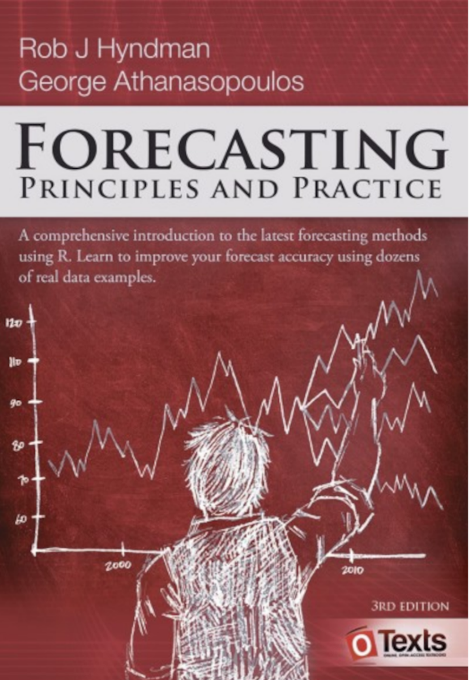


8. Exponential smoothing

8.3 Methods with seasonality

OTexts.org/fpp3/

Holt-Winters additive method

Holt and Winters extended Holt's method to capture seasonality.

Component form

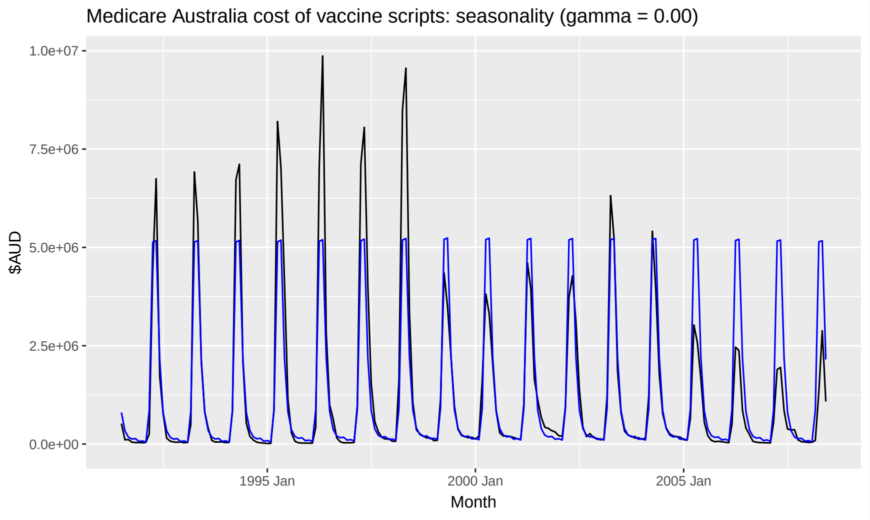
$$
\begin{aligned}
\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)} \\
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}
\end{aligned}
$$

- k = integer part of $(h-1)/m$. Ensures estimates from the final year are used for forecasting.
- Parameters: $0 \le \alpha \le 1$, $0 \le \beta^* \le 1$, $0 \le \gamma \le 1-\alpha$ and m = period of seasonality (e.g. $m = 4$ for quarterly data).

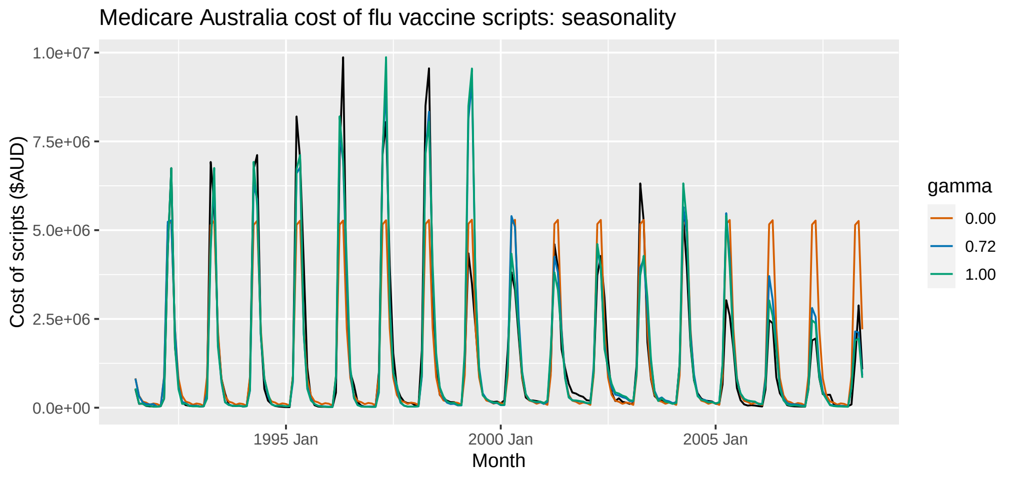
Holt-Winters additive method

- Seasonal component is usually expressed as
  $s_t = \gamma^*(y_t - \ell_t) + (1 - \gamma^*)s_{t-m}.$
- Substitute in for ℓ_t:
  $s_t = \gamma^*(1 - \alpha)(y_t - \ell_{t-1} - b_{t-1}) + [1 - \gamma^*(1 - \alpha)]s_{t-m}$
- We set $\gamma = \gamma^*(1 - \alpha)$.
- The usual parameter restriction is $0 \leq \gamma^* \leq 1$, which translates to $0 \leq \gamma \leq (1 - \alpha)$.

Exponential smoothing: seasonality



Exponential smoothing: seasonality



Holt-Winters multiplicative method

Seasonal variations change in proportion to the level of the series.

Component form

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

- k is integer part of $(h-1)/m$.
- Additive method: s_t in absolute terms — within each year $\sum_i s_i \approx 0$.
- Multiplicative method: s_t in relative terms — within each year $\sum_i s_i \approx m$.

Example: Australian holiday tourism

```r
aus_holidays <- tourism |>
  filter(Purpose == "Holiday") |>
  summarise(Trips = sum(Trips))
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()
```

Estimated coefficients

```
tidy(fit) |>
  spread(.model, estimate)
```

```
## # A tibble: 9 x 3
##   term     additive multiplicative
##   <chr>       <dbl>          <dbl>
## 1 alpha    0.236          0.186
## 2 b[0]   -37.4          -33.4
## 3 beta     0.0298         0.0248
## 4 gamma    0.000100       0.000100
## 5 l[0]  9899.          9853.
## 6 s[-1] -684.             0.926
## 7 s[-2] -290.             0.970
## 8 s[-3] 1512.             1.16
## 9 s[0]  -538.             0.943
```

Estimated components

```r
components(fit) |> filter(.model=="additive") |>
  left_join(fitted(fit), by = c(".model", "Quarter"))
```

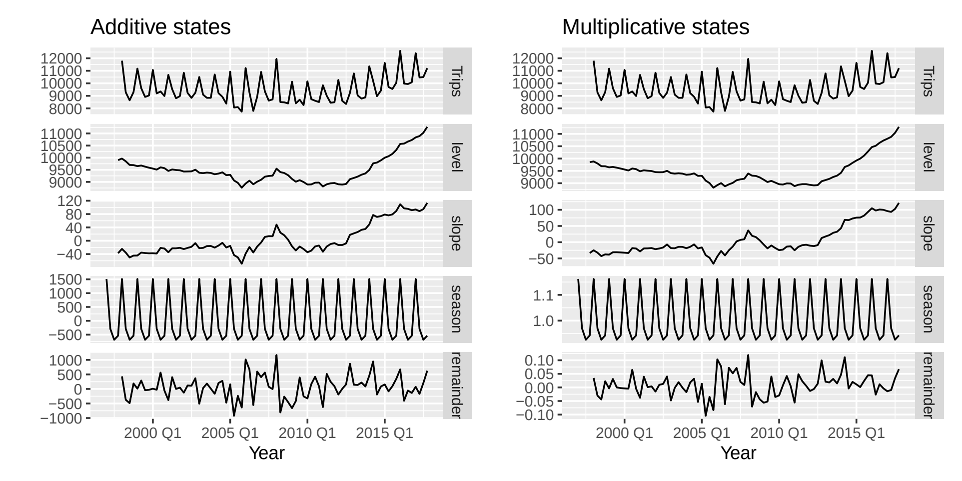
```
## # A dable: 84 x 8 [1Q]
## # Key:     .model [1]
## # :        Trips = lag(level, 1) + lag(slope, 1) + lag(season, 4) +
## #   remainder
##    .model   Quarter  Trips level slope season remainder .fitted
##    <chr>      <qtr>  <dbl> <dbl> <dbl>  <dbl>     <dbl>   <dbl>
##  1 additive 1997 Q1    NA     NA    NA   1512.       NA      NA
##  2 additive 1997 Q2    NA     NA    NA   -290.       NA      NA
##  3 additive 1997 Q3    NA     NA    NA   -684.       NA      NA
##  4 additive 1997 Q4    NA   9899. -37.4  -538.       NA      NA
##  5 additive 1998 Q1 11806.  9964. -24.5  1512.      433.  11373.
##  6 additive 1998 Q2  9276.  9851. -35.6  -290.     -374.   9649.
##  7 additive 1998 Q3  8642.  9700. -50.2  -684.     -489.   9131.
```

Estimated components

```r
components(fit) |> filter(.model=="multiplicative") |>
  left_join(fitted(fit), by = c(".model", "Quarter"))
```

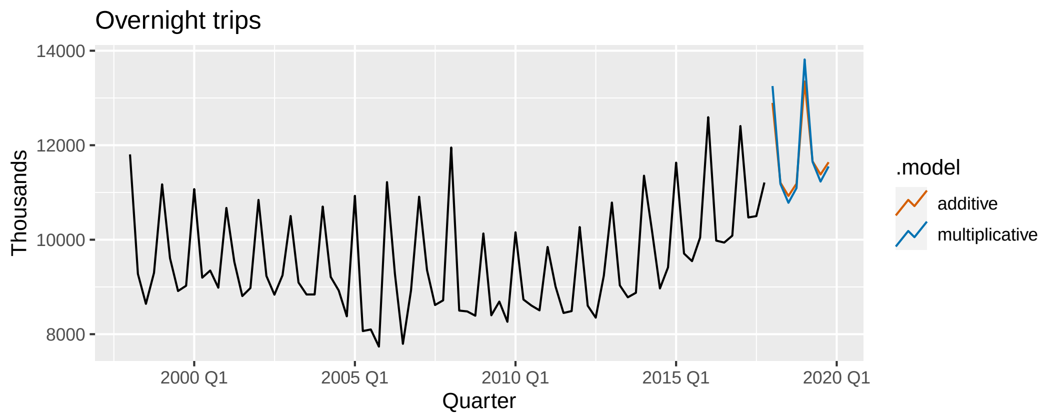
```
## # A dable: 84 x 8 [1Q]
## # Key:       .model [1]
## # :          Trips = lag(level, 1) + lag(slope, 1) + lag(season, 4) +
## #   remainder
##    .model         Quarter  Trips level slope season remainder .fitted
##    <chr>            <qtr>  <dbl> <dbl> <dbl>  <dbl>     <dbl>   <dbl>
##  1 multiplicative 1997 Q1    NA    NA    NA    1.16   NA           NA
##  2 multiplicative 1997 Q2    NA    NA    NA    0.970  NA           NA
##  3 multiplicative 1997 Q3    NA    NA    NA    0.926  NA           NA
##  4 multiplicative 1997 Q4    NA  9853. -33.4  0.943  NA           NA
##  5 multiplicative 1998 Q1 11806. 9883. -24.9  1.16    0.0348   11409.
##  6 multiplicative 1998 Q2  9276. 9803. -32.3  0.970  -0.0299    9562.
##  7 multiplicative 1998 Q3  8642. 9690. -43.0  0.926  -0.0444    9044.
```

Estimated components



Example: Australian holiday tourism

```
fc |>
  autoplot(aus_holidays, level = NULL) +
  labs(y = "Thousands", title = "Overnight trips")
```



Holt-Winters damped method

Often the single most accurate forecasting method for seasonal data:

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

Holt-Winters with daily data

```
sth_cross_ped <- pedestrian |>
  filter(
    Date >= "2016-07-01",
    Sensor == "Southern Cross Station"
  ) |>
  index_by(Date) |>
  summarise(Count = sum(Count) / 1000)
sth_cross_ped |>
  filter(Date <= "2016-07-31") |>
  model(hw = ETS(Count ~ error("M") + trend("Ad") + season("M"))) |>
  forecast(h = "2 weeks") |>
  autoplot(sth_cross_ped |> filter(Date <= "2016-08-14")) +
  labs(
    title = "Daily traffic: Southern Cross",
    y = "Pedestrians ('000)"
  )
```

Holt-Winters with daily data