Financial Time Series

Lecture 2

Multiple regression, Decomposition and Smoothing

Multiple regression and forecasting

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t$$

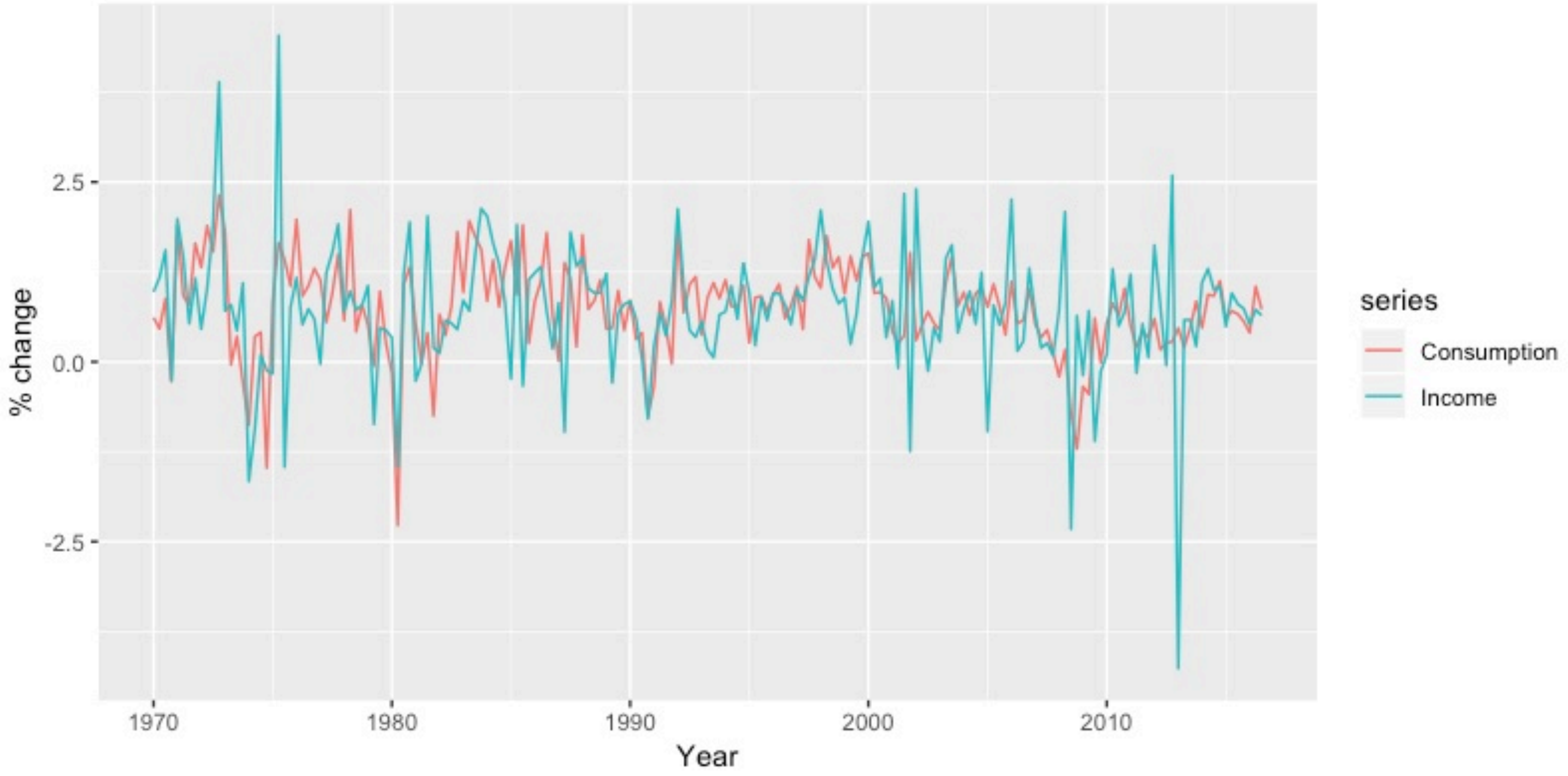
- y_t is the variable we want to predict: the "response" variable
- Each $x_{j,t}$ is numerical and is called a "predictor". They are usually assumed to be known for all past and future times.
- The coefficients $\beta_1, \ldots, \beta_k$ measure the effect of each predictor after taking account of the effect of all other predictors in the model.

That is, the coefficients measure the marginal effects.

- ε_t is a white noise error term

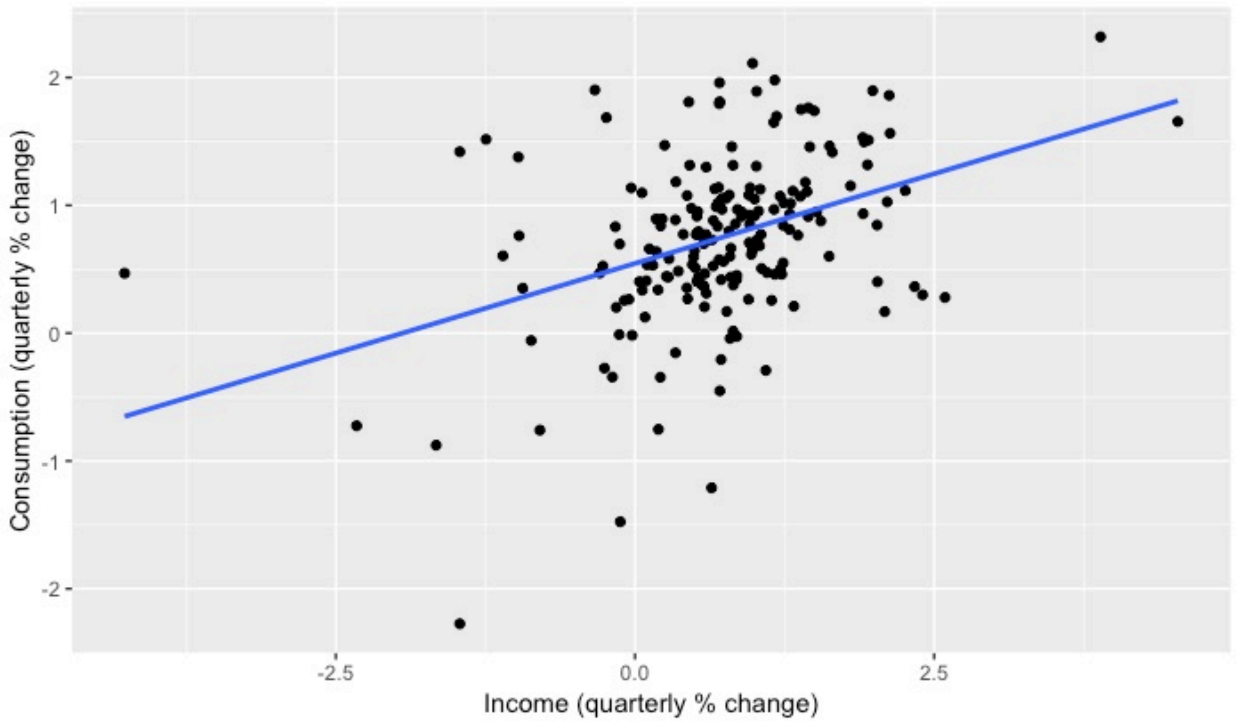
Example: US consumption expenditure

- autoplot(uschange[,c("Consumption","Income")]) + ylab("% change") + xlab("Year")



Example: US consumption expenditure

```
uschange %>%
as.data.frame() %>%
ggplot(aes(x=Income, y=Consumption)) + ylab("Consumption (quarterly % change)") +
xlab("Income (quarterly % change)") + geom_point() + geom_smooth(method="lm", se=FALSE)
```



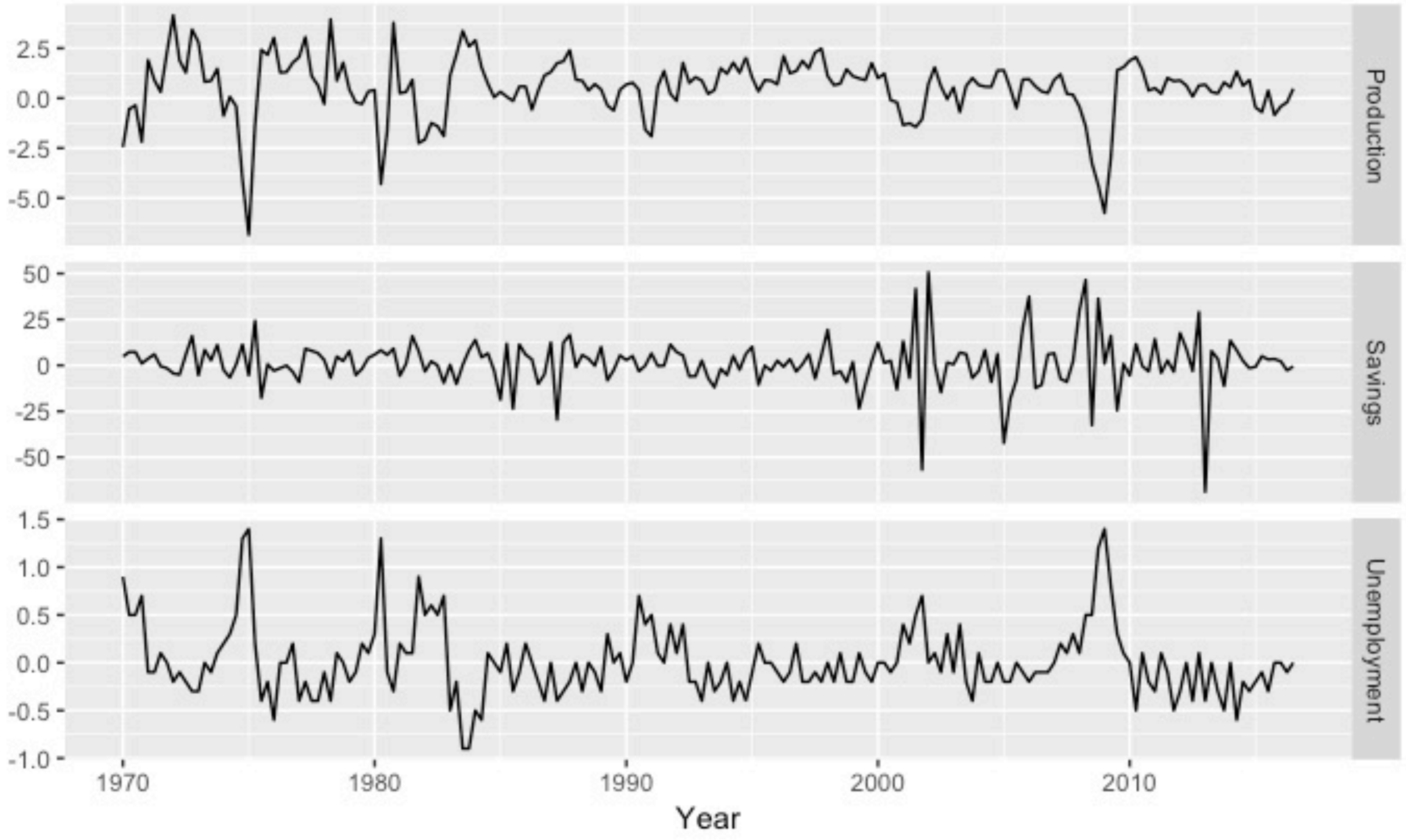
Example: US consumption expenditure

- **tslm**(Consumption ~ Income, data=uschange) %>% summary

```
Call:
tslm(formula = Consumption ~ Income, data = uschange)
Residuals:
     Min       1Q   Median       3Q      Max
-2.40845 -0.31816  0.02558  0.29978  1.45157
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.54510    0.05569   9.789  < 2e-16 ***
Income       0.28060    0.04744   5.915 1.58e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.6026 on 185 degrees of freedom
Multiple R-squared:  0.159,	Adjusted R-squared:  0.1545
F-statistic: 34.98 on 1 and 185 DF,  p-value: 1.577e-08
```

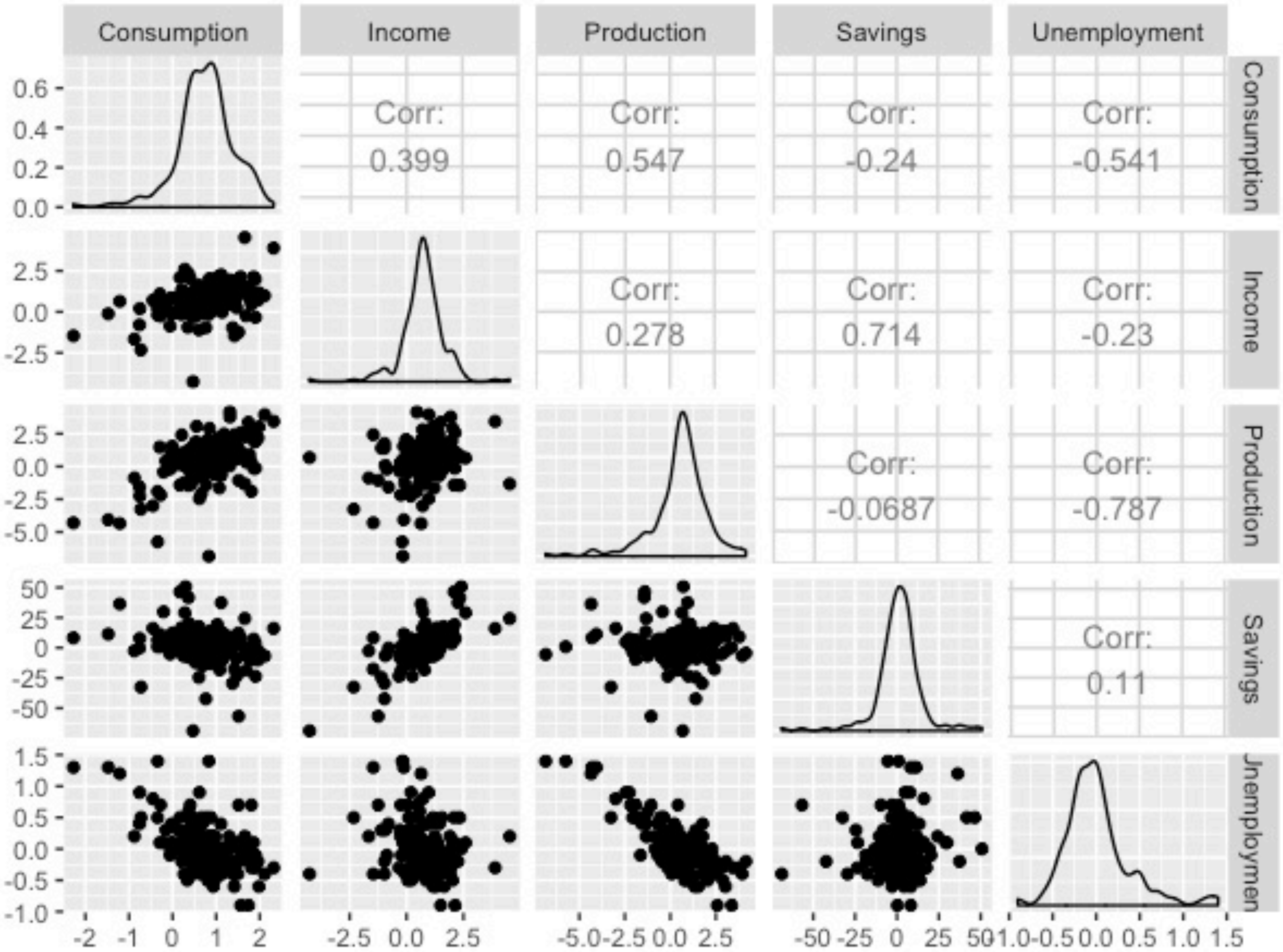
Example: US consumption expenditure

- autoplot(uschange[,3:5], facets=TRUE) + xlab("Year")+ ylab("")



Example: US consumption expenditure

- install.packages("GGally")
- library("GGally")
- **ggpairs**(**as.data.frame**(usdata[,1:5]))



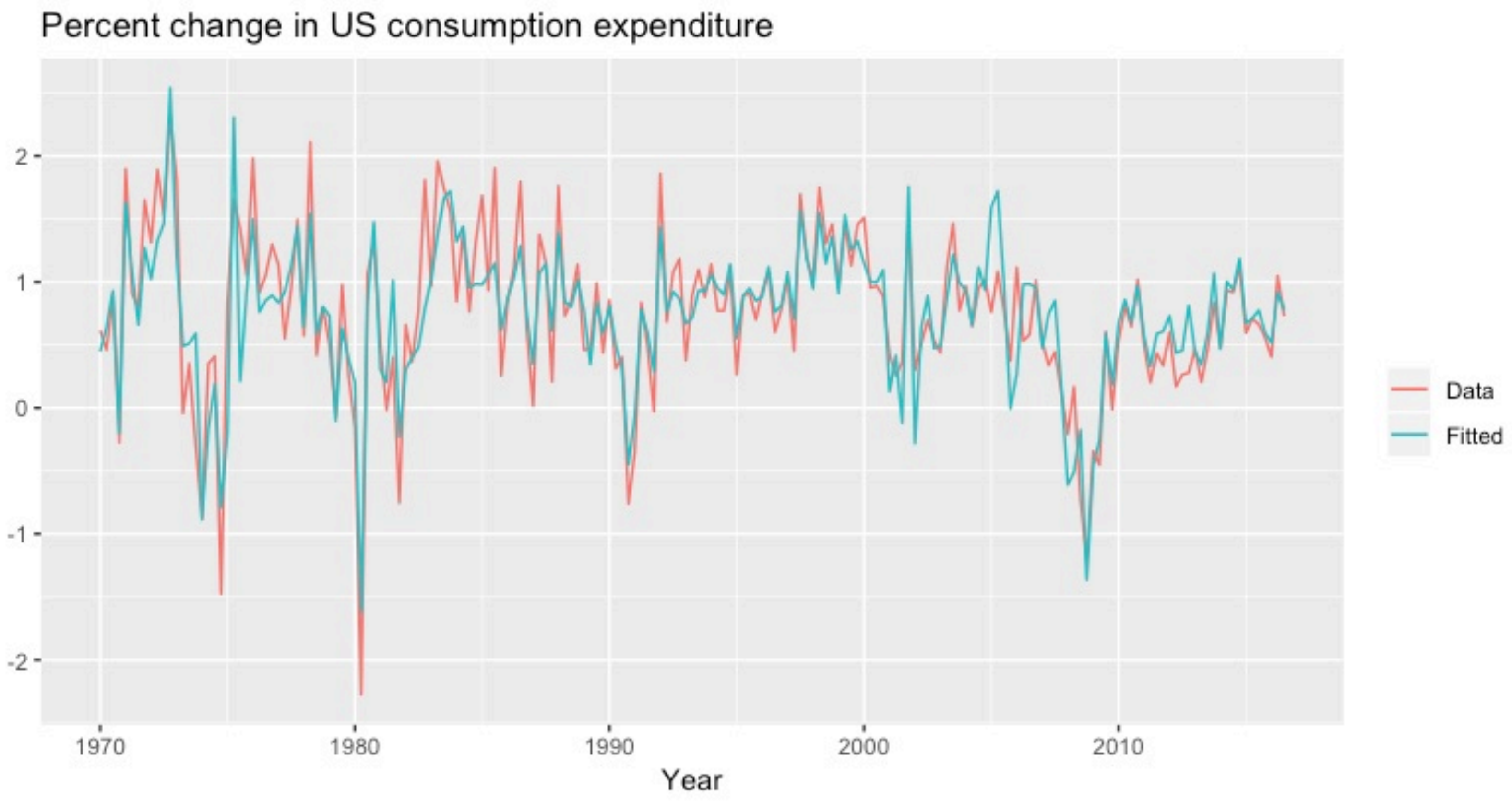
Example: US consumption expenditure

- fit.consMR <- tslm(Consumption ~ Income + Production + Unemployment + Savings, data=uschange) summary(fit.consMR)

```
Call:
tslm(formula = Consumption ~ Income + Production + Unemployment +  Savings, data = uschange)
Residuals:
  Min       1Q   Median       3Q      Max
-0.88296 -0.17638 -0.03679  0.15251  1.20553
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.26729    0.03721   7.184 1.68e-11 ***
Income        0.71449    0.04219  16.934  < 2e-16 ***
Production    0.04589    0.02588   1.773   0.0778 .
Unemployment -0.20477    0.10550  -1.941   0.0538 .
Savings      -0.04527    0.00278 -16.287  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 0.3286 on 182 degrees of freedom
Multiple R-squared:  0.754,                    Adjusted R-squared:  0.7486
F-statistic: 139.5 on 4 and 182 DF,  p-value: < 2.2e-16
```

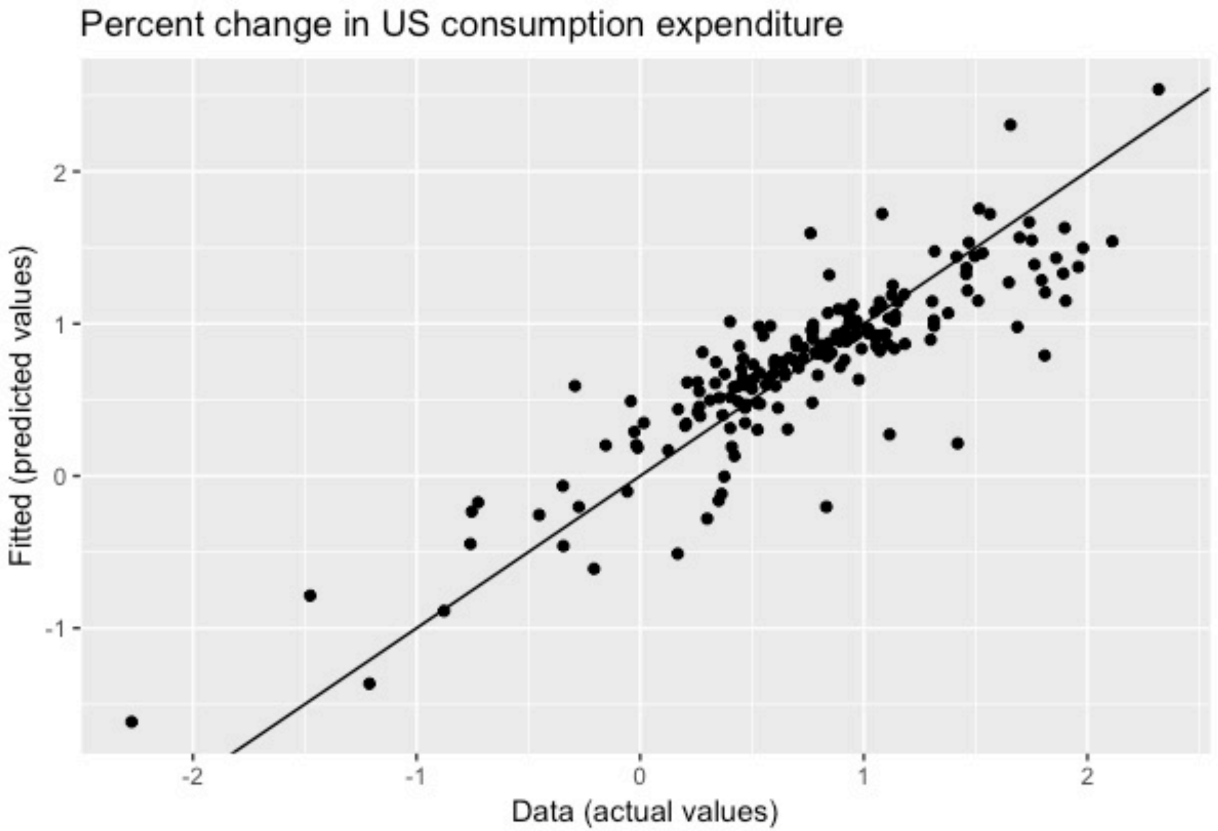
Example: US consumption expenditure

- **autoplot**(uschange[,'Consumption'], series="Data") + **autolayer**(**fitted**(fit.consMR), series="Fitted") + **xlab**("Year") + **ylab**("") + **ggtitle**("Percent change in US consumption expenditure") + **guides**(colour=**guide_legend**(title=" "))



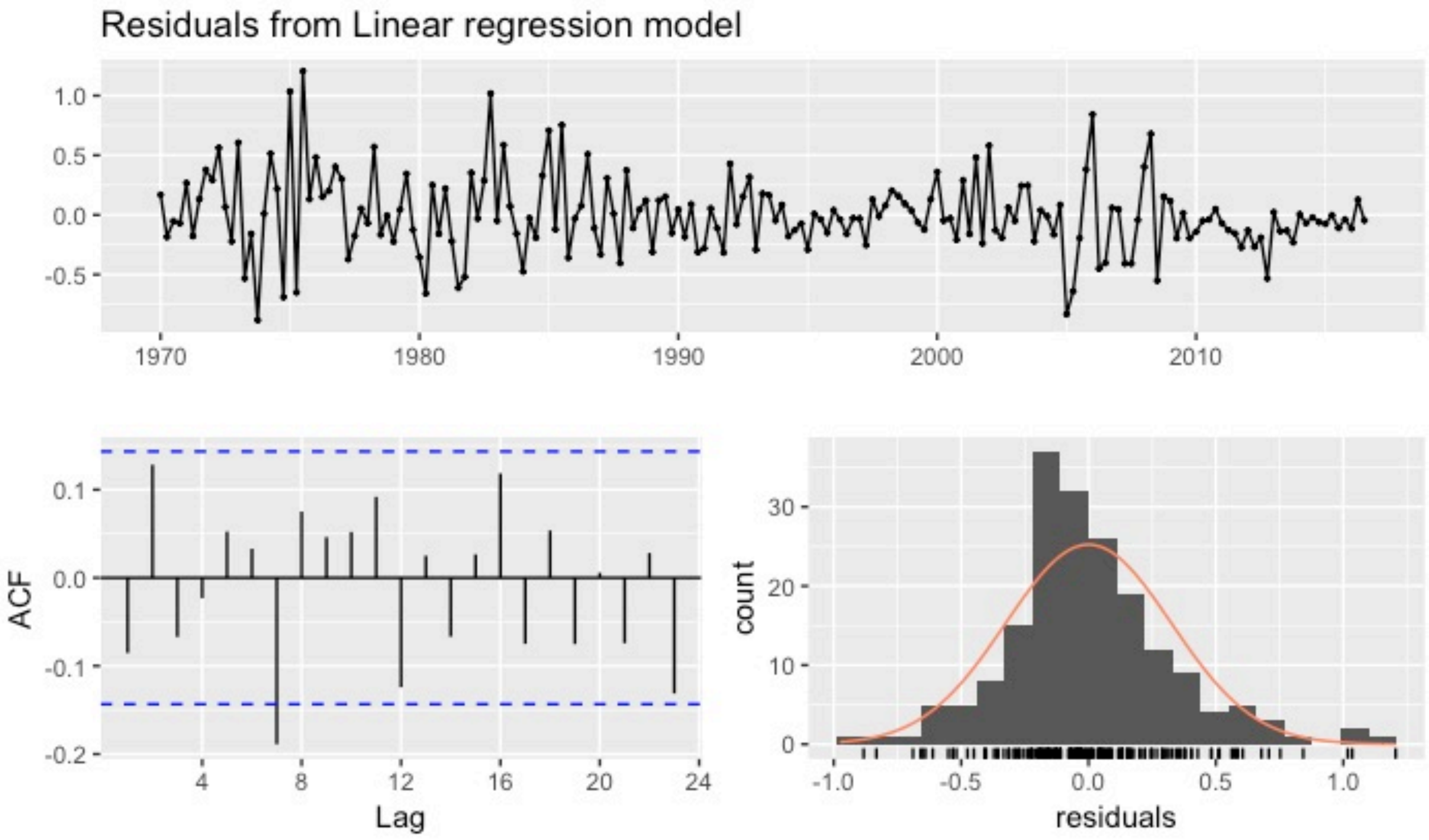
Example: US consumption expenditure

- **cbind**(Data = uschange[,"Consumption"], Fitted = **fitted**(fit.consMR)) %>% **as.data.frame**() %>% **ggplot**(**aes**(x=Data, y=Fitted)) + **geom_point**() + **ylab**("Fitted (predicted values)") + **xlab**("Data (actual values)") + **ggtitle**("Percent change in US consumption expenditure") + **geom_abline**(intercept=0, slope=1)



Example: US consumption expenditure

- **checkresiduals**(fit.consMR)



Multiple regression and forecasting

For forecasting purposes, we require the following assumptions:

- ε_t are uncorrelated and zero mean
- ε_t are uncorrelated with each $x_{j,t}$

It is useful to also have $\varepsilon_t \sim N(0, \sigma^2)$ when producing prediction intervals or doing statistical tests.

Trend and Dummy variables

- Linear trend

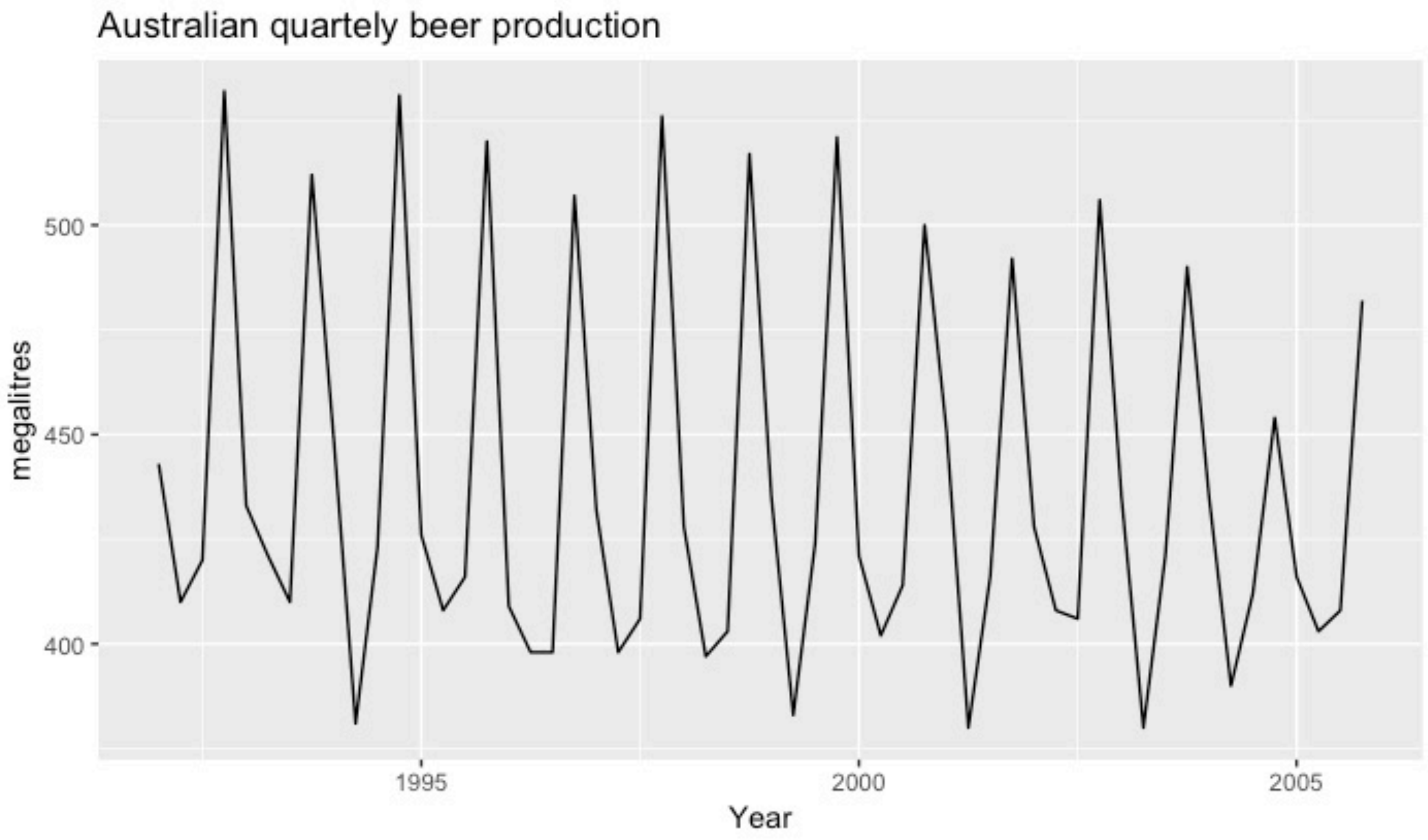
$$x_t = t, \quad t = 1, 2, \ldots, T$$

Strong assumption that trend will continue

- If a categorical variable takes only two values (e.g., ‘Yes’ or ‘No’), then an equivalent numerical variable can be constructed taking value 1 if yes and 0 if no. This is called a dummy variable

Example Beer Production

- beer2 <- window(ausbeer,start=1992,end=2006-.1)
- autoplot(beer2) + labs(title="Australian quartely beer production",x="Year",y="megalitres")



Example Beer Production

- fit <- tslm(beer2 ~ trend + season)
- summary(fit)

```
Call:
tslm(formula = beer2 ~ trend + season)
esiduals:
    Min      1Q  Median      3Q     Max
-44.024  -8.390   0.249   8.619  23.320
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 441.8141     4.5338  97.449  < 2e-16 ***
trend        -0.3820     0.1078  -3.544 0.000854 ***
season2     -34.0466     4.9174  -6.924 7.18e-09 ***
season3     -18.0931     4.9209  -3.677 0.000568 ***
season4      76.0746     4.9268  15.441  < 2e-16 ***
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 13.01 on 51 degrees of freedom
Multiple R-squared:  0.921,	Adjusted R-squared:  0.9149
F-statistic: 148.7 on 4 and 51 DF,  p-value: < 2.2e-16
```

Example Beer Production

- x1=1:56
- x2=rep(c(0,1,0,0),14)
- x3=rep(c(0,0,1,0),14)
- x4=rep(c(0,0,0,1),14)
- f1=lm(beer2~x1+x2+x3+x4)
- summary(f1)

- Then we have the same result as in the previous slide.

Selecting Predictors

- When there are many predictors, how should we choose which ones to use?
- We need a way of comparing two competing models.

What not to do!

- Plot y against a particular predictor (x_j) and if it shows no noticeable relationship, drop it.
- Do a multiple linear regression on all the predictors and disregard all variables whose p values are greater than 0.05.
- Maximize R^2 or minimize MSE

Adjusted R^2

$$\bar{R}^2 = 1 - (1 - R^2)\frac{T-1}{T-k-1}$$

- This is an improvement on R^2, as it will no longer increase with each added predictor. Using this measure, the best model will be the one with the largest value of $\bar{R}^2$. Maximizing $\bar{R}^2$ is equivalent to minimizing the standard error $\hat{\sigma}_\epsilon$.

Akaike's Information Criterion

$$\text{AIC} = -2\log(L) + 2(k+2)$$

where L is the likelihood and k is the number of predictors in the model.

- This is a penalized likelihood approach.
- Minimizing the AIC gives the best model for prediction.
- AIC penalizes terms more heavily than $\bar{R}^2$.

Bayesian Information Criterion

$$\text{BIC} = -2\log(L) + (k+2)\log(T)$$

where L is the likelihood and k is the number of predictors in the model.

- BIC penalizes terms more heavily than AIC
- Also called SBIC and SC.

Choosing regression variables

Backwards stepwise regression

- Start with a model containing all variables.
- Try subtracting one variable at a time. Keep the model if it has lower AIC.
- Iterate until no further improvement.

Notes

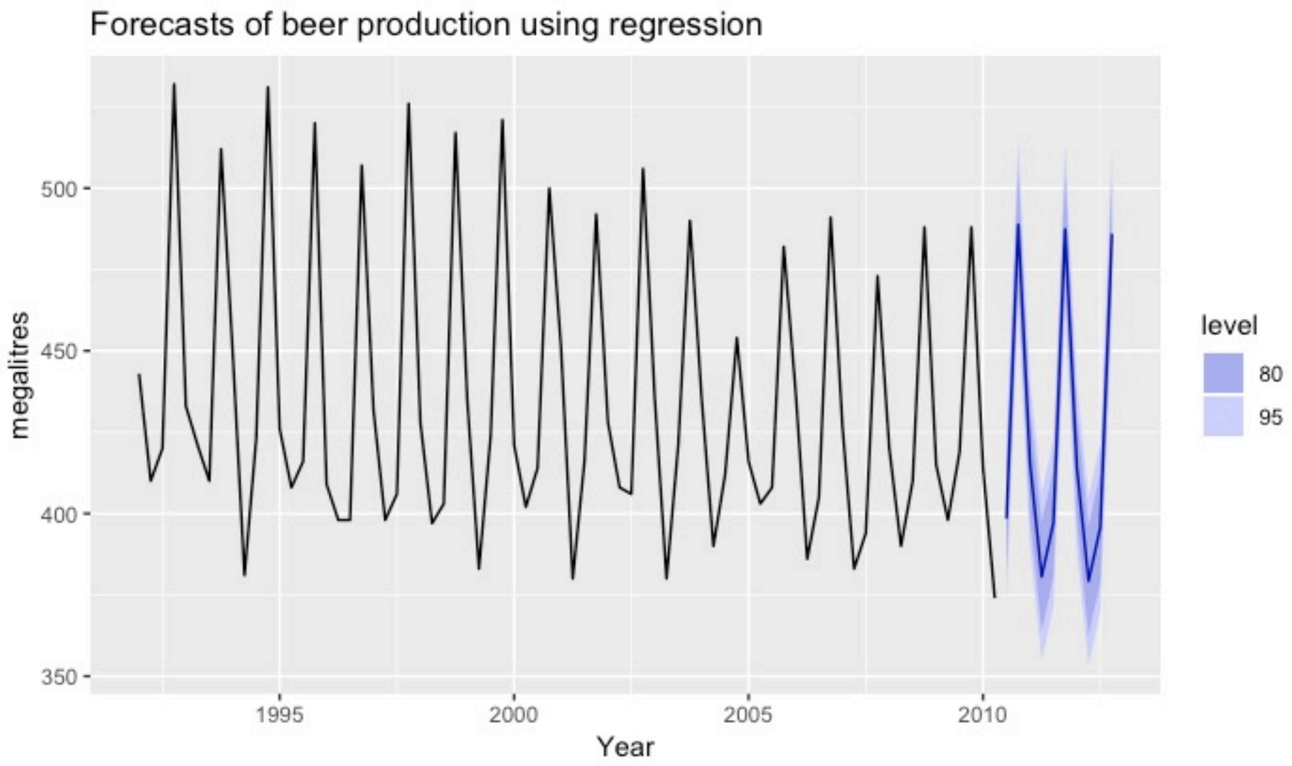
- Stepwise regression is not guaranteed to lead to the best possible model.
- Inference on coefficients of final model will be wrong.

How to know if the model is best fit for your data?

STATISTIC	CRITERION
R-Squared	Higher the better *(> 0.70)*
Adj R-Squared	Higher the better
F-Statistic	Higher the better
Std. Error	Closer to zero the better
t-statistic	Should be greater 1.96 for p-value to be less than 0.05
AIC	Lower the better
BIC	Lower the better

Forecasting in Beer production example

- beer2 <- window(ausbeer, start=1992)
- fit.beer <- tslm(beer2 ~ trend + season)
- fcast <- forecast(fit.beer) autoplot(fcast) + ggtitle("Forecasts of beer production using regression") + xlab("Year") + ylab("megalitres")



Time series components

Recall

- Trend pattern exists when there is a long-term increase or decrease in the data.
- Cyclic pattern exists when data exhibit rises and falls that are not of fixed period (duration usually of at least 2 years).
- Seasonal pattern exists when a series is influenced by seasonal factors (e.g., the quarter of the year, the month, or day of the week).

Time series decomposition

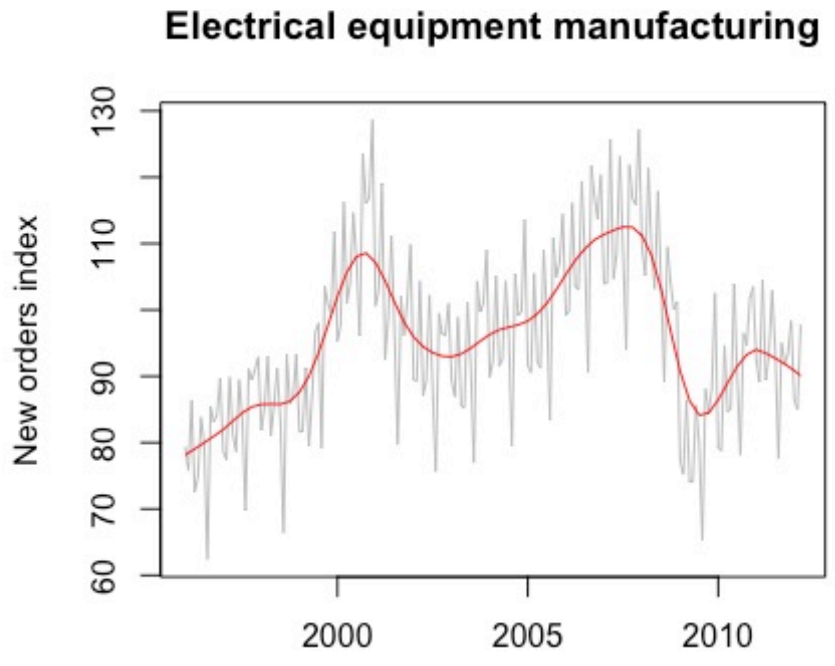
$$y_t = S_t + T_t + R_t$$

where

- y_t = data at period t
- T_t = trend-cycle component at period t
- S_t = seasonal component at period t
- R_t = remainder component at period t

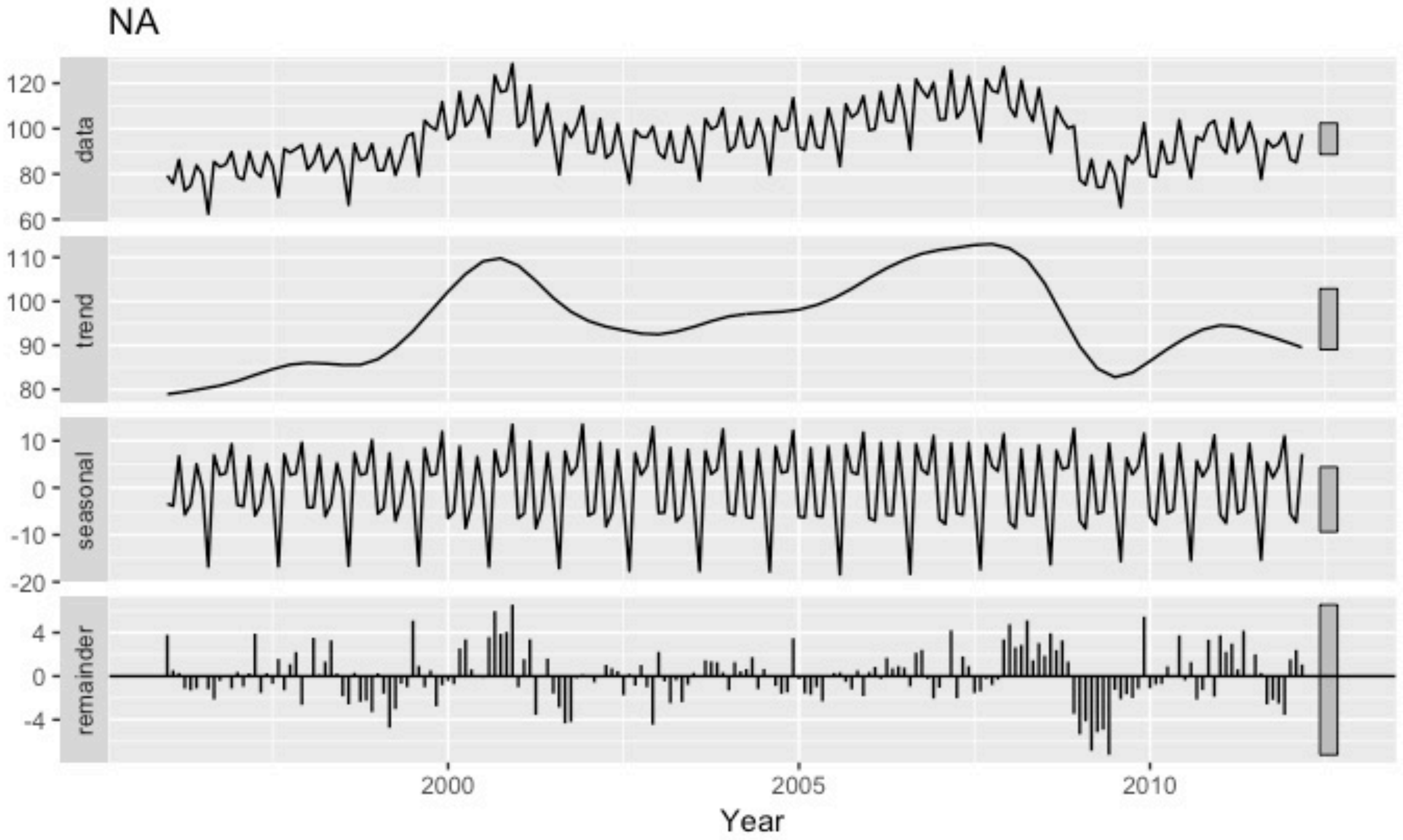
Electrical equipment manufacturing example

- fit <- stl(elecequip, s.window=5)
- plot(elecequip, col="gray", main="Electrical equipment manufacturing", ylab="New orders index", xlab="") lines(fit$time.series[,2],col="red",ylab="Trend")



Electrical equipment manufacturing example

- fit <- stl(elecequip, s.window=7)
- autoplot(fit) + xlab("Year")



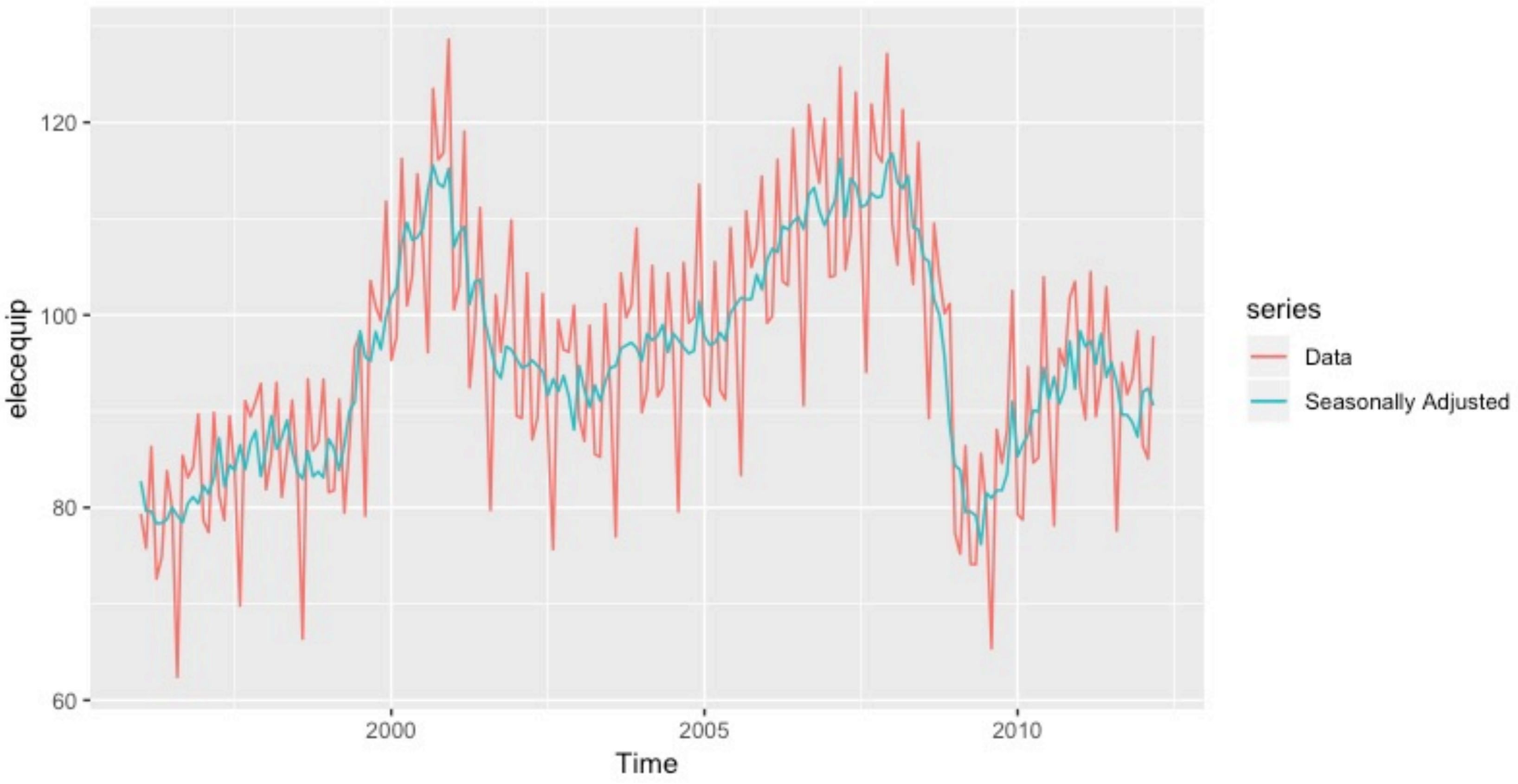
Seasonal adjustment

- Useful by-product of decomposition: an easy way to calculate seasonally adjusted data.
- seasonally adjusted data given by

$$y_t - S_t = T_t + R_t$$

Electrical equipment manufacturing example

- fit <- stl(elecequip, s.window=7)
- autoplot(elecequip, series="Data") + autolayer(seasadj(fit), series="Seasonally Adjusted")



Forecast with decomposition

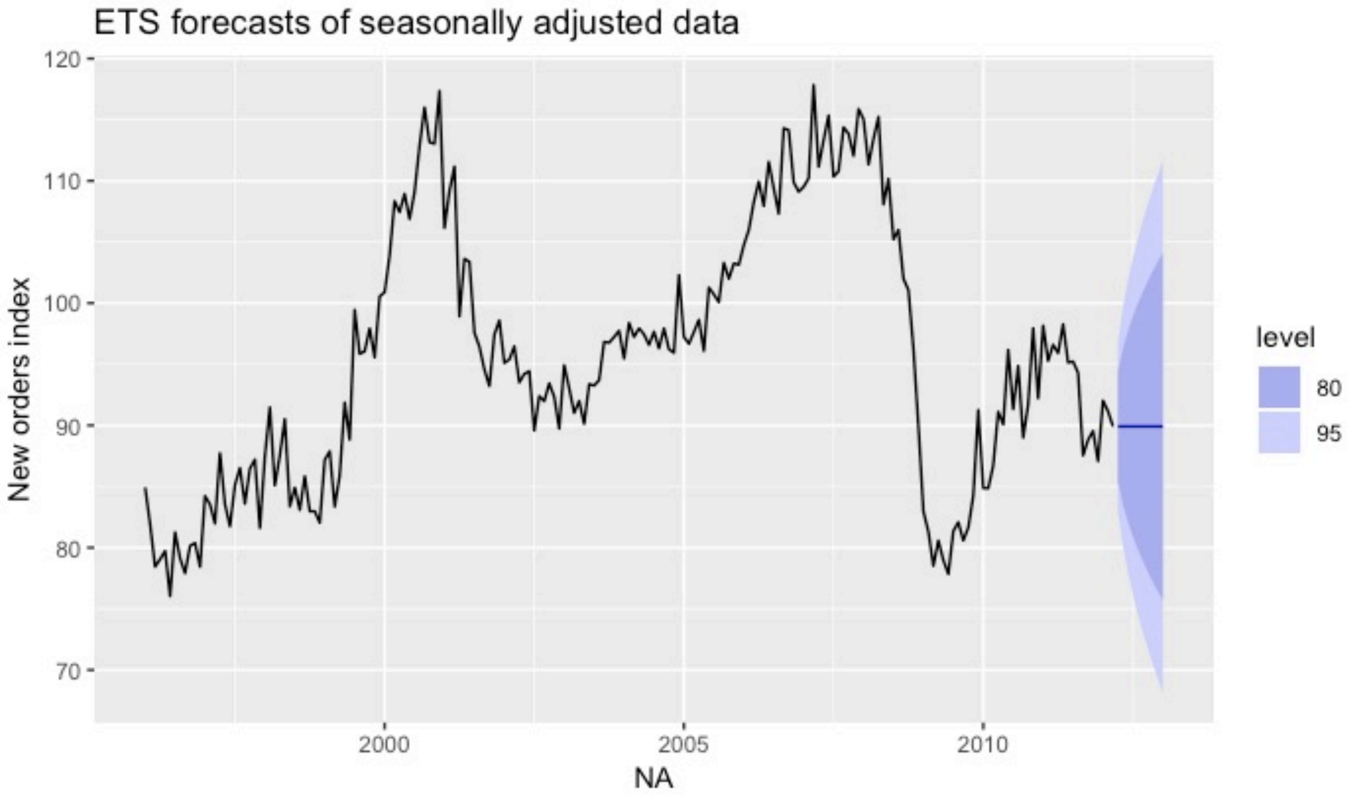
- Assuming an additive decomposition, the decomposed time series can be written as $y_t = \hat{S}_t + \hat{A}_t$,

where $\hat{A}_t = \hat{T}_t + \hat{R}_t$ is the seasonally adjusted component.

- To forecast a decomposed time series, we forecast the seasonal component, $\hat{S}_t$, and the seasonally adjusted component $\hat{A}_t$, separately.

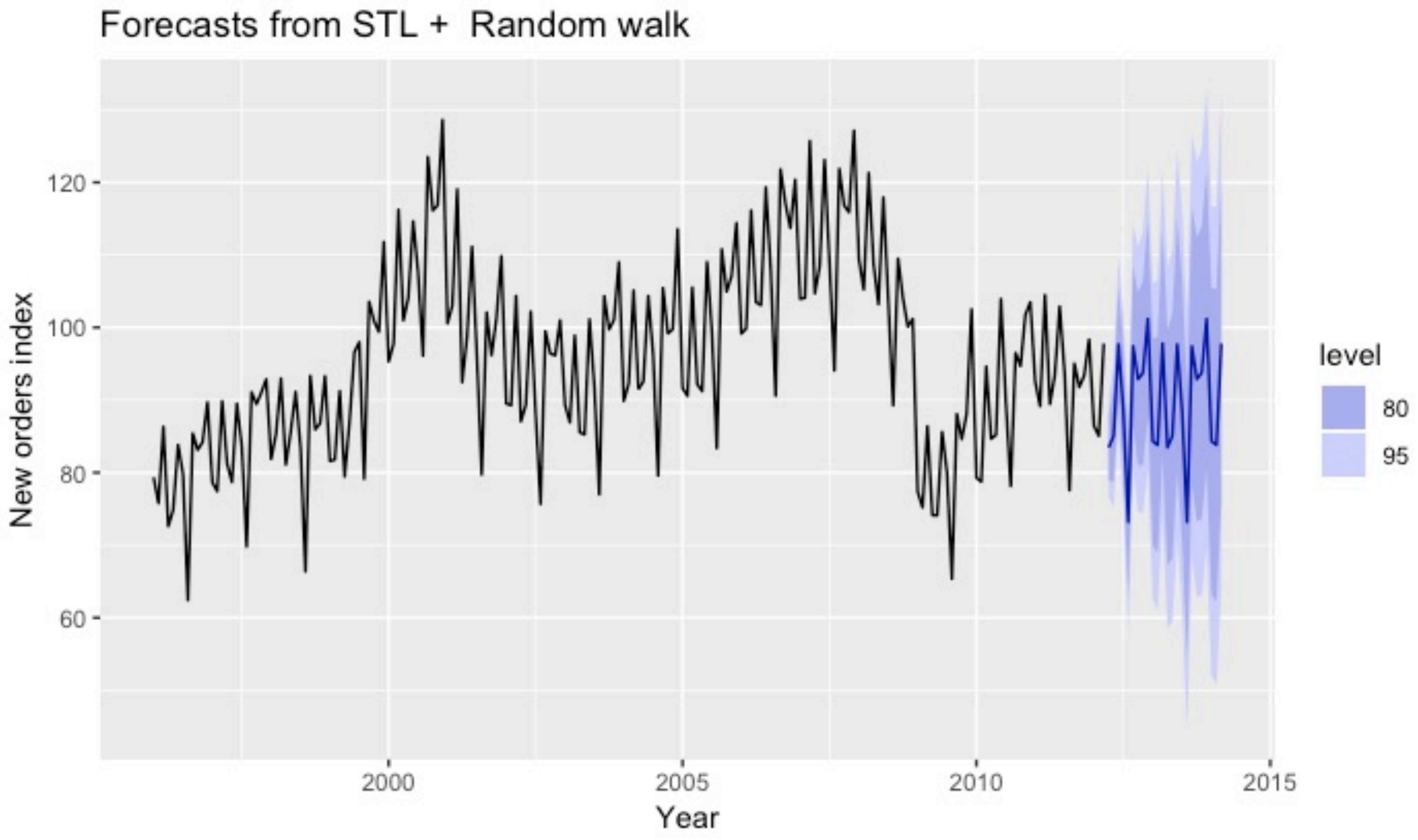
Forecast with decomposition

- fit <- stl(elecequip, t.window=13, s.window="periodic")
- fit %>% seasadj() %>% naive() %>%
- autoplot() + ylab("New orders index") + ggtitle("ETS forecasts of seasonally adjusted data")



Forecast with decomposition

- fit %>% forecast(method='naive') %>% autoplot() + ylab("New orders index") + xlab("Year")



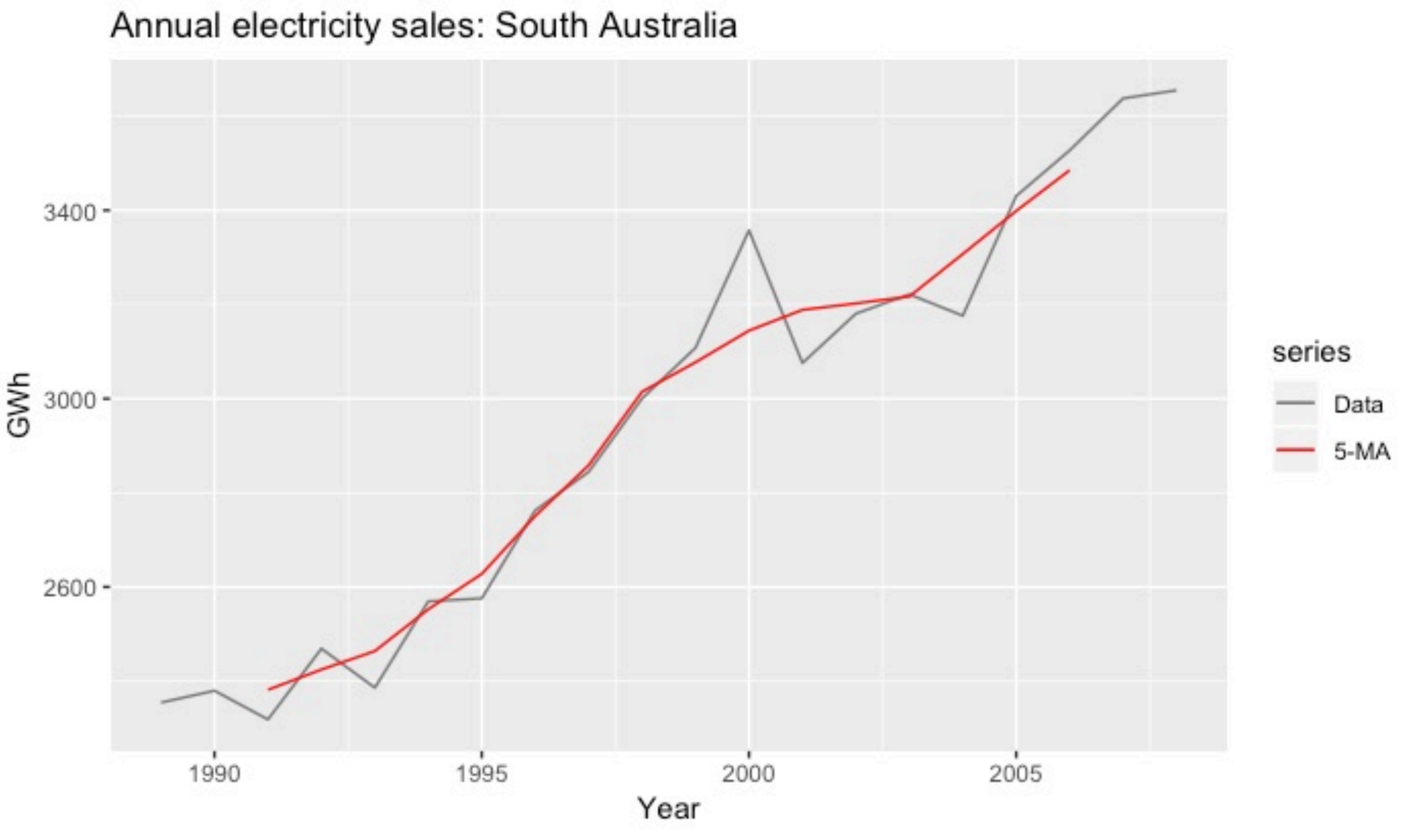
Moving average smoothing

- A moving average of order m can be written as

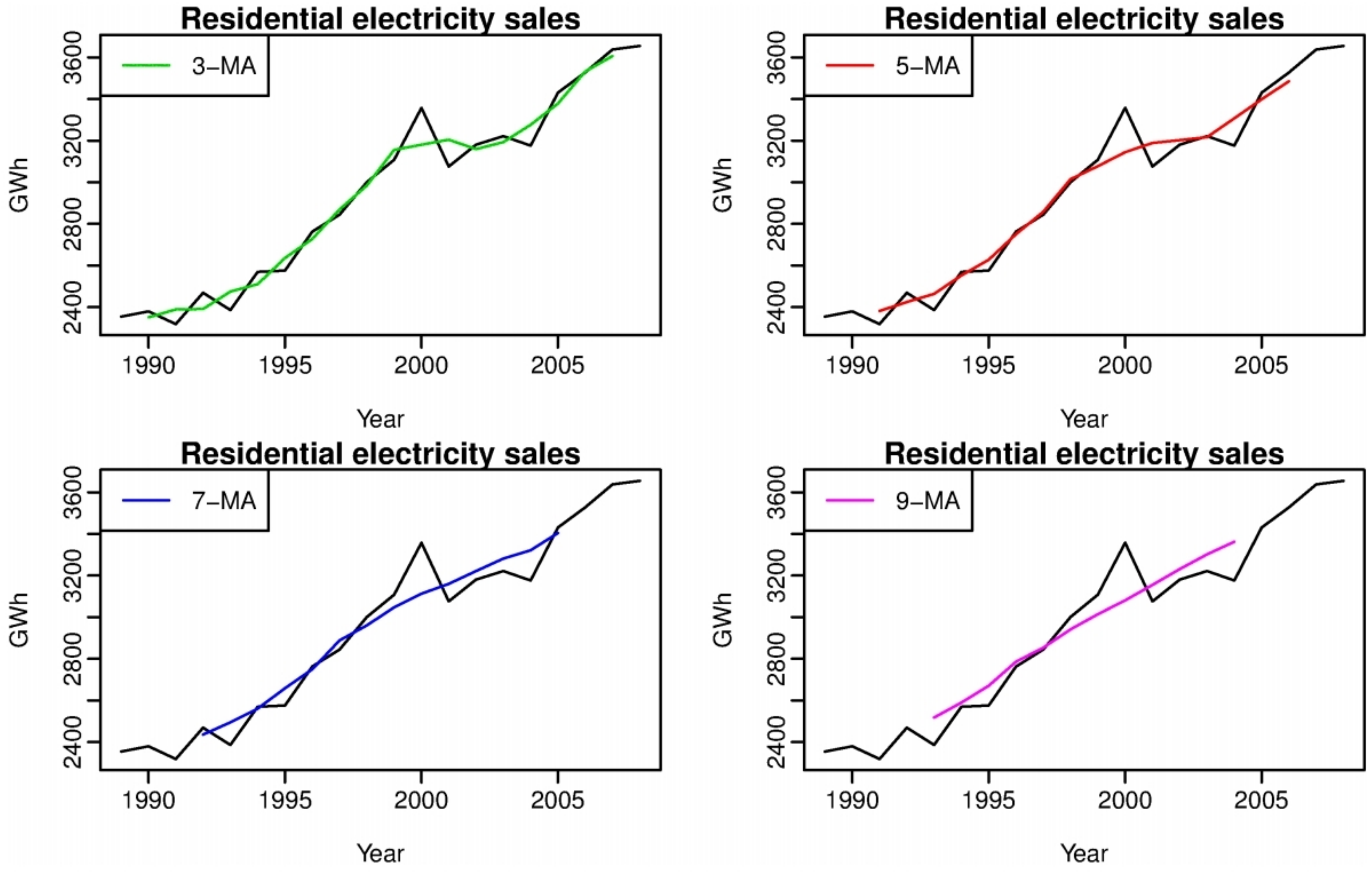
$$\hat{T}_t = \frac{1}{m}\sum_{j=-k}^{k} y_{t+j}, \qquad m = 2k + 1$$

Moving average smoothing

- **autoplot**(elecsales, series="Data") + **autolayer**(**ma**(elecsales,5), series="5-MA") + **xlab**("Year") + **ylab**("GWh") + **ggtitle**("Annual electricity sales: South Australia") + **scale_colour_manual**(values=**c**("Data"="grey50","5-MA"="red"), breaks=**c**("Data","5-MA"))



Moving average at different orders



Moving average of moving average

- One reason for doing this is to make an even-order moving average symmetric.
- A 2x4-MA can be written as

$$\frac{1}{2}[\frac{1}{4}(y_{t-2} + y_{t-1} + y_t + y_{t+1}) + \frac{1}{4}(y_{t-1} + y_t + y_{t+1} + y_{t+2})]$$

$$= \frac{1}{8}y_{t-2} + \frac{1}{4}y_{t-1} + \frac{1}{4}y_t + \frac{1}{4}y_{t+1} + \frac{1}{8}y_{t+2}.$$

- In general, an even order MA should be followed by an even order MA to make it symmetric. Similarly, an odd order MA should be followed by an odd order MA.

Estimating trend cycle with seasonal data

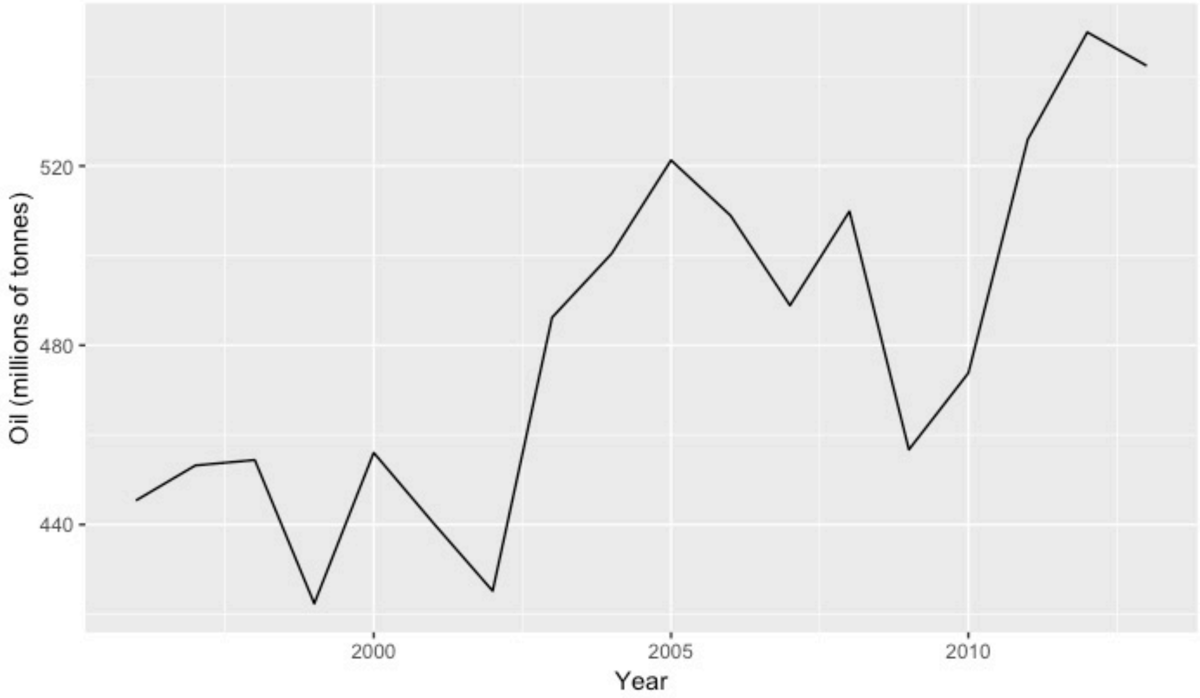
- The 2×4-MA can be used to estimate quarterly data.
- $2 \times m$-MA v.s. $m+1$-MA.
- If the seasonal period is even and of order m, use $2 \times m$-MA to estimate the trend cycle.
- If it's odd , then use m-MA to estimate the trend cycle.
- Thus 2×12-MA for monthly trend, and 7-MA for daily trend.

Electrical equipment manufacturing

- **autoplot**(elecequip, series="Data") + **autolayer**(**ma**(elecequip, 12), series="12-MA") + **xlab**("Year") + **ylab**("New orders index") + **ggtitle**("Electrical equipment manufacturing (Euro area)") + **scale_colour_manual**(values=**c**("Data"="grey","12-MA"="red"), breaks=**c**("Data","12-MA"))

Smoothing

- The simplest of the exponentially smoothing methods is naturally called **simple exponential smoothing** (SES). This method is suitable for forecasting data with no clear trend or seasonal pattern.
- oildata <- **window**(oil, start=1996) **autoplot**(oildata) + **ylab**("Oil (millions of tonnes)") + **xlab**("Year")



Models

- Previously we discussed the naive method:

$$\hat{y}_{T+h} = y_T \quad for\ all\ h$$

Thus the most recent obs. is the only important one for prediction purposes.

- The other method we mentioned, the average method:

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

Thus all obs. are of equal importance for forecasting.

- Naturally, intuition tells us that a better forecasting method should lie in between these two extremes. More recent obs. should be more important and assigned more weights.

SES model

- Forecasts are calculated using weighted averages with exponential decaying weights.

$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha) y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots + (1-\alpha)^T y_{1|0}$$

α is a smoothing parameter.

Obs.	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$
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

- The weight goes down exponentially. When α = 1, this reduces to the naive estimate.

Weighted average form

- By definition, the forecast at t + 1 equals to a weighted average between the most recent obs. y_t and the most recent forecast:

$$\hat{y}_{t+1|t} = \alpha y_t + (1-\alpha)\hat{y}_{t|t-1}.$$

$$\begin{aligned}
\hat{y}_{2|1} &= \alpha y_1 + (1-\alpha)\ell_0 \\
\hat{y}_{3|2} &= \alpha y_2 + (1-\alpha)\hat{y}_{2|1} \\
\hat{y}_{4|3} &= \alpha y_3 + (1-\alpha)\hat{y}_{3|2} \\
&\vdots \\
\hat{y}_{T+1|T} &= \alpha y_T + (1-\alpha)\hat{y}_{T|T-1}
\end{aligned}$$

Weighted average form

- By substitution we have

$$
\begin{aligned}
\hat{y}_{3|2} &= \alpha y_2 + (1-\alpha)\left[\alpha y_1 + (1-\alpha)\ell_0\right] \\
&= \alpha y_2 + \alpha(1-\alpha) y_1 + (1-\alpha)^2 \ell_0 \\
\hat{y}_{4|3} &= \alpha y_3 + (1-\alpha)[\alpha y_2 + \alpha(1-\alpha) y_1 + (1-\alpha)^2 \ell_0] \\
&= \alpha y_3 + \alpha(1-\alpha) y_2 + \alpha(1-\alpha)^2 y_1 + (1-\alpha)^3 \ell_0 \\
&\vdots \\
\hat{y}_{T+1|T} &= \sum_{j=0}^{T-1} \alpha(1-\alpha)^j y_{T-j} + (1-\alpha)^T \ell_0.
\end{aligned}
$$

Component form

- Time series components consist of level, trend, and seasonal component.
- For simple exponential smoothing, only level I_t is needed

$$\begin{aligned} \text{Forecast equation} \quad & \hat{y}_{t+1|t} = l_t \\ \text{Smoothing equation} \quad & l_t = \alpha y_t + (1-\alpha) l_{t-1}, \end{aligned}$$

where I_t is the level of the time series at time t.

Error correction form

- We can rearrange the component form to get the following, where e_t is the one step within-sample forecast error at time t.

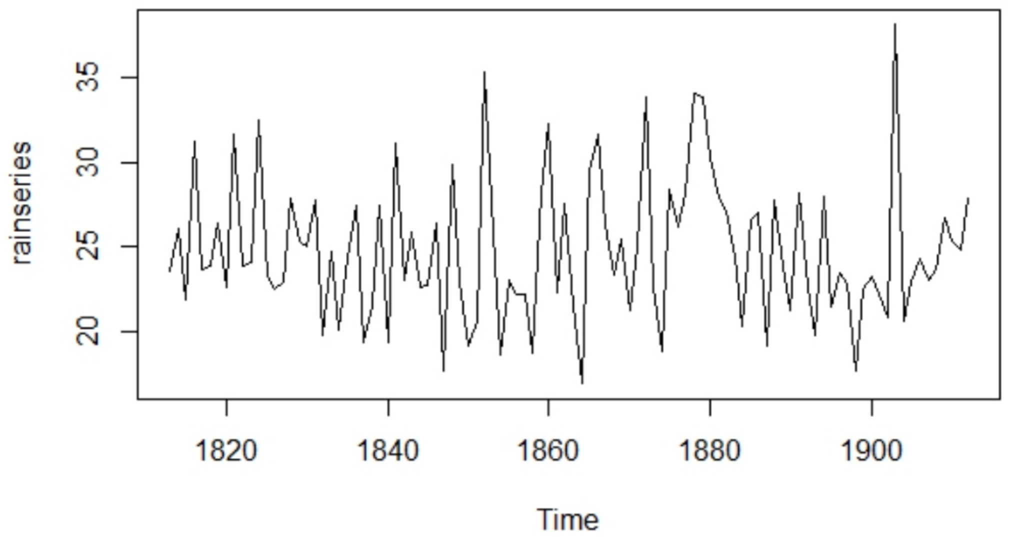
$$
\begin{aligned}
l_t &= l_{t-1} + \alpha(y_t - l_{t-1}) \\
&= l_{t-1} + \alpha e_t
\end{aligned}
$$

$$\text{where } e_t = y_t - l_{t-1} = y_t - \hat{y}_{t|t-1} \text{ for } t = 1, \cdots, T.$$

- If the error at time t is negative, then the level at t − 1 is over-estimated. The new level is then the previous level adjusted downwards.
- α controls the smoothness of the level forecast, smaller implies smoother.
- We can obtain the optimal α via minimizing the SSE

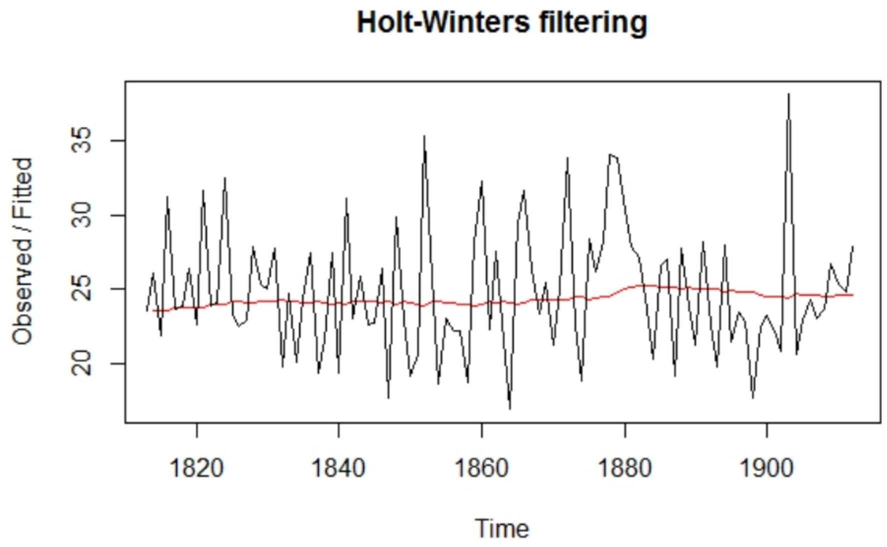
Simple exponential smoothing example

- If the time series contains no trend and seasonality, then we can use simple linear smoothing to make short term forecast.
- Degree of smoothing is controlled by alpha.
- The following figure shows the annual rainfall by inches for London.
- rain= scan("http://robjhyndman.com/tsdldata/hurst/precip1.dat",skip=1)
- rainseries <- ts(rain,start=c(1813))
- plot.ts(rainseries)



Simple exponential smoothing example

- From this plot, the mean stays at constant level, and doesn't have much seasonality.
- Use function HoltWinters(). For simple exponential smoothing, we set the parameters beta=FALSE and gamma=FALSE.
- The following figure shows the fitted line over original data.
- rainseriesforecasts <- HoltWinters(rainseries, beta=FALSE, gamma=FALSE)



Simple exponential smoothing example

- For each year, we can compute the error between the observed value and fitted value, thus the in sample sum of square errors(for all the years we have data with)
- rainseriesforecasts$SSE
- [1] 1828.855
- For simple exponential smoothing, we use the first value (23.56) as the initial value. This can be specified in HoltWinters using "l.start" option.
- rainseriesforecasts2<-HoltWinters(rainseries, beta=FALSE, gamma=FALSE, l.start=23.56)
- forecast(rainseriesforecasts2, h=2)

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
1913	24.67819	19.17493	30.18145	16.26169	33.09470
1914	24.67819	19.17333	30.18305	16.25924	33.09715