MONASH University

MONASH BUSINESS SCHOOL

High dimensional time series analysis

tsibble feasts Fable

4. Automatic forecasting algorithms

robjhyndman.com/hdtsa

Outline

Outline

Historical perspective

- Developed in the 1950s and 1960s as methods (algorithms) to produce point forecasts.
- Combine a “level”, “trend” (slope) and “seasonal” component to describe a time series.
- The rate of change of the components are controlled by “smoothing parameters”: α, β and γ respectively.
- Need to choose best values for the smoothing parameters (and initial states).
- Equivalent ETS state space models developed in the 1990s and 2000s.

A model for levels, trends, and seasonalities

We want a model that captures the level (ℓ_t), trend (b_t) and seasonality (s_t).

How do we combine these elements?

A model for levels, trends, and seasonalities

We want a model that captures the level (ℓ_t), trend (b_t) and seasonality (s_t).

How do we combine these elements?

Additively?

$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$

A model for levels, trends, and seasonalities

We want a model that captures the level (ℓ_t), trend (b_t) and seasonality (s_t).

How do we combine these elements?

Additively?

$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$

Multiplicatively?

$y_t = \ell_{t-1} b_{t-1} s_{t-m} (1 + \varepsilon_t)$

A model for levels, trends, and seasonalities

We want a model that captures the level (ℓ_t), trend (b_t) and seasonality (s_t).

How do we combine these elements?

Additively?

$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$

Multiplicatively?

$y_t = \ell_{t-1} b_{t-1} s_{t-m}(1 + \varepsilon_t)$

Perhaps a mix of both?

$y_t = (\ell_{t-1} + b_{t-1}) s_{t-m} + \varepsilon_t$

A model for levels, trends, and seasonalities

We want a model that captures the level (ℓ_t), trend (b_t) and seasonality (s_t).

How do we combine these elements?

Additively?

$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$

Multiplicatively?

$y_t = \ell_{t-1} b_{t-1} s_{t-m}(1 + \varepsilon_t)$

Perhaps a mix of both?

$y_t = (\ell_{t-1} + b_{t-1}) s_{t-m} + \varepsilon_t$

How do the level, trend and seasonal components evolve over time?

ETS models

General notation E T S : **E**xponen**T**ial **S**moothing

$\nearrow$ $\uparrow$ $\nwarrow$

Error **T**rend **S**eason

Error: Additive (`"A"`) or multiplicative (`"M"`)

ETS models

General notation

$$\begin{array}{ccc} & \text{E T S} : & \textbf{E}\text{xponen}\textbf{T}\text{ial }\textbf{S}\text{moothing} \\ \nearrow & \uparrow \quad \nwarrow & \\ \textbf{E}\text{rror} & \textbf{T}\text{rend} \quad \textbf{S}\text{eason} & \end{array}$$

Error: Additive (`"A"`) or multiplicative (`"M"`)

Trend: None (`"N"`), additive (`"A"`), multiplicative (`"M"`), or damped (`"Ad"` or `"Md"`).

ETS models

General notation $\quad$ E T S : **E**xponen**T**ial **S**moothing

$\nearrow \quad \uparrow \quad \nwarrow$

Error **T**rend **S**eason

Error: Additive (`"A"`) or multiplicative (`"M"`)

Trend: None (`"N"`), additive (`"A"`), multiplicative (`"M"`), or damped (`"Ad"` or `"Md"`).

Seasonality: None (`"N"`), additive (`"A"`) or multiplicative (`"M"`)

ETS(A,N,N): SES with additive errors

Measurement equation	$y_t = \ell_{t-1} + \varepsilon_t$
State equation	$\ell_t = \ell_{t-1} + \alpha\varepsilon_t$

where $\varepsilon_t \sim \text{NID}(0, \sigma^2)$.

- "innovations" or "single source of error" because equations have the same error process, ε_t.
- Measurement equation: relationship between observations and states.
- Transition/state equation(s): evolution of the state(s) through time.

ETS(M,N,N): SES with multiplicative errors.

Measurement equation	$y_t = \ell_{t-1}(1 + \varepsilon_t)$
State equation	$\ell_t = \ell_{t-1}(1 + \alpha\varepsilon_t)$

ETS(M,N,N): SES with multiplicative errors.

$$\begin{aligned} \text{Measurement equation} \qquad y_t &= \ell_{t-1}(1 + \varepsilon_t) \\ \text{State equation} \qquad \ell_t &= \ell_{t-1}(1 + \alpha\varepsilon_t) \end{aligned}$$

- Relative errors $\varepsilon_t = \frac{y_t - \hat{y}_{t|t-1}}{\hat{y}_{t|t-1}} \sim \text{NID}(0, \sigma^2)$
- Models with additive and multiplicative errors with the same parameters generate the same point forecasts but different prediction intervals.

ETS(A,A,N): Holt's linear trend

Additive errors

Measurement equation	$y_t = \ell_{t-1} + b_{t-1} + \varepsilon_t$
State equations	$\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$
	$b_t = b_{t-1} + \beta\varepsilon_t$

ETS(A,A,N): Holt's linear trend

Additive errors

Measurement equation $\quad y_t = \ell_{t-1} + b_{t-1} + \varepsilon_t$

State equations $\quad \ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$

$b_t = b_{t-1} + \beta\varepsilon_t$

Multiplicative errors

Measurement equation $\quad y_t = (\ell_{t-1} + b_{t-1})(1 + \varepsilon_t)$

State equations $\quad \ell_t = (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_t)$

$b_t = b_{t-1} + \beta\varepsilon_t$

Example: Australian population

```r
aus_economy <- global_economy %>% filter(Code == "AUS") %>%
  mutate(Pop = Population/1e6)
fit <- aus_economy %>% model(AAN = ETS(Pop))
report(fit)
```

```
## Series: Pop
## Model: ETS(A,A,N)
##   Smoothing parameters:
##     alpha = 1
##     beta  = 0.327
##
##   Initial states:
##     l      b
##  10.1 0.222
##
##
##   sigma^2:  0.0041
##
##   AIC  AICc   BIC
```

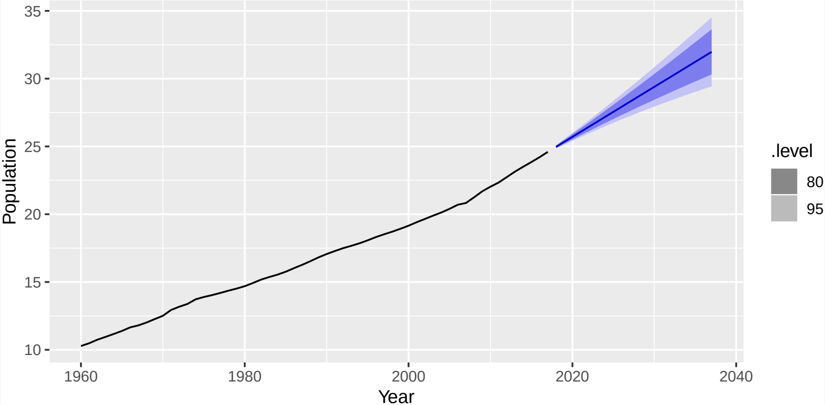
Example: Australian population

```r
components(fit) %>%
  left_join(fitted(fit), by = c("Country", ".model", "Year"))
```

```
## # A tsibble: 59 x 8 [1Y]
## # Key:       Country, .model [1]
##    Country  .model  Year   Pop level slope remainder .fitted
##    <fct>    <chr>  <dbl> <dbl> <dbl> <dbl>     <dbl>   <dbl>
##  1 Austral~ AAN     1959  NA    10.1 0.222 NA          NA
##  2 Austral~ AAN     1960  10.3  10.3 0.222 -0.000145   10.3
##  3 Austral~ AAN     1961  10.5  10.5 0.217 -0.0159     10.5
##  4 Austral~ AAN     1962  10.7  10.7 0.231  0.0418     10.7
##  5 Austral~ AAN     1963  11.0  11.0 0.223 -0.0229     11.0
##  6 Austral~ AAN     1964  11.2  11.2 0.221 -0.00641    11.2
##  7 Austral~ AAN     1965  11.4  11.4 0.221 -0.000314   11.4
##  8 Austral~ AAN     1966  11.7  11.7 0.235  0.0418     11.6
##  9 Austral~ AAN     1967  11.8  11.8 0.206 -0.0869     11.9
## 10 Austral~ AAN     1968  12.0  12.0 0.208  0.00350    12.0
## # ... with 49 more rows
```

Example: Australian population

```r
fit %>%
  forecast(h = 20) %>%
  autoplot(aus_economy) +
  ylab("Population") + xlab("Year")
```



ETS(A,Ad,N): Damped trend method

Additive errors

Measurement equation	$y_t = (\ell_{t-1} + \phi b_{t-1}) + \varepsilon_t$
State equations	$\ell_t = (\ell_{t-1} + \phi b_{t-1}) + \alpha\varepsilon_t$
	$b_t = \phi b_{t-1} + \beta\varepsilon_t$

ETS(A,Ad,N): Damped trend method

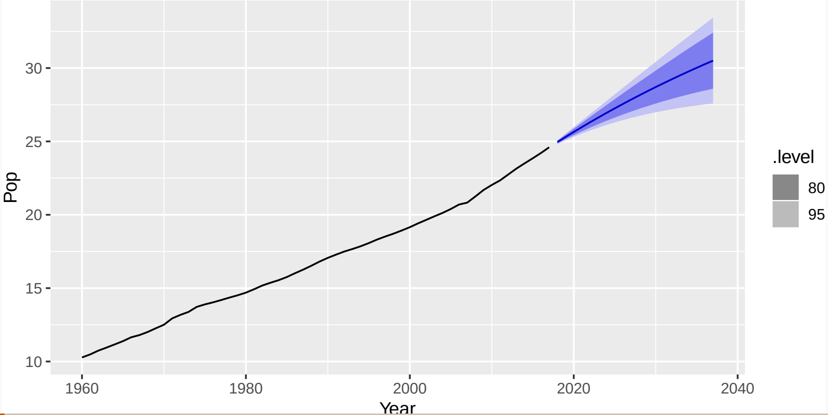
Additive errors

$$\begin{aligned} \text{Measurement equation} \quad y_t &= (\ell_{t-1} + \phi b_{t-1}) + \varepsilon_t \\ \text{State equations} \quad \ell_t &= (\ell_{t-1} + \phi b_{t-1}) + \alpha\varepsilon_t \\ b_t &= \phi b_{t-1} + \beta\varepsilon_t \end{aligned}$$

- Damping parameter $0 < \phi < 1$.
- If $\phi = 1$, identical to Holt's linear trend.
- As $h \rightarrow \infty$, $\hat{y}_{T+h|T} \rightarrow \ell_T + \phi b_T/(1 - \phi)$.
- Short-run forecasts trended, long-run forecasts constant.

Example: Australian population

```r
aus_economy %>%
  model(holt = ETS(Pop ~ trend("Ad"))) %>%
  forecast(h = 20) %>%
  autoplot(aus_economy)
```



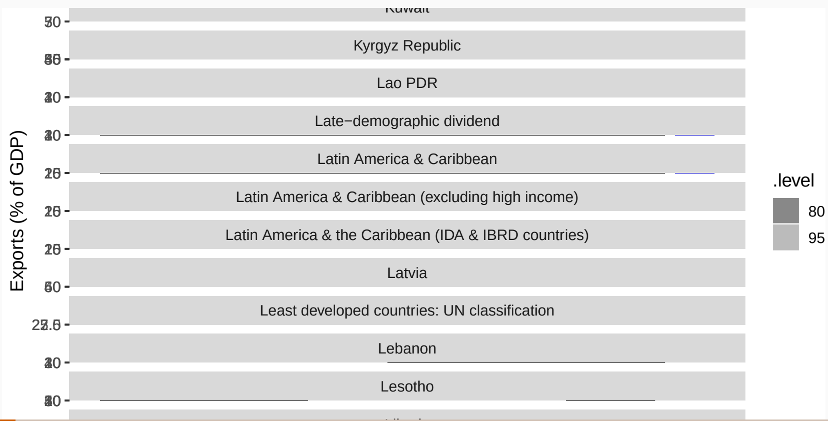
Example: National Exports

```
fit <- global_economy %>%
  model(ets = ETS(Exports))
fit
```

```
## # A mable: 263 x 2
## # Key:     Country [263]
##    Country              ets
##    <fct>                <model>
##  1 Afghanistan          <NULL model>
##  2 Albania              <NULL model>
##  3 Algeria              <ETS(M,N,N)>
##  4 American Samoa       <NULL model>
##  5 Andorra              <NULL model>
##  6 Angola               <NULL model>
##  7 Antigua and Barbuda <NULL model>
##  8 Arab World           <NULL model>
##  9 Argentina            <ETS(M,N,N)>
## 10 Armenia              <NULL model>
## # ... with 253 more rows
```

Example: National Exports

```r
fit %>%
  forecast(h = 5) %>%
  autoplot(global_economy) +
  ylab("Exports (% of GDP)") + xlab("Year")
```



ETS(A,A,A): Holt-Winters additive method

Forecast equation	$\hat{y}_{t+h\|t} = \ell_t + hb_t + s_{t+h-m(k+1)}$
Observation equation	$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$
State equations	$\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$
	$b_t = b_{t-1} + \beta\varepsilon_t$
	$s_t = s_{t-m} + \gamma\varepsilon_t$

- k = integer part of $(h-1)/m$.
- $\sum_i s_i \approx 0$.
- 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).

ETS(M,A,M): Holt-Winters multiplicative method

$$
\begin{aligned}
\text{Forecast equation} \quad \hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t+h-m(k+1)} \\
\text{Observation equation} \quad y_t &= (\ell_{t-1} + b_{t-1})s_{t-m}(1 + \varepsilon_t) \\
\text{State equations} \quad \ell_t &= (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_t) \\
b_t &= b_{t-1}(1 + \beta\varepsilon_t) \\
s_t &= s_{t-m}(1 + \gamma\varepsilon_t)
\end{aligned}
$$

- k is integer part of $(h-1)/m$.
- $\sum_i s_i \approx m$.
- 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).

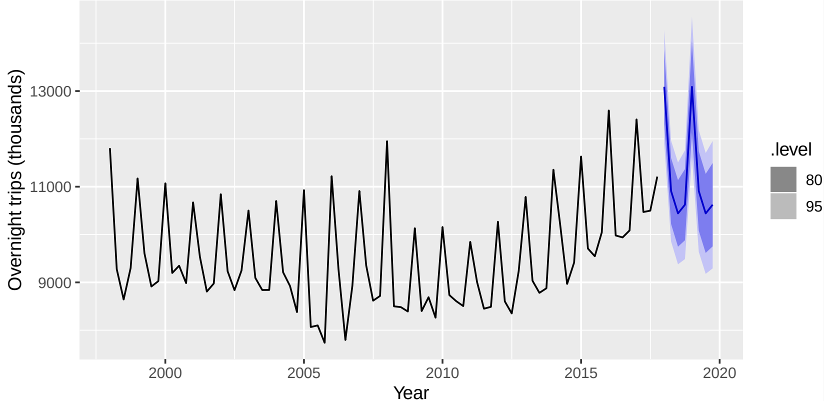
Example: Australian holiday tourism

```r
holidays <- tourism %>%
  filter(Purpose == "Holiday")
fit <- holidays %>% model(ets = ETS(Trips))
fit
```

```
## # A mable: 76 x 4
## # Key:     Region, State, Purpose [76]
##    Region                 State          Purpose ets
##    <chr>                  <chr>          <chr>   <model>
##  1 Adelaide               South Australia Holiday <ETS(A,N,~
##  2 Adelaide Hills         South Australia Holiday <ETS(A,A,~
##  3 Alice Springs          Northern Terri~ Holiday <ETS(M,N,~
##  4 Australia's Coral Coa~ Western Austra~ Holiday <ETS(M,N,~
##  5 Australia's Golden Ou~ Western Austra~ Holiday <ETS(M,N,~
##  6 Australia's North West Western Austra~ Holiday <ETS(A,N,~
##  7 Australia's South West Western Austra~ Holiday <ETS(M,N,~
##  8 Ballarat               Victoria        Holiday <ETS(M,N,~
##  9 Barkly                 Northern Terri~ Holiday <ETS(A,N,~
## 10 Barossa                South Australia Holiday <ETS(A,N,~
```

Example: Australian holiday tourism

```
fit %>% forecast() %>%
  autoplot(holidays) +
    xlab("Year") + ylab("Overnight trips (thousands)")
```



Your turn

Find an ETS model for the Gas data from `aus_production`.

1. Why is multiplicative seasonality necessary here?
2. Experiment with making the trend damped.
3. Check that the residuals from the best method look like white noise.

ETS models

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

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

Additive error models

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

Multiplicative error models

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

Estimating ETS models

- Smoothing parameters α, β, γ and ϕ, and the initial states ℓ_0, b_0, $s_0, s_{-1}, \dots, s_{-m+1}$ are estimated by maximising the "likelihood" = the probability of the data arising from the specified model.
- For models with additive errors equivalent to minimising SSE.
- For models with multiplicative errors, **not** equivalent to minimising SSE.

Innovations state space models

Let $\boldsymbol{x}_t = (\ell_t, b_t, s_t, s_{t-1}, \dots, s_{t-m+1})$ and $\varepsilon_t \overset{\text{iid}}{\sim} \text{N}(0, \sigma^2)$.

$$
\begin{aligned}
y_t &= \underbrace{h(\boldsymbol{x}_{t-1})}_{\mu_t} + \underbrace{k(\boldsymbol{x}_{t-1})\varepsilon_t}_{e_t} \\
\boldsymbol{x}_t &= f(\boldsymbol{x}_{t-1}) + g(\boldsymbol{x}_{t-1})\varepsilon_t
\end{aligned}
$$

Additive errors

$$k(x) = 1. \qquad y_t = \mu_t + \varepsilon_t.$$

Multiplicative errors

$$k(\boldsymbol{x}_{t-1}) = \mu_t. \qquad y_t = \mu_t(1 + \varepsilon_t).$$

$\varepsilon_t = (y_t - \mu_t)/\mu_t$ is relative error.

Innovations state space models

Estimation

$$\begin{aligned} L^*(\boldsymbol{\theta}, \boldsymbol{x}_0) &= n \log\left(\sum_{t=1}^{n} \varepsilon_t^2 / k^2(\boldsymbol{x}_{t-1}) \right) + 2 \sum_{t=1}^{n} \log |k(\boldsymbol{x}_{t-1})| \\ &= -2 \log(\text{Likelihood}) + \text{constant} \end{aligned}$$

- Estimate parameters $\boldsymbol{\theta} = (\alpha, \beta, \gamma, \phi)$ and initial states $\boldsymbol{x}_0 = (\ell_0, b_0, s_0, s_{-1}, \dots, s_{-m+1})$ by minimizing L^*.

Model selection

Akaike's Information Criterion

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

where L is the likelihood and k is the number of parameters initial states estimated in the model.

Model selection

Akaike's Information Criterion

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

where L is the likelihood and k is the number of parameters initial states estimated in the model.

Corrected AIC

$$\text{AIC}_\text{c} = \text{AIC} + \frac{2(k+1)(k+2)}{T-k}$$

which is the AIC corrected (for small sample bias).

Model selection

Akaike's Information Criterion

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

where L is the likelihood and k is the number of parameters initial states estimated in the model.

Corrected AIC

$$\text{AIC}_\text{c} = \text{AIC} + \frac{2(k+1)(k+2)}{T-k}$$

which is the AIC corrected (for small sample bias).

Bayesian Information Criterion

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

AIC and cross-validation

Minimizing the AIC assuming Gaussian residuals is asymptotically equivalent to minimizing one-step time series cross validation MSE.

Automatic forecasting

From Hyndman et al. (IJF, 2002):

- Apply each model that is appropriate to the data. Optimize parameters and initial values using MLE (or some other criterion).
- Select best method using AICc:
- Produce forecasts using best method.
- Obtain forecast intervals using underlying state space model.

Method performed very well in M3 competition.

Some unstable models

- Some of the combinations of (Error, Trend, Seasonal) can lead to numerical difficulties; see equations with division by a state.
- These are: ETS(A,N,M), ETS(A,A,M), ETS(A,A_d,M).
- Models with multiplicative errors are useful for strictly positive data, but are not numerically stable with data containing zeros or negative values. In that case only the six fully additive models will be applied.

Exponential smoothing models

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

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

Example: Australian holiday tourism

```r
fit <- holidays %>% model(ETS(Trips))
report(fit)
```

```
## Series: Trips
## Model: ETS(M,N,M)
##   Smoothing parameters:
##     alpha = 0.358
##     gamma = 0.000969
##
##   Initial states:
##     l     s1    s2    s3   s4
##  9667 0.943 0.927 0.968 1.16
##
##
##   sigma^2:  0.0022
##
##  AIC AICc  BIC
## 1331 1333 1348
```

Example: Australian holiday tourism

Model selected: ETS(M,N,M)

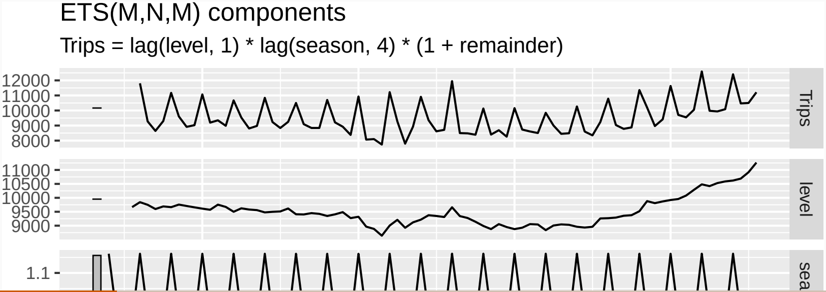
$$\begin{aligned} y_t &= \ell_{t-1}s_{t-m}(1+\varepsilon_t) \\ \ell_t &= \ell_{t-1}(1+\alpha\varepsilon_t) \\ s_t &= s_{t-m}(1+\gamma\varepsilon_t). \end{aligned}$$

$\hat{\alpha} = 0.3578$, and $\hat{\gamma} = 0.000969$.

Example: Australian holiday tourism

```r
components(fit)
```

```r
components(fit) %>%
  autoplot() +
  ggtitle("ETS(M,N,M) components")
```



Residuals

Response residuals

$$\hat{e}_t = y_t - \hat{y}_{t|t-1}$$

Innovation residuals

Additive error model:

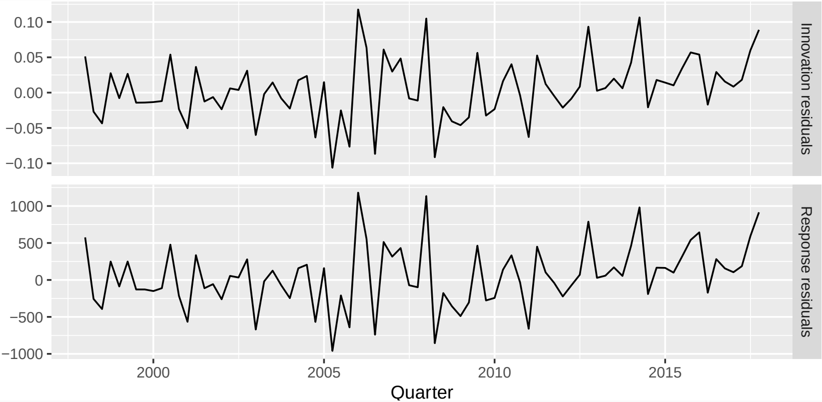
$$\hat{\varepsilon}_t = y_t - \hat{y}_{t|t-1}$$

Multiplicative error model:

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

Example: Australian holiday tourism

```r
residuals(fit)
residuals(fit, type = "response")
```



Forecasting with ETS models

Point forecasts: iterate the equations for $t = T + 1, T + 2, \dots, T + h$ and set all $\varepsilon_t = 0$ for $t > T$.

Forecasting with ETS models

Point forecasts: iterate the equations for $t = T+1, T+2, \ldots, T+h$ and set all $\varepsilon_t = 0$ for $t > T$.

- Not the same as $\mathrm{E}(y_{t+h}|\boldsymbol{x}_t)$ unless trend and seasonality are both additive.
- Point forecasts for ETS(A,*,*) are identical to ETS(M,*,*) if the parameters are the same.

Example: ETS(A,A,N)

$$
\begin{aligned}
y_{T+1} &= \ell_T + b_T + \varepsilon_{T+1} \\
\hat{y}_{T+1|T} &= \ell_T + b_T \\
y_{T+2} &= \ell_{T+1} + b_{T+1} + \varepsilon_{T+2} \\
&= (\ell_T + b_T + \alpha\varepsilon_{T+1}) + (b_T + \beta\varepsilon_{T+1}) + \varepsilon_{T+2} \\
\hat{y}_{T+2|T} &= \ell_T + 2b_T
\end{aligned}
$$

etc.

Example: ETS(M,A,N)

$$
\begin{aligned}
y_{T+1} &= (\ell_T + b_T)(1 + \varepsilon_{T+1}) \\
\hat{y}_{T+1|T} &= \ell_T + b_T. \\
y_{T+2} &= (\ell_{T+1} + b_{T+1})(1 + \varepsilon_{T+2}) \\
&= \left\{(\ell_T + b_T)(1 + \alpha\varepsilon_{T+1}) + [b_T + \beta(\ell_T + b_T)\varepsilon_{T+1}]\right\}(1 + \varepsilon_{T+2}) \\
\hat{y}_{T+2|T} &= \ell_T + 2b_T
\end{aligned}
$$

etc.

Forecasting with ETS models

Prediction intervals: can only generated using the models.

- The prediction intervals will differ between models with additive and multiplicative errors.
- Exact formulae for some models.
- More general to simulate future sample paths, conditional on the last estimate of the states, and to obtain prediction intervals from the percentiles of these simulated future paths.

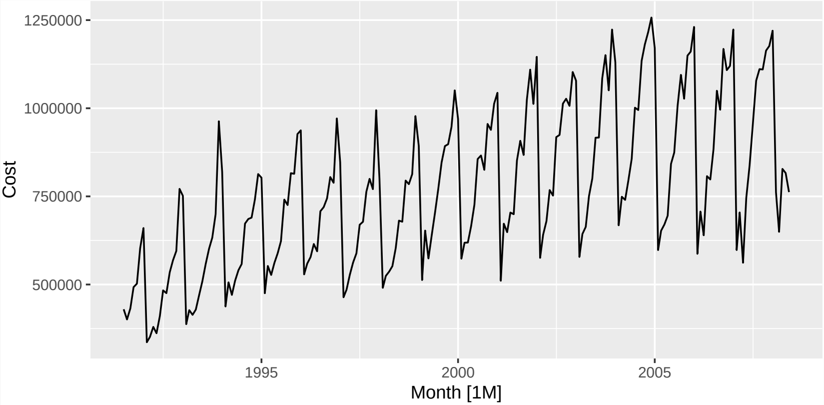
Prediction intervals

PI for most ETS models: $\hat{y}_{T+h|T} \pm c\sigma_h$, where c depends on coverage probability and σ_h is forecast standard deviation.

(A,N,N)	$\sigma_h = \sigma^2\left[1 + \alpha^2(h-1)\right]$
(A,A,N)	$\sigma_h = \sigma^2\left[1 + (h-1)\{\alpha^2 + \alpha\beta h + \frac{1}{6}\beta^2 h(2h-1)\}\right]$
(A,A$_d$,N)	$\sigma_h = \sigma^2\Big[1 + \alpha^2(h-1) + \frac{\beta\phi h}{(1-\phi)^2}\left\{2\alpha(1-\phi) + \beta\phi\right\} - \frac{\beta\phi(1-\phi^h)}{(1-\phi)^2(1-\phi^2)}\left\{2\alpha(1-\phi^2) + \beta\phi(1+2\phi-\phi^h)\right\}\Big]$
(A,N,A)	$\sigma_h = \sigma^2\left[1 + \alpha^2(h-1) + \gamma k(2\alpha+\gamma)\right]$
(A,A,A)	$\sigma_h = \sigma^2\left[1 + (h-1)\{\alpha^2 + \alpha\beta h + \frac{1}{6}\beta^2 h(2h-1)\} + \gamma k\{2\alpha + \gamma + \beta m(k+1)\}\right]$
(A,A$_d$,A)	$\sigma_h = \sigma^2\Big[1 + \alpha^2(h-1) + \frac{\beta\phi h}{(1-\phi)^2}\left\{2\alpha(1-\phi) + \beta\phi\right\} - \frac{\beta\phi(1-\phi^h)}{(1-\phi)^2(1-\phi^2)}\left\{2\alpha(1-\phi^2) + \beta\phi(1+2\phi-\phi^h)\right\} + \gamma k(2\alpha+\gamma) + \frac{2\beta\gamma\phi}{(1-\phi)(1-\phi^m)}\left\{k(1-\phi^m) - \phi^m(1-\phi^{mk})\right\}\Big]$

Example: Corticosteroid drug sales

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



Example: Corticosteroid drug sales

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

```
## Series: Cost
## Model: ETS(M,Ad,M)
##   Smoothing parameters:
##     alpha = 0.307
##     beta  = 0.000101
##     gamma = 0.000101
##     phi   = 0.978
##
##   Initial states:
##        l    b    s1    s2    s3    s4    s5   s6   s7   s8
##  417269 8206 0.872 0.826 0.756 0.773 0.687 1.28 1.32 1.18
##    s9 s10  s11   s12
##  1.16 1.1 1.05 0.981
##
##
##   sigma^2:  0.0046
##
##  AIC AICc  BIC
## 5515 5519 5575
```

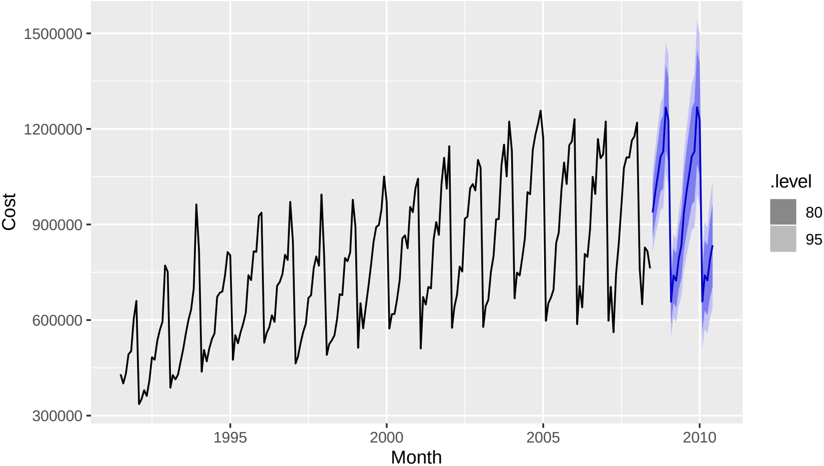
Example: Corticosteroid drug sales

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

```
## Series: Cost
## Model: ETS(A,A,A)
##   Smoothing parameters:
##     alpha = 0.17
##     beta  = 0.00631
##     gamma = 0.455
##
##   Initial states:
##       l    b     s1      s2      s3      s4      s5     s6
##  409706 9097 -99075 -136602 -191496 -174531 -241437 210644
##      s7      s8      s9    s10    s11    s12
##  244644 145368 130570 84458 39132 -11674
##
##
##   sigma^2:  3.5e+09
##
##  AIC AICc  BIC
## 5585 5589 5642
```

Example: Corticosteroid drug sales

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



Example: Corticosteroid drug sales

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

```r
h02 %>%
  model(
    auto = ETS(Cost),
    AAA = ETS(Cost ~ error("A") + trend("A") + season("A"))
  ) %>%
  accuracy() %>%
  transmute(Model = .model, ME, MAE, RMSE, MAPE, MASE) %>%
  knitr::kable(booktabs = TRUE)
```

Your turn

- Use `ETS()` on some of these series:

  `tourism`, *`gafa_stock`*, *`pelt`*

- Does it always give good forecasts?
- Find an example where it does not work well. Can you figure out why?

Outline

Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \ldots, y_{t+s})$ does not depend on t.

Stationarity

Definition

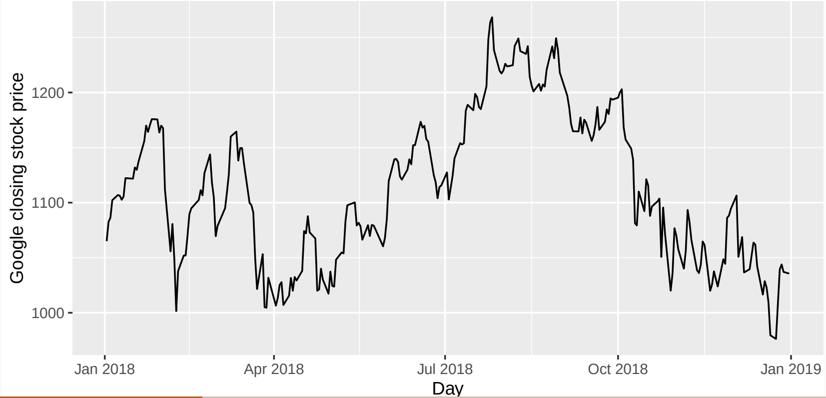
If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

A **stationary series** is:

- roughly horizontal
- constant variance
- no patterns predictable in the long-term

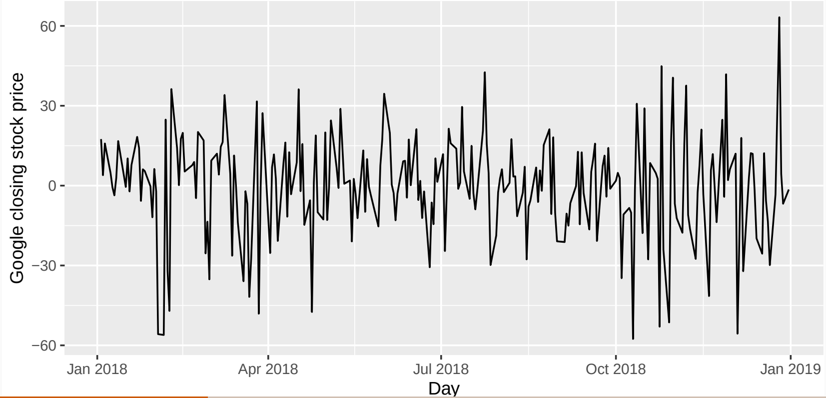
Stationary?

```
gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018)
  autoplot(Close) + ylab("Google closing stock
```



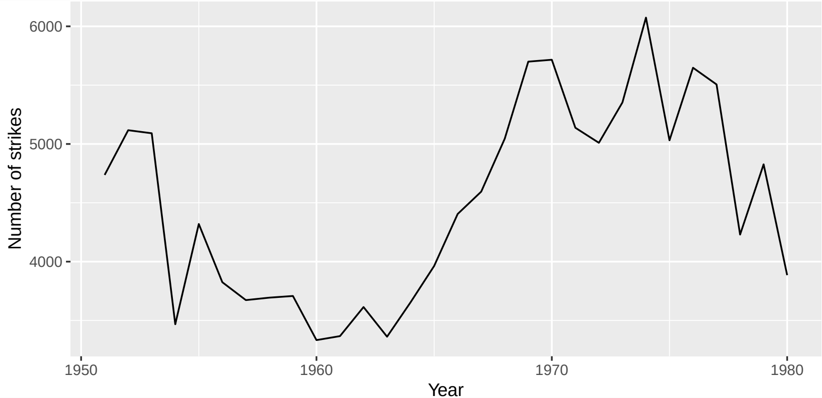
Stationary?

```
gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018)
  autoplot(difference(Close)) + ylab("Google c
```



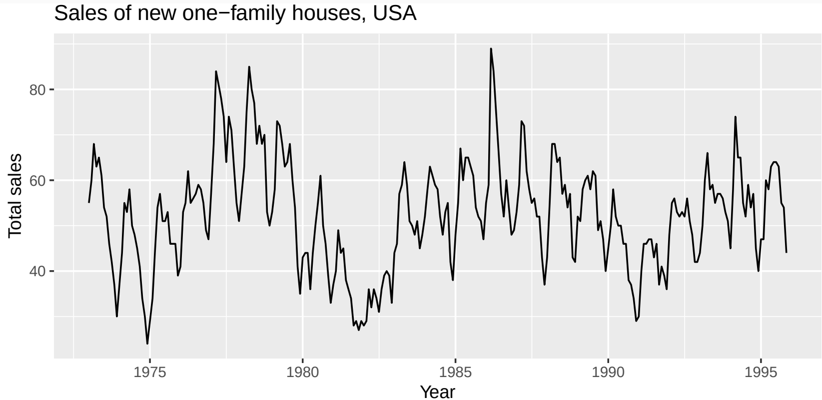
Stationary?

```
as_tsibble(fma::strikes) %>% autoplot(value) +
  ylab("Number of strikes") + xlab("Year")
```



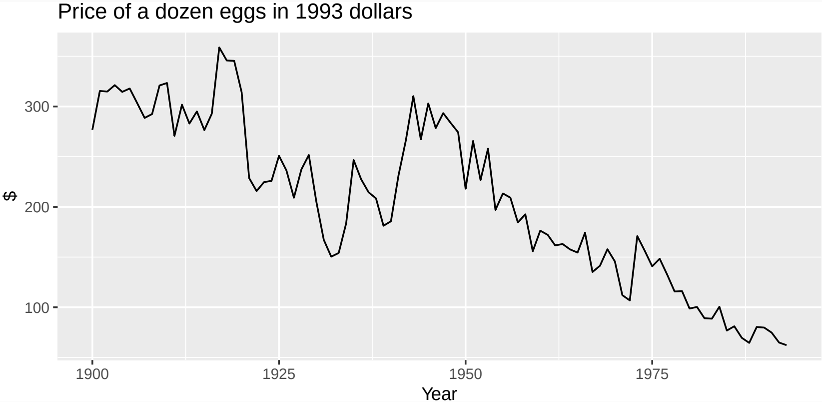
Stationary?

```
as_tsibble(fma::hsales) %>% autoplot(value) +
  ggtitle("Sales of new one-family houses, USA
```



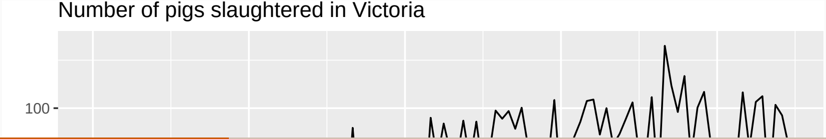
Stationary?

```
as_tsibble(fma::eggs) %>% autoplot(value) + xl
  ggtitle("Price of a dozen eggs in 1993 dolla
```



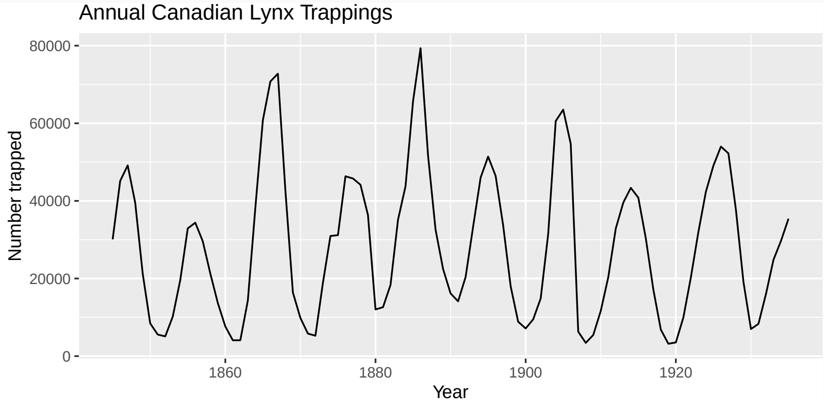
Stationary?

```
aus_livestock %>%
  filter(
    Animal == "Pigs",
    State == "Victoria",
    year(Month) >= 2010
  ) %>%
  autoplot(Count/1e3) + xlab("Year") + ylab("t
  ggtitle("Number of pigs slaughtered in Victo
```



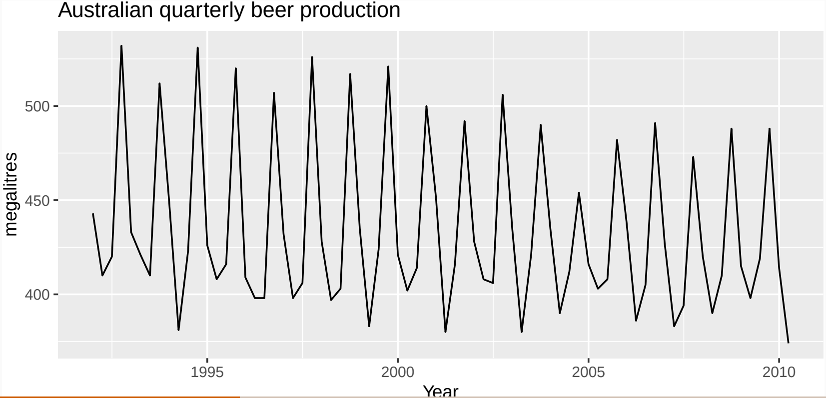
Stationary?

```
pelt %>% autoplot(Lynx) + xlab("Year") + ylab(
  ggtitle("Annual Canadian Lynx Trappings")
```



Stationary?

```
aus_production %>% filter_index("1992" ~ .) %>
  autoplot(Beer) + xlab("Year") + ylab("megali
  ggtitle("Australian quarterly beer productio
```



Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \ldots, y_{t+s})$ does not depend on t.

Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

Transformations help to **stabilize the variance**.

For ARIMA modelling, we also need to **stabilize the mean**.

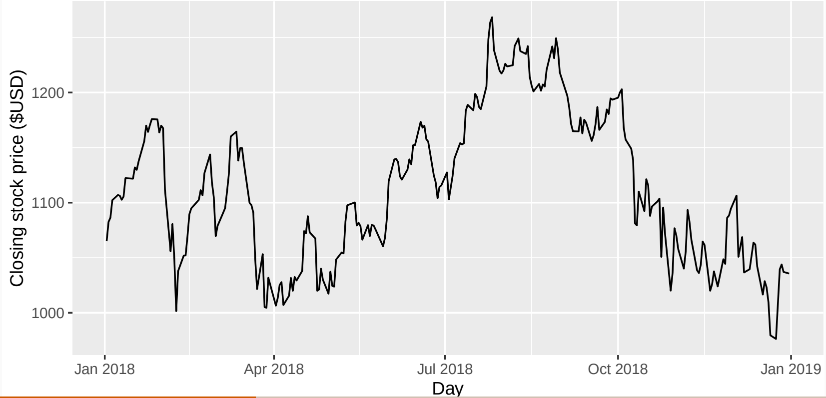
Non-stationarity in the mean

Identifying non-stationary series

- time plot.
- The ACF of stationary data drops to zero relatively quickly
- The ACF of non-stationary data decreases slowly.
- For non-stationary data, the value of r_1 is often large and positive.

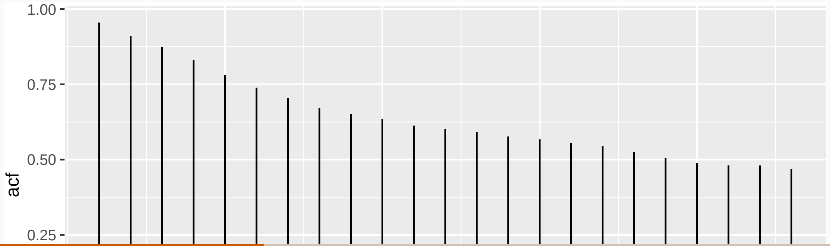
Example: Google stock price

```
gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018)
  autoplot(Close) + ylab("Closing stock price
```



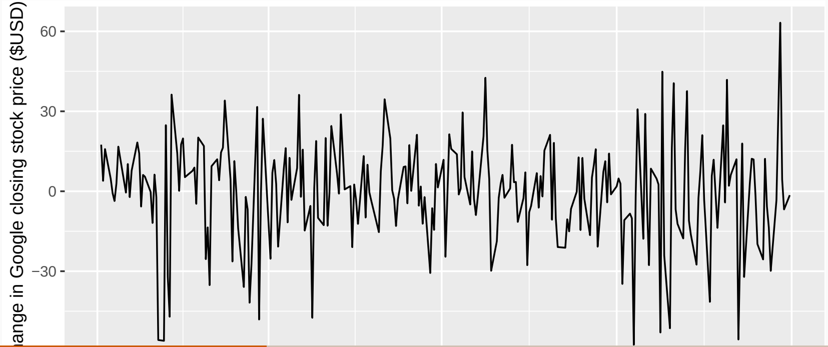
Example: Google stock price

```
google_2018 <- gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018)
  mutate(trading_day = row_number()) %>%
  update_tsibble(index = trading_day, regular
google_2018 %>%
  ACF(Close) %>% autoplot()
```



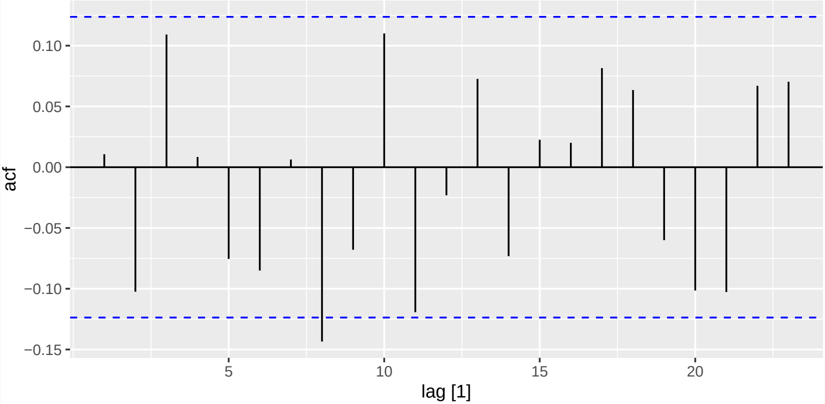
Example: Google stock price

```
gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018)
  autoplot(difference(Close)) +
  ylab("Change in Google closing stock price (
```



Example: Google stock price

```
google_2018 %>% ACF(difference(Close)) %>% aut
```



Differencing

- Differencing helps to **stabilize the mean**.
- The differenced series is the *change* between each observation in the original series: $y'_t = y_t - y_{t-1}$.
- The differenced series will have only $T - 1$ values since it is not possible to calculate a difference y'_1 for the first observation.

Second-order differencing

Occasionally the differenced data will not appear stationary and it may be necessary to difference the data a second time:

Second-order differencing

Occasionally the differenced data will not appear stationary and it may be necessary to difference the data a second time:

$$\begin{aligned} y''_t &= y'_t - y'_{t-1} \\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\ &= y_t - 2y_{t-1} + y_{t-2}. \end{aligned}$$

Second-order differencing

Occasionally the differenced data will not appear stationary and it may be necessary to difference the data a second time:

$$\begin{aligned} y''_t &= y'_t - y'_{t-1} \\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\ &= y_t - 2y_{t-1} + y_{t-2}. \end{aligned}$$

- y''_t will have $T - 2$ values.
- In practice, it is almost never necessary to go beyond second-order differences.

Seasonal differencing

A seasonal difference is the difference between an observation and the corresponding observation from the previous year.

Seasonal differencing

A seasonal difference is the difference between an observation and the corresponding observation from the previous year.

$$y'_t = y_t - y_{t-m}$$

where m = number of seasons.

Seasonal differencing

A seasonal difference is the difference between an observation and the corresponding observation from the previous year.

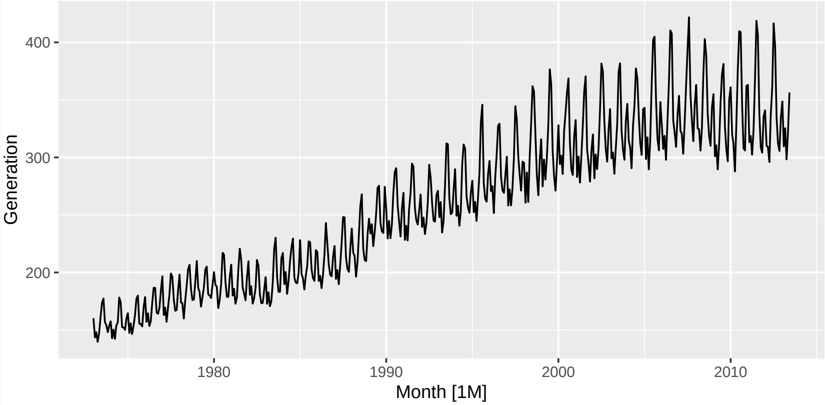
$$y'_t = y_t - y_{t-m}$$

where m = number of seasons.

- For monthly data $m = 12$.
- For quarterly data $m = 4$.

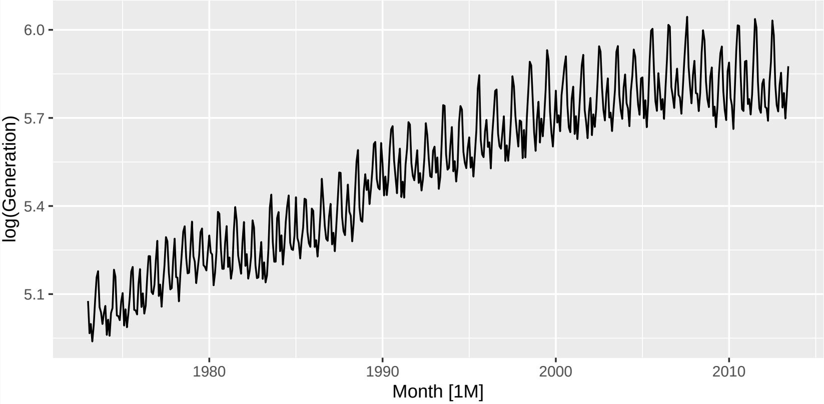
Electricity production

```
usmelec %>% autoplot(
  Generation
)
```



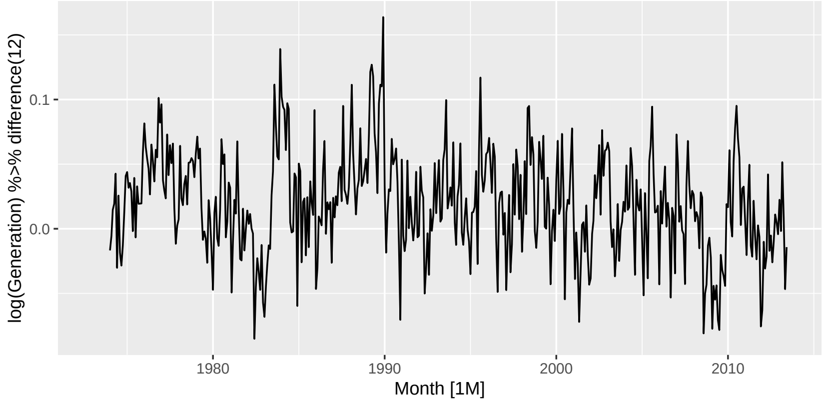
Electricity production

```
usmelec %>% autoplot(
  log(Generation)
)
```



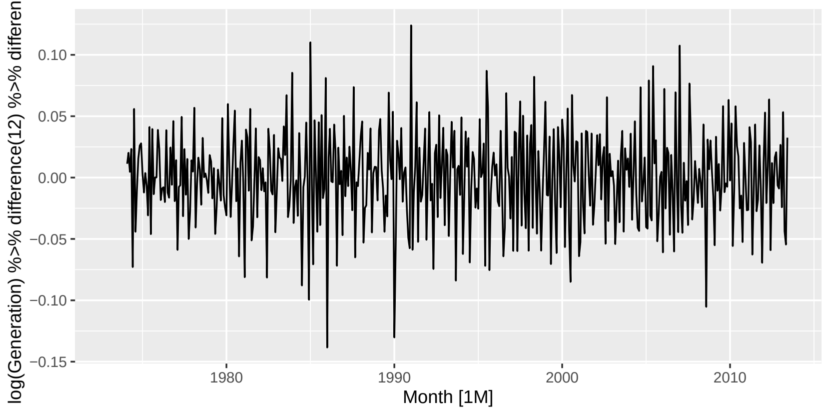
Electricity production

```r
usmelec %>% autoplot(
  log(Generation) %>% difference(12)
)
```



Electricity production

```
usmelec %>% autoplot(
  log(Generation) %>% difference(12) %>% difference()
)
```



Electricity production

- Seasonally differenced series is closer to being stationary.
- Remaining non-stationarity can be removed with further first difference.

If $y'_t = y_t - y_{t-12}$ denotes seasonally differenced series, then twice-differenced series is

$$\begin{aligned} y^*_t &= y'_t - y'_{t-1} \\ &= (y_t - y_{t-12}) - (y_{t-1} - y_{t-13}) \\ &= y_t - y_{t-1} - y_{t-12} + y_{t-13}\,. \end{aligned}$$

Seasonal differencing

When both seasonal and first differences are applied…

Seasonal differencing

When both seasonal and first differences are applied…

- it makes no difference which is done first—the result will be the same.
- If seasonality is strong, we recommend that seasonal differencing be done first because sometimes the resulting series will be stationary and there will be no need for further first difference.

Seasonal differencing

When both seasonal and first differences are applied…

- it makes no difference which is done first—the result will be the same.
- If seasonality is strong, we recommend that seasonal differencing be done first because sometimes the resulting series will be stationary and there will be no need for further first difference.

It is important that if differencing is used, the differences are interpretable.

Interpretation of differencing

- first differences are the change between **one observation and the next**;
- seasonal differences are the change between **one year to the next**.

Interpretation of differencing

- first differences are the change between **one observation and the next**;
- seasonal differences are the change between **one year to the next**.

But taking lag 3 differences for yearly data, for example, results in a model which cannot be sensibly interpreted.

Unit root tests

Statistical tests to determine the required order of differencing.

1. Augmented Dickey Fuller test: null hypothesis is that the data are non-stationary and non-seasonal.
2. Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test: null hypothesis is that the data are stationary and non-seasonal.
3. Other tests available for seasonal data.

KPSS test

```r
google_2018 %>%
  features(Close, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Symbol kpss_stat kpss_pvalue
##   <chr>      <dbl>       <dbl>
## 1 GOOG       0.573      0.0252
```

KPSS test

```r
google_2018 %>%
  features(Close, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Symbol kpss_stat kpss_pvalue
##   <chr>      <dbl>       <dbl>
## 1 GOOG       0.573      0.0252
```

```r
google_2018 %>%
  features(Close, unitroot_ndiffs)
```

```
## # A tibble: 1 x 2
##   Symbol ndiffs
##   <chr>   <int>
## 1 GOOG        1
```

Automatically selecting differences

STL decomposition: $y_t = T_t + S_t + R_t$

Seasonal strength $F_s = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(S_t+R_t)}\right)$

If $F_s > 0.64$, do one seasonal difference.

```
usmelec %>% mutate(log_gen = log(Generation)) %>%
  features(log_gen, list(unitroot_nsdiffs, feat_stl))
```

```
## # A tibble: 1 x 8
##   nsdiffs trend_strength seasonal_streng~ spikiness
##     <int>          <dbl>            <dbl>     <dbl>
## 1       1          0.994            0.941  1.54e-12
## # ... with 4 more variables: linearity <dbl>,
## #   curvature <dbl>, seasonal_peak_year <dbl>,
## #   seasonal_trough_year <dbl>
```

Automatically selecting differences

```r
usmelec %>% mutate(log_gen = log(Generation)) %>%
  features(log_gen, unitroot_nsdiffs)
```

```
## # A tibble: 1 x 1
##   nsdiffs
##     <int>
## 1       1
```

```r
usmelec %>% mutate(d_log_gen = difference(log(Generation), 12)) %>%
  features(d_log_gen, unitroot_ndiffs)
```

```
## # A tibble: 1 x 1
##   ndiffs
##    <int>
## 1      1
```

Your turn

For the `tourism` dataset, compute the total number of trips and find an appropriate differencing (after transformation if necessary) to obtain stationary data.

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

In other words, B, operating on y_t, has the effect of **shifting the data back one period**.

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

In other words, B, operating on y_t, has the effect of **shifting the data back one period**. Two applications of B to y_t **shifts the data back two periods**:

$$B(By_t) = B^2y_t = y_{t-2}$$

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

In other words, B, operating on y_t, has the effect of **shifting the data back one period**. Two applications of B to y_t **shifts the data back two periods**:

$$B(By_t) = B^2y_t = y_{t-2}$$

For monthly data, if we wish to shift attention to "the same month last year", then B^{12} is used, and the notation is $B^{12}y_t = y_{t-12}$.

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*.

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*. A first difference can be written as

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*. A first difference can be written as

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Note that a first difference is represented by $(1 - B)$.

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*. A first difference can be written as

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Note that a first difference is represented by $(1 - B)$.

Similarly, if second-order differences (i.e., first differences of first differences) have to be computed, then:

$$y''_t = y_t - 2y_{t-1} + y_{t-2} = (1 - B)^2 y_t$$

Backshift notation

- Second-order difference is denoted $(1 - B)^2$.
- *Second-order difference* is not the same as a *second difference*, which would be denoted $1 - B^2$;
- In general, a dth-order difference can be written as
$$(1 - B)^d y_t$$
- A seasonal difference followed by a first difference can be written as
$$(1 - B)(1 - B^m)y_t$$

Backshift notation

The "backshift" notation is convenient because the terms can be multiplied together to see the combined effect.

$$\begin{aligned}(1-B)(1-B^m)y_t &= (1-B-B^m+B^{m+1})y_t \\ &= y_t - y_{t-1} - y_{t-m} + y_{t-m-1}.\end{aligned}$$

Backshift notation

The "backshift" notation is convenient because the terms can be multiplied together to see the combined effect.

$$\begin{aligned}(1-B)(1-B^m)y_t &= (1-B-B^m+B^{m+1})y_t \\ &= y_t - y_{t-1} - y_{t-m} + y_{t-m-1}.\end{aligned}$$

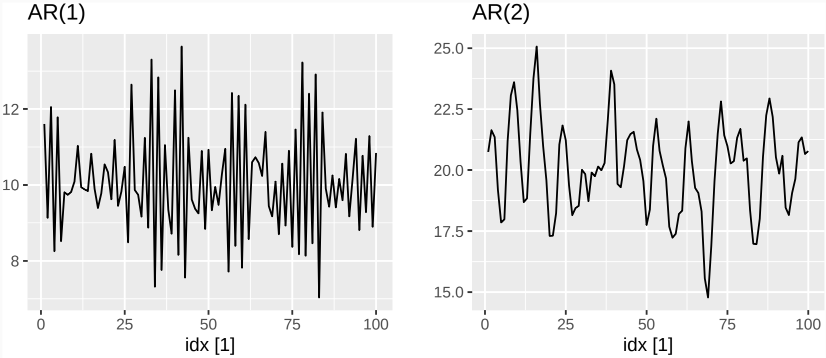
For monthly data, $m = 12$ and we obtain the same result as earlier.

Autoregressive models

Autoregressive (AR) models:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

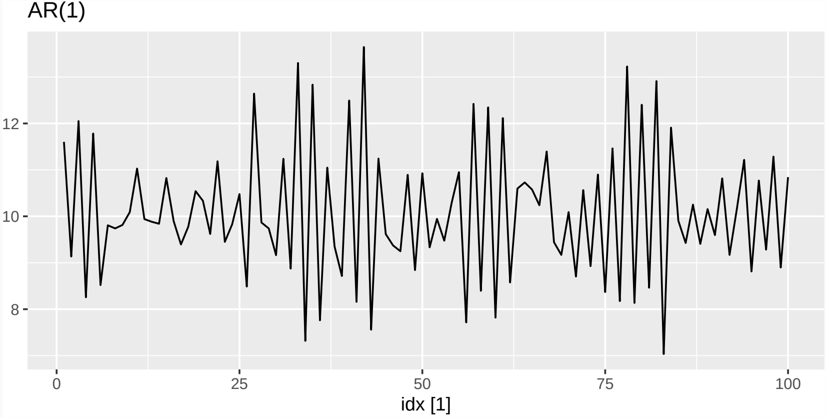
where ε_t is white noise. This is a multiple regression with **lagged values** of y_t as predictors.



AR(1) model

$$y_t = 2 - 0.8y_{t-1} + \varepsilon_t$$

$$\varepsilon_t \sim N(0, 1), \quad T = 100.$$



AR(1) model

$$y_t = c + \phi_1 y_{t-1} + \varepsilon_t$$

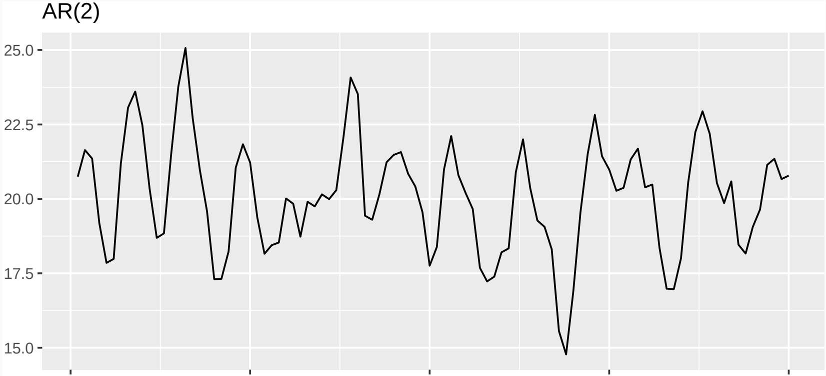
- When $\phi_1 = 0$, y_t is **equivalent to WN**
- When $\phi_1 = 1$ and $c = 0$, y_t is **equivalent to a RW**
- When $\phi_1 = 1$ and $c \neq 0$, y_t is **equivalent to a RW with drift**
- When $\phi_1 < 0$, y_t tends to **oscillate between positive and negative values**.

AR(2) model

$$y_t = 8 + 1.3y_{t-1} - 0.7y_{t-2} + \varepsilon_t$$

$$\varepsilon_t \sim N(0, 1), \qquad T = 100.$$

```
p2
```



Stationarity conditions

We normally restrict autoregressive models to stationary data, and then some constraints on the values of the parameters are required.

General condition for stationarity

Complex roots of $1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ lie outside the unit circle on the complex plane.

Stationarity conditions

We normally restrict autoregressive models to stationary data, and then some constraints on the values of the parameters are required.

General condition for stationarity

Complex roots of $1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ lie outside the unit circle on the complex plane.

- For $p = 1$: $-1 < \phi_1 < 1$.
- For $p = 2$:
  $-1 < \phi_2 < 1 \qquad \phi_2 + \phi_1 < 1 \qquad \phi_2 - \phi_1 < 1$.
- More complicated conditions hold for $p \geq 3$.
- Estimation software takes care of this.

Moving Average (MA) models

Moving Average (MA) models:

$$y_t = c + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q},$$

where ε_t is white noise. This is a multiple regression with ***past errors*** as predictors. *Don't confuse this with moving average smoothing!*

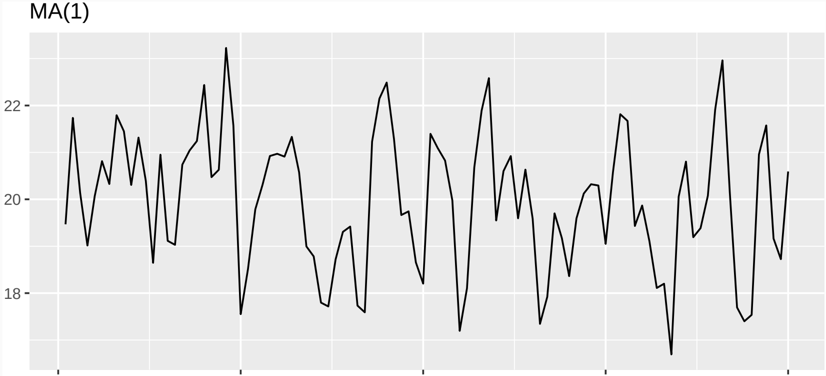
```
set.seed(2)
p1 <- tsibble(idx = seq_len(100), sim = 20 + a
  autoplot(sim) + ylab("") + ggtitle("MA(1)")
p2 <- tsibble(idx = seq_len(100), sim = arima.
  autoplot(sim) + ylab("") + ggtitle("MA(2)")
```

MA(1) model

$$y_t = 20 + \varepsilon_t + 0.8\varepsilon_{t-1}$$

$$\varepsilon_t \sim N(0, 1), \quad T = 100.$$

```
p1
```

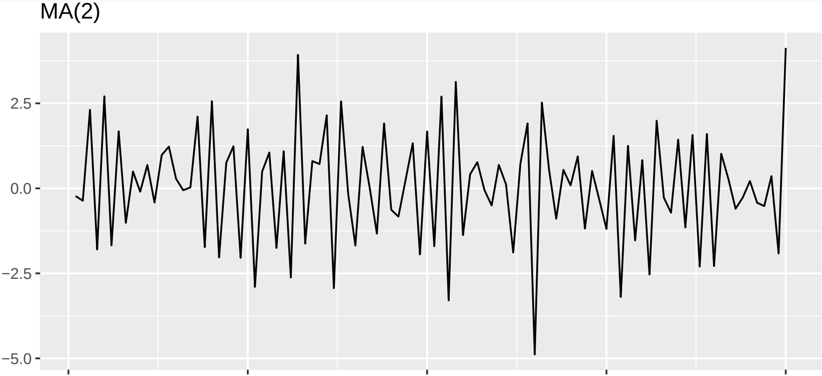


MA(2) model

$$y_t = \varepsilon_t - \varepsilon_{t-1} + 0.8\varepsilon_{t-2}$$

$$\varepsilon_t \sim N(0, 1), \quad T = 100.$$

```
p2
```



MA(∞) models

It is possible to write any stationary AR(p) process as an MA(∞) process.

Example: AR(1)

$$\begin{aligned} y_t &= \phi_1 y_{t-1} + \varepsilon_t \\ &= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\ &= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &\dots \end{aligned}$$

MA(∞) models

It is possible to write any stationary AR(p) process as an MA(∞) process.

Example: AR(1)

$$
\begin{aligned}
y_t &= \phi_1 y_{t-1} + \varepsilon_t \\
&= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\
&= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\
&= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\
&\dots
\end{aligned}
$$

Provided $-1 < \phi_1 < 1$:

$$
y_t = \varepsilon_t + \phi_1 \varepsilon_{t-1} + \phi_1^2 \varepsilon_{t-2} + \phi_1^3 \varepsilon_{t-3} + \cdots
$$

Invertibility

- Any MA(q) process can be written as an AR(∞) process if we impose some constraints on the MA parameters.
- Then the MA model is called "invertible".
- Invertible models have some mathematical properties that make them easier to use in practice.
- Invertibility of an ARIMA model is equivalent to forecastability of an ETS model.

Invertibility

General condition for invertibility

Complex roots of $1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$ lie outside the unit circle on the complex plane.

Invertibility

General condition for invertibility

Complex roots of $1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$ lie outside the unit circle on the complex plane.

- For $q = 1$: $-1 < \theta_1 < 1$.
- For $q = 2$:

  $-1 < \theta_2 < 1 \qquad \theta_2 + \theta_1 > -1 \qquad \theta_1 - \theta_2 < 1.$
- More complicated conditions hold for $q \geq 3$.
- Estimation software takes care of this.

ARIMA models

Autoregressive Moving Average models:

$$
\begin{aligned}
y_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} \\
+ \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t.
\end{aligned}
$$

ARIMA models

Autoregressive Moving Average models:

$$y_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t.$$

- Predictors include both **lagged values of y_t and lagged errors.**
- Conditions on coefficients ensure stationarity.
- Conditions on coefficients ensure invertibility.

ARIMA models

Autoregressive Moving Average models:

$$y_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t.$$

- Predictors include both **lagged values of y_t and lagged errors.**
- Conditions on coefficients ensure stationarity.
- Conditions on coefficients ensure invertibility.

Autoregressive Integrated Moving Average models

- Combine ARMA model with **differencing**.
- $(1 - B)^d y_t$ follows an ARMA model.

ARIMA models

Autoregressive Integrated Moving Average models

ARIMA(p, d, q) model

AR:	p = order of the autoregressive part
I:	d = degree of first differencing involved
MA:	q = order of the moving average part.

- White noise model: ARIMA(0,0,0)
- Random walk: ARIMA(0,1,0) with no constant
- Random walk with drift: ARIMA(0,1,0) with const.
- AR(p): ARIMA(p,0,0)
- MA(q): ARIMA(0,0,q)

Backshift notation for ARIMA

- ARMA model:

$$y_t = c + \phi_1 B y_t + \cdots + \phi_p B^p y_t + \varepsilon_t + \theta_1 B \varepsilon_t + \cdots + \theta_q B^q \varepsilon_t$$

$$\text{or} \quad (1 - \phi_1 B - \cdots - \phi_p B^p) y_t = c + (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

- ARIMA(1,1,1) model:

$$\underset{\substack{\uparrow \\ \text{AR(1)}}}{(1 - \phi_1 B)} \quad \underset{\substack{\uparrow \\ \text{First} \\ \text{difference}}}{(1 - B)} y_t = c + \underset{\substack{\uparrow \\ \text{MA(1)}}}{(1 + \theta_1 B)} \varepsilon_t$$

Backshift notation for ARIMA

- ARMA model:

$$\begin{aligned} y_t &= c + \phi_1 B y_t + \cdots + \phi_p B^p y_t + \varepsilon_t + \theta_1 B \varepsilon_t + \cdots + \theta_q B^q \varepsilon_t \\ \text{or} \quad (1 - \phi_1 B - \cdots - \phi_p B^p) y_t &= c + (1 + \theta_1 B + \cdots + \theta_q B^q) \varepsilon_t \end{aligned}$$

- ARIMA(1,1,1) model:

$$\begin{array}{ccccc} (1 - \phi_1 B) & (1 - B) y_t & = & c + (1 + \theta_1 B) \varepsilon_t \\ \uparrow & \uparrow & & \uparrow \\ \text{AR(1)} & \text{First} & & \text{MA(1)} \\ & \text{difference} & & \end{array}$$

Written out:

$$y_t = c + y_{t-1} + \phi_1 y_{t-1} - \phi_1 y_{t-2} + \theta_1 \varepsilon_{t-1} + \varepsilon_t$$

R model

Intercept form

$$(1 - \phi_1 B - \cdots - \phi_p B^p) y_t' = c + (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

Mean form

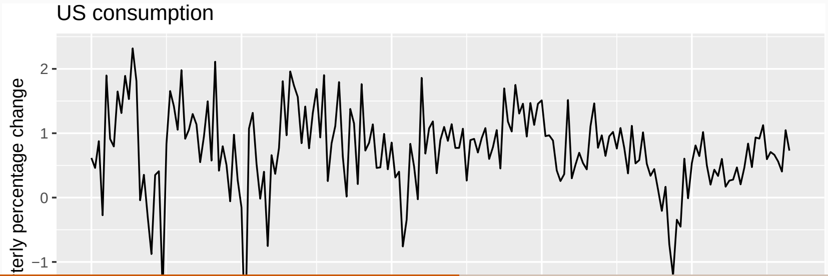
$$(1 - \phi_1 B - \cdots - \phi_p B^p)(y_t' - \mu) = (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

- $y_t' = (1 - B)^d y_t$
- μ is the mean of y_t'.
- $c = \mu(1 - \phi_1 - \cdots - \phi_p)$.
- R uses mean form
- fable uses intercept form

Australian household expenditure

```
us_change <- read_csv(
  "https://otexts.com/fpp3/extrafiles/us_change.csv") %>%
  mutate(Time = yearquarter(Time)) %>%
  as_tsibble(index = Time)
```

```
us_change %>% autoplot(Consumption) +
  xlab("Year") +
  ylab("Quarterly percentage change") +
  ggtitle("US consumption")
```



US personal consumption

```r
fit <- us_change %>% model(arima = ARIMA(Consumption ~ PDQ(0,0,0)))
report(fit)
```

```
## Series: Consumption
## Model: ARIMA(1,0,3) w/ mean
##
## Coefficients:
##          ar1     ma1     ma2     ma3  constant
##        0.589  -0.353  0.0846  0.1739    0.3067
## s.e.   0.154   0.166  0.0818  0.0843    0.0383
##
## sigma^2 estimated as 0.3499:  log likelihood=-165
## AIC=342   AICc=342   BIC=361
```

US personal consumption

```r
fit <- us_change %>% model(arima = ARIMA(Consumption ~ PDQ(0,0,0)))
report(fit)
```

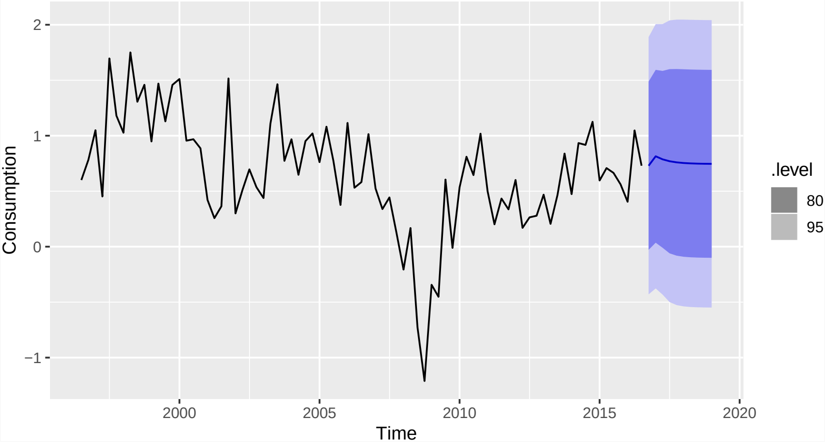
```
## Series: Consumption
## Model: ARIMA(1,0,3) w/ mean
##
## Coefficients:
##          ar1     ma1     ma2     ma3  constant
##        0.589  -0.353  0.0846  0.1739    0.3067
## s.e.   0.154   0.166  0.0818  0.0843    0.0383
##
## sigma^2 estimated as 0.3499:  log likelihood=-165
## AIC=342   AICc=342   BIC=361
```

```r
if(!grepl("ARIMA\\(1,0,3\\)", format(fit$arima)))
  warning("Needs fixing")
```

ARIMA(1,0,3) model:

US personal consumption

```
fit %>% forecast(h=10) %>%
  autoplot(slice(us_change, (n()-80):n()))
```



Understanding ARIMA models

- If $c = 0$ and $d = 0$, the long-term forecasts will go to zero.
- If $c = 0$ and $d = 1$, the long-term forecasts will go to a non-zero constant.
- If $c = 0$ and $d = 2$, the long-term forecasts will follow a straight line.
- If $c \neq 0$ and $d = 0$, the long-term forecasts will go to the mean of the data.
- If $c \neq 0$ and $d = 1$, the long-term forecasts will follow a straight line.
- If $c \neq 0$ and $d = 2$, the long-term forecasts will follow a quadratic trend.

Understanding ARIMA models

Forecast variance and *d*

- The higher the value of d, the more rapidly the prediction intervals increase in size.
- For $d = 0$, the long-term forecast standard deviation will go to the standard deviation of the historical data.

Cyclic behaviour

- For cyclic forecasts, $p \geq 2$ and some restrictions on coefficients are required.
- If $p = 2$, we need $\phi_1^2 + 4\phi_2 < 0$. Then average cycle of length

$$(2\pi)/\left[\text{arc cos}(-\phi_1(1-\phi_2)/(4\phi_2))\right].$$

Maximum likelihood estimation

Having identified the model order, we need to estimate the parameters $c, \phi_1, \dots, \phi_p, \theta_1, \dots, \theta_q$.

Maximum likelihood estimation

Having identified the model order, we need to estimate the parameters $c, \phi_1, \dots, \phi_p, \theta_1, \dots, \theta_q$.

- MLE is very similar to least squares estimation obtained by minimizing
$$\sum_{t-1}^{T} e_t^2$$
- The `ARIMA()` model allows CLS or MLE estimation.
- Non-linear optimization must be used in either case.
- Different software will give different estimates.

Partial autocorrelations

Partial autocorrelations measure relationship between y_t and y_{t-k}, when the effects of other time lags — $1, 2, 3, \dots, k-1$ — are removed.

Partial autocorrelations

Partial autocorrelations measure relationship between y_t and y_{t-k}, when the effects of other time lags — $1, 2, 3, \dots, k-1$ — are removed.

$$\begin{aligned} \alpha_k &= k\text{th partial autocorrelation coefficient} \\ &= \text{equal to the estimate of } \phi_k \text{ in regression:} \\ & \quad y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_k y_{t-k}. \end{aligned}$$

Partial autocorrelations

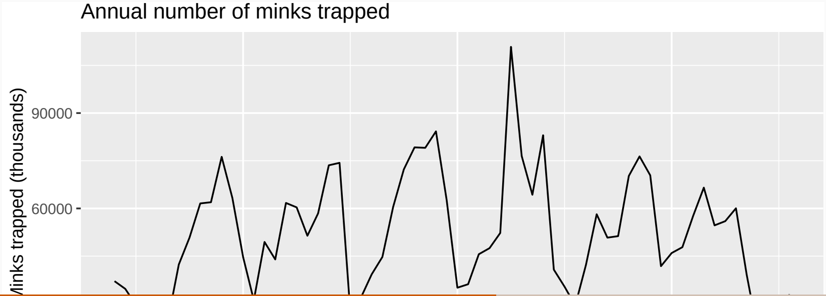
Partial autocorrelations measure relationship between y_t and y_{t-k}, when the effects of other time lags — $1, 2, 3, \dots, k-1$ — are removed.

$$\begin{aligned} \alpha_k &= k\text{th partial autocorrelation coefficient} \\ &= \text{equal to the estimate of } \phi_k \text{ in regression:} \\ & \qquad y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_k y_{t-k}. \end{aligned}$$

- Varying number of terms on RHS gives α_k for different values of k.
- There are more efficient ways of calculating α_k.
- $\alpha_1 = \rho_1$
- same critical values of $\pm 1.96/\sqrt{T}$ as for ACF.

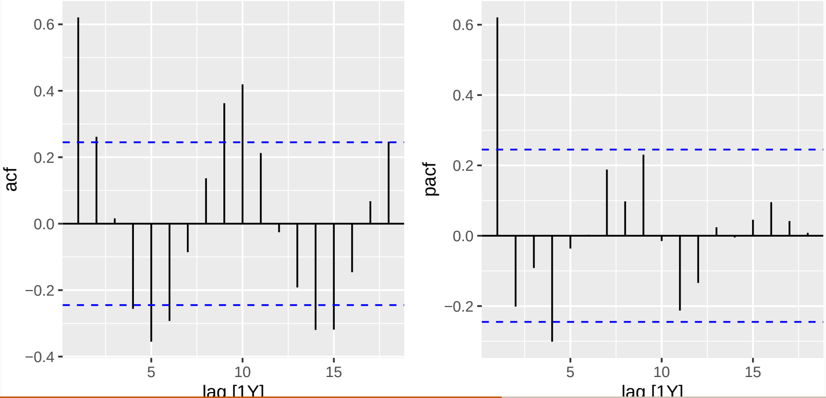
Example: Mink trapping

```r
mink <- as_tsibble(fma::mink)
mink %>% autoplot(value) +
  xlab("Year") +
  ylab("Minks trapped (thousands)") +
  ggtitle("Annual number of minks trapped")
```



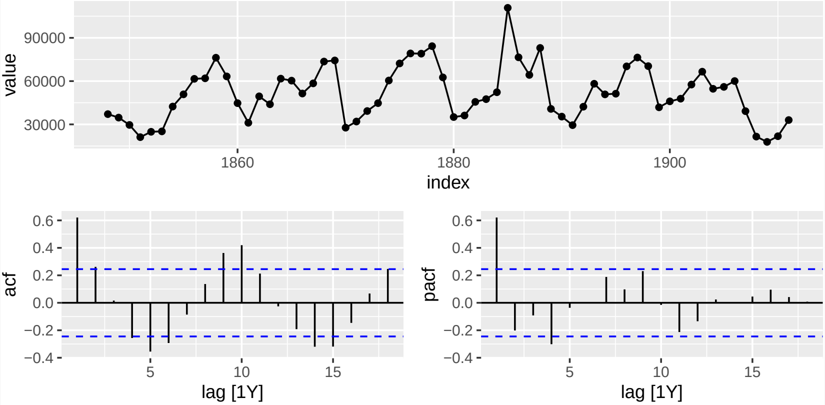
Example: Mink trapping

```r
p1 <- mink %>% ACF(value) %>% autoplot()
p2 <- mink %>% PACF(value) %>% autoplot()
gridExtra::grid.arrange(p1,p2,nrow=1)
```



Example: Mink trapping

```r
mink %>% gg_tsdisplay(value)
```



ACF and PACF interpretation

AR(1)

$$\rho_k = \phi_1^k \qquad \text{for } k = 1, 2, \dots;$$
$$\alpha_1 = \phi_1 \qquad \alpha_k = 0 \qquad \text{for } k = 2, 3, \dots.$$

So we have an AR(1) model when

- autocorrelations exponentially decay
- there is a single significant partial autocorrelation.

ACF and PACF interpretation

AR(p)

- ACF dies out in an exponential or damped sine-wave manner
- PACF has all zero spikes beyond the pth spike

So we have an AR(p) model when

- the ACF is exponentially decaying or sinusoidal
- there is a significant spike at lag p in PACF, but none beyond p

ACF and PACF interpretation

MA(1)

$$\rho_1 = \theta_1 \qquad \rho_k = 0 \qquad \text{for } k = 2, 3, \ldots;$$
$$\alpha_k = -(-\theta_1)^k$$

So we have an MA(1) model when

- the PACF is exponentially decaying and
- there is a single significant spike in ACF

ACF and PACF interpretation

MA(q)

- PACF dies out in an exponential or damped sine-wave manner
- ACF has all zero spikes beyond the qth spike

So we have an MA(q) model when

- the PACF is exponentially decaying or sinusoidal
- there is a significant spike at lag q in ACF, but none beyond q

Information criteria

Akaike's Information Criterion (AIC):

$$\text{AIC} = -2\log(L) + 2(p + q + k + 1),$$

where L is the likelihood of the data,

$k = 1$ if $c \neq 0$ and $k = 0$ if $c = 0$.

Information criteria

Akaike's Information Criterion (AIC):

$$\text{AIC} = -2\log(L) + 2(p + q + k + 1),$$

where L is the likelihood of the data,

$k = 1$ if $c \neq 0$ and $k = 0$ if $c = 0$.

Corrected AIC:

$$\text{AICc} = \text{AIC} + \frac{2(p+q+k+1)(p+q+k+2)}{T-p-q-k-2}.$$

Information criteria

Akaike's Information Criterion (AIC):

$$\text{AIC} = -2\log(L) + 2(p+q+k+1),$$

where L is the likelihood of the data,
$k = 1$ if $c \neq 0$ and $k = 0$ if $c = 0$.

Corrected AIC:

$$\text{AICc} = \text{AIC} + \frac{2(p+q+k+1)(p+q+k+2)}{T-p-q-k-2}.$$

Bayesian Information Criterion:

$$\text{BIC} = \text{AIC} + [\log(T) - 2](p+q+k-1).$$

Information criteria

Akaike's Information Criterion (AIC):

$$\text{AIC} = -2\log(L) + 2(p+q+k+1),$$

where L is the likelihood of the data,

$k = 1$ if $c \neq 0$ and $k = 0$ if $c = 0$.

Corrected AIC:

$$\text{AICc} = \text{AIC} + \frac{2(p+q+k+1)(p+q+k+2)}{T-p-q-k-2}.$$

Bayesian Information Criterion:

$$\text{BIC} = \text{AIC} + [\log(T) - 2](p+q+k-1).$$

Good models are obtained by minimizing either the AIC, AICc or BIC. Our preference is to use the AICc.

How does ARIMA() work?

A non-seasonal ARIMA process

$$\phi(B)(1 - B)^d y_t = c + \theta(B)\varepsilon_t$$

Need to select appropriate orders: p, q, d

Hyndman and Khandakar (JSS, 2008) algorithm:

- Select no. differences d and D via KPSS test and seasonal strength measure.
- Select p, q by minimising AICc.
- Use stepwise search to traverse model space.

How does ARIMA() work?

$$\text{AICc} = -2\log(L) + 2(p+q+k+1)\left[1 + \frac{(p+q+k+2)}{T-p-q-k-2}\right].$$

where L is the maximised likelihood fitted to the *differenced* data, $k = 1$ if $c \neq 0$ and $k = 0$ otherwise.

How does ARIMA() work?

$$\text{AICc} = -2\log(L) + 2(p+q+k+1)\left[1 + \tfrac{(p+q+k+2)}{T-p-q-k-2}\right].$$

where L is the maximised likelihood fitted to the *differenced* data, $k = 1$ if $c \neq 0$ and $k = 0$ otherwise.

Step1: Select current model (with smallest AICc) from:
ARIMA$(2, d, 2)$
ARIMA$(0, d, 0)$
ARIMA$(1, d, 0)$
ARIMA$(0, d, 1)$

How does ARIMA() work?

$$\text{AICc} = -2\log(L) + 2(p+q+k+1)\left[1 + \frac{(p+q+k+2)}{T-p-q-k-2}\right].$$

where L is the maximised likelihood fitted to the *differenced* data, $k = 1$ if $c \neq 0$ and $k = 0$ otherwise.

Step1: Select current model (with smallest AICc) from:
ARIMA$(2, d, 2)$
ARIMA$(0, d, 0)$
ARIMA$(1, d, 0)$
ARIMA$(0, d, 1)$

Step 2: Consider variations of current model:

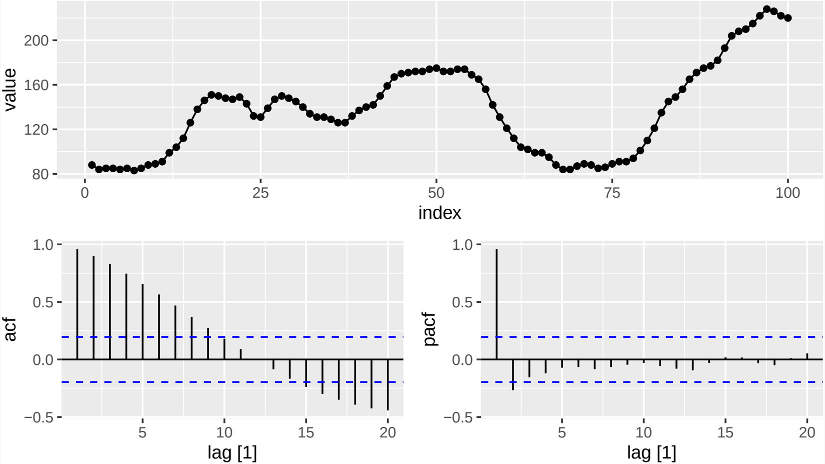
- vary one of $p, q,$ from current model by ± 1;
- p, q both vary from current model by ± 1;
- Include/exclude c from current model.

Model with lowest AICc becomes current model.

Repeat Step 2 until no lower AICc can be found.

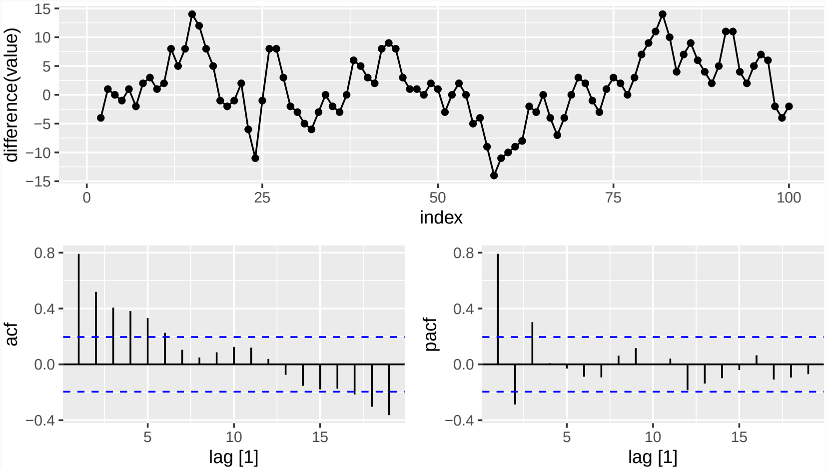
Choosing your own model

```
web_usage <- as_tsibble(WWWusage)
web_usage %>% gg_tsdisplay(value)
```



Choosing your own model

```
web_usage %>% gg_tsdisplay(difference(value))
```



Choosing your own model

```
fit <- web_usage %>% model(
  arima = ARIMA(value ~ pdq(3, 1, 0)))
report(fit)
```

```
## Series: value
## Model: ARIMA(3,1,0)
##
## Coefficients:
##          ar1     ar2     ar3
##        1.151  -0.661  0.3407
## s.e.   0.095   0.135  0.0941
##
## sigma^2 estimated as 9.656:  log likelihood=-252
## AIC=512   AICc=512   BIC=522
```

Choosing your own model

```r
web_usage %>% model(ARIMA(value ~ pdq(d=1))) %>% report()
```

```
## Series: value
## Model: ARIMA(1,1,1)
##
## Coefficients:
##          ar1     ma1
##       0.6504  0.5256
## s.e.  0.0842  0.0896
##
## sigma^2 estimated as 9.995:  log likelihood=-254
## AIC=514   AICc=515   BIC=522
```

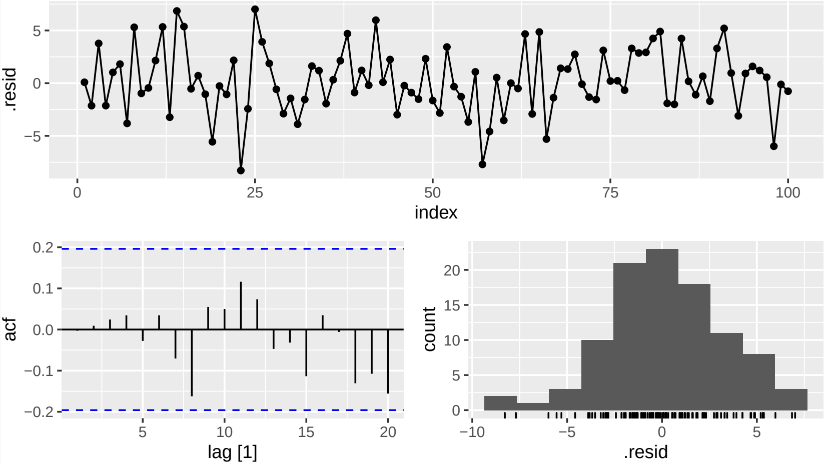
Choosing your own model

```r
web_usage %>%
  model(ARIMA(value ~ pdq(d=1), stepwise = FALSE,
    approximation = FALSE)) %>% report()
```

```
## Series: value
## Model: ARIMA(3,1,0)
##
## Coefficients:
##          ar1     ar2     ar3
##        1.151  -0.661  0.3407
## s.e.   0.095   0.135  0.0941
##
## sigma^2 estimated as 9.656:  log likelihood=-252
## AIC=512   AICc=512   BIC=522
```

Choosing your own model

```r
augment(fit) %>%
  gg_tsdisplay(.resid, plot_type = "hist")
```



Choosing your own model

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 10, dof = 3)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 arima     4.49     0.722
```

Choosing your own model

```
fit %>% forecast(h = 10) %>%
  autoplot(web_usage)
```

Modelling procedure with `ARIMA`

1. Plot the data. Identify any unusual observations.
2. If necessary, transform the data (using a Box-Cox transformation) to stabilize the variance.
3. If the data are non-stationary: take first differences of the data until the data are stationary.
4. Examine the ACF/PACF: Is an AR(p) or MA(q) model appropriate?
5. Try your chosen model(s), and use the AICc to search for a better model.
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals. If they do not look like white noise, try a modified model.
7. Once the residuals look like white noise, calculate forecasts.

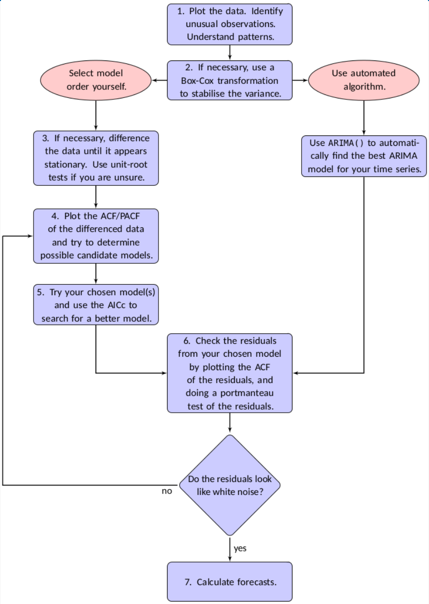
Automatic modelling procedure with `ARIMA`

1. Plot the data. Identify any unusual observations.
2. If necessary, transform the data (using a Box-Cox transformation) to stabilize the variance.

3. Use `ARIMA` to automatically select a model.

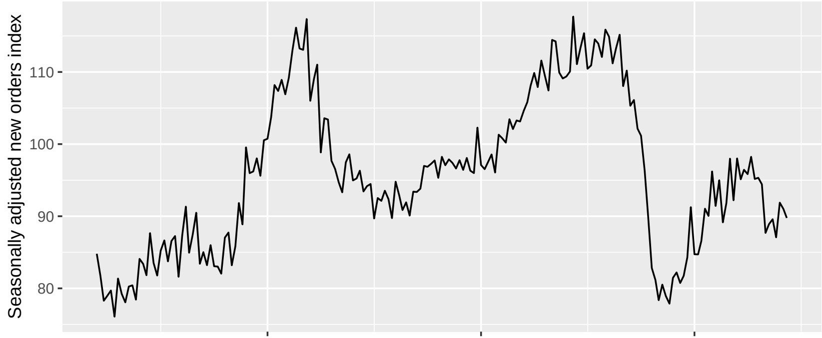
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals. If they do not look like white noise, try a modified model.
7. Once the residuals look like white noise, calculate forecasts.

Modelling procedure



Seasonally adjusted electrical equipment

```r
elecequip <- as_tsibble(fpp2::elecequip)
dcmp <- elecequip %>%
  STL(value ~ season(window = "periodic"))
dcmp %>% as_tsibble %>%
  autoplot(season_adjust) + xlab("Year") +
  ylab("Seasonally adjusted new orders index")
```

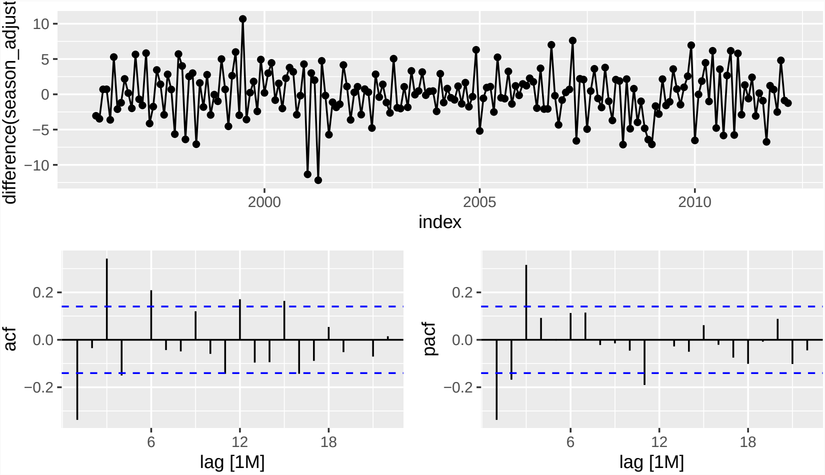


Seasonally adjusted electrical equipment

1. Time plot shows sudden changes, particularly big drop in 2008/2009 due to global economic environment. Otherwise nothing unusual and no need for data adjustments.
2. No evidence of changing variance, so no Box-Cox transformation.
3. Data are clearly non-stationary, so we take first differences.

Seasonally adjusted electrical equipment

```
dcmp %>% gg_tsdisplay(difference(season_adjust))
```



Seasonally adjusted electrical equipment

4. PACF is suggestive of AR(3). So initial candidate model is ARIMA(3,1,0). No other obvious candidates.
5. Fit ARIMA(3,1,0) model along with variations: ARIMA(4,1,0), ARIMA(2,1,0), ARIMA(3,1,1), etc. ARIMA(3,1,1) has smallest AICc value.

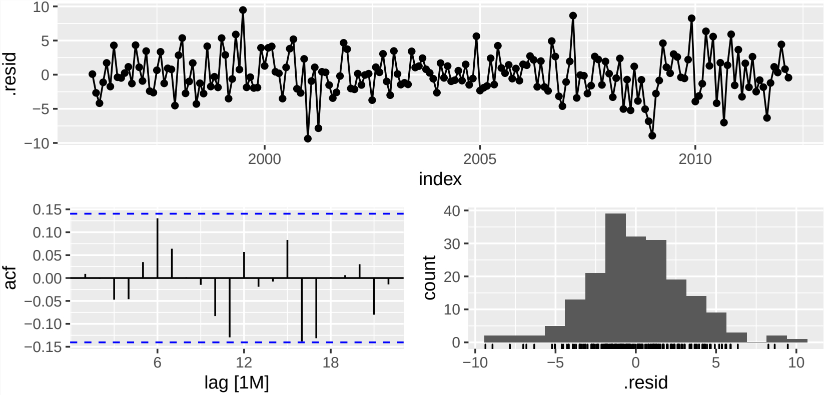
Seasonally adjusted electrical equipment

```r
fit <- dcmp %>%
  model(
    arima = ARIMA(season_adjust ~ pdq(3,1,1) + PDQ(0,0,0))
  )
report(fit)
```

```
## Series: season_adjust
## Model: ARIMA(3,1,1)
##
## Coefficients:
##          ar1     ar2     ar3     ma1
##       0.0044  0.0916  0.3698  -0.392
## s.e.  0.2201  0.0984  0.0669   0.243
##
## sigma^2 estimated as 9.577:  log likelihood=-493
## AIC=995   AICc=996   BIC=1012
```

Seasonally adjusted electrical equipment

6. ACF plot of residuals from ARIMA(3,1,1) model look like white noise.



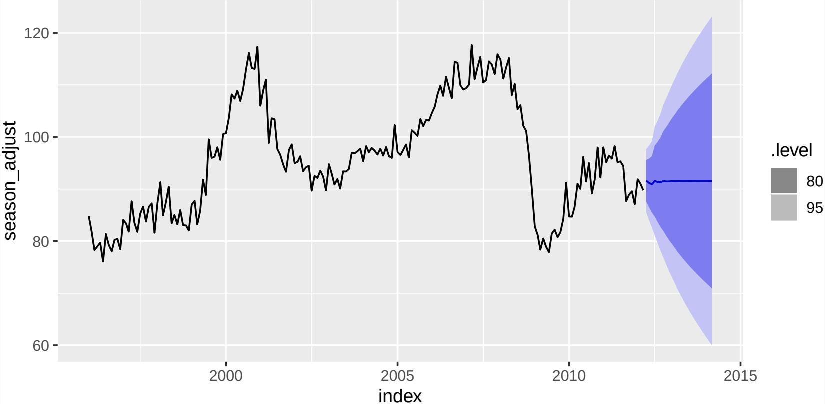
Seasonally adjusted electrical equipment

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 24, dof = 4)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 arima     24.0     0.241
```

Seasonally adjusted electrical equipment

```
fit %>% forecast %>% autoplot(dcmp)
```



Point forecasts

1. Rearrange ARIMA equation so y_t is on LHS.
2. Rewrite equation by replacing t by $T + h$.
3. On RHS, replace future observations by their forecasts, future errors by zero, and past errors by corresponding residuals.

Start with $h = 1$. Repeat for $h = 2, 3, \ldots$.

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3)(1 - B)y_t = (1 + \theta_1 B)\varepsilon_t,$$

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1-\phi_1B-\phi_2B^2-\phi_3B^3)(1-B)y_t = (1+\theta_1B)\varepsilon_t,$$

$$\begin{aligned}\left[1-(1+\phi_1)B+(\phi_1-\phi_2)B^2+(\phi_2-\phi_3)B^3+\phi_3B^4\right] y_t \\ = (1+\theta_1B)\varepsilon_t,\end{aligned}$$

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3)(1 - B) y_t = (1 + \theta_1 B)\varepsilon_t,$$

$$\begin{aligned}\left[1 - (1 + \phi_1)B + (\phi_1 - \phi_2)B^2 + (\phi_2 - \phi_3)B^3 + \phi_3 B^4\right] y_t \\ = (1 + \theta_1 B)\varepsilon_t,\end{aligned}$$

$$\begin{aligned}y_t - (1 + \phi_1)y_{t-1} + (\phi_1 - \phi_2)y_{t-2} + (\phi_2 - \phi_3)y_{t-3} \\ + \phi_3 y_{t-4} = \varepsilon_t + \theta_1 \varepsilon_{t-1}.\end{aligned}$$

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3)(1 - B) y_t = (1 + \theta_1 B)\varepsilon_t,$$

$$\begin{aligned}\left[1 - (1 + \phi_1)B + (\phi_1 - \phi_2)B^2 + (\phi_2 - \phi_3)B^3 + \phi_3 B^4\right] y_t \\ = (1 + \theta_1 B)\varepsilon_t,\end{aligned}$$

$$\begin{aligned}y_t - (1 + \phi_1)y_{t-1} + (\phi_1 - \phi_2)y_{t-2} + (\phi_2 - \phi_3)y_{t-3} \\ + \phi_3 y_{t-4} = \varepsilon_t + \theta_1 \varepsilon_{t-1}.\end{aligned}$$

$$\begin{aligned}y_t = (1 + \phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\ - \phi_3 y_{t-4} + \varepsilon_t + \theta_1 \varepsilon_{t-1}.\end{aligned}$$

Point forecasts (h=1)

$$\begin{aligned} y_t = (1 + \phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\ - \phi_3 y_{t-4} + \varepsilon_t + \theta_1 \varepsilon_{t-1}. \end{aligned}$$

Point forecasts (h=1)

$$y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

ARIMA(3,1,1) forecasts: Step 2

$$y_{T+1} = (1+\phi_1)y_T - (\phi_1 - \phi_2)y_{T-1} - (\phi_2 - \phi_3)y_{T-2} - \phi_3 y_{T-3} + \varepsilon_{T+1} + \theta_1\varepsilon_T.$$

Point forecasts (h=1)

$$y_t = (1 + \phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

ARIMA(3,1,1) forecasts: Step 2

$$y_{T+1} = (1 + \phi_1)y_T - (\phi_1 - \phi_2)y_{T-1} - (\phi_2 - \phi_3)y_{T-2} - \phi_3 y_{T-3} + \varepsilon_{T+1} + \theta_1\varepsilon_T.$$

ARIMA(3,1,1) forecasts: Step 3

$$\hat{y}_{T+1|T} = (1 + \phi_1)y_T - (\phi_1 - \phi_2)y_{T-1} - (\phi_2 - \phi_3)y_{T-2} - \phi_3 y_{T-3} + \theta_1 e_T.$$

Point forecasts (h=2)

$$
\begin{aligned}
y_t = (1 + \phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\
- \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.
\end{aligned}
$$

Point forecasts (h=2)

$$\begin{aligned} y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\ - \phi_3 y_{t-4} + \varepsilon_t + \theta_1 \varepsilon_{t-1}. \end{aligned}$$

ARIMA(3,1,1) forecasts: Step 2

$$\begin{aligned} y_{T+2} = (1+\phi_1)y_{T+1} - (\phi_1 - \phi_2)y_T - (\phi_2 - \phi_3)y_{T-1} \\ - \phi_3 y_{T-2} + \varepsilon_{T+2} + \theta_1 \varepsilon_{T+1}. \end{aligned}$$

Point forecasts (h=2)

$$y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

ARIMA(3,1,1) forecasts: Step 2

$$y_{T+2} = (1+\phi_1)y_{T+1} - (\phi_1 - \phi_2)y_{T} - (\phi_2 - \phi_3)y_{T-1} - \phi_3 y_{T-2} + \varepsilon_{T+2} + \theta_1\varepsilon_{T+1}.$$

ARIMA(3,1,1) forecasts: Step 3

$$\hat{y}_{T+2|T} = (1+\phi_1)\hat{y}_{T+1|T} - (\phi_1 - \phi_2)y_{T} - (\phi_2 - \phi_3)y_{T-1} - \phi_3 y_{T-2}.$$

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

- $v_{T+1|T} = \hat{\sigma}^2$ for all ARIMA models regardless of parameters and orders.
- Multi-step prediction intervals for ARIMA(0,0,q):

$$y_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

$$v_{T|T+h} = \hat{\sigma}^2 \left[1 + \sum_{i=1}^{h-1} \theta_i^2\right], \qquad \text{for } h = 2, 3, \dots.$$

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

- Multi-step prediction intervals for ARIMA(0,0,q):

$$y_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

$$v_{T|T+h} = \hat{\sigma}^2 \left[1 + \sum_{i=1}^{h-1} \theta_i^2\right], \qquad \text{for } h = 2, 3, \dots .$$

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

- Multi-step prediction intervals for ARIMA(0,0,q):

$$y_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

$$v_{T|T+h} = \hat{\sigma}^2 \left[1 + \sum_{i=1}^{h-1} \theta_i^2\right], \qquad \text{for } h = 2, 3, \dots.$$

- AR(1): Rewrite as MA(∞) and use above result.
- Other models beyond scope of this subject.

Prediction intervals

- Prediction intervals **increase in size with forecast horizon**.
- Prediction intervals can be difficult to calculate by hand
- Calculations assume residuals are **uncorrelated** and **normally distributed**.
- Prediction intervals tend to be too narrow.
  - the uncertainty in the parameter estimates has not been accounted for.
  - the ARIMA model assumes historical patterns will not change during the forecast period.
  - the ARIMA model assumes uncorrelated future errors

Your turn

For the United States GDP data (from `global_economy`):

- if necessary, find a suitable Box-Cox transformation for the data;
- fit a suitable ARIMA model to the transformed data;
- check the residual diagnostics;
- produce forecasts of your fitted model. Do the forecasts look reasonable?

Seasonal ARIMA models

ARIMA	$\underbrace{(p, d, q)}$	$\underbrace{(P, D, Q)_m}$
	$\uparrow$	$\uparrow$
	Non-seasonal part of the model	Seasonal part of of the model

where m = number of observations per year.

Seasonal ARIMA models

E.g., ARIMA$(1, 1, 1)(1, 1, 1)_4$ model (without constant)

Seasonal ARIMA models

E.g., ARIMA(1, 1, 1)(1, 1, 1)$_4$ model (without constant)

$$(1-\phi_1 B)(1-\Phi_1 B^4)(1-B)(1-B^4)y_t = (1+\theta_1 B)(1+\Theta_1 B^4)\varepsilon_t.$$

Seasonal ARIMA models

E.g., ARIMA$(1, 1, 1)(1, 1, 1)_4$ model (without constant)

$$\underset{\begin{pmatrix}\text{Non-seasonal}\\ \text{AR(1)}\end{pmatrix}}{(1-\phi_1 B)}\;\underset{\begin{pmatrix}\text{Seasonal}\\ \text{AR(1)}\end{pmatrix}}{(1-\Phi_1 B^4)}\;\underset{\begin{pmatrix}\text{Non-seasonal}\\ \text{difference}\end{pmatrix}}{(1-B)}\;\underset{\begin{pmatrix}\text{Seasonal}\\ \text{difference}\end{pmatrix}}{(1-B^4)}\,y_t \;=\; \underset{\begin{pmatrix}\text{Non-seasonal}\\ \text{MA(1)}\end{pmatrix}}{(1+\theta_1 B)}\;\underset{\begin{pmatrix}\text{Seasonal}\\ \text{MA(1)}\end{pmatrix}}{(1+\Theta_1 B^4)}\,\varepsilon_t.$$

Seasonal ARIMA models

E.g., ARIMA(1, 1, 1)(1, 1, 1)$_4$ model (without constant)

$$(1-\phi_1 B)(1-\Phi_1 B^4)(1-B)(1-B^4)y_t = (1+\theta_1 B)(1+\Theta_1 B^4)\varepsilon_t.$$

All the factors can be multiplied out and the general model written as follows:

$$\begin{aligned}
y_t &= (1+\phi_1)y_{t-1} - \phi_1 y_{t-2} + (1+\Phi_1)y_{t-4} \\
&\quad - (1+\phi_1+\Phi_1+\phi_1\Phi_1)y_{t-5} + (\phi_1+\phi_1\Phi_1)y_{t-6} \\
&\quad - \Phi_1 y_{t-8} + (\Phi_1+\phi_1\Phi_1)y_{t-9} - \phi_1\Phi_1 y_{t-10} \\
&\quad + \varepsilon_t + \theta_1\varepsilon_{t-1} + \Theta_1\varepsilon_{t-4} + \theta_1\Theta_1\varepsilon_{t-5}.
\end{aligned}$$

Common ARIMA models

The US Census Bureau uses the following models most often:

ARIMA(0,1,1)(0,1,1)$_m$	with log transformation
ARIMA(0,1,2)(0,1,1)$_m$	with log transformation
ARIMA(2,1,0)(0,1,1)$_m$	with log transformation
ARIMA(0,2,2)(0,1,1)$_m$	with log transformation
ARIMA(2,1,2)(0,1,1)$_m$	with no transformation

Seasonal ARIMA models

The seasonal part of an AR or MA model will be seen in the seasonal lags of the PACF and ACF.

ARIMA(0,0,0)(0,0,1)$_{12}$ will show:

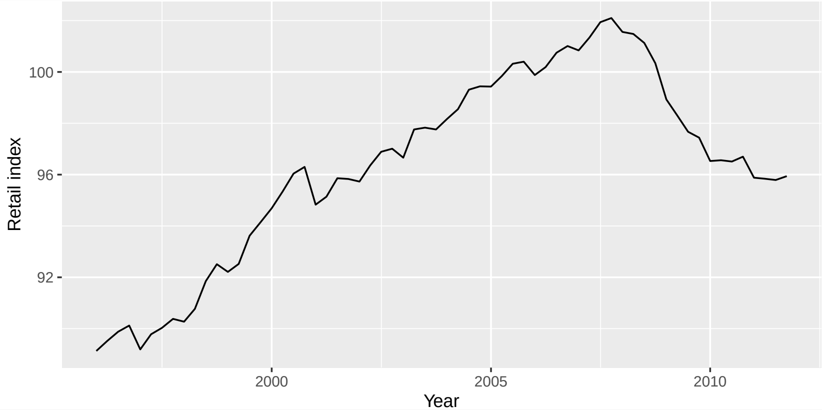
- a spike at lag 12 in the ACF but no other significant spikes.
- The PACF will show exponential decay in the seasonal lags; that is, at lags 12, 24, 36, ….

ARIMA(0,0,0)(1,0,0)$_{12}$ will show:

- exponential decay in the seasonal lags of the ACF
- a single significant spike at lag 12 in the PACF.

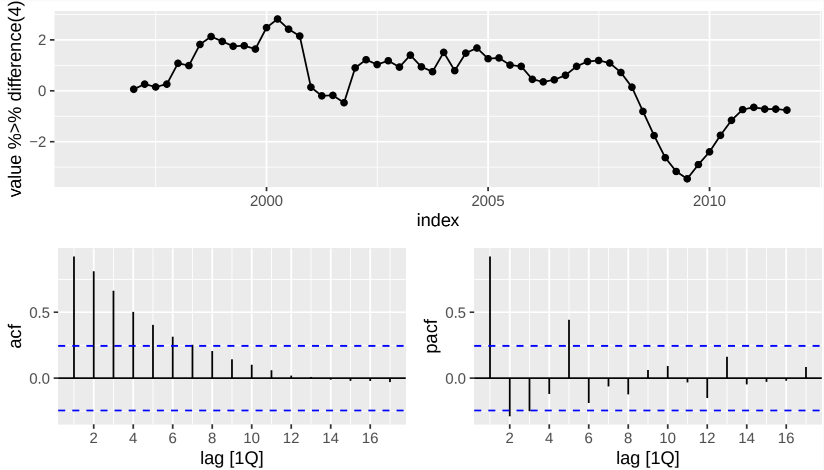
European quarterly retail trade

```r
eu_retail %>% autoplot(value) +
  xlab("Year") + ylab("Retail index")
```



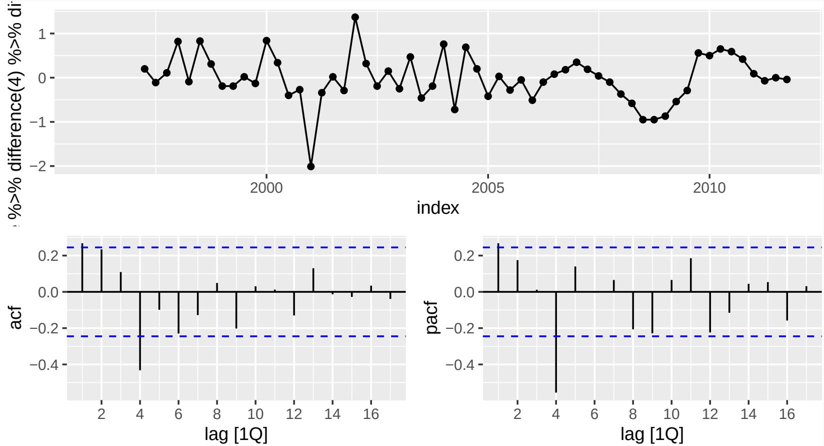
European quarterly retail trade

```
eu_retail %>% gg_tsdisplay(
  value %>% difference(4))
```



European quarterly retail trade

```
eu_retail %>% gg_tsdisplay(
  value %>% difference(4) %>% difference(1))
```

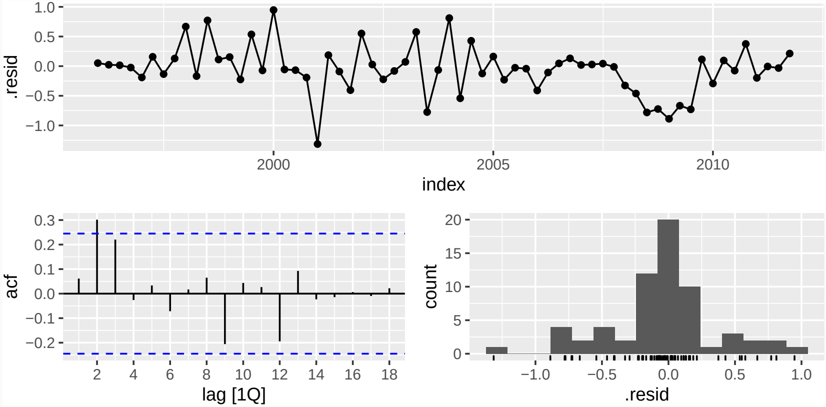


European quarterly retail trade

- $d = 1$ and $D = 1$ seems necessary.
- Significant spike at lag 1 in ACF suggests non-seasonal MA(1) component.
- Significant spike at lag 4 in ACF suggests seasonal MA(1) component.
- Initial candidate model: ARIMA(0,1,1)(0,1,1)$_4$.
- We could also have started with ARIMA(1,1,0)(1,1,0)$_4$.

European quarterly retail trade

```
fit <- eu_retail %>%
  model(arima = ARIMA(value ~ pdq(0,1,1) + PDQ(0,1,1)))
augment(fit) %>% gg_tsdisplay(.resid, plot_type = "hist")
```



European quarterly retail trade

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 8, dof = 2)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 arima     10.7    0.0997
```

European quarterly retail trade

- ACF and PACF of residuals show significant spikes at lag 2, and maybe lag 3.
- AICc of ARIMA$(0,1,1)(0,1,1)_4$ model is 75.72
- AICc of ARIMA$(0,1,2)(0,1,1)_4$ model is 74.27.
- AICc of ARIMA$(0,1,3)(0,1,1)_4$ model is 68.39.
- AICc of ARIMA$(0,1,4)(0,1,1)_4$ model is 70.73.

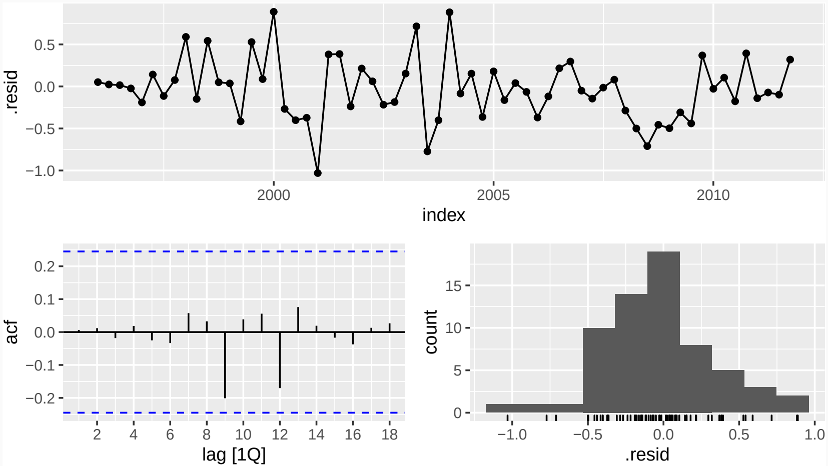
European quarterly retail trade

```r
fit <- eu_retail %>%
  model(
    arima013011 = ARIMA(value ~ pdq(0,1,3) + PDQ(0,1,1))
  )
report(fit)
```

```
## Series: value
## Model: ARIMA(0,1,3)(0,1,1)[4]
##
## Coefficients:
##          ma1    ma2    ma3    sma1
##        0.263  0.369  0.420  -0.664
## s.e.   0.124  0.126  0.129   0.154
##
## sigma^2 estimated as 0.156:  log likelihood=-28.6
## AIC=67.3   AICc=68.4   BIC=77.7
```

European quarterly retail trade

```r
augment(fit) %>%
  gg_tsdisplay(.resid, plot_type = "hist")
```



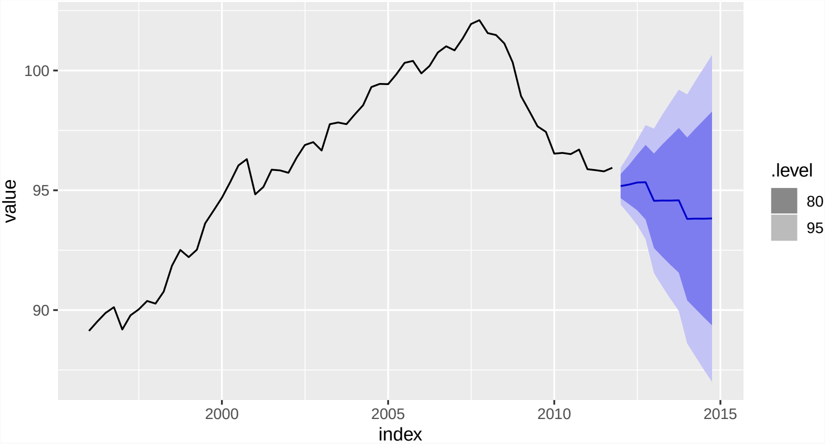
European quarterly retail trade

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 8, dof = 4)
```

```
## # A tibble: 1 x 3
##   .model       lb_stat lb_pvalue
##   <chr>          <dbl>     <dbl>
## 1 arima013011    0.511     0.972
```

European quarterly retail trade

```r
fit %>% forecast(h = "3 years") %>%
  autoplot(eu_retail)
```



European quarterly retail trade

```r
eu_retail %>% model(ARIMA(value)) %>% report()
```

```
## Series: value
## Model: ARIMA(0,1,3)(0,1,1)[4]
##
## Coefficients:
##          ma1    ma2    ma3     sma1
##        0.263  0.369  0.420  -0.664
## s.e.   0.124  0.126  0.129   0.154
##
## sigma^2 estimated as 0.156:  log likelihood=-28.6
## AIC=67.3   AICc=68.4   BIC=77.7
```

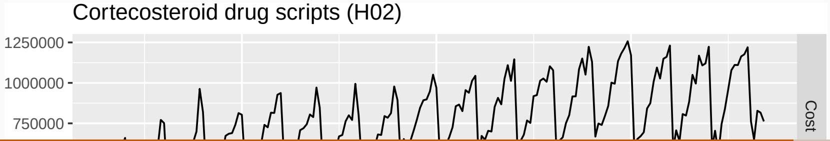
European quarterly retail trade

```r
eu_retail %>% model(ARIMA(value, stepwise = FALSE,
  approximation = FALSE)) %>% report()
```

```
## Series: value
## Model: ARIMA(0,1,3)(0,1,1)[4]
##
## Coefficients:
##          ma1    ma2    ma3    sma1
##        0.263  0.369  0.420  -0.664
## s.e.   0.124  0.126  0.129   0.154
##
## sigma^2 estimated as 0.156:  log likelihood=-28.6
## AIC=67.3   AICc=68.4   BIC=77.7
```

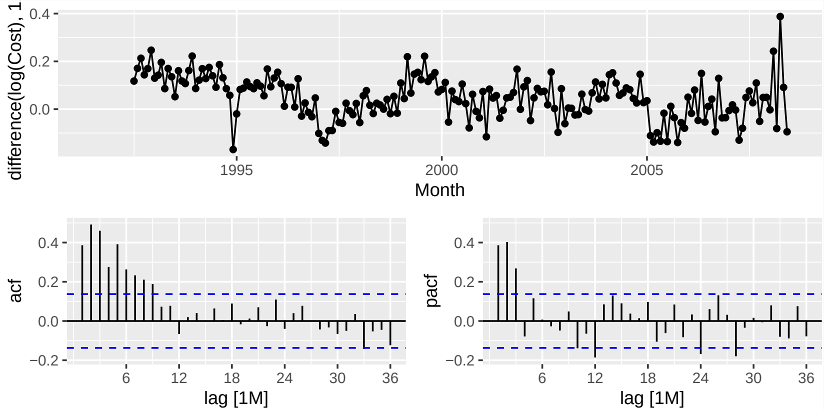
Cortecosteroid drug sales

```
h02 %>%
  mutate(log(Cost)) %>%
  gather() %>%
  ggplot(aes(x = Month, y = value)) +
  geom_line() +
  facet_grid(key ~ ., scales = "free_y") +
  xlab("Year") + ylab("") +
  ggtitle("Cortecosteroid drug scripts (H02)")
```



Cortecosteroid drug sales

```
h02 %>% gg_tsdisplay(difference(log(Cost),12),
```



Cortecosteroid drug sales

- Choose $D = 1$ and $d = 0$.
- Spikes in PACF at lags 12 and 24 suggest seasonal AR(2) term.
- Spikes in PACF suggests possible non-seasonal AR(3) term.
- Initial candidate model: ARIMA$(3,0,0)(2,1,0)_{12}$.

Cortecosteroid drug sales

.model	AICc
ARIMA(3,0,1)(0,1,2)[12]	-485
ARIMA(3,0,1)(1,1,1)[12]	-484
ARIMA(3,0,1)(0,1,1)[12]	-484
ARIMA(3,0,1)(2,1,0)[12]	-476
ARIMA(3,0,0)(2,1,0)[12]	-475
ARIMA(3,0,2)(2,1,0)[12]	-475
ARIMA(3,0,1)(1,1,0)[12]	-463

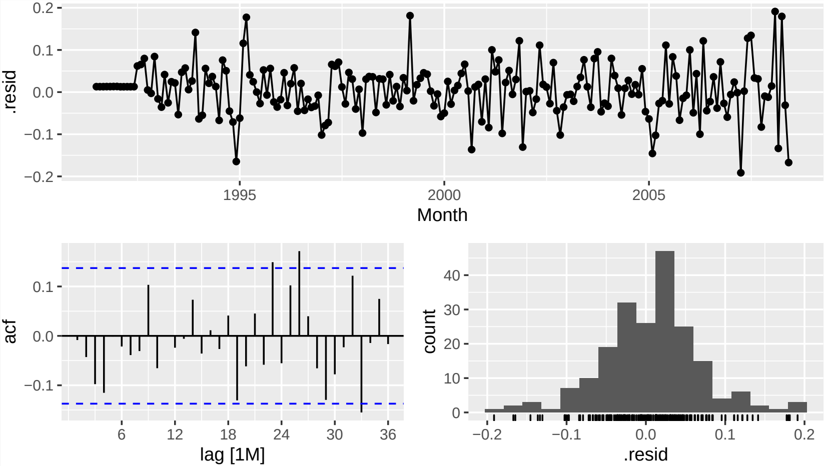
Cortecosteroid drug sales

```
fit <- h02 %>%
  model(best = ARIMA(log(Cost) ~ 0 + pdq(3,0,1) + PDQ(0,1,2)))
report(fit)
```

```
## Series: Cost
## Model: ARIMA(3,0,1)(0,1,2)[12]
## Transformation: log(.x)
##
## Coefficients:
##          ar1     ar2     ar3    ma1     sma1     sma2
##       -0.160  0.5481  0.5678  0.383  -0.5222  -0.1769
## s.e.   0.164  0.0878  0.0942  0.190   0.0861   0.0872
##
## sigma^2 estimated as 0.004289:  log likelihood=250
## AIC=-486   AICc=-485   BIC=-463
```

Cortecosteroid drug sales

```r
augment(fit) %>%
  gg_tsdisplay(.resid, lag_max=36, plot_type = "hist")
```



Cortecosteroid drug sales

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 36, dof = 6)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>     <dbl>     <dbl>
## 1 best       50.5    0.0109
```

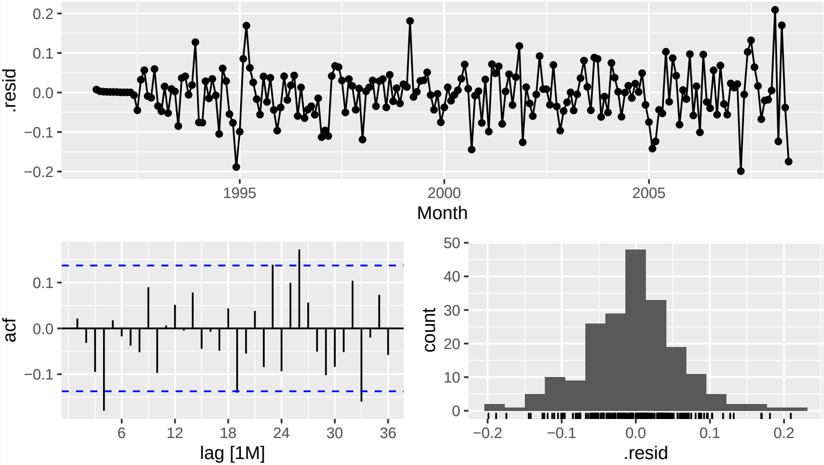
Cortecosteroid drug sales

```r
fit <- h02 %>% model(auto = ARIMA(log(Cost)))
report(fit)
```

```
## Series: Cost
## Model: ARIMA(2,1,0)(0,1,1)[12]
## Transformation: log(.x)
##
## Coefficients:
##           ar1      ar2     sma1
##       -0.8491  -0.4207  -0.6401
## s.e.   0.0712   0.0714   0.0694
##
## sigma^2 estimated as 0.004399:  log likelihood=245
## AIC=-483   AICc=-483   BIC=-470
```

Cortecosteroid drug sales

```r
augment(fit) %>%
  gg_tsdisplay(.resid, lag_max = 36, plot_type = "hist")
```



Cortecosteroid drug sales

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 36, dof = 5)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 auto      57.5   0.00260
```

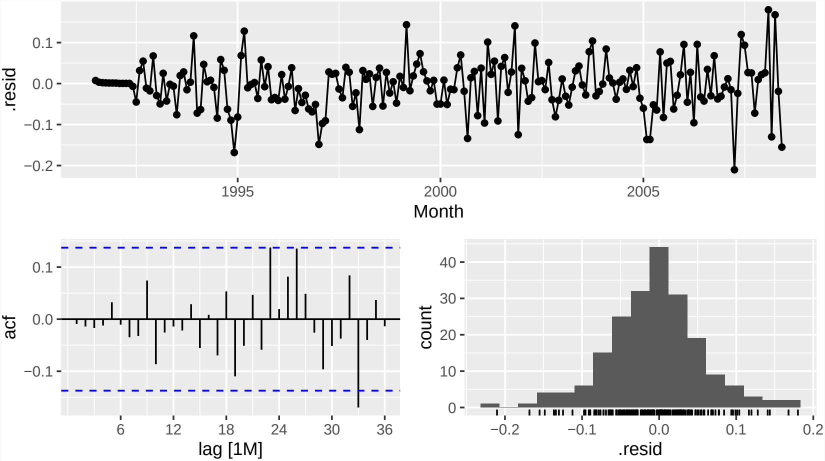
Cortecosteroid drug sales

```
fit <- h02 %>%
  model(best = ARIMA(log(Cost), stepwise = FALSE,
                     approximation = FALSE,
                     order_constraint = p + q + P + Q <= 9))
report(fit)
```

```
## Series: Cost
## Model: ARIMA(4,1,1)(2,1,2)[12]
## Transformation: log(.x)
##
## Coefficients:
##          ar1    ar2    ar3     ar4     ma1   sar1    sar2
##      -0.0426  0.210  0.202  -0.227  -0.742  0.621  -0.383
## s.e.  0.2167  0.181  0.114   0.081   0.207  0.242   0.118
##         sma1   sma2
##       -1.202  0.496
## s.e.   0.249  0.214
##
## sigma^2 estimated as 0.004061:  log likelihood=254
## AIC=-489   AICc=-487   BIC=-456
```

Cortecosteroid drug sales

```r
augment(fit) %>%
  gg_tsdisplay(.resid, lag_max = 36, plot_type = "hist")
```



Cortecosteroid drug sales

```r
augment(fit) %>%
  features(.resid, ljung_box, lag = 36, dof = 9)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 best      35.1     0.136
```

Cortecosteroid drug sales

Training data: July 1991 to June 2006

Test data: July 2006–June 2008

```
fit <- h02 %>%
  filter_index(~ "2006 Jun") %>%
  model(
    ARIMA(log(Cost) ~ pdq(3, 0, 0) + PDQ(2, 1, 0)),
    ARIMA(log(Cost) ~ pdq(3, 0, 1) + PDQ(2, 1, 0)),
    ARIMA(log(Cost) ~ pdq(3, 0, 2) + PDQ(2, 1, 0)),
    ARIMA(log(Cost) ~ pdq(3, 0, 1) + PDQ(1, 1, 0))
    # ... #
  )

fit %>%
  forecast(h = "2 years") %>%
  accuracy(h02 %>% filter_index("2006 Jul" ~ .))
```

Cortecosteroid drug sales

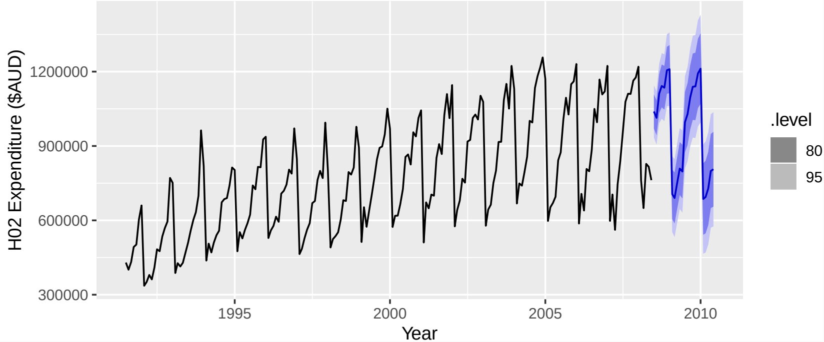
```
models <- list(
  c(3,0,0,2,1,0),
  c(3,0,1,2,1,0),
  c(3,0,2,2,1,0),
  c(3,0,1,1,1,0),
  c(3,0,1,0,1,1),
  c(3,0,1,0,1,2),
  c(3,0,1,1,1,1),
  c(3,0,3,0,1,1),
  c(3,0,2,0,1,1),
  c(2,1,3,0,1,1),
  c(2,1,4,0,1,1),
  c(2,1,5,0,1,1),
  c(4,1,1,2,1,2))
```

Cortecosteroid drug sales

- Models with lowest AICc values tend to give slightly better results than the other models.
- AICc comparisons must have the same orders of differencing. But RMSE test set comparisons can involve any models.
- Use the best model available, even if it does not pass all tests.

Cortecosteroid drug sales

```
fit <- h02 %>%
  model(ARIMA(Cost ~ 0 + pdq(3,0,1) + PDQ(0,1,2)))
fit %>% forecast %>% autoplot(h02) +
  ylab("H02 Expenditure ($AUD)") + xlab("Year")
```



Outline

ARIMA vs ETS

- Myth that ARIMA models are more general than exponential smoothing.
- Linear exponential smoothing models all special cases of ARIMA models.
- Non-linear exponential smoothing models have no equivalent ARIMA counterparts.
- Many ARIMA models have no exponential smoothing counterparts.
- ETS models all non-stationary. Models with seasonality or non-damped trend (or both) have two unit roots; all other models have one unit root.

Equivalences

ETS model	ARIMA model	Parameters
ETS(A,N,N)	ARIMA(0,1,1)	$\theta_1 = \alpha - 1$
ETS(A,A,N)	ARIMA(0,2,2)	$\theta_1 = \alpha + \beta - 2$
		$\theta_2 = 1 - \alpha$
ETS(A,A,N)	ARIMA(1,1,2)	$\phi_1 = \phi$
		$\theta_1 = \alpha + \phi\beta - 1 - \phi$
		$\theta_2 = (1 - \alpha)\phi$
ETS(A,N,A)	ARIMA(0,0,m)(0,1,0)$_m$	
ETS(A,A,A)	ARIMA(0,1,$m+1$)(0,1,0)$_m$	
ETS(A,A,A)	ARIMA(1,0,$m+1$)(0,1,0)$_m$	

Your turn

For the `fma::condmilk` series:

- Do the data need transforming? If so, find a suitable transformation.
- Are the data stationary? If not, find an appropriate differencing which yields stationary data.
- Identify a couple of ARIMA models that might be useful in describing the time series.
- Which of your models is the best according to their AIC values?
- Estimate the parameters of your best model and do diagnostic testing on the residuals. Do the residuals resemble white noise? If not, try to find another ARIMA model which fits better.
- Forecast the next 24 months of data using your preferred model.