# STAT481/581: Introduction to Time Series Analysis

Dynamic regression models

OTexts.org/fpp3/

# Outline

# Outline

# Regression with ARIMA errors

## Regression models

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t,$$

- $y_t$ modeled as function of $k$ explanatory variables $x_{1,t}, \dots, x_{k,t}$.
- In regression, we assume that $\varepsilon_t$ was WN.
- Now we want to allow $\varepsilon_t$ to be autocorrelated.

## Example: ARIMA(1,1,1) errors

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$
$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

where $\varepsilon_t$ is white noise.

# Residuals and errors

**Example: $\eta_t$ = ARIMA(1,1,1)**

$$y_t = \beta_0 + \beta_1 x_{1,t} + \dots + \beta_k x_{k,t} + \eta_t,$$
$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

- Be careful in distinguishing $\eta_t$ from $\varepsilon_t$.
- Only the errors $\varepsilon_t$ are assumed to be white noise.
- In ordinary regression, $\eta_t$ is assumed to be white noise and so $\eta_t = \varepsilon_t$.

# Estimation

If we minimize $\sum \eta_t^2$ (by using ordinary regression):

1. Estimated coefficients $\hat{\beta}_0, \dots, \hat{\beta}_k$ are no longer optimal as some information ignored;
2. Statistical tests associated with the model (e.g., t-tests on the coefficients) are incorrect.
3. *p*-values for coefficients usually too small ("spurious regression").
4. AIC of fitted models misleading.

- Minimizing $\sum \varepsilon_t^2$ avoids these problems.
- Maximizing likelihood similar to minimizing $\sum \varepsilon_t^2$.

# Stationarity

## Regression with ARMA errors

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

where $\eta_t$ is an ARMA process.

- All variables in the model must be stationary.
- If we estimate the model while any of these are non-stationary, the estimated coefficients can be incorrect.
- Difference variables until all stationary.
- If necessary, apply same differencing to all variables.

# Stationarity

**Model with ARIMA(1,1,1) errors**

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$
$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

**Equivalent to model with ARIMA(1,0,1) errors**

$$y'_t = \beta_1 x'_{1,t} + \cdots + \beta_k x'_{k,t} + \eta'_t,$$
$$(1 - \phi_1 B)\eta'_t = (1 + \theta_1 B)\varepsilon_t,$$

where $y'_t = y_t - y_{t-1}$, $x'_{t,i} = x_{t,i} - x_{t-1,i}$ and $\eta'_t = \eta_t - \eta_{t-1}$.

# Regression with ARIMA errors

Any regression with an ARIMA error can be rewritten as a regression with an ARMA error by differencing all variables with the same differencing operator as in the ARIMA model.

## Original data

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t$$

$$\text{where} \quad \phi(B)(1-B)^d \eta_t = \theta(B)\varepsilon_t$$

## After differencing all variables

$$y'_t = \beta_1 x'_{1,t} + \cdots + \beta_k x'_{k,t} + \eta'_t.$$

$$\text{where} \quad \phi(B)\eta_t = \theta(B)\varepsilon_t$$

$$\text{and} \quad y'_t = (1-B)^d y_t$$

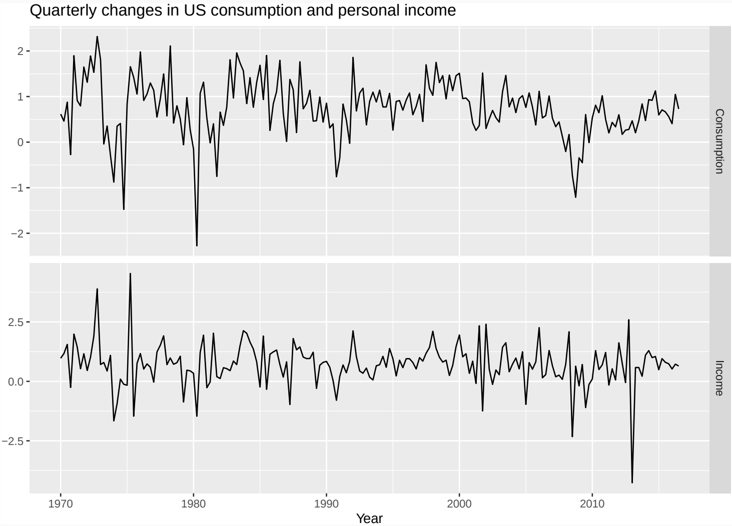
# Model selection

- Fit regression model with automatically selected ARIMA errors. (R will take care of differencing before estimation.)
- Check that $\varepsilon_t$ series looks like white noise.

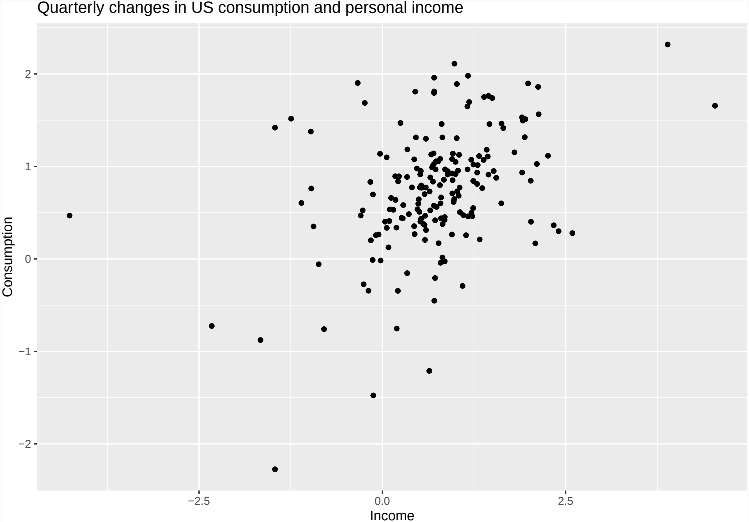
## Selecting predictors

- AICc can be calculated for final model.
- Repeat procedure for all subsets of predictors to be considered, and select model with lowest AICc value.

# US personal consumption and income



# US personal consumption and income



# US personal consumption and income

- No need for transformations or further differencing.
- Increase in income does not necessarily translate into instant increase in consumption (e.g., after the loss of a job, it may take a few months for expenses to be reduced to allow for the new circumstances). We will ignore this for now.

# US personal consumption and income

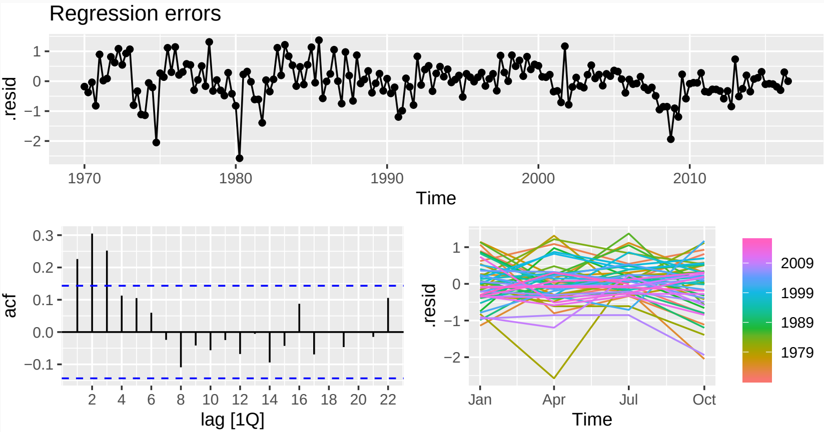
```
fit <- us_change %>% model(ARIMA(Consumption ~ Income))
report(fit)
```

```
## Series: Consumption
## Model: LM w/ ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1     ma2  Income  intercept
##       0.6922  -0.5758  0.1984  0.2028     0.5990
## s.e.  0.1159   0.1301  0.0756  0.0461     0.0884
##
## sigma^2 estimated as 0.3219:  log likelihood=-156.9
## AIC=325.9   AICc=326.4   BIC=345.3
```

Write down the equations for the fitted model.

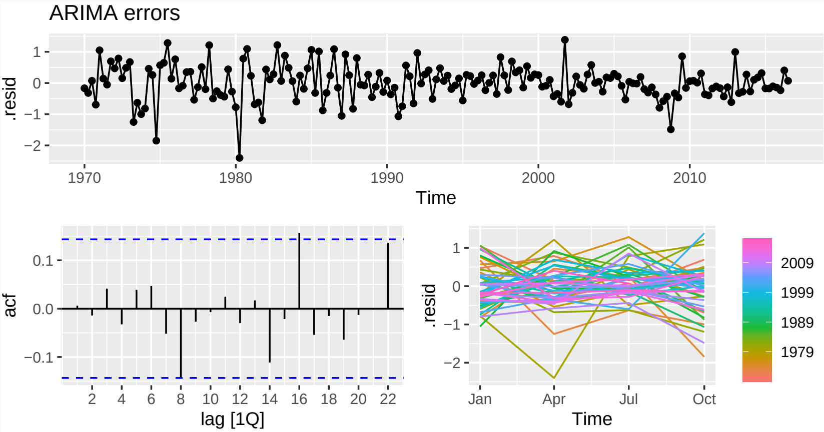
# US personal consumption and income

```r
residuals(fit, type='regression') %>%
  gg_tsdisplay(.resid) + ggtitle("Regression errors")
```



# US personal consumption and income

```
residuals(fit, type='response') %>%
  gg_tsdisplay(.resid) + ggtitle("ARIMA errors")
```



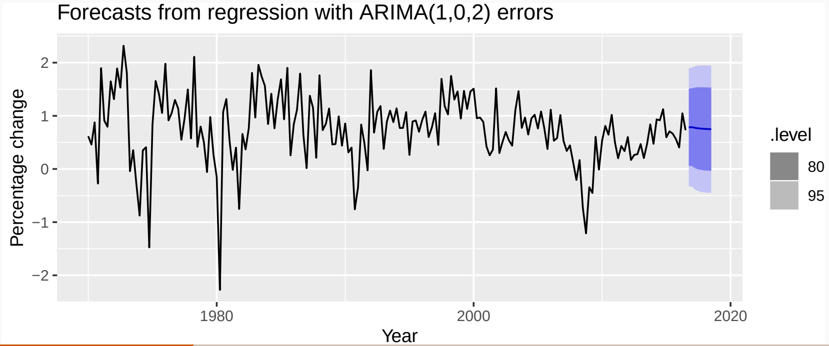
# US personal consumption and income

```r
augment(fit) %>%
  features(.resid, ljung_box, dof = 5, lag = 12)
```

```
## # A tibble: 1 x 3
##   .model                            lb_stat lb_pvalue
##   <chr>                               <dbl>     <dbl>
## 1 ARIMA(Consumption ~ Income)          6.35     0.500
```

# US personal consumption and income

```
us_change_future <- new_data(us_change, 8) %>%
  mutate(Income = mean(us_change$Income))
forecast(fit, new_data = us_change_future) %>%
  autoplot(us_change) +
  labs(x = "Year", y = "Percentage change",
       title = "Forecasts from regression with ARIMA(1,0,2) errors")
```



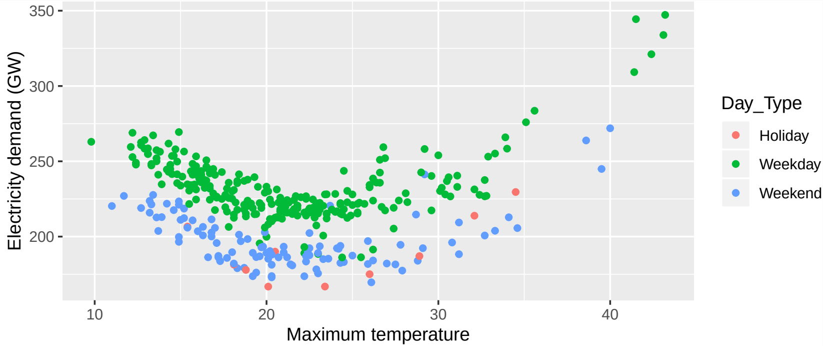
# Forecasting

- To forecast a regression model with ARIMA errors, we need to forecast the regression part of the model and the ARIMA part of the model and combine the results.
- Some predictors are known into the future (e.g., time, dummies).
- Separate forecasting models may be needed for other predictors.
- Forecast intervals ignore the uncertainty in forecasting the predictors.

# Daily electricity demand

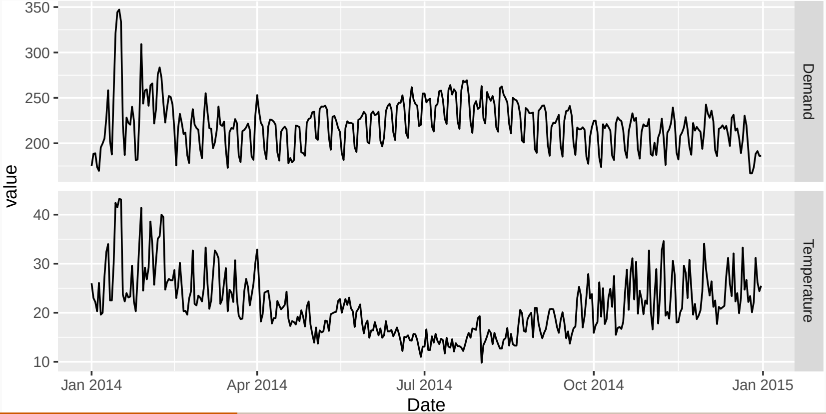
Model daily electricity demand as a function of temperature using quadratic regression with ARMA errors.

```
vic_elec_daily %>%
  ggplot(aes(x=Temperature, y=Demand, colour=Day_Type)) +
  geom_point() +
  labs(x = "Maximum temperature", y = "Electricity demand (GW)")
```



# Daily electricity demand

```r
vic_elec_daily %>%
  gather("var", "value", Demand, Temperature) %>%
  ggplot(aes(x = Date, y = value)) + geom_line() +
  facet_grid(vars(var), scales = "free_y")
```



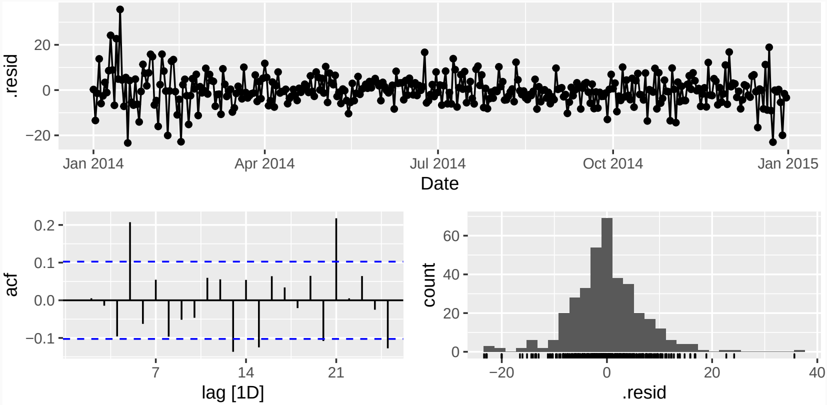
# Daily electricity demand

```
fit <- vic_elec_daily %>%
  model(ARIMA(Demand ~ Temperature + I(Temperature^2) +
                (Day_Type=="Weekday")))
report(fit)
```

```
## Series: Demand
## Model: LM w/ ARIMA(2,1,2)(0,0,2)[7] errors
##
## Coefficients:
##          ar1      ar2      ma1     ma2    sma1    sma2
##       1.1521  -0.2750  -1.3851  0.4071  0.1589  0.3103
## s.e.  0.6265   0.4812   0.6082  0.5804  0.0591  0.0538
##       Temperature  I(Temperature^2)
##            -7.947            0.1865
## s.e.        0.492            0.0097
##       Day_Type == "Weekday"TRUE
##                          31.825
## s.e.                      1.019
##
```

# Daily electricity demand

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



# Daily electricity demand

```r
augment(fit) %>%
  features(.resid, ljung_box, dof = 8, lag = 14)
```

```
## # A tibble: 1 x 3
##   .model                                           lb_stat lb_pvalue
##   <chr>                                              <dbl>     <dbl>
## 1 "ARIMA(Demand ~ Temperature + I(Tempera~    38.1   1.05e-6
```

# Daily electricity demand

```r
# Forecast one day ahead
vic_next_day <- new_data(vic_elec_daily, 1) %>%
  mutate(Temperature = 26, Day_Type = "Holiday")
forecast(fit, vic_next_day)
```

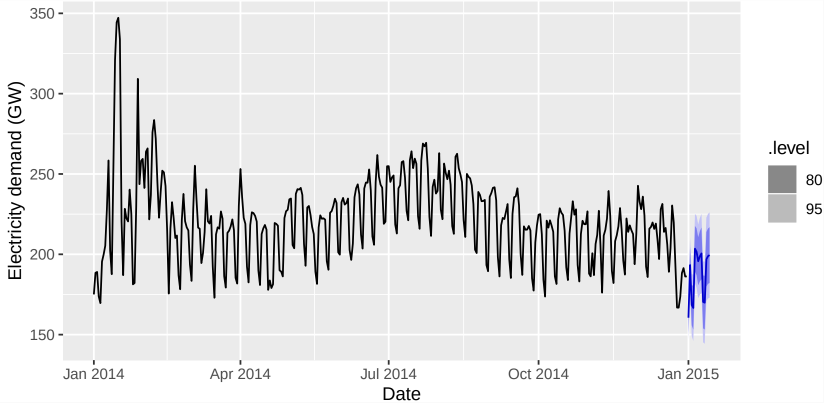
```
## # A fable: 1 x 6 [1D]
## # Key:     .model [1]
##   .model Date       Demand .distribution Temperature
##   <chr>  <date>      <dbl> <dist>              <dbl>
## 1 "ARIM~ 2015-01-01   161. N(161, 49)             26
## # ... with 1 more variable: Day_Type <chr>
```

# Daily electricity demand

```
vic_elec_future <- new_data(vic_elec_daily, 14) %>%
  mutate(
    Temperature = 26,
    Holiday = c(TRUE, rep(FALSE, 13)),
    Day_Type = case_when(
      Holiday ~ "Holiday",
      wday(Date) %in% 2:6 ~ "Weekday",
      TRUE ~ "Weekend"
    )
  )
```

# Daily electricity demand

```
forecast(fit, vic_elec_future) %>%
  autoplot(vic_elec_daily) + ylab("Electricity demand (GW)")
```



# Outline

# Stochastic & deterministic trends

## Deterministic trend

$$y_t = \beta_0 + \beta_1 t + \eta_t$$

where $\eta_t$ is ARMA process.

## Stochastic trend

$$y_t = \beta_0 + \beta_1 t + \eta_t$$

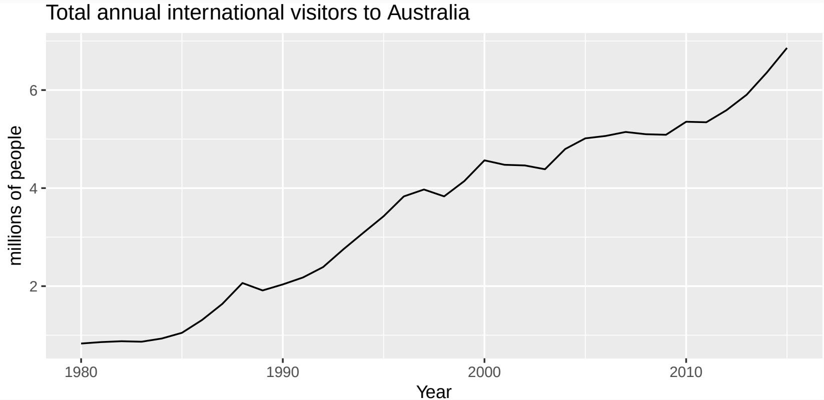
where $\eta_t$ is ARIMA process with $d \geq 1$.

Difference both sides until $\eta_t$ is stationary:

$$y'_t = \beta_1 + \eta'_t$$

where $\eta'_t$ is ARMA process.

# International visitors



# International visitors

### Deterministic trend

```
fit_deterministic <- aus_visitors %>%
  model(Deterministic = ARIMA(value ~ trend() + pdq(d = 0)))
report(fit_deterministic)
```

```
## Series: value
## Model: LM w/ ARIMA(2,0,0) errors
##
## Coefficients:
##          ar1      ar2  trend()  intercept
##        1.113  -0.3805   0.1710     0.4156
## s.e.  0.160   0.1585   0.0088     0.1897
##
## sigma^2 estimated as 0.02979:  log likelihood=13.6
## AIC=-17.2   AICc=-15.2   BIC=-9.28
```

# International visitors

$$\begin{aligned}
y_t &= 0.42 + 0.17t + \eta_t \\
\eta_t &= 1.11\eta_{t-1} - 0.38\eta_{t-2} + \varepsilon_t \\
\varepsilon_t &\sim \text{NID}(0, 0.0298).
\end{aligned}$$

# International visitors

### Stochastic trend

```
fit_stochastic <- aus_visitors %>%
  model(Stochastic = ARIMA(value ~ pdq(d=1)))
report(fit_stochastic)
```

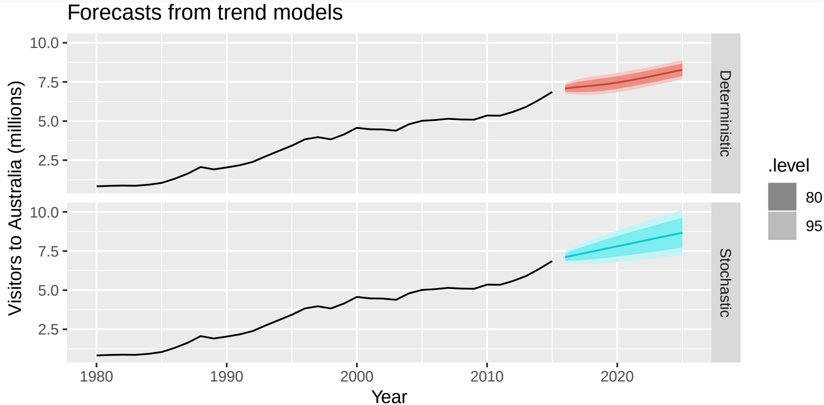
```
## Series: value
## Model: ARIMA(0,1,1) w/ drift
##
## Coefficients:
##          ma1  constant
##       0.3006    0.1735
## s.e.  0.1647    0.0390
##
## sigma^2 estimated as 0.03376:  log likelihood=10.62
## AIC=-15.24   AICc=-14.46   BIC=-10.57
```

$$y_t - y_{t-1} = 0.17 + \varepsilon_t$$

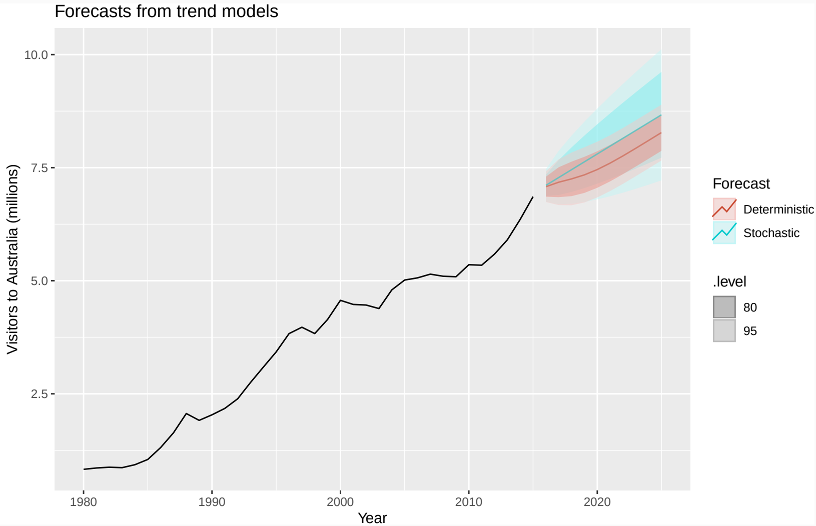
$$y_t = y_0 + 0.17t + \eta_t$$

$$\eta_t = \eta_{t-1} + 0.30\varepsilon_{t-1} + \varepsilon_t$$

# International visitors



# International visitors



# Forecasting with trend

- Point forecasts are almost identical, but prediction intervals differ.
- Stochastic trends have much wider prediction intervals because the errors are non-stationary.
- Be careful of forecasting with deterministic trends too far ahead.

# Outline

# Dynamic harmonic regression

## Combine Fourier terms with ARIMA errors

### Advantages

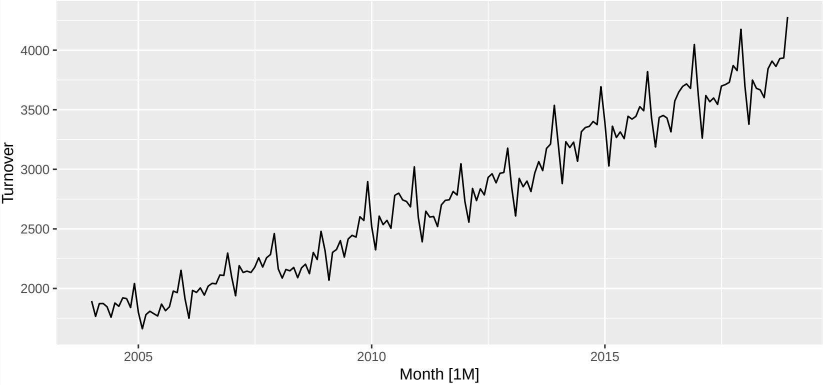
- it allows any length seasonality;
- for data with more than one seasonal period, you can include Fourier terms of different frequencies;
- the seasonal pattern is smooth for small values of $K$ (but more wiggly seasonality can be handled by increasing $K$);
- the short-term dynamics are easily handled with a simple ARMA error.

### Disadvantages

- seasonality is assumed to be fixed

# Eating-out expenditure

```
aus_cafe <- aus_retail %>% filter(
    Industry == "Cafes, restaurants and takeaway food services",
    year(Month) %in% 2004:2018
  ) %>% summarise(Turnover = sum(Turnover))
aus_cafe %>% autoplot(Turnover)
```

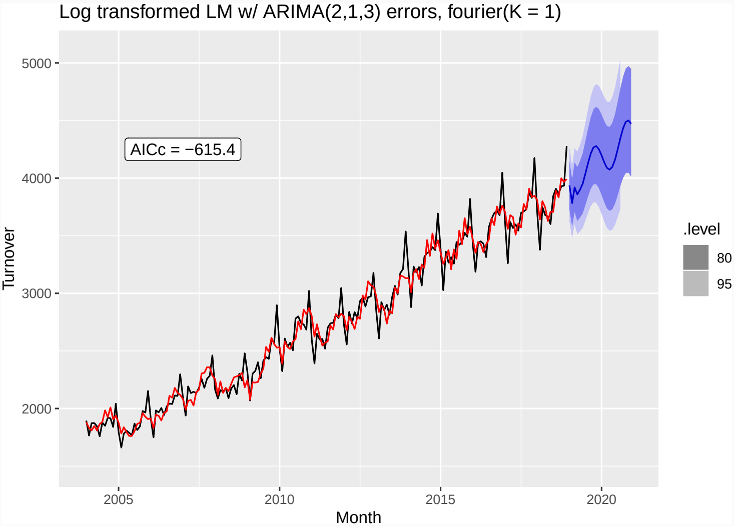


# Eating-out expenditure

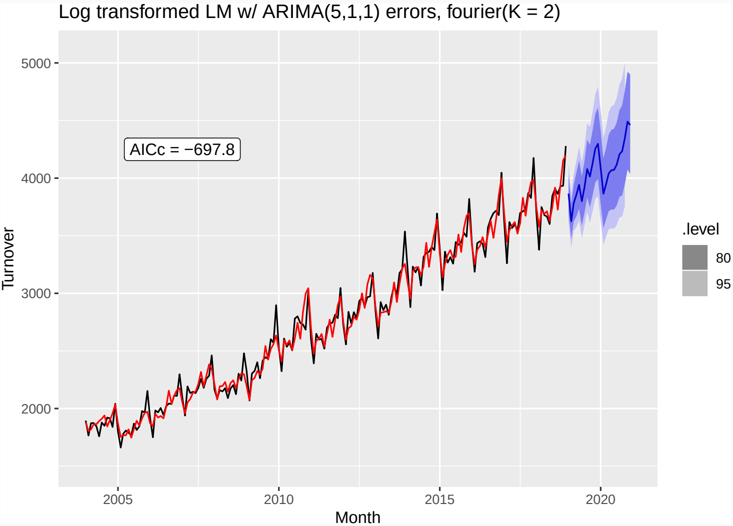
```
fit <- aus_cafe %>% model(
    K = 1 = ARIMA(log(Turnover) ~ fourier(K = 1) + PDQ(0,0,0)),
    K = 2 = ARIMA(log(Turnover) ~ fourier(K = 2) + PDQ(0,0,0)),
    K = 3 = ARIMA(log(Turnover) ~ fourier(K = 3) + PDQ(0,0,0)),
    K = 4 = ARIMA(log(Turnover) ~ fourier(K = 4) + PDQ(0,0,0)),
    K = 5 = ARIMA(log(Turnover) ~ fourier(K = 5) + PDQ(0,0,0)),
    K = 6 = ARIMA(log(Turnover) ~ fourier(K = 6) + PDQ(0,0,0)))
glance(fit)
```

| .model | sigma2 | AIC | AICc | BIC |
|---|---|---|---|---|
| K = 1 | 0.0017 | -616.5 | -615.4 | -587.8 |
| K = 2 | 0.0011 | -699.7 | -697.8 | -661.5 |
| K = 3 | 0.0008 | -763.2 | -761.3 | -725.0 |
| K = 4 | 0.0005 | -821.6 | -818.2 | -770.6 |
| K = 5 | 0.0003 | -919.5 | -916.9 | -874.8 |
| K = 6 | 0.0003 | -920.1 | -917.5 | -875.4 |

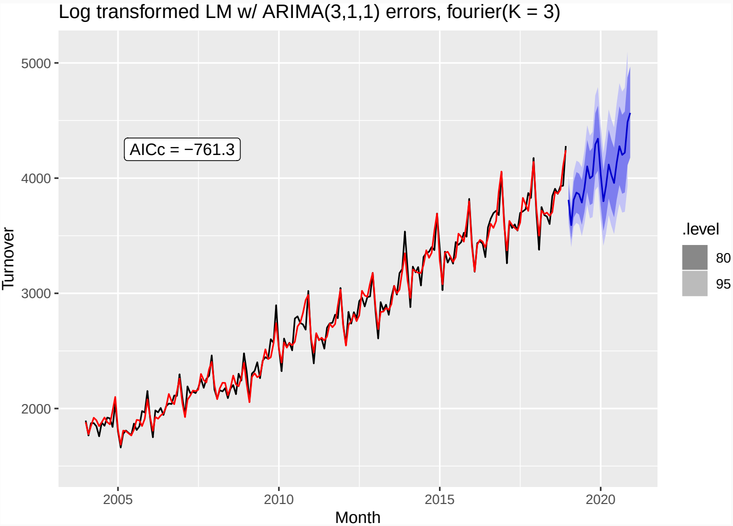
# Eating-out expenditure



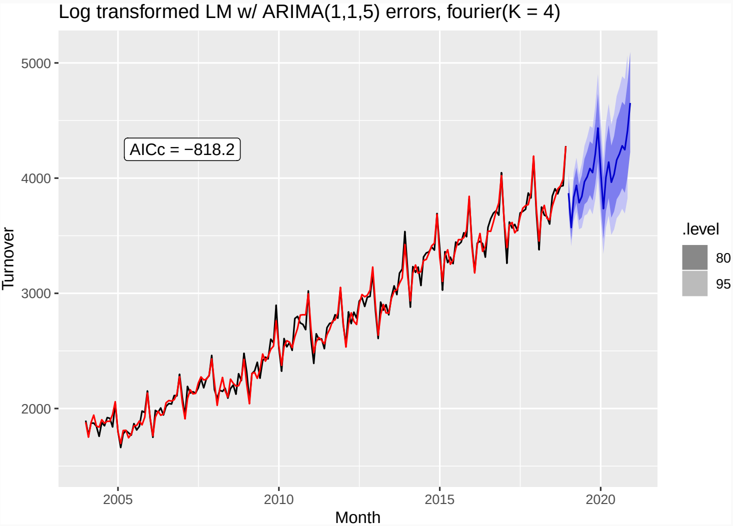
# Eating-out expenditure



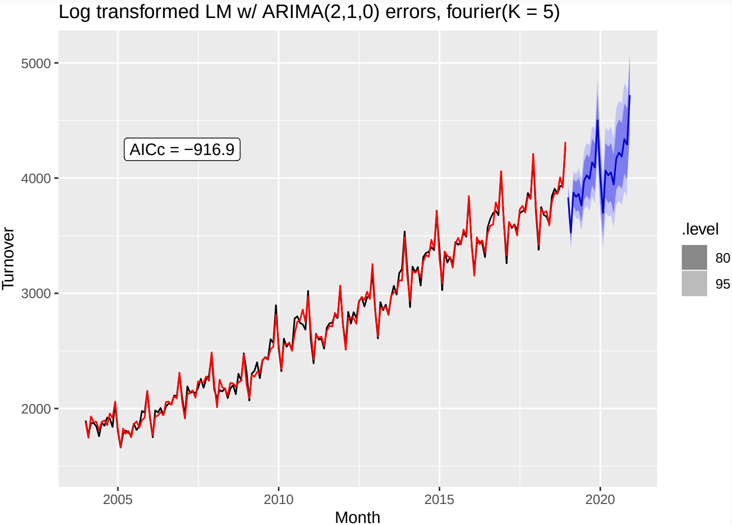
# Eating-out expenditure



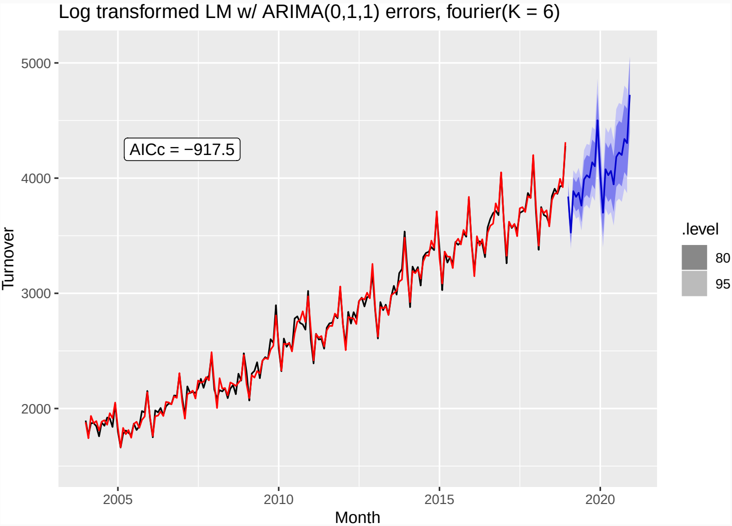
# Eating-out expenditure



# Eating-out expenditure



# Eating-out expenditure



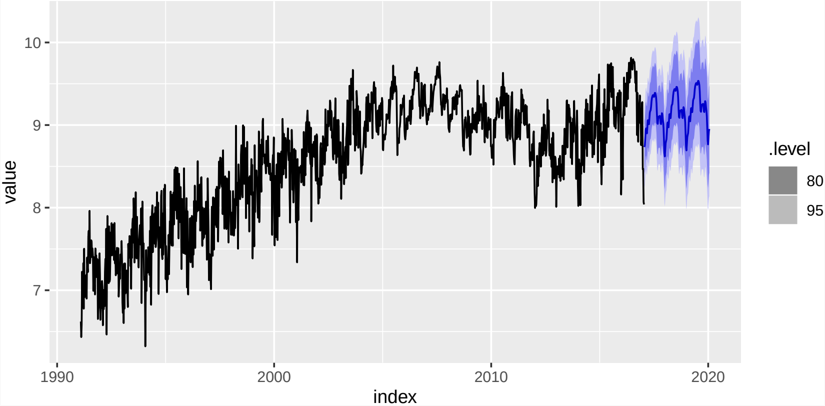
# Example: weekly gasoline products

```
gasoline <- as_tsibble(fpp2::gasoline)
fit <- gasoline %>% model(ARIMA(value ~ fourier(K = 13) + PDQ(0,0,0)))
report(fit)
```

```
## Series: value
## Model: LM w/ ARIMA(0,1,1) errors
##
## Coefficients:
##           ma1  fourier(K = 13)C1_52  fourier(K = 13)S1_52
##       -0.8934               -0.1121               -0.2300
## s.e.   0.0132                0.0123                0.0122
##       fourier(K = 13)C2_52  fourier(K = 13)S2_52
##                     0.0420                0.0317
## s.e.                0.0099                0.0099
##       fourier(K = 13)C3_52  fourier(K = 13)S3_52
##                     0.0832                0.0346
## s.e.                0.0094                0.0094
##       fourier(K = 13)C4_52  fourier(K = 13)S4_52
##                     0.0185                0.0398
## s.e.                0.0092                0.0092
##       fourier(K = 13)C5_52  fourier(K = 13)S5_52
##                    -0.0315                0.0009
## s.e.                0.0091                0.0091
##       fourier(K = 13)C6_52  fourier(K = 13)S6_52
```

# Example: weekly gasoline products

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



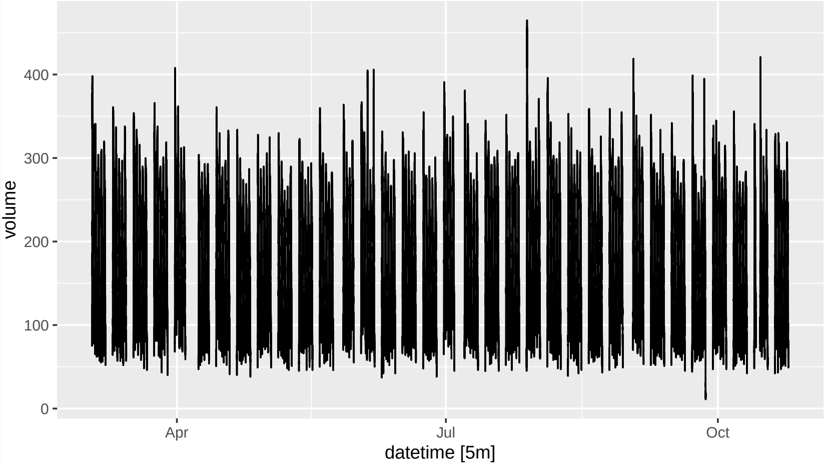
# 5-minute call centre volume

```
(calls <- read_tsv("http://robjhyndman.com/data/callcenter.txt") %>%
  gather("date", "volume", -X1) %>% transmute(
    time = X1, date = as.Date(date, format = "%d/%m/%Y"),
    datetime = as_datetime(date) + time, volume) %>%
  as_tsibble(index = datetime))
```

```
## # A tsibble: 27,716 x 4 [5m] <UTC>
##    time   date       datetime            volume
##    <time> <date>     <dttm>               <dbl>
##  1 07:00  2003-03-03 2003-03-03 07:00:00    111
##  2 07:05  2003-03-03 2003-03-03 07:05:00    113
##  3 07:10  2003-03-03 2003-03-03 07:10:00     76
##  4 07:15  2003-03-03 2003-03-03 07:15:00     82
##  5 07:20  2003-03-03 2003-03-03 07:20:00     91
##  6 07:25  2003-03-03 2003-03-03 07:25:00     87
##  7 07:30  2003-03-03 2003-03-03 07:30:00     75
##  8 07:35  2003-03-03 2003-03-03 07:35:00     89
##  9 07:40  2003-03-03 2003-03-03 07:40:00     99
## 10 07:45  2003-03-03 2003-03-03 07:45:00    125
```

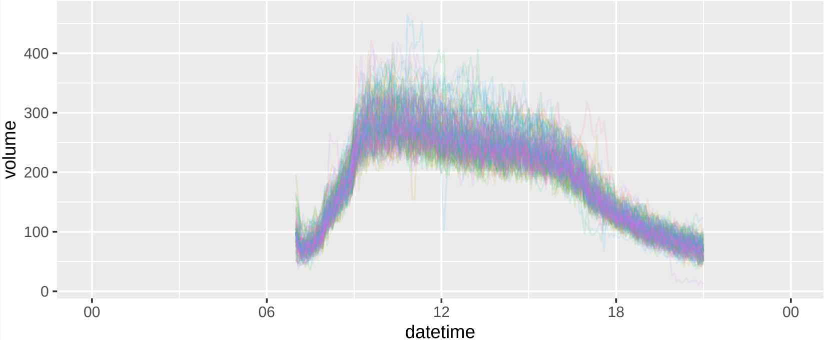
# 5-minute call centre volume

```
calls %>% fill_gaps() %>% autoplot(volume)
```



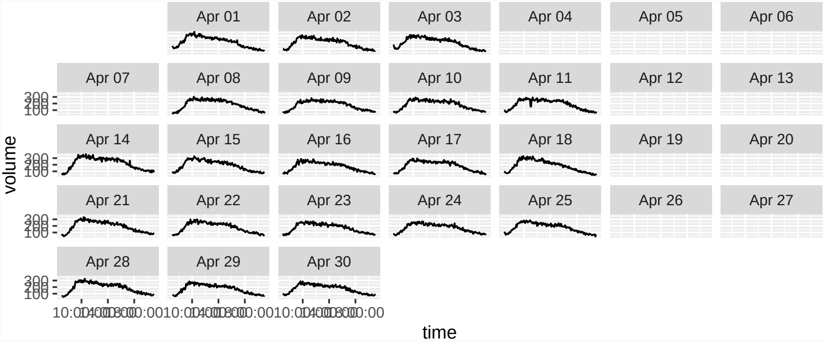
# 5-minute call centre volume

```
calls %>% fill_gaps() %>%
  gg_season(volume, period = "day", alpha = 0.1) +
  guides(colour = FALSE)
```



# 5-minute call centre volume

```r
library(sugrrants)
calls %>% filter(month(date, label = TRUE) == "Apr") %>%
  ggplot(aes(x = time, y = volume)) +
  geom_line() + facet_calendar(date)
```



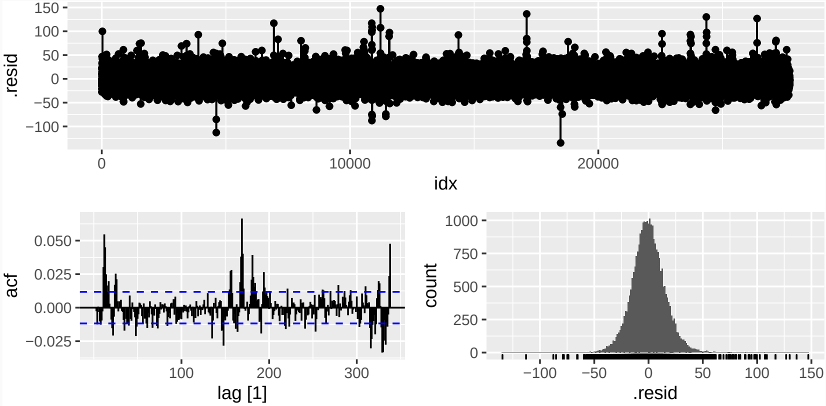
# 5-minute call centre volume

```
calls_mdl <- calls %>%
  mutate(idx = row_number()) %>%
  update_tsibble(index = idx)
fit <- calls_mdl %>%
  model(ARIMA(volume ~ fourier(169, K = 10) + pdq(d=0) + PDQ(0,0,0)))
report(fit)
```

```
## Series: volume
## Model: LM w/ ARIMA(1,0,3) errors
##
## Coefficients:
##          ar1      ma1      ma2      ma3
##       0.9894  -0.7383  -0.0333  -0.0282
## s.e.  0.0010   0.0061   0.0075   0.0060
##       fourier(169, K = 10)C1_169
##                         -79.0702
## s.e.                      0.7001
##       fourier(169, K = 10)S1_169
##                          55.2985
## s.e.                      0.7006
##       fourier(169, K = 10)C2_169
##                         -32.3615
## s.e.                      0.3784
##       fourier(169, K = 10)S2_169
```

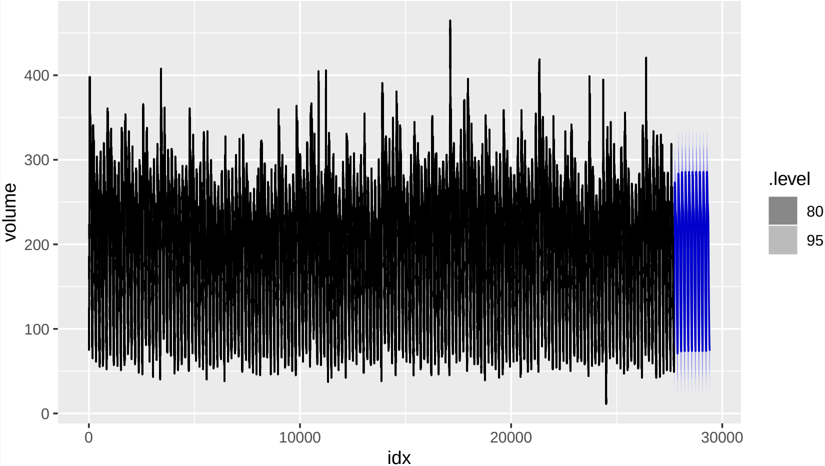
# 5-minute call centre volume

```
augment(fit) %>%
  gg_tsdisplay(.resid, plot_type = "histogram", lag_max = 338)
```



# 5-minute call centre volume

```
fit %>% forecast(h = 1690) %>%
  autoplot(calls_mdl)
```



# Outline

# Lagged predictors

Sometimes a change in $x_t$ does not affect $y_t$ instantaneously

- $y_t$ = sales, $x_t$ = advertising.
- $y_t$ = stream flow, $x_t$ = rainfall.
- $y_t$ = size of herd, $x_t$ = breeding stock.

- These are dynamic systems with input ($x_t$) and output ($y_t$).
- $x_t$ is often a leading indicator.
- There can be multiple predictors.

# Lagged predictors

The model include present and past values of predictor: $x_t, x_{t-1}, x_{t-2}, \ldots$.

$$y_t = a + \nu_0 x_t + \nu_1 x_{t-1} + \cdots + \nu_k x_{t-k} + \eta_t$$

where $\eta_t$ is an ARIMA process.

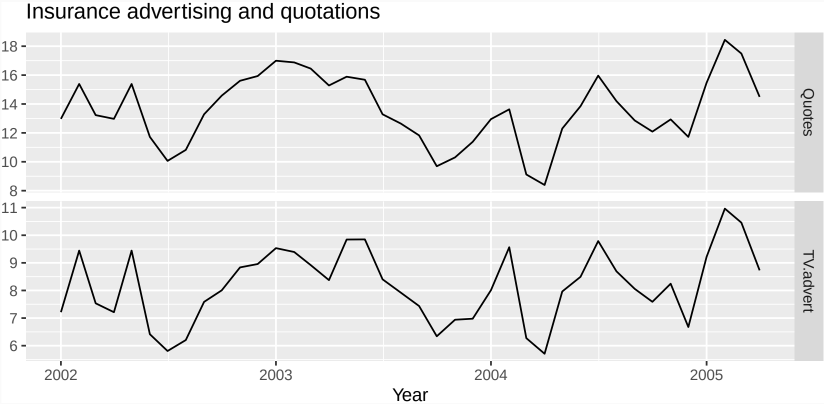
# Lagged predictors

**Rewrite model as**

$$\begin{aligned} y_t &= a + (\nu_0 + \nu_1 B + \nu_2 B^2 + \cdots + \nu_k B^k)x_t + \eta_t \\ &= a + \nu(B)x_t + \eta_t. \end{aligned}$$

- $\nu(B)$ is called a *transfer function* since it describes how change in $x_t$ is transferred to $y_t$.
- $x$ can influence $y$, but $y$ is not allowed to influence $x$.

# Example: Insurance quotes and TV adverts



# Example: Insurance quotes and TV adverts

```r
fit <- insurance %>%
  # Restrict data so models use same fitting period
  mutate(Quotes = c(NA,NA,NA,Quotes[4:40])) %>%
  # Estimate models
  model(
    ARIMA(Quotes ~ pdq(d = 0) + TV.advert),
    ARIMA(Quotes ~ pdq(d = 0) + TV.advert + lag(TV.advert)),
    ARIMA(Quotes ~ pdq(d = 0) + TV.advert + lag(TV.advert) +
            lag(TV.advert, 2)),
    ARIMA(Quotes ~ pdq(d = 0) + TV.advert + lag(TV.advert) +
            lag(TV.advert, 2) + lag(TV.advert, 3))
  )
```

# Example: Insurance quotes and TV adverts

```r
glance(fit)
```

| Lag order | sigma2 | AIC | AICc | BIC |
|---:|---:|---:|---:|---:|
| 0 | 0.2650 | 66.56 | 68.33 | 75.01 |
| 1 | 0.2094 | 58.09 | 59.85 | 66.53 |
| 2 | 0.2150 | 60.03 | 62.58 | 70.17 |
| 3 | 0.2056 | 60.31 | 64.96 | 73.83 |

# Example: Insurance quotes and TV adverts

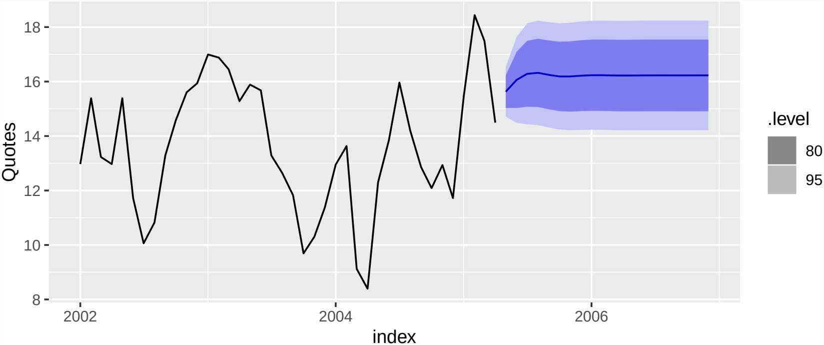
```r
fit <- insurance %>%
  model(ARIMA(Quotes ~ pdq(3, 0, 0) + TV.advert + lag(TV.advert)))
report(fit)
```

```
## Series: Quotes
## Model: LM w/ ARIMA(3,0,0) errors
##
## Coefficients:
##          ar1      ar2     ar3  TV.advert  lag(TV.advert)
##       1.4117  -0.9317  0.3591     1.2564          0.1625
## s.e.  0.1698   0.2545  0.1592     0.0667          0.0591
##       intercept
##          2.0393
## s.e.     0.9931
##
## sigma^2 estimated as 0.2165:  log likelihood=-23.89
## AIC=61.78   AICc=65.28   BIC=73.6
```

$$y_t = 2.04 + 1.26x_t + 0.16x_{t-1} + \eta_t,$$

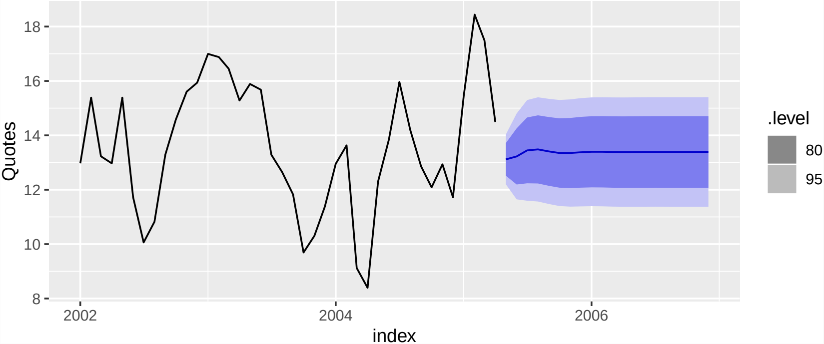
# Example: Insurance quotes and TV adverts

```
advert_a <- new_data(insurance, 20) %>%
  mutate(TV.advert = 10)
forecast(fit, advert_a) %>% autoplot(insurance)
```



# Example: Insurance quotes and TV adverts

```
advert_b <- new_data(insurance, 20) %>%
  mutate(TV.advert = 8)
forecast(fit, advert_b) %>% autoplot(insurance)
```



# Example: Insurance quotes and TV adverts

```
advert_c <- new_data(insurance, 20) %>%
  mutate(TV.advert = 6)
forecast(fit, advert_c) %>% autoplot(insurance)
```