Time Series Analysis & Forecasting Using R

11. Multivariate modelling

tsibble feasts Fable

Outline

Outline

Multivariate modelling

What is it?

- Multiple models?
- Multiple time series?
- Multiple variables!

Multivariate modelling

Why it is useful?

- Introduce the importance of modelling multiple time series together.
- Real-world applications (e.g., macroeconomic variables, financial data).

Outline

1. Multivariate modelling
2. Vector Autoregression (VAR)
3. Vector Error Correction Models (VECM)
4. VARIMA Models
5. Forecast reconciliation
6. Hierarchical and grouped time series
7. Optimal combination forecasts

Autoregression (AR)

The AR(p) model for a **univariate** time series y_t is:

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} + \varepsilon_t$$

where:

- y_t is the time series at time t,
- $a_1, a_2, \ldots, a_p$ are the coefficients for the autoregressive lags,
- p is the order of the AR process,
- ε_t is the white noise error term at time t.

Cross correlations

The Cross-Correlation Function (CCF) measures the correlation between two time series at different time lags.

The CCF provides insight into how variables influence each other over time, suggesting the appropriate number of lags for the VAR model.

Endogeneity

Endogeneity occurs when an explanatory variable is correlated with the error term in a model.

Endogeneity

- Leads to biased and inconsistent estimates in time series models.
- Compromises the validity of hypothesis testing and forecasting.

Common Causes of Endogeneity in Time Series:

- Simultaneity: Variables mutually affect each other (e.g., supply and demand).

Vector Autoregression (VAR)

The general form of a VAR model with k time series variables $Y_t = (y_{1t}, y_{2t}, \ldots, y_{kt})'$ is:

$$Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \cdots + A_p Y_{t-p} + \varepsilon_t$$

where:

- Y_t is a vector of endogenous variables at time t,
- A_i are coefficient matrices (each of size $k \times k$),
- p is the lag order,
- ε_t is a vector of error terms (with mean zero and covariance matrix Σ)

Vector Autoregression (VAR)

In matrix form, the VAR(p) model is represented as:

$$\Phi(L)Y_t = \varepsilon_t$$

where:

- Y_t is a $k \times 1$ vector of endogenous variables $Y_t = \begin{pmatrix} y_{1t} \\ y_{2t} \\ \vdots \\ y_{kt} \end{pmatrix}$,
- $\Phi(L)$ is a matrix polynomial of AR coefficients in the lag operator L

Vector Autoregression (VAR)

TODO:

- Multivariate forecasting:
  - Demonstrate model estimation
  - Show forecasts
- Granger Causality:
  - Concept and how it's tested in VAR.
  - Provide an example of testing Granger causality between two series.
- Impulse Response Functions (IRFs):
  - Explain the concept of IRFs and how they reveal the response of one variable to shocks in another.
  - Show an example using IRFs in a VAR setting.

Outline

Vector Error Correction Models (VECM)

The VECM is used when variables are cointegrated. The form of a VECM for Y_t is:

$$\Delta Y_t = \Pi Y_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \Delta Y_{t-i} + \varepsilon_t$$

where:

- $\Delta Y_t = Y_t - Y_{t-1}$ is the first difference of Y_t,
- Π is the cointegration matrix ($\Pi = \alpha\beta'$, where α are adjustment coefficients and β are cointegration vectors),
- Γ_i are short-run adjustment coefficients,

Cointegration

Definition: Cointegration occurs when two or more non-stationary time series, each integrated of the same order, are linked by a long-term equilibrium relationship.

For example: If x and y are non-stationary, and a linear combination of them is stationary, then x and y are cointegrated.

The Johansen test

The Johansen test helps to determine how many long-term equilibrium relationships exist between the variables, guiding model specification.

Estimating a VECM

TODO:

- Explain the link between cointegration and VECM, and between VECM and VAR.
- Discuss long-term equilibrium relationships and short-term dynamics.
- Provide an example of a VECM model and interpretation of error correction terms.

Outline

VARIMA Models

The VARIMA model extends an ARIMA model to multiple time series.

It combines autoregression (AR), differencing (I), and moving average (MA) components:

$$\Phi(L)\Delta^d Y_t = \Theta(L)\varepsilon_t$$

where:

- $\Phi(L)$ is the matrix polynomial in the lag operator L for the AR part,

VARIMA Models

TODO:

- Introduction to VARIMA:
  - Discuss the role of differencing for non-stationary data.
- Modelling Process:
  - Model selection for VARIMA
  - Briefly compare it to VAR and VECM in terms of applicability.

Outline

Outline

Australian tourism

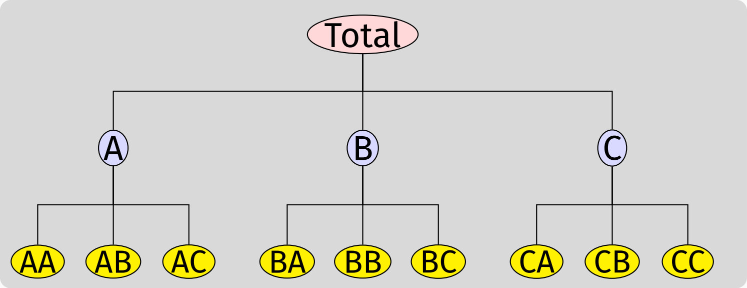
```
tourism
```

```
# A tsibble: 24,320 x 5 [1Q]
# Key:       Region, State, Purpose [304]
   Quarter Region   State           Purpose  T
     <qtr> <chr>    <chr>           <chr>    <
 1 1998 Q1 Adelaide South Australia Business
 2 1998 Q2 Adelaide South Australia Business
 3 1998 Q3 Adelaide South Australia Business
 4 1998 Q4 Adelaide South Australia Business
 5 1999 Q1 Adelaide South Australia Business
 6 1999 Q2 Adelaide South Australia Business
 7 1999 Q3 Adelaide South Australia Business
 8 1999 Q4 Adelaide South Australia Business
 9 2000 Q1 Adelaide South Australia Business
10 2000 Q2 Adelaide South Australia Business
# i 24,310 more rows
```

- Quarterly data on visitor nights, 1998:Q1 – 2017:Q4
- From: *National Visitor Survey*, based on annual interviews of 120,000 Australians aged 15+, collected by Tourism Research Australia.
- Split by 8 states and 76 regions
- Split by purpose of travel
  - Holiday
  - Visiting friends and relatives (VFR)
  - Business
  - Other
- 304 bottom-level series

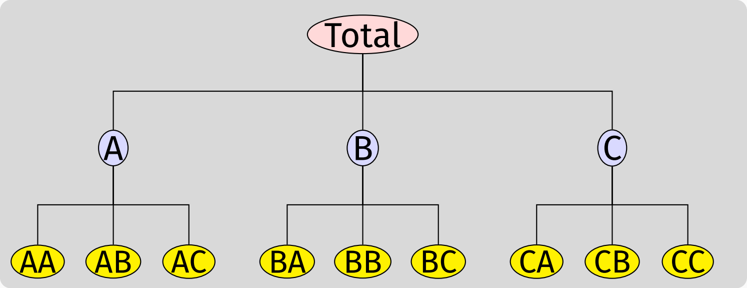
Hierarchical time series

A **hierarchical time series** is a collection of several time series that are linked together in a hierarchical structure.



Hierarchical time series

A **hierarchical time series** is a collection of several time series that are linked together in a hierarchical structure.

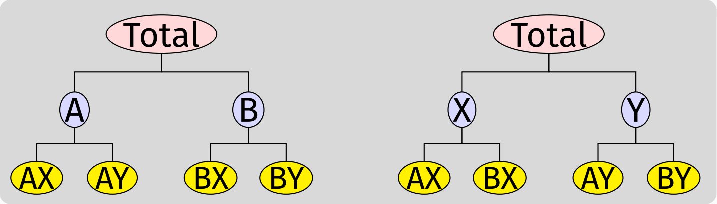


Examples

- PBS sales by ATC groups
- Tourism demand by states, regions

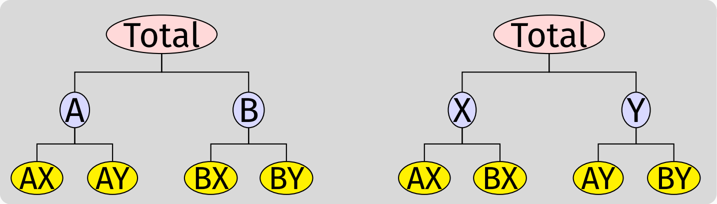
Grouped time series

A **grouped time series** is a collection of time series that can be grouped together in a number of non-hierarchical ways.



Grouped time series

A **grouped time series** is a collection of time series that can be grouped together in a number of non-hierarchical ways.



Examples

- Tourism by state and purpose of travel
- Retail sales by product groups/sub groups, and by countries/regions

Creating aggregates

```r
tourism |>
  aggregate_key(Purpose * (State / Region), Trips = sum(Trips)) |>
  filter(Quarter == yearquarter("1998 Q1")) |>
  print(n = 15)
```

```
# A tsibble: 425 x 5 [1Q]
# Key:       Purpose, State, Region [425]
   Quarter Purpose      State              Region        Trips
     <qtr> <chr*>       <chr*>             <chr*>        <dbl>
 1 1998 Q1 <aggregated> <aggregated>       <aggregated>  23182.
 2 1998 Q1 Business     <aggregated>       <aggregated>   3599.
 3 1998 Q1 Holiday      <aggregated>       <aggregated>  11806.
 4 1998 Q1 Other        <aggregated>       <aggregated>    680.
 5 1998 Q1 Visiting     <aggregated>       <aggregated>   7098.
 6 1998 Q1 <aggregated> ACT                <aggregated>    551.
 7 1998 Q1 <aggregated> New South Wales    <aggregated>   8040.
 8 1998 Q1 <aggregated> Northern Territory <aggregated>    181.
 9 1998 Q1 <aggregated> Queensland         <aggregated>   4041.
10 1998 Q1 <aggregated> South Australia    <aggregated>   1735.
11 1998 Q1 <aggregated> Tasmania           <aggregated>    982.
12 1998 Q1 <aggregated> Victoria           <aggregated>   6010.
13 1998 Q1 <aggregated> Western Australia  <aggregated>   1641.
14 1998 Q1 <aggregated> ACT                Canberra        551.
```

Creating aggregates

- Similar to `summarise()` but using the key structure
- A grouped structure is specified using `grpl * grp2`
- A nested structure is specified via `parent / child`.
- Groups and nesting can be mixed:

```
(country/region/city) * (brand/product)
```

- All possible aggregates are produced.
- These are useful when forecasting at different levels of aggregation.

Forecast reconciliation: the problem

1. How to forecast time series at all nodes such that the forecasts add up in the same way as the original data?
2. Can we exploit relationships between the series to improve the forecasts?

Forecast reconciliation: the problem

1. How to forecast time series at all nodes such that the forecasts add up in the same way as the original data?
2. Can we exploit relationships between the series to improve the forecasts?

Forecast reconciliation: the solution

1. Forecast all series at all levels of aggregation using an automatic forecasting algorithm.
   (e.g., `ETS`, `ARIMA`, ...)
2. Reconcile the resulting forecasts so they add up correctly using least squares optimization (i.e., find closest reconciled forecasts to the original forecasts).
3. This is available using `reconcile()`.

Forecast reconciliation

```
tourism |>
  aggregate_key(Purpose * (State / Region), Trips = sum(Trips)) |>
  model(ets = ETS(Trips)) |>
  reconcile(ets_adjusted = min_trace(ets)) |>
  forecast(h = 2)
```

```
# A fable: 1,700 x 7 [1Q]
# Key:     Purpose, State, Region, .model [850]
   Purpose  State           Region          .model       Quarter
   <chr*>   <chr*>          <chr*>          <chr>          <qtr>
 1 Business ACT             Canberra        ets          2018 Q1
 2 Business ACT             Canberra        ets          2018 Q2
 3 Business ACT             Canberra        ets_adjusted 2018 Q1
 4 Business ACT             Canberra        ets_adjusted 2018 Q2
 5 Business ACT             <aggregated>    ets          2018 Q1
 6 Business ACT             <aggregated>    ets          2018 Q2
 7 Business ACT             <aggregated>    ets_adjusted 2018 Q1
 8 Business ACT             <aggregated>    ets_adjusted 2018 Q2
 9 Business New South Wales Blue Mountains ets          2018 Q1
```

Hierarchical and grouped time series

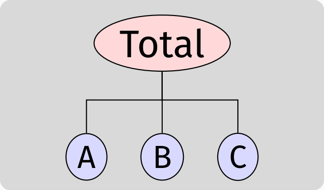
Every collection of time series with aggregation constraints can be written as

$$\boldsymbol{y}_t = \boldsymbol{S}\boldsymbol{b}_t$$

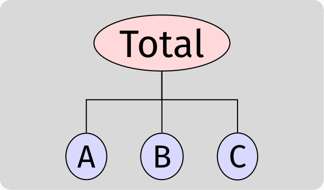
where

- $\boldsymbol{y}_t$ is a vector of all series at time t
- $\boldsymbol{b}_t$ is a vector of the most disaggregated series at time t
- $\boldsymbol{S}$ is a "summing matrix'' containing the aggregation constraints.

Hierarchical time series



Hierarchical time series

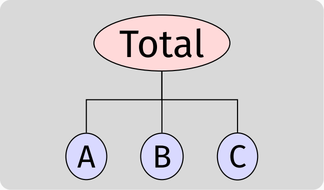


y_t : observed aggregate of all series at time t.

$y_{X,t}$: observation on series X at time t.

$\boldsymbol{b}_t$: vector of all series at bottom level in time t.

Hierarchical time series



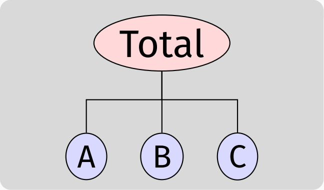
y_t : observed aggregate of all series at time t.

$y_{X,t}$: observation on series X at time t.

$\boldsymbol{b}_t$: vector of all series at bottom level in time t.

$$\boldsymbol{y}_t = \begin{pmatrix} y_t \\ y_{A,t} \\ y_{B,t} \\ y_{C,t} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_{A,t} \\ y_{B,t} \\ y_{C,t} \end{pmatrix}$$

Hierarchical time series



y_t : observed aggregate of all series at time t.

$y_{X,t}$: observation on series X at time t.

$\boldsymbol{b}_t$: vector of all series at bottom level in time t.

$$\boldsymbol{y}_t = \begin{pmatrix} y_t \\ y_{A,t} \\ y_{B,t} \\ y_{C,t} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\boldsymbol{S}} \underbrace{\begin{pmatrix} y_{A,t} \\ y_{B,t} \\ y_{C,t} \end{pmatrix}}_{\boldsymbol{b}_t}$$

$$\boxed{\boldsymbol{y}_t = \boldsymbol{S}\boldsymbol{b}_t}$$

Forecasting notation

Let $\hat{\boldsymbol{y}}_n(h)$ be vector of initial h-step forecasts, made at time n, stacked in same order as $\boldsymbol{y}_t$.

Forecasting notation

Let $\hat{\boldsymbol{y}}_n(h)$ be vector of initial h-step forecasts, made at time n, stacked in same order as $\boldsymbol{y}_t$.
(In general, they will not "add up".)

Forecasting notation

Let $\hat{\boldsymbol{y}}_n(h)$ be vector of initial h-step forecasts, made at time n, stacked in same order as $\boldsymbol{y}_t$.
(In general, they will not "add up".)

Reconciled forecasts must be of the form:

$$\tilde{\boldsymbol{y}}_n(h) = \boldsymbol{S}\boldsymbol{G}\hat{\boldsymbol{y}}_n(h)$$

for some matrix $\boldsymbol{G}$.

Forecasting notation

Let $\hat{\boldsymbol{y}}_n(h)$ be vector of initial h-step forecasts, made at time n, stacked in same order as $\boldsymbol{y}_t$.
(In general, they will not "add up".)

Reconciled forecasts must be of the form:

$$\tilde{\boldsymbol{y}}_n(h) = \boldsymbol{S}\boldsymbol{G}\hat{\boldsymbol{y}}_n(h)$$

for some matrix $\boldsymbol{G}$.

- $\boldsymbol{G}$ extracts and combines base forecasts $\hat{\boldsymbol{y}}_n(h)$ to get bottom-level forecasts.
- $\boldsymbol{S}$ adds them up

Outline

Optimal combination forecasts

Main result

The best (minimum sum of variances) unbiased forecasts are obtained when $\boldsymbol{G} = (\boldsymbol{S}'\Sigma_h^{-1}\boldsymbol{S})^{-1}\boldsymbol{S}'\Sigma_h^{-1}$, where Σ_h is the h-step base forecast error covariance matrix.

Optimal combination forecasts

Main result

The best (minimum sum of variances) unbiased forecasts are obtained when $\boldsymbol{G} = (\boldsymbol{S}'\Sigma_h^{-1}\boldsymbol{S})^{-1}\boldsymbol{S}'\Sigma_h^{-1}$, where Σ_h is the h-step base forecast error covariance matrix.

$$\tilde{\boldsymbol{y}}_n(h) = \boldsymbol{S}(\boldsymbol{S}'\Sigma_h^{-1}\boldsymbol{S})^{-1}\boldsymbol{S}'\Sigma_h^{-1}\hat{\boldsymbol{y}}_n(h)$$

Problem: Σ_h hard to estimate, especially for $h > 1$.

Solutions:

- Ignore Σ_h (OLS) [`min_trace(method='ols')`]
- Assume $\Sigma_h = k_h\Sigma_1$ is diagonal (WLS) [`min_trace(method='wls')`]
- Assume $\Sigma_h = k_h\Sigma_1$ and estimate it (GLS) [`min_trace(method='shrink')` (the default)]

Features

- Covariates can be included in initial forecasts.
- Adjustments can be made to initial forecasts at any level.
- Very simple and flexible method. Can work with *any* hierarchical or grouped time series.
- Conceptually easy to implement: regression of base forecasts on structure matrix.

Example: Australian tourism

```r
tourism_agg <- tourism |>
  aggregate_key(Purpose * (State / Region),
    Trips = sum(Trips)
  )
fc <- tourism_agg |>
  filter_index(. ~ "2015 Q4") |>
  model(ets = ETS(Trips)) |>
  reconcile(ets_adjusted = min_trace(ets)) |>
  forecast(h = "2 years")
```

Example: Australian tourism

```r
fc |>
  filter(is_aggregated(Purpose) & is_aggregated(State)) |>
  autoplot(tourism_agg, level = 95)
```