Applied Economic Forecasting

6. Moving Averages & Exponential Smoothing

Section 1

Simple exponential smoothing

Problem

Say, we are interested in generating some forecasts for a variable, y_t. We could do so using some of the methods we observed earlier.

- In the case of a **naïve forecast** model:

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

so all the weight is assigned to the last observation.

- With the **mean** forecast,

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

we assigned equal weights to the most recent observations as well as those far into the past (and are potentially less correlated with y_t?).

$$\hat{y}_{t+1} = \frac{y_t + y_{t-1} + \cdots + y_{t-k+1}}{k}$$

Solution: Exponential Smoothing Methods

The issue is clear but how do we address this?

Solution: Getting something in between!

- Assign less weights to past observations
- Assign higher weights to more recent data

Simple Exponential Smoothing to the “rescue!”

Simple Exponential Smoothing Methods

This method is suitable for forecasting data with no clear trend or seasonal pattern.

- The exponential smoothing equation can take the form

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

OR

$$\hat{y}_{T+1|T} = \alpha y_T + (1-\alpha)\hat{y}_{T|T-1},$$

where $0 \leq \alpha \leq 1$ is the smoothing constant or weighting factor.

- Eqn. 1 can be rewritten as

$$\begin{aligned} \hat{y}_{t+1} &= \alpha y_t + \hat{y}_t - \alpha \hat{y}_t \\ &= \hat{y}_t + \alpha \left(y_t - \hat{y}_t \right) \end{aligned}$$

- The new forecast is the old forecast adjusted by α times the forecast error in the old forecast

Simple Exponential Smoothing Methods

Question:

In this equation,

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

1. What happens if $\alpha \to 1$?

Answer: As α is getting bigger, more weight is given to the most recent observation, but less weight to old information.

2. What happens if $\alpha \to 0$?

Answer: In this case, as α is getting smaller, less weight is given to the most recent observation, but more weight to old information.

Simple Exponential Smoothing Methods

- Substitute $\hat{y}_t = \alpha y_{t-1} + (1-\alpha)\,\hat{y}_{t-1}$ into Eqn 1 to yield

$$\begin{aligned}\hat{y}_{t+1} &= \alpha y_t + (1-\alpha)\left[\alpha y_{t-1} + (1-\alpha)\,\hat{y}_{t-1}\right] \\ &= \alpha y_t + \alpha\,(1-\alpha)\,y_{t-1} + (1-\alpha)^2\,\hat{y}_{t-1}\end{aligned}$$

Simple Exponential Smoothing Methods

- Substitute $\hat{y}_t = \alpha y_{t-1} + (1-\alpha)\,\hat{y}_{t-1}$ into Eqn 1 to yield

$$\begin{aligned}\hat{y}_{t+1} &= \alpha y_t + (1-\alpha)\left[\alpha y_{t-1} + (1-\alpha)\,\hat{y}_{t-1}\right] \\ &= \alpha y_t + \alpha\,(1-\alpha)\,y_{t-1} + (1-\alpha)^2\,\hat{y}_{t-1}\end{aligned}$$

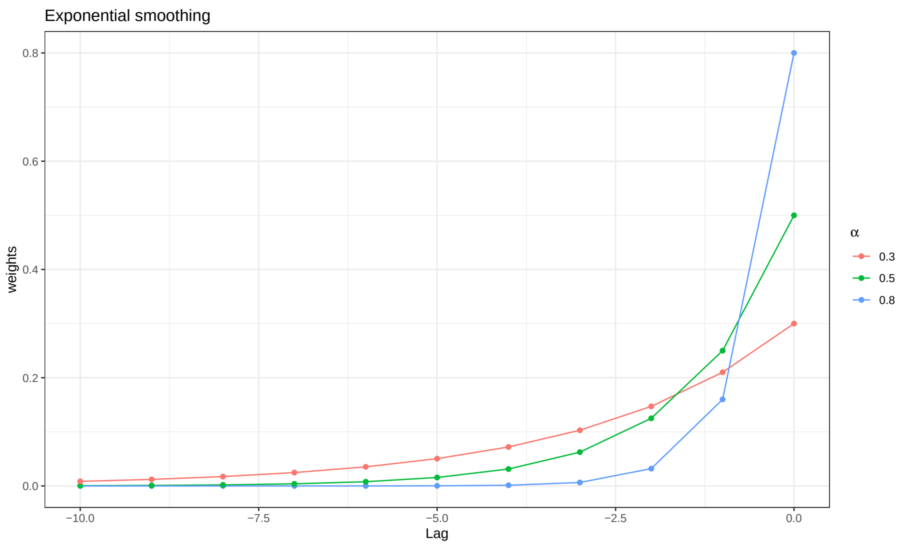
- Continue this substitution to obtain:

$$\begin{aligned}\hat{y}_{t+1} = \alpha y_t + \alpha\,(1-\alpha)\,y_{t-1} + \alpha\,(1-\alpha)^2\,y_{t-2} + \\ \alpha\,(1-\alpha)^3\,y_{t-3} + \alpha\,(1-\alpha)^4\,y_{t-4} + \cdots\end{aligned} \tag{2}$$

Note α determines the speed of decaying impacts of past observations on the forecast value

Visualing the smoothing parameter

```r
alf <- c(0.8, 0.5, 0.3)
TT <- 10
tibble(lag = TT:0) %>%
  mutate(`0.8` = alf[1]*(1-alf[1])^lag,
         `0.5` = alf[2]*(1-alf[2])^lag,
         `0.3` = alf[3]*(1-alf[3])^lag) %>%
  pivot_longer(-lag, names_to = "alpha") %>%
  ggplot(aes(x = -1*lag, y = value)) +
  geom_line(aes(col = alpha)) +
  geom_point(aes(col = alpha)) +
  labs(x = "Lag", y = "weights",
       title = "Exponential smoothing") +
  guides(col = guide_legend(title = bquote(alpha)))
```


Exponential smoothing
weights
0.8
0.6
0.4
0.2
0.0
−10.0
−7.5
−5.0
−2.5
0.0
Lag
α
0.3
0.5
0.8

Adding more context to the earlier slides:

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

Since the fitted values are simply one-step forecasts of the training data. The question arises, where does the process start? We let the first fitted value at time 1 be denoted by ℓ_0 (which we will have to estimate). Then

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

By recursive substitution we get:

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

The last term becomes smaller the large t becomes. As $T \to \infty$, this collapses to Eq (2)

This is often referred to as the Weighted Average Form

Simple Exponential Smoothing Methods

Alternatively, we can express Eq (3) in component form. For simple exponential smoothing, the only component included is the level, ℓ_t.

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

where ℓ_t is the level (or the smoothed value) of the series at time t. Setting $h = 1$ gives the fitted values, while setting $t = T$ gives the true forecasts beyond the training data.

The *forecast equation* shows that the forecast 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.

Simple Exponential Smoothing Methods

Simple Exponential Smoothing have a flat forecast function as all forecasts will assume the same value as the last level component.

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

Exponential Smoothing Methods

To use exponential smoothing methods, we need to determine:

1. the value of α (can be determined by minimizing MSE)

$$\text{SSE} = \sum_{t=1}^{T} (y_t - \hat{y}_{t|t-1})^2 = \sum_{t=1}^{T} e_t^2. \tag{4}$$

2. the initial value for the smoothing
   a. the first observation
   b. the average of the first five or six observations

Simple Exponential Smoothing

```r
algeria <- global_economy %>% filter(Country == "Algeria")
ses.fit <- algeria %>% model(
 ANN = ETS(Exports ~ error("A") + trend("N") + season("N"))
)

ses.fit %>% report()
```

```
## Series: Exports
## Model: ETS(A,N,N)
##   Smoothing parameters:
##     alpha = 0.84
##
##   Initial states:
##   l[0]
##  39.54
##
##   sigma^2:  35.63
##
##   AIC  AICc   BIC
## 446.7 447.2 452.9
```

```
ses.fit %>% components()
```

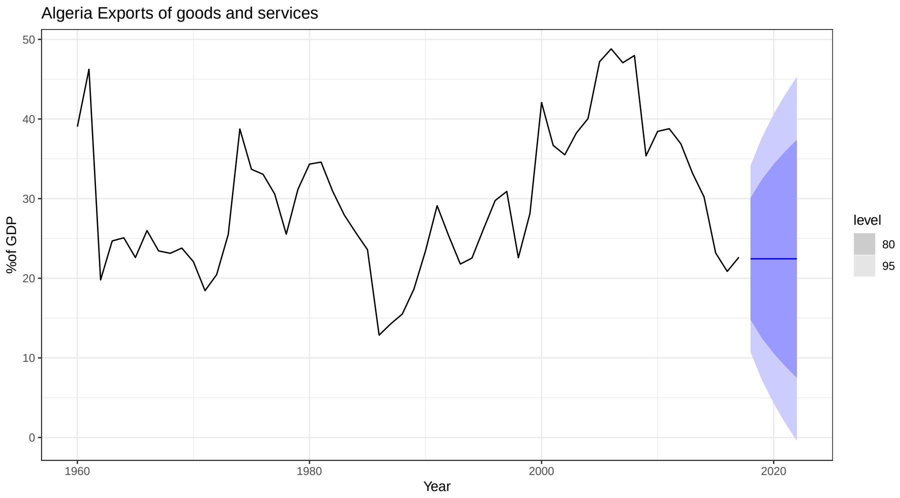
```
## # A dable: 59 x 6 [1Y]
## # Key:     Country, .model [1]
## # :        Exports = lag(level, 1) + remainder
##    Country .model  Year Exports level remainder
##    <fct>   <chr>  <dbl>   <dbl> <dbl>     <dbl>
##  1 Algeria ANN     1959    NA    39.5    NA
##  2 Algeria ANN     1960    39.0  39.1    -0.496
##  3 Algeria ANN     1961    46.2  45.1     7.12
##  4 Algeria ANN     1962    19.8  23.8   -25.3
##  5 Algeria ANN     1963    24.7  24.6     0.841
##  6 Algeria ANN     1964    25.1  25.0     0.534
##  7 Algeria ANN     1965    22.6  23.0    -2.39
##  8 Algeria ANN     1966    26.0  25.5     3.00
##  9 Algeria ANN     1967    23.4  23.8    -2.07
## 10 Algeria ANN     1968    23.1  23.2    -0.630
## # ... with 49 more rows
```

Simple Exponential Smoothing

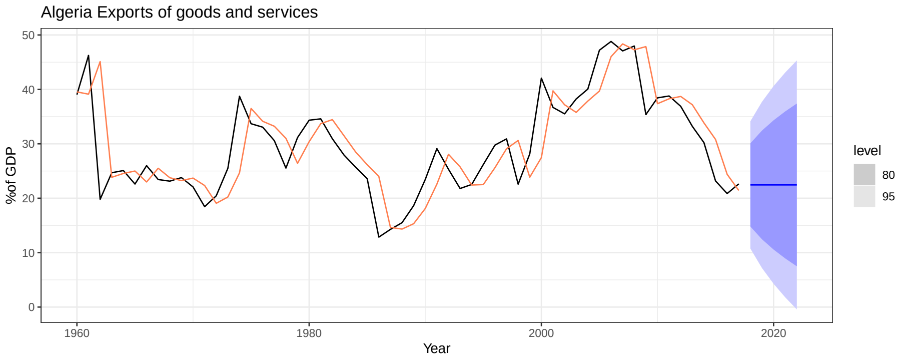
```
ses.fit %>% components() %>%
  left_join(fitted(ses.fit),
            by = c("Country",".model", "Year")) %>%
  relocate(.fitted, .after = level)
```

```
## # A tsibble: 59 x 7 [1Y]
## # Key:       Country, .model [1]
##    Country .model  Year Exports level .fitted remainder
##    <fct>   <chr>  <dbl>   <dbl> <dbl>   <dbl>     <dbl>
##  1 Algeria ANN     1959    NA    39.5    NA       NA
##  2 Algeria ANN     1960    39.0  39.1    39.5   -0.496
##  3 Algeria ANN     1961    46.2  45.1    39.1    7.12
##  4 Algeria ANN     1962    19.8  23.8    45.1  -25.3
##  5 Algeria ANN     1963    24.7  24.6    23.8    0.841
##  6 Algeria ANN     1964    25.1  25.0    24.6    0.534
##  7 Algeria ANN     1965    22.6  23.0    25.0   -2.39
##  8 Algeria ANN     1966    26.0  25.5    23.0    3.00
##  9 Algeria ANN     1967    23.4  23.8    25.5   -2.07
## 10 Algeria ANN     1968    23.1  23.2    23.8   -0.630
## # ... with 49 more rows
```

```r
ses.fit %>% forecast(h = 5) %>%
  autoplot(algeria) +
  labs(title = "Algeria Exports of goods and services",
       y = "%of GDP")
```



```r
ses.fit %>% forecast(h = 5) %>%
  autoplot(algeria) +
  geom_line(aes(y = .fitted), data = fitted(ses.fit), col = "coral") +
  labs(title = "Algeria Exports of goods and services", y = "%of GDP")
```



CAUTION!!!

Notice that our forecast projects a "flat-line" estimate into the future. This does not capture the possible positive trend in the data.
This is why a simple exponential smoothing should not be used on data with a trend or seasonal component.

Section 2

Exponential Smoothing with Trend

Exponential Smoothing Adjusted for Trend

Holt's (1957) Method

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

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, $0 \leq \beta^* \leq 1$.

Exponential Smoothing Adjusted for Trend

Holt's (1957) Method

- allows for evolving local linear trends in a time series
- can be used to generate forecasts
- Advantage: flexible to track changing in level and trend.

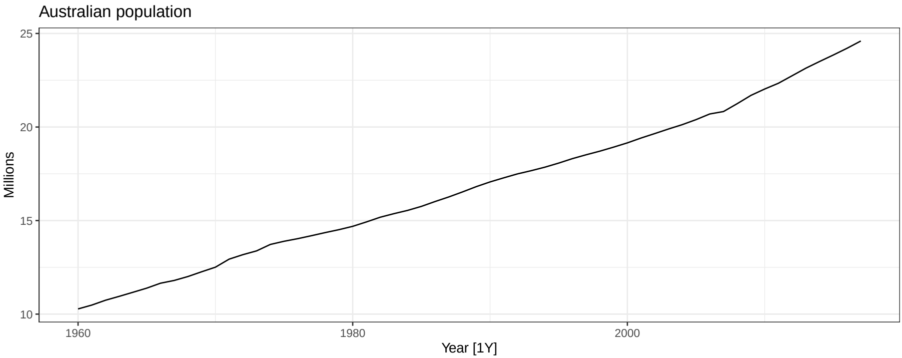
- The forecast function is no longer flat but trending.
- The h-step-ahead forecast is equal to the last estimated level plus h times the last estimated trend value.

Hence the forecasts are a linear function of h.

Holt's Method

```r
aus <- global_economy  %>%
  filter(Code == "AUS")  %>%
  mutate(Pop = Population / 1e6)

aus %>% autoplot(Pop) +
  labs(y = "Millions", title = "Australian population")
```

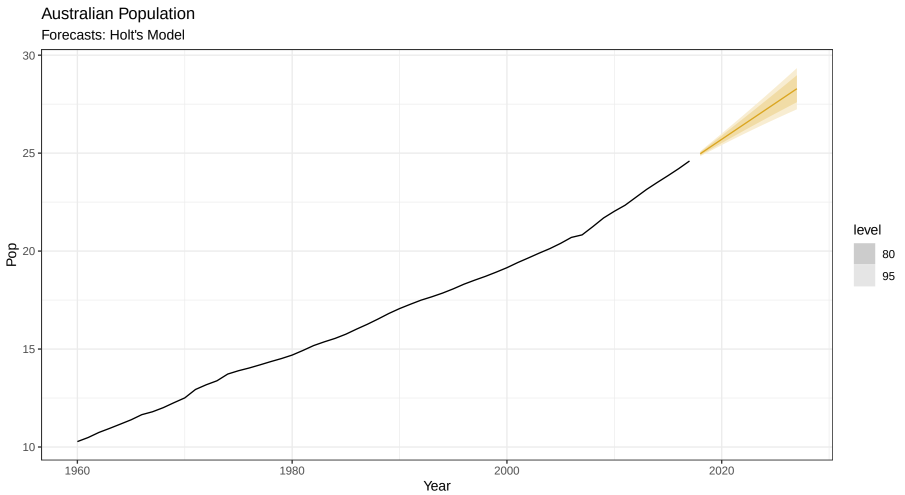


```r
aus.fit <- aus %>% model(ANN = ETS(Pop ~ error("A") + trend("A")))

aus.fit %>% report()
```

```
## Series: Pop
## Model: ETS(A,A,N)
##   Smoothing parameters:
##     alpha = 0.9999
##     beta  = 0.3266
##
##   Initial states:
##   l[0]   b[0]
##  10.05 0.2225
##
##   sigma^2:  0.0041
##
##    AIC   AICc    BIC
## -76.99 -75.83 -66.68
```

```r
aus.fit %>% forecast(h = 10) %>%
  autoplot(aus, color = "goldenrod") +
  labs(title = "Australian Population",
       subtitle = "Forecasts: Holt's Model")
```



Holt's Linear Trend Method with a Damped Trend

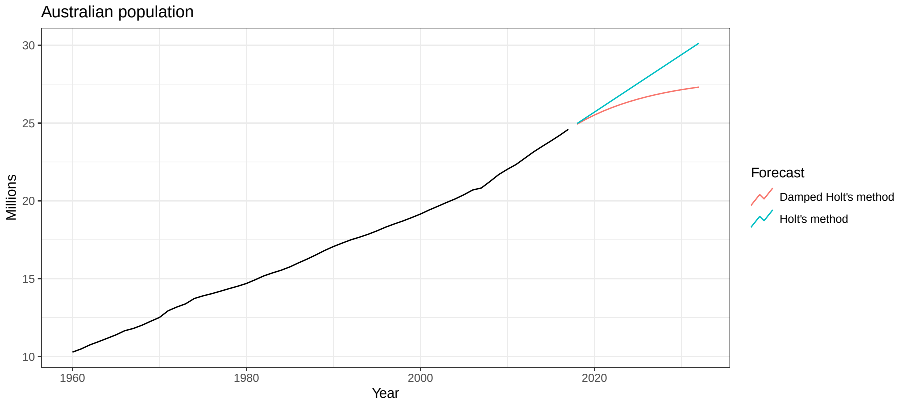
- The forecasts generated by Holt's linear method display a constant trend (increasing or decreasing) indefinitely into the future.
- Empirical evidence indicates that these methods tend to over-forecast, especially for longer forecast horizons.
- Gardner & McKenzie (1985) introduced a parameter that "dampens" the trend to a flat line some time in the future.

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

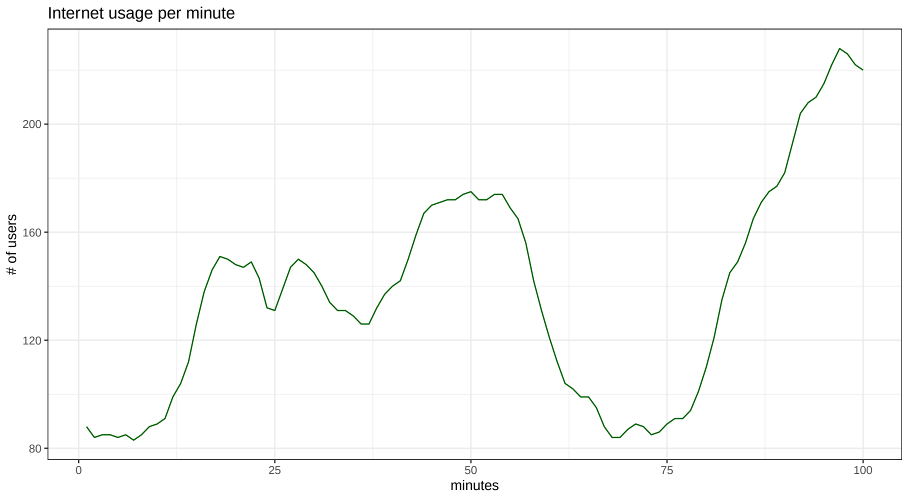
Holt's Linear Trend Method with a Damped Trend

- Damping parameter $0 < \phi < 1$.
- If $\phi = 1$, identical to Holt's linear trend.
- For values between 0 and 1, ϕ dampens the trend so that it approaches a constant some time in the future.
- In practice, ϕ is rarely less than 0.8 as the damping 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.
  - We usually restrict ϕ to a minimum of 0.8 and a maximum of 0.98.

```r
aus %>%
  model(
    `Holt's method` = ETS(Pop ~ error("A") + trend("A")),
    `Damped Holt's method` = ETS(Pop ~ error("A") + trend("Ad", phi = 0.9))
  ) %>% forecast(h = 15) %>%
  autoplot(aus, level = NULL) +
  labs(title = "Australian population", y = "Millions") +
  guides(colour = guide_legend(title = "Forecast"))
```



```r
net <- WWWusage %>% as_tsibble()
net %>% autoplot(value, col = "darkgreen") +
  labs(title = "Internet usage per minute",
       y = "# of users", x = "minutes")
```



Model Comparison

For this Example, we will use the CV Method.

```
net %>%
#Specify the width of estimation windows
  stretch_tsibble(.init = 10) %>%
  model(
    SES = ETS(value ~ error("A") + trend("N") + season("N")),
    Holt = ETS(value ~ error("A") + trend("A") + season("N")),
    Damped = ETS(value ~ error("A") + trend("Ad") +
                   season("N"))
  )  %>% forecast(h = 1)  %>% accuracy(net)
```

```
## # A tibble: 3 x 10
##   .model .type     ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>  <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Damped Test  0.288   3.69  3.00 0.347  2.26 0.663 0.636 0.336
## 2 Holt   Test  0.0610  3.87  3.17 0.244  2.38 0.701 0.668 0.296
## 3 SES    Test  1.46    6.05  4.81 0.904  3.55 1.06  1.04  0.803
```

Model Comparison

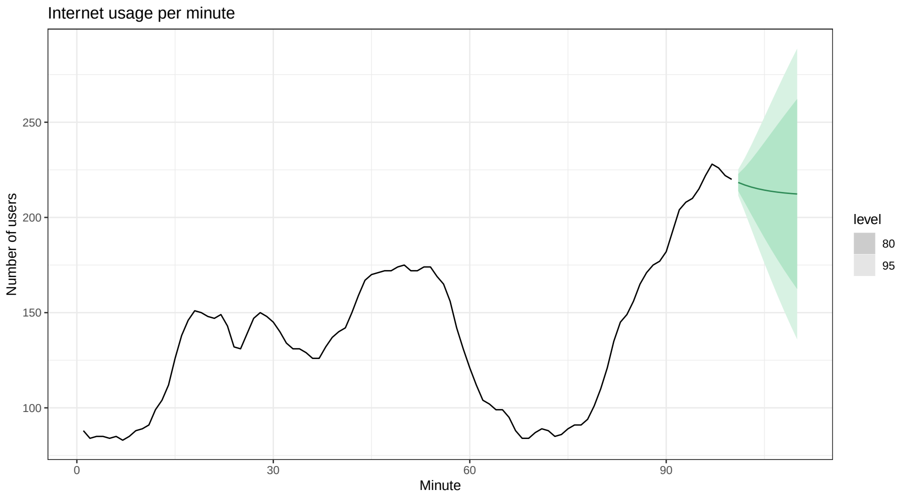
```
d.holt <- net %>%
  model(
    Damped = ETS(value ~ error("A") + trend("Ad") +
                   season("N"))
  )
d.holt %>% tidy()
```

```
## # A tibble: 5 x 3
##   .model term  estimate
##   <chr>  <chr>    <dbl>
## 1 Damped alpha   1.00
## 2 Damped beta    0.997
## 3 Damped phi     0.815
## 4 Damped l[0]   90.4
## 5 Damped b[0]   -0.0173
```

Takeaways

- Smoothing parameter for the slope (β^*) is estimated to be almost 1 $\Longrightarrow$ trend changes to mostly reflect the slope between the last two minutes of internet usage.
- α is very close to one $\Longrightarrow$ the level reacts strongly to each new observation.

```r
d.holt %>%
  forecast(h = 10) %>%
  autoplot(net, color = "seagreen") +
  labs(x="Minute", y="Number of users",
       title = "Internet usage per minute")
```



Section 3

Exponential Smoothing with Trend & Seasonality

Holt-Winter's Exponential Smoothing Methods

Adjusted for Trend & Seasonal Variation

- Three parameter linear and seasonal exponential smoothing method
- an extension of Holt's method
- one additional equation for seasonality
- Has both an additive and multiplicative method.

Additive Form

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

Multiplicative Form

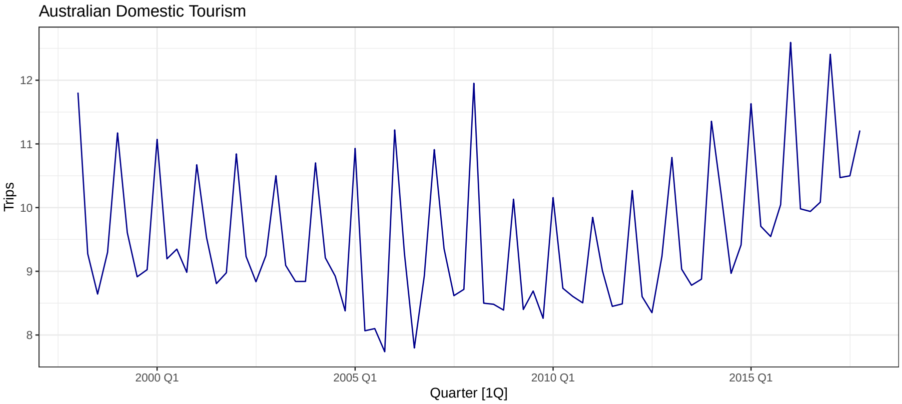
$$
\begin{aligned}
\hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t+h-m(k+1)} \\
\ell_t &= \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma\frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}.
\end{aligned}
$$

Holt-Winter's Exponential Smoothing Methods

- Winter's method provides an easy way to account for seasonality when data have a seasonal pattern.
- In general, exponential smoothing methods have the major advantages of low cost and simplicity.
- The methods assign weights that decline exponentially as the observations get older.

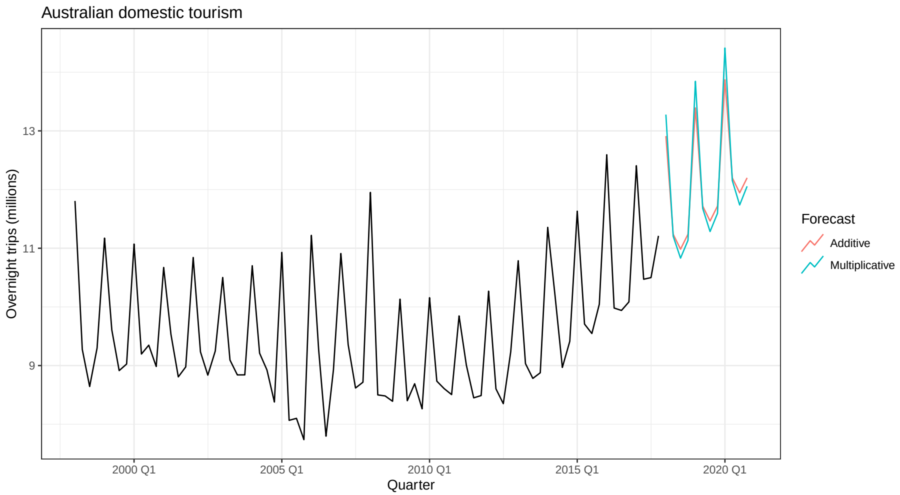
```r
aus_holidays <- tourism  %>%
  filter(Purpose == "Holiday")  %>%
  summarise(Trips = sum(Trips)/1e3)

aus_holidays %>% autoplot(Trips, col = "blue4") +
  labs(title = "Australian Domestic Tourism")
```



```r
fit <- aus_holidays  %>%
  model(
    Additive = ETS(Trips ~ error("A") + trend("A") +
                          season("A")),
    Multiplicative = ETS(Trips ~ error("M") + trend("A") +
                                season("M"))
  )
fc <- fit  %>% forecast(h = "3 years")
```

```
fc %>%
  autoplot(aus_holidays, level = NULL) +
  labs(title="Australian domestic tourism",
       y="Overnight trips (millions)") +
  guides(colour = guide_legend(title = "Forecast"))
```



```r
fit %>% tidy()
```

```
## # A tibble: 18 x 3
##    .model         term   estimate
##    <chr>          <chr>     <dbl>
##  1 Additive       alpha  0.262
##  2 Additive       beta   0.0431
##  3 Additive       gamma  0.000100
##  4 Additive       l[0]   9.79
##  5 Additive       b[0]   0.0211
##  6 Additive       s[0]  -0.534
##  7 Additive       s[-1] -0.670
##  8 Additive       s[-2] -0.294
##  9 Additive       s[-3]  1.50
## 10 Multiplicative alpha  0.224
## 11 Multiplicative beta   0.0304
## 12 Multiplicative gamma  0.000100
## 13 Multiplicative l[0]  10.0
## 14 Multiplicative b[0]  -0.0114
## 15 Multiplicative s[0]   0.943
## 16 Multiplicative s[-1]  0.927
## 17 Multiplicative s[-2]  0.969
## 18 Multiplicative s[-3]  1.16
```

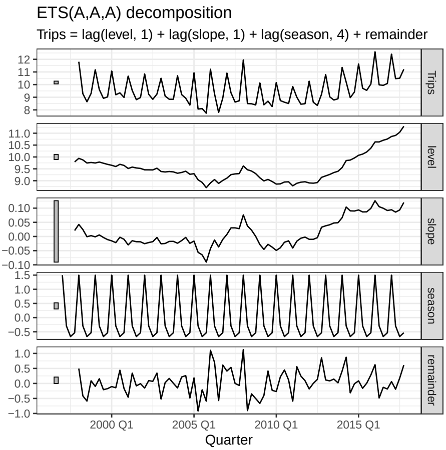
Takeaways

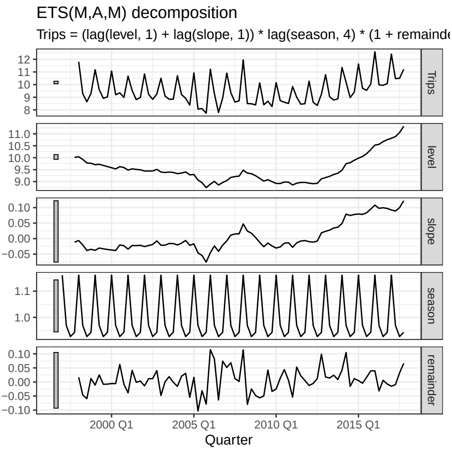
- The small value of γ for the multiplicative model means that the seasonal component hardly changes over time.
- The small value of β^* means the slope component hardly changes over time (compare the vertical scales of the slope and level components).

Model Components

```
g1 <- fit %>% select(Additive) %>% components() %>%  autoplot()
g2 <- fit %>% select(Multiplicative) %>% components() %>%   autoplot()

gridExtra::grid.arrange(g1,g2, ncol = 2)
```





Holt'Winter's Damped Trend

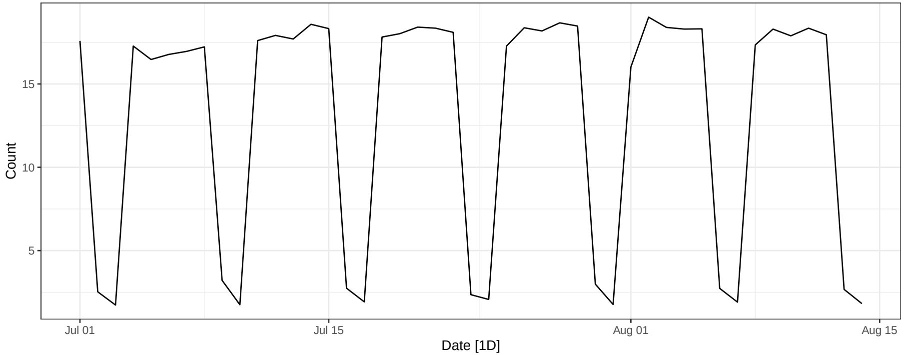
Damping is possible with both additive and multiplicative Holt-Winters' methods. A method that often provides accurate and robust forecasts for seasonal data is the **Holt-Winters method with a damped trend and multiplicative seasonality**:

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

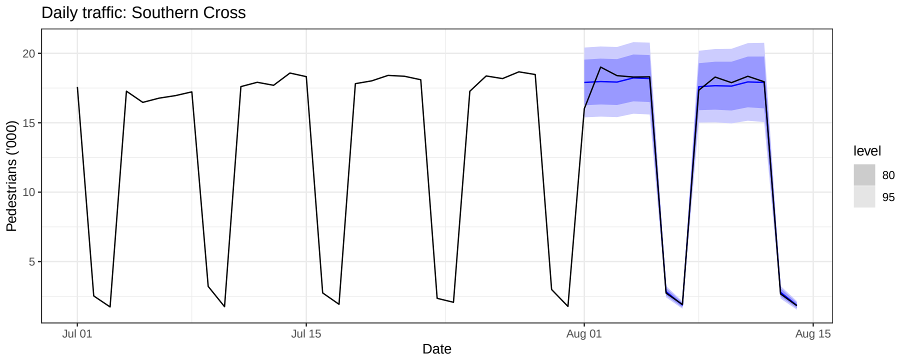
Holt-Winters method with daily data

```
sth_cross_ped <- pedestrian %>%
  filter_index("2016-07-01" ~ "2016-08-14") %>%
  filter(Sensor == "Southern Cross Station") %>%
  index_by(Date) %>%
  summarise(Count = sum(Count)/1000)

sth_cross_ped  %>%  autoplot(Count)
```

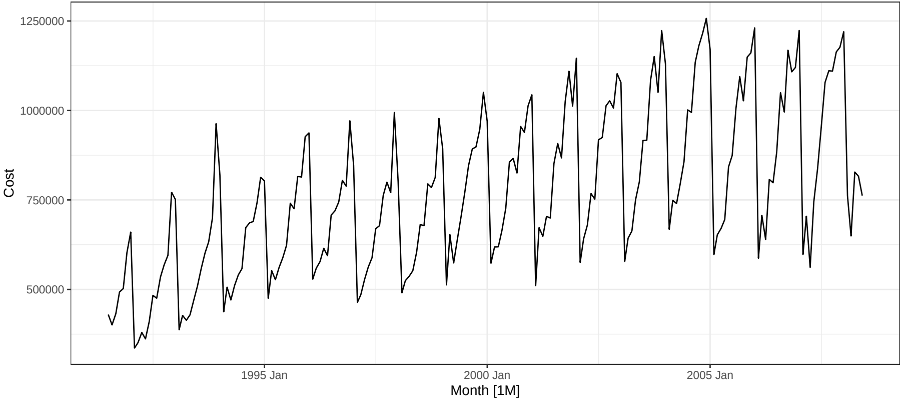


```
sth_cross_ped %>%
  filter_index(.~"2016-07-31") %>%
  model(
    hw = ETS(Count ~ error("M") + trend("Ad") + season("M"))
  ) %>%
  forecast(h = "2 weeks") %>%
  autoplot(sth_cross_ped) +
  labs(title = "Daily traffic: Southern Cross",
       y="Pedestrians ('000)")
```



Which model would you have chosen?

```
h02 <- PBS %>% filter(ATC2 == "H02") %>%
  summarise(Cost = sum(Cost))
h02 %>% autoplot(Cost)
```



Automated

```
h02 %>% model(ETS(Cost)) %>% report()
```

```
## Series: Cost
## Model: ETS(M,Ad,M)
##   Smoothing parameters:
##     alpha = 0.3071
##     beta  = 0.0001007
##     gamma = 0.0001007
##     phi   = 0.9775
##
##   Initial states:
##    l[0] b[0]   s[0] s[-1]  s[-2]  s[-3]  s[-4] s[-5] s[-6] s[-7] s[-8] s
##  417269 8206 0.8717 0.826 0.7563 0.7733 0.6872 1.284 1.325  1.18 1.164 1
##  s[-10] s[-11]
##   1.048 0.9806
##
##   sigma^2:  0.0046
##
##  AIC AICc  BIC
## 5515 5519 5575
```

User Defined

```
h02 %>% model(ETS(Cost ~ error("A") +
                    trend("A") + season("A"))) %>%
  report()
```

```
## Series: Cost
## Model: ETS(A,A,A)
##   Smoothing parameters:
##     alpha = 0.1702
##     beta  = 0.006311
##     gamma = 0.4546
##
##   Initial states:
##     l[0] b[0]   s[0]   s[-1]   s[-2]   s[-3]   s[-4]  s[-5]  s[-6]  s[-7]
##  409706 9097 -99075 -136602 -191496 -174531 -241437 210644 244644 145368
##  s[-9] s[-10] s[-11]
##  84458  39132 -11674
##
##   sigma^2:  3.499e+09
##
##  AIC AICc  BIC
## 5585 5589 5642
```

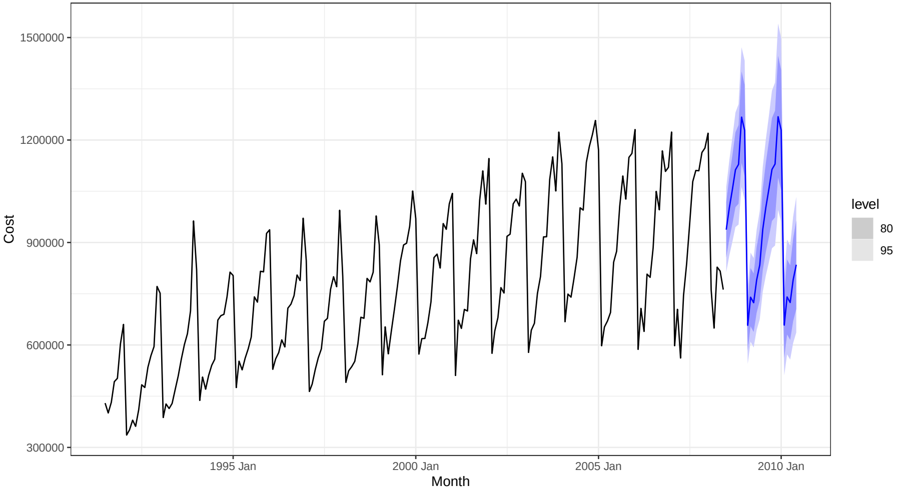
Which is better?

```
h02 %>% model(auto = ETS(Cost),
              AAA = ETS(Cost ~ error("A") + trend("A") +
                              season("A"))) %>%
  accuracy()
```

```
## # A tibble: 2 x 10
##   .model .type          ME   RMSE    MAE     MPE  MAPE  MASE
##   <chr>  <chr>       <dbl>  <dbl>  <dbl>   <dbl> <dbl> <dbl>
## 1 auto   Training  2461. 51102. 38649. -0.0127  4.99 0.638
## 2 AAA    Training -5780. 56784. 43378. -1.30    6.05 0.716
```

Forecast with Best Model

```
h02 %>% model(ETS(Cost)) %>% forecast() %>%
  autoplot(h02)
```



Your turn

Try applying the Holt-Winter's Method to the `Gas` data in the `aus_production` series.

1. First plot the data and determine whether additive or multiplicative seasonality would be most appropriate.
2. Experiment with specifications with and without a damped trend.
3. Which model does best over the training period?
4. Hold out the last 3 years of data and provide forecasts from each of your candidate models.
5. Can you check if the residuals from the "best method" according to the MAE criteria look like white noise.
6. Can you now use these methods to forecast the `seasonal` and `nonseasonal` elements of variables? (see Module 5!)