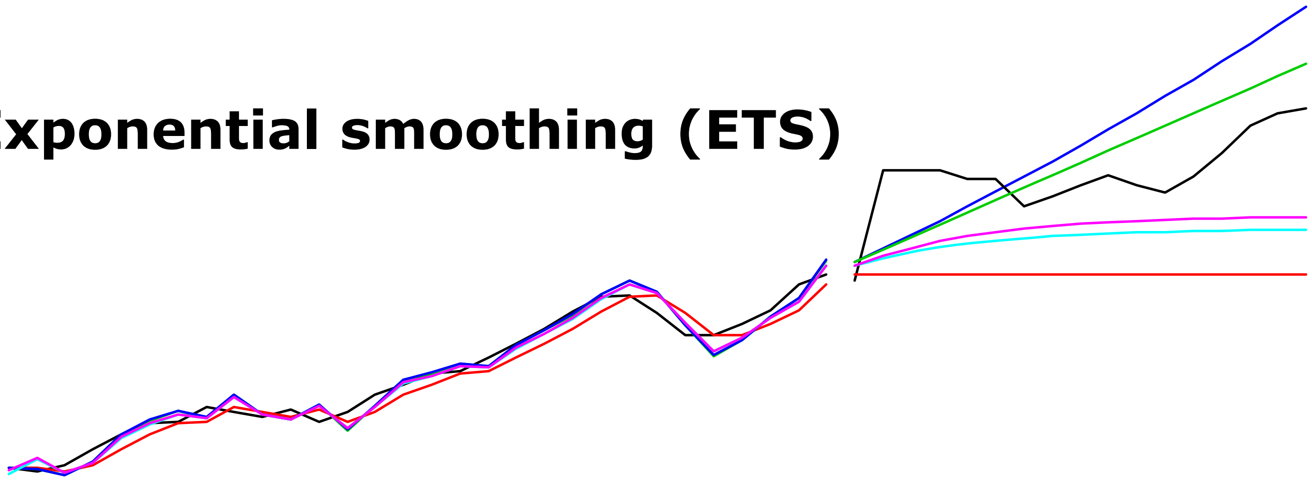
Hochschule Esslingen
University of Applied Sciences

04_Advanced Forecasting

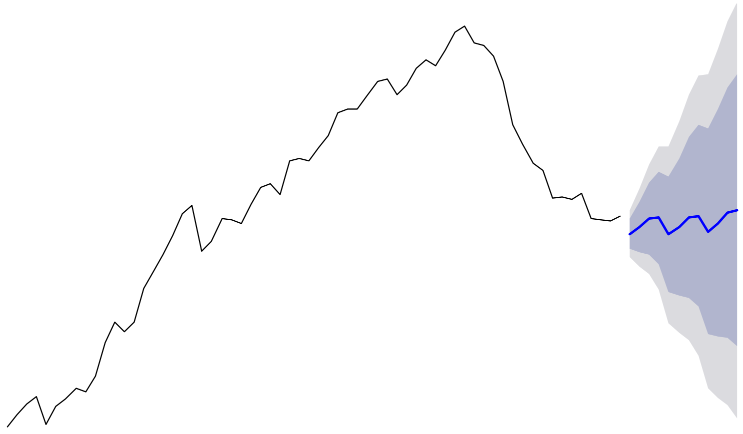
Lecture: Intelligent Data Analytics

Overview

1. **Exponential smoothing (ETS)**



2. **Autoregressive integrated moving average (ARIMA) models**



1. **Exponential smoothing (ETS)**
   - Motivation
   - Simple exponential smoothing (SES)
   - Weighted average form
   - Initialization
   - Optimization
   - Holt's linear trend method
   - Exponential trend method

1. **Autoregressive integrated moving average (ARIMA) models**

Motivation

Hochschule Esslingen
University of Applied Sciences

- The **naïve method** assumes that only the most current observation is important for the future.
- You can think of it as a weighted average where all the weight is given to the last observation.

$$\hat{y}_{T+h|T} = y_T$$

- The **average method** assumes that all observations are equally important for the future.
- Therefore, it assigns equal weights to all observations.

$$\hat{y}_{T+h|T} = \bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t$$

➢ **What could be a method between these two extremes for forecasting by using weighted observations?**

Simple exponential smoothing

- **Main idea:** Assign larger weights to more recent observations than to observations from the distant past.

- **Simple exponential smoothing (SES)** uses a weighted moving average with weights that decrease *exponentially* as observations come from further in the past.

Simple exponential smoothing (SES): Forecast equation

0.2   0.16   0.032

$$\hat{y}_{T+h|T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots$$

- The *smoothing parameter* α with $0 < \alpha \leq 1$ controls the weights decrease. Gewichte müssen in Summe 1 sein, deshalb $\alpha(1-\alpha)^{T-t}$

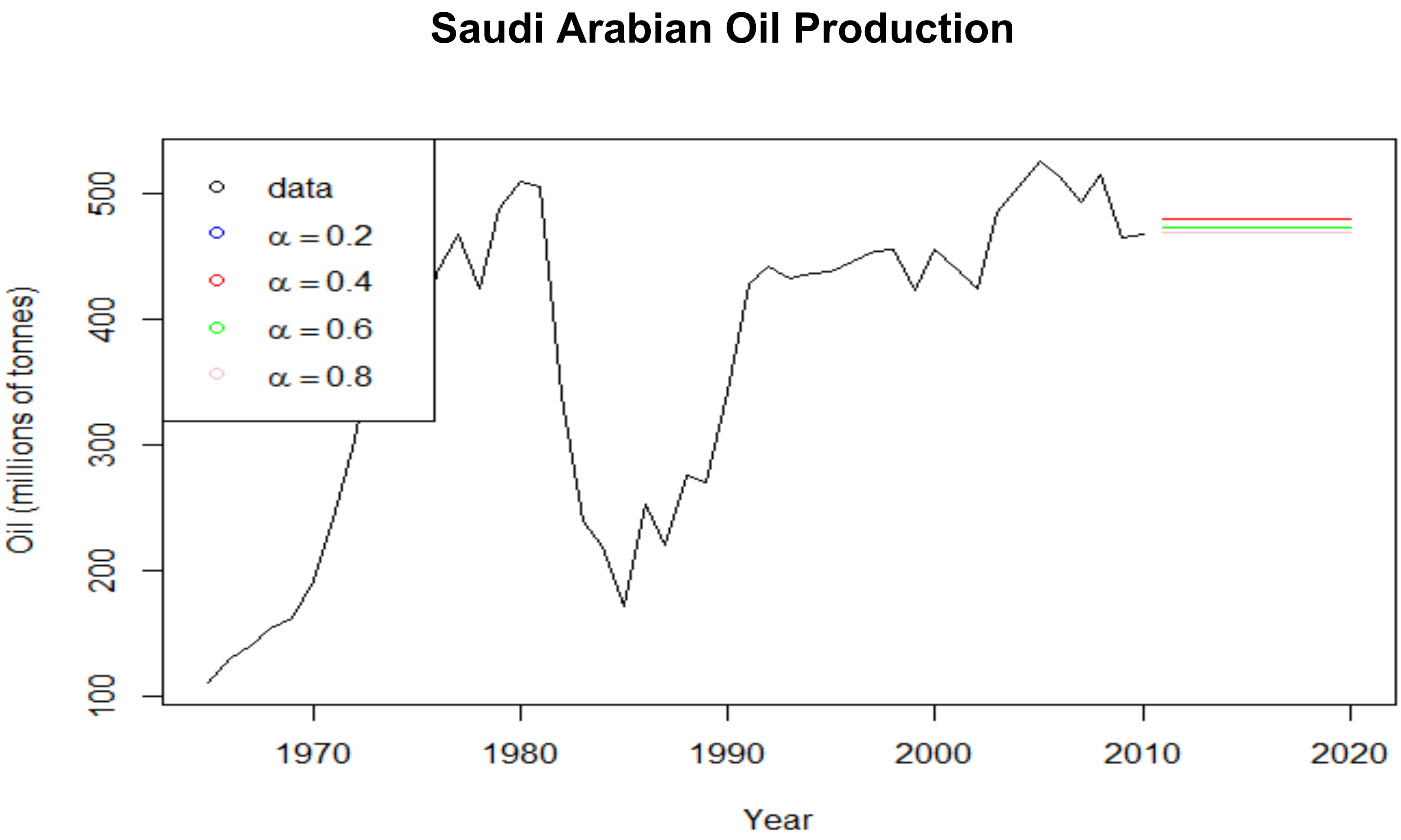
```r
ses(y, h=10, alpha=0.2, level=FALSE)
#y: the time series; h: forecasting horizon
#alpha: smoothing parameter; if NULL, it is estimated.
#level: confidence levels for prediction intervals
```


$\alpha(1 + (1-\alpha) + (1-\alpha)^2 \ldots)$ geometrische Reihe! $\frac{1}{1-q} = \frac{1}{1-(1-\alpha)} = \frac{1}{\alpha}$ | $\alpha \cdot \frac{1}{\alpha} = 1$

meanf $\hat{=}$ $\alpha \approx 0$; alle Werte gleich gewichten

naive $\hat{=}$ $\alpha \approx 1$; letzten Wert übergewichten

Simple exponential smoothing (forecast)



Simple exponential smoothing

- The following table shows the weights attached to observations for four different choices of α when forecasting using SES:

observation	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$
y_T	0.2	0.4	0.6	0.8
y_{T-1}	0.16	0.24	0.24	0.16
y_{T-2}	0.128	0.144	0.096	0.032
y_{T-3}	0.1024	0.0864	0.0384	0.0064
y_{T-4}	$(0.2)(0.8)^4$	$(0.4)(0.6)^4$	$(0.6)(0.4)^4$	$(0.8)(0.2)^4$
y_{T-5}	$(0.2)(0.8)^5$	$(0.4)(0.6)^5$	$(0.6)(0.4)^5$	$(0.8)(0.2)^5$

- What happens if $\alpha \downarrow 0$, $\alpha \uparrow 1$, or $\alpha = 1$?
  - $\alpha \downarrow 0$: observations from the more distant past get more important
  - $\alpha \uparrow 1$: more weight is given to more recent observations
  - $\alpha = 1$: $\hat{y}_{T+h|T} = y_T$, i.e. we get the naïve method.

Hochschule Esslingen
University of Applied Sciences

A different way to look at "simple exponential smoothing" is its "weighted average form".

- At each time period T an "exponentially smoothed" level l_T is calculated, which updates the previous level l_{T-1} as the current best forecast:

gilt für alle T > 0

(1) $l_T = \alpha y_T + (1-\alpha) l_{T-1}, \quad 0 \leq \alpha \leq 1,$

$l_{T-1} = \alpha y_{T-1} + (1-\alpha) l_{T-2}$

(2) $\hat{y}_{T+1|T} = l_T$ *

* Level wird für Vorhersage verwendet

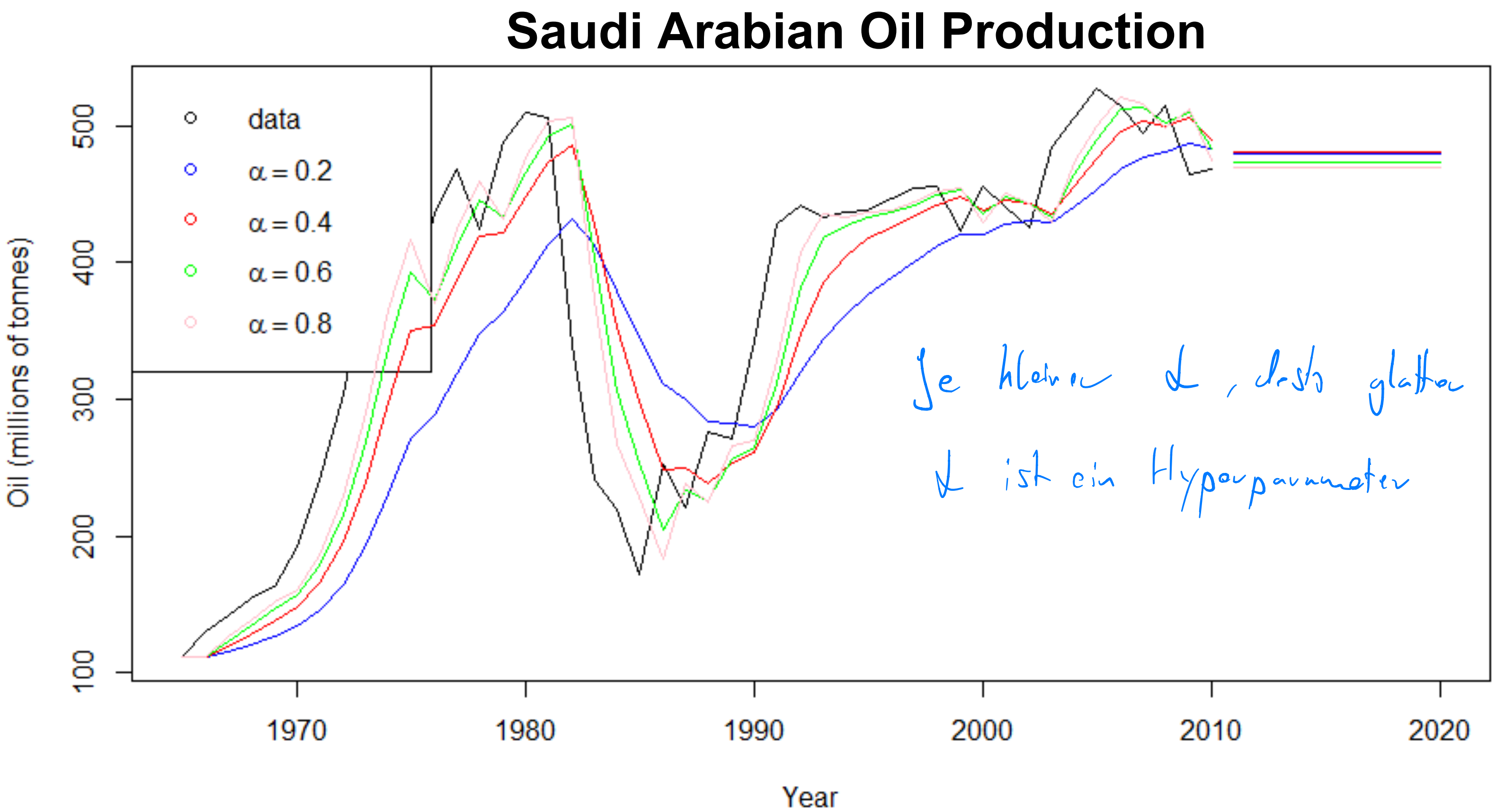
- The forecast is the weighted sum of the previous level l_{T-1} and the current time series value y_T.

Proof: By recursively evaluating $\hat{y}_{T+1|T}$ one obtains:

$$\hat{y}_{T+1|T} = \alpha y_T + (1-\alpha) l_{T-1} = \alpha y_T + \alpha(1-\alpha) y_{T-1} + (1-\alpha)^2 l_{T-2} = \cdots$$
$$= \alpha y_T + \alpha(1-\alpha) y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots + (1-\alpha)^T l_0$$

Simple Exp. Smoothing (smoothing)

Exponential smoothing is also a method for smoothing the time series (as moving average filter) and forecasting this smoothed time series.



Je kleiner α, desto glatter

α ist ein Hyperparameter

Simple Exp. Smoothing (parameters)

Once a value for α as been selected the forecast $\hat{y}_{T+1|T}$ depends only on the two values

- the current actual time series value y_T
- the level l_{T-1} of the previous period (or the forecast for the current period $\hat{y}_{T|T-1}$)

Initialization step needed to start the smoothing period:

$$\boxed{l_0 = y_0}$$

α als auch l_0 = Hyperparameter, die festgelegt werden müssen, wenn das beste Modell gesucht wird

Simple exponential smoothing

- **Disadvantage of SES:** It is only suitable for forecasting data with *no trend* and *no seasonal pattern*.

- (We will restrict ourselves to *non-seasonal* methods of exponential smoothing in this lecture, but we will learn how we can respect trends in the time series.)

- Therefore, SES only has a "flat" forecast function, i.e. for longer forecast horizons $h \geq 2$, we get:

$$\hat{y}_{T+h|T} = \hat{y}_{T+1|T}$$

Example (library: fpp)


```
#Train data: oil time series from 1965-2010
Oildata=oil #or window(oil,start=1996,end=2007)
fit1=ses(oildata, alpha=0.2, initial="simple",h=10, level=FALSE)
fit2=ses(oildata, alpha=0.4, initial="simple",h=10, level=FALSE)
fit3=ses(oildata, alpha=0.6, initial="simple",h=10, level=FALSE)
fit4=ses(oildata, alpha=0.8, initial="simple",h=10, level=FALSE)
#Create plot
plot(fit1, ylab="Oil (millions of tonnes)", xlab="Year",
main="Annual oil production in Saudi Arabia (1995-2010)",
fcol="white", lwd=2)
lines(fitted(fit1), col="blue",lwd=2) # smoothing
lines(fitted(fit2), col="red",lwd=2)  # smoothing
lines(fitted(fit3), col="green",lwd=2)# smoothing
lines(fitted(fit4), col="pink",lwd=2) # smoothing
lines(fit1$mean, col="blue", lwd=2, lty=1) # forecasting
lines(fit2$mean, col="red", lwd=2, lty=1)  # forecasting
lines(fit3$mean, col="green", lwd=2,lty=1) # forecasting
lines(fit4$mean, col="pink", lwd=2,lty=1)  # forecasting
legend("topleft",lty="solid",col=c("black","blue","red","green",
"pink"),c("data", expression(alpha==0.2),
expression(alpha==0.4),expression(alpha==0.6),
expression(alpha==0.8)))
```

Holt's linear trend method

- Extension of SES to allow forecasting of data *with a trend*.

Holt's *linear* trend method

Forecast equation: $\hat{y}_{t+h|t} = \ell_t + hb_t$; b_t ist Trend

Level equation: $\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$

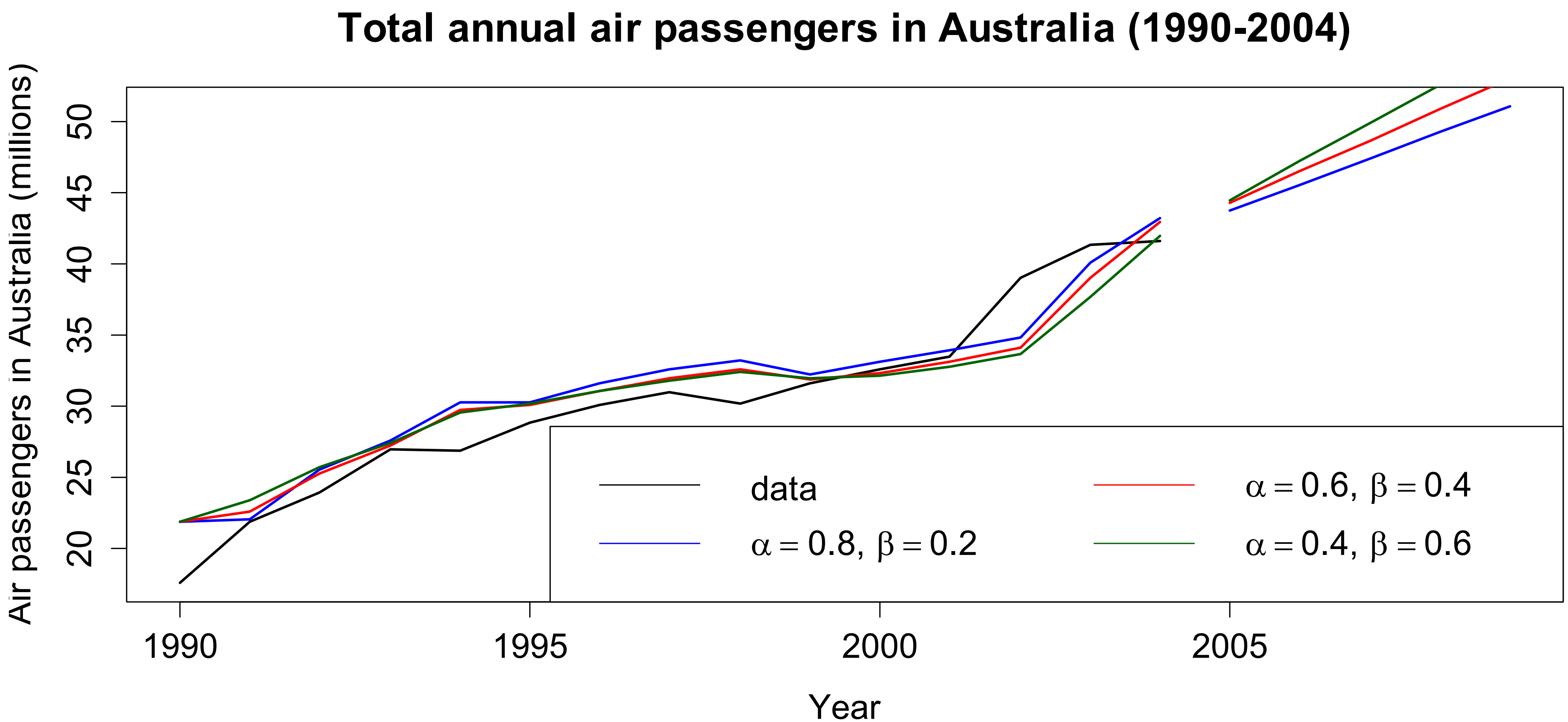
Trend equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$

- α and β^* (with values between 0 and 1) are the smoothing parameters. (We use β^* for β to be consistent with [Hyndman].)
- ℓ_t denotes an estimate of the level of the series at time t.
- b_t denotes an estimate of the trend (slope) of the series at time t.

Optimization can be used to determine parameters α, β^*, ℓ_0, and b_0.

[cf. Slide 19 of "ExponentialSmoothing"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Holt's linear trend method




```r
holt(y, h=10, alpha=NULL, beta=NULL, level=c(80,95))
#y: the time series; h: forecasting horizon
#alpha/beta: smoothing parameters;if NULL => estimated
#level: confidence levels for prediction intervals
```

Holt's linear trend method


```
air <- window(ausair,start=1990,end=2004)
fit1<-holt(air,alpha=0.8,beta=0.2,initial="simple",h=5)
fit2<-holt(air,alpha=0.6,beta=0.4,initial="simple",h=5)
fit3<-holt(air,alpha=0.4,beta=0.6,initial="simple",h=5)
#Create plot
plot(fit1, ylab="Air passengers in Australia (mio.)",
xlab="Year",main="Total annual air passengers in
Australia (1990-2004)",fcol="white",plot.conf=F,lwd=2)
lines(fitted(fit1),col="blue",lwd=2)
lines(fitted(fit2),col="red",lwd=2)
lines(fitted(fit3),col="darkgreen",lwd=2)
lines(fit1$mean,col="blue",lwd=2)
lines(fit2$mean,col="red",lwd=2)
lines(fit3$mean,col="darkgreen",lwd=2)
legend("bottomright",lty="solid",ncol=2,col=c("black","bl
ue","red","darkgreen"),c("data",expression(paste(alpha==0
.8,",",beta==0.2)),expression(paste(alpha==0.6,",",beta==
0.4)),expression(paste(alpha==0.4,", ", beta==0.6))))
```

Hochschule Esslingen
University of Applied Sciences

- A variation from Holt's linear trend method is the *exponential trend method:* ähnlich wie normaler Holt

Exponential trend method

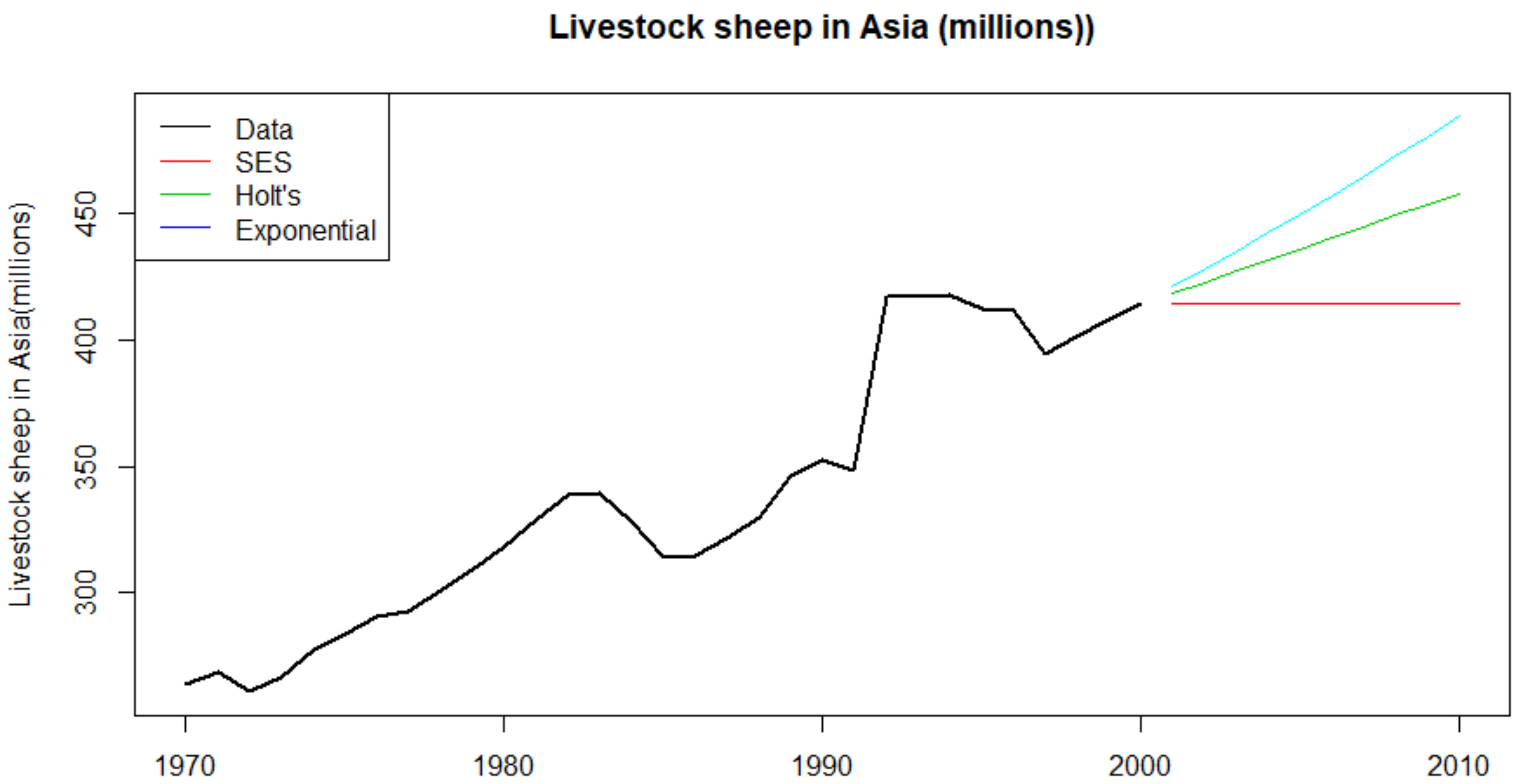
$$\hat{y}_{t+h|t} = \ell_t \cdot b_t^h$$
$$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} \cdot b_{t-1})$$
$$b_t = \beta^* \left({}^{\ell_t}/_{\ell_{t-1}} \right) + (1-\beta^*) b_{t-1}$$

```
holt(y, h=10, alpha=NULL, beta=NULL, exponential=TRUE)
#y: the time series; h: forecasting horizon
#alpha/beta: smoothing parameters;if NULL => estimated
#exponential: logical, if exponential trend is fitted
```

- b_t now represents an estimated *growth rate* (in relative rather than absolute terms) $\Rightarrow$ b_t is multiplied instead of added to the estimated level ℓ_t

Trend methods: Comparison

Hochschule Esslingen
University of Applied Sciences



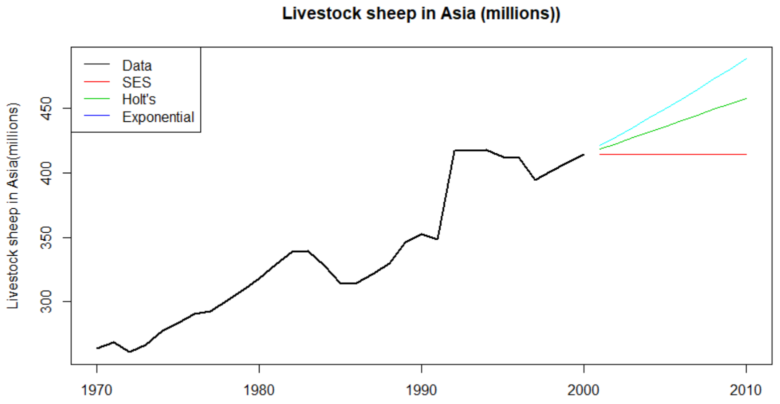
Comparison of trend methods

Hochschule Esslingen
University of Applied Sciences

```
lstock=window(livestock,start=1970,end=2000)
fit1=ses(lstock,initial="simple",h=10, level=FALSE)
fit2=holt(lstock,initial="simple",h=10, level=FALSE)
fit3=holt(lstock, initial="simple",exponential=TRUE,h=10,
level=FALSE)
#Create plot
plot(fit3, ylab="Livestock sheep in Asia(millions)",
main="Livestock sheep in Asia (millions))",
fcol="white",lwd=2)
lines(fit1$mean,col=2,lwd=2)
lines(fit2$mean,col=3,lwd=2)
lines(fit3$mean,col=5,lwd=2)
legend("topleft",lty=1,col=1:6,c("Data", "SES","Holt's",
"Exponential"))
```

Trend methods: Comparison

Parameter	SES	Holt's	Exponential
α	1	0.98	0.98
β^*	-	0	0
l_0	263.92	257.78	255.52
b_0	-	5.01	1.01
Training errors			
RMSE	14.77	13.92	14.06
Forecast errors			
RMSE	25.46	11.88	12.50




```r
accuracy(fit1) # training set
accuracy(fit2)
accuracy(fit3)
accuracy(fit1, livestock) # test set
accuracy(fit2, livestock) # test set
accuracy(fit3, livestock) # test set
```

Overview

1. Exponential smoothing (ETS)

Zeitreihen müssen ggf. differenziert werden

2. **Autoregressive integrated moving average (ARIMA) models**
   - Motivation
   - Stationarity and differencing
   - Backshift notation
   - Autoregressive models $AR(p)$
   - Moving average models $MA(q)$
   - Non-seasonal $ARIMA(p, d, q)$ models
   - Forecasting with $ARIMA$ models

Motivation

- So far, we have seen that exponential smoothing models are based on a **description of trend** (and seasonality) in the data.

- A different approach is to **describe the *autocorrelations*** in the data. This is the aim of Autoregressive integrated moving average (ARIMA) models.

- *Autocorrelation* is a measure for the *linear* relationship between *lagged values* of time series y, e.g. between y_t and y_{t-1}, or between y_t and y_{t-2}, etc.

Differencing

- **Differencing:** Computation of the differences between consecutive observations of a time series.
- The transformation "differencing" is one way to make time series stationary.
- Differencing helps to stabilize the mean of a time series by removing changes in the level of a time series, and so eliminating trend and seasonality.

Differenced series:	$y'_t = y_t - y_{t-1}$

- A differenced series only has $T - 1$ values.

Seasonal differenced series:	$y'_t = y_t - y_{t-m}$, m =number of seasons

Backshift notation

- A very useful notational device is the backward shift operator, *B (in some literature: L)* , which is used as follows:

Backshift operator:	$By_t = y_{t-1}$

- In other words, *B*, operating on y_t, has the effect of **shifting the data back one period**.

- Two applications of *B* to y_t **shifts the data back two periods**:

$$B(By_t) = B^2 y_t = y_{t-2}$$

- Differencing can then be expressed as follows:

Wenn Nullstelle mit Betrag 1 entspricht Einheitswurzel ?

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

[cf. Slide 3 of "Seasonal-ARIMA"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Autoregressive models AR(p)

Hochschule Esslingen
University of Applied Sciences

- **Autoregressive models** model the current value y_t of a time series using a linear combination of its p *past (=lagged) values*.

$AR(p)$:	$y_t = c + a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} + e_t$

- p is the order of the model, $a_1, a_2, \dots, a_p$ are coefficients, e_t is white noise (also called the *random shock*, or the *residual*), c is an optional constant, and $y_{t-1}, y_{t-2}, \dots, y_{t-p}$ are the **lagged values** of y_t.

Moving average models MA(q)

- **Moving average models** model the current value y_t as the weighted average of a white noise process e_t, i.e. using a linear combination of the q *past (=lagged) forecast errors*.

$MA(q)$: $$y_t = \mu + b_1 e_{t-1} + b_2 e_{t-2} + \cdots + b_q e_{t-q} + e_t$$

- E.g.: e_{t-1} is the difference between the actual value and the forecasted value in the previous observation.

- q is the order of the model, μ is the mean of the series (often assumed to be zero), and $b_1, b_2, \dots, b_q$ are coefficients.

- $e_t,\ e_{t-1}, \dots, e_{t-q}$ are white noise, i.e. the current and past, unobserved forecast errors.

Non-seasonal AR(I)MA models

- **Autoregressive moving average models (ARMA)** can model the current value y_t by combining the properties of $AR(p)$ and $MA(q)$ models, i.e. by including both **lagged values of y_t and lagged errors.**

$ARMA(p,q)$:

$$y_t = c + a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} + \\ + b_1 e_{t-1} + b_2 e_{t-2} + \cdots + b_q e_{t-q} + e_t$$

- They model the short-term dynamics of stationary time series.

Wenn nicht stat. ⇒ Diff mit ARIMA

- An ARMA model applied to *differenced* data is called an **ARIMA** model ("I"="*Integrated*" means the reverse of *differencing)*.

Common feature of AR(I)MA Model

Hochschule Esslingen
University of Applied Sciences

- Every stationary ARMA model specifies y_t as a weighted sum of past residuals (error terms) e.g.

  1.) AR(1) : $y_t = ay_{t-1} + e_t \Leftrightarrow y_t = a^2 y_{t-2} + a \cdot e_{t-1} + e_t \Leftrightarrow$

  $y_{t-1} = a y_{t-1} + e_{t-1}$

  AR(1)-Modelle können als MA(t) Modelle geschrieben werden

  MA(t): $y_t = e_t + ae_{t-1} + a^2 e_{t-2} + \cdots + a^t y_0$ $= \mu$

  2.) ARIMA(1,0,1): $y_t = ay_{t-1} + e_t + \mathrm{b}e_{t-1}$

  $\Leftrightarrow y_t = a^2 y_{t-2} + a \cdot e_{t-1} + abe_{t-2} + e_t + \mathrm{b}e_{t-1}$

  MA(t):

  $\Leftrightarrow y_t = a^t y_0 + e_t + (a+b)e_{t-1} + a(a+b)e_{t-2} + \cdots +$

- In an ARIMA model differencing produces a stationary series. These differences are a weighted average of prior errors.

AR Modell kann in MA überführt werden

bis Zeitpunkt t

AR(2):

$$x_t = a_1 x_{t-1} + a_2 \cdot x_{t-2} + \varepsilon_t \quad : \text{für alle } t$$

$$x_{t-1} = a_1 \cdot x_{t-2} + a_2 \cdot x_{t-3} + \varepsilon_{t-1}$$

$$x_{t-2} = a_1 \cdot x_{t-3} + a_2 \cdot x_{t-4} + \varepsilon_{t-2}$$

$$x_t = \overbrace{a_1 \cdot a_1}^{\text{wird immer kleiner}} x_{t-2} + a_1 a_2 x_{t-3} + a_1 \varepsilon_{t-1}$$
$$+ a_2 \cdot a_1 x_{t-3} + a_2^2 x_{t-4} + a_2 \cdot \varepsilon_{t-2} + \varepsilon_t$$

ARMA(4; 2)

$x_{t-3} = \dots \qquad x_{t-3} \propto x_{t-2},\ x_{t-5},\ \varepsilon_{t-3}$

$x_{t-4} = \dots \qquad x_{t-4} \propto x_{t-5} + x_{t-6},\ \varepsilon_{t-4}$

$\Rightarrow$ ARMA(6; 4)

bis bei x_0 (Intercept)

Non-seasonal AR(I)MA models

$ARIMA(p, d, q)$:

$$y'_t = c + a_1 y'_{t-1} + a_2 y'_{t-2} + \cdots + a_p y'_{t-p} + \\ + b_1 e_{t-1} + b_2 e_{t-2} + \cdots + b_q e_{t-q} + e_t$$

- y'_t is the differenced series (it may have been differenced more than once).

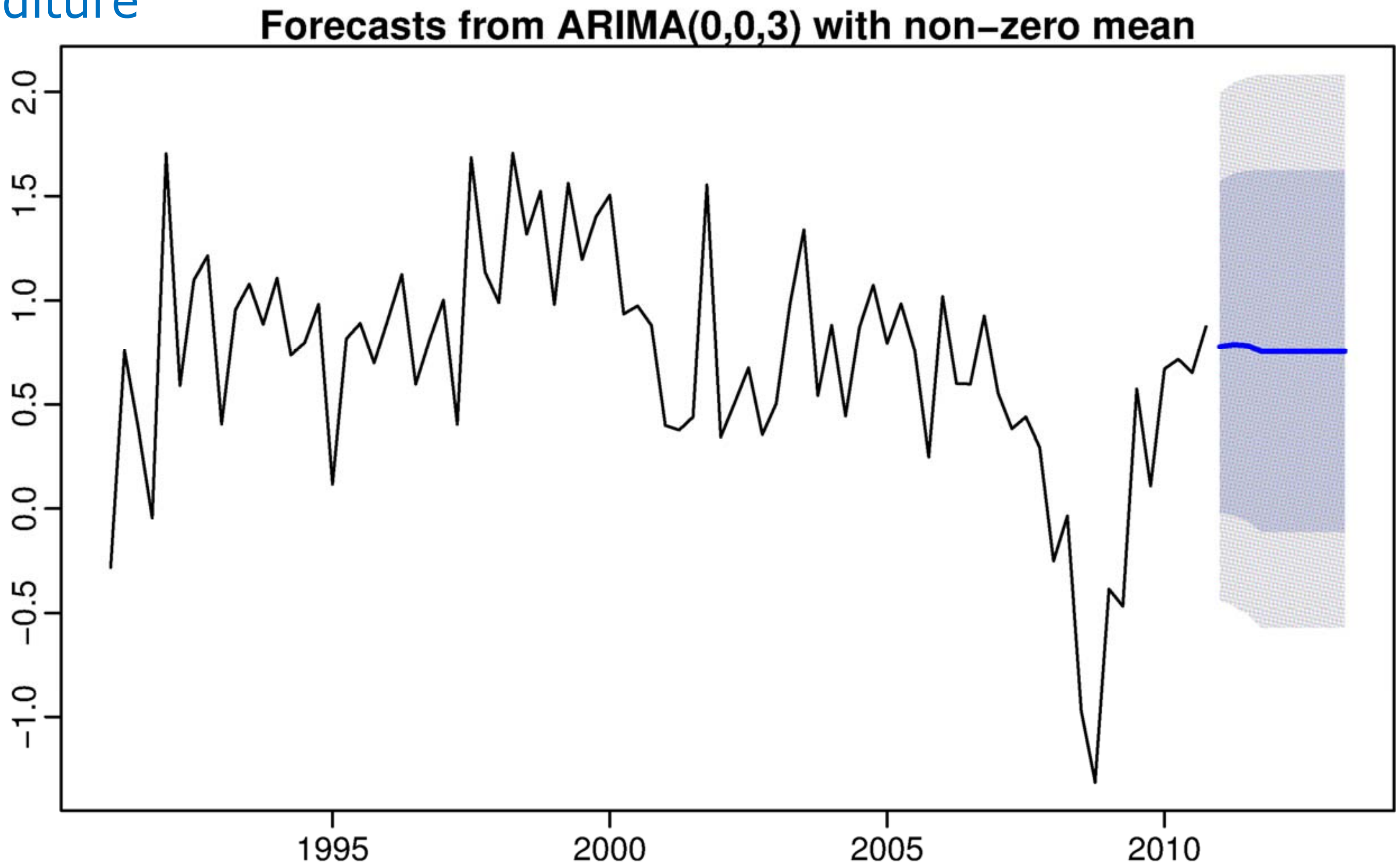
- p = order of the autoregressive part;
  d = degree of first differencing involved;
  q = order of the moving average part.

d mal differenzieren, um stationär zu werden

- $AR(p) = ARIMA(p, 0,0)$ and $MA(q) = ARIMA(0,0, q)$.

Non-seasonal ARIMA - Example

Example: Forecasting percentage change in US consumption expenditure



Forecasting AR(I)MA Model

- Characteristics
  - Forecasts from stationary models revert to the mean
  - Integrated models revert to the trend
  - Accuracy of forecast deteriorates as one extrapolates further
    - » Variance of prediction error grows
    - » Prediction intervals for fixed levels (e.g. 95%) get wider

- Calculations
  - Fill in the unknown values in the model with predictions
  - Pretend estimated model is the true model

Mittelwert = 0 von white noise verwenden?

e.g.: ARMA(2,1): $y_t = a_0 + a_1 y_{t-1} + a_2 y_{t-2} + e_t + \mathrm{b}e_{t-1}$

forecast h=1: $\hat{y}_{t+1|t} = a_0 + a_1 y_t + a_2 y_{t-1} + (\hat{e}_{t+1|t} = 0) + \mathrm{b}e_t$

forecast h=2: $\hat{y}_{t+2|t} = a_0 + a_1 \hat{y}_{t+1|t} + a_2 y_t + (\hat{e}_{t+2|t} = 0) + \mathrm{b}(\hat{e}_{t+1|t} = 0)$

forecast h=3: $\hat{y}_{t+3|t} = a_0 + a_1 \hat{y}_{t+2|t} + a_2 \hat{y}_{t+1|t} + 0 + 0$

h=4: $\hat{y}_{t+4|t} = a_0 + a_1 \hat{y}_{t+3|t} + a_2 \hat{y}_{t+2|t} + 0 + 0$

AR-terms gradually become smaller, MA-terms disappear!

Accuracy of AR(I)MA Forecasts

- Assumptions
  - ARMA models represent y_t as weighted sum of past residuals

- Theory: Forecasts work with unknown error terms set to 0

e.g.:

(1)
$$y_{t+1} = \mu + e_{t+1} + b_1 e_t + b_2 e_{t-1} + b_3 e_{t-2} \ldots$$
known values
$$\hat{y}_{t+1|t} = \mu + \quad b_1 e_t + b_2 e_{t-1} + b_3 e_{t-2} \ldots$$
$$\Longrightarrow y_{t+1} - \hat{y}_{t+1|t} = e_{t+1}$$
$\Longrightarrow$ variance of forecast error=variance(e_{t+1})=σ^2

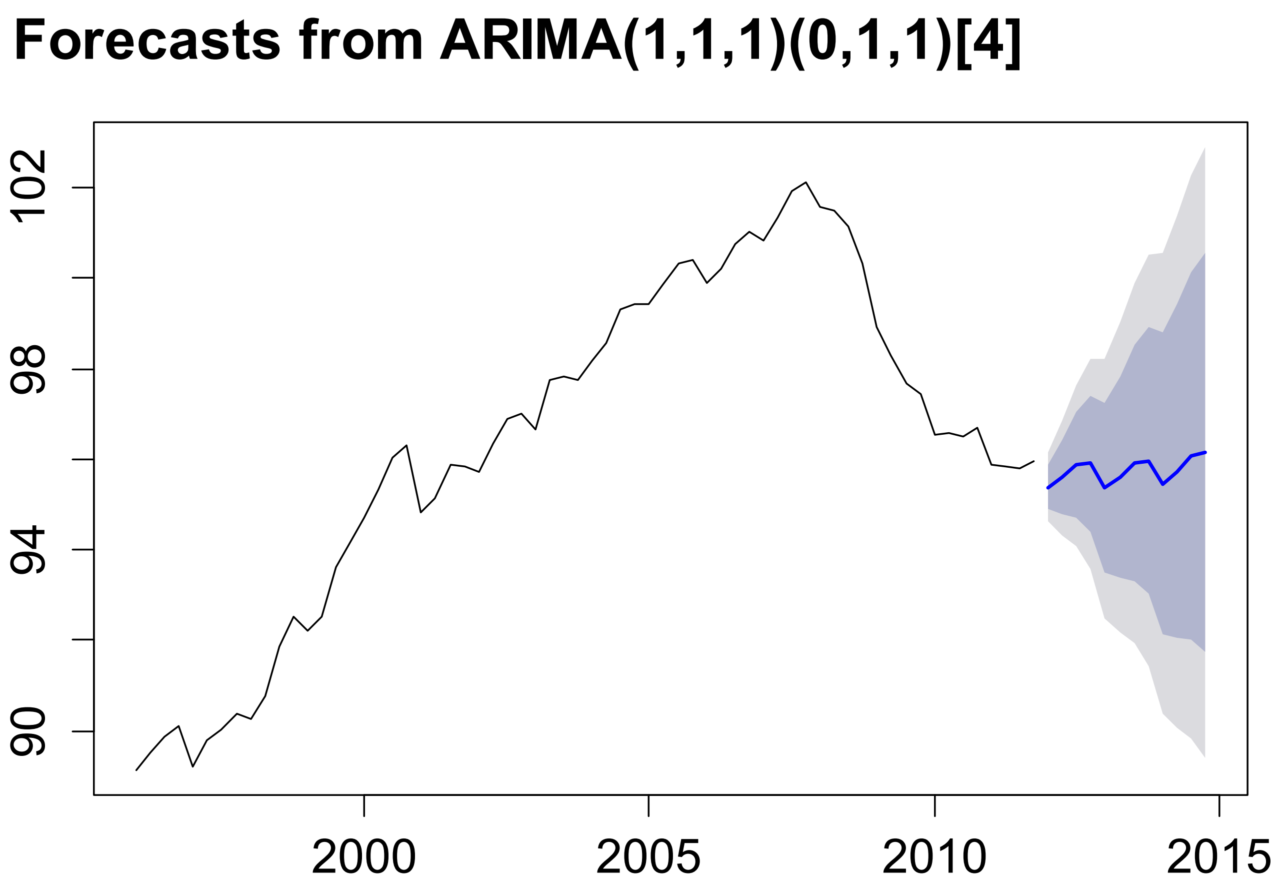
(2)
$$y_{t+2} = \mu + e_{t+2} + b_1 e_{t+1} + b_2 e_t + b_3 e_{t-1} \ldots$$
known values
$$\hat{y}_{t+2|t} = \mu + \quad + b_2 e_t + b_3 e_{t-1} \ldots$$
$$\Longrightarrow y_{t+2} - \hat{y}_{t+2|t} = e_{t+2} + b_1 e_{t+1}$$
$\Longrightarrow$ variance of forecast error=variance($e_{t+2} + b_1 e_{t+1}$)=$\sigma^2 + b_1 \sigma^2$

> Variance of forecast error for horizon h grows with $\sigma^2(1 + b_1 + \cdots + b_{h-1})$

Seasonal ARIMA - Example

Hochschule Esslingen
University of Applied Sciences

Example: Forecasting the European retail trade index



Example


```r
auto.arima(y, seasonal=TRUE)
#y: the time series
#seasonal: logical, if searching for a seasonal model
```

```r
#NON-SEASONAL EXAMPLE:
#Plot quarterly percentage changes in US consumption
#expenditure
plot(usconsumption[, 1], xlab="Year", ylab="Quarterly
percentage change", main="US consumption")
#Automatically fit an ARIMA model
fit <- auto.arima(usconsumption[,1], seasonal=FALSE)
#Forecasting
plot(forecast(fit, h=10), include=80)


#SEASONAL EXAMPLE:
fit <- auto.arima(euretail, seasonal=TRUE)
plot(forecast(fit, h=12))
```

ARIMA vs. ETS

- **Simple exponential smoothing:**
  Forecasts equivalent to $\boldsymbol{ARIMA(0,1,1).}$ = SES

- **Holt's method:**
  Forecasts equivalent to $\boldsymbol{ARIMA(0,2,2).}$ = Holt

[cf. Slide 25 of "Seasonal-ARIMA"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Stationarity

The main assumption when forcasting on time series data has been stationarity, s.t.

- the mean value $\mu_x(t)$ at time t is independent of t s.t. $\mu_x(t) = \mu_x$ and
- the covariance function and variance function

$$\gamma_x(t+h,t) = cov(x_{t+h}, x_t) = E[(x_{t+h} - \mu_{t+h})(x_t - \mu_t)]$$

is independent of t for every $h \geq 0$.

Examples of non-stationarity:

Unit root tests bright nights?

(A) Deterministic trends (trend stationarity).

(B) Level shifts

Can be captured with a uni root test

(C) Variance changes.

(D) Unit roots (stochastic trends i.e. random walk).

[cf. Slide 25 of "Seasonal-ARIMA"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Unit Root and ARIMA Models

- A time series is non-stationary if it contains a unit root

$\boxed{\text{unit root} \Rightarrow \text{nonstationary}}$ The reverse is not true.

- Many results of traditional statistical theory do not apply to unit root process, such as law of large number and central limit theory.

- For unit root process, we need to apply ARIMA model; that is, we take difference (maybe several times) before applying the ARMA model.

[cf. Slide 25 of "Seasonal-ARIMA"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Unit Root and ARIMA Models

Consider the AR(1) time series

$$y_t = c + a_1 y_{t-1} + e_t, \quad \text{with } e_t \text{ being a white noise process}$$

This time series has a unit root if $a_1 = 1$ (random walk)

In that case the series converges to

$$y_t = ct + y_0 + (e_t + e_{t-1} + e_{t-2} + \cdots + e_0), \quad \text{where}$$

- the ct term implies that the series will have a trend if $c \neq 0$

- the series has non-constant variance
$var(y_t) = var(e_t + e_{t-1} + e_{t-2} + \cdots + e_0) = t\sigma^2$

$\Longrightarrow$ the process is not stationary

[cf. Slide 25 of "Seasonal-ARIMA"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Question: Why do we call the random walk time series

$$y_t = y_{t-1} + e_t, \quad \text{with } e_t \text{ being a white noise process}$$

a unit root process?

Answer: We can use the lag operator B to rewrite the time series as

$$y_t = By_t + e_t \qquad \Leftrightarrow (1 - B)y_t = e_t$$

The equation $1 - B = 0$ has the root $B = 1$ which is called unit root.

[cf. Slide 25 of "Seasonal-ARIMA"-slides of R. J. Hyndman, G. Athanasopoulos: Forecasting: principles and practice]

Multiple Unit Roots:

A times series possibly has multiple unit roots!

For example:

$$(1-B)(1-B)y_t = e_t \Leftrightarrow (1-2B+B^2)y_t$$

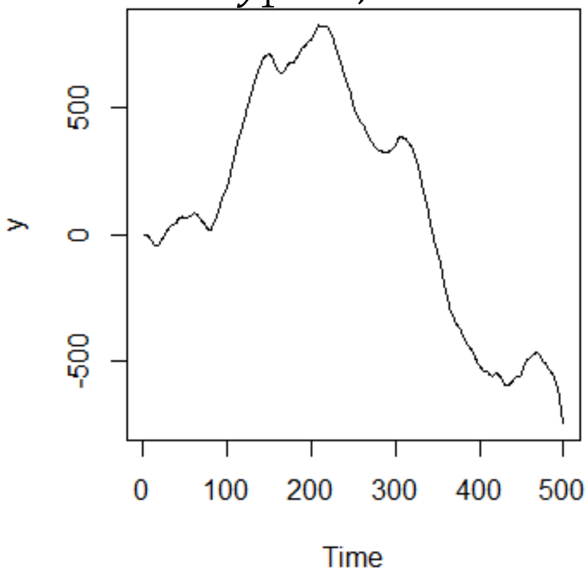
$$\Leftrightarrow \; y_t = 2y_{t-1} - y_{t-2} + e_t$$

So the AR(2) process

$$y_t = 2y_{t-1} - y_{t-2} + e_t$$

is non-stationary and has two unit roots.

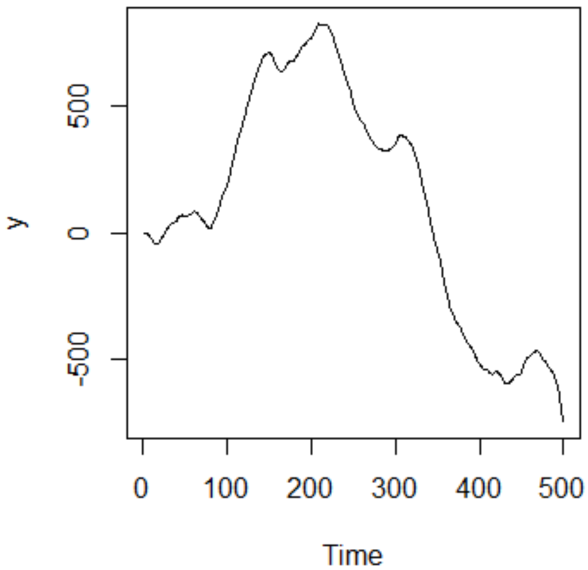
Simulation with $y_0 = 0{,}1$ and $y_1 = 0{,}05$



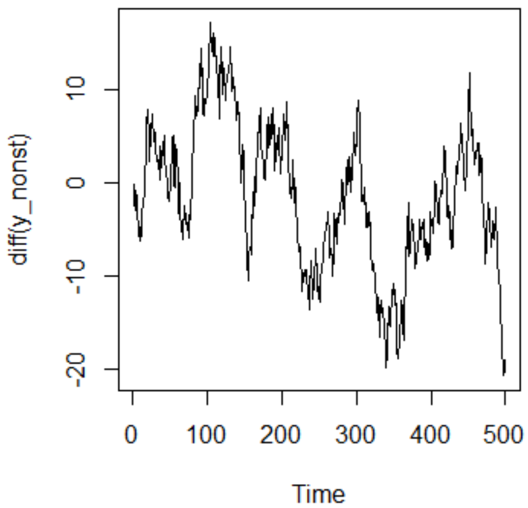
Unit Root and ARIMA Models

- ARMA-Models cannot be applied to unit roots processes!!
- A times series needs to be transformed into a stationary time series:
  - by differencing
  - by differencing more than once

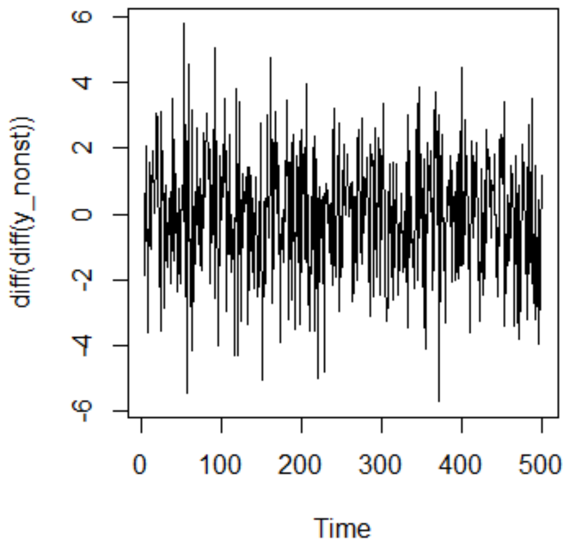
time series y_t



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



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



Unit Root Tests

Given the AR(1) time series

$$y_t = a_1 y_{t-1} + e_t, \text{ with } e_t \text{ being a white noise process}$$

test the Null Hypothesis

$$H_0: \; a_1 = 1$$

This is equivalent to rewriting the time series to

$$\Delta y_t = \beta y_{t-1} + e_t, \text{ with } \; \beta = a_1 - 1$$

and testing the Null Hypothesis

$$H_0: \beta = 0$$

Dickey-Fuller (DF) is the most popular unit root test

Dickey-Fuller (DF) unit root tests

Three kind of Dickey-Fuller (DF) tests:

(1) $y_t = a_1 y_{t-1} + e_t$, with e_t white noise: H_0: $a_1 = 1$

(2) $y_t = c + a_1 y_{t-1} + e_t$, with e_t white noise H_0: $a_1 = 1$

(3) $y_t = c + mt + a_1 y_{t-1} + e_t$, with e_t white noise H_0: $a_1 = 1$

All have a different test statistic!

One as to know in advance,

if the model has a drift ($c \neq 0$) or a trend ($m \neq 0$)!

Augmented Dickey-Fuller (ADF) unit root tests

Three kind of Augmented Dickey Fuller (ADF) tests:

(1) $y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots a_p y_{t-p} + e_t,$ with e_t white noise $\Leftrightarrow$

$\Delta y_t = \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots \beta_p y_{t-p} + e_t,$ with $\beta_1 = a_1 + a_2 + \cdots a_p - 1,\ \beta_2 = \cdots$

$H_0\colon\ \beta_1 = 0$

(2) $y_t = c + a_1 y_{t-1} + a_2 y_{t-2} + \cdots a_p y_{t-p} + e_t,$ with e_t white noise $\Leftrightarrow$

$\Delta y_t = c + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots \beta_p y_{t-p} + e_t,$ with $\beta_1 = a_1 + a_2 + \cdots a_p - 1,$

$H_0\colon\ \beta_1 = 0$

(3) $y_t = c + mt + a_1 y_{t-1} + a_2 y_{t-2} + \cdots a_p y_{t-p} + e_t,$ with e_t white noise $\Leftrightarrow$ ….

Literature

- R. J. Hyndman, G. Athanasopoulos: *Forecasting: principles and practice.* Available online at https://www.otexts.org/fpp, 2014
- Sven F. Crone: Tutorial *'Forecasting with Artificial Neural Networks'* at the 2005 IEEE Summer School in Computational Intelligence EVIC'05, 14.-16.12.2005, Santiago, Chile. Available online at http://www.neural-forecasting.com/
- J. Han, M. Kamber, J. Pei: Data Mining – Concepts and Techniques (3rd Edition), pp. 398-408, 2012, Morgan Kaufmann Verlag