

ETF3231/5231: Business forecasting

Week 3: Time series decomposition

<https://bf.numbat.space/>



Outline

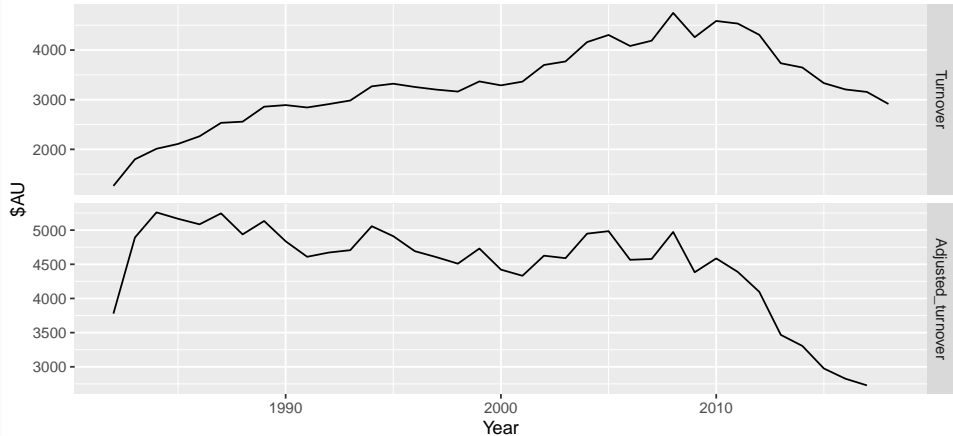
- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition
- 5 History of time series decomposition
- 6 STL decomposition

Outline

- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition
- 5 History of time series decomposition
- 6 STL decomposition

Inflation adjustments

Turnover: Australian print media industry



Mathematical transformations

If the data show different variation at different levels of the series, then a transformation can be useful.

Mathematical transformations

If the data show different variation at different levels of the series, then a transformation can be useful.

Denote original observations as y_1, \dots, y_T and transformed observations as w_1, \dots, w_T .

Mathematical transformations for stabilizing variation

Square root	$w_t = \sqrt{y_t}$	\downarrow
Cube root	$w_t = \sqrt[3]{y_t}$	Increasing
Logarithm	$w_t = \log(y_t)$	strength

Logarithms, in particular, are useful because they are more interpretable: changes in a log value are **relative (percent) changes on the original scale**.

Box-Cox transformations

Each of these transformations is close to a member of the family of **Box-Cox transformations**:

$$w_t = \begin{cases} \log(y_t), & \lambda = 0; \\ (\text{sign}(y_t)|y_t|^\lambda - 1)/\lambda, & \lambda \neq 0. \end{cases}$$

- Actually the Bickel-Doksum transformation (allowing for $y_t < 0$)
- $\lambda = 1$: (No substantive transformation)
- $\lambda = \frac{1}{2}$: (Square root plus linear transformation)
- $\lambda = 0$: (Natural logarithm)
- $\lambda = -1$: (Inverse plus 1)

Transformations

- Often **no transformation** needed.
- **Simple transformations** are easier to explain and work well enough.
- Transformations can have **very large effect on PI**.
- If some data are **zero or negative**, then use $\lambda > 0$.
- `log1p()` can also be useful for **data with zeros**.
- Choosing logs is a simple way to force forecasts to be positive
- Transformations must be reversed to obtain forecasts on the original scale. (Handled automatically by `fab1e`.)

Transformations

- Often **no transformation** needed.
- **Simple transformations** are easier to explain and work well enough.
- Transformations can have **very large effect on PI**.
- If some data are **zero or negative**, then use $\lambda > 0$.
- $\log_{1p}()$ can also be useful for **data with zeros**.
- Choosing logs is a simple way to force forecasts to be positive
- Transformations must be reversed to obtain forecasts on the original scale. (Handled automatically by `fab1e`.)

We often use logs.

Outline

- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition
- 5 History of time series decomposition
- 6 STL decomposition

Time series patterns

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

$$y_t = f(S_t, T_t, R_t)$$

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

Additive decomposition: $y_t = S_t + T_t + R_t$.

Multiplicative decomposition: $y_t = S_t \times T_t \times R_t$.

Time series decomposition

- **Additive model** appropriate if magnitude of seasonal fluctuations does not vary with level.
- If seasonal are proportional to level of series, then **multiplicative model** appropriate.
- Multiplicative decomposition more prevalent with economic series
- Alternative: use a Box-Cox transformation, and then use additive decomposition.
- Logs turn multiplicative relationship into an additive relationship:

$$y_t = S_t \times T_t \times R_t \quad \Rightarrow \quad \log y_t = \log S_t + \log T_t + \log R_t.$$

US Retail Employment

```
us_retail_employment <- us_employment |>
  filter(year(Month) >= 1990, Title == "Retail Trade") |>
  select(-Series_ID)
us_retail_employment
```

```
## # A tsibble: 357 x 3 [1M]
##       Month Title      Employed
##       <mth> <chr>      <dbl>
## 1 1990 Jan Retail Trade 13256.
## 2 1990 Feb Retail Trade 12966.
## 3 1990 Mar Retail Trade 12938.
## 4 1990 Apr Retail Trade 13012.
## 5 1990 May Retail Trade 13108.
## 6 1990 Jun Retail Trade 13183.
## 7 1990 Jul Retail Trade 13170.
## 8 1990 Aug Retail Trade 13160.
## 9 1990 Sep Retail Trade 13113.
## 10 1990 Oct Retail Trade 13185.
```

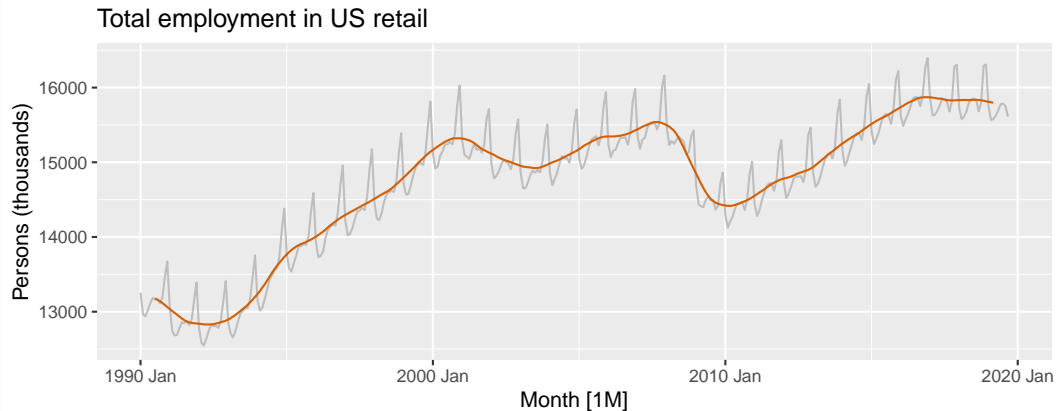
Seasonal adjustment

- We use estimates of S_t based on past values to seasonally adjust a current value.
- Seasonally adjusted series reflect **remainders** as well as **trend**. Therefore they are not “smooth” and “downturns” or “upturns” can be misleading.
- It is better to use the trend-cycle component to look for turning points.

Outline

- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition
- 5 History of time series decomposition
- 6 STL decomposition

Moving average trend-cycle



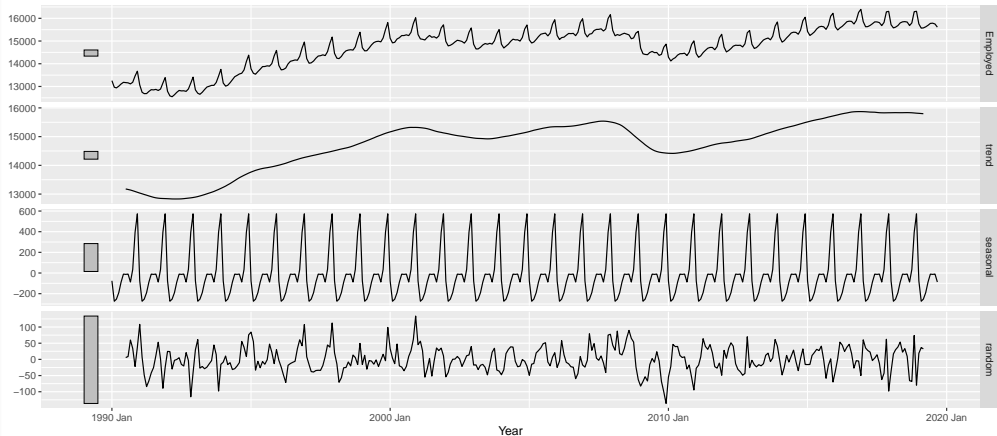
Outline

- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition**
- 5 History of time series decomposition
- 6 STL decomposition

Additive classical decomposition

Classical multiplicative decomposition of total US retail employment

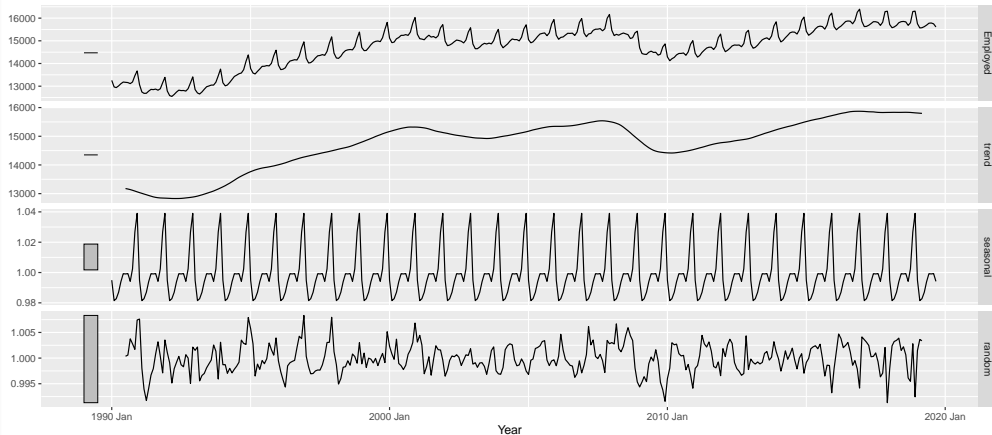
Employed = trend + seasonal + random



Multiplicative classical decomposition

Classical multiplicative decomposition of total US retail employment

Employed = trend * seasonal * random



Comments on classical decomposition

- Estimate of trend is **unavailable** for first few and last few observations.
- **Seasonal component repeats** from year to year. May not be realistic.
- **Not robust to outliers.**
- Newer methods designed to overcome these problems.

Outline

- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition
- 5 History of time series decomposition
- 6 STL decomposition

History of time series decomposition

- Classical method originated in 1920s.
- Census II method introduced in 1957. Basis for X-11 method and variants (including X-12-ARIMA, X-13-ARIMA)
- STL method introduced in 1983
- TRAMO/SEATS introduced in 1990s.

National Statistics Offices

- ABS uses X-12-ARIMA
- US Census Bureau uses X-13ARIMA-SEATS
- Statistics Canada uses X-12-ARIMA
- ONS (UK) uses X-12-ARIMA
- EuroStat use X-13ARIMA-SEATS

Outline

- 1 Transformations and adjustments
- 2 Time series components
- 3 Moving averages
- 4 Classical decomposition
- 5 History of time series decomposition
- 6 STL decomposition

STL decomposition

- STL: “Seasonal and Trend decomposition using Loess”
- Very versatile and robust.
- Unlike X-12-ARIMA, STL will handle any type of seasonality.
- Seasonal component allowed to change over time, and rate of change controlled by user.
- Smoothness of trend-cycle also controlled by user.
- Robust to outliers
- No trading day or calendar adjustments.
- Only additive.
- Take logs to get multiplicative decomposition.
- Use Box-Cox transformations to get other decompositions.

STL decomposition

```
us_retail_employment |>  
  model(STL(Employed)) |>  
  components()
```

- `trend(window = ?)` controls wiggleness of trend component.
- `season(window = ?)` controls variation on seasonal component.
- `season(window = 'periodic')` is equivalent to an infinite window.

Default setting

- Season window = 13
- Trend window = $\text{nextodd}(\text{ceiling}((1.5 \times \text{period}) / (1 - (1.5 / \text{s.window}))))$
- Robust robust=FALSE