Smoothing Methods

Smoothing is "data driven"

- Regression methods assume underlying unchanging structure (linear, exponential, polynomial)
- Smoothing derives forecasts based directly on the data alone (e.g. averaging), with no mathematical structural assumptions
- Suitable where the components (trend, seasonality) change over time

Simple moving average (MA)

Set window width "*w*" – take average of the *w* values.

For centered moving average, window is centered around forecast point

- For w=5, the forecast for t_3 averages the values t_1 … t_5
- Not useful for future forecasts

For future forecasts, use "trailing average" = the value being forecast is at the end of the window

Choosing window width

Goal is to suppress seasonality and noise

Typically choose window width = season length

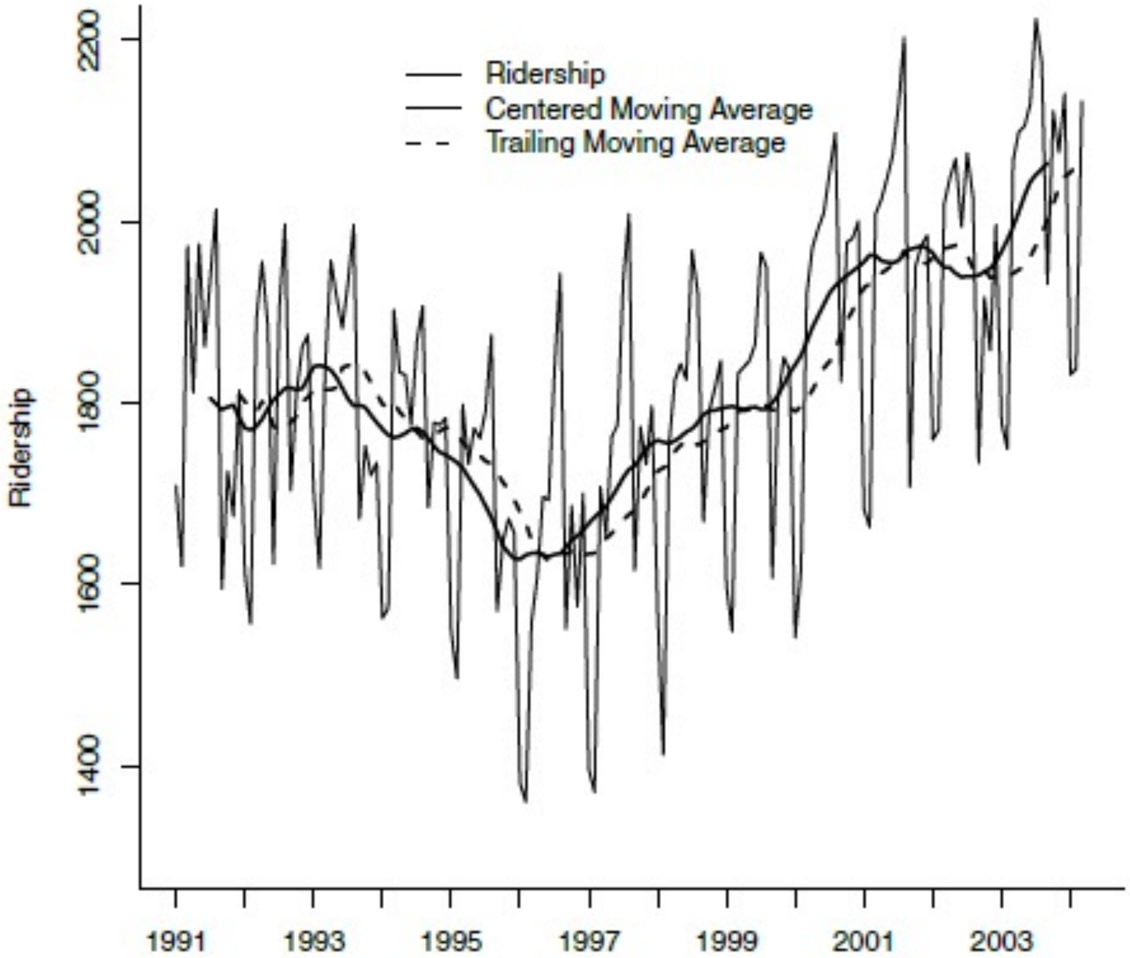
In Excel:

Add Trendline > Moving Average

Moving Average Functions

```
library(zoo)

# centered moving average with window order = 12

ma.centered <- ma(ridership.ts, order = 12)

# trailing moving average with window k = 12
# in rollmean(), use argument align = right to calculate a
# trailing moving average.

ma.trailing <- rollmean(ridership.ts, k = 12, align =
"right")
```

Amtrak Ridership: Moving Average Smoothing Window W = 12



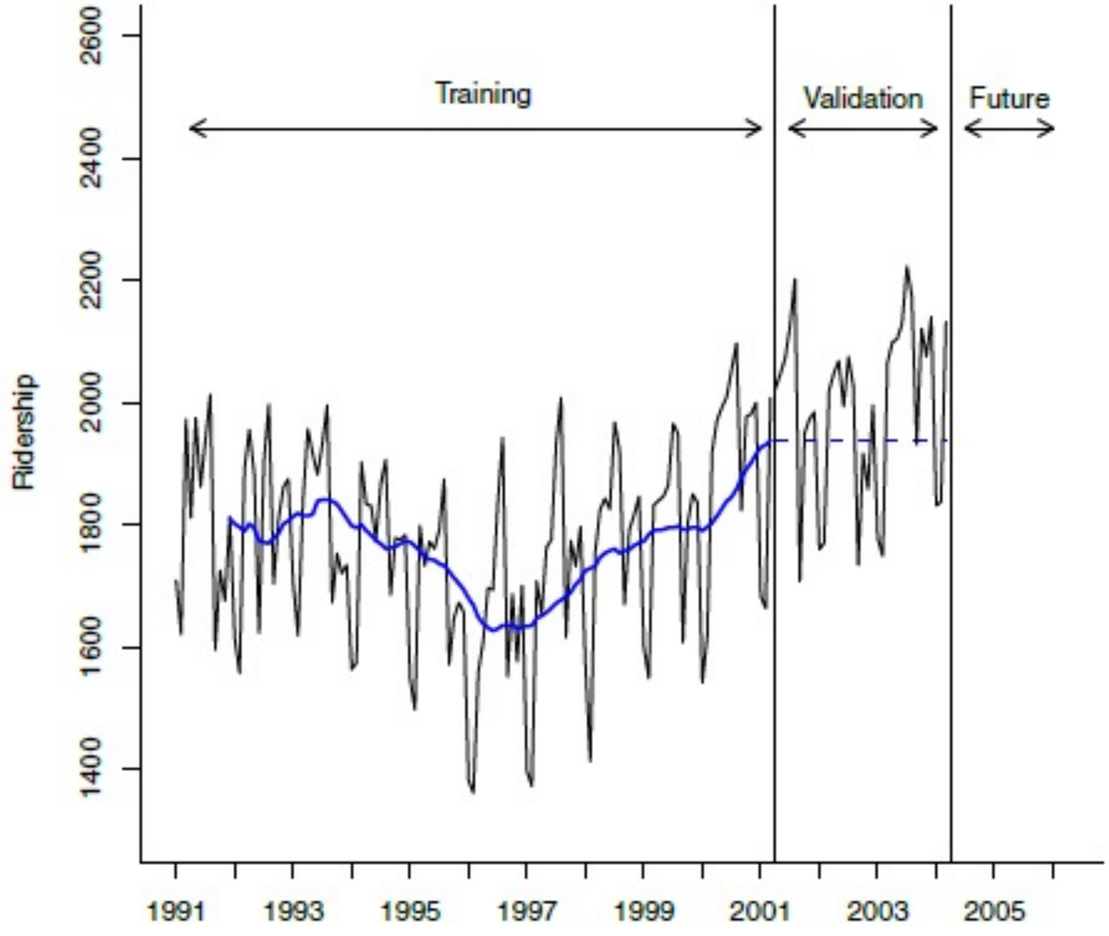
Plotting Code

```
# generate a plot
plot(ridership.ts, ylim = c(1300, 2200), ylab =
     "Ridership",
xlab = "Time", bty = "l", xaxt = "n",
xlim = c(1991,2004.25), main = "")
axis(1, at = seq(1991, 2004.25, 1), labels =
format(seq(1991, 2004.25, 1)))
lines(ma.centered, lwd = 2)
lines(ma.trailing, lwd = 2, lty = 2)
legend(1994,2200, c("Ridership","Centered Moving
     Average", "Trailing Moving Average"),
     lty=c(1,1,2), lwd=c(1,2,2), bty = "n")
```

MA forecast, and checking it in the validation period:

```
# partition the data

nValid <- 36
nTrain <- length(ridership.ts) - nValid
train.ts <- window(ridership.ts, start = c(1991, 1), end =
     c(1991, nTrain))
valid.ts <- window(ridership.ts, start = c(1991, nTrain + 1),
     end = c(1991, nTrain + nValid))

# moving average on training

ma.trailing <- rollmean(train.ts, k = 12, align = "right")

# obtain the last moving average in the training period

last.ma <- tail(ma.trailing, 1)

# create forecast based on last MA

ma.trailing.pred <- ts(rep(last.ma, nValid), start = c(1991,
     nTrain + 1), end = c(1991, nTrain + nValid), freq = 12)
```

Validation forecast is the last moving average from training period



Moving Average for Forecasting

Shortcomings

- Suppresses seasonality, but does not forecast seasonal component
- Lags behind trends
- Thus, simple Moving Average useful only for series that lack trend and seasonality

Coping with these shortcomings

- Use regression model to de-trend and de-seasonalize
- Use Moving Average to forecast the de-trended and de-seasonalized series
- Add trend and seasonality back to the forecast

Simple exponential smoothing

Like MA, except use weighted average of all past values, instead of simple average in a window

Forecast at time t+1:

$$F_{t+1} = \alpha Y_t + \alpha(1 - \alpha)Y_{t-1} + \alpha(1 - \alpha)^2 Y_{t-2} + \ldots$$

Equivalent to

$$F_{t+1} = F_t + \alpha E_t$$

E is forecast error at time *t*

Smoothing parameter α

Simple exponential smoother corrects based on error

- If last period forecast was too high, next period is adjusted down
- If last period forecast was too low, next period is adjusted up

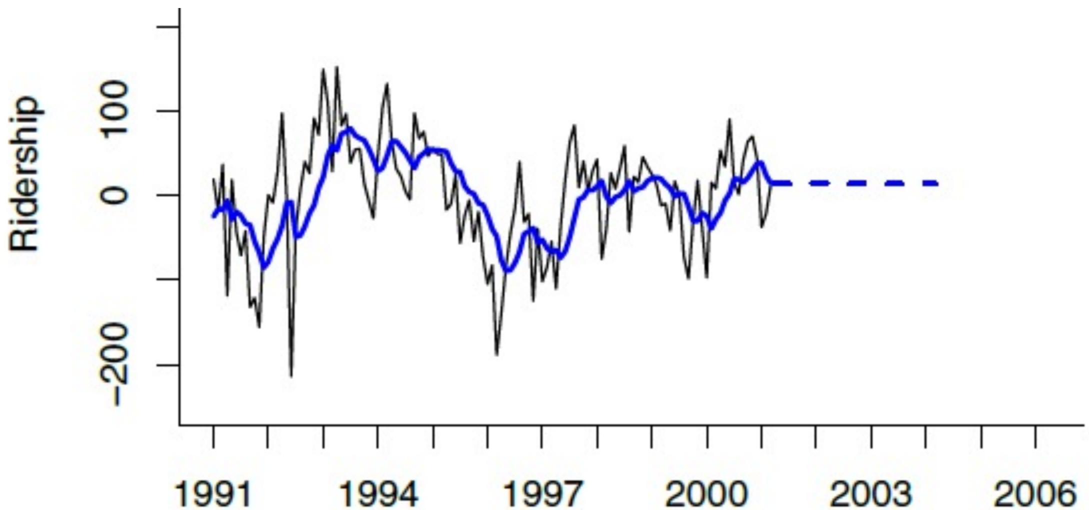
Amount of correction depends on value of α

- Value close to 1 > fast learning, close to 0 > low learning

Output for simple exponential smoothing applied to residuals from regression model:

```
# get residuals
residuals.ts <- train.lm.trend.season$residuals

# run simple exponential smoothing
# use ets() with model = "ANN" (additive error (A),
# no trend (N), no seasonality (N))
# and alpha = 0.2 to fit simple exponential smoothing.

ses <- ets(residuals.ts, model = "ANN", alpha = 0.2)
ses.pred <- forecast(ses, h = nValid, level = 0)
```

y-axis is residual from regression model



Moving average and simple exponential smoothing can be used only when there is no trend or seasonality. When those features are present:

- One solution is to remove those components via regression
- Another is to use advanced exponential smoothing, which can capture trend and seasonality
- Double-exponential smoothing used for series with a trend

Double exponential smoothing

Incorporates trend

K-step ahead forecast is derived from the level (L) and trend (T) estimates at time t

$F_{t+k} = L_t + kT_t$

where

$L_t = \alpha Y_t + (1-\alpha)(L_{t-1} + T_{t-1})$

$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}$

Holt Winters exponential smoothing

- Extension of double exponential smoothing
- Incorporate both trend and seasonality

Holt Winters forecast for time t+k

Adds seasonality to double exponential

For M seasons (e.g. M=7 for weekly), forecast is

$$F_{t+k} = (L_t + kT_t)S_{t+k-M}$$

Where L = level, T = trend, S = season

Updating L, T and S

Like eq. for double exponential, except for seasonal adjustment term

$$L_t = \frac{\alpha Y_t}{S_{t-M}} + (1-\alpha)(L_{t-1} + T_{t-1}),$$

Like double exponential equation

$$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1},$$

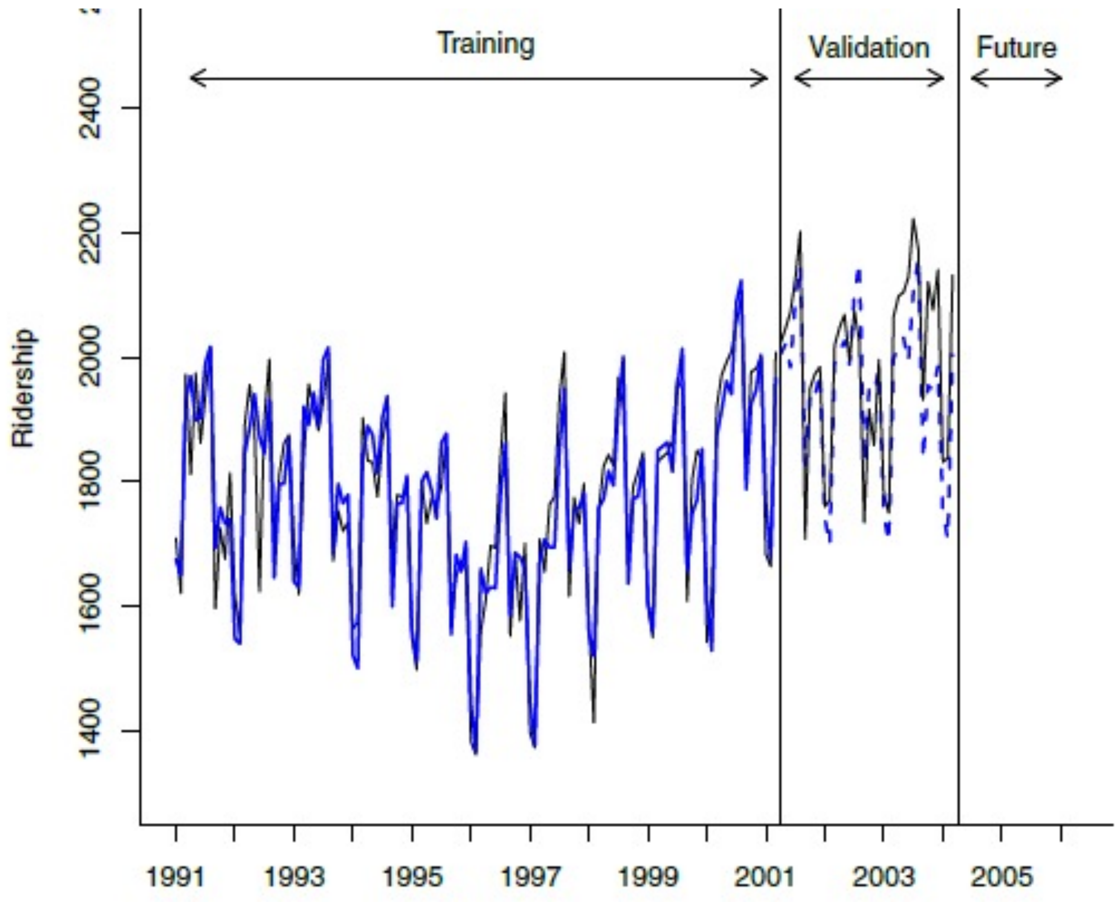
Equation to update seasonal index

$$S_t = \frac{\gamma Y_t}{L_t} + (1-\gamma)S_{t-M}.$$

Holt Winters predictions:

```
# run Holt-Winters exponential smoothing
# use ets() with option model = "MAA" to fit Holt-
# Winter's exponential smoothing
# with multiplicative error, additive trend, and
# additive seasonality.

hwin <- ets(train.ts, model = "MAA")

# create predictions

hwin.pred <- forecast(hwin, h = nValid, level = 0)
```

Holt-Winters Predictions



Summary

- Smoothing methods rely on local data, not mathematical structure
- Simple smoothing does not account for trend and seasonality, but can be combined with model-based forecasts to improve the forecast
- Holt-Winters smoothing incorporates seasonality and trend