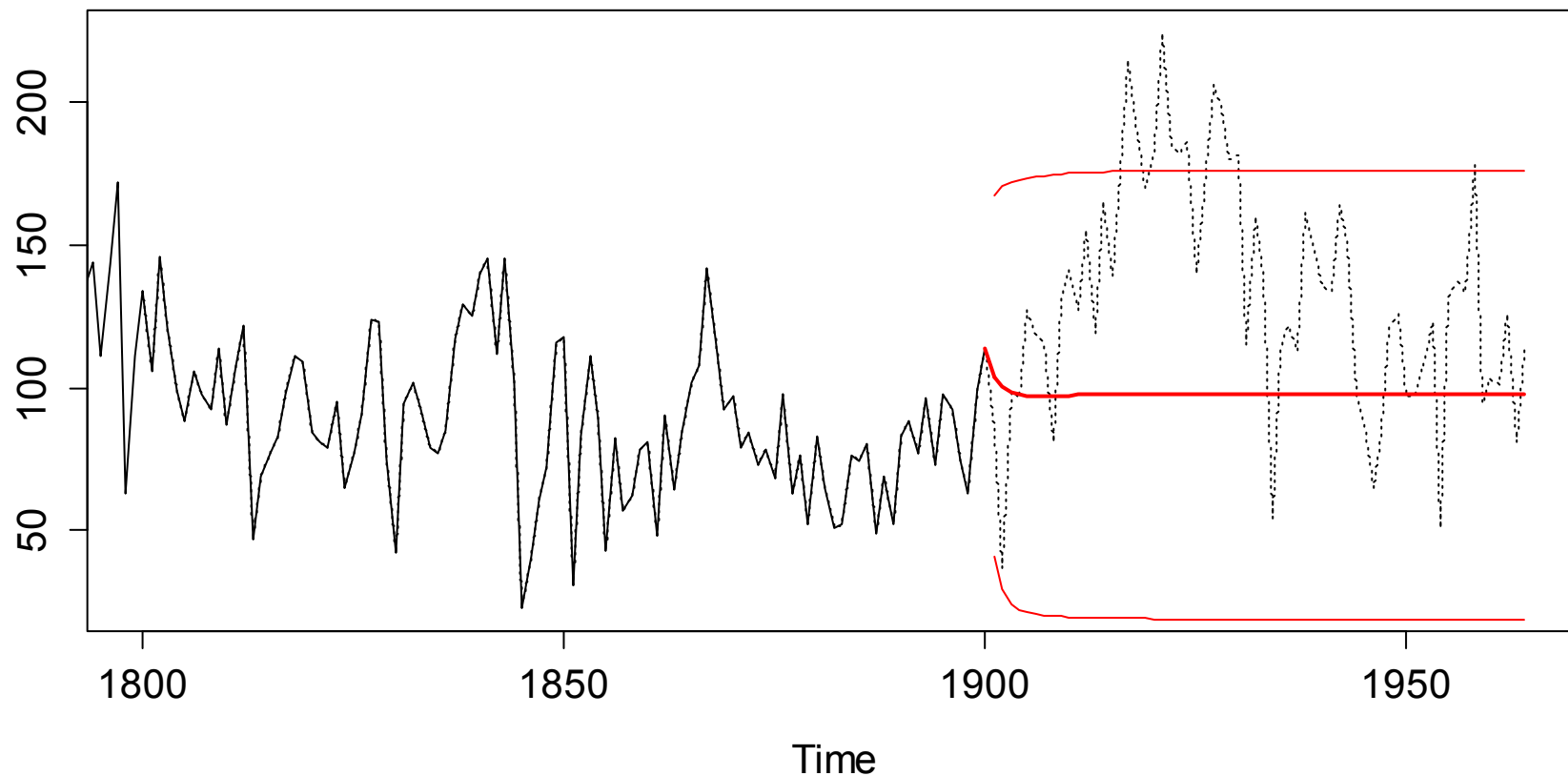
In any case: $\hat{X}_{n+k;1:n} = \sum_{i=0}^{n-1} \hat{\psi}_i x_{n-i}$, implemented in `predict()`

Applied Time Series Analysis

SS 2016 – Forecasting

ARMA(4,1) Forecasting: Example

Douglas Fir Data: 64-Step Prediction Based on ARMA(4,1)



Applied Time Series Analysis

SS 2016 – Forecasting

ARMA(4,1) Forecasting: Code

```
> train <- window(douglasfir, start=1107, end=1900)
> fit    <- arima(train, order=c(4,0,1))
> fc     <- predict(fit, n.ahead=64)
> plot(window(douglasfir, 1800, 1964), lty=3, ylab="")
> lines(train, lwd=1)
> lines(fc$pred, lwd=2, col="red")
> lines(fc$pred+fc$se*1.96, col="red")
> lines(fc$pred-fc$se*1.96, col="red")
> title("Douglas Fir Data: 64-Step Prediction..")
```

We notice that the forecast tracks the true values only poorly. Plus, 12 out of 64 forecasts (18.75%) exceed the prognosis interval which by construction should only happen in 5% of the cases. What does this results mean?

Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting with Trend & Season

Time series with a trend and/or seasonal effect can either be predicted after decomposing or with exponential smoothing. It is also very easy and quick to predict from a SARIMA model.

- The ARIMA/SARIMA model is fitted in R as usual. Then, we can simply employ the `predict()` command and obtain the forecast plus a prediction interval.
- Technically, the forecast comes from the stationary ARMA model that is obtained after differencing the series.
- Finally, these forecasts need to be integrated again. This procedure has a bit the touch of a **black box approach**.

Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting for ARIMA Models

We assume that X_t is an ARIMA(p,1,q) series, so after lag 1 differencing, we have $Y_t = X_t - X_{t-1}$ which is an ARMA(p,q).

- Anchor: $\hat{X}_{n+1;1:n} = \hat{Y}_{n+1;1:n} + x_n$

The longer horizon forecasts with $k > 1$ are obtained from:

$$\hat{X}_{n+1;1:n} = \hat{Y}_{n+1;1:n} + x_n$$

$$\hat{X}_{n+2;1:n} = \hat{Y}_{n+2;1:n} + \hat{X}_{n+1;1:n} = x_n + \hat{Y}_{n+1;1:n} + \hat{Y}_{n+2;1:n}$$

⋮

$$\hat{X}_{n+k;1:n} = x_n + \hat{Y}_{n+1;1:n} + \dots + \hat{Y}_{n+k;1:n}$$

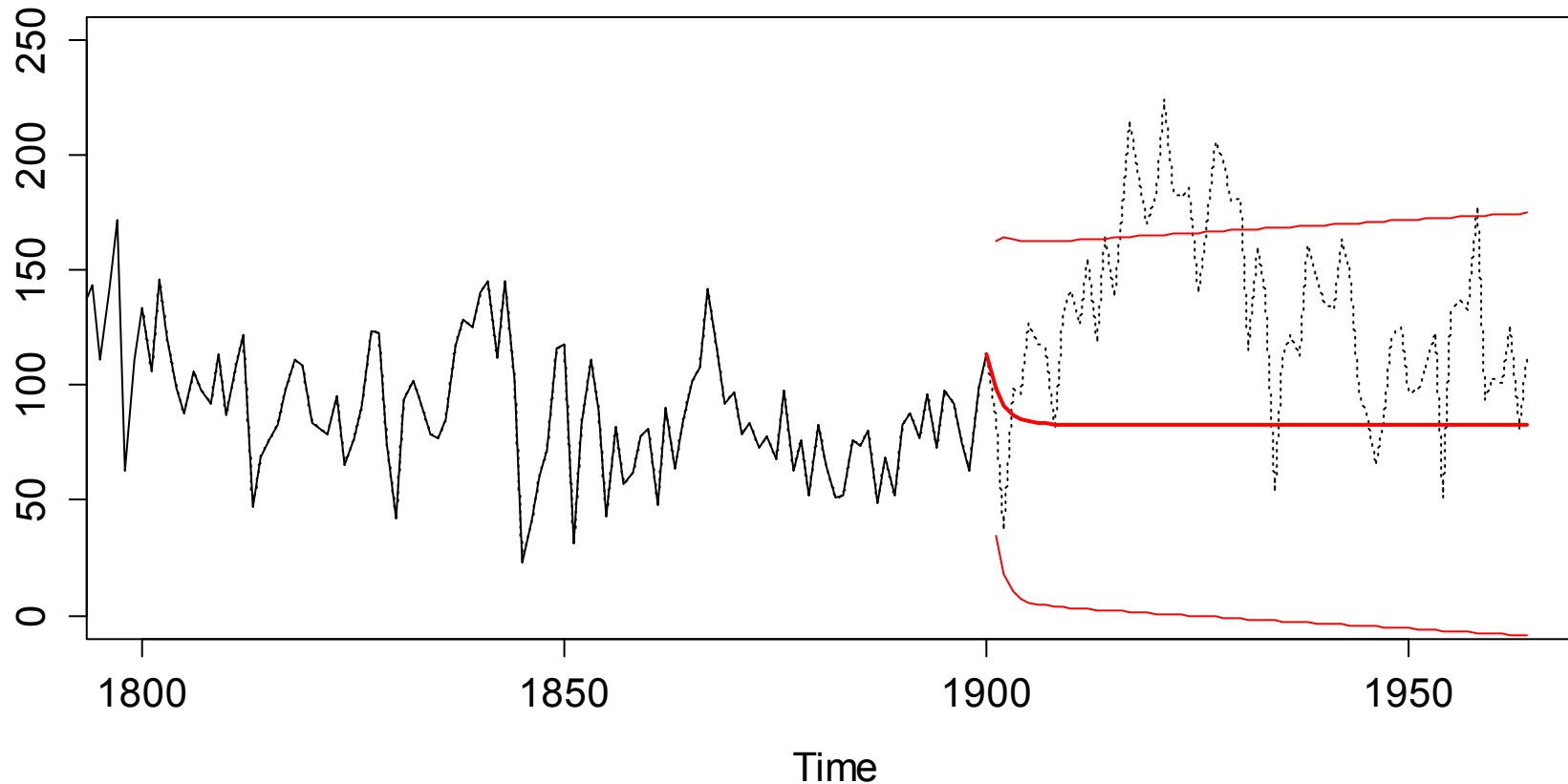
As we can see, this is the cumulative sum of all terms!

Applied Time Series Analysis

SS 2016 – Forecasting

ARIMA(1,1,1) Forecasting: Example

Douglas Fir Data: 64-Step Prediction Based on ARIMA(1,1,1)

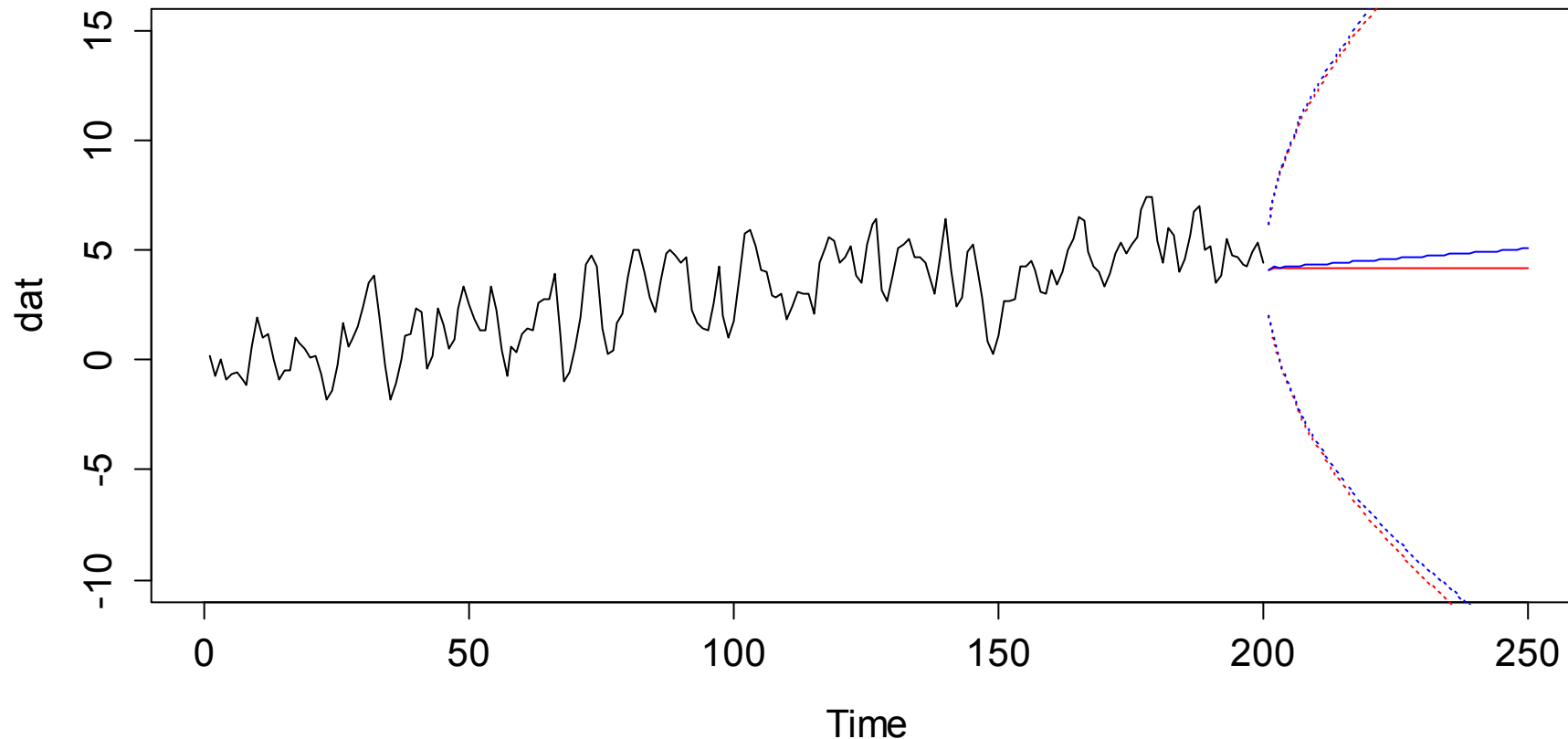


Applied Time Series Analysis

SS 2016 – Forecasting

Artificial Example where ARIMA Fails

ARIMA(1,1,1) Forecast for ARMA(1,1) with Linear Trend

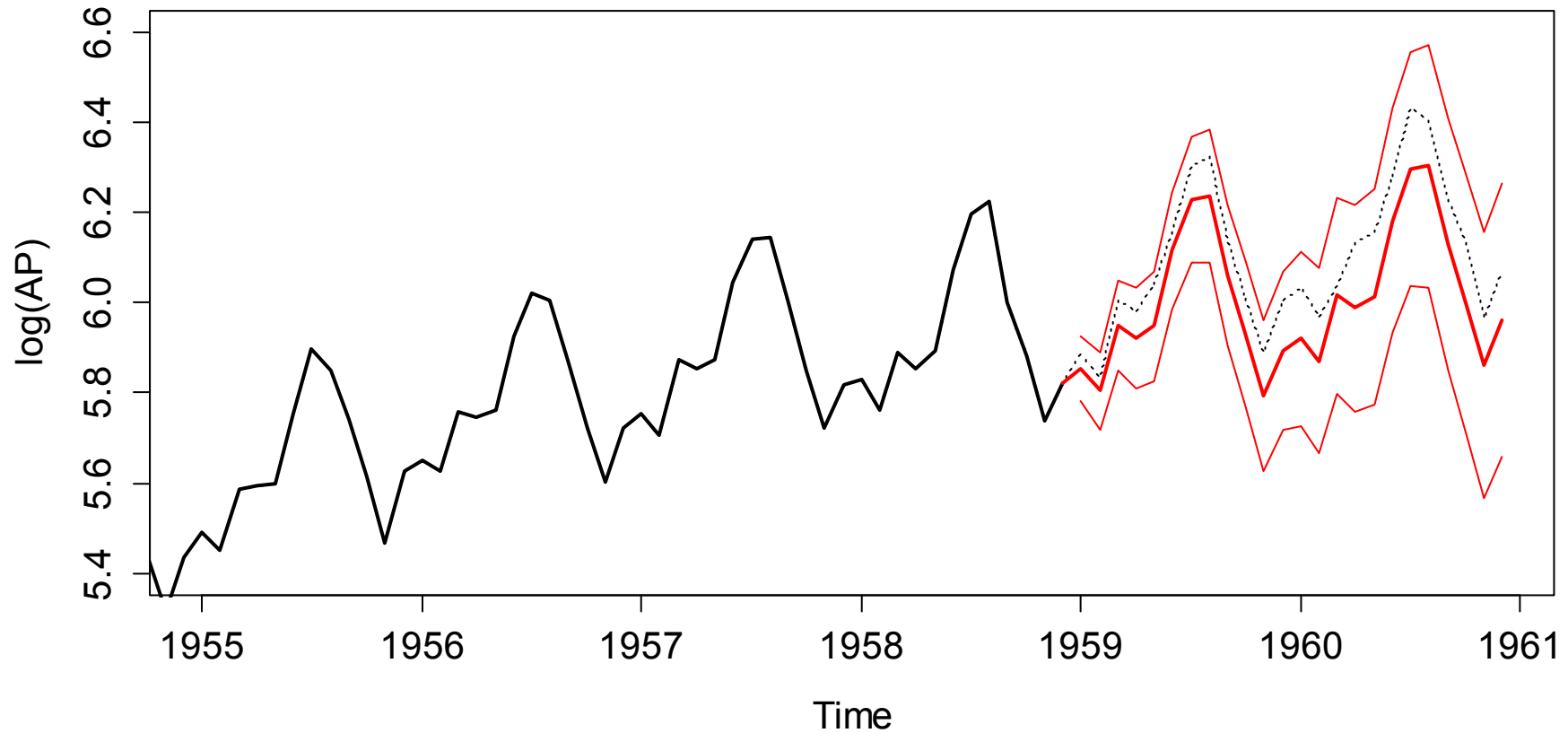


Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting with SARIMA: Example

Forecast of $\log(\text{AP})$ with SARIMA(0,1,1)(0,1,1)



Applied Time Series Analysis

SS 2016 – Forecasting

(S)ARIMA Forecasting: Summary

- When using a (S)ARIMA model for forecasting, we are able to deal with both seasonality and trend.
- As we can see, the prognosis intervals also cover the effect of trend and seasonality. They become (much) wider for longer forecasting horizons, different to what we observe with stationary series.
- There is no control about the trend forecast and as we have seen in the example of an ARMA(1,1) with linear trend, the forecast can be poor.

→ There is room for decomposition based forecasting!!!

Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series

The principle for forecasting time series that are decomposed into trend, seasonal effect and remainder is:

1) Stationary Remainder

Is usually modelled with an ARMA(p,q), so we can generate a time series forecast with the methodology from before.

2) Seasonal Effect

Is assumed as remaining “as is”, or “as it was last” (in the case of evolving seasonal effect) and extrapolated.

3) Trend

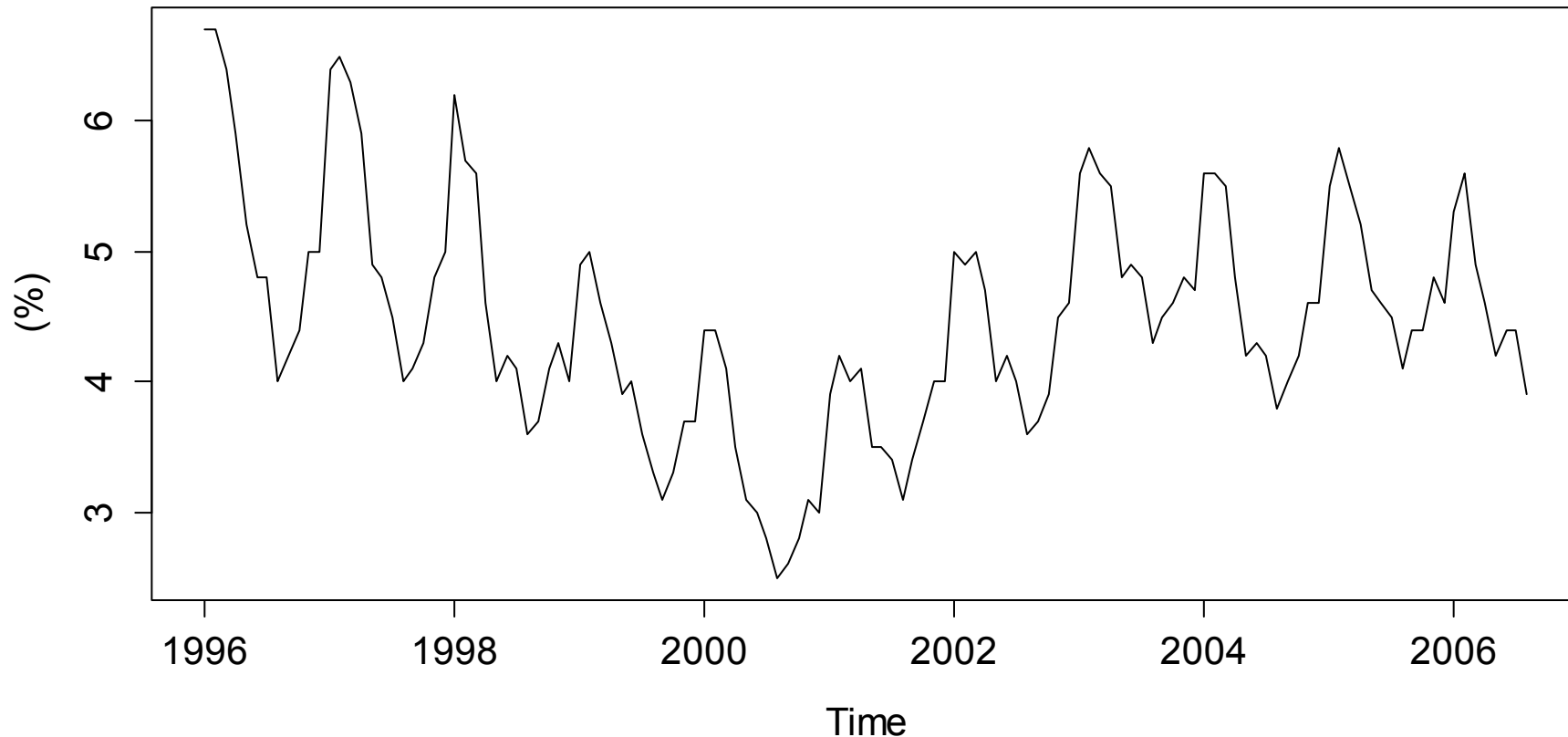
Is either extrapolated linearly, or sometimes even manually.

Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series: Example

Unemployment in Maine

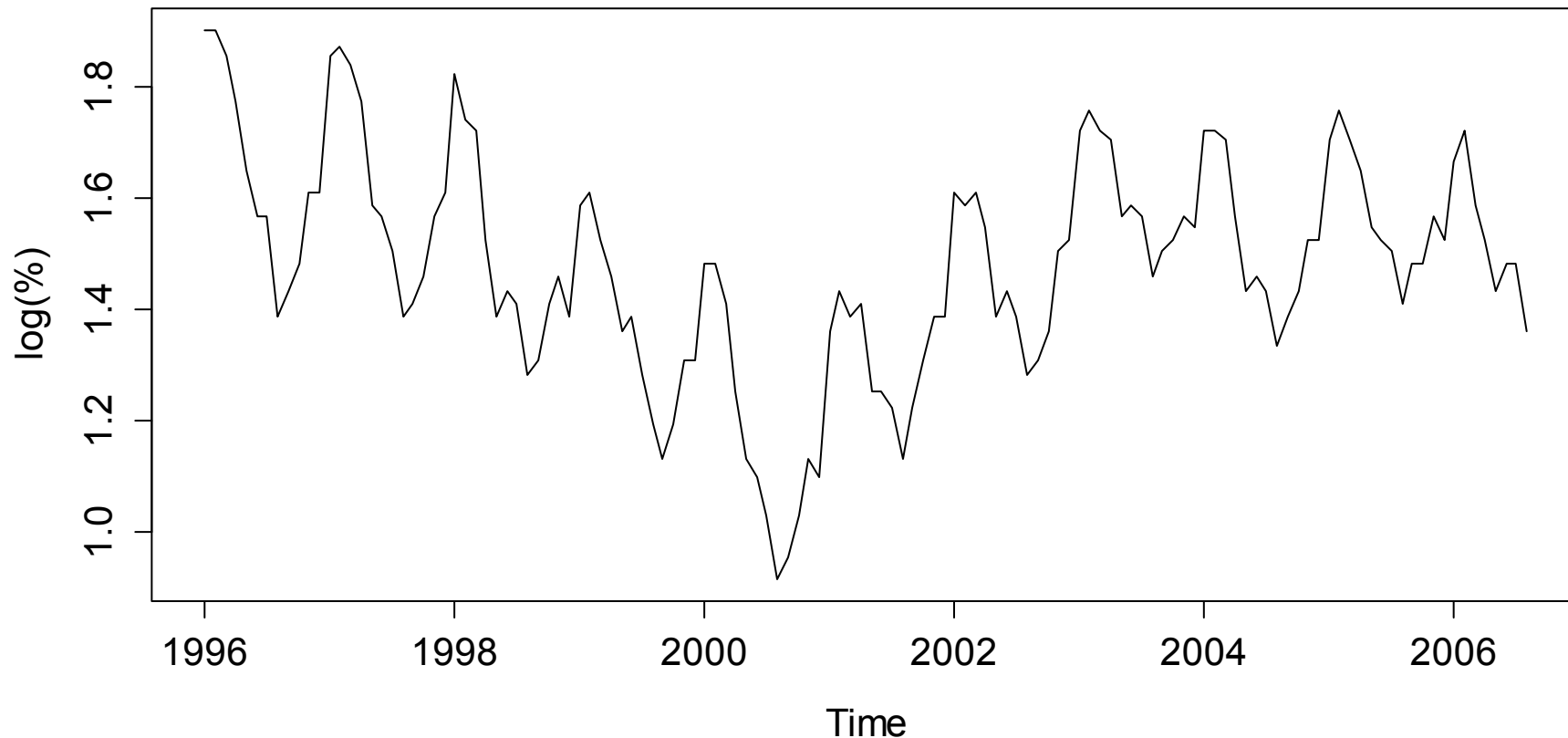


Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series: Example

Logged Unemployment in Maine

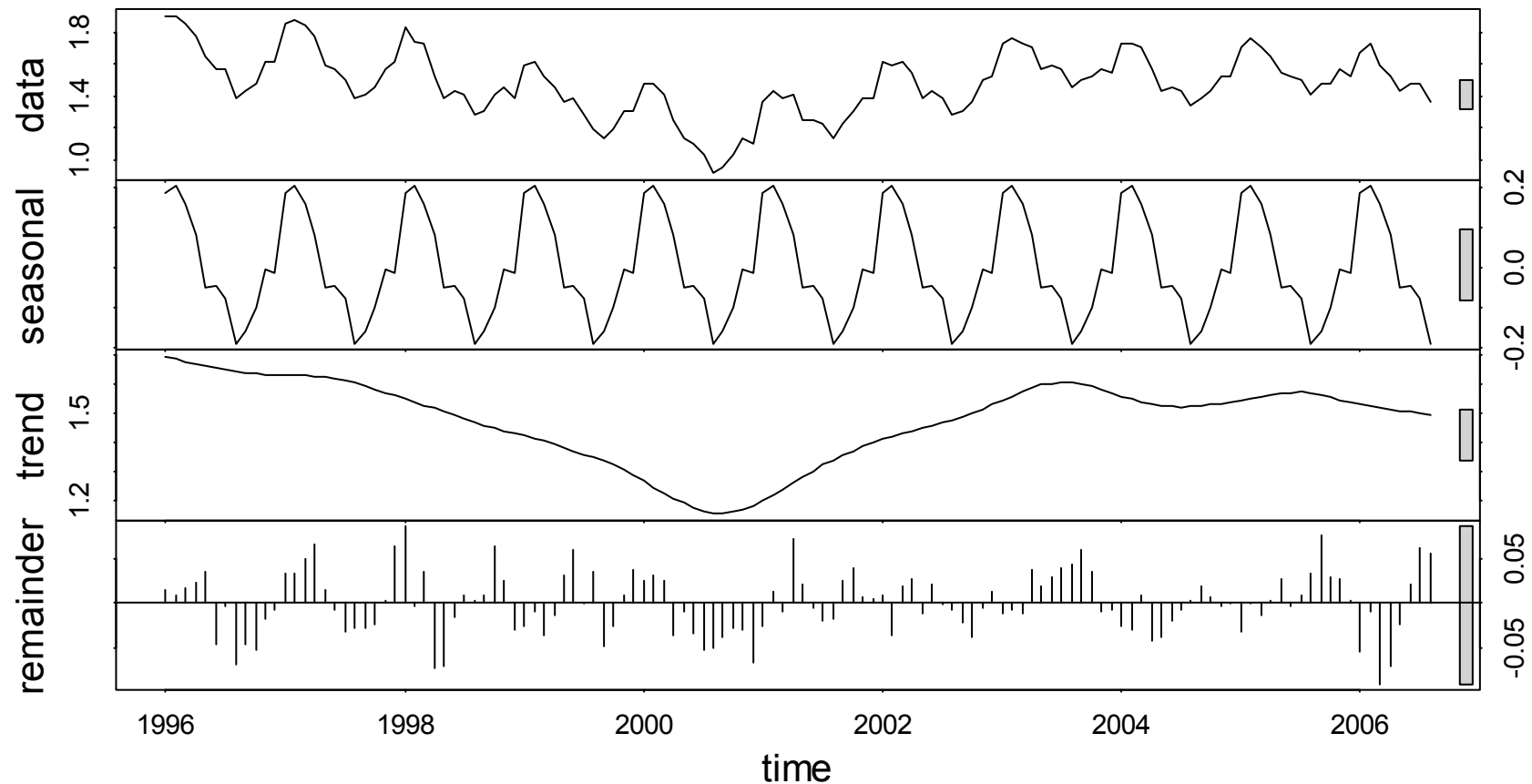


Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series: Example

STL-Decomposition of Logged Maine Unemployment Series

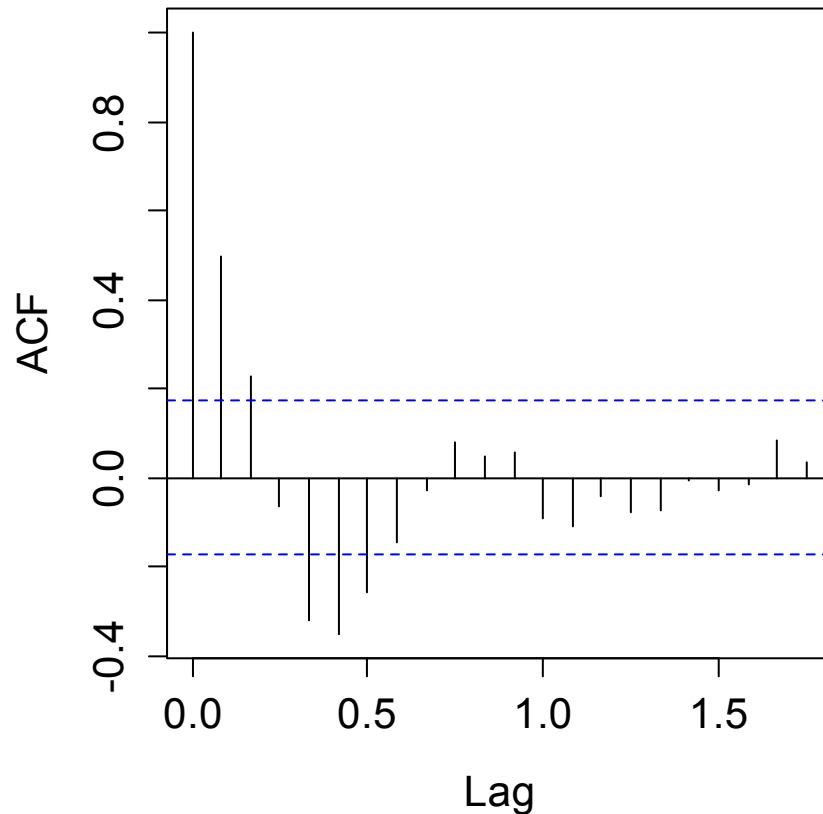


Applied Time Series Analysis

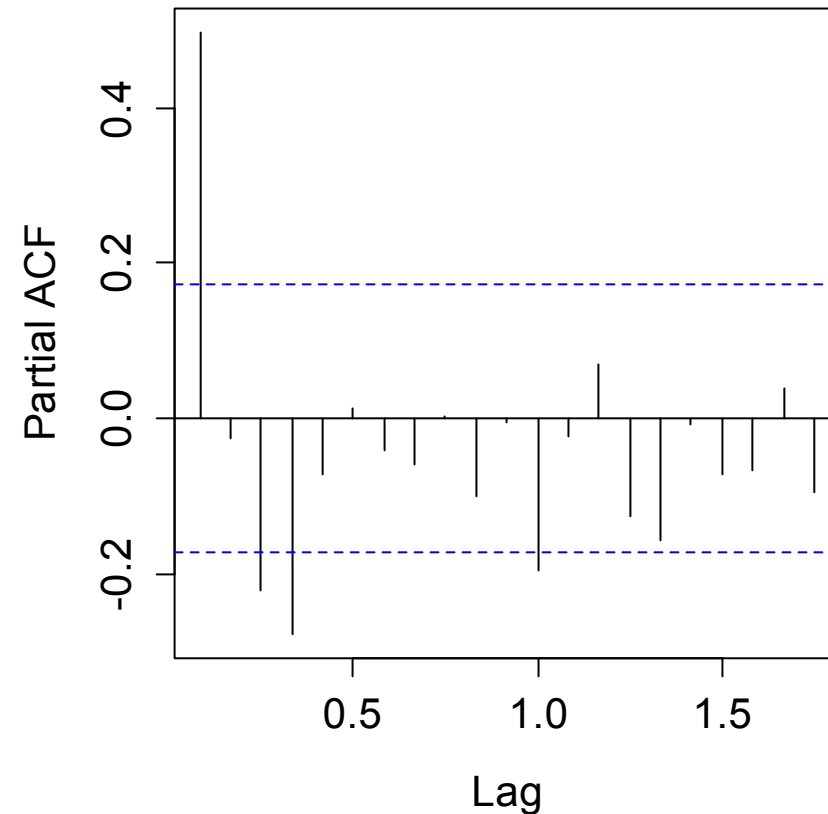
SS 2016 – Forecasting

Forecasting Decomposed Series: Example

ACF of Remainder Series



PACF of Remainder Series

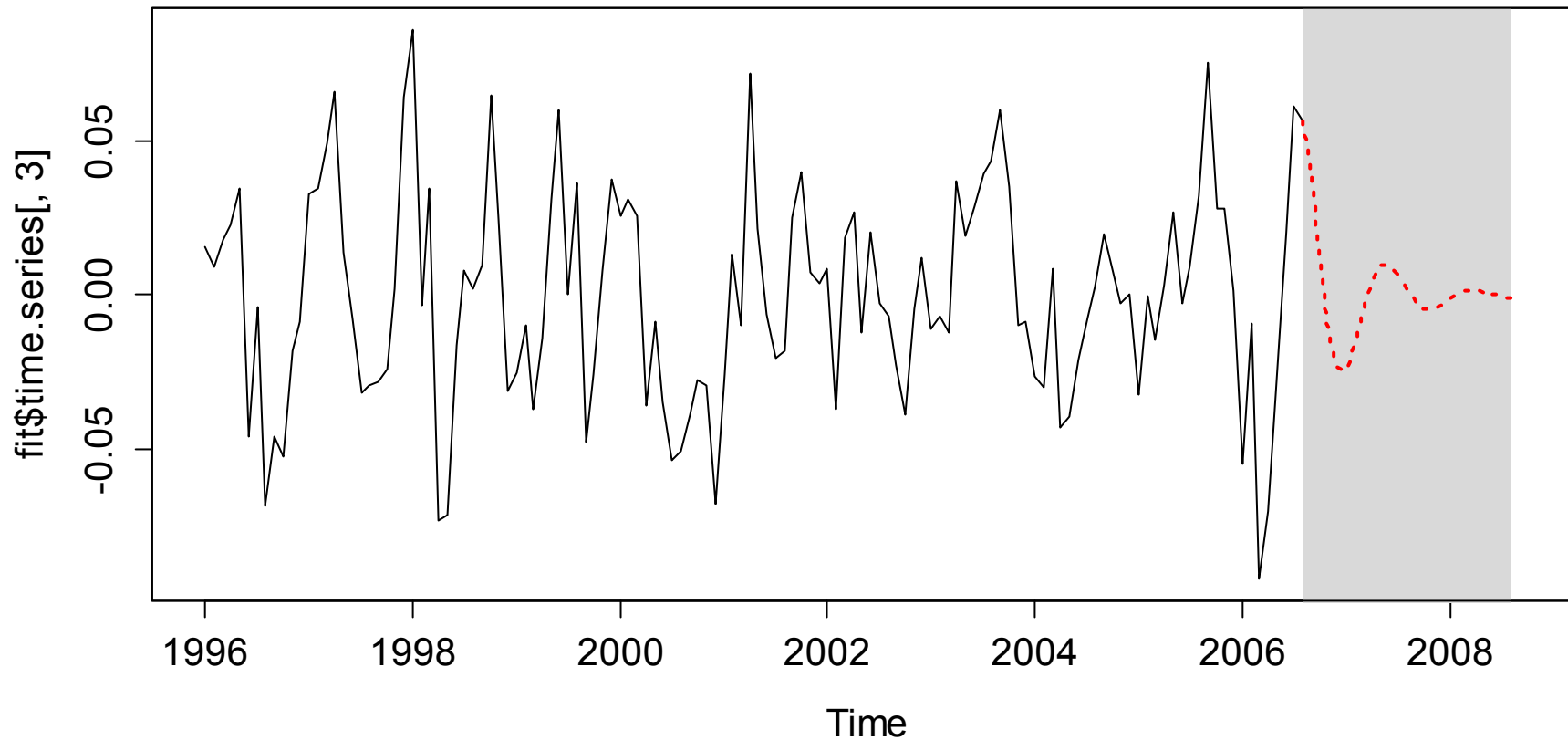


Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series: Example

AR(4) Forecast for Remainder Series

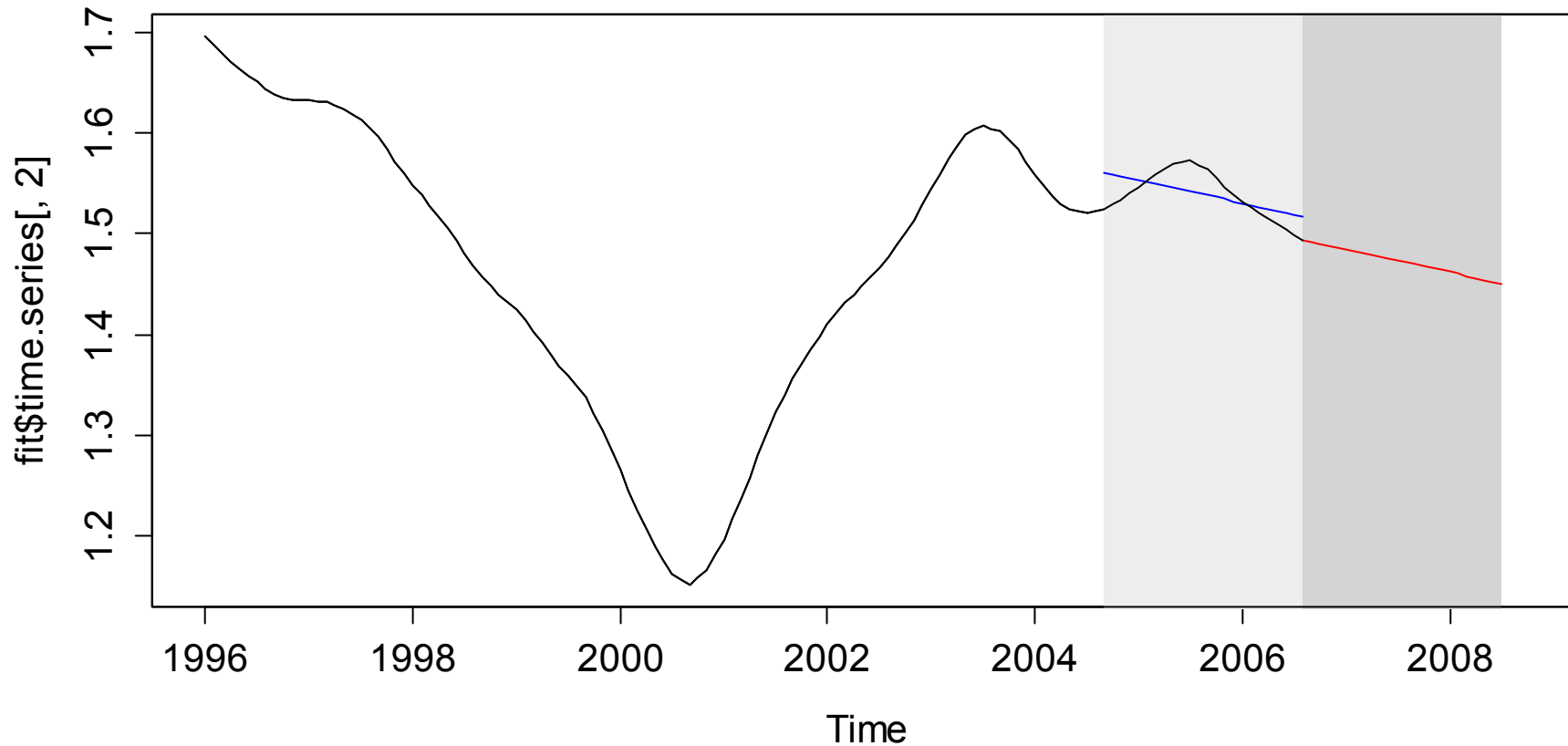


Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series: Example

Trend Forecast by Linear Extrapolation

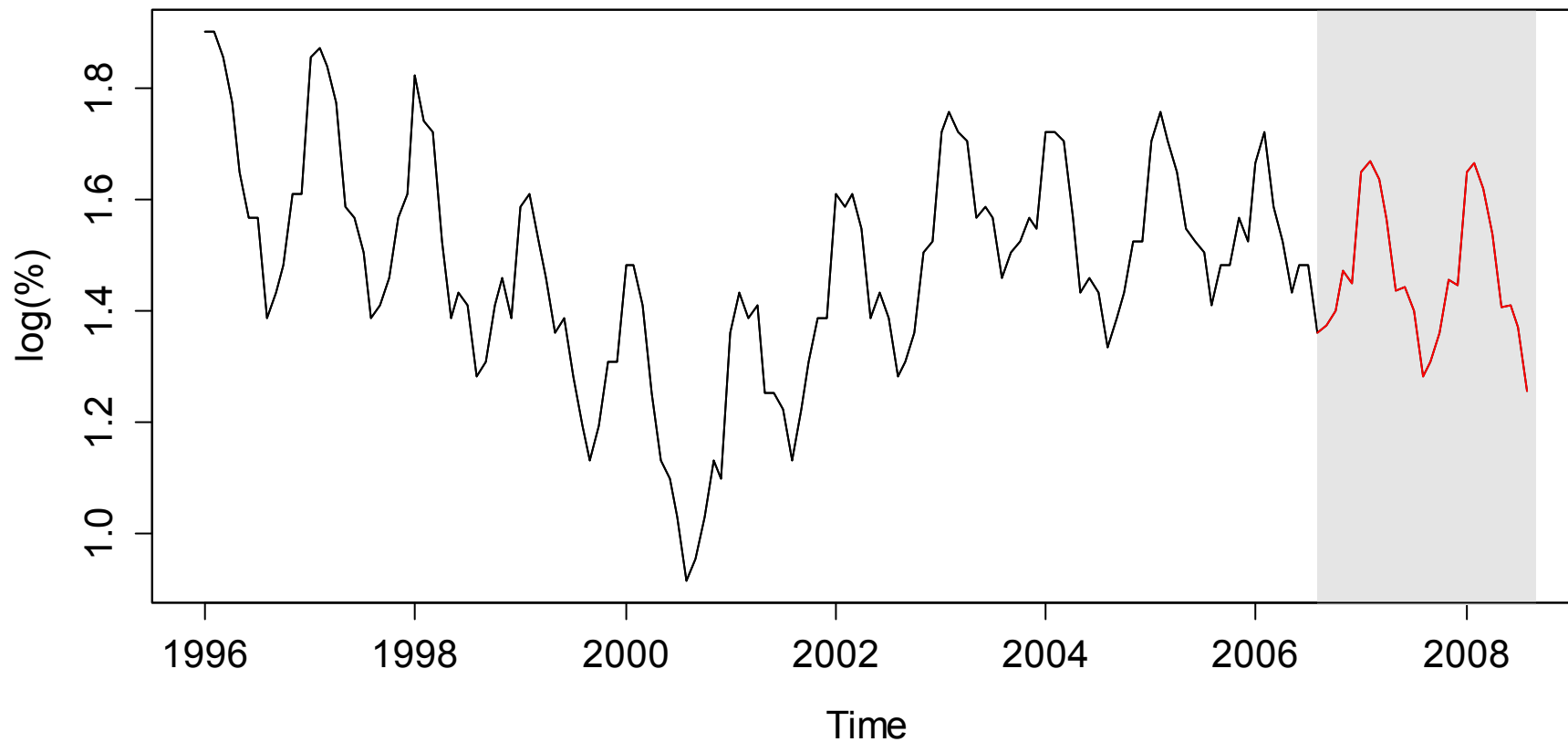


Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting Decomposed Series: Example

Forecast of Logged Unemployment in Maine



Applied Time Series Analysis

SS 2016 – Forecasting

Simple Exponential Smoothing

This is a quick approach for estimating the current level of a time series, as well as for forecasting future values. It works for any stationary time series without a trend and season.

The **simple, intuitive idea behind** is:

$$\hat{X}_{n+1,1:n} = \sum_{i=0}^{n-1} w_i x_{n-i} \text{ where } w_0 \geq w_1 \geq w_2 \geq \dots \geq 0 \text{ and } \sum_{i=0}^{n-1} w_i = 1$$

The weights are often chosen to be exponentially decaying, two examples with different parameters are on the next slide. However, there is also a deeper mathematical notion of ExpSmo.

→ **See the blackboard for the derivation...**

Applied Time Series Analysis

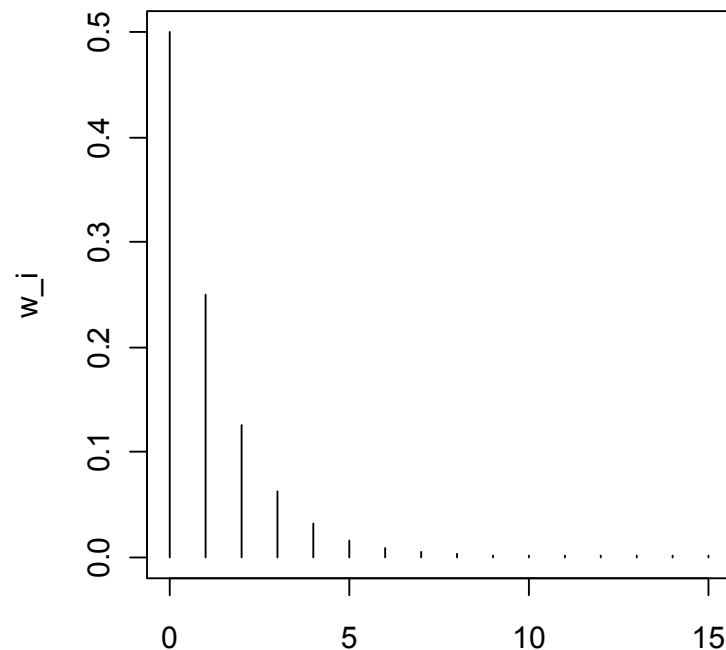
SS 2016 – Forecasting

Choice of Weights

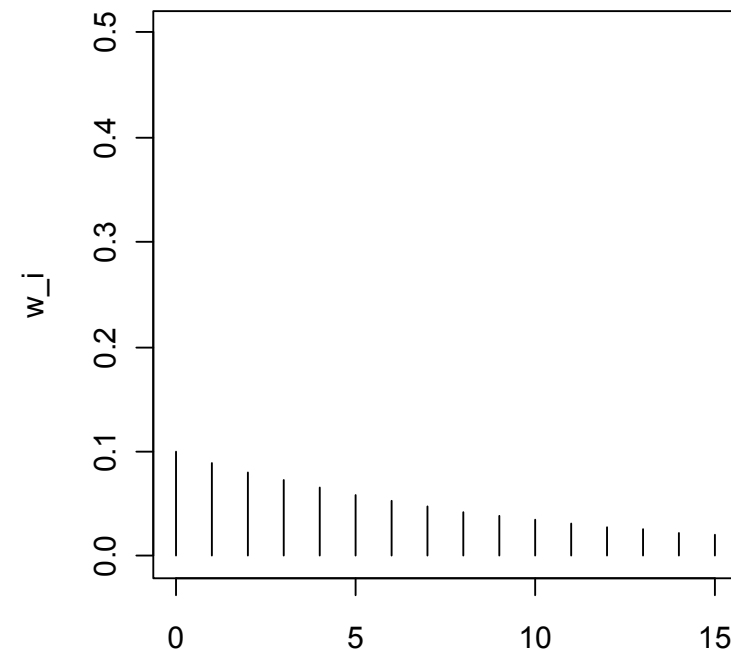
An usual choice are exponentially decaying weights:

$$w_i = \alpha(1-\alpha)^i \text{ where } \alpha \in (0,1)$$

a=0.5



a=0.1



Applied Time Series Analysis

SS 2016 – Forecasting

Simple Exponential Smoothing: Summary

What is it?

- A method for estimating and forecasting the conditional mean

Basic notion: $X_t = \mu_t + E_t$

- μ_t is the conditional expectation, which we try to estimate from the data. The estimate a_t is called level of the series.
- E_t is a completely random innovation term.

Estimation of the level: two notions exist...

- Weighted updating: $a_t = \alpha x_t + (1 - \alpha)a_{t-1}$
- Exponential smoothing: $a_t = \sum_{i=0}^{\infty} \alpha(1 - \alpha)^i x_{t-i}$

Applied Time Series Analysis

SS 2016 – Forecasting

Forecasting with Exponential Smoothing

The forecast, for any horizon $k > 0$ is:

$$\hat{X}_{n+k,1:n} = a_n$$

Hence, the forecast is given by the current level, and it is constant for all horizons k . However, it does depend on the choice of the smoothing parameter α . In R, a data-adaptive solution is available by minimizing SS1PE:

$$\text{1-step-prediction-error: } e_t = x_t - \hat{X}_{t;1:(t-1)} = x_t - a_{t-1}$$

$$\hat{\alpha} = \arg \min_{\alpha} \sum_{i=2}^n e_i^2$$

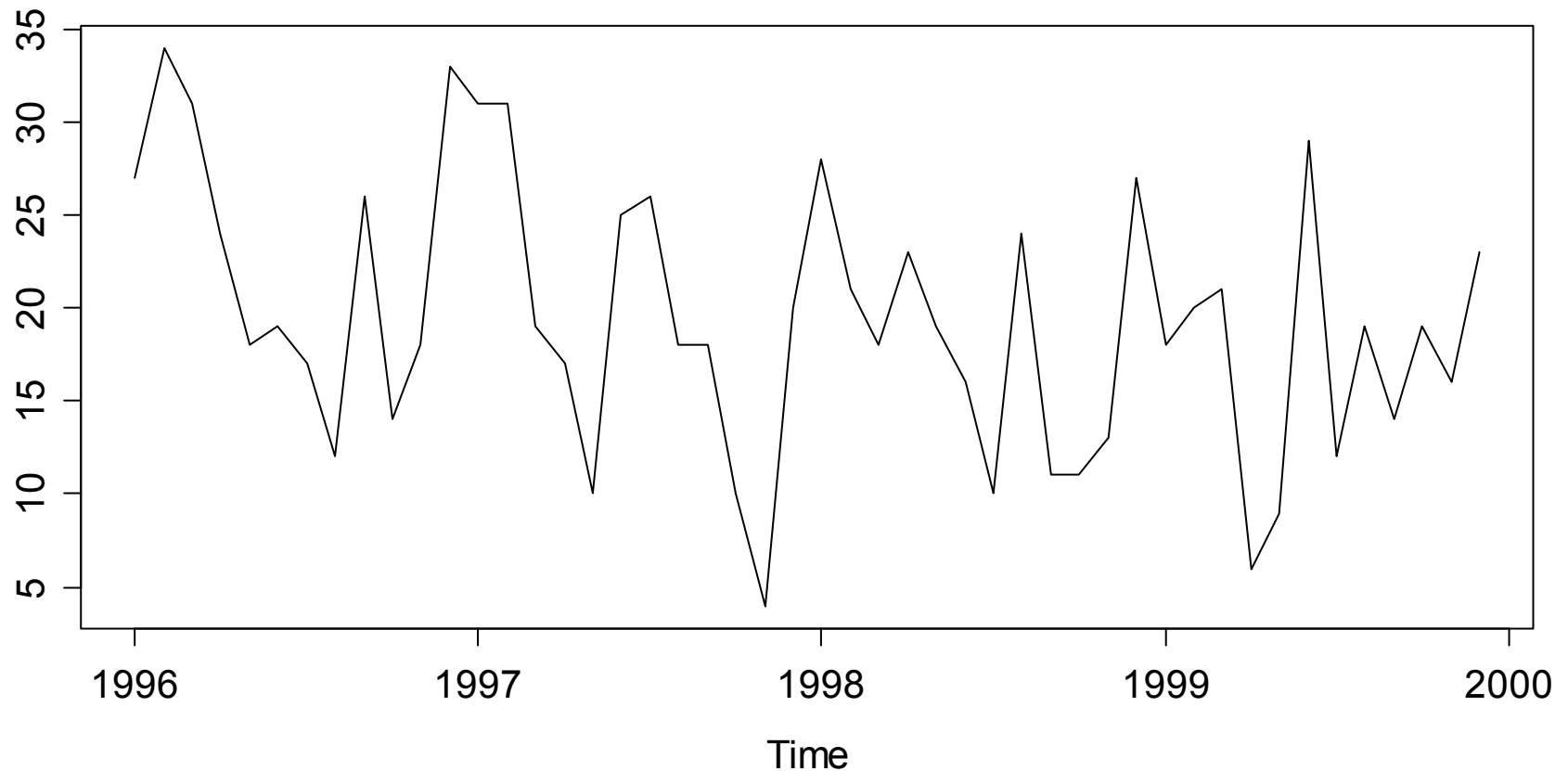
The solution needs to be found with numerical optimization.

Applied Time Series Analysis

SS 2016 – Forecasting

Exponential Smoothing: Example

Complaints to a Motorizing Organization



Applied Time Series Analysis

SS 2016 – Forecasting

Exponential Smoothing: Example

```
> fit <- HoltWinters(cmpl, beta=F, gamma=F)
```

```
Holt-Winters exponential smoothing without trend  
and without seasonal component.
```

```
Smoothing parameters:
```

```
alpha: 0.1429622
```

```
beta : FALSE
```

```
gamma: FALSE
```

```
Coefficients:
```

```
 [,1]
```

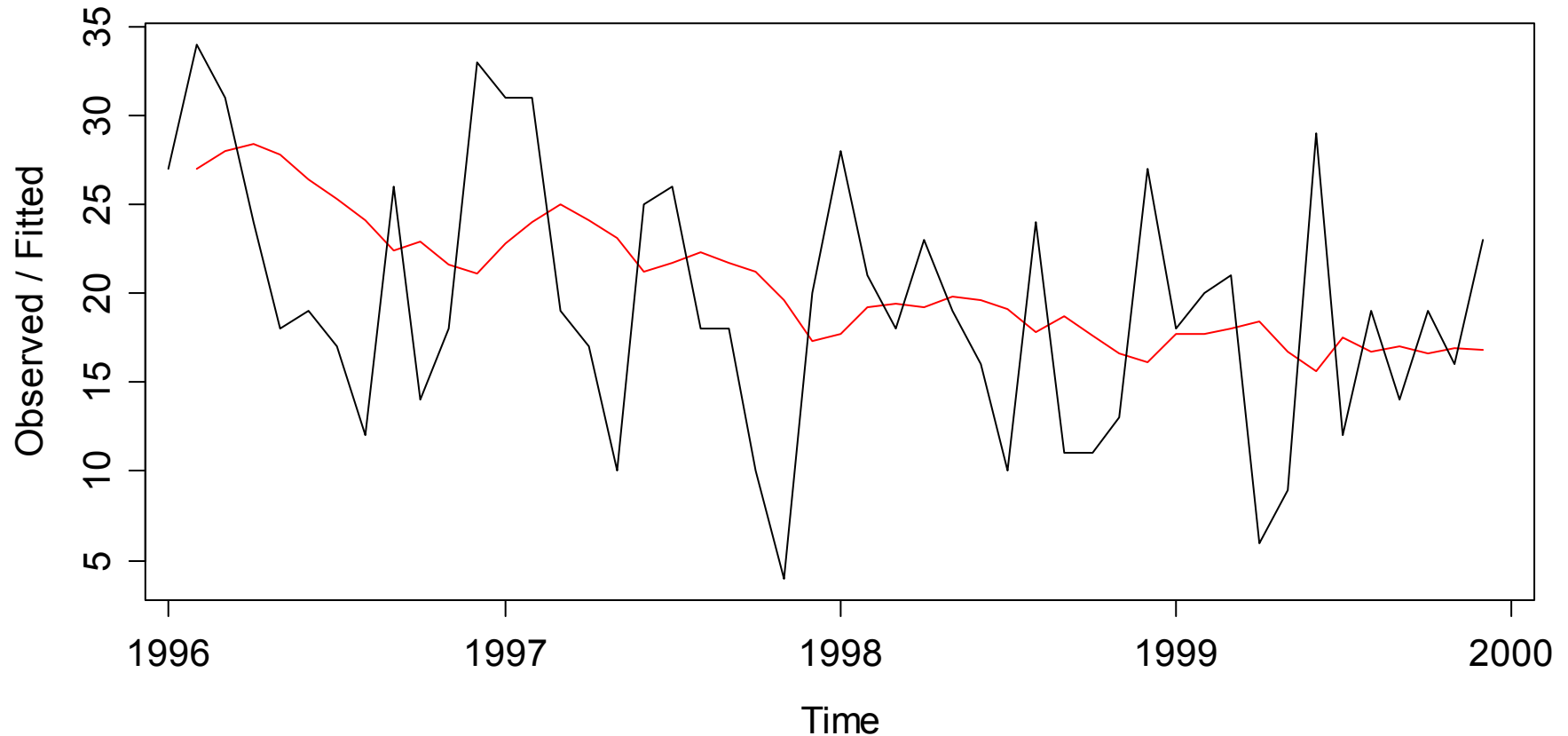
```
a 17.70343
```

Applied Time Series Analysis

SS 2016 – Forecasting

Exponential Smoothing: Example

Holt-Winters filtering



Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters Method

Purpose:

- is for time series with deterministic trend and/or seasonality
- is still a heuristic, model-free approach
- again based on weighted averaging

Is based on these 3 formulae:

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

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

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

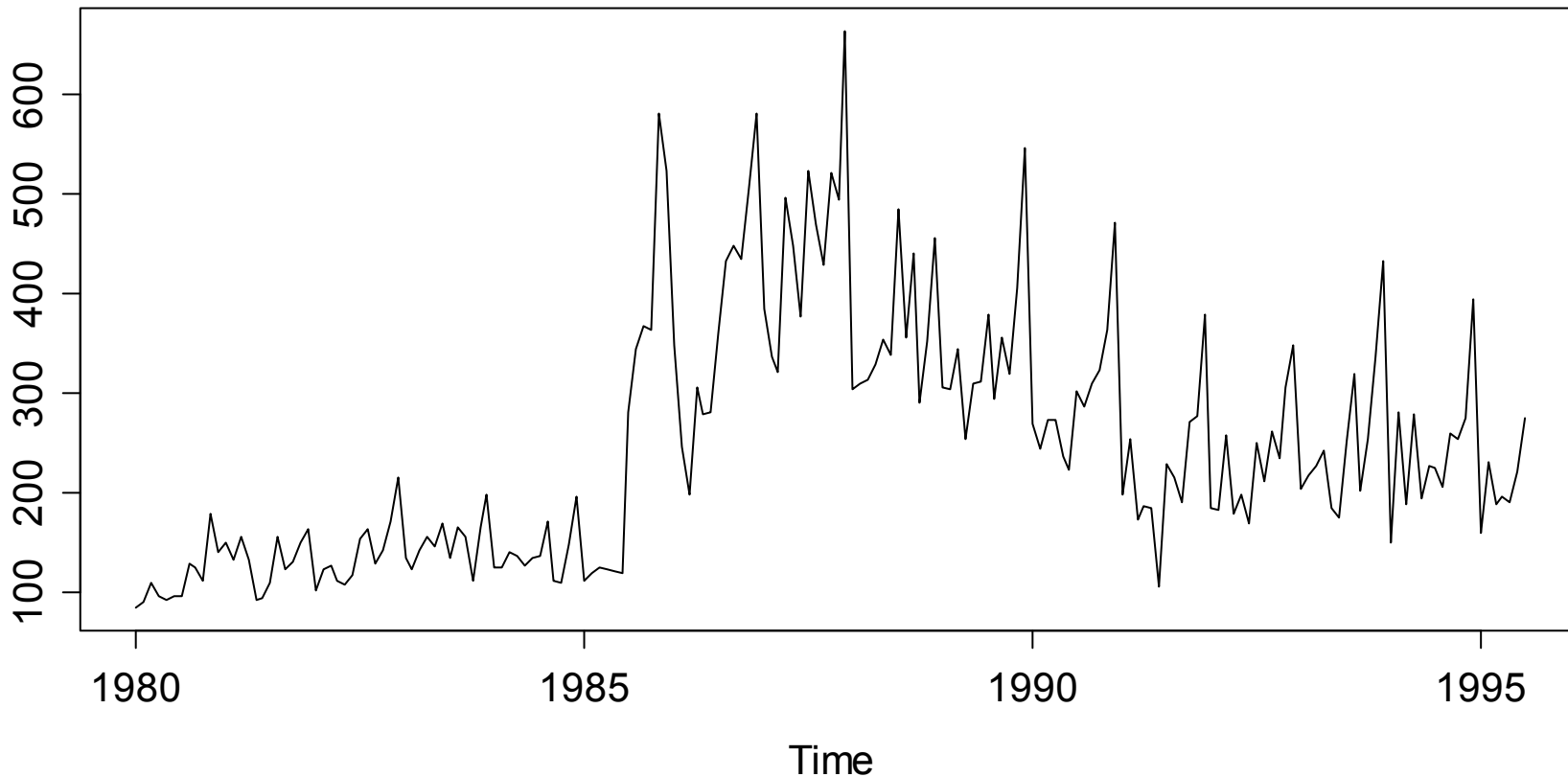
→ **See the blackboard for the derivation...**

Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters: Example

Sales of Australian White Wine

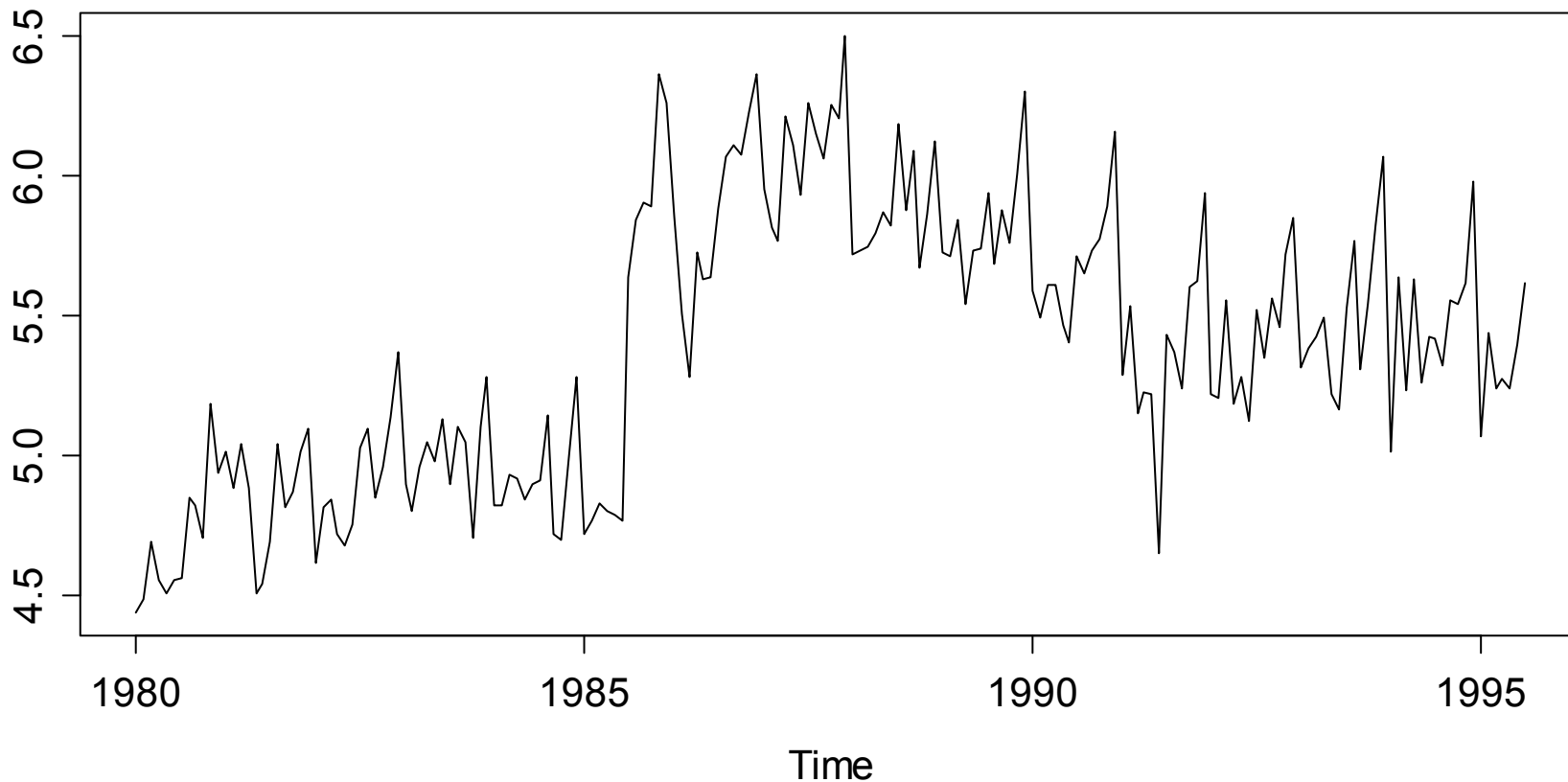


Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters: Example

Logged Sales of Australian White Wine



Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters: R-Code and Output

```
> HoltWinters(x = log(aww))
```

Holt-Winters exponential smoothing with trend and additive seasonal component.

Smoothing parameters:

```
alpha: 0.4148028; beta : 0; gamma: 0.4741967
```

Coefficients:

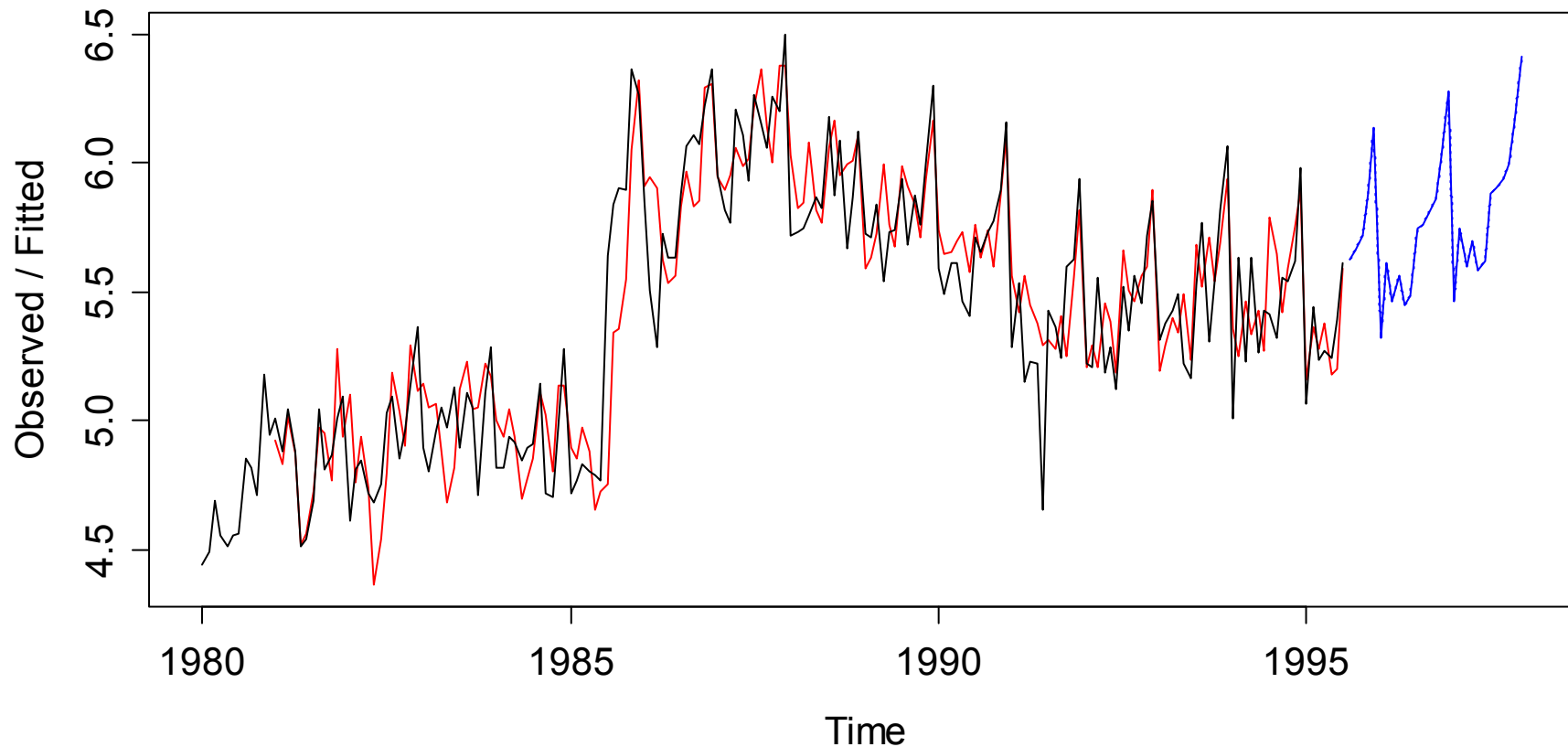
```
a      5.62591329; b      0.01148402
s1     -0.01230437; s2     0.01344762; s3     0.06000025
s4     0.20894897; s5     0.45515787; s6     -0.37315236
s7     -0.09709593; s8     -0.25718994; s9     -0.17107682
s10    -0.29304652; s11    -0.26986816; s12    -0.01984965
```


Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters: Fitted Values & Predictions

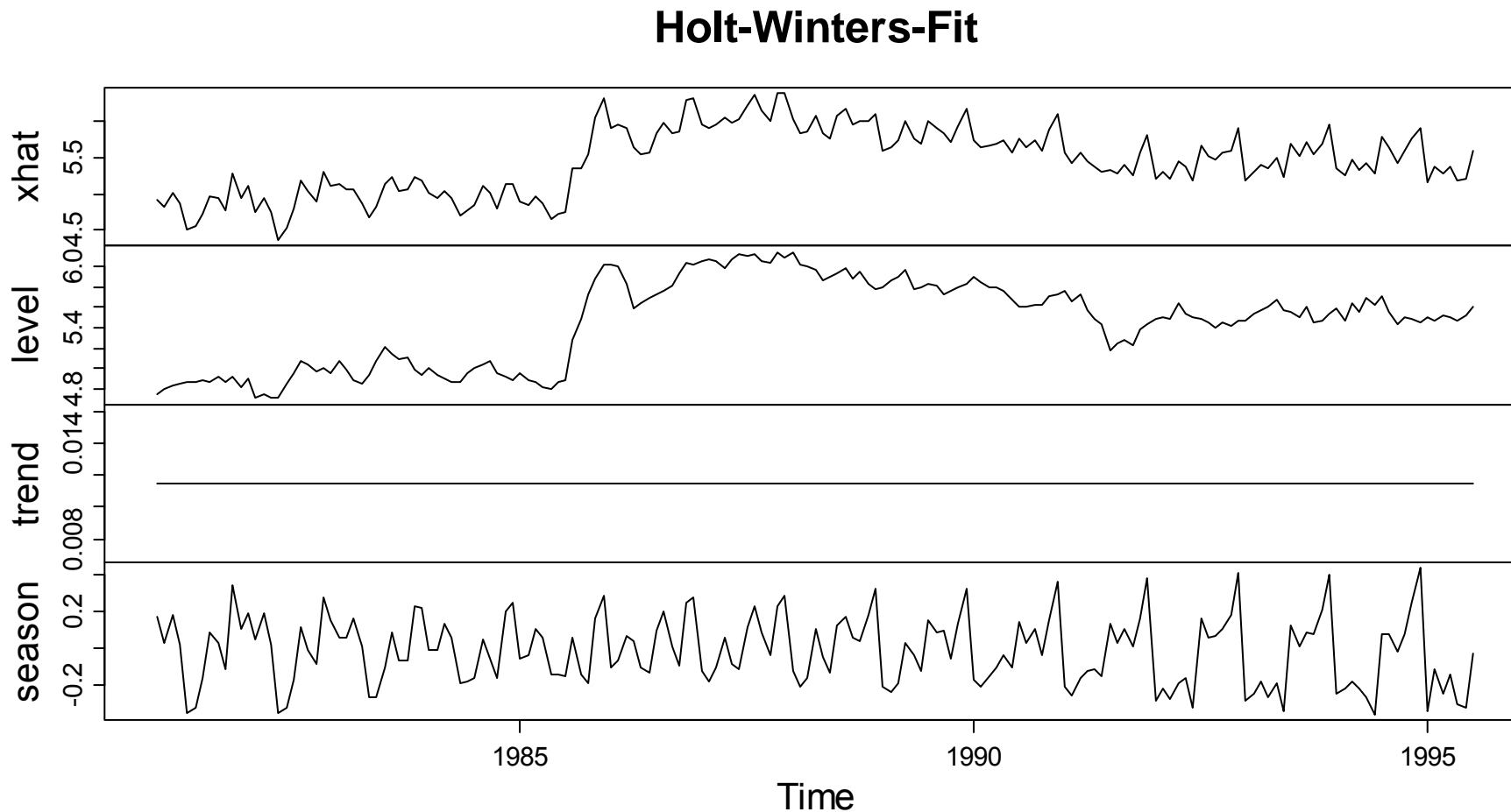
Holt-Winters filtering



Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters: In-Sample Analysis

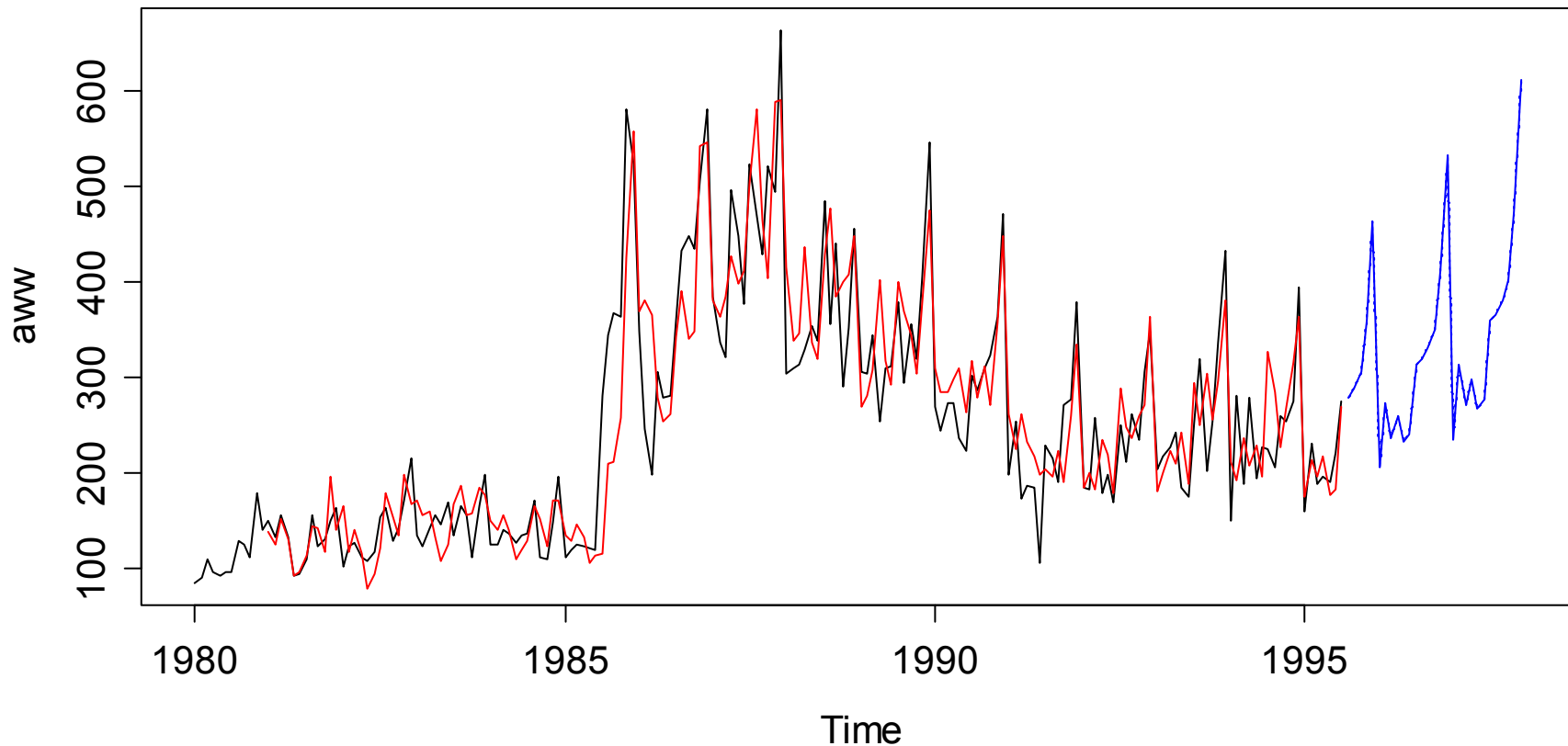


Applied Time Series Analysis

SS 2016 – Forecasting

Holt-Winters: Predictions on Original Scale

Holt-Winters-Forecast for the Original Series



Applied Time Series Analysis

SS 2016 – Forecasting

Exercise

Data:

- *use the Australian white wine sales data...*
- *... or any other dataset you like*

Goal:

- *Find a good model describing these data*
- *Evaluate which model yields the best predictions*
- *Generate a 29-month forecast from this model*

Method:

- *Remove the last 29 observations and mimic oos-forecasting*

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Multivariate Time Series Analysis

Idea: Infer the relation between two time series

$$X_1 = (X_{1,t}) \text{ and } X_2 = (X_{2,t}).$$

What is the difference to time series regression?

- Here, the two series arise „on an equal footing“, and we are interested in the correlation between them.
- In time series regression, the two (or more) series are causally related and we are interested in inferring that relation. There is an independent and several dependent variables.
- The difference is comparable to the difference between correlation and regression.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Example: Permafrost Boreholes



A collaboration between the Swiss Institute for Snow and Avalanche Research with the Zurich University of Applied Sciences:

Evelyn Zenklusen Mutter & Marcel Dettling

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Example: Permafrost Boreholes

- Given is a bivariate time series with 2×92 observations
- 2 measurements were made everyday in summer 2006
- Series 1: air temperature at Plathorn 3345m
- Series 2: soil temperature at Hörnli hut 3295m

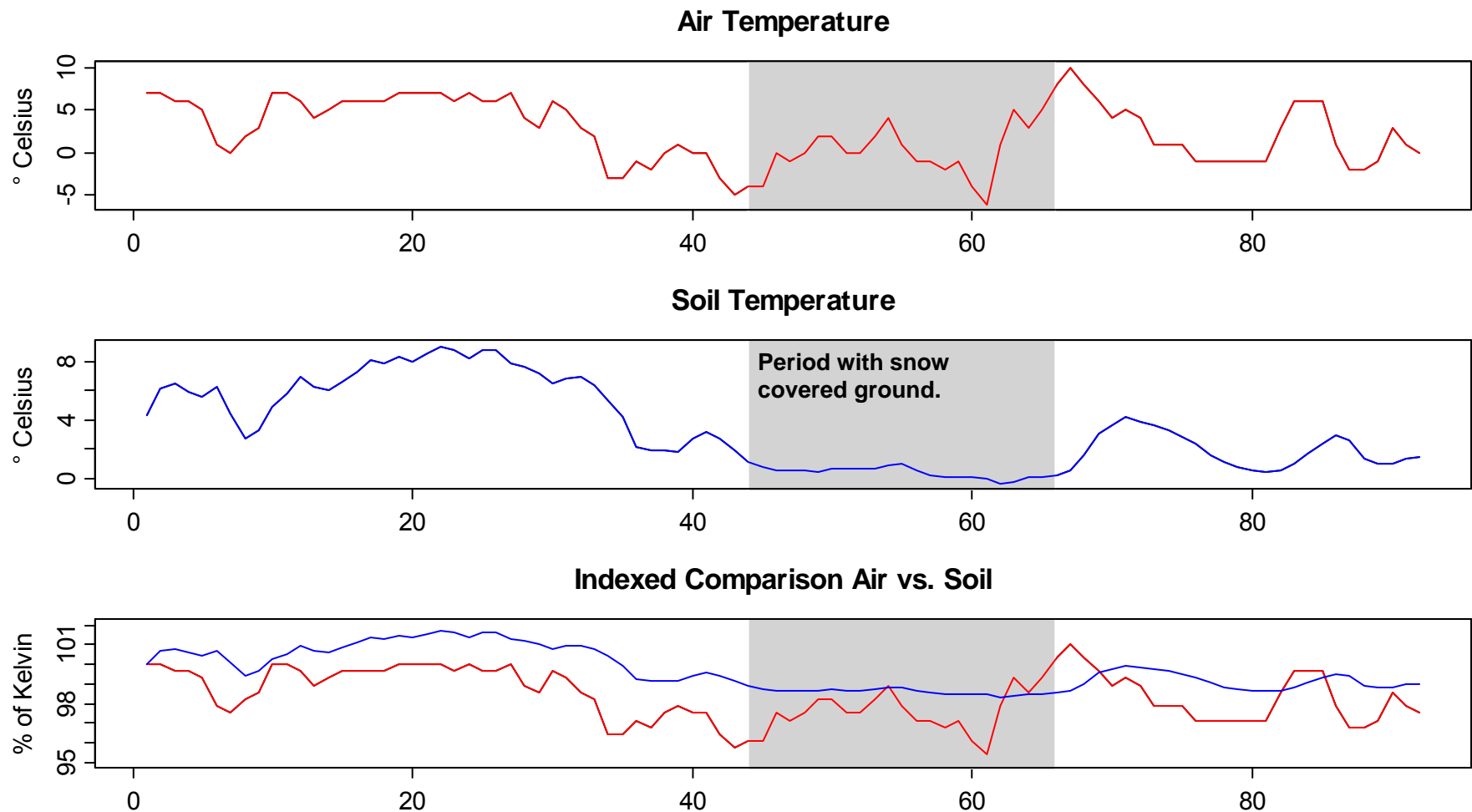
Goal of the analysis:

- 1) Answer whether changes in the air temperature are correlated with changes in the soil temperature.
- 2) If a correlation is present, what is the delay?

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Air & Soil Temperature Comparison

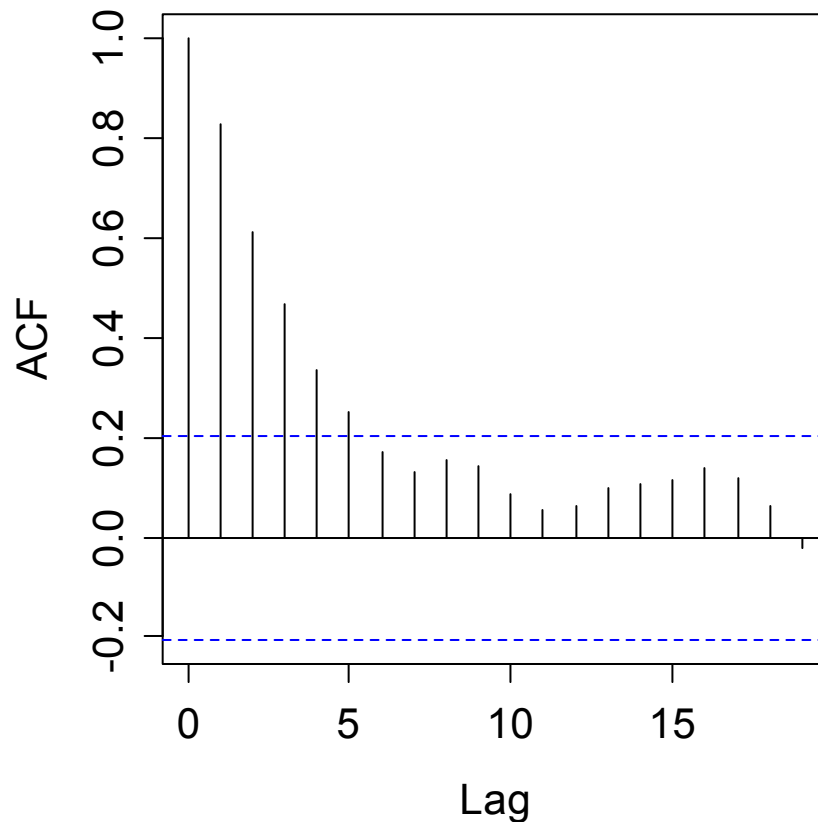


Applied Time Series Analysis

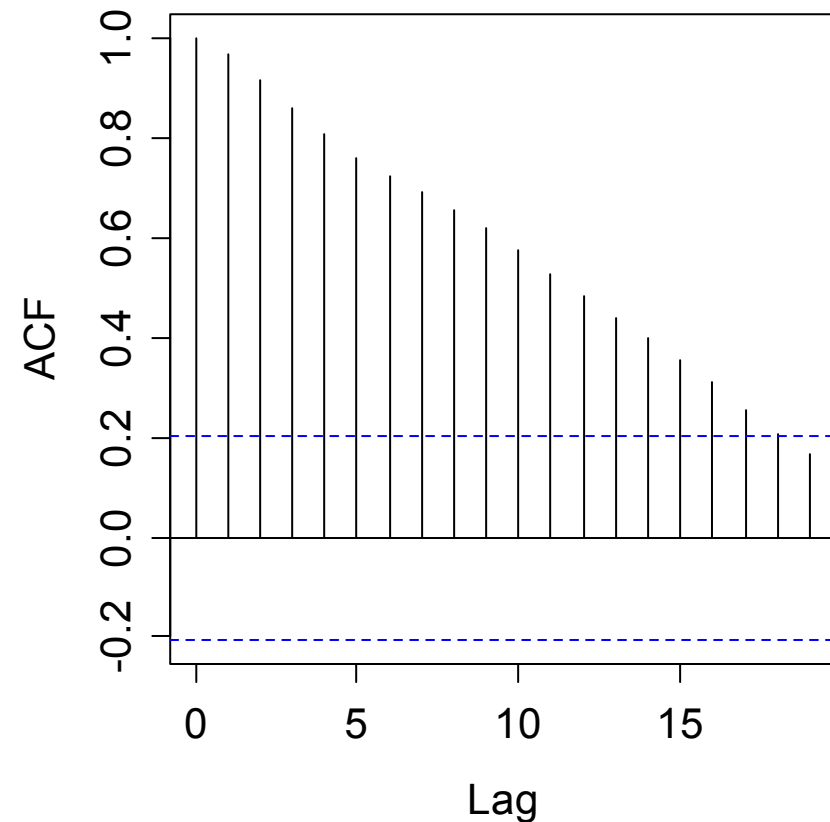
SS 2016 – Multivariate Time Series Analysis

Are the Series Stationary?

ACF of Air Temperature



ACF of Soil Temperature



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

How to Proceed?

- 1) The series seem to have „long memory“
 - 2) Pure AR/MA/ARMA do not fit the data well
- Differencing may help with this

Another advantage of taking differences:

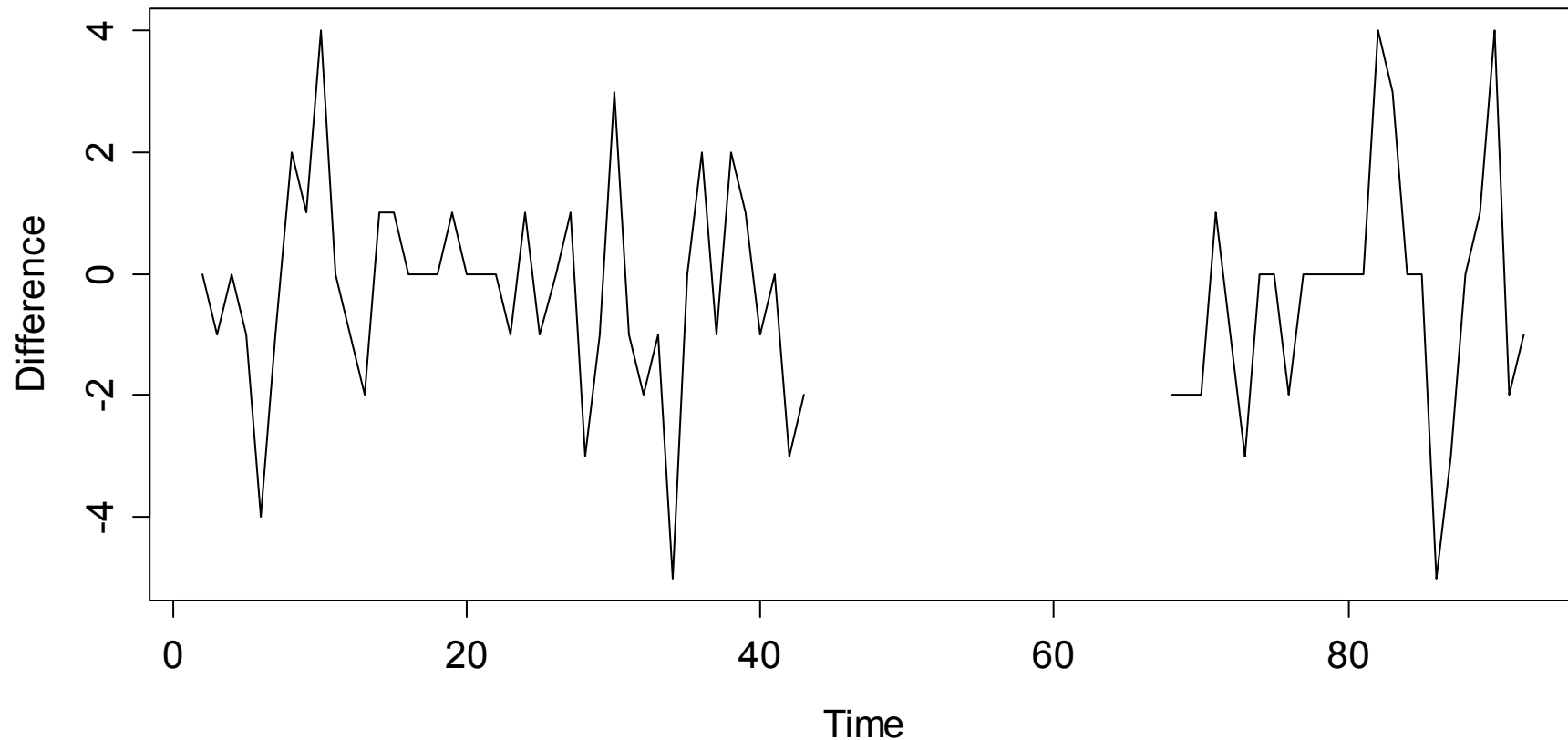
- we infer, whether there is a relation between the changes in the air temperatures, and the changes in the soil temperatures.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Changes in the Air Temperature

Changes in the Air Temperature

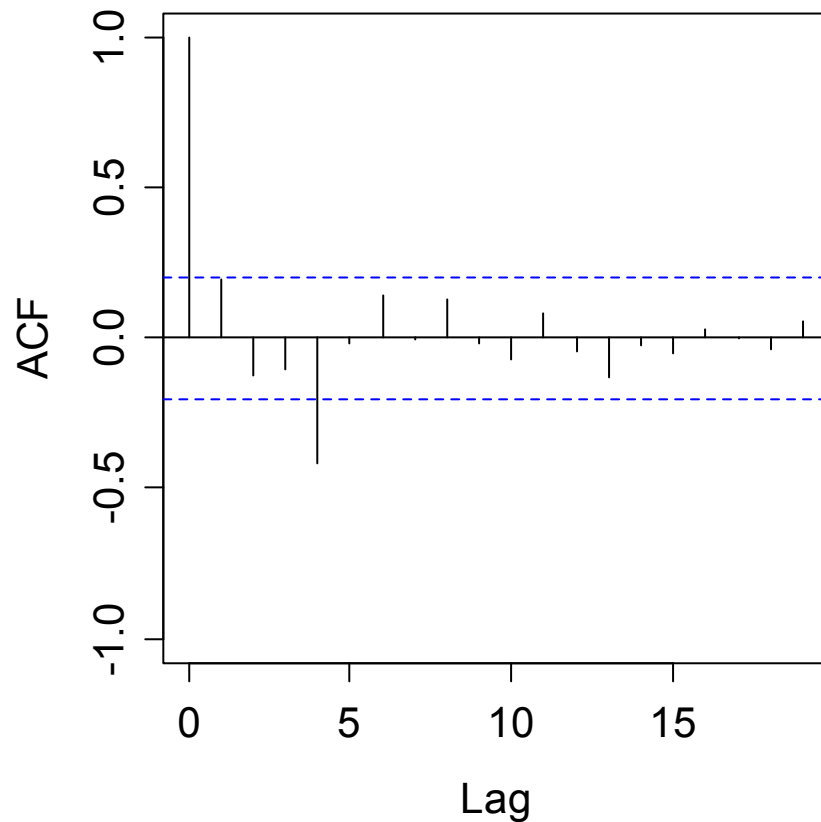


Applied Time Series Analysis

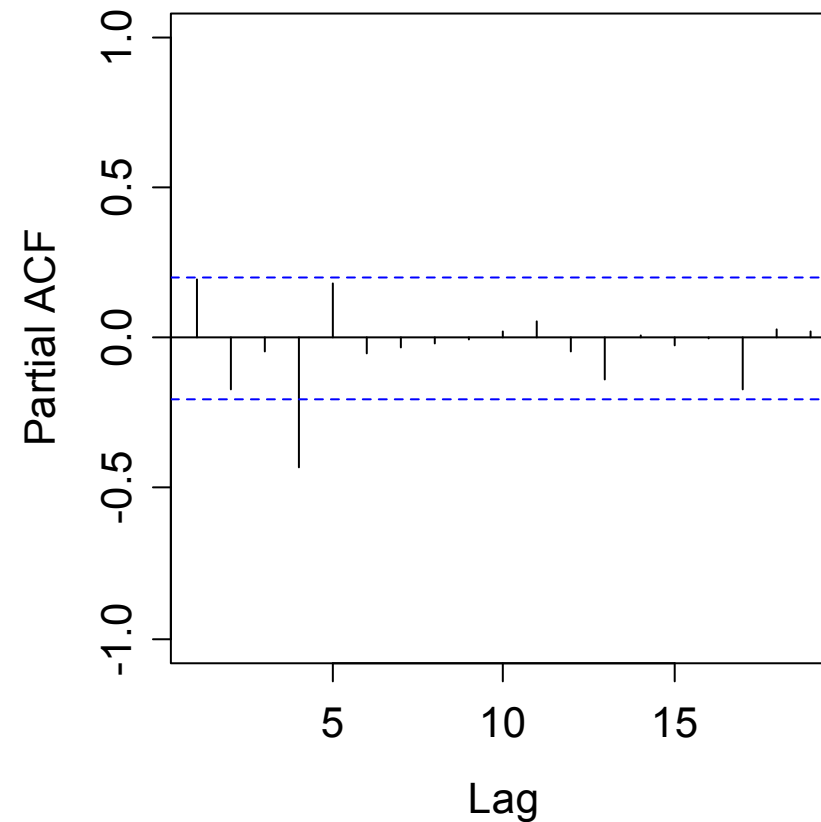
SS 2016 – Multivariate Time Series Analysis

ACF/PACF for Air Temperature Changes

ACF



PACF

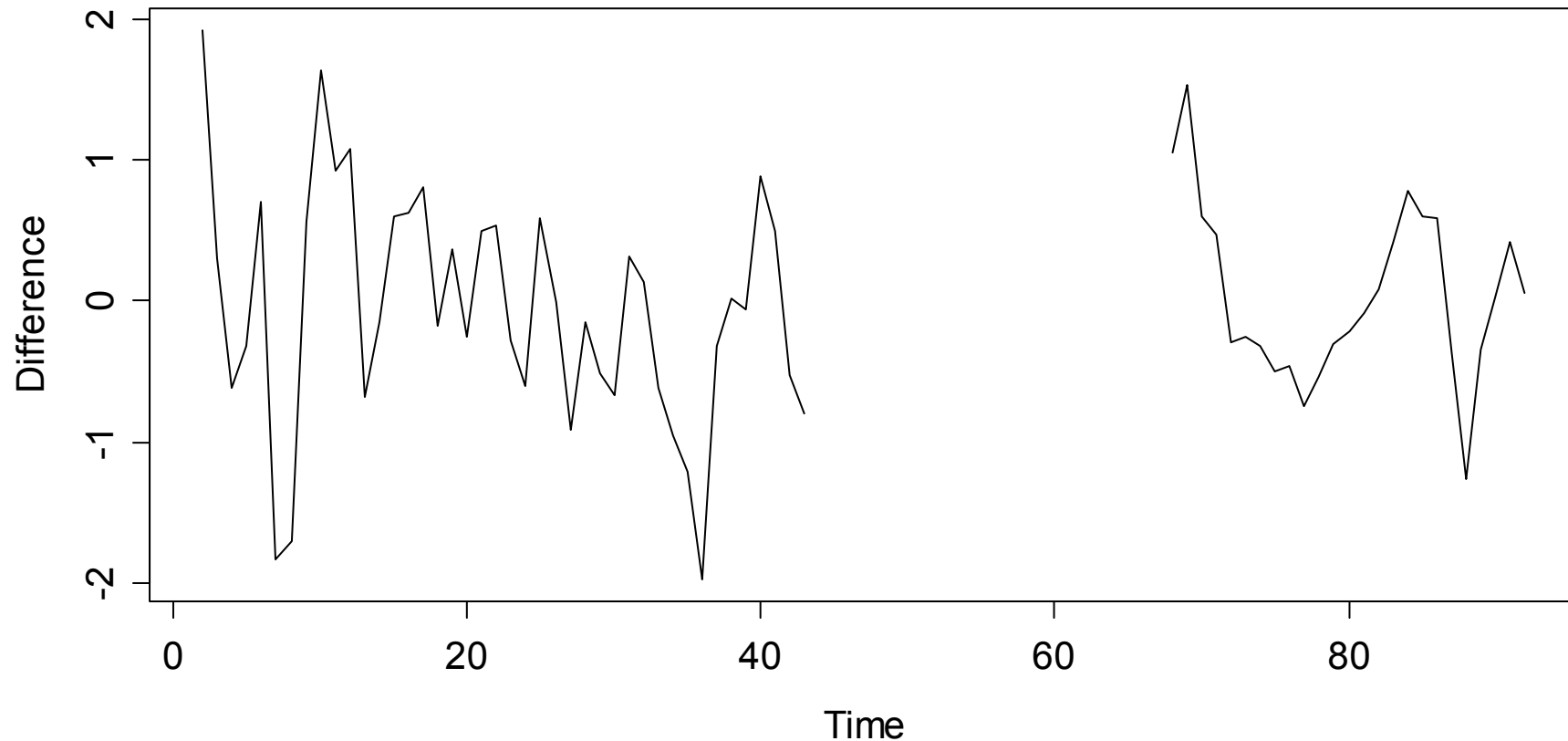


Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Changes in the Soil Temperature

Changes in the Soil Temperature

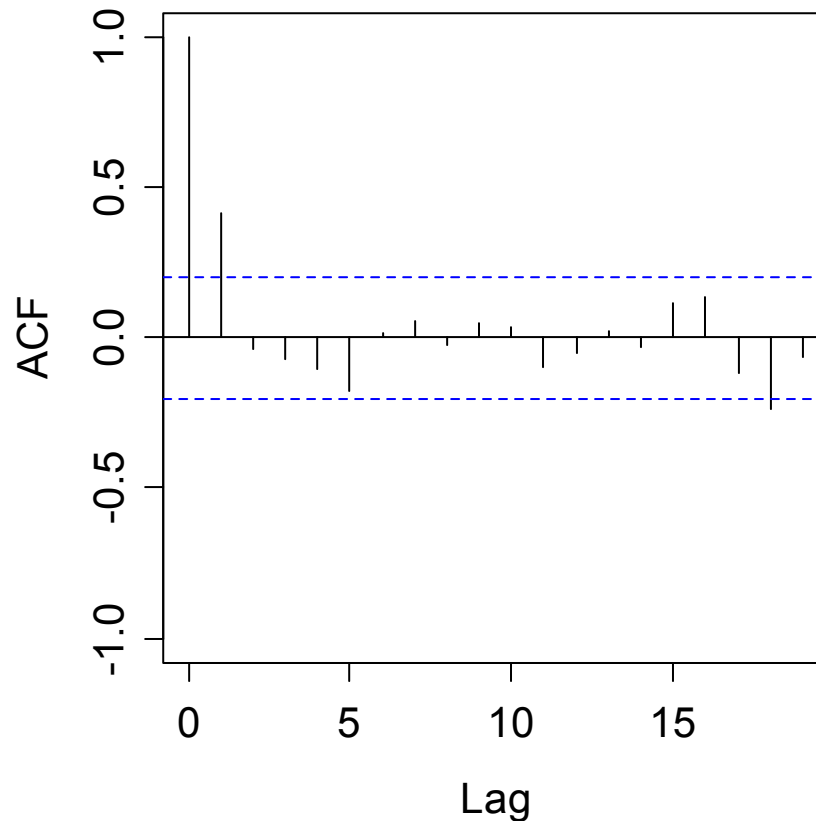


Applied Time Series Analysis

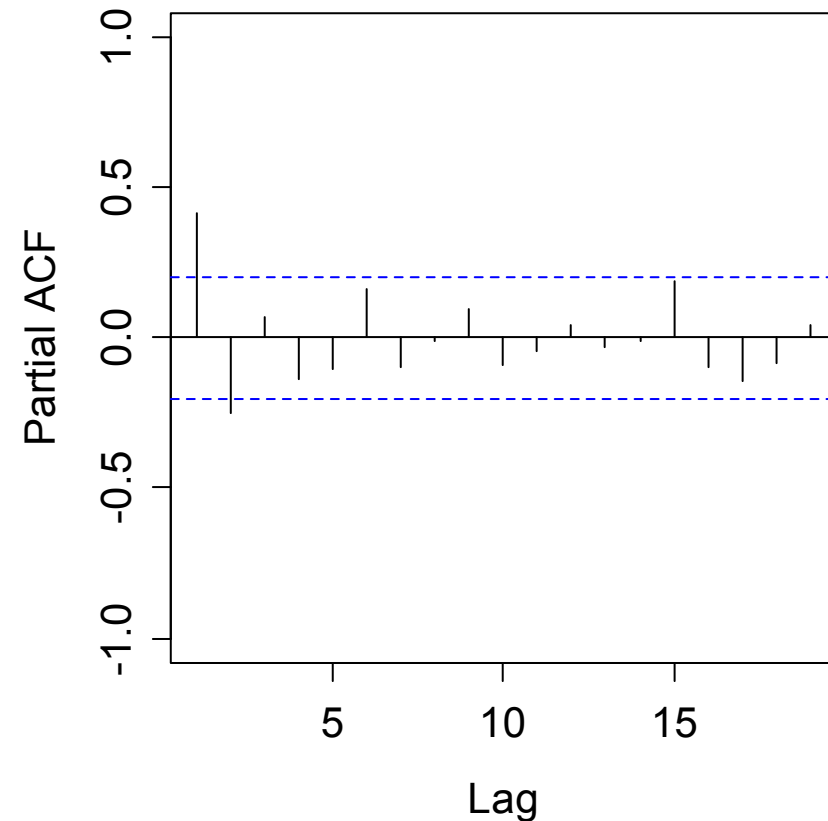
SS 2016 – Multivariate Time Series Analysis

ACF/PACF for Soil Temperature Changes

ACF



PACF



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Cross Covariance

The cross correlations describe the relation between two time series. However, note that the interpretation is quite tricky!

$$\left. \begin{aligned} \gamma_{11}(k) &= \text{Cov}(X_{1,t+k}, X_{1,t}) \\ \gamma_{22}(k) &= \text{Cov}(X_{2,t+k}, X_{2,t}) \end{aligned} \right\} \begin{array}{l} \text{usual „within series“} \\ \text{covariance} \end{array}$$

$$\left. \begin{aligned} \gamma_{12}(k) &= \text{Cov}(X_{1,t+k}, X_{2,t}) \\ \gamma_{21}(k) &= \text{Cov}(X_{2,t+k}, X_{1,t}) \end{aligned} \right\} \begin{array}{l} \text{cross covariance,} \\ \text{independent from } t \end{array}$$

Also, we have: $\gamma_{12}(-k) = \text{Cov}(X_{1,t-k}, X_{2,t}) = \text{Cov}(X_{2,t+k}, X_{1,t}) = \gamma_{21}(k)$

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Cross Correlations

It suffices to analyze $\gamma_{12}(k)$, and neglect $\gamma_{21}(k)$, but we have to regard both positive and negative lags k .

We again prefer to work with correlations:

$$\rho_{12}(k) = \frac{\gamma_{12}(k)}{\sqrt{\gamma_{11}(0)\gamma_{22}(0)}}$$

which describe the linear relation between two values of X_1 and X_2 , when the series X_1 is k time units ahead.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Estimation

Cross covariances and correlations are estimated as follows:

$$\hat{\gamma}_{12}(k) = \frac{1}{n} \sum_t (x_{1,t+k} - \bar{x}_1)(x_{2,t} - \bar{x}_2)$$

and

$$\hat{\rho}_{12}(k) = \frac{\hat{\gamma}_{12}(k)}{\sqrt{\hat{\gamma}_{11}(0)\hat{\gamma}_{22}(0)}}, \quad \text{respectively.}$$

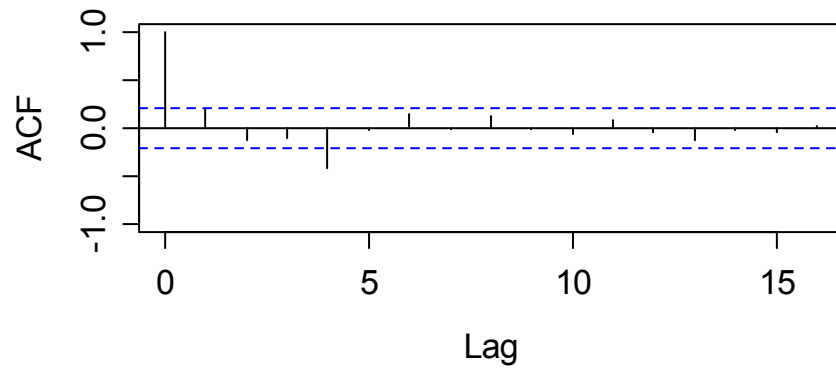
The plot of $\hat{\rho}_{12}(k)$ versus the lag k is called the cross correlogram. It has to be inspected for both $+$ and $-k$.

Applied Time Series Analysis

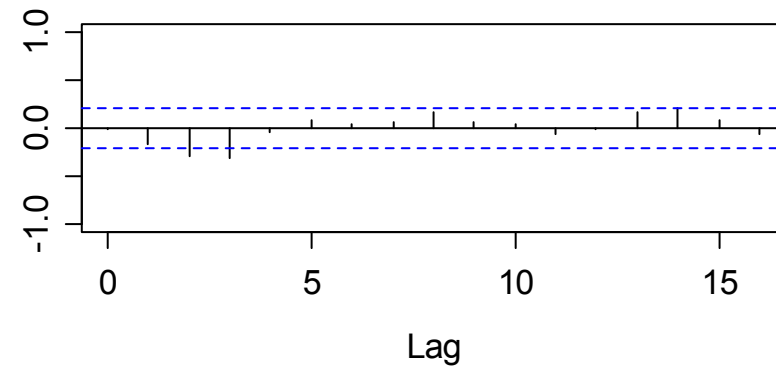
SS 2016 – Multivariate Time Series Analysis

Sample Cross Correlation

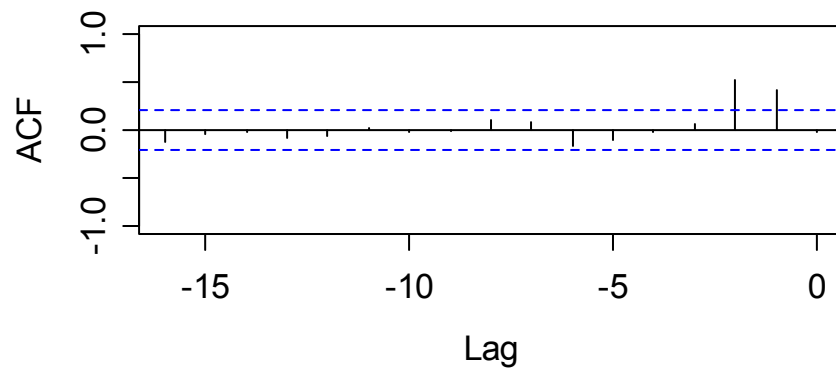
air.changes



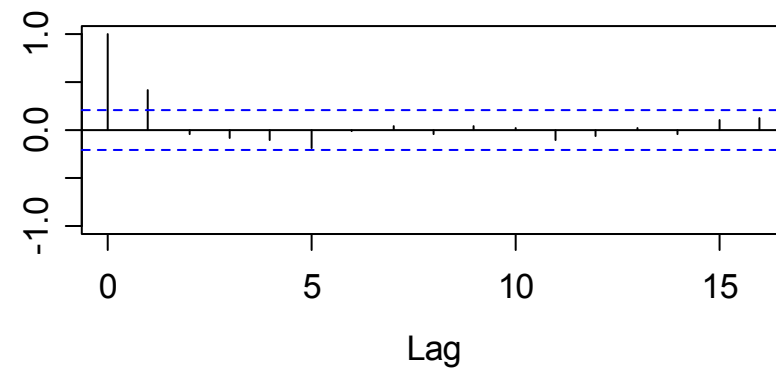
air.changes & soil.changes



soil.changes & air.changes



soil.changes



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Interpreting the Sample Cross Correlation

The confidence bounds in the sample cross correlation are only valid in some special cases, i.e. if there is no cross correlation and at least one of the series is uncorrelated.

Important: the confidence bounds are often too small!

For computing them, we need: $Var(\hat{\rho}_{12}(k))$

This is a difficult problem. We are going to discuss a few special cases and then show how the problem can be circumvented.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Special Case 1

We assume that there is no cross correlation for large lags k :

If $\rho_{12}(j) = 0$ for $|j| \geq m$, we have for $|k| \geq m$:

$$\text{Var}(\hat{\rho}_{12}(k)) \approx \frac{1}{n} \sum_{j=-\infty}^{\infty} (\rho_{11}(j)\rho_{22}(j) + \rho_{12}(j+k)\rho_{12}(j-k))$$

This goes to zero for large n and we thus have consistency. For giving statements about the confidence bounds, we would have to know more about the cross correlations, though.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Special Case 2

There is no cross correlation, but X_1 and X_2 are both time series that show correlation „within“:

$$\text{Var}(\hat{\rho}_{12}(k)) \approx \frac{1}{n} \sum_{j=-\infty}^{\infty} \rho_{11}(j) \rho_{22}(j)$$

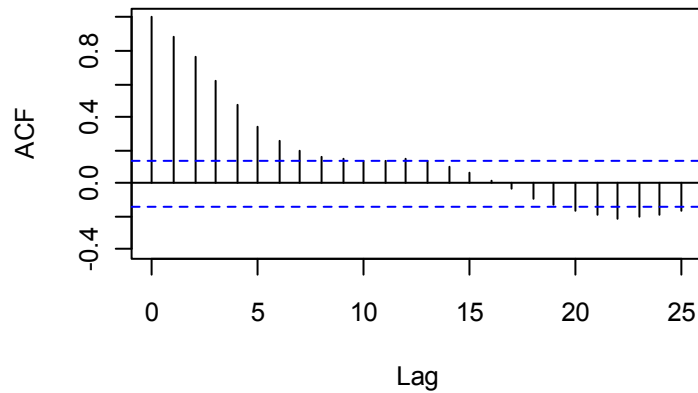
See the blackboard... for the important example showing that the cross correlation estimations can be arbitrarily bad!

Applied Time Series Analysis

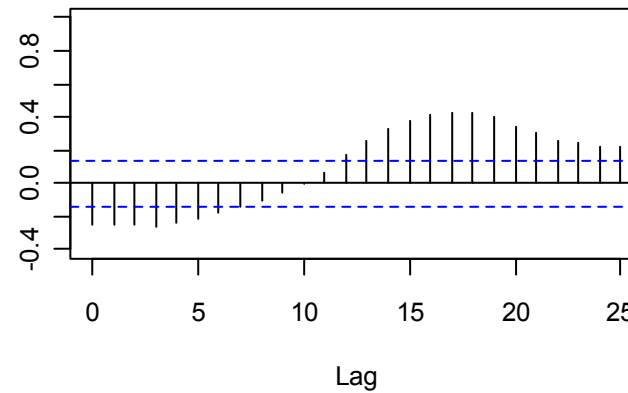
SS 2016 – Multivariate Time Series Analysis

Special Case 2: Simulation Example

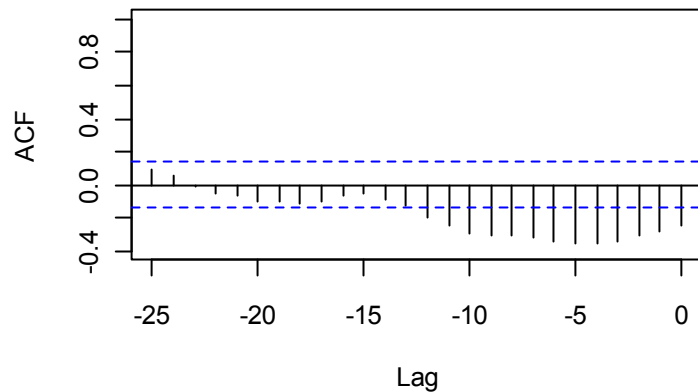
Y1



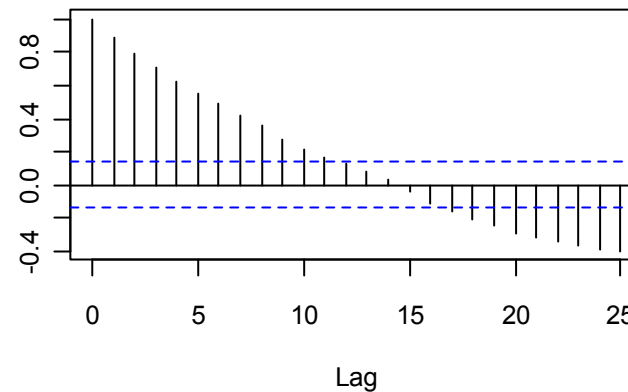
Y1 & Y2



Y2 & Y1



Y2



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Special Case 3

There is no cross correlation, and X_1 is a White Noise series that is independent from X_2 . Then, the estimation variance simplifies to:

$$\text{Var}(\hat{\rho}_{12}(k)) \approx \frac{1}{n}$$

Thus, the confidence bounds are valid in this case.

However, we introduced the concept of cross correlation to infer the relation between correlated series. The trick of the so-called „prewhitening“ helps.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Prewhitening

Prewhitening means that the time series is transformed such that it becomes a white noise process, i.e. is uncorrelated.

We assume that both stationary processes X_1 and X_2 can be rewritten as follows:

$$U_t = \sum_{i=0}^{\infty} a_i X_{1,t-i} \quad \text{and} \quad V_t = \sum_{i=0}^{\infty} b_i X_{2,t-i},$$

with uncorrelated U_t and V_t . Note that this is possible for ARMA(p,q) processes by writing them as an $AR(\infty)$. The left hand side of the equation then is the innovation.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Cross Correlation of Prewhitened Series

The cross correlation between U_t and V_t can be derived from the one between X_1 and X_2 :

$$\rho_{UV}(k) = \sum_{j=0}^{\infty} \sum_{i=0}^{\infty} a_i b_j \rho_{X_1 X_2}(k + i - j)$$

Thus we have:

$$\rho_{UV}(k) = 0 \quad \text{for all } k \Leftrightarrow \rho_{X_1 X_2}(k) = 0 \quad \text{for all } k$$

Now: generate U_t, V_t ; estimate cross correlations; and, by using the confidence bands, check whether they are significant

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Simulation Example

Since we are dealing with simulated series, we know that:

$$X_{i,t} = 0.9 \cdot X_{i,t-1} + E_t, \text{ thus } E_t = X_{i,t} - 0.9 \cdot X_{i,t-1}$$

In practice, we don't know the AR-coefficients, but plug-in the respective estimates:

$$U_t = X_{1,t} - \hat{\alpha}_{1,1} X_{1,t-1} \quad \text{with} \quad \hat{\alpha}_{1,1} = 0.911$$

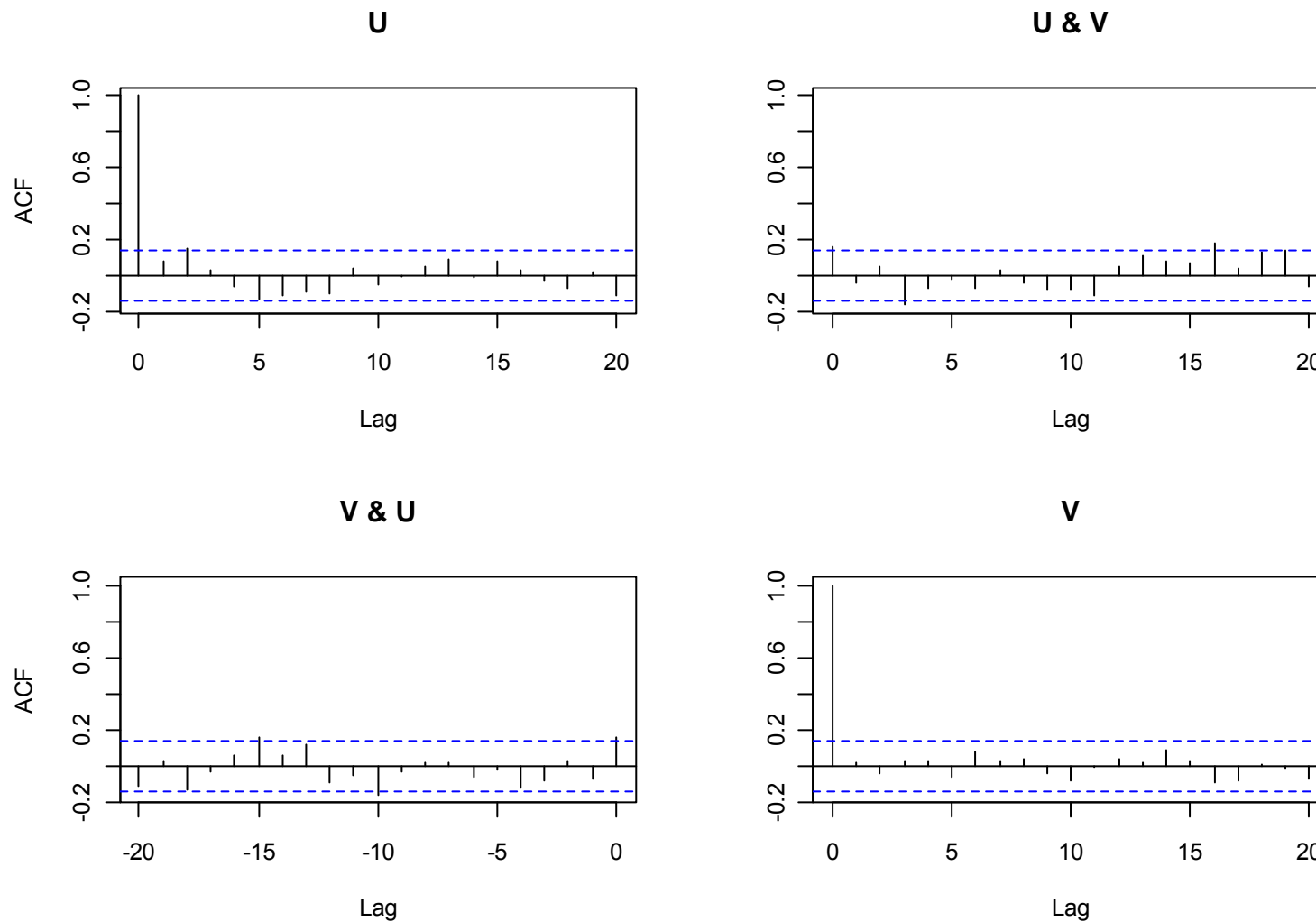
$$V_t = X_{2,t} - \hat{\alpha}_{2,1} X_{2,t-1} \quad \text{with} \quad \hat{\alpha}_{2,1} = 0.822$$

We will now analyse the sample cross correlation of U_t and V_t , which will also allow to draw conclusions about X_1 and X_2 .

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Cross Correlation in the Simulation Example



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Cross Correlation in the Simulation Example

We observe that:

- U_t and V_t are white noise processes
- There are no (strongly) significant cross correlations

We conjecture that:

- X_1 and X_2 are not cross correlated either.

→ This matches our „expectations“, or better, true process.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Prewhitening the Borehole Data

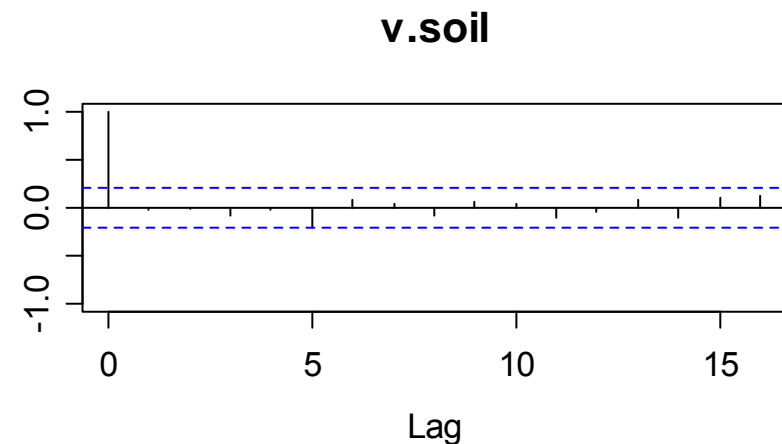
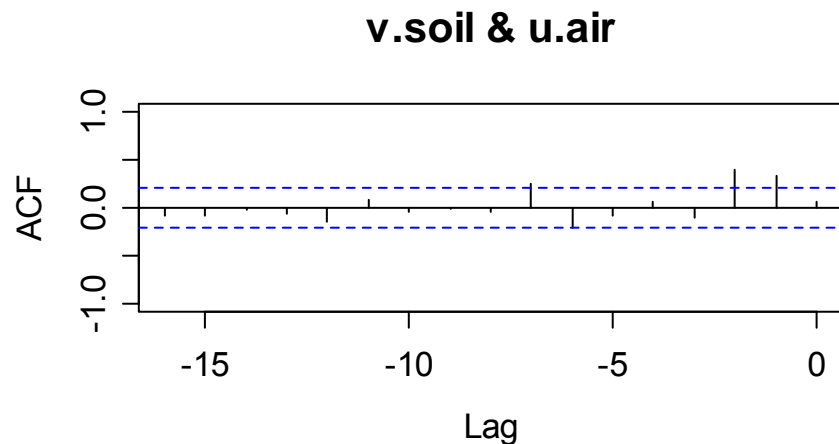
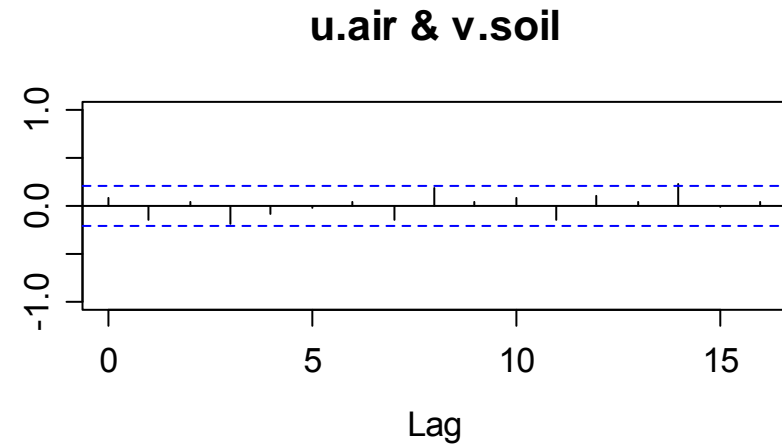
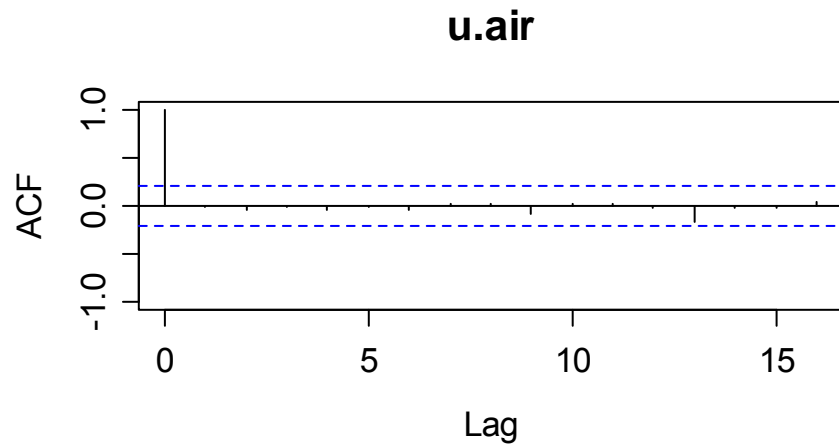
What to do:

- ARMA(p,q)-models are fitted to the differenced series
- Best choice: AR(5) for the air temperature differences
MA(1) for the soil temperature differences
- The residual time series are U_t and W_t , White Noise
- Check the sample cross correlation (see next slide)
- Model the output as a linear combination of past input values: transfer function model.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Prewhitening the Borehole Data



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Transfer Function Models

Properties:

- Transfer function models are an option to describe the dependency between two time series.
- The first (input) series influences the second (output) one, but there is no feedback from output to input.
- The influence from input to output only goes „forward“.

The model is:

$$X_{2,t} - \mu_2 = \sum_{j=0}^{\infty} v_j (X_{1,t-j} - \mu_1) + E_t$$

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Transfer Function Models

The model is:

$$X_{2,t} - \mu_2 = \sum_{j=0}^{\infty} v_j (X_{1,t-j} - \mu_1) + E_t$$

- $E[E_t] = 0$
- E_t and $X_{1,s}$ are uncorrelated for all t and s .
- E_t and E_s are usually correlated.
- For simplicity of notation, we here assume that the series have been mean centered.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Cross Covariance

When plugging-in, we obtain for the cross covariance:

$$\gamma_{21}(k) = \text{Cov}(X_{2,t+k}, X_{1,t}) = \text{Cov}\left(\sum_{j=0}^{\infty} v_j X_{1,t+k-j}, X_{1,t}\right) = \sum_{j=0}^{\infty} v_j \gamma_{11}(k-j)$$

- If only finitely many coefficients are different from zero, we could generate a linear equation system, plug-in $\hat{\gamma}_{11}$ and $\hat{\gamma}_{21}$ to obtain the estimates \hat{v}_j .

→ This is not a statistically efficient estimation method.

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Special Case: $X_{1,t}$ Uncorrelated

If $X_{1,t}$ was an uncorrelated series, we would obtain the coefficients of the transfer function model quite easily:

$$v_k = \frac{\gamma_{21}(k)}{\gamma_{11}(0)}$$

However, this is usually not the case. We can then:

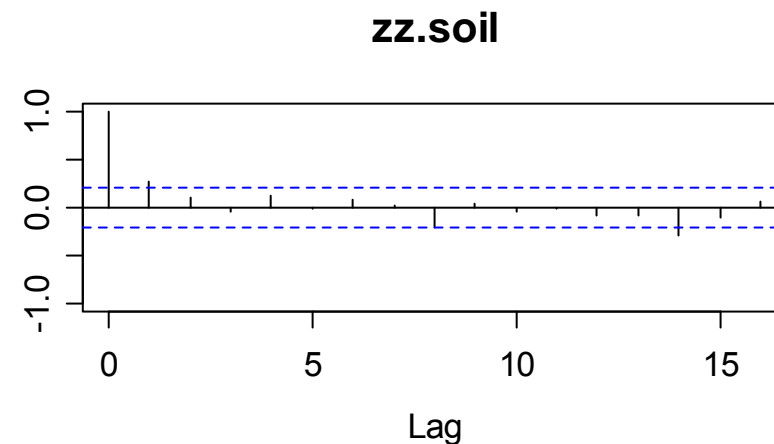
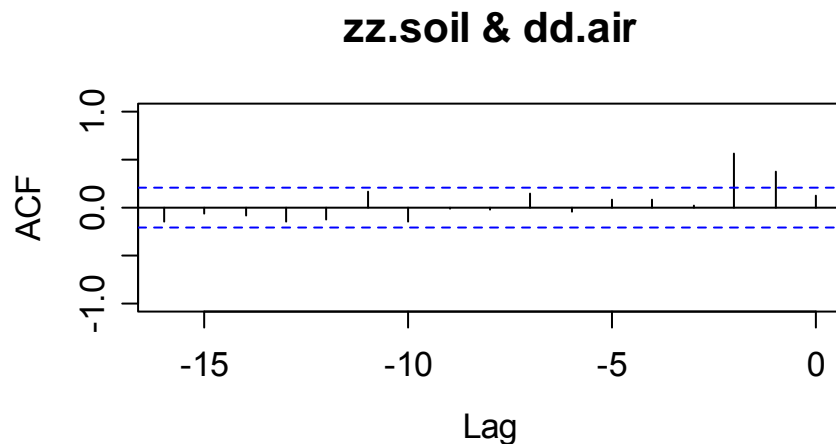
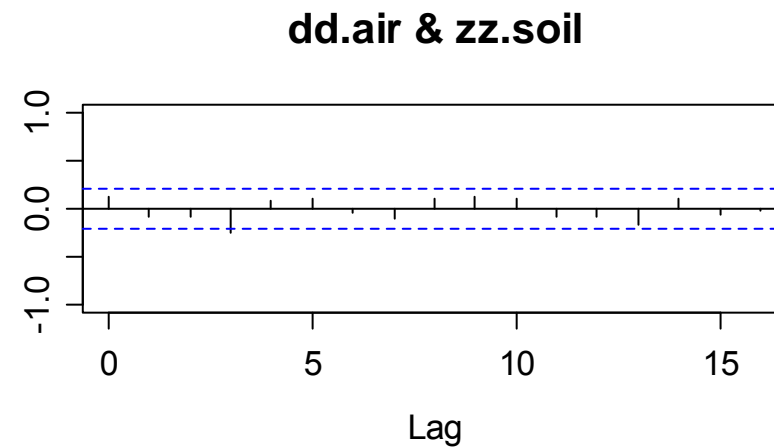
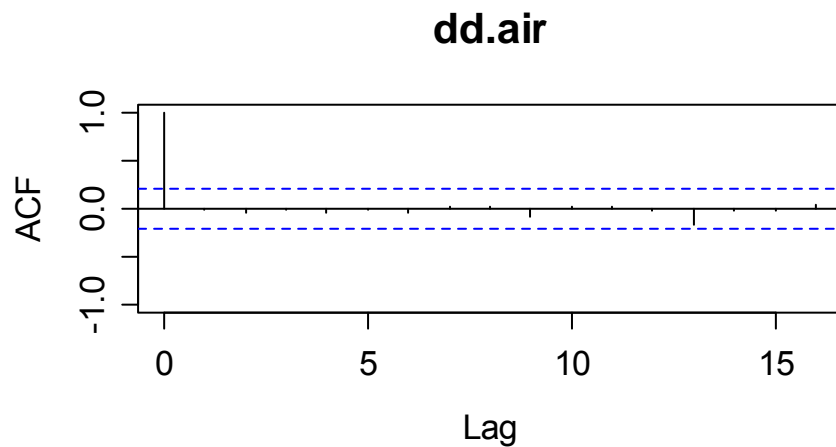
- transform all series in a clever way
- the transfer function model has identical coefficients
- the new, transformed input series is uncorrelated

→ see blackboard for the transformation

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Borehole Transformed



Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Borehole: Final Remarks

- In the previous slide, we see the empirical cross correlations $\hat{\rho}_{21}(k)$ of the two series D_t and Z_t .
- The coefficients \hat{v}_k from the transfer function model will be proportional to the empirical cross correlations. We can already now conjecture that the output is delayed by 1-2 days.
- The formula for the transfer function model coefficients is:

$$\hat{v}_k = \frac{\hat{\sigma}_Z}{\hat{\sigma}_D} \hat{\rho}_{21}(k)$$

Applied Time Series Analysis

SS 2016 – Multivariate Time Series Analysis

Borehole: R-Code and Results

```
> dd.air <- resid(fit.air)
> coefs <- coef(fit.air)[1:5]
> zz.soil <- filter(diff(soil.na), c(1, -coefs, sides=1))
> as.int <- ts.intersect(dd.air, zz.soil)
> acf.val <- acf(as.int, na.action=na.pass)
```

Transfer Function Model Coefficients:

```
> multip <- sd(zz.soil, na.rm=..)/sd(dd.air, na.rm=..)
> multip*acf.val$acf[,2,1]
[1] 0.054305137 0.165729551 0.250648114 0.008416697
[5] 0.036091971 0.042582917 -0.014780751 0.065008411
[9] -0.002900099 -0.001487220 -0.062670672 0.073479065
[13] -0.049352348 -0.060899602 -0.032943583 -0.025975790
[17] -0.057824007
```

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Spectral Analysis

Basis: Many time series show (stochastic) periodic behavior. The goal of spectral analysis is to understand the cycles at which highs & lows in the data appear.

Idea: Time series are interpreted as a combination of cyclic components. For observed series, a decomposition into a linear combination of harmonic oscillations is set up and used as a basis for estimating the spectrum.

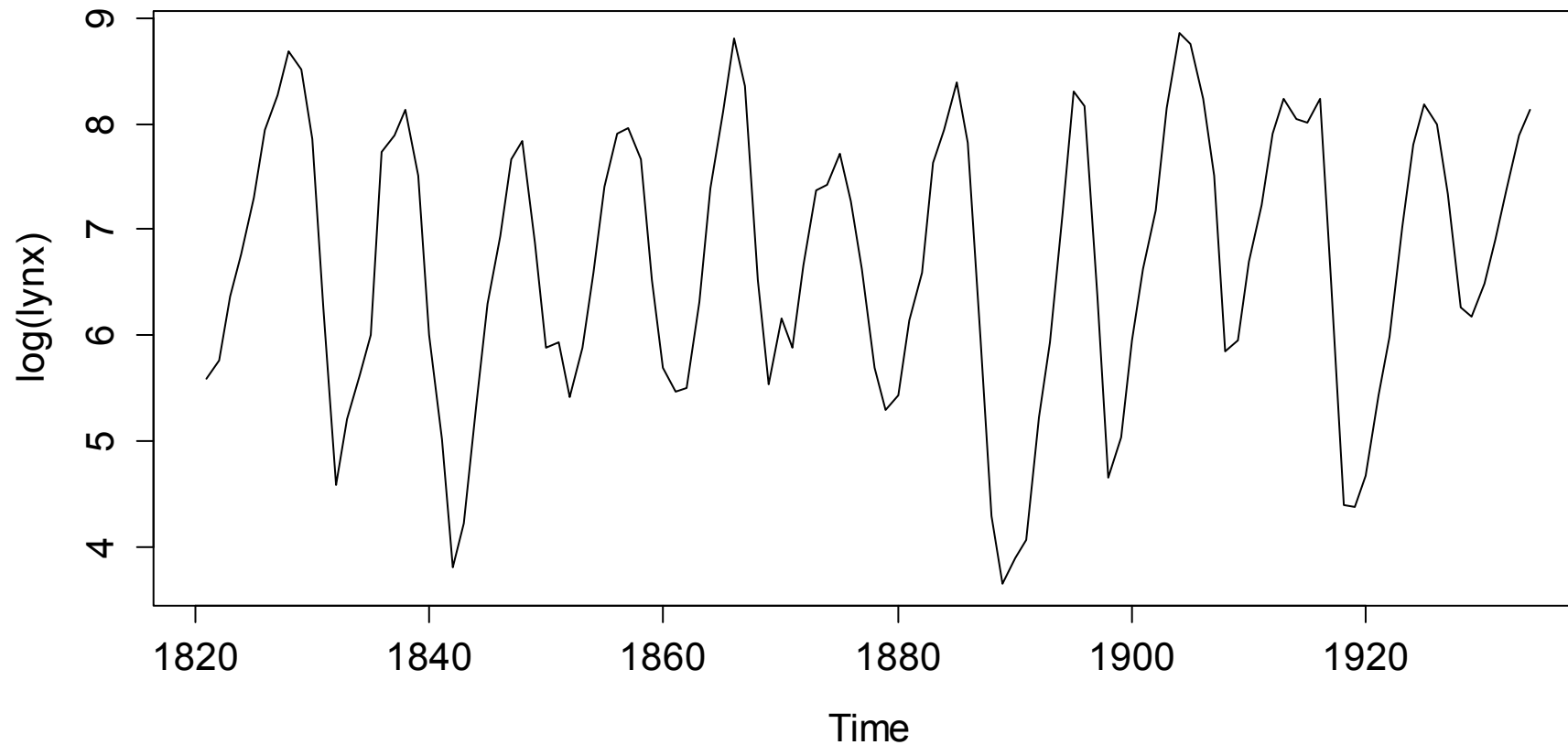
Why: As a descriptive means, showing the character and the dependency structure within the series. There are some important applications in engineering, economics and medicine.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Lynx Data

Time Series Plot of $\log(\text{lynx})$

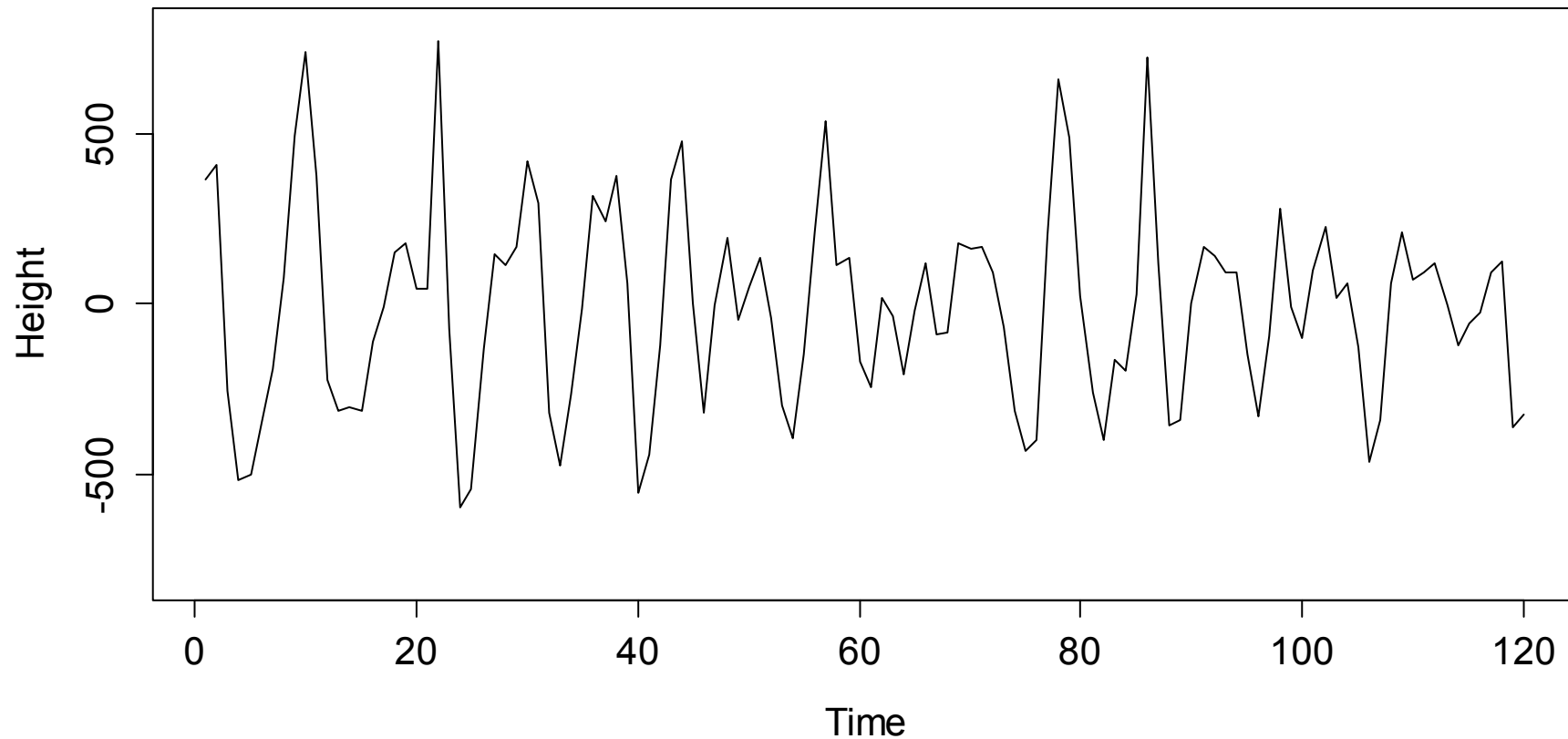


Applied Time Series Analysis

SS 2016 – Spectral Analysis

Wave Data

Time Series Plot of Wave Tank Data



Applied Time Series Analysis

SS 2016 – Spectral Analysis

Harmonic Oscillations

The most simple periodic functions are sine and cosine, which we will use as the basis of our decomposition analysis.

A harmonic oscillation has the following form:

$$y(t) = \alpha \cos(2\pi\nu t) + \beta \sin(2\pi\nu t)$$

For the derivation, [see the blackboard...](#)

- In discrete time, we have aliasing, i.e. some frequencies cannot be distinguished (→ see next slide).
- The periodic analysis is limited to frequencies between 0 and 0.5, i.e. things we observe at least twice in the series.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Regression Model for Decomposition

We can decompose any time series with a regression model containing sine and cosine terms at the Fourier frequencies.

$$X_t = \alpha_0 + \sum_{k=1}^m (\alpha_k \cos(2\pi\nu_k t) + \beta_k \sin(2\pi\nu_k t)) + E_t, \text{ where}$$
$$\nu_k = k / n \text{ for } k = 1, \dots, m \text{ with } m = \lfloor n / 2 \rfloor.$$

We are limited to this set of frequencies which provides an orthogonal fit. As we are spending n degrees of freedom on n we will have a perfect fit with zero residuals.

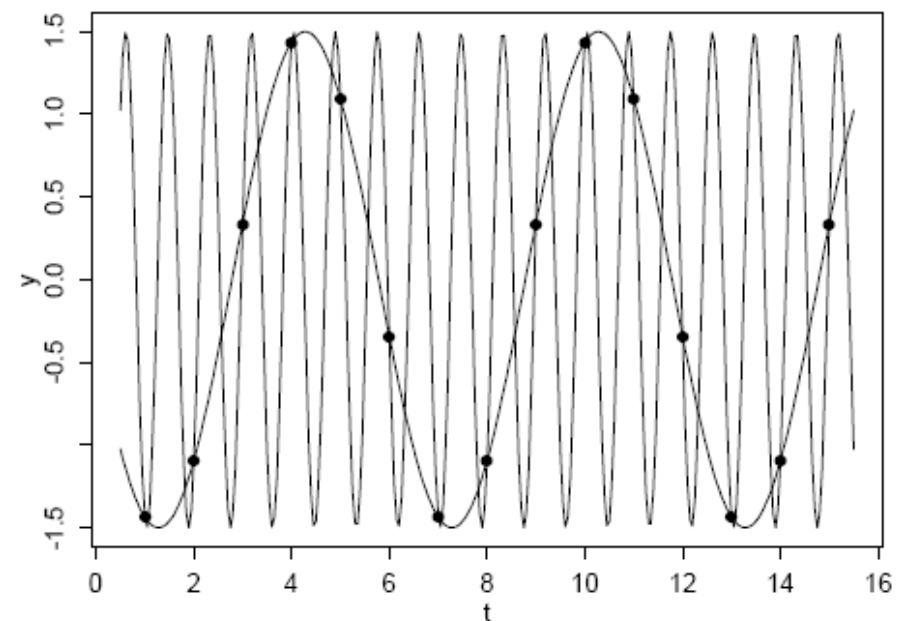
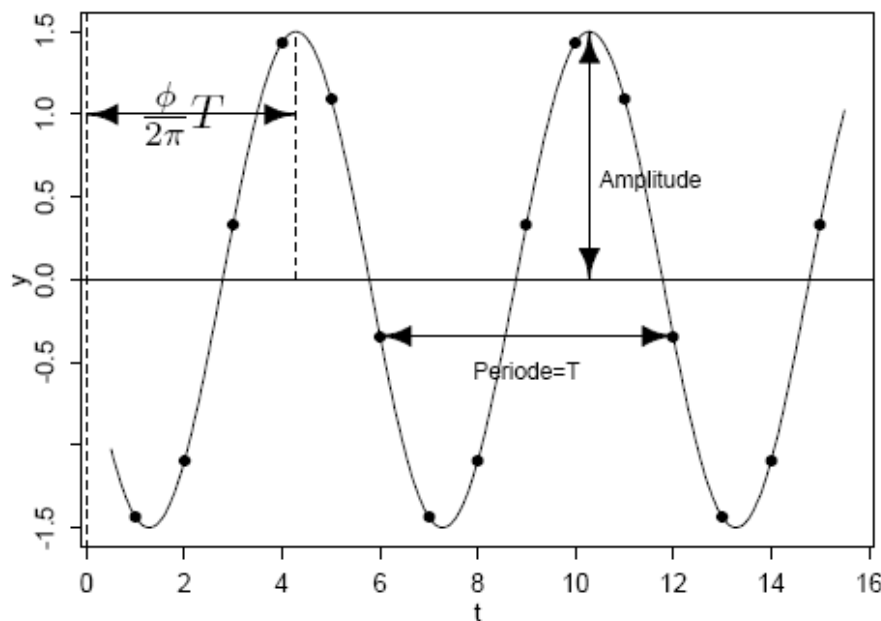
Note that the Fourier frequencies are not necessarily the correct frequencies, there may be *aliasing* and *leakage* problems.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Aliasing

The aliasing problem is based on the fact that if frequency ν fits the data, then frequencies $\nu + 1, \nu + 2, \dots$ will do so, too. In our spectral decomposition, the fastest frequency will be $\nu = 1/2$.



Applied Time Series Analysis

SS 2016 – Spectral Analysis

The Periodogram

If frequency ν_k is omitted from the decomposition model, the residual sum of squares increases by the amount of:

$$\frac{n}{2} \cdot \left(\hat{\alpha}_k^2 + \hat{\beta}_k^2 \right) = 2 \cdot I_n(\nu_k) \text{ for } k = 1, \dots, m$$

This value measures the importance of ν_k in the spectral decomposition and is the basis of the raw periodogram, which shows that importance for all Fourier frequencies.

Note: the period of frequency ν_k is $1/\nu_k = n/k$. Or we can also say that the respective peaks at this frequency repeat themselves for k times in the observed time series.

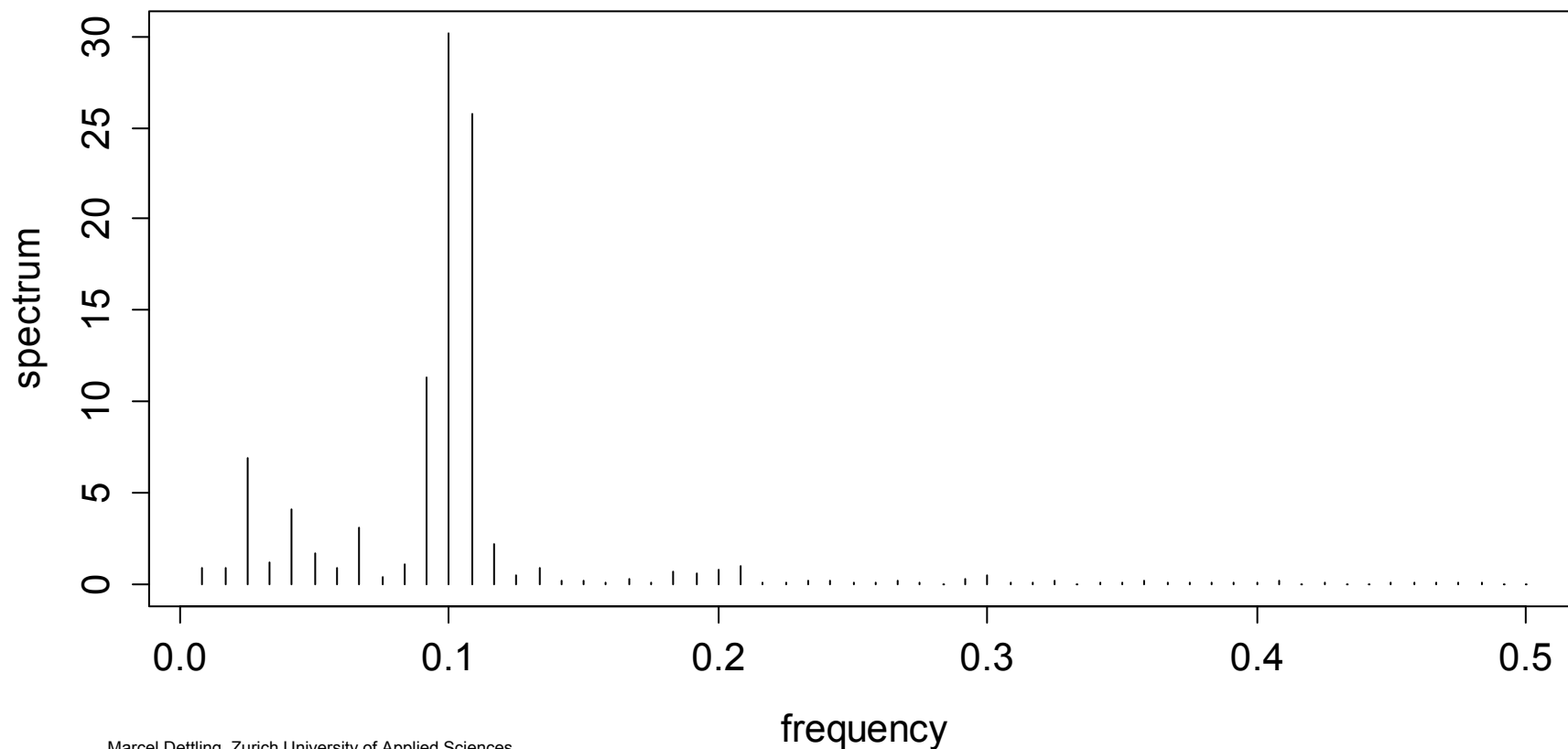
Applied Time Series Analysis

SS 2016 – Spectral Analysis

Raw Periodogram of Lynx Data

```
> spec.pgram(log(lynx), log="no", type="h")
```

Raw Periodogram of log(lynx)



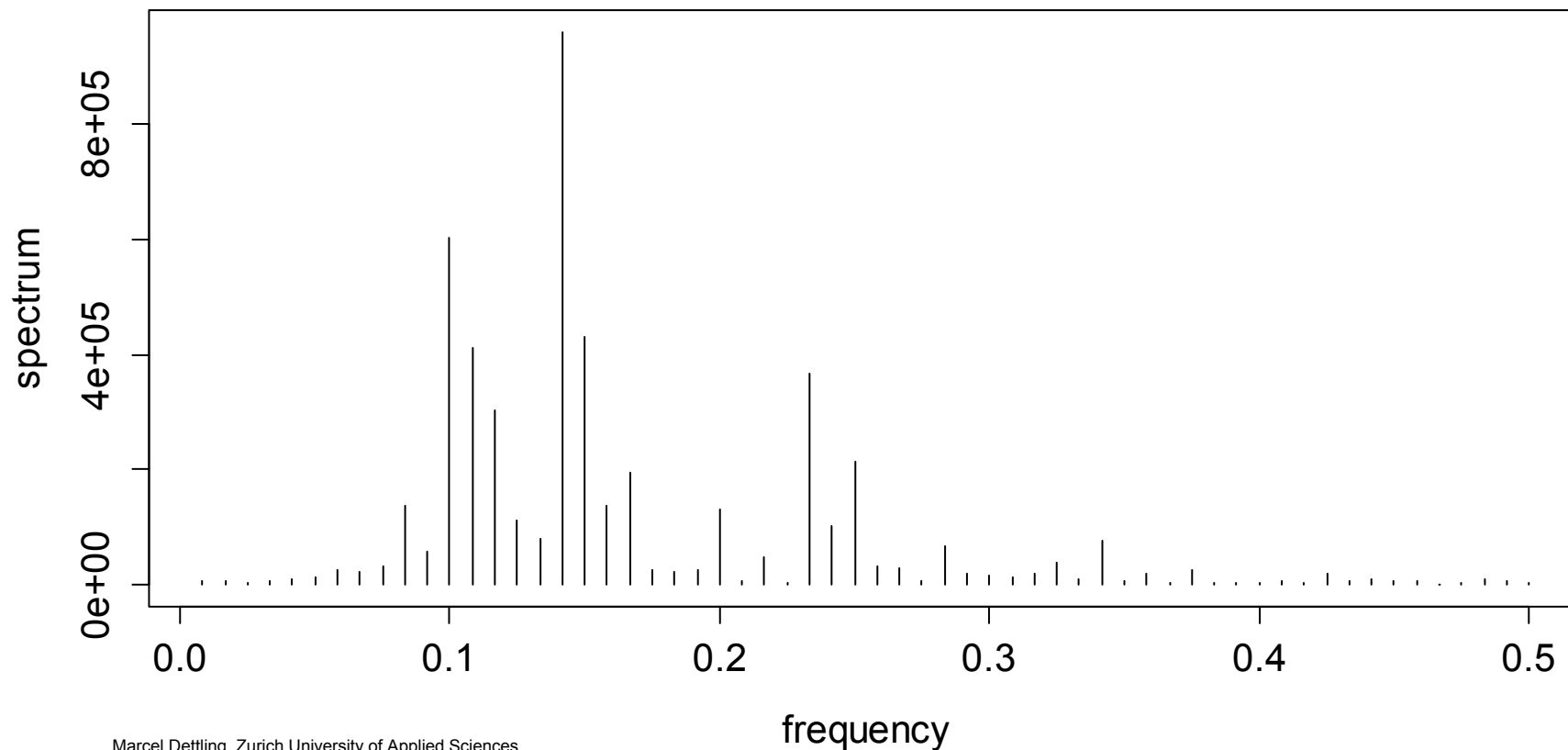
Applied Time Series Analysis

SS 2016 – Spectral Analysis

Raw Periodogram of Wave Data

```
> spec.pgram(log(lynx), log="no", type="h")
```

Raw Periodogram of Wave Tank Data

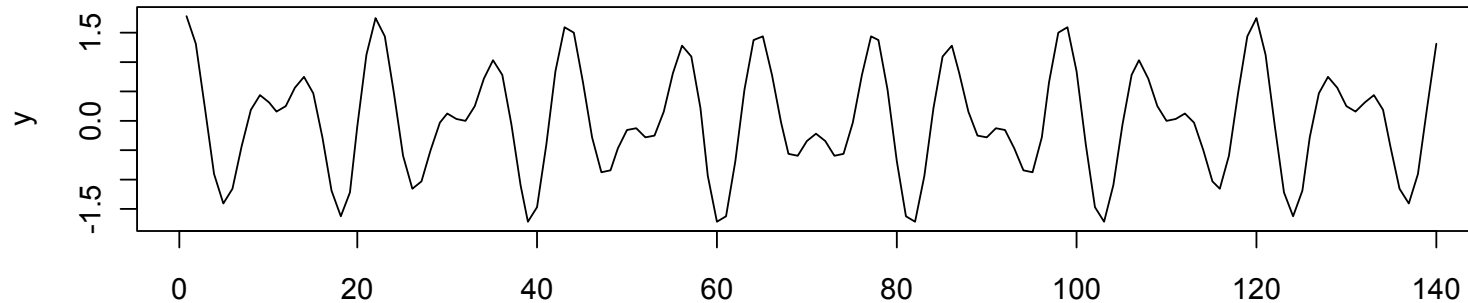


Applied Time Series Analysis

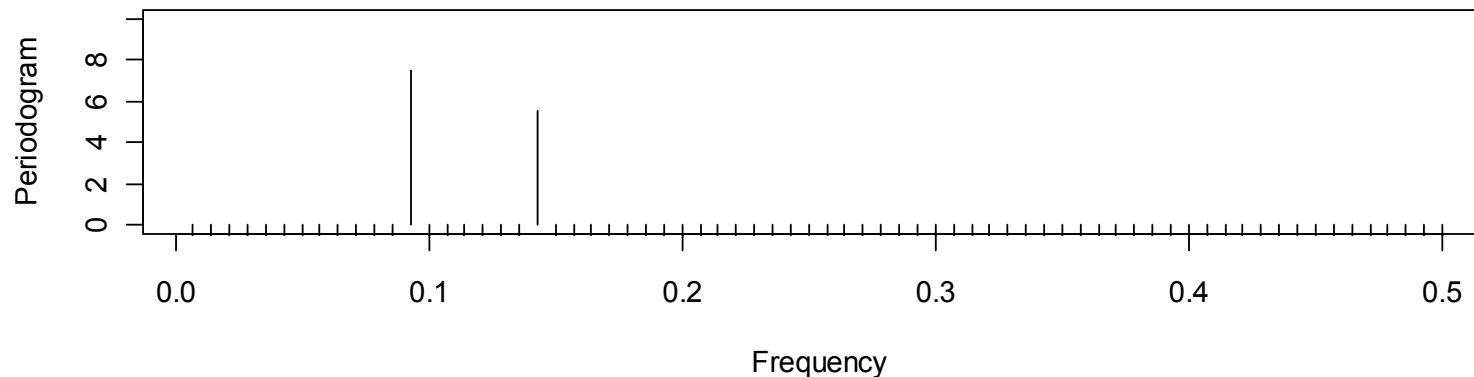
SS 2016 – Spectral Analysis

Periodogram of a Simulated Series

Simulated Series



$$X_t = \cos\left(\frac{2\pi \cdot 13 \cdot t}{140}\right) + 0.8 \cdot \cos\left(\frac{2\pi \cdot 20 \cdot t}{140}\right)$$

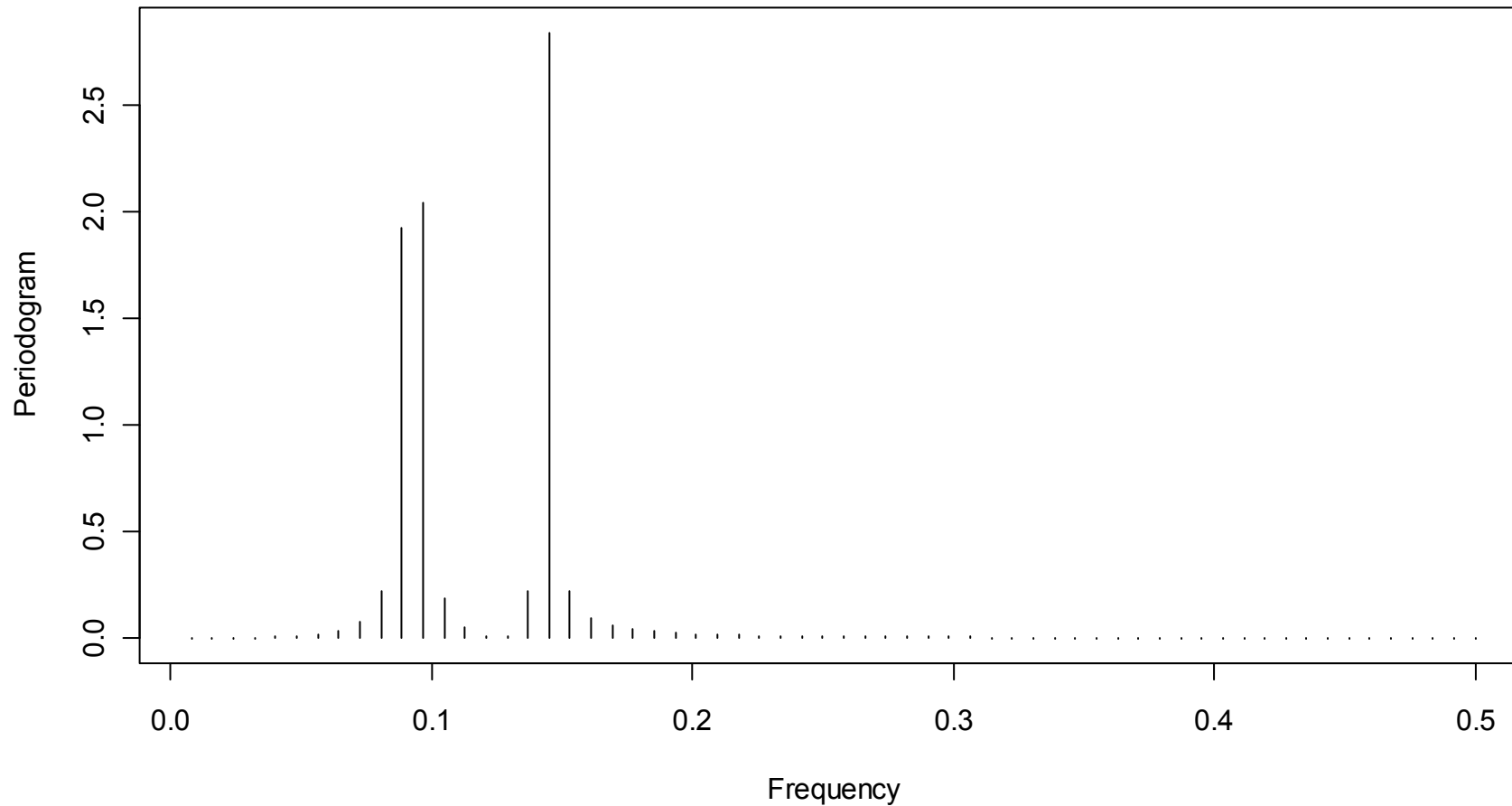


Applied Time Series Analysis

SS 2016 – Spectral Analysis

Periodogram of the Shortened Series

Periodogram of the Shortened Series



Applied Time Series Analysis

SS 2016 – Spectral Analysis

The Spectrum

The spectrum of a time series process is a function telling us the importance of particular frequencies to the variation of the series.

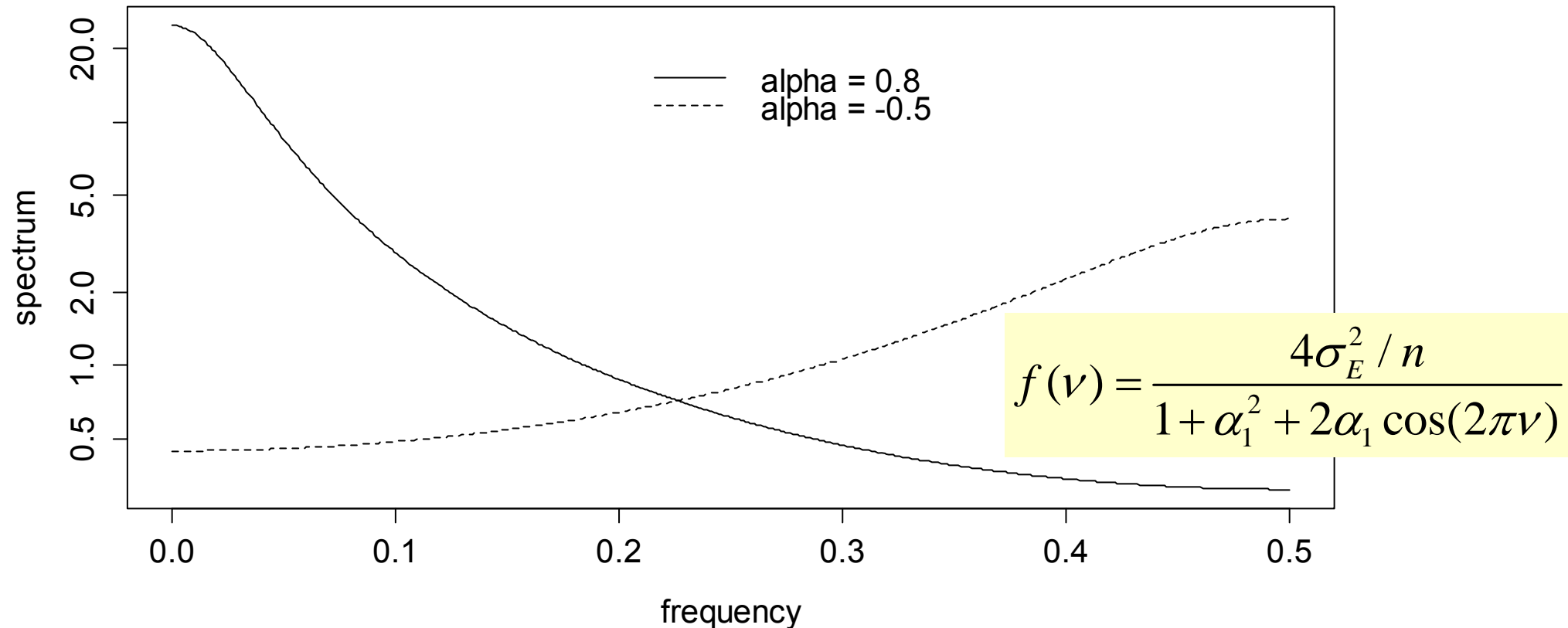
- Usually, time series processes have a continuous frequency spectrum and do not only consist of a few single frequencies.
- For ARMA(p,q) process, the spectrum is continuous and there are explicit formulae, depending on the model parameters.
- Subsequently, we will pursue the difficult task of estimating the spectrum, based on the raw periodogram.
- There is a 1:1 correspondence between the autocovariance function of a time series process and its spectrum.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Example: Spectrum of AR(1)-Processes

Spectrum of AR(1)-Processes

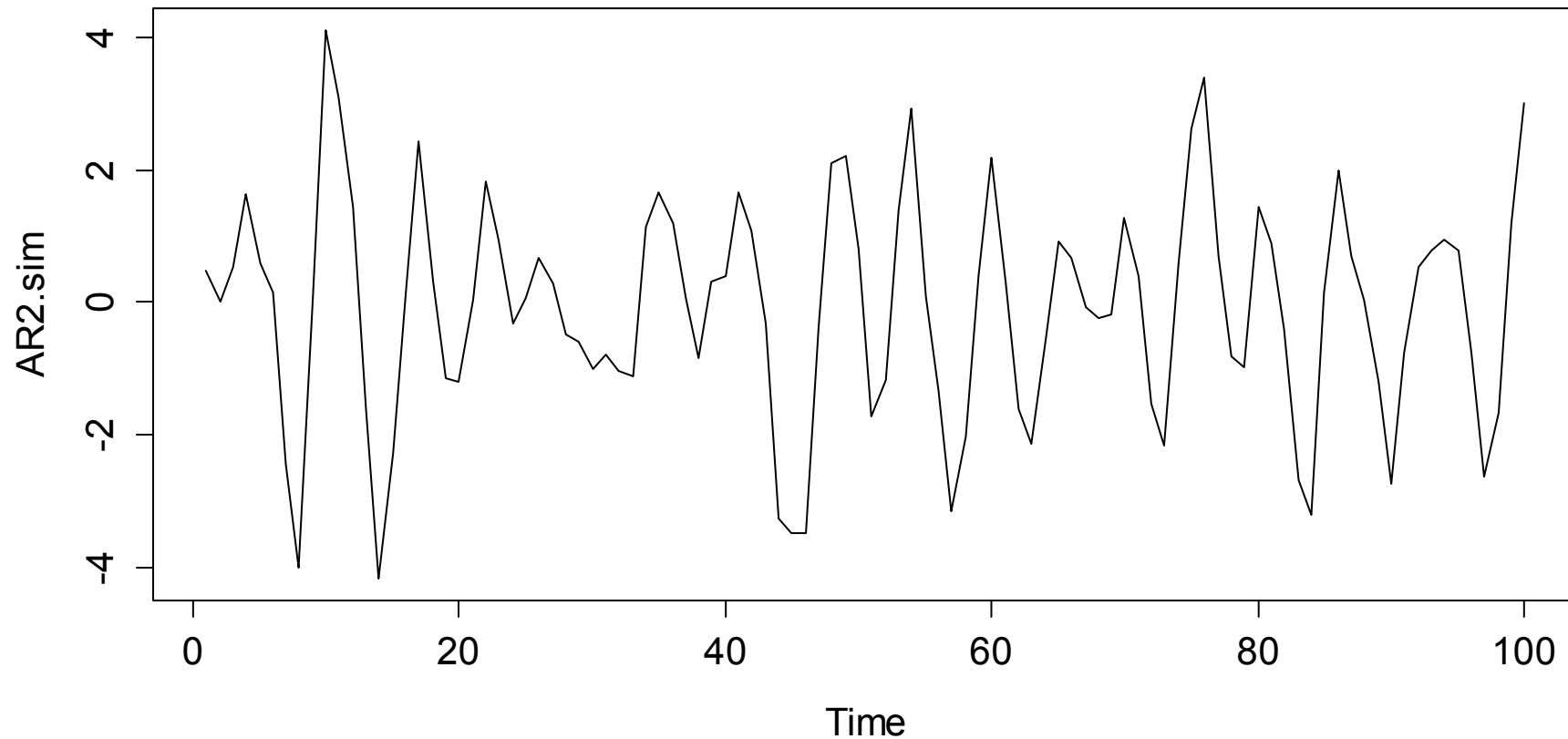


Applied Time Series Analysis

SS 2016 – Spectral Analysis

Example: Simulated AR(2)

Simulated AR(2)

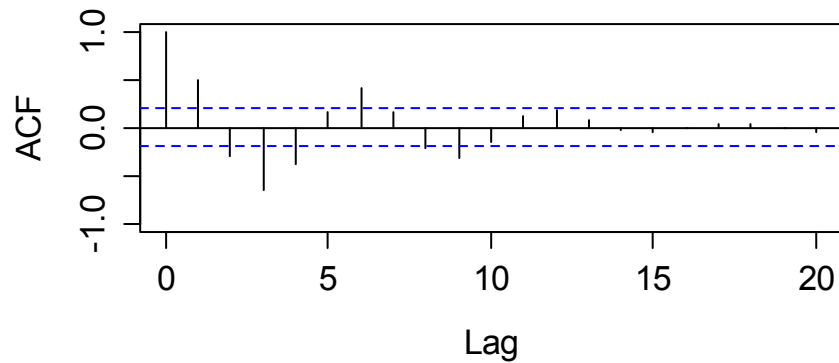


Applied Time Series Analysis

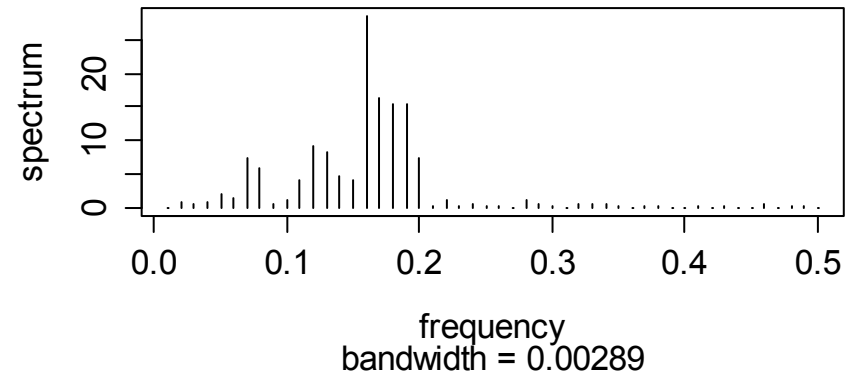
SS 2016 – Spectral Analysis

Example: Simulated AR(2)

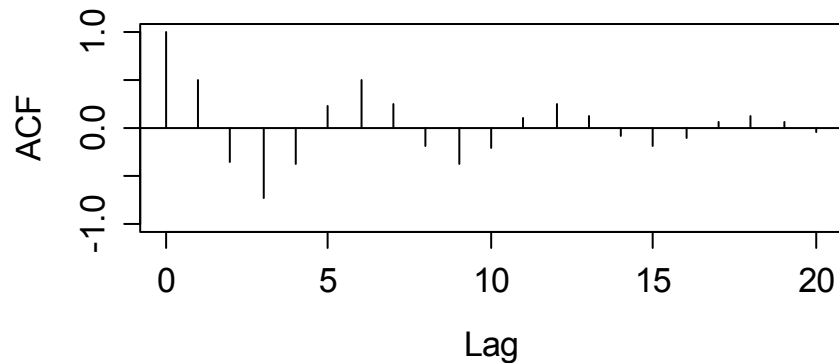
Estimated ACF



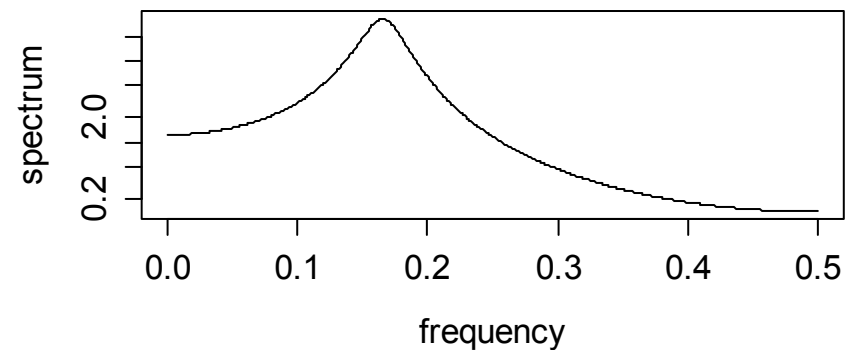
Periodogram



True ACF



Spectrum



Applied Time Series Analysis

SS 2016 – Spectral Analysis

Estimating the Spectrum

Our goal is estimating the spectrum of e.g. an ARMA(p,q). There is quite a discrepancy between the discrete raw periodogram and the continuous spectrum. The following issues arise:

- The periodogram is noisy, and there may be leakage.
- The periodogram value at frequency ν_k is an unbiased estimator of the spectrum value $f(\nu_k)$. However, it is inconsistent due to its variability, owing to the fact that we estimate n periodogram values from n observations.
- Theory tells us that ν_k and ν_j for $k \neq j$ are asymptotically independent. This will be exploited to improve estimation.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Smoothing the Periodogram

Due to asymptotic independence and unbiasedness and the smooth nature of the spectrum, smoothing approaches help in achieving qualitatively good, consistent spectral estimates.

Running Mean Estimator:

$$f(\hat{\nu}_j) = \frac{1}{2L+1} \sum_{k=-L}^{+L} I_n(\nu_{j+k})$$

The choice of the bandwidth $B = 2L/n$ is crucial. If chosen appropriately, the spectral estimates at the Fourier frequencies will be consistent.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

Daniell Smoother

An option for improving the Running Mean is to use weights. They need to be symmetric, decaying and sum up to one. The formula:

Weighted Running Mean

$$f(\hat{v}_j) = \frac{1}{2L+1} \sum_{k=-L}^{+L} w_k I_n(v_{j+k})$$

The challenge lies in the choice of the weights. The **Daniell Smoother** is a Weighted Running Mean with $w_k = 1/2L$ for $k < L$ and $w_k = 1/4L$ for $k = L$. This is the default in the R function `spec.pgram()` if argument `spans=2L+1`, see next slide. Also this estimate is consistent.

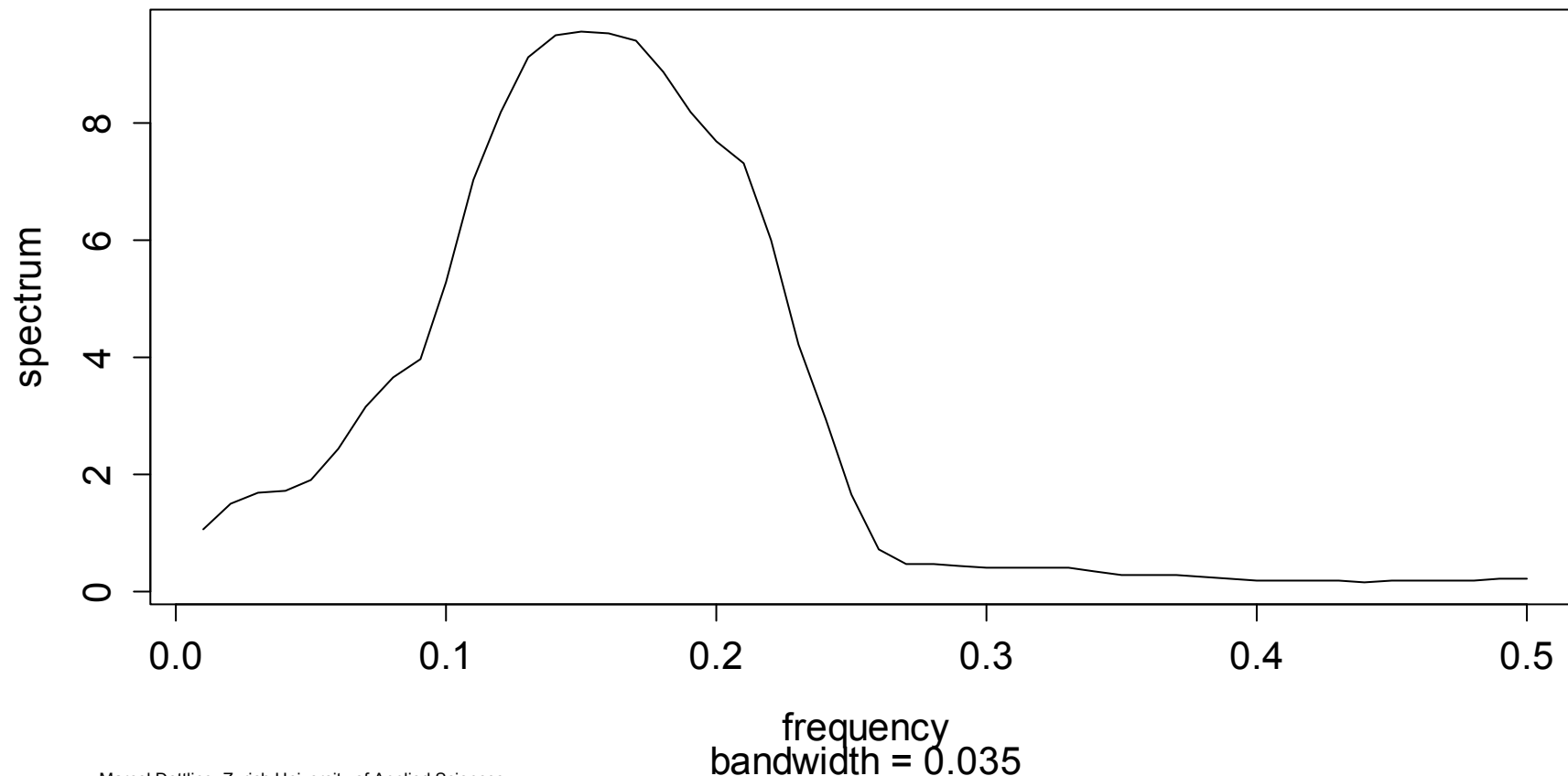
Applied Time Series Analysis

SS 2016 – Spectral Analysis

Daniel Smoother for Simulated AR(2)

```
> spec.pgram(AR2.sim, spans= 13, log="no")
```

Daniel Smoother, L=6



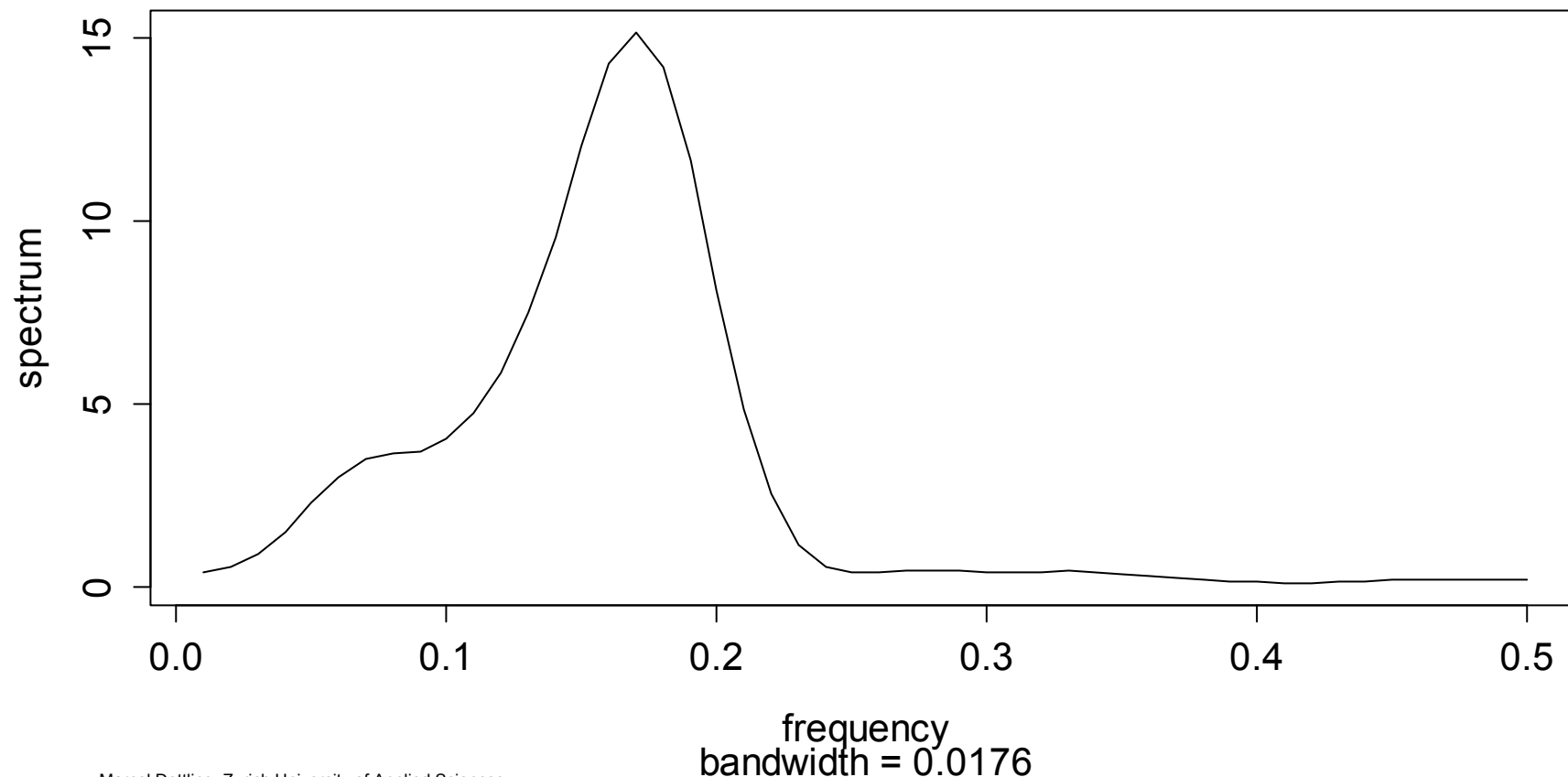
Applied Time Series Analysis

SS 2016 – Spectral Analysis

Double Daniell Smoother

```
> spec.pgram(AR2.sim, spans=c(5,5), log="no")
```

Double Smoother, L=2



Applied Time Series Analysis

SS 2016 – Spectral Analysis

Tapering

Tapering is a technique to further improve spectral estimates. The R function `spec.pgram()` applies it by default, and unless you know much better, you must keep it that way.

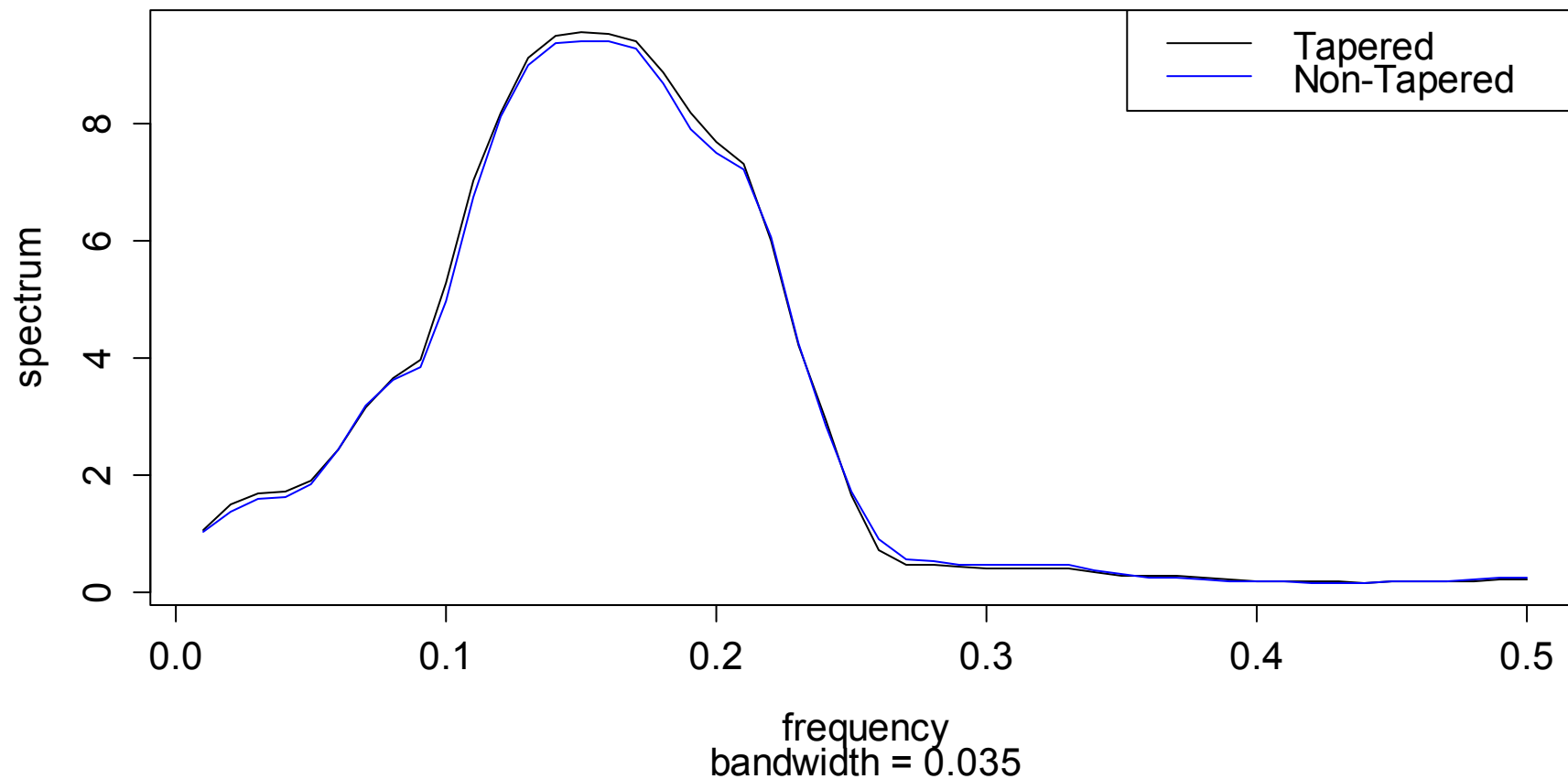
- In spectral analysis, a time series is seen as a finite sample with a rectangular window of an infinitely long process.
- This rectangular window distorts spectral estimation in several ways, among others also via the effect of leaking.
- Tapering means that the ends of the time series are altered to mitigate these effects, i.e. they gradually taper down towards zero.

Applied Time Series Analysis

SS 2016 – Spectral Analysis

The Effect of Tapering

The Effect of Tapering



Applied Time Series Analysis

SS 2016 – Spectral Analysis

Model Based Spectral Estimation

The fundamental idea for this type of spectral estimate is to fit an $AR(p)$ model to an observed series and then derive the theoretical spectrum by plugging-in the estimated coefficients

- This approach is not related to the periodogram based smoothing approaches presented before.
- By nature, it always provides a smooth spectral estimate.
- There is an excellent implementation in R: `spec.ar()`.

Please note that spectral estimates are usually plotted on the dB-scale which is logarithmic. Also, the R function provides a confidence interval.

Applied Time Series Analysis

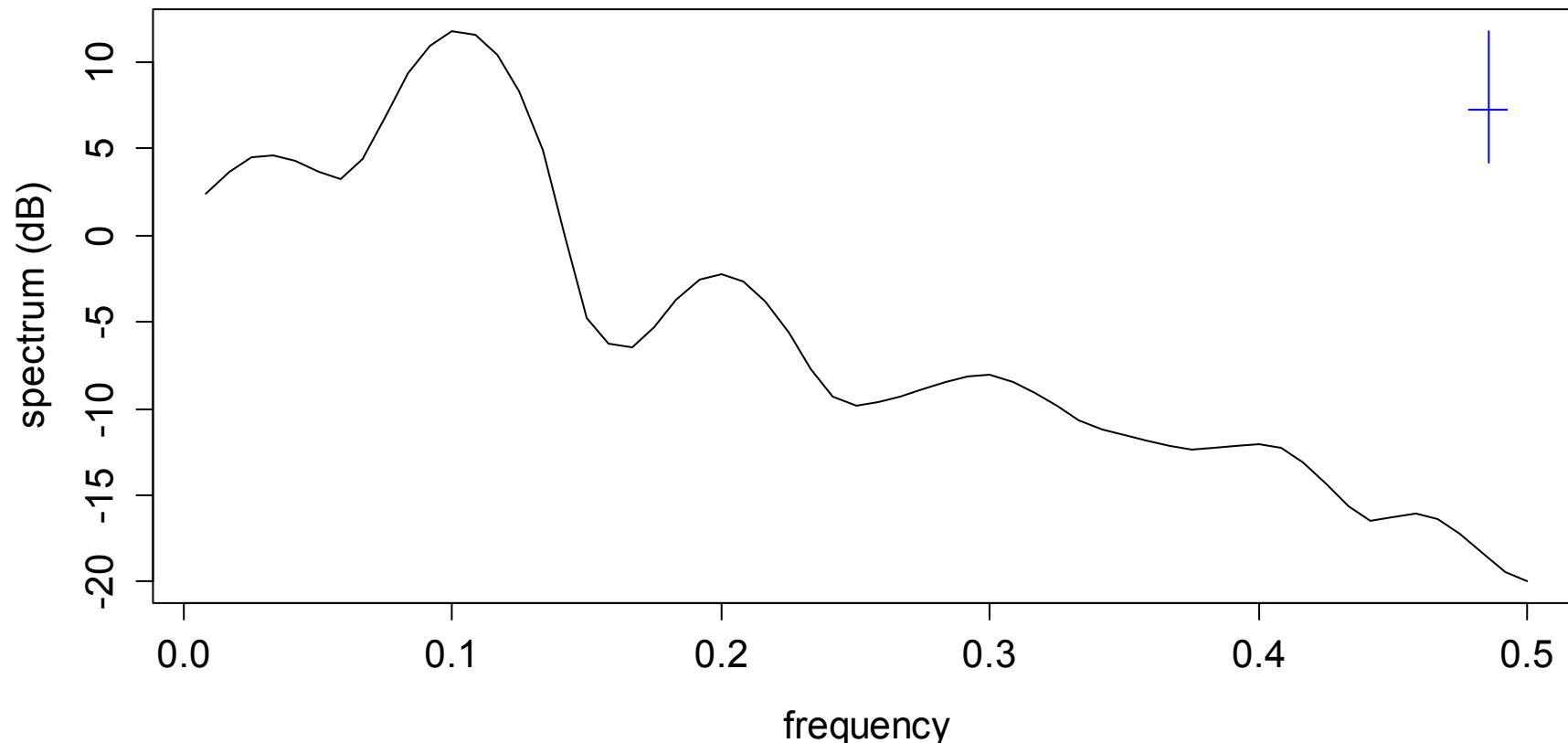
SS 2016 – Spectral Analysis

Model Choice: Daniell Smoother

```
> spec.pgram(log(lynx), spans=c(5,5), log="dB")
```

Series: log(lynx)

Smoothed Periodogram



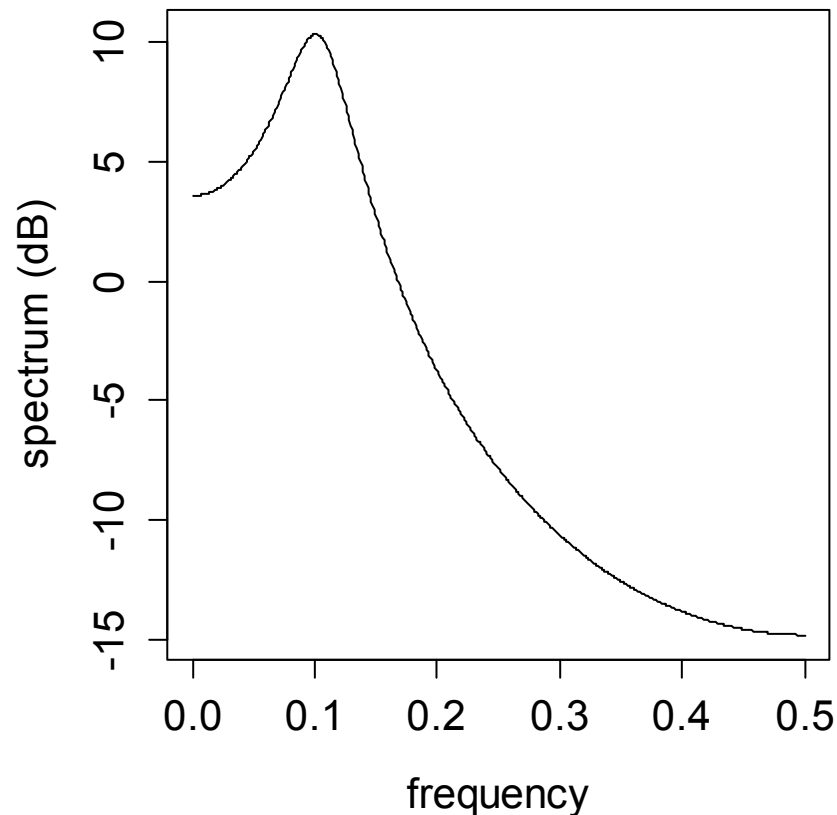
Applied Time Series Analysis

SS 2016 – Spectral Analysis

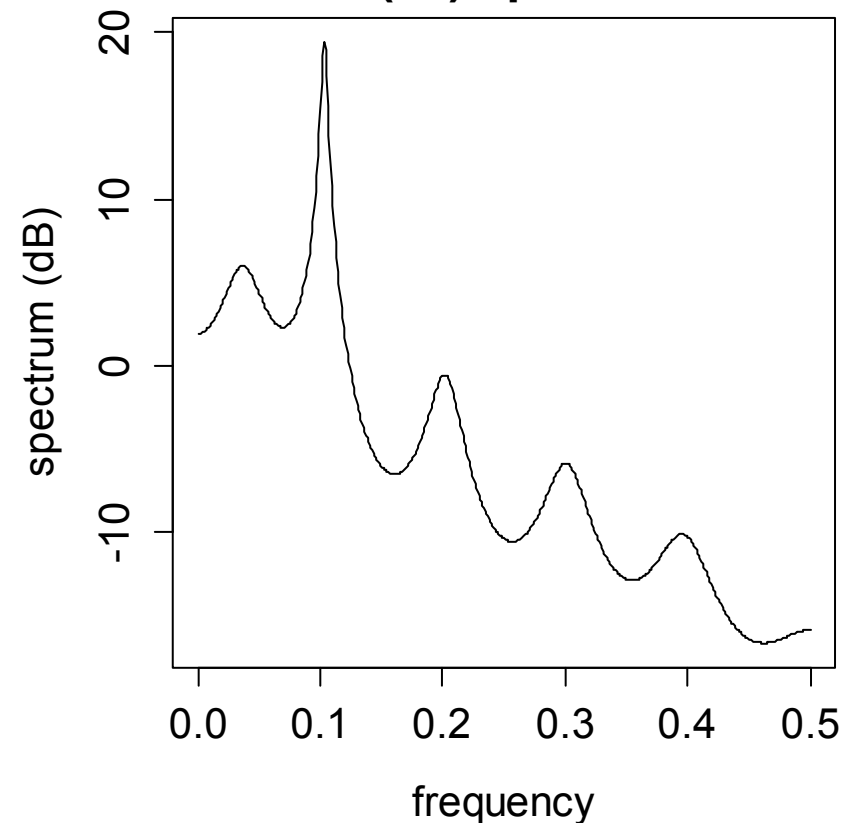
Model Choice: Plug-In

```
> spec.ar(log(lynx), order=11, log="dB")
```

Series: log(lynx)
AR (2) spectrum



Series: log(lynx)
AR (11) spectrum

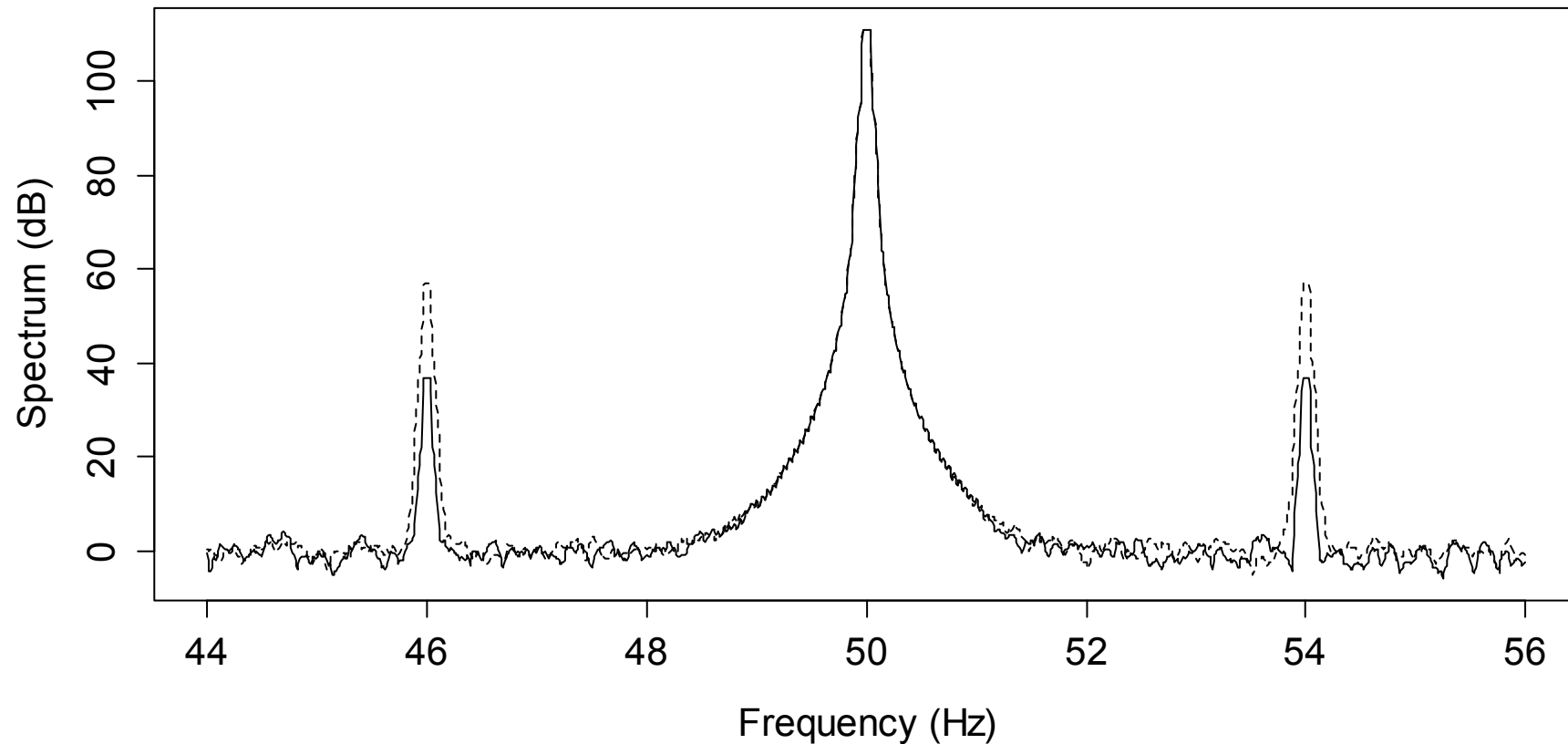


Applied Time Series Analysis

SS 2016 – Spectral Analysis

Real World Example

Broken Motor Example



Applied Time Series Analysis

SS 2016 – State Space Models

State Space Models

Basic idea: There is a stochastic process/time series X_t which we cannot directly observe, but only under the addition of some measurement noise.

Thus: We observe the time series $Y_t = X_t + V_t$,
with iid measurement errors $V_t \sim N(0, \sigma_V^2)$

Example: $X_t = \#$ of fish in a lake
 $Y_t = \#$ estimated number of fish from a sample

Other:

- Dynamic linear modeling
- Regression with time-varying coefficients

Applied Time Series Analysis

SS 2016 – State Space Models

State Space Formulation

State space models are always built on two different equations, one of which aims for the process, and the other for the measurement noise:

State Equation: $X_t = G_t X_{t-1} + W_t$, where $W_t \sim N(0, w_t)$

Observation Equation: $Y_t = F_t X_t + V_t$, where $V_t \sim N(0, v_t)$

All matrices in this model, i.e. G_t, F_t, w_t, v_t can be time-varying. However, often they are time-constant, if anything, then F_t is adapting over time.

Note: such models are usually estimated with the Kalman filter.

Applied Time Series Analysis

SS 2016 – State Space Models

AR(1) with Measurement Noise

We assume that the true underlying process is an AR(1), i.e.

$$X_t = \alpha_1 X_{t-1} + W_t,$$

where

$W_t \sim N(0, \sigma_W^2)$ are i.i.d. innovations, „process noise“.

In practice, we only observe y_t , as realizations of the process

$$Y_t = X_t + V_t, \text{ with } V_t \sim N(0, \sigma_V^2), \text{ i.i.d.}$$

and additionally, the V_t are independent of X_s, W_s for all s, t , thus they are independent „observation white noise“.

Applied Time Series Analysis

SS 2016 – State Space Models

More Terminology

We call

$$X_t = \alpha_1 X_{t-1} + W_t$$

the „state equation“, and

$$Y_t = X_t + V_t$$

the „observation equation“.

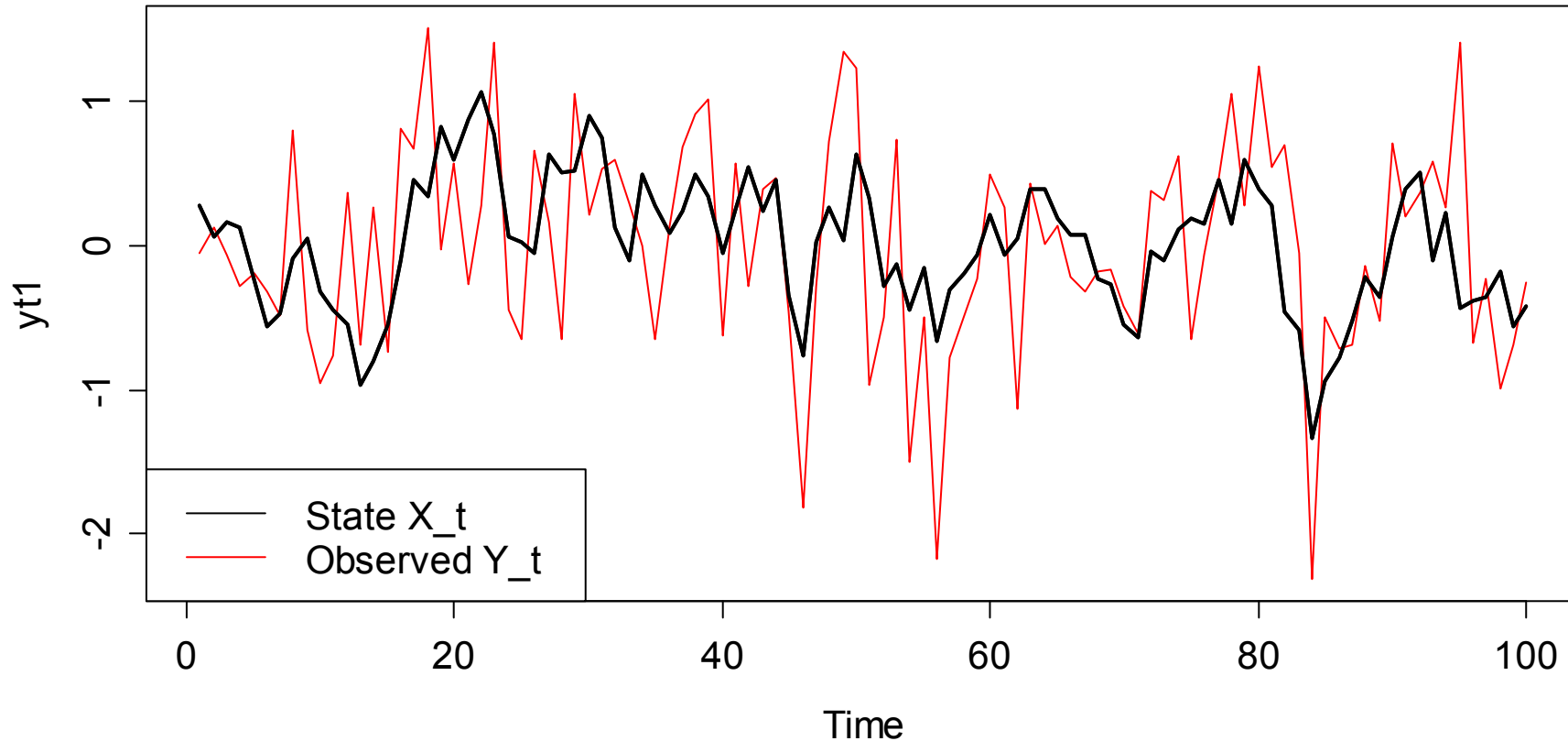
On top of that, we remember once again that the „process noise“ W_t is an innovation that affects all future values X_{t+k} and thus also Y_{t+k} , whereas V_t only influences the current observation Y_t , but no future ones.

Applied Time Series Analysis

SS 2016 – State Space Models

AR(1)-Example with $\alpha=0.7$

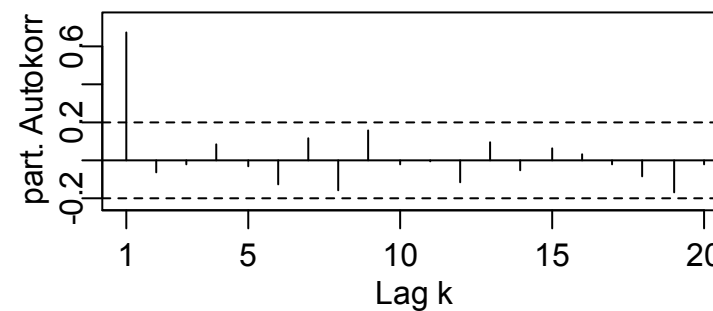
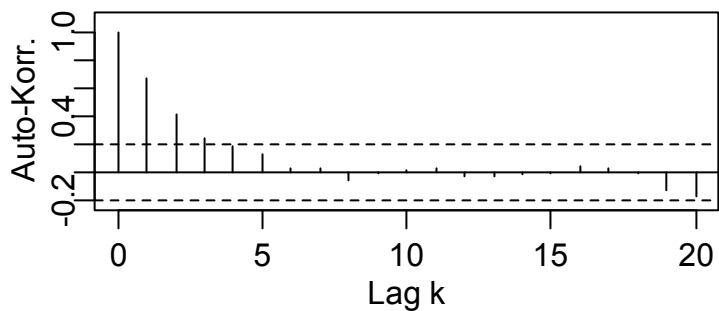
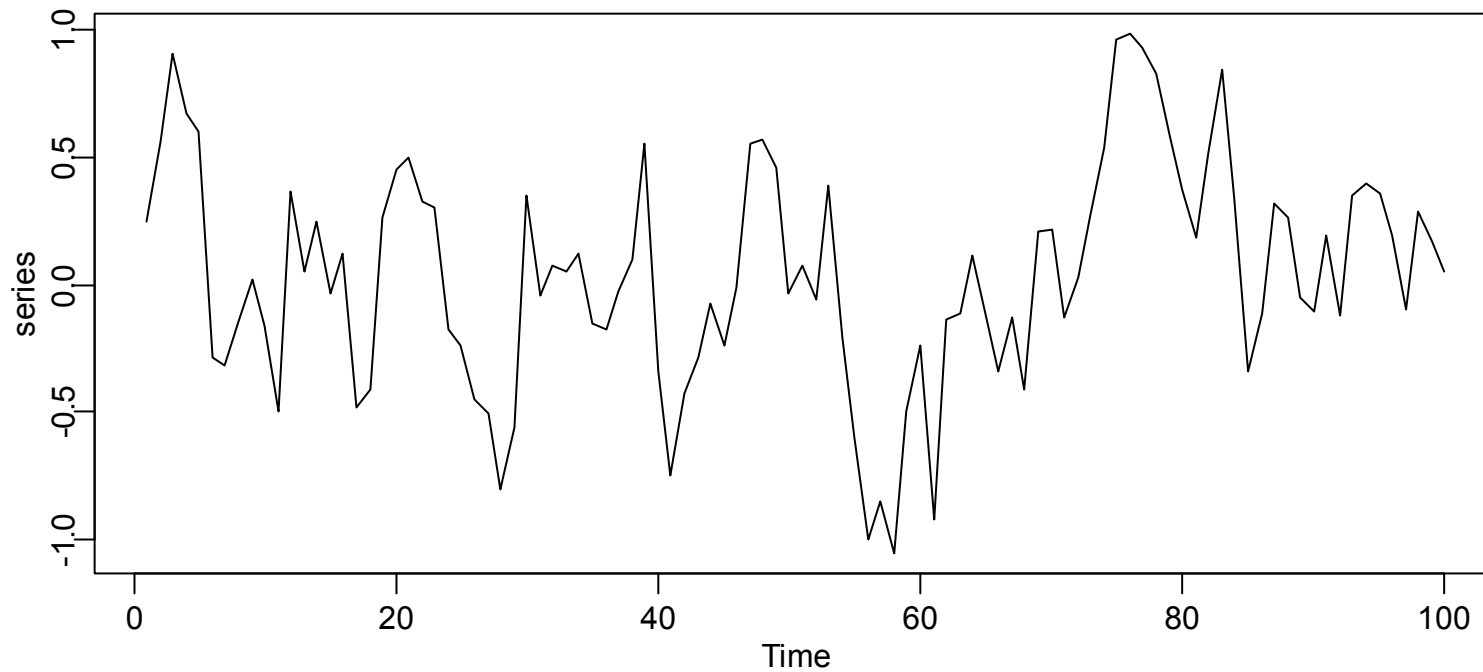
AR(1) Simulation Example



Applied Time Series Analysis

SS 2016 – State Space Models

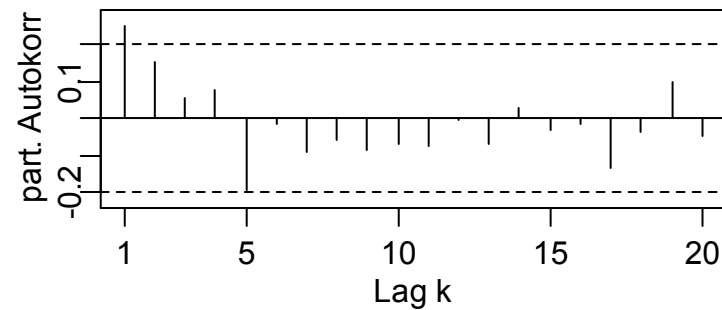
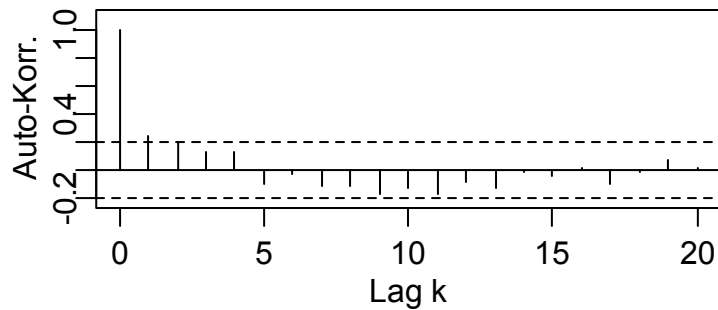
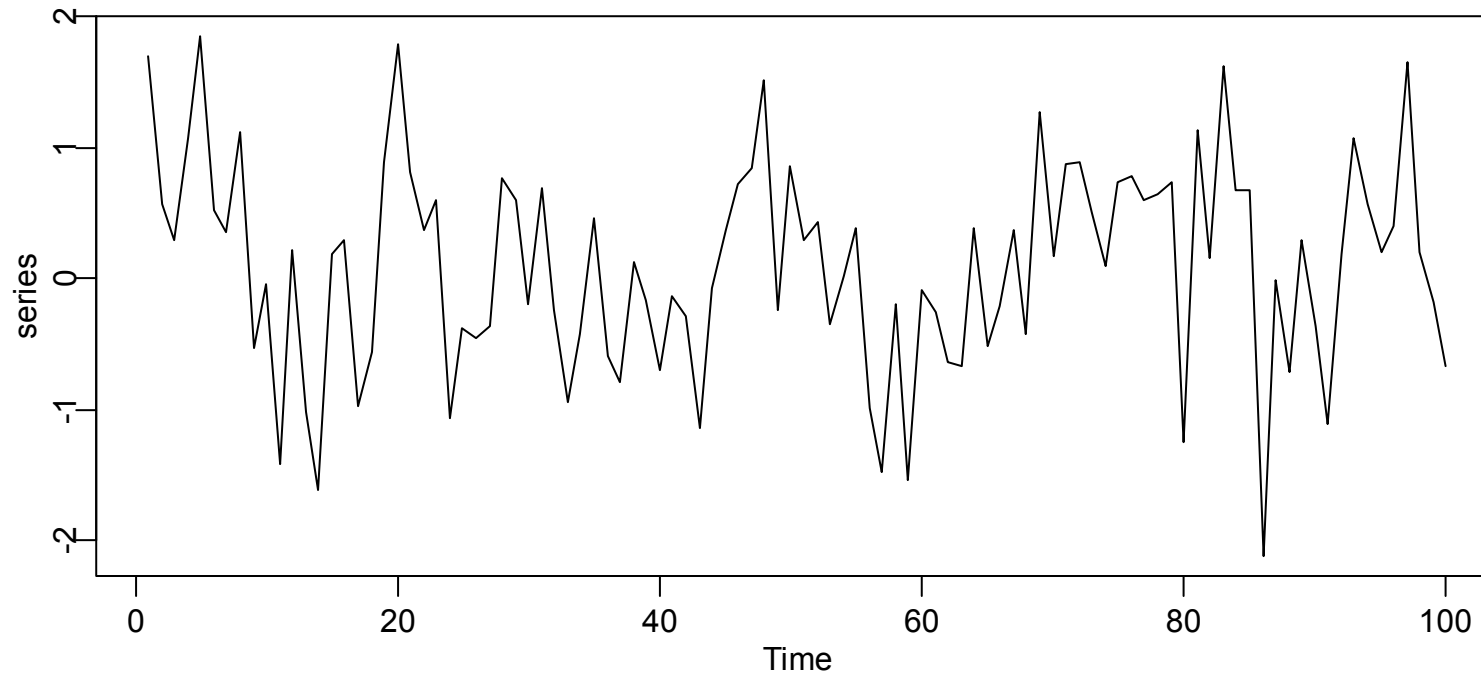
ACF/PACF of X_t



Applied Time Series Analysis

SS 2016 – State Space Models

ACF/PACF of Y_t



Applied Time Series Analysis

SS 2016 – State Space Models

What is the goal?

The goal of State Space Modeling/Kalman Filtering is:

To uncover the „de-noised“ process X_t from the observed process Y_t .

- The algorithm of Kalman Filtering works with non-stationary time series, too.
- The algorithm is based on a maximum-likelihood-principle where one assume normal distortions.
- There are extensions to multi-dimensional state space models. **See blackboard** for an example how the state space formulation of an AR(2) is set up .

Applied Time Series Analysis

SS 2016 – State Space Models

State Space and Kalman Filtering in R

```
## Load the package for Kalman filtering
library(sspir)

## State Space Formulation
ssf <- SS(y = as.matrix(obs),
          Fmat = function(tt,x,phi) {return(matrix(1))},
          Gmat = function(tt,x,phi) {return(matrix(0.7))},
          Vmat = function(tt,x,phi) {return(matrix(0.5))},
          Wmat = function(tt,x,phi) {return(matrix(0.1))},
          m0 = matrix(0), C0 = matrix(0.1))

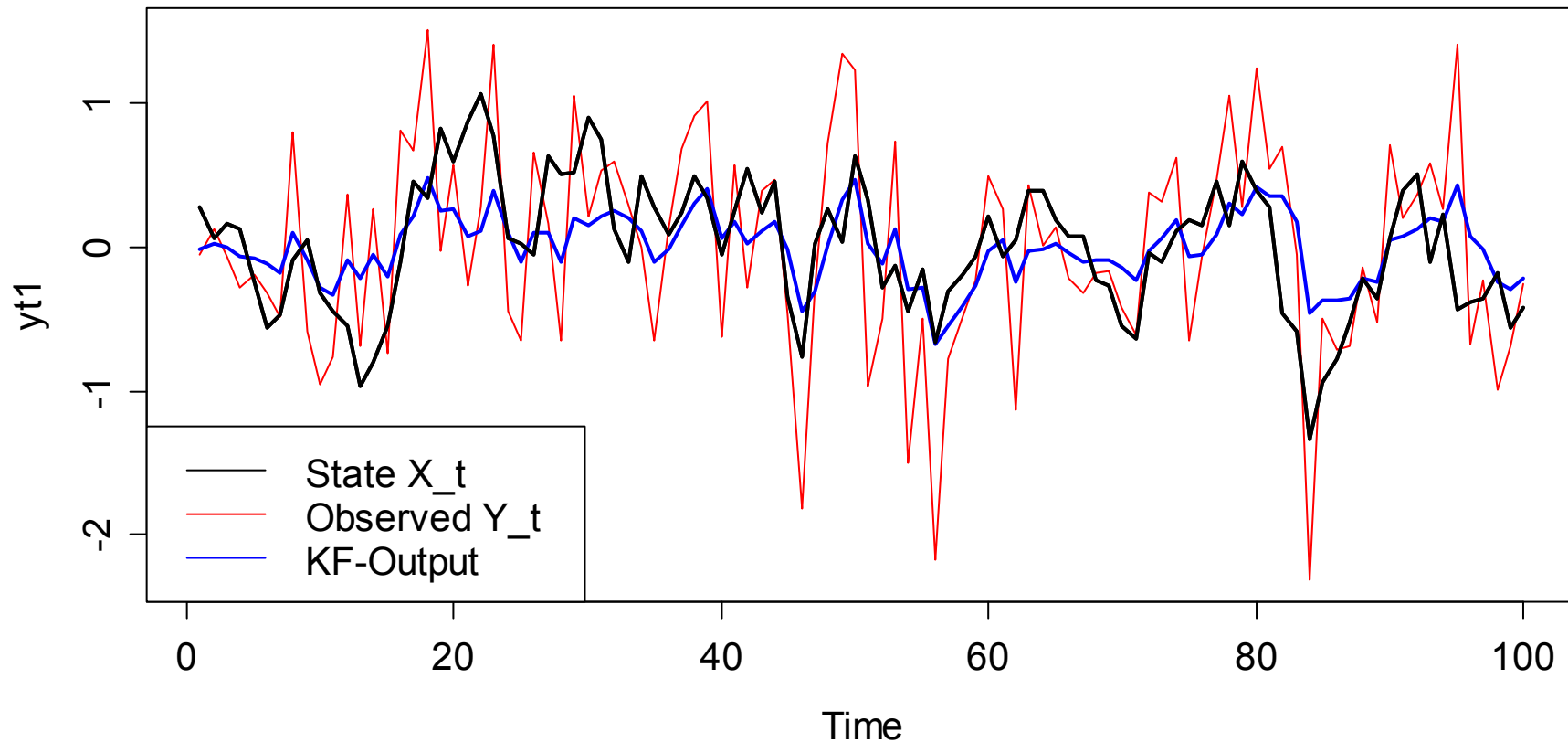
## Kalman Filtering
fit <- kfilter(ssf)
```


Applied Time Series Analysis

SS 2016 – State Space Models

Kalman Filter Solution

AR(1) Simulation Example with Kalman Filter Output



Applied Time Series Analysis

SS 2016 – State Space Models

State Space Formulation of an AR(2)

→ see blackboard...

Applied Time Series Analysis

SS 2016 – State Space Models

Dynamic Linear Models

In particular: *regression models with time-varying coefficients*

Example: the sales of a housing company depend on the general level of sales in that area at time t , and on the pricing policy at time t .

$$S_t = L_t + \beta_t P_t + V_t$$

This is a regression model with price as the predictor, and the general sales level as the intercept. They are time-varying:

$$L_t = L_{t-1} + \Delta L_t \quad \beta_t = \beta_{t-1} + \Delta \beta_t$$

Here, $V_t, \Delta L_t, \Delta \beta_t$ are random elements, noise & perturbations

Applied Time Series Analysis

SS 2016 – State Space Models

Simulation Example

→ see blackboard...

Applied Time Series Analysis

SS 2016 – State Space Models

Kalman Filtering for Regression

```
#### State Space Formulation
ssf <- SS(y=y.mat, x=x.mat,
         Fmat=function(tt,x,phi) return(matrix(c(x[tt,1],x[tt,2]),2,1)),
         Gmat=function(tt,x,phi) return(diag(2)),
         Wmat=function(tt,x,phi) return(0.1*diag(2)),
         Vmat=function(tt,x,phi) return(matrix(1)),
         m0=matrix(c(5,3),1,2),C0=10*diag(2))

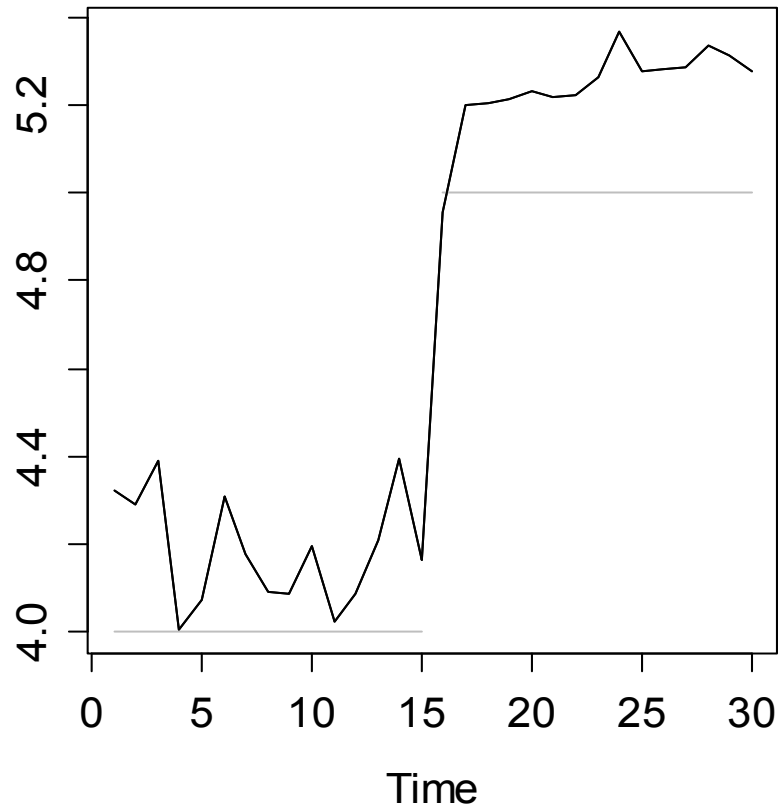
## Kalman-Filtering
fit <- kfilter(ssf)
plot(fit$m[,1], type="l", xlab="Time", ylab="Intercept")
plot(fit$m[,2], type="l", xlab="Time", ylab="Slope")
```

Applied Time Series Analysis

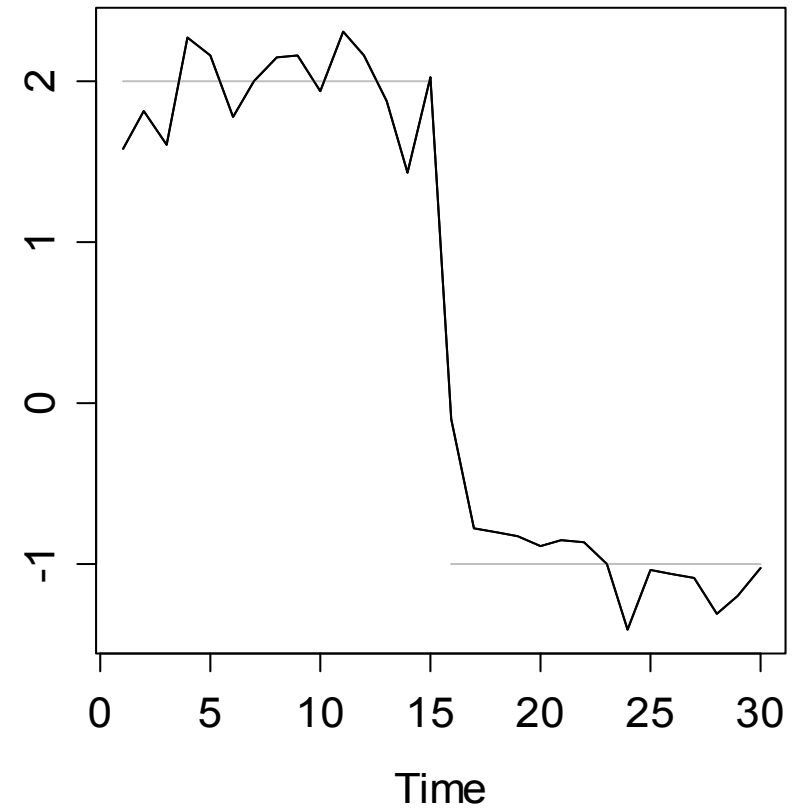
SS 2016 – State Space Models

Kalman Filter Solution

Kalman Filtered Intercept



Kalman Filtered Slope



Applied Time Series Analysis

SS 2016 – State Space Models

Summary of Kalman Filtering

Summary:

- 1) The Kalman Filter is a recursive algorithm
- 2) It relies on an update idea, i.e. we update the forecast $\hat{X}_{t+1,t}$ with the difference $(y_{t+1} - \hat{Y}_{t+1,t})$.
- 3) The weight of the update is determined by the relation between the process variance σ_W^2 and the measurement noise σ_V^2 .
- 4) This relies on the knowledge of $G, F, \sigma_W^2, \sigma_V^2$. In R we have procedures where everything is estimated simultaneously.

Applied Time Series Analysis

SS 2016 – State Space Models

Additional Remarks

- 1) For the recursive approach of Kalman filtering, initial values are necessary. Their choice is not crucial, their influence cancels out rapidly.
- 2) The procedures yield forecast and filter intervals:
$$\hat{X}_{t+1,t} \pm 1.96 \cdot \sqrt{R_{t+1,t}} \quad \text{and} \quad \hat{X}_{t+1,t+1} \pm 1.96 \cdot \sqrt{R_{t+1,t+1}}$$
- 3) State space models are a very rich class. Every ARIMA(p,d,q) can be written in state space form, and the Kalman filter can be used for estimating the coefficients.