Exponential smoothing (Lecture 7)

Forecasting: principles and practice

book by Rob Hyndman and George Athanasopoulos

Exponential smoothing

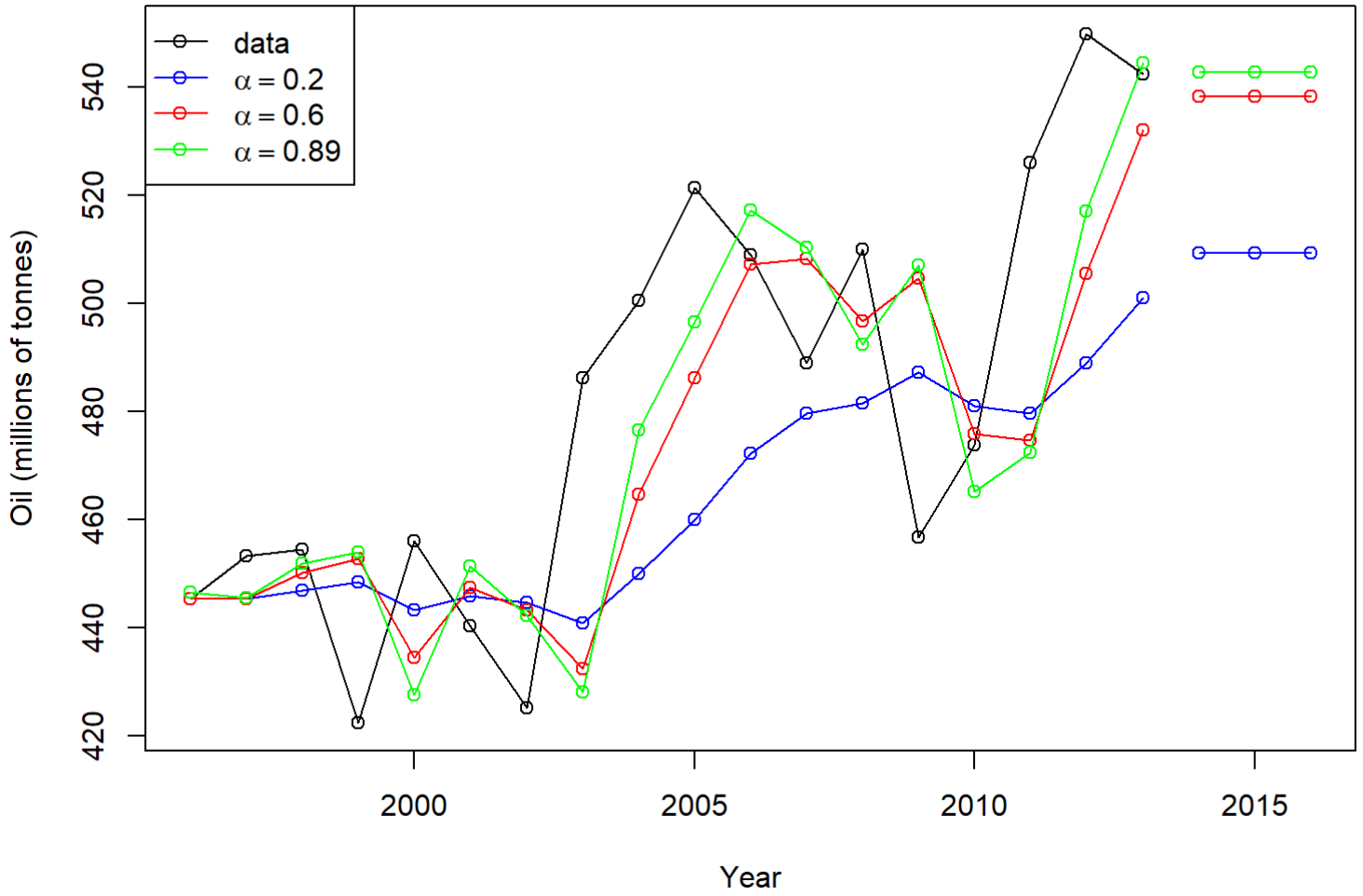
```
library(fpp2)
library(seasonal)
```

Simple exponential smoothing in Weighted average form

The forecast at time $t+1$ is equal to a weighted average between the most recent observation y_t and the most recent forecast $\hat{y}_{t|t-1}$,

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

for $t = 1, \ldots, T$, where $0 \leq \alpha \leq 1$ is the smoothing parameter.


data
α = 0.2
α = 0.6
α = 0.89
Oil (millions of tonnes)
420
440
460
480
500
520
540
2000
2005
2010
2015
Year

Component form

For simple exponential smoothing the only component included is the level, ℓ_t. (Other methods considered later in this lecture may also include a trend (slope) b_t and seasonal component s_t.)

Component form representations of exponential smoothing methods comprise a forecast equation and a smoothing equation for each of the components included in the method:

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

where ℓ_t is the level (or the smoothed value) of the series at time t.

The forecast equation shows that the forecasted value at time $t+1$ is the estimated level at time t. The smoothing equation for the level (usually referred to as the level equation) gives the estimated level of the series at each period t.

Applying the forecast equation for time T gives, $\hat{y}_{T+1|T} = \ell_T$, the most recent estimated level.

If we replace ℓ_t by $\hat{y}_{t+1|t}$ and ℓ_{t-1} by $\hat{y}_{t|t-1}$ in the smoothing equation, we will recover the weighted average form of simple exponential smoothing.

Multi-horizon Forecasts

Simple exponential smoothing has a flat forecast function, and therefore for longer forecast horizons,

$$\hat{y}_{T+h|T} = \hat{y}_{T+1|T} = \ell_T, \qquad h = 2, 3, \dots.$$

Remember these forecasts will only be suitable if the time series has no trend or seasonal component.

Optimization

Similar to regression model, the unknown parameters and the initial values for any exponential smoothing method can be estimated by minimizing the SSE.

The errors are specified as $e_t = y_t - \hat{y}_{t|t-1}$ for $t = 1, \ldots, T$ (the one-step-ahead within-sample forecast errors).

$$\text{SSE} = \sum_{t=1}^{T} (y_t - \hat{y}_{t|t-1})^2 = \sum_{t=1}^{T} e_t^2.$$

This involves a non-linear minimization problem and we need to use an optimization tool to perform this.

In the forecast package, this is done with the function ses function in the forecast package (https://www.rdocumentation.org/packages/forecast/versions/8.1/topics/ses) or page 97 of the forecast package documentation (https://cran.r-project.org/web/packages/forecast/forecast.pdf)

Example: Oil Production

```
library(fpp2)
oildata <- window(oil, start=1996)
# Estimate parameters
fc <- ses(oildata, h=5)
# Accuracy of one-step-ahead training errors over period 1--12
round(accuracy(fc),2)
```

```
##                ME  RMSE   MAE MPE MAPE MASE  ACF1
## Training set 6.4 28.12 22.26 1.1 4.61 0.93 -0.03
```

```
#>              ME RMSE  MAE MPE MAPE MASE  ACF1
#> Training set 6.4 28.1 22.3 1.1 4.61 0.93 -0.03
```

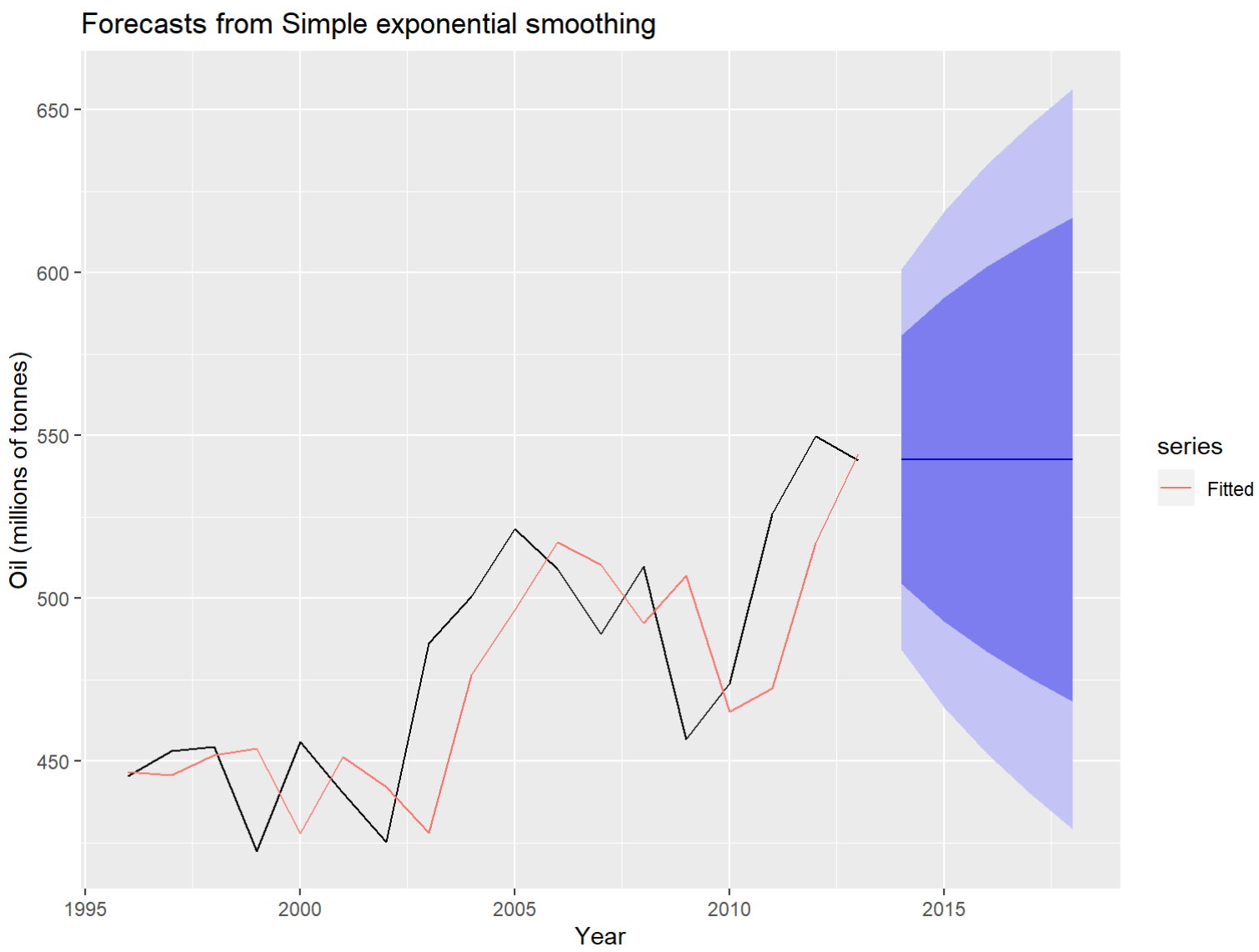
This gives parameters $\alpha = 0.83$ and $l_0 = 446.58$, obtained by minimizing SSE over periods $t = 1, 2, \ldots, 12$, subject to the restriction that $0 \leqslant \alpha \leqslant 1$.

Note that these values could be shown by running on R-prompt: >summary(fc) i.e.:

```
summary(fc)
```

This is plotted in the next slide

```
autoplot(fc) +
  forecast::autolayer(fitted(fc), series="Fitted") +
  ylab("Oil (millions of tonnes)") + xlab("Year")
```


Forecasts from Simple exponential smoothing
Oil (millions of tonnes)
650
600
550
500
450
1995
2000
2005
2010
2015
Year
series
Fitted

Holt's linear trend method

Extended simple exponential smoothing to allow forecasting of data with a trend. This method involves a forecast equation and two smoothing equations (one for the level and one for the trend):

$$\begin{aligned} \text{Forecast equation} \quad & \hat{y}_{t+h|t} = \ell_t + hb_t \\ \text{Level equation} \quad & \ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ \text{Trend (slope) equation} \quad & b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \end{aligned}$$

where 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, α is the smoothing parameter for the level, $0 \leq \alpha \leq 1$ and β^* is the smoothing parameter for the trend (slope), $0 \leq \beta^* \leq 1$ (we denote this as β^* instead of β.)

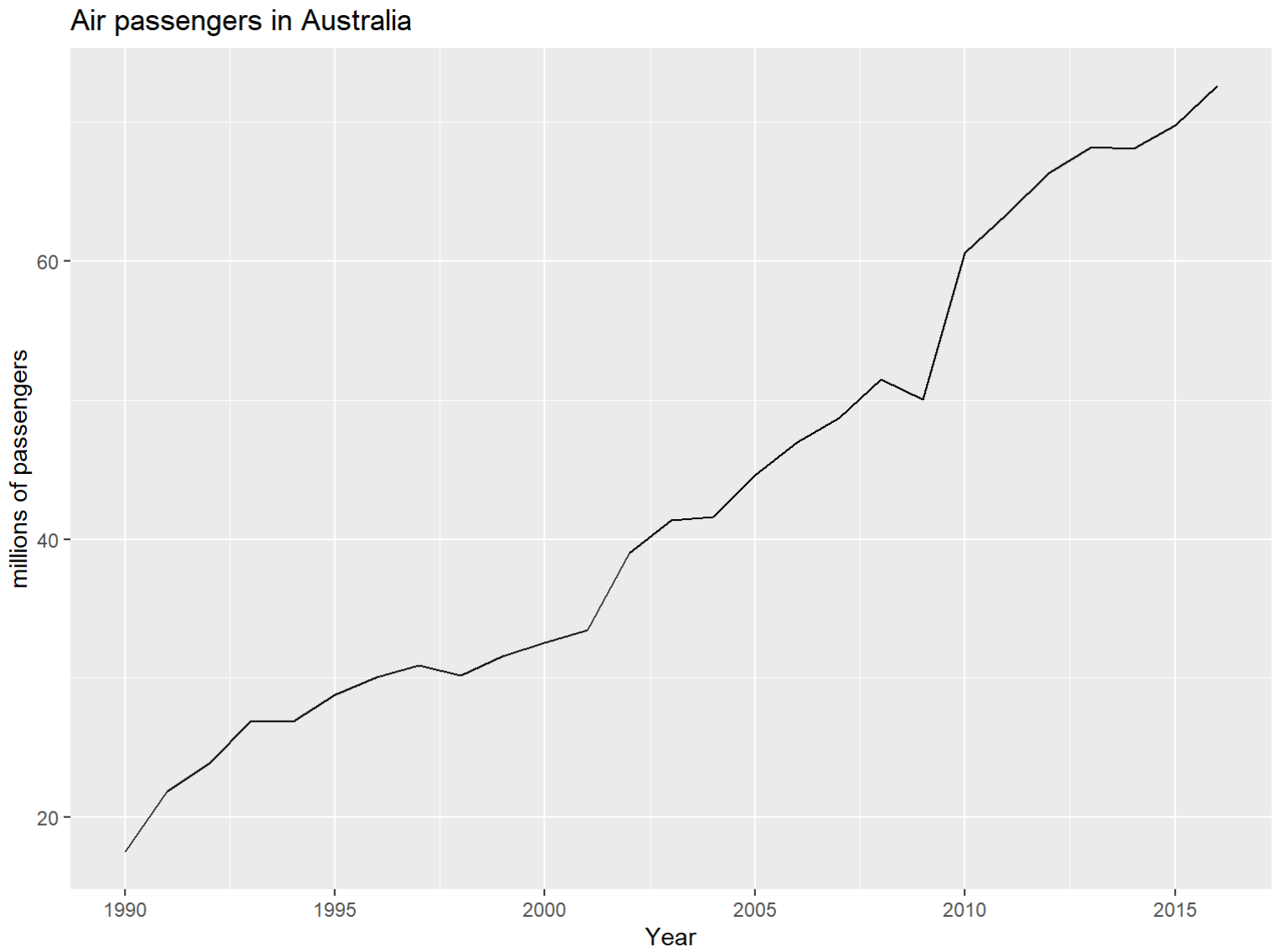
As with simple exponential smoothing, the level equation here shows that ℓ_t is a weighted average of observation y_t and the within-sample one-step-ahead forecast for time t, here given by $\ell_{t-1} + b_{t-1}$. The trend equation shows that b_t is a weighted average of the estimated trend (slope) at time t based on $\ell_t - \ell_{t-1}$ and b_{t-1}, the previous estimate of the trend (slope).

NOTE: In this chapter, the authors refer to b_t, which is the trend (slope) at t, only as "trend" in Chapter 6.

The forecast function is no longer flat but trending. The h-step-ahead forecast is equal to the last estimated level plus h times the last estimated trend value. Hence the forecasts are a linear function of h.

Example: Air Passengers

```
air <- window(ausair, start=1990)
autoplot(air) +
ggtitle("Air passengers in Australia") +
xlab("Year") + ylab("millions of passengers")
```


Air passengers in Australia
millions of passengers
60
40
20
1990
1995
2000
2005
2010
2015
Year

Holt's linear trend method is used in the holt function in r

```
fc <- holt(air, h=5)
```

```
summary(fc)
```

```
##
## Forecast method: Holt's method
##
## Model Information:
## Holt's method
##
## Call:
##  holt(y = air, h = 5)
##
##   Smoothing parameters:
##     alpha = 0.8302
##     beta  = 1e-04
##
##   Initial states:
##     l = 15.5715
##     b = 2.1017
##
##   sigma:  2.3645
##
##      AIC     AICc      BIC
## 141.1291 143.9863 147.6083
##
## Error measures:
##                       ME     RMSE     MAE        MPE     MAPE      MASE
## Training set 0.008359331 2.182343 1.52892 -0.3244107 3.820787 0.6654839
##                     ACF1
## Training set -0.01335362
##
## Forecasts:
##      Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
```

```
## 2017       74.60130 71.57106 77.63154 69.96695 79.23566
## 2018       76.70304 72.76440 80.64169 70.67941 82.72668
## 2019       78.80478 74.13092 83.47864 71.65673 85.95284
## 2020       80.90652 75.59817 86.21487 72.78810 89.02494
## 2021       83.00826 77.13343 88.88310 74.02348 91.99305
```

Among others, it shows:

Smoothing parameters: alpha = 0.8 ; beta = 1e-04

Initial states: ℓ = 15.8811 ; b = 2.0612

Damped trend methods

The forecasts generated by Holt's linear method display a constant trend (increasing or decreasing) indefinitely into the future.

Empirical evidence indicates that these methods tend to over-forecast, especially for longer forecast horizons. Motivated by this, a dampening parameter is introduced so that the trend approaches a flat line some time in the future.

In conjunction with the smoothing parameters α and β^* (with values between 0 and 1 as in Holt's method), this method also includes a dampening parameter $0 < \phi < 1$:

$$\begin{aligned}
\text{Forecast equation} \quad & \hat{y}_{t+h|t} = \ell_t + (\phi + \phi^2 + \cdots + \phi^h)b_t \\
\text{Level equation} \quad & \ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \phi b_{t-1}) \\
\text{Trend (slope) equation} \quad & b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}
\end{aligned}$$

Example: Air Passengers

```
fc <- holt(air, h=15)
fc2 <- holt(air, damped=TRUE, phi = 0.9, h=15)
autoplot(air) +
  forecast::autolayer(fc$mean, series="Holt's method") +
  forecast::autolayer(fc2$mean, series="Damped Holt's method") +
  ggtitle("Forecasts from Holt's method") +
  xlab("Year") + ylab("Air passengers in Australia (millions)") +
  guides(colour=guide_legend(title="Forecast"))
```

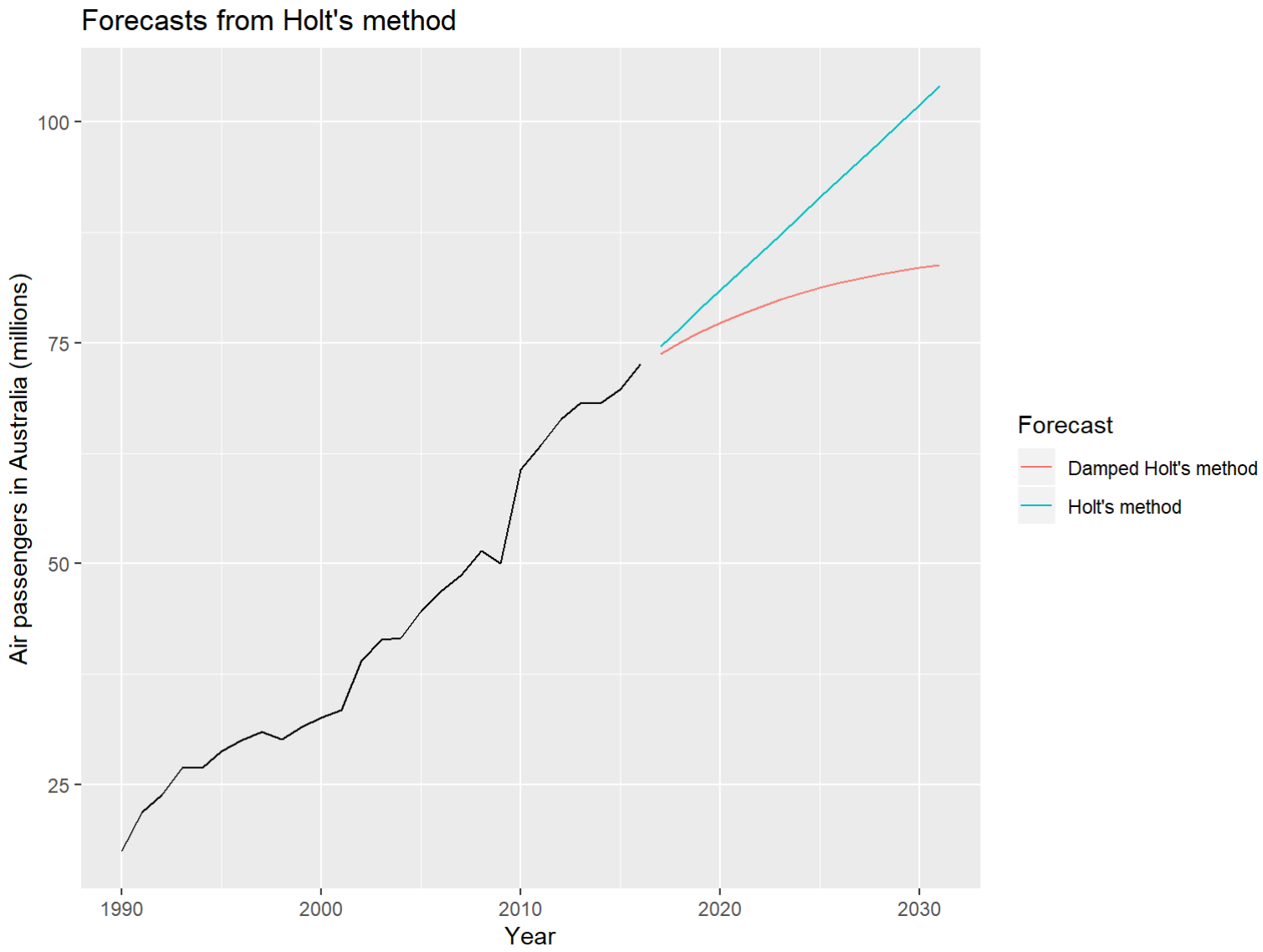
In this example, the dampening parameter is set to a relatively low number of ($\phi = 0.90$) to exagerate the effect of dampening for comparison.

ϕ is usually estimated along with the other parameters if it is not specified, i.e., fc2 <- holt(air, damped=TRUE, h=15) which in this case will be estimated at $\phi = 0.98$.

In practice, ϕ is rarely less than 0.8 as the dampening has a very strong effect for smaller values. Values of ϕ close to 1 will mean that a damped model is not able to be distinguished from a non-damped model. For these reasons, we usually restrict ϕ to a minimum of 0.8 and a maximum of 0.98.

The figure is shown in the next slide

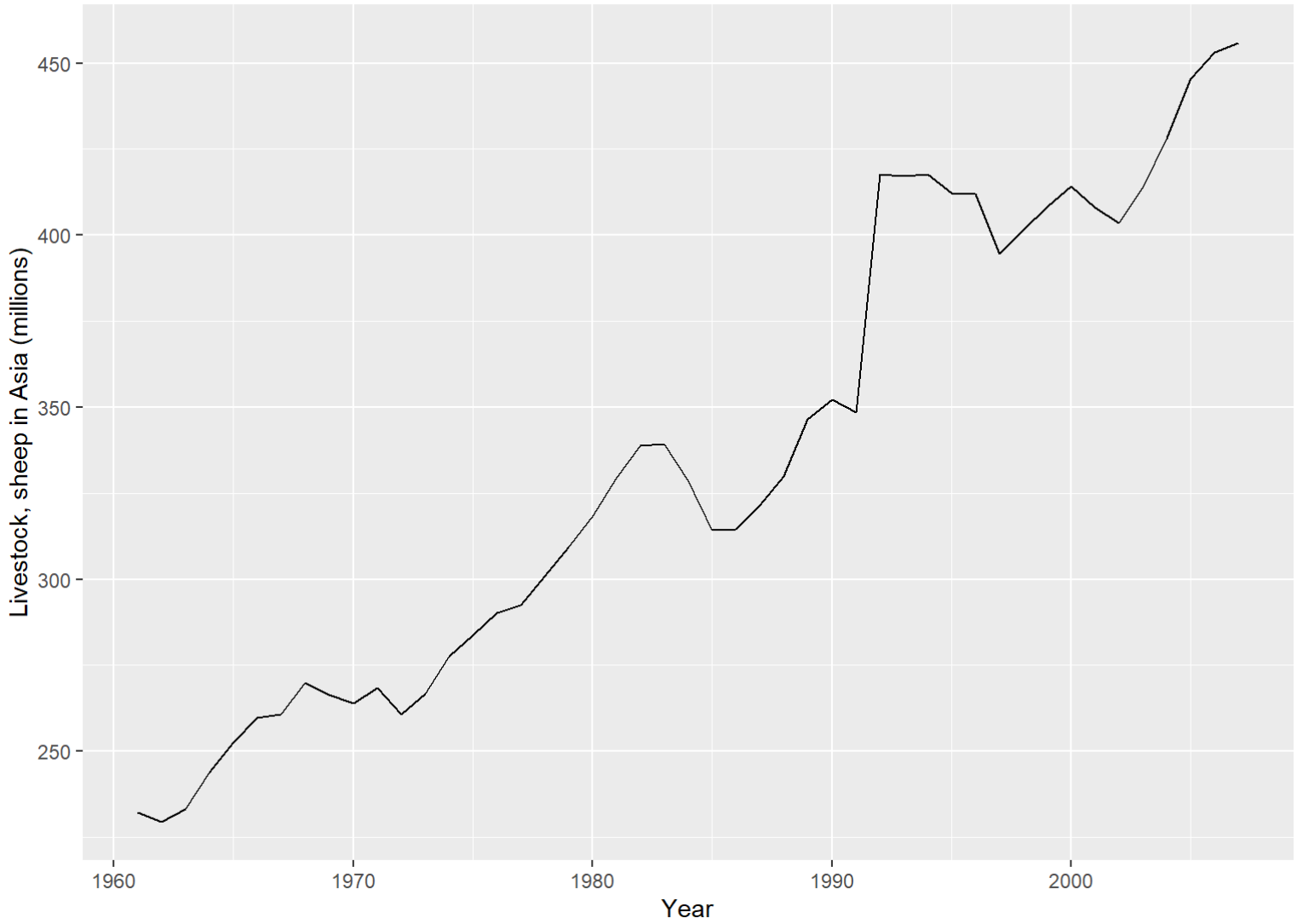
Note from the figure that ϕ dampens the trend so that it approaches a constant some time in the future.


Forecasts from Holt's method
Air passengers in Australia (millions)
100
75
50
25
1990
2000
2010
2020
2030
Year
Forecast
Damped Holt's method
Holt's method

Example: Sheep in Asia

In this example, the forecasting performance of the three exponential smoothing methods that we have considered so far in forecasting the sheep livestock population in Asia.

```
autoplot(livestock) +
xlab("Year") + ylab("Livestock, sheep in Asia (millions)")
```


Livestock, sheep in Asia (millions)
450
400
350
300
250
1960
1970
1980
1990
2000
Year

The time series cross-validation is used to compare the one-step forecast accuracy of the three methods

```
e1 <- tsCV(livestock, ses, h=1)
e2 <- tsCV(livestock, holt, h=1)
e3 <- tsCV(livestock, holt, damped=TRUE, h=1)
# Compare MSE:
mean(e1^2, na.rm=TRUE)
#> [1] 178
mean(e2^2, na.rm=TRUE)
#> [1] 173
mean(e3^2, na.rm=TRUE)
#> [1] 163
# Compare MAE:
mean(abs(e1), na.rm=TRUE)
#> [1] 8.53
mean(abs(e2), na.rm=TRUE)
#> [1] 8.8
mean(abs(e3), na.rm=TRUE)
#> [1] 8.02
```

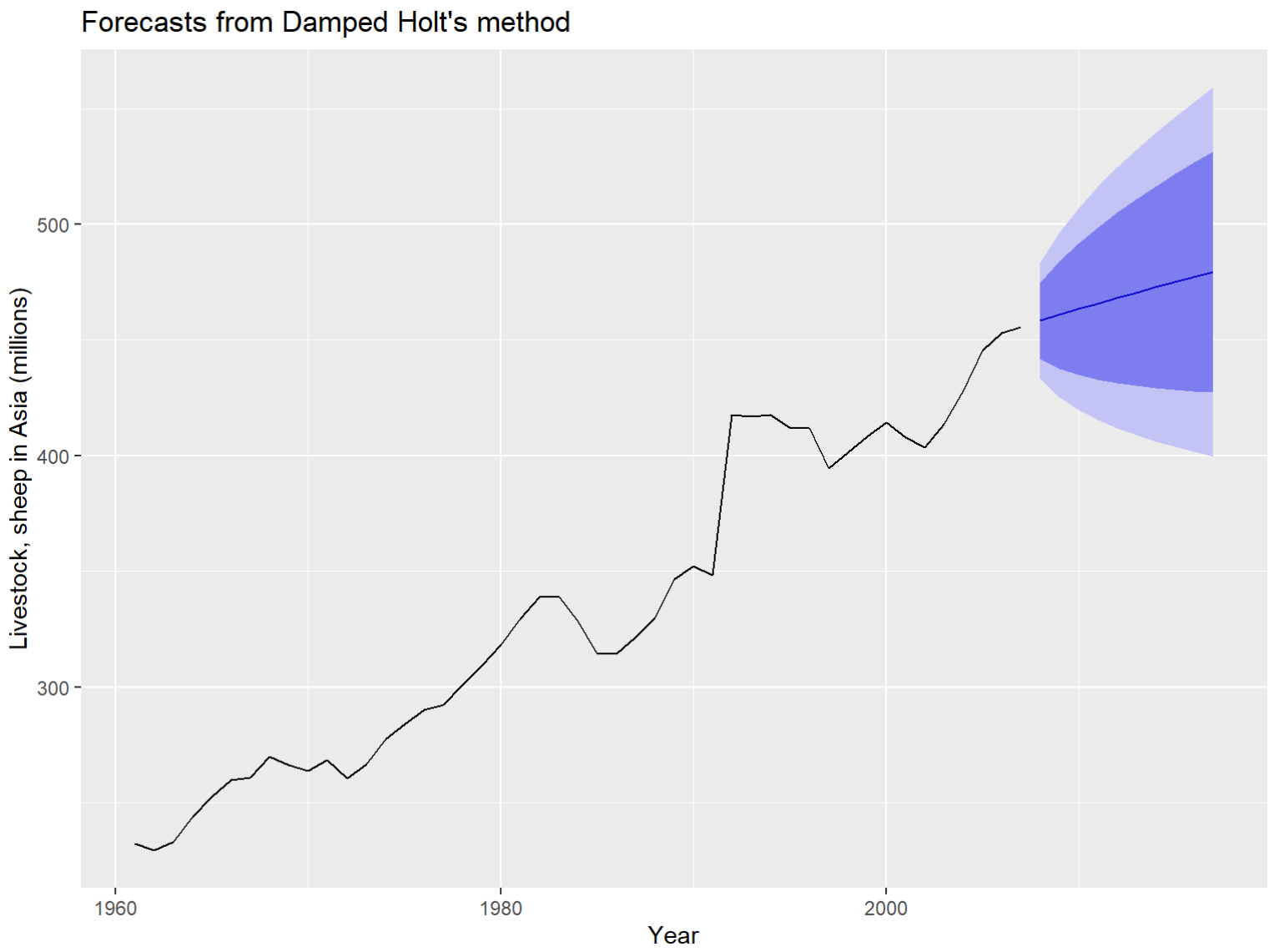
Damped Holt's method is best whether you compare MAE or MSE values. So we will proceed with using the damped Holt's method and apply it to the whole data set to get forecasts for future years.

```r
fc <- holt(livestock, damped=TRUE)
# Estimated parameters:
fc[["model"]]
```

```
## Damped Holt's method
##
## Call:
##  holt(y = livestock, damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.9999
##     beta  = 3e-04
##     phi   = 0.9798
##
##   Initial states:
##     l = 223.35
##     b = 6.9046
##
##   sigma:  12.8435
##
##      AIC     AICc      BIC
## 427.6370 429.7370 438.7379
```

The smoothing parameter for the slope parameter is estimated to be essentially zero, indicating that the trend is not changing over time. The value of α is very close to one, showing that the level reacts strongly to each new observation.

```
autoplot(fc) +
xlab("Year") + ylab("Livestock, sheep in Asia (millions)")
```



The resulting forecasts look sensible with increasing trend, and relatively wide prediction intervals reflecting the variation in the historical data. The prediction intervals are calculated using the methods in section 7.5.

In this example, the process of selecting a method was relatively easy as both MSE and MAE comparisons suggested the same method (damped Holt's).

However, sometimes different accuracy measures will suggest different forecasting methods, and then a decision is required as to which forecasting method we prefer to use.

As forecasting tasks can vary by many dimensions (length of forecast horizon, size of test set, forecast error measures, frequency of data, etc.), it is unlikely that one method will be better than all others for all forecasting scenarios.

What we require from a forecasting method are consistently sensible forecasts, and these should be frequently evaluated against the task at hand.

Holt-Winters seasonal method

The Holt-Winters seasonal method consists of the forecast equation and three smoothing equations

- one for the level ℓ_t,
- one for trend b_t, and
- one for the seasonal component denoted by s_t,

with smoothing parameters α, β^* and γ. We use m to denote the period of the seasonality, i.e., the number of seasons in a year ($m = 4$ for quarterly data and $m = 12$ for monthly data).

There are two variations to this method that differ in the nature of the seasonal component.

- The additive method is preferred when the seasonal variations are roughly constant through the series,
- while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series.

Holt-Winters additive method

With the additive method, the seasonal component is expressed in absolute terms in the scale of the observed series, and in the level equation the series is seasonally adjusted by subtracting the seasonal component. Within each year, the seasonal component will add up to approximately zero.

The component form for the additive method is:

$$\begin{aligned}
\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t-m+h_m^+} \\
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},
\end{aligned}$$

where $h_m^+ = \lfloor (h-1)/m \rfloor + 1$, which ensures that the estimates of the seasonal indices used for forecasting come from the final year of the sample. (The notation $\lfloor u \rfloor$ means the largest integer not greater than u.)

The level equation shows a weighted average between the seasonally adjusted observation $(y_t - s_{t-m})$ and the non-seasonal forecast $(\ell_{t-1} + b_{t-1})$ for time t. The trend equation is identical to Holt's linear method.

The seasonal equation shows a weighted average between the current seasonal index, $(y_t - \ell_{t-1} - b_{t-1})$, and the seasonal index of the same season last year (i.e., m time periods ago).

Holt-Winters multiplicative method

With the multiplicative method, the seasonal component is expressed in relative terms (percentages), and the series is seasonally adjusted by dividing through by the seasonal component. Within each year, the seasonal component will sum up to approximately m.

The component form for the multiplicative method is:

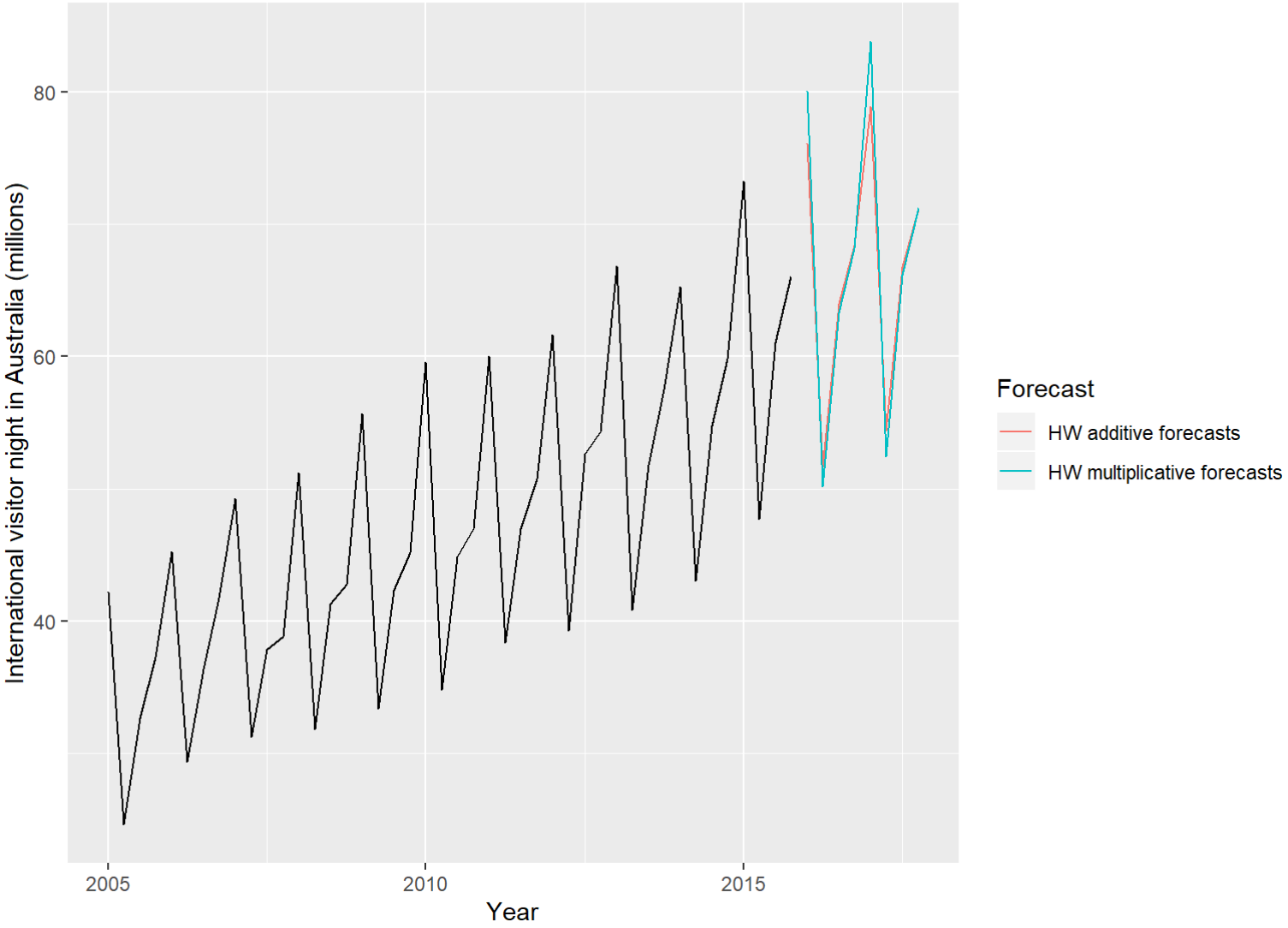
$$\begin{aligned}
\hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t-m+h_m^+}. \\
\ell_t &= \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma\frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}
\end{aligned}$$

hw function is used in the forecast package

Example International tourist visitor nights in Australia

```
aust <- window(austourists,start=2005)
fit1 <- hw(aust,seasonal="additive")
fit2 <- hw(aust,seasonal="multiplicative")
autoplot(aust) +
forecast::autolayer(fit1$mean, series="HW additive forecasts") +
forecast::autolayer(fit2$mean, series="HW multiplicative forecasts") +
xlab("Year") + ylab("International visitor night in Australia (millions)") +
guides(colour=guide_legend(title="Forecast"))
```

This applies the Holt-Winters method with both additive and multiplicative seasonality to forecast quarterly visitor nights in Australia spent by international tourist with data from 2005, and the forecast from 2016-2017. The smoothing parameters and initial estimates for the components have been estimated by minimizing the RMSE


International visitor night in Australia (millions)
80
60
40
2005
2010
2015
Year
Forecast
HW additive forecasts
HW multiplicative forecasts

Comparing the RMSE of both models:

```
accuracy(fit1)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.008115785 1.763305 1.374062 -0.2860248 2.973922 0.4502579
##                      ACF1
## Training set -0.06272507
```

```
accuracy(fit2)
```

```
##                       ME     RMSE     MAE           MPE     MAPE      MASE
## Training set 0.09206228 1.575631 1.25496 -0.0006505533 2.70539 0.4112302
##                      ACF1
## Training set -0.07955726
```

The method with multiplicative seasonality fits the data best. This was to be expected, as the time plot shows that the seasonal variation in the data increases as the level of the series increases. This is also reflected in the two sets of forecasts; the forecasts generated by the method with the multiplicative seasonality display larger and increasing seasonal variation as the level of the forecasts increases compared to the forecasts generated by the method with additive seasonality.

```
summary(fit1)
summary(fit2)
```

The two commands above give values for fit1 and fit2 as follows:

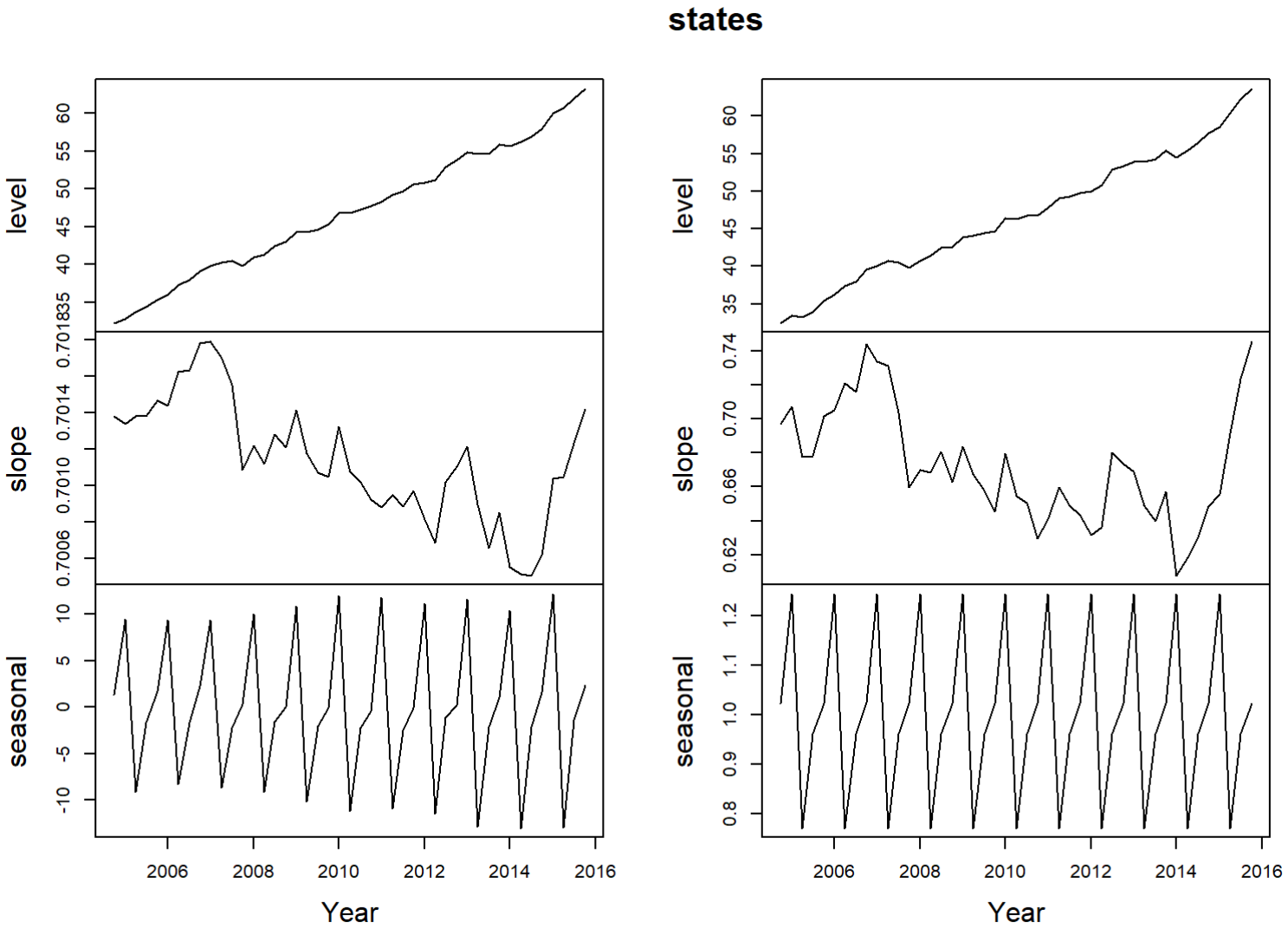
fit1: $\alpha = 0.3063, \beta^* = 1e-04, \gamma = 0.4263$ (additive seasonal component)

fit2: $\alpha = 0.4406, \beta^* = 0.0134, \gamma = 0.0023$ (multiplicative seasonal component)

The small value of γ for the multiplicative model means that the seasonal component hardly changes over time. The small value of β^* for the additive model means the slope component hardly changes over time (check the vertical scale).

The estimated states for both models are plotted in the next slide. The figures on the left are the additive state and the figure on the right are the multiplicative state

```
states <- cbind(fit1$model$states[,1:3],fit2$model$states[,1:3])
colnames(states) <- c("level","slope","seasonal","level","slope","seasonal")
plot(states, xlab="Year")
```


states
level
slope
seasonal
Year
2006
2008
2010
2012
2014
2016

Holt-Winters damped method

A method that is often the single most accurate forecasting method for seasonal data is the Holt-Winters method with a damped trend and multiplicative seasonality:

$$\begin{aligned}
\hat{y}_{t+h|t} &= [\ell_t + (\phi + \phi^2 + \cdots + \phi^h)b_t]s_{t-m+h_m^+}. \\
\ell_t &= \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1} \\
s_t &= \gamma\frac{y_t}{(\ell_{t-1} + \phi b_{t-1})} + (1-\gamma)s_{t-m}
\end{aligned}$$

This is implemented by the command:

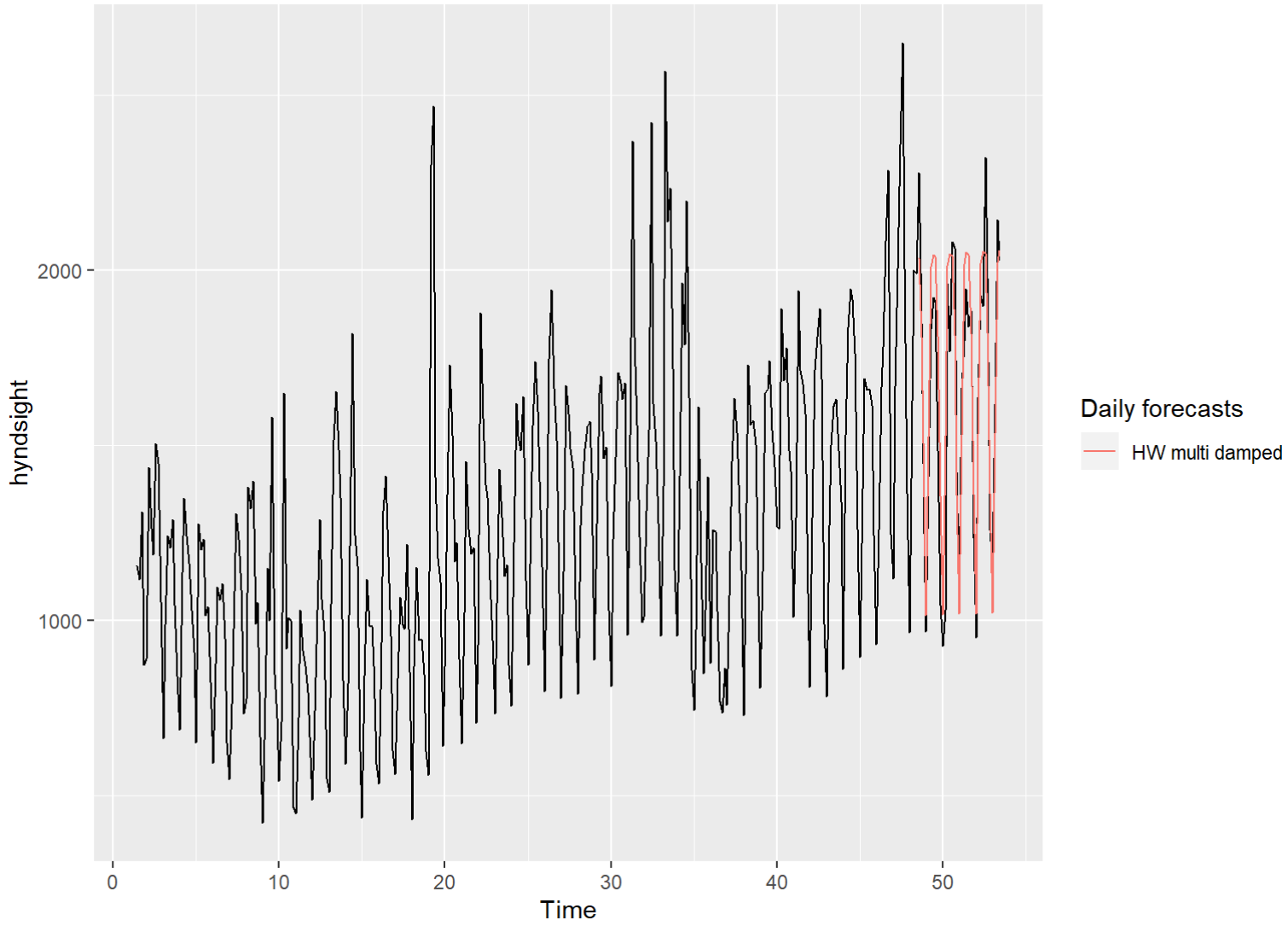
```
hw(x, damped=TRUE, seasonal="multiplicative")
```

Example: Holt-Winters method with daily data

The Holt-Winters method can also be used for daily type of data, where the seasonal pattern is of length 7, and the appropriate unit of time for is h in days. Here, we generate daily forecasts for the last five weeks for the hyndsight data, which contains the daily pageviews on the Hyndsight blog for one year starting April 30, 2014.

```
fc <- hw(subset(hyndsight,end=length(hyndsight)-35),
   damped = TRUE, seasonal="multiplicative", h=35)

autoplot(hyndsight) +
forecast::autolayer(fc$mean, series="HW multi damped")+
guides(colour=guide_legend(title="Daily forecasts"))
```


hyndsight
2000
1000
0
10
20
30
40
50
Time
Daily forecasts
HW multi damped

Taxonomy of exponential smoothing methods

Table 7.6: A two-way classification of exponential smoothing methods.

Trend Component	Seasonal Component		
	N (None)	A (Additive)	M (Multiplicative)
N (None)	(N,N)	(N,A)	(N,M)
A (Additive)	(A,N)	(A,A)	(A,M)
A_d (Additive damped)	(A_d,N)	(A_d,A)	(A_d,M)

Some of these methods we have already seen:

Two way classification of exponential smoothing models

Each method is labelled by a pair of letters (T,S) defining the type of 'Trend' and 'Seasonal' components. For example, (A,M) is the method with an additive trend and multiplicative seasonality; $(\mathrm{A}_d, \mathrm{N})$ is the method with damped trend and no seasonality; and so on.

Short hand	Method
(N,N)	Simple exponential smoothing
(A,N)	Holt's linear method
(A_d,N)	Additive damped trend method
(A,A)	Additive Holt-Winters' method
(A,M)	Multiplicative Holt-Winters' method
(A_d,M)	Holt-Winters' damped method

Short hand notation of methods already studied

The authors don't consider multiplicative trend methods as they produce poor forecast (Hydman et al. (2008))

Trend	Seasonal: N	Seasonal: A	Seasonal: M
	$\hat{y}_{t+h\|t} = \ell_t$	$\hat{y}_{t+h\|t} = \ell_t + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = \ell_t s_{t-m+h_m^+}$
N	$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)\ell_{t-1}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)\ell_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/\ell_{t-1}) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t + hb_t$	$\hat{y}_{t+h\|t} = \ell_t + hb_t + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = (\ell_t + hb_t)s_{t-m+h_m^+}$
A	$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$
	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/(\ell_{t-1} + b_{t-1})) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t + \phi_h b_t$	$\hat{y}_{t+h\|t} = \ell_t + \phi_h b_t + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = (\ell_t + \phi_h b_t)s_{t-m+h_m^+}$
A_d	$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$
	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1} - \phi b_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/(\ell_{t-1} + \phi b_{t-1})) + (1-\gamma)s_{t-m}$

Formula for recursive calculations and point forecast

The table gives the recursive formulae for applying the nine exponential smoothing methods. Each cell includes the forecast equation for generating h-step-ahead forecasts, and the smoothing equations for applying the method.

As discussed, ℓ_t denotes the series at time t, b_t denotes the slope at time t, s_t denotes the seasonal component of the series at time t, and m denotes the number of seasons in a year; α, β^*, γ and ϕ are smoothing parameters, $\phi_h = \phi + \phi^2 + \cdots + \phi_h$ and $h_m^+ = \lfloor (h-1)/m \rfloor + 1$

Innovations state space models for exponential smoothing

The exponential smoothing methods presented so far are algorithms that generate point forecasts.

There are statistical models that generate the same point forecasts, but can also generate prediction (or forecast) intervals. A statistical model is a stochastic (or random) data generating process that can produce an entire forecast distribution.

Each model consists of a measurement equation that describes the observed data and some transition equations that describe how the unobserved components or states (level, trend, seasonal) change over time. Hence these are referred to as state space models.

Each state space model can be labeled as ETS (Error, Trend, Seasonal). The possibilities for each component are:

Error ={A,M},

Trend ={N,A,A_d} and

Seasonal ={N,A,M},

where N =none, A = additive, M = multiplicative, and $_d$=damped.

The models can be identified using information criteria and estimated in R using the `ets()` function in the forecast package.

(Note: As mentioned earlier, the authors did not consider multiplicative trends as they are known to forecast poorly)

ETS(A,N,N): simple exponential smoothing with additive errors

Recall the component form in simple exponential smoothing:

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

Error correction form

The third form of simple exponential smoothing is obtained by re-arranging the level equation in the component form to get what we refer to as the error correction form

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

where $e_t = y_t - \ell_{t-1} = y_t - \hat{y}_{t|t-1}$ for $t = 1, \ldots, T$.

That is, e_t is the one-step within-sample forecast error at time t.

The "training-period" errors lead to the adjustment/correction of the estimated level throughout the smoothing process for $t = 1, \ldots, T$.

For example, if the error at time t is negative, then $\hat{y}_{t|t-1} > y_t$ and so the level at time $t-1$ has been over-estimated.

The new level ℓ_t is then the previous level ℓ_{t-1} adjusted downwards.

The closer α is to one the rougher the estimate of the level (large adjustments take place). The smaller the α the smoother the level (small adjustments take place).

We can also write $y_t = \ell_{t-1} + e_t$ (substitute into the last equation in the previous slide the smoothing equation).

To make this into an innovations state space model, all we need to do is specify the probability distribution of e_t.

For a model with additive errors, we assume the one-step ahead forecast errors e_t to be normally distributed white noise with mean 0 and variance σ^2 or $e_t \equiv \epsilon_t \sim NID(0, \sigma^2)$; NID stands for "normally and independently distributed."

The equations in the model can then be rewritten as:

$$y_t = \ell_{t-1} + \epsilon_t \qquad \text{measurement (or observation) eq}$$
$$\ell_t = \ell_{t-1} + \alpha\epsilon_t \qquad \text{state (or transition) eq}$$

These two equations, together with the statistical distribution of the errors, form a fully specified statistical model. Specifically, these constitute an innovations state space model underlying simple exponential smoothing.

The term "innovations" comes from the fact that all equations in this type of specification use the same random error process, ϵ_t.

The measurement equation shows the relationship between the observations and the unobserved states.

In this case, observation y_t is a linear function of the level ℓ_{t-1}, the predictable part of y_t, and the random error ϵ_t, the unpredictable part of y_t.

For other innovations state space models, this relationship may be nonlinear.

The transition equation shows the evolution of the state through time. The influence of the smoothing parameter α is the same as for the methods discussed earlier.

For example,α governs the degree of change in successive levels. The higher the value of α, the more rapid the changes in the level; the lower the value of α, the smoother the changes.

At the lowest extreme, where $\alpha = 0$, the level of the series does not change over time.

At the other extreme, where $\alpha = 1$, the model reduces to a random walk model, $y_t = y_{t-1} + \epsilon_t$.

i.e. $\epsilon_t = y_t - \ell_{t-1}$ so when $\alpha = 1$, $\ell_t = \ell_{t-1} + (y_t - \ell_{t-1})$ or $\ell_t = y_t$ and recursively, $\ell_{t-1} = y_{t-1}$

ETS(M,N,N): simple exponential smoothing with multiplicative errors

Similarly, we can specify models with multiplicative errors by writing one-step random errors as relative errors:

$$\epsilon_t = \frac{y_t - \hat{y}_{t|t-1}}{\hat{y}_{t|t-1}}$$

where $\epsilon_t \sim NID(0, \sigma^2)$. Substituting $\hat{y}_{t|t-1} = \ell_{t-1}$ gives the measurement equation: $y_t = \ell_{t-1} + \ell_{t-1}\epsilon_t$.

The level equation is derived from $\ell_t = \ell_{t-1} + \alpha(y_t - \ell_{t-1})$ as before, and from above, $\ell_{t-1}\epsilon_t = y_t - \hat{y}_{t|t-1}$. By definition, the forecast error $e_t = y_t - \hat{y}_{t|t-1}$. Hence, $e_t = y_t - \hat{y}_{t|t-1} = \ell_{t-1}\epsilon_t$. Substituting into the level equation previously derived gives the state or transition equation below.

Then the multiplicative form of the state space model is:

$$\begin{aligned} y_t &= \ell_{t-1}(1 + \epsilon_t) && \text{measurement (or observation) eq} \\ \ell_t &= \ell_{t-1}(1 + \alpha\epsilon_t) && \text{state (or transition) eq} \end{aligned}$$

ETS(A,A,N): Holt's linear method with additive errors

For this model, we assume that the one-step forecast errors are given by $e_t = y_t - \ell_{t-1} - b_{t-1} \equiv \epsilon_t \sim NID(0, \sigma^2)$. Substituting this into the error correction equations for Holt's linear method gives:

$$\begin{aligned} y_t &= \ell_{t-1} + b_{t-1} + \epsilon_t \\ \ell_t &= \ell_{t-1} + b_{t-1} + \alpha\epsilon_t \\ b_t &= b_{t-1} + \beta\epsilon_t \end{aligned}$$

where for simplicity, $\beta = \alpha\beta^*$

ETS(M,A,N): Holt's linear method with multiplicative errors

Specifying one-step forecast errors as relative errors such that

$$\epsilon_t = \frac{y_t - (\ell_{t-1} + b_{t-1})}{\ell_{t-1} + b_{t-1}}$$

and following an approach similar to that used above, the innovations state space model underlying Holt's linear method with multiplicative errors is specified as:

$$\begin{aligned} y_t &= (\ell_{t-1} + b_{t-1})(1 + \epsilon_t) \\ \ell_t &= (\ell_{t-1} + b_{t-1})(1 + \alpha\epsilon_t) \\ b_t &= b_{t-1} + \beta(\ell_{t-1} + b_{t-1})\epsilon_t \end{aligned}$$

where $\beta = \alpha\beta^*$ and $\epsilon_t \sim NID(0, \sigma^2)$

Other ETS models

The tables below show the equations for all the models in the ETS framework.

ADDITIVE ERROR MODELS

Trend	Seasonal: N	A	M
N	$y_t = \ell_{t-1} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \alpha\varepsilon_t$	$y_t = \ell_{t-1} + s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \alpha\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = \ell_{t-1}s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/\ell_{t-1}$
A	$y_t = \ell_{t-1} + b_{t-1} + \varepsilon_t$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$ $b_t = b_{t-1} + \beta\varepsilon_t$	$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$ $b_t = b_{t-1} + \beta\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = (\ell_{t-1} + b_{t-1})s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $b_t = b_{t-1} + \beta\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/(\ell_{t-1} + b_{t-1})$
A_d	$y_t = \ell_{t-1} + \phi b_{t-1} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t$ $b_t = \phi b_{t-1} + \beta\varepsilon_t$	$y_t = \ell_{t-1} + \phi b_{t-1} + s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t$ $b_t = \phi b_{t-1} + \beta\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = (\ell_{t-1} + \phi b_{t-1})s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $b_t = \phi b_{t-1} + \beta\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/(\ell_{t-1} + \phi b_{t-1})$

State space equations for each of the models in the ETS with additive errors

MULTIPLICATIVE ERROR MODELS

Trend	Seasonal		
	N	**A**	**M**
N	$y_t = \ell_{t-1}(1+\varepsilon_t)$ $\ell_t = \ell_{t-1}(1+\alpha\varepsilon_t)$	$y_t = (\ell_{t-1}+s_{t-m})(1+\varepsilon_t)$ $\ell_t = \ell_{t-1}+\alpha(\ell_{t-1}+s_{t-m})\varepsilon_t$ $s_t = s_{t-m}+\gamma(\ell_{t-1}+s_{t-m})\varepsilon_t$	$y_t = \ell_{t-1}s_{t-m}(1+\varepsilon_t)$ $\ell_t = \ell_{t-1}(1+\alpha\varepsilon_t)$ $s_t = s_{t-m}(1+\gamma\varepsilon_t)$
A	$y_t = (\ell_{t-1}+b_{t-1})(1+\varepsilon_t)$ $\ell_t = (\ell_{t-1}+b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = b_{t-1}+\beta(\ell_{t-1}+b_{t-1})\varepsilon_t$	$y_t = (\ell_{t-1}+b_{t-1}+s_{t-m})(1+\varepsilon_t)$ $\ell_t = \ell_{t-1}+b_{t-1}+\alpha(\ell_{t-1}+b_{t-1}+s_{t-m})\varepsilon_t$ $b_t = b_{t-1}+\beta(\ell_{t-1}+b_{t-1}+s_{t-m})\varepsilon_t$ $s_t = s_{t-m}+\gamma(\ell_{t-1}+b_{t-1}+s_{t-m})\varepsilon_t$	$y_t = (\ell_{t-1}+b_{t-1})s_{t-m}(1+\varepsilon_t)$ $\ell_t = (\ell_{t-1}+b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = b_{t-1}+\beta(\ell_{t-1}+b_{t-1})\varepsilon_t$ $s_t = s_{t-m}(1+\gamma\varepsilon_t)$
$\mathbf{A_d}$	$y_t = (\ell_{t-1}+\phi b_{t-1})(1+\varepsilon_t)$ $\ell_t = (\ell_{t-1}+\phi b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = \phi b_{t-1}+\beta(\ell_{t-1}+\phi b_{t-1})\varepsilon_t$	$y_t = (\ell_{t-1}+\phi b_{t-1}+s_{t-m})(1+\varepsilon_t)$ $\ell_t = \ell_{t-1}+\phi b_{t-1}+\alpha(\ell_{t-1}+\phi b_{t-1}+s_{t-m})\varepsilon_t$ $b_t = \phi b_{t-1}+\beta(\ell_{t-1}+\phi b_{t-1}+s_{t-m})\varepsilon_t$ $s_t = s_{t-m}+\gamma(\ell_{t-1}+\phi b_{t-1}+s_{t-m})\varepsilon_t$	$y_t = (\ell_{t-1}+\phi b_{t-1})s_{t-m}(1+\varepsilon_t)$ $\ell_t = (\ell_{t-1}+\phi b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = \phi b_{t-1}+\beta(\ell_{t-1}+\phi b_{t-1})\varepsilon_t$ $s_t = s_{t-m}(1+\gamma\varepsilon_t)$

State space equations for each of the models in the ETS with multiplicative errors

Estimating ETS models

ETS models are estimated in the forecasting package using the function *ets()* (more on the ets() function later.)

The ets() function estimates ETS models maximizing the "likelihood".

The likelihood is the probability of the data arising from the specified model. Thus, a large likelihood is associated with a good model.

Hence, using ets() function, we will estimate the smoothing parameters $\alpha, \beta, \gamma \text{ and } \phi$ and the initial states $\ell_0, b_0, s_0, s_{-1}, \cdots, s_{-m+1}$ by maximizing the likelihood.

The possible values that the smoothing parameters can take are restricted.

Traditionally, the parameters have been constrained to lie between 0 and 1 so that the equations can be interpreted as weighted averages.

That is, $0 < \alpha, \beta^*, \gamma^*, \phi < 1$.

For the state space models, we have set $\beta = \alpha\beta^*$.

Therefore, the traditional restrictions translate to $0 < \alpha < 1$, $0 < \beta < \alpha$.

In practice, the dampening parameter ϕ is usually constrained further to prevent numerical difficulties in estimating the model. In R, it is restricted so that $0.8 < \phi < 0.98$.

Another way to view the parameters is through a consideration of the mathematical properties of the state space models.

The parameters are constrained in order to prevent observations in the distant past having a continuing effect on current forecasts. This leads to some admissibility constraints on the parameters, which are usually (but not always) less restrictive than the usual region.

For example:

For the ETS(A,N,N) model, the usual parameter region is $0 < \alpha < 1$ but the admissible region is $0 < \alpha < 2$.

For the ETS(A,A,N) model, the usual parameter region is $0 < \alpha < 1$ and $0 < \beta < \alpha$ but the admissible region is $0 < \alpha < 2$ and $0 < \beta < 4 - 2\alpha$.

Model selection

A great advantage of the ETS statistical framework is that information criteria can calculated from the likelihood function L.

The AIC, AIC_C and BIC, introduced in Section 5.5, can be used here to determine which of the ETS models is most appropriate for a given time series.

For ETS models, Akaike's Information Criterion (AIC) is defined as

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

where L is the likelihood of the model and k is the total number of parameters and initial states that have been estimated (including the residual variance).

The AIC corrected for small sample bias (AIC_C) is defined as

$$\text{AIC}_\text{c} = \text{AIC} + \frac{k(k+1)}{T-k-1}.$$

and the Bayesian Information Criterion (BIC) is

$$\text{BIC} = \text{AIC} + k[\log(T) - 2].$$

Three of the combinations of (Error, Trend, Seasonal) can lead to numerical difficulties. Specifically, the models that can cause such instabilities are ETS(A,N,M), ETS(A,A,M), and ETS(A,A ,M). We normally do not consider these particular combinations when selecting a model.

For more information on these instabilities, please refer to the paper by Hydman and Akram (2006) available at (https://www.monash.edu/business/econometrics-and-business-statistics/research/publications/ebs/wp03-06.pdf).

The ets() function in R

The models can be estimated in R using the ets() function in the forecast package. Unlike the ses, holt and hw functions, the ets function does not produce forecasts. Rather, it estimates the model parameters and returns information about the fitted model.

The R code below shows the most important arguments that this function can take, and their default values. If only the time series is specified, and all other arguments are left at their default values, then an appropriate model will be selected automatically. The arguments are explained below. See the Rdocumentation on ets for a more complete description (https://www.rdocumentation.org/packages/forecast/versions/8.1/topics/ets).

```
ets(y, model="ZZZ", damped=NULL, alpha=NULL, beta=NULL, gamma=NULL,
phi=NULL, lambda=NULL, biasadj=FALSE, additive.only=FALSE,
restrict=TRUE, allow.multiplicative.trend=FALSE)
```

y

The time series to be forecast.

model

A three-letter code indicating the model to be estimated using the ETS classification and notation. The possible inputs are “N” for none, “A” for additive, “M” for multiplicative, or “Z” for automatic selection. If any of the inputs is left as “Z”, then this component is selected according to the information criterion chosen. The default value of ZZZ ensures that all components are selected using the information criterion.

damped

If damped=TRUE , then a damped trend will be used (either A or M). If damped=FALSE , then a nondamped trend will used. If damped=NULL (the default), then either a damped or a non-damped trend will be selected, depending on which model has the smallest value for the information criterion.

alpha , beta , gamma , phi

The values of the smoothing parameters can be specified using these arguments. If they are set to NULL (the default setting for each of them), the parameters are estimated.

lambda

Box-Cox transformation parameter. It will be ignored if lambda=NULL (the default value). Otherwise, the time series will be transformed before the model is estimated. When lambda is not NULL, additive.only is set to TRUE.

biasadj

If TRUE and lambda is not NULL, then the back-transformed fitted values and forecasts will be biasadjusted.

additive.only

Only models with additive components will be considered if additive.only=TRUE . Otherwise, all models will be considered.

restrict

If restrict=TRUE (the default), the models that cause numerical difficulties are not considered in model selection.

allow.multiplicative.trend

Multiplicative trend models are also available, but not covered in the textbook. Set this argument to TRUE to allow these models to be considered.

Working with ets objects

The ets() function will return an object of class ets. There are many R functions designed to make working with ets objects easy. A few of them are described below.

coef()

returns all fitted parameters.

accuracy()

returns accuracy measures computed on the training data.

summary()

prints some summary information about the fitted model.

autoplot() and plot()

produce time plots of the components.

residuals()

returns residuals from the estimated model.

fitted()

returns one-step forecasts for the training data.

simulate()

will simulate future sample paths from the fitted model.

forecast() computes point forecasts and prediction intervals, as described in the next section.

Example International tourist visitor nights in Australia

the ETS statistical framework is used to forecast tourist visitor nights in Australia by international arrivals over the period 2016-2019. The ets() function selects the model by minimizing the AICc.

```
aust <- window(austourists, start=2005)
fit <- ets(aust)
summary(fit)
```

```
## ETS(M,A,M)
##
## Call:
##  ets(y = aust)
##
##   Smoothing parameters:
##     alpha = 0.1908
##     beta  = 0.0392
##     gamma = 2e-04
##
##   Initial states:
##     l = 32.3679
##     b = 0.9281
##     s = 1.0218 0.9628 0.7683 1.2471
##
##   sigma:  0.0383
##
##      AIC     AICc      BIC
## 224.8628 230.1569 240.9205
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE     MASE
## Training set 0.04836907 1.670893 1.24954 -0.1845609 2.692849 0.409454
##                    ACF1
## Training set 0.2005962
```

From the results above, ets() chooses the model: ETS(M,A,M):

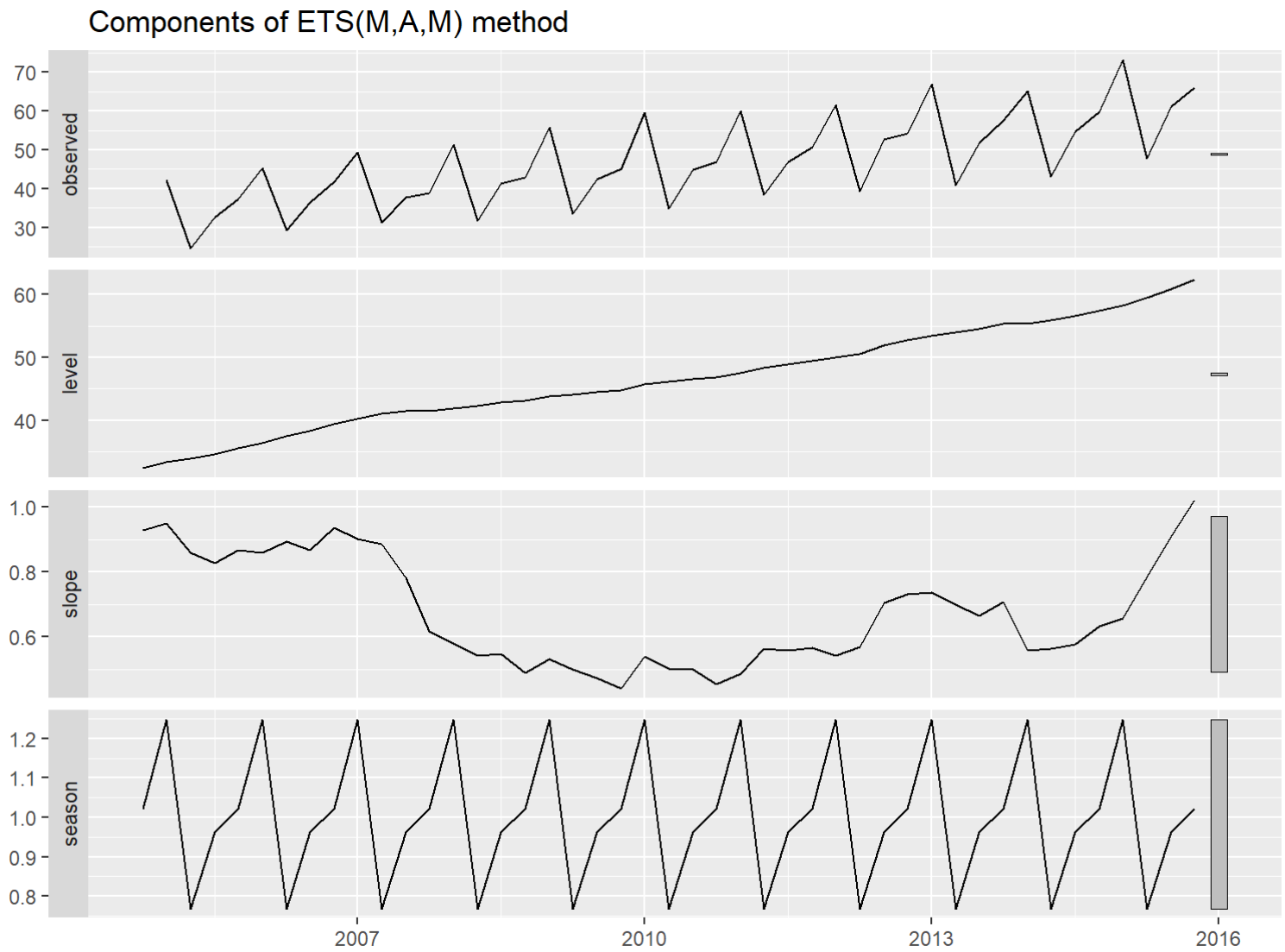
$$\begin{aligned}
y_t &= (\ell_{t-1} + b_{t-1})s_{t-m}(1 + \epsilon_t) \\
\ell_t &= (\ell_{t-1} + b_{t-1})(1 + \alpha\epsilon_t) \\
b_t &= b_{t-1} + \beta(\ell_{t-1} + b_{t-1})\epsilon_t \\
s_t &= s_{t-m}(1 + \gamma\epsilon_t).
\end{aligned}$$

From the output above, the parameter estimates are $\alpha = 0.198$, $\beta = 0.0319$ and $\gamma = 0.00019$. The output also returns ℓ_0, b_0, s_0, s_{-1}, s_{-2} and s_{-3}.

The ETS(M,A,M) model gives equivalent point forecast to the multiplicative Holt-Winters' method. The difference is that the ETS(M,A,M) default estimation method is the maximum likelihood while the Holt-Winters estimation method is the minimum least squares.

The next slide shows the point forecast and prediction intervals generated from the model. The small values of β and γ mean that the slope and seasonal components change very little over time. The narrow prediction intervals indicate that the series is relatively easy to forecast due to the strong trend and seasonality.

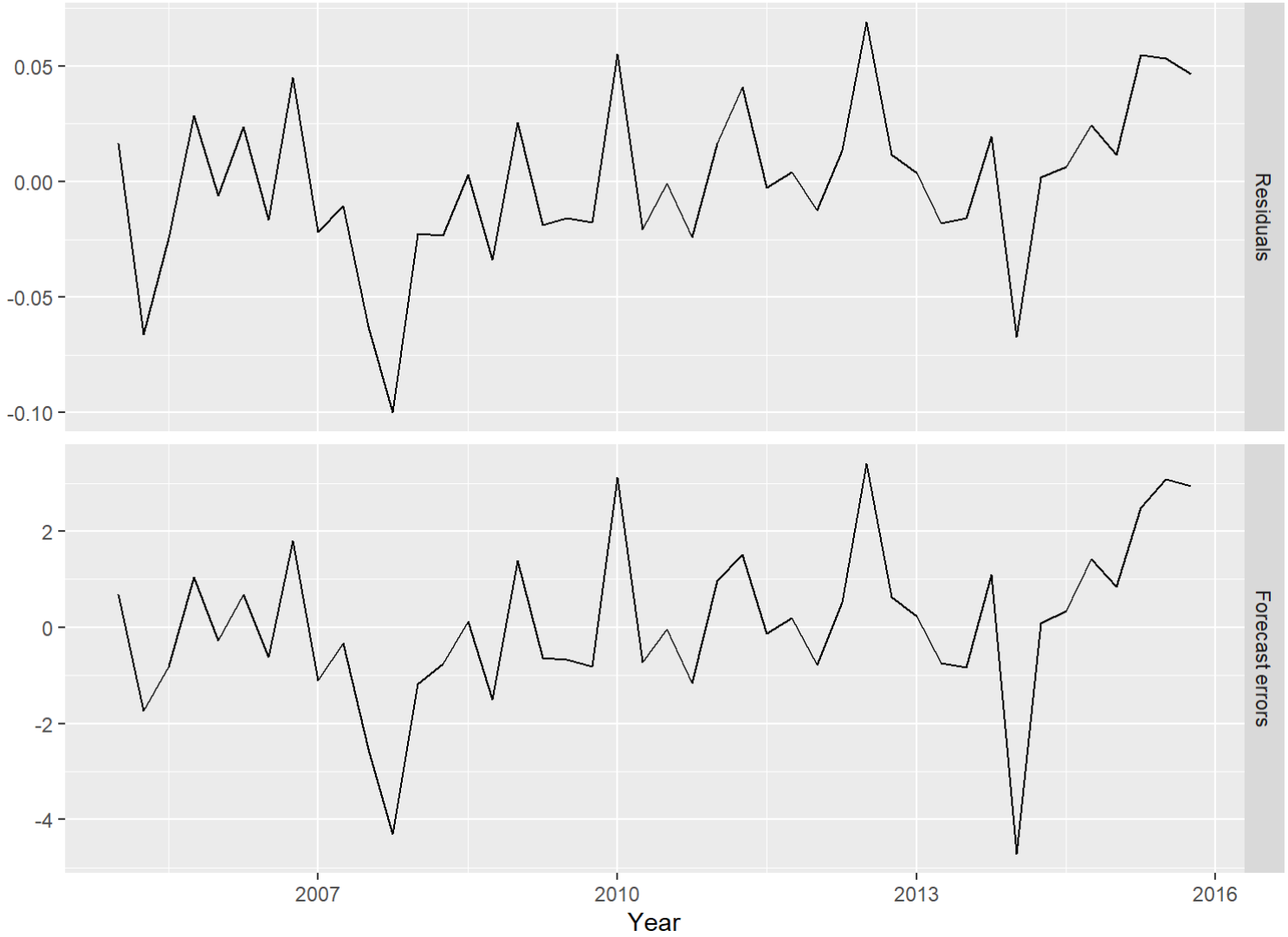
```
autoplot(fit)
```



Note that the ETS(M,A,M) has multiplicative errors. The residuals $\hat{\epsilon}_t$ and the one-step ahead forecast errors $y_t - \hat{y}_t$ could be obtained by the residuals function:

```
cbind('Residuals' = residuals(fit),
      'Forecast errors' = residuals(fit, type='response')) %>%
 autoplot(facet=TRUE) + xlab("Year") + ylab("")
```

The type argument is used in the residuals function to distinguish between residuals and forecast errors. The default is type='innovation' which gives regular residuals


0.05
0.00
-0.05
-0.10
Residuals
2
0
-2
-4
Forecast errors
2007
2010
2013
2016
Year

Forecasting with ETS models

Point forecasts are obtained from the models by iterating the equations for $t = T+1, \cdots, T+h$ and setting all $\epsilon = 0$ for $t > T$.

For example, for model ETS(M,A,N), $y_{T+1} = (\ell_T + b_T)(1 + \epsilon_{T+1})$. Therefore $\hat{y}_{T+1|T} = \ell_T + b_T$

Similarly,

$$\begin{aligned} y_{T+2} &= (\ell_{T+1} + b_{T+1})(1 + \epsilon_{T+1}) \\ &= [(\ell_T + b_T)(1 + \alpha\epsilon_{T+1}) + b_T + \beta(\ell_T + b_T)\epsilon_{T+1}](1 + \epsilon_{T+1}) \end{aligned}$$

Therefore, $\hat{y}_{T+2|T} = \ell_T + 2b_T$ and so on.

To derive the above, plug in the observation equation and state equations of ETS(M,A,N).

ETS(M,A,N)

$$
\begin{aligned}
y_t &= (\ell_{t-1} + b_{t-1})(1 + \epsilon_t) \\
\ell_t &= (\ell_{t-1} + b_{t-1})(1 + \alpha\epsilon_t) \\
b_t &= b_{t-1} + \beta(\ell_{t-1} + b_{t-1})\epsilon_t
\end{aligned}
$$

ETS point forecasts are equal to the medians of the forecast distributions. For models with only additive components, the forecast distributions are normal, so the medians and means are equal.

For ETS models with multiplicative errors, or with multiplicative seasonality, the point forecasts will not be equal to the means of the forecast distributions.

To obtain forecasts from an ETS model, we use the forecast function.

The R code below shows the possible arguments that this function takes when applied to an ETS model. We explain each of the arguments in what follows.

The forecast command:

```
forecast(object, h=ifelse(object$m>1, 2*object$m, 10),
level=c(80,95), fan=FALSE, simulate=FALSE, bootstrap=FALSE, npaths=5000,
PI=TRUE, lambda=object$lambda, biasadj=NULL, ...)
```

object

The object returned by the ets() function.

h

The forecast horizon - the number of periods to be forecast.

level

The confidence level for the prediction intervals.

fan

If fan=TRUE, this is suitable for fan plots.

simulate

If simulate=TRUE , prediction intervals are produced by simulation rather than using algebraic formulae. Simulation will also be used (even if simulate=FALSE) where there are no algebraic

formulae available for the particular model.

bootstrap

If bootstrap=TRUE and simulate=TRUE , then the simulated prediction intervals use re-sampled errors rather than normally distributed errors.

npaths

The number of sample paths used in computing simulated prediction intervals.

PI

If PI=TRUE, then prediction intervals are produced; otherwise only point forecasts are calculated.

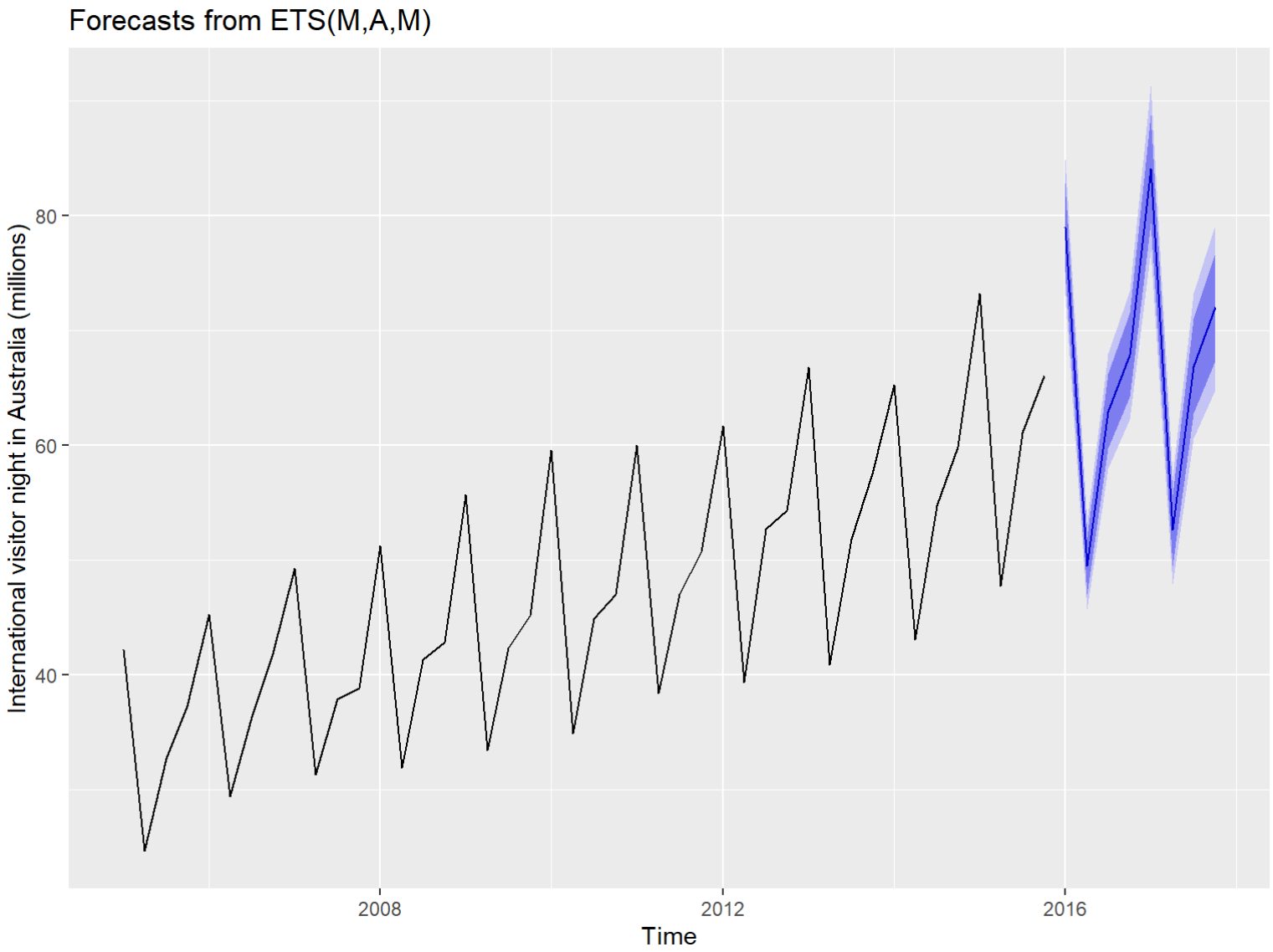
lambda

The Box-Cox transformation parameter. This is ignored if lambda=NULL . Otherwise, the forecasts are back-transformed via an inverse Box-Cox transformation.

biasadj

If lambda is not NULL , the backtransformed forecasts (and prediction intervals) are bias-adjusted.

```
fit %>% forecast(h=8) %>%
autoplot() +
ylab("International visitor night in Australia (millions)")
```



Prediction intervals

The prediction intervals will differ between models with additive and multiplicative methods.

For most ETS models, a prediction interval can be written as

$$\hat{y}_{T+h|T} \pm k\sigma_h$$

where k depends on the coverage probability, and σ_h is the forecast variance. Values for k were given in Table 3.1 based on the normal distribution.

For ETS models, the formula for σ_h can be complicated; the details are given in Chapter 6 of Hyndman et al. (2008).

In the table below, the formula given are for the additive ETS models, which are the simplest. In the table, the forecast variance expressions for each additive state space model are shown, where σ^2 is the residual variance, m is the seasonal period, $h_m = \lfloor (h-1)/m \rfloor$ and $\lfloor u \rfloor$ denote the integer part of u.

Model	Forecast variance: σ_h
(A,N,N)	$\sigma_h = \sigma^2\big[1+\alpha^2(h-1)\big]$
(A,A,N)	$\sigma_h = \sigma^2\Big[1+(h-1)\big\{\alpha^2+\alpha\beta h+\frac{1}{6}\beta^2 h(2h-1)\big\}\Big]$
(A,A_d,N)	$\sigma_h = \sigma^2\Big[1+\alpha^2(h-1)+\frac{\beta\phi h}{(1-\phi)^2}\{2\alpha(1-\phi)+\beta\phi\}$
	$-\frac{\beta\phi(1-\phi^h)}{(1-\phi)^2(1-\phi^2)}\{2\alpha(1-\phi^2)+\beta\phi(1+2\phi-\phi^h)\}\Big]$
(A,N,A)	$\sigma_h = \sigma^2\Big[1+\alpha^2(h-1)+\gamma h_m(2\alpha+\gamma)\Big]$
(A,A,A)	$\sigma_h = \sigma^2\Big[1+(h-1)\big\{\alpha^2+\alpha\beta h+\frac{1}{6}\beta^2 h(2h-1)\big\}+\gamma h_m\big\{2\alpha+\gamma+\beta m(h_m+1)\big\}\Big]$
(A,A_d,A)	$\sigma_h = \sigma^2\Big[1+\alpha^2(h-1)+\frac{\beta\phi h}{(1-\phi)^2}\{2\alpha(1-\phi)+\beta\phi\}$
	$-\frac{\beta\phi(1-\phi^h)}{(1-\phi)^2(1-\phi^2)}\{2\alpha(1-\phi^2)+\beta\phi(1+2\phi-\phi^h)\}$
	$+\gamma h_m(2\alpha+\gamma)+\frac{2\beta\gamma\phi}{(1-\phi)(1-\phi^m)}\big\{h_m(1-\phi^m)-\phi^m(1-\phi^{mh_m})\big\}\Big]$

Forecast variance expressions for each additive state model

For a few ETS models, there are no known formula for prediction intervals. In these cases, the forecast.ets function uses simulated future sample paths and computes prediction intervals from the percentiles of these simulated future paths.

Thank you for your attention