Tidy Time Series & Forecasting in R

9. Dynamic regression

bit.ly/fable2020

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 ε_t was WN.
- Now we want to allow ε_t to be autocorrelated.

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 $x_{1,t}, \ldots, x_{k,t}$.
- In regression, we assume that ε_t was WN.
- Now we want to allow ε_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 ε_t is white noise.

Residuals and errors

Example: η_t = ARIMA(1,1,1)

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

Residuals and errors

Example: η_t = ARIMA(1,1,1)

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

- Be careful in distinguishing η_t from ε_t.
- Only the errors ε_t are assumed to be white noise.
- In ordinary regression, η_t is assumed to be white noise and so $\eta_t = \varepsilon_t$.
- If η_t is non-stationary, the model is equivalent to regressing differenced y_t on differenced $x_{1,t}, \dots, x_{k,t}$.

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.

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.

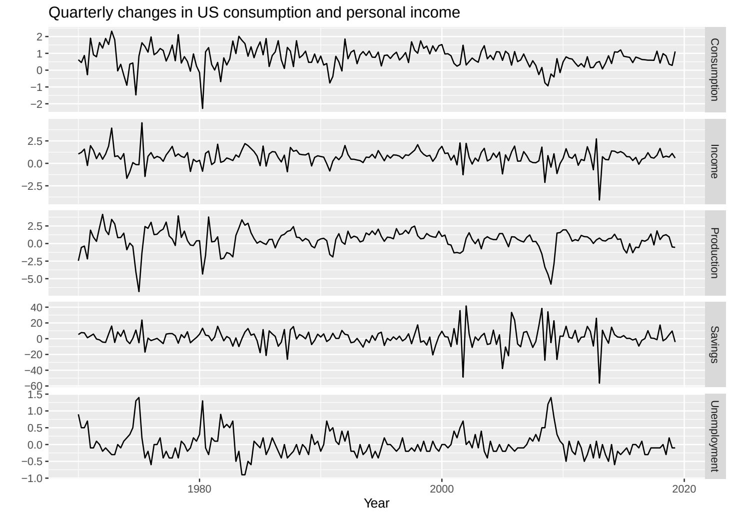
- Minimizing $\sum \varepsilon_t^2$ avoids these problems.
- Maximizing likelihood similar to minimizing $\sum \varepsilon_t^2$.

Selecting predictors

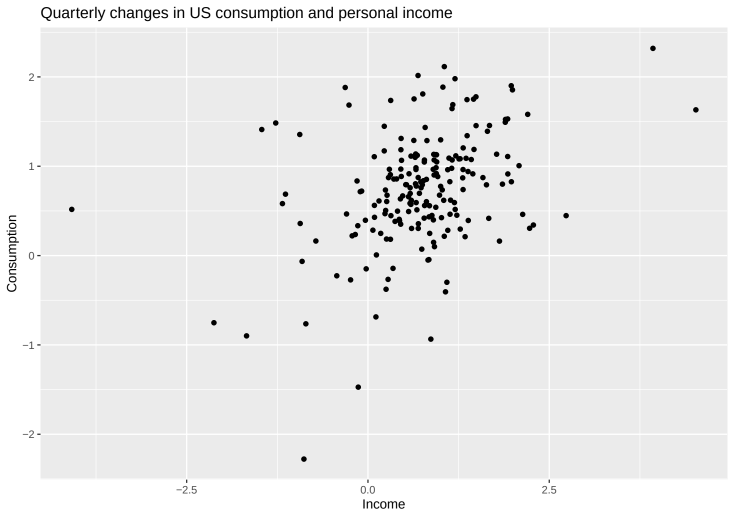
$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$
$$\eta_t \sim \text{ARIMA}(p, d, q)$$

- 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

```r
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
##       0.7070  -0.6172  0.2066  0.1976     0.5949
## s.e.  0.1068   0.1218  0.0741  0.0462     0.0850
##
## sigma^2 estimated as 0.3113:  log likelihood=-163.04
## AIC=338.07   AICc=338.51   BIC=357.8
```

US personal consumption and income

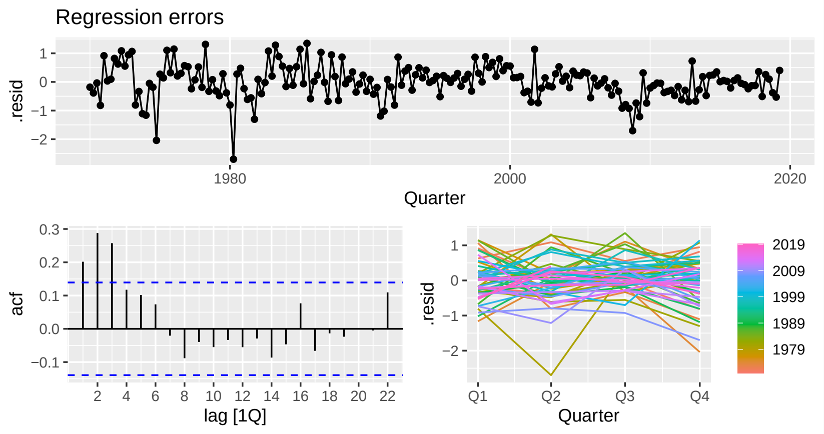
```r
fit <- us_change %>% model(ARIMA(Consumption ~ Income))
report(fit)
```

```
## Series: Consumption
## Model: LM w/ ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1     ma2  Income  intercept
##       0.7070  -0.6172  0.2066  0.1976     0.5949
## s.e.  0.1068   0.1218  0.0741  0.0462     0.0850
##
## sigma^2 estimated as 0.3113:  log likelihood=-163.04
## AIC=338.07   AICc=338.51   BIC=357.8
```

Write down the equations for the fitted model.

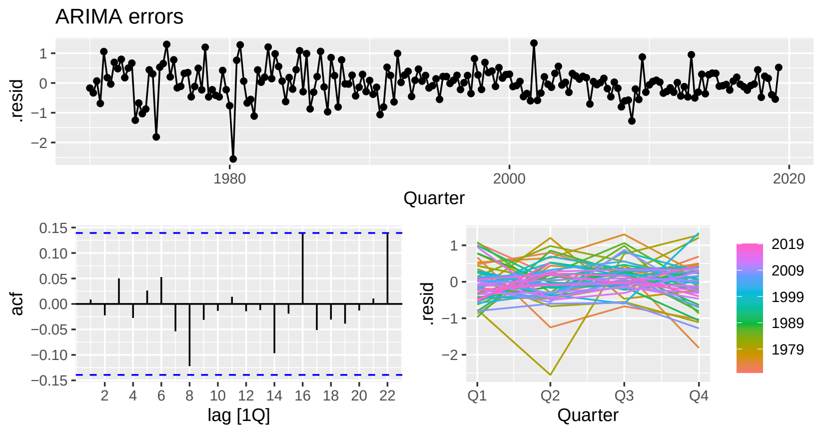
US personal consumption and income

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



US personal consumption and income

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



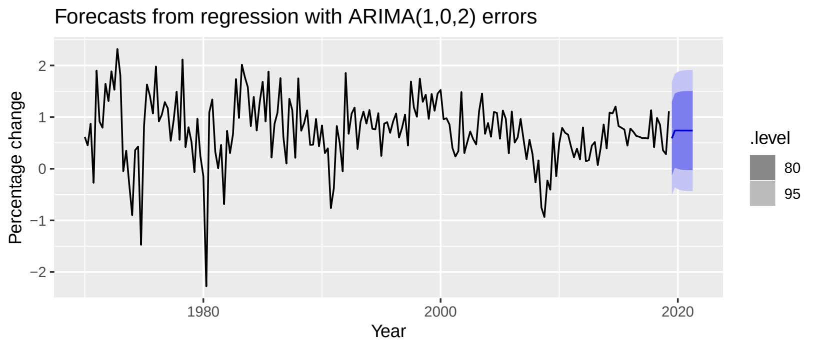
US personal consumption and income

```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)         5.54     0.595
```

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"
  )
```



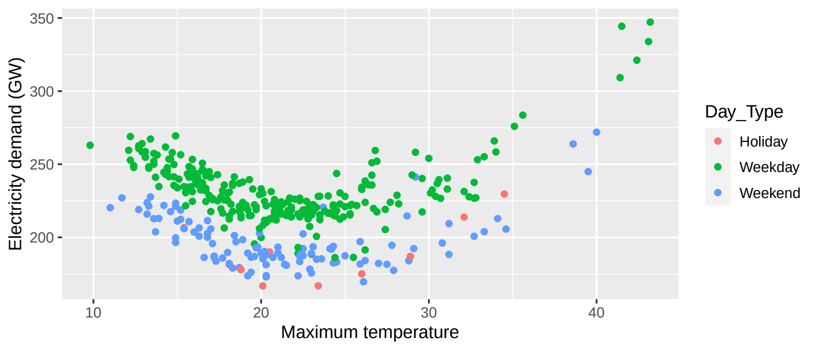
Forecasting

- 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

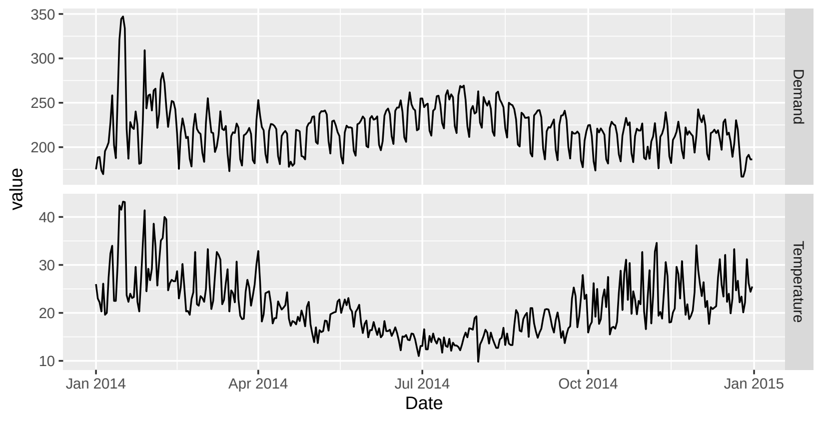
Model daily electricity demand as a function of temperature using quadratic regression with ARMA errors.

```
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

```
vic_elec_daily %>%
  pivot_longer(c(Demand, Temperature)) %>%
  ggplot(aes(x = Date, y = value)) + geom_line() +
  facet_grid(vars(name), scales = "free_y")
```



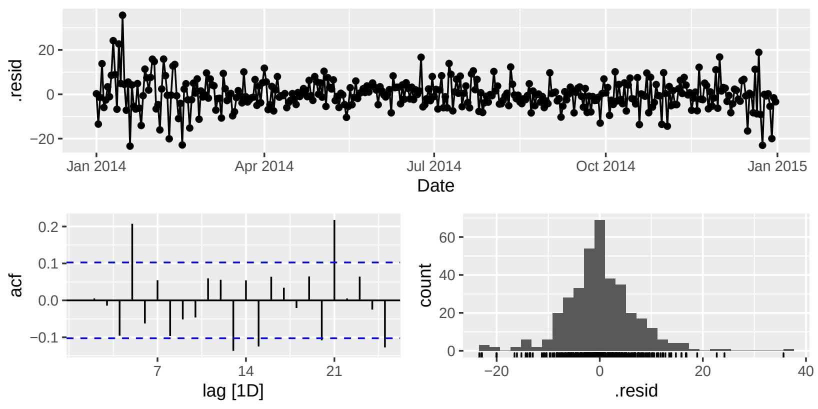
Daily electricity demand

```r
fit <- vic_elec_daily %>%
  model(fit = 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  Temperature
##       1.1521  -0.2750  -1.3851  0.4071  0.1589  0.3103      -7.9467
## s.e.  0.6265   0.4812   0.6082  0.5804  0.0591  0.0538       0.4920
##       I(Temperature^2)  Day_Type == "Weekday"TRUE
##                 0.1865                    31.8245
## s.e.            0.0097                     1.0189
##
## sigma^2 estimated as 48.82:  log likelihood=-1220.48
## AIC=2460.96   AICc=2461.58   BIC=2499.93
```

Daily electricity demand

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



Daily electricity demand

```r
augment(fit) %>%
  features(.resid, ljung_box, dof = 9, lag = 14)
```

```
## # A tibble: 1 x 3
##   .model lb_stat   lb_pvalue
##   <chr>    <dbl>       <dbl>
## 1 fit       38.1 0.000000354
```

Daily electricity demand

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

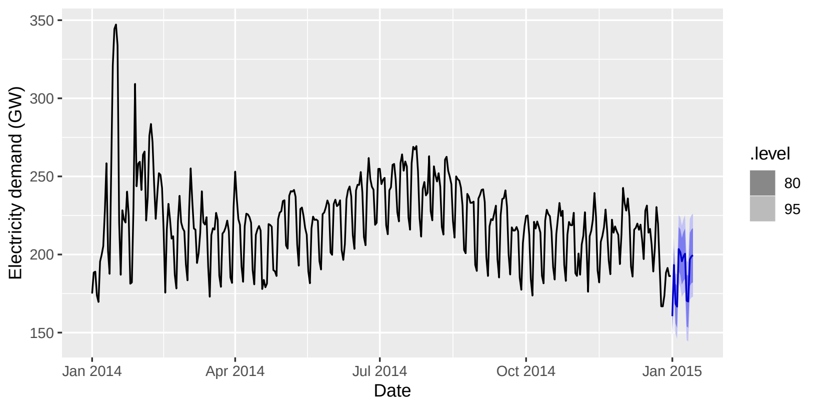
```
## # A fable: 1 x 6 [1D]
## # Key:     .model [1]
##   .model Date       Demand .distribution Temperature Day_Type
##   <chr>  <date>      <dbl> <dist>              <dbl> <chr>
## 1 fit    2015-01-01   161. N(161, 49)             26 Holiday
```

Daily electricity demand

```r
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

Lab Session 18

Repeat the daily electricity example, but instead of using a quadratic function of temperature, use a piecewise linear function with the “knot” around 20 degrees Celsius (use predictors `Temperature` & `Temp2`). How can you optimize the choice of knot?

The data can be created as follows.

```
vic_elec_daily <- vic_elec %>%
  filter(year(Time) == 2014) %>%
  index_by(Date = date(Time)) %>%
  summarise(
    Demand = sum(Demand) / 1e3,
    Temperature = max(Temperature),
    Holiday = any(Holiday)
  ) %>%
  mutate(
    Temp2 = I(pmax(Temperature - 20, 0)),
    Day_Type = case_when(
      Holiday ~ "Holiday",
      wday(Date) %in% 2:6 ~ "Weekday",
      TRUE ~ "Weekend"))
```

Outline

Stochastic & deterministic trends

Deterministic trend

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

where η_t is ARMA process.

Stochastic & deterministic trends

Deterministic trend

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

where η_t is ARMA process.

Stochastic trend

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

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

Stochastic & deterministic trends

Deterministic trend

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

where η_t is ARMA process.

Stochastic trend

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

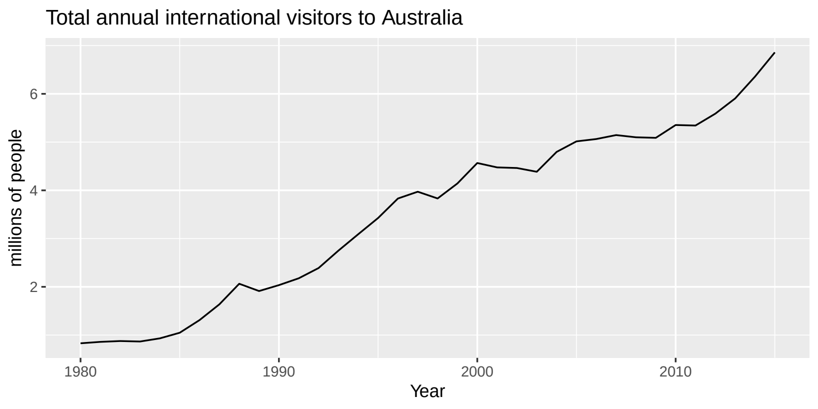
where η_t is ARIMA process with $d \geq 1$.

Difference both sides until η_t is stationary:

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

where η'_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
##       1.1127  -0.3805   0.1710     0.4156
## s.e.  0.1600   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

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
##       1.1127  -0.3805   0.1710     0.4156
## s.e.  0.1600   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
```

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

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
```

International visitors

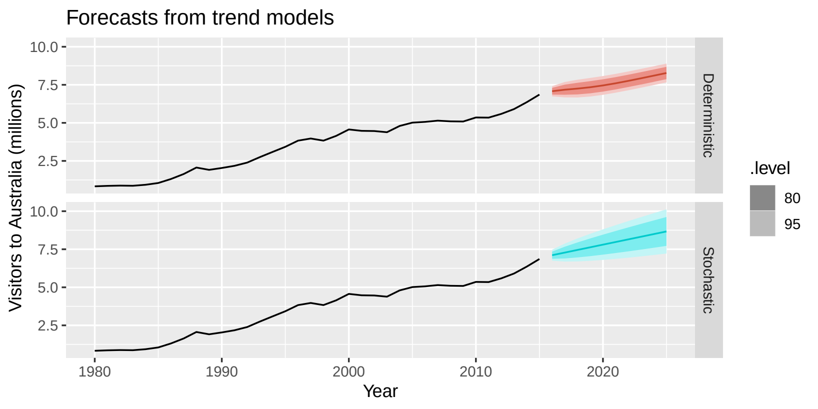
Stochastic trend

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

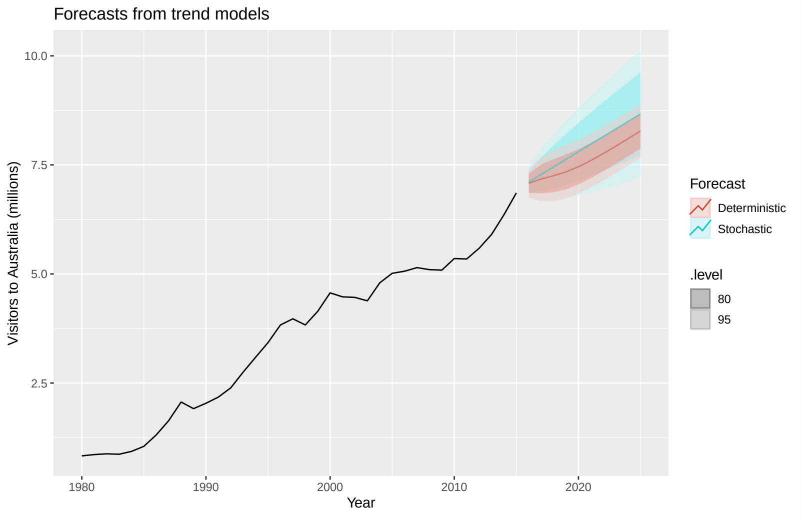
```
## Series: value
## Model: ARIMA(0,1,1) w/ drift
##
## Coefficients:
##          ma1  constant
##       0.3006    0.1735
## s.e.  0.1647    0.0390
##
## sigma^2 estimated as 0.03376:  log likelihood=10.62
## AIC=-15.24   AICc=-14.46   BIC=-10.57
```

$$
\begin{aligned}
y_t - y_{t-1} &= 0.17 + 0.30\varepsilon_{t-1} + \varepsilon_t \\
y_t &= y_0 + 0.17t + \eta_t \\
\eta_t &= \eta_{t-1} + 0.30\varepsilon_{t-1} + \varepsilon_t \\
\varepsilon_t &\sim \text{NID}(0, 0.0338).
\end{aligned}
$$

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

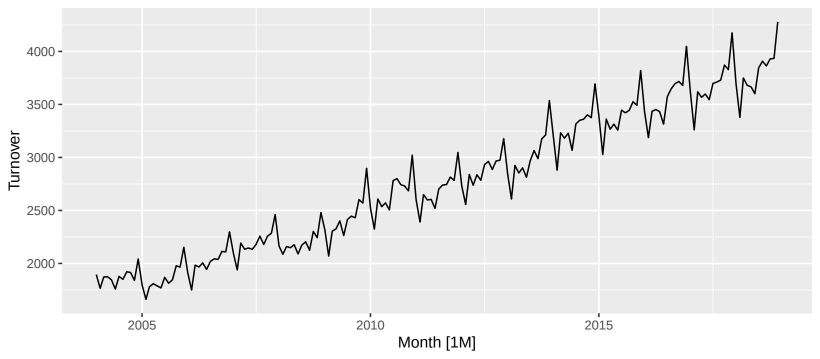
- it allows any length seasonality;
- for data with more than one seasonal period, you can include Fourier terms of different frequencies;
- the seasonal pattern is smooth for small values of K (but more wiggly seasonality can be handled by increasing K);
- the short-term dynamics are easily handled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

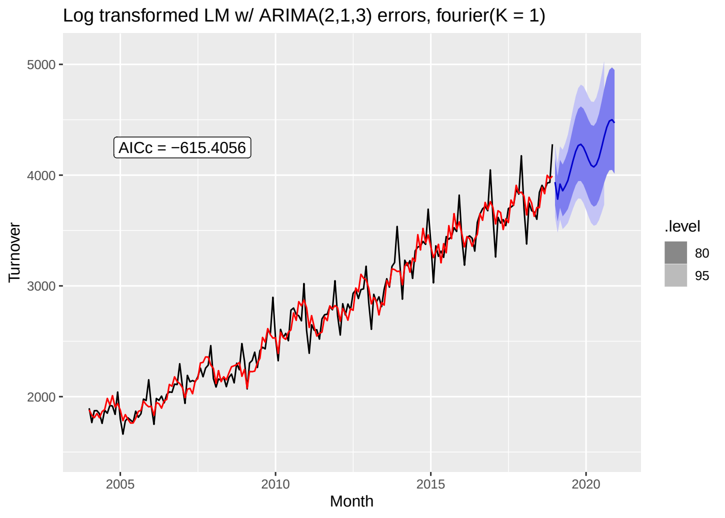


Eating-out expenditure

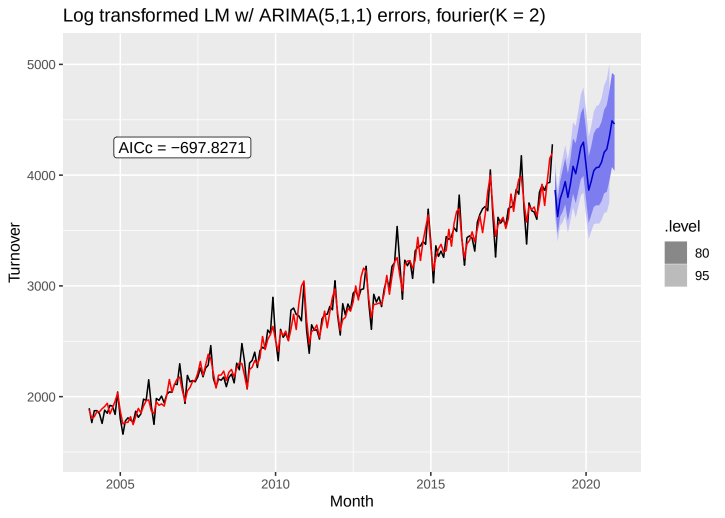
```r
fit <- aus_cafe %>% model(
  K = 1 = ARIMA(log(Turnover) ~ fourier(K = 1) + PDQ(0, 0, 0)),
  K = 2 = ARIMA(log(Turnover) ~ fourier(K = 2) + PDQ(0, 0, 0)),
  K = 3 = ARIMA(log(Turnover) ~ fourier(K = 3) + PDQ(0, 0, 0)),
  K = 4 = ARIMA(log(Turnover) ~ fourier(K = 4) + PDQ(0, 0, 0)),
  K = 5 = ARIMA(log(Turnover)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

.model	sigma2	log_lik	AIC	AICc	BIC
K = 1	0.0017471	317.2353	-616.4707	-615.4056	-587.7842
K = 2	0.0010732	361.8533	-699.7066	-697.8271	-661.4579
K = 3	0.0007609	393.6062	-763.2125	-761.3329	-724.9638
K = 4	0.0005386	426.7839	-821.5678	-818.2098	-770.5697
K = 5	0.0003173	473.7344	-919.4688	-916.9078	-874.8454
K = 6	0.0003163	474.0307	-920.0614	-917.5004	-875.4380

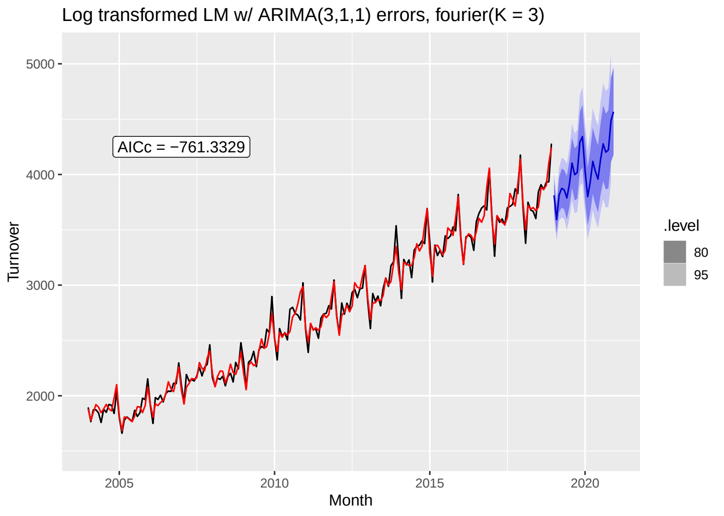
Eating-out expenditure



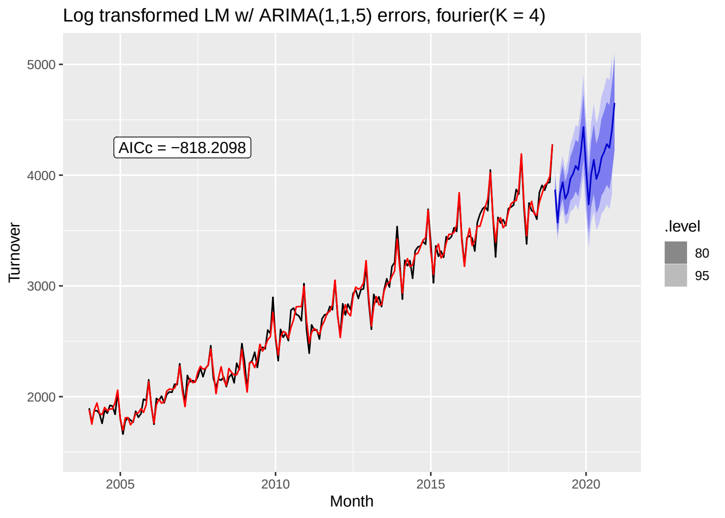
Eating-out expenditure



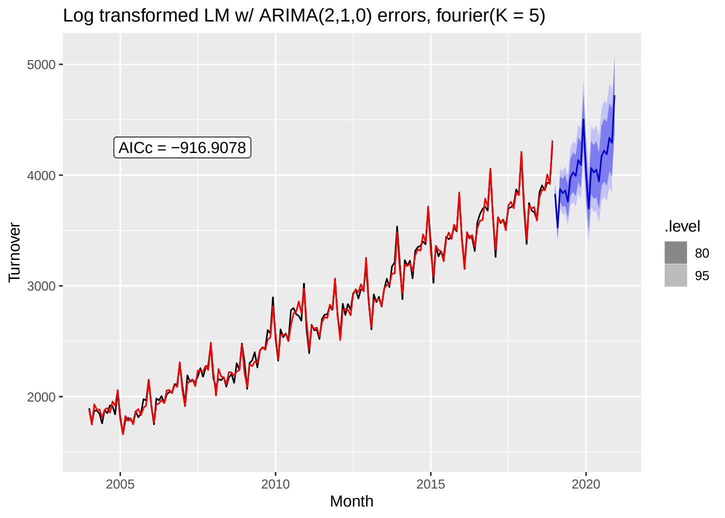
Eating-out expenditure



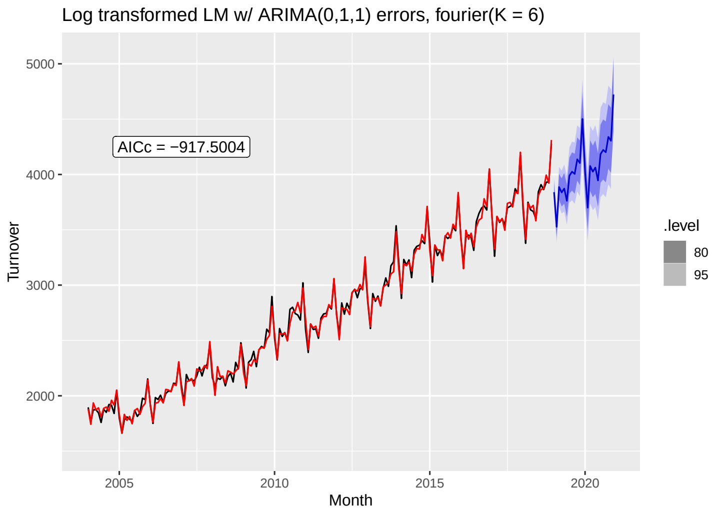
Eating-out expenditure



Eating-out expenditure



Eating-out expenditure



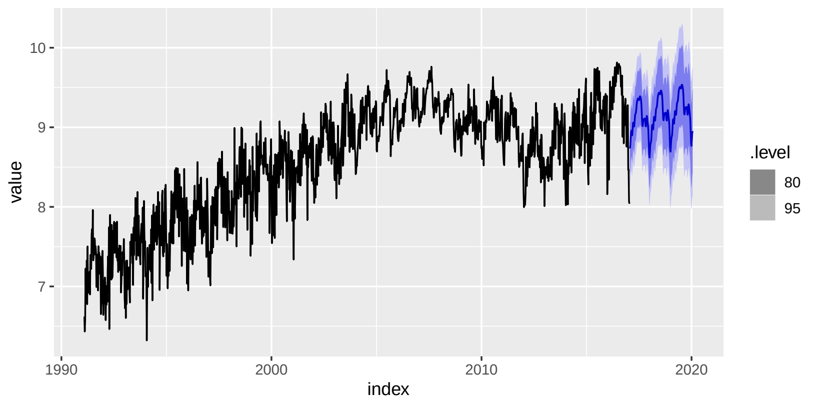
Example: weekly gasoline products

```r
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  fourier(K = 13)C2_52
##       -0.8934               -0.1121               -0.2300                0.0420
## s.e.   0.0132                0.0123                0.0122                0.0099
##       fourier(K = 13)S2_52  fourier(K = 13)C3_52  fourier(K = 13)S3_52
##                     0.0317                0.0832                0.0346
## s.e.                0.0099                0.0094                0.0094
##       fourier(K = 13)C4_52  fourier(K = 13)S4_52  fourier(K = 13)C5_52
##                     0.0185                0.0398               -0.0315
## s.e.                0.0092                0.0092                0.0091
##       fourier(K = 13)S5_52  fourier(K = 13)C6_52  fourier(K = 13)S6_52
##                     0.0009               -0.0522                 0.000
## s.e.                0.0091                0.0090                 0.009
##       fourier(K = 13)C7_52  fourier(K = 13)S7_52  fourier(K = 13)C8_52
##                    -0.0173                0.0053                0.0075
## s.e.                0.0090                0.0090                0.0090
##       fourier(K = 13)S8_52  fourier(K = 13)C9_52  fourier(K = 13)S9_52
```

Example: weekly gasoline products

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



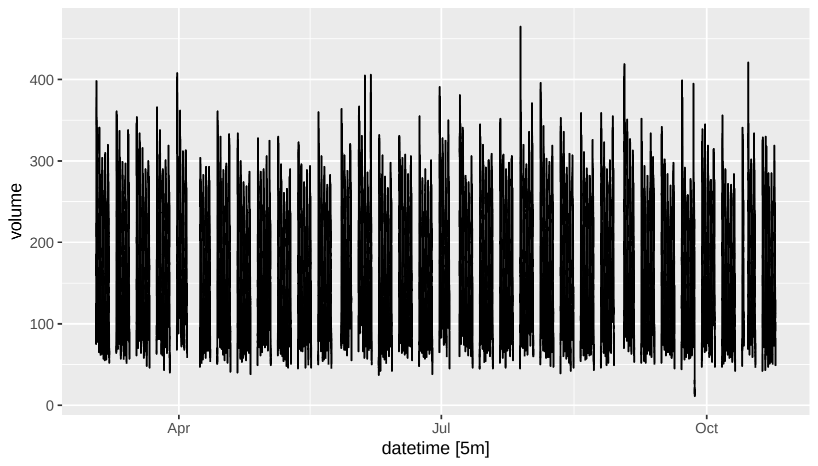
5-minute call centre volume

```
calls <- readr::read_tsv("http://robjhyndman.com/data/callcenter.txt") %>%
  pivot_longer(-X1, names_to = "date", values_to = "volume") %>%
  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
```

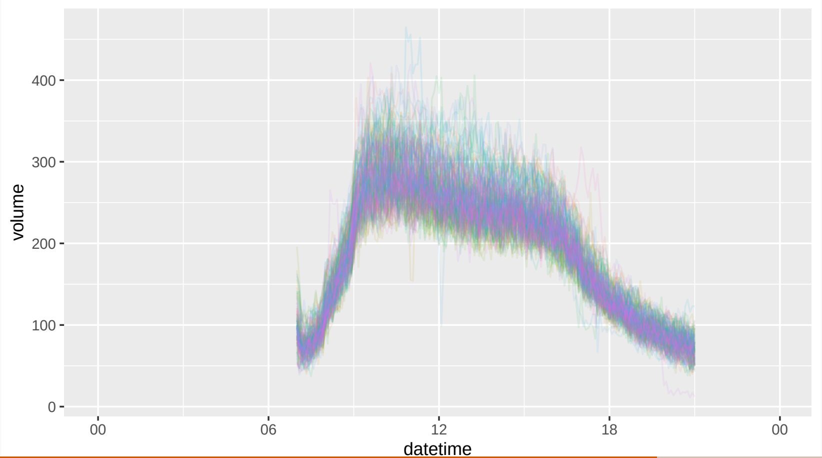
5-minute call centre volume

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



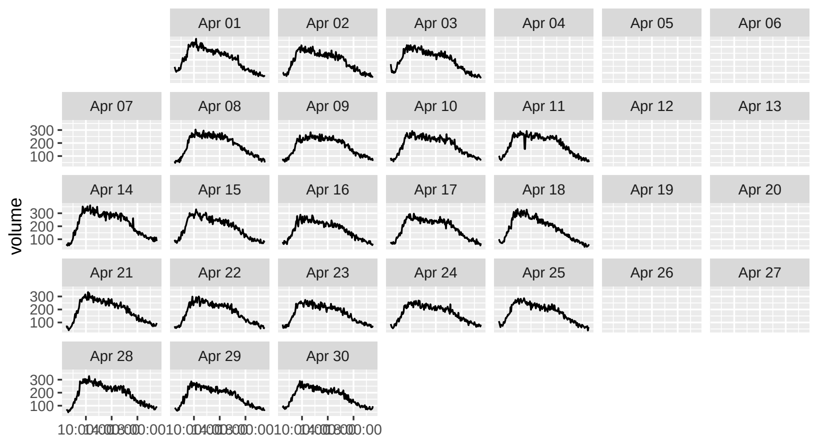
5-minute call centre volume

```r
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)
```



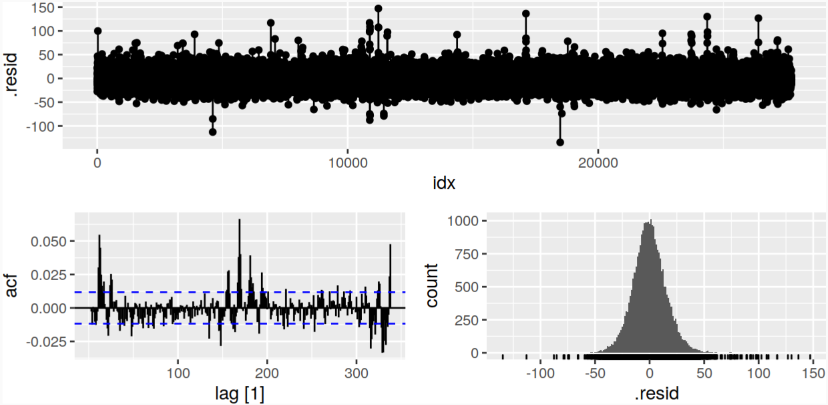
5-minute call centre volume

```r
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  fourier(169, K = 10)C1_169
##       0.9894  -0.7383  -0.0333  -0.0282                    -79.0702
## s.e.  0.0010   0.0061   0.0075   0.0060                      0.7001
##       fourier(169, K = 10)S1_169  fourier(169, K = 10)C2_169
##                          55.2985                    -32.3615
## s.e.                      0.7006                      0.3784
##       fourier(169, K = 10)S2_169  fourier(169, K = 10)C3_169
##                          13.7417                     -9.3180
## s.e.                      0.3786                      0.2725
##       fourier(169, K = 10)S3_169  fourier(169, K = 10)C4_169
##                         -13.6446                     -2.7913
## s.e.                      0.2726                      0.2230
##       fourier(169, K = 10)S4_169  fourier(169, K = 10)C5_169
```

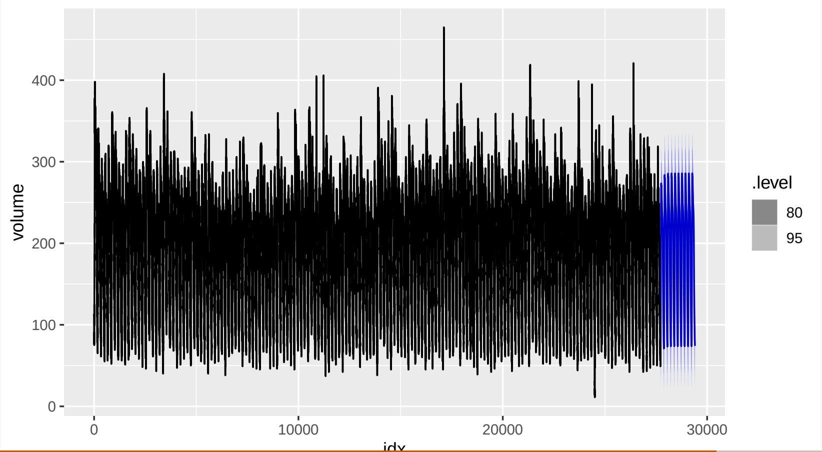
5-minute call centre volume

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



5-minute call centre volume

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



Outline

Lab Session 19

Repeat Lab Session 18 but using all available data, and handling the annual seasonality using Fourier terms.

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.

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 η_t is an ARIMA process.

Lagged predictors

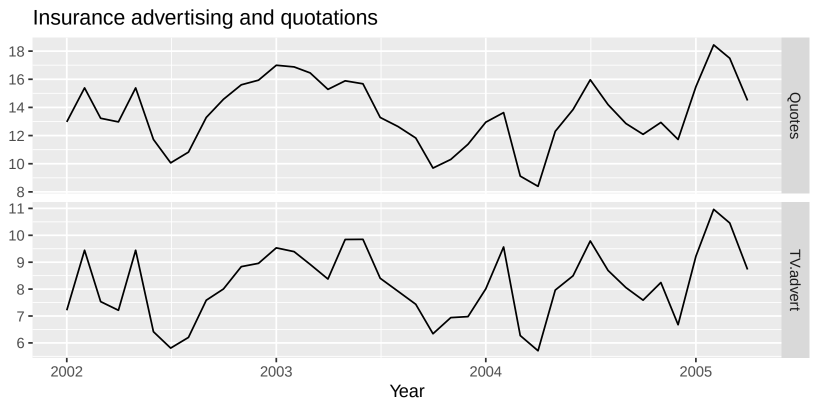
The model include present and past values of predictor: $x_t, x_{t-1}, x_{t-2}, \ldots$.

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

where η_t is an ARIMA process.

- 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	log_lik	AIC	AICc	BIC
0	0.2649757	-28.28210	66.56420	68.32890	75.00859
1	0.2094368	-24.04404	58.08809	59.85279	66.53249
2	0.2150429	-24.01627	60.03254	62.57799	70.16581
3	0.2056454	-22.15731	60.31461	64.95977	73.82565

Example: Insurance quotes and TV adverts

```
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)  intercept
##       1.4117  -0.9317  0.3591     1.2564          0.1625     2.0393
## s.e.  0.1698   0.2545  0.1592     0.0667          0.0591     0.9931
##
## sigma^2 estimated as 0.2165:  log likelihood=-23.89
## AIC=61.78   AICc=65.28   BIC=73.6
```

Example: Insurance quotes and TV adverts

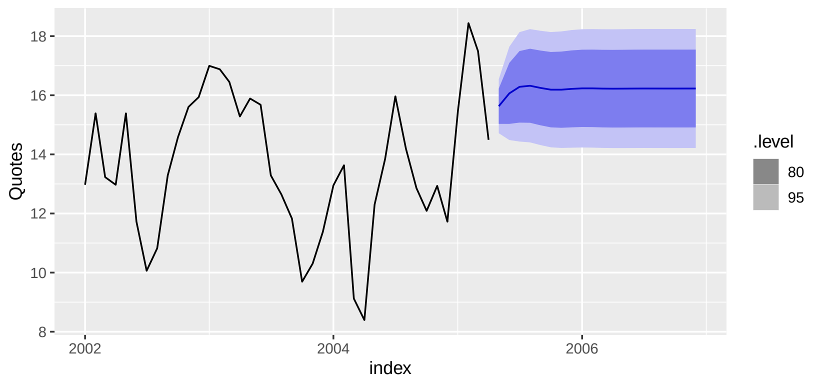
```r
fit <- insurance %>%
  model(ARIMA(Quotes ~ pdq(3, 0, 0) + TV.advert + lag(TV.advert)))
report(fit)
```

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

$$
\begin{aligned}
y_t &= 2.04 + 1.26x_t + 0.16x_{t-1} + \eta_t, \\
\eta_t &= 1.41\eta_{t-1} - 0.93\eta_{t-2} + 0.36\eta_{t-3} + \varepsilon_t,
\end{aligned}
$$

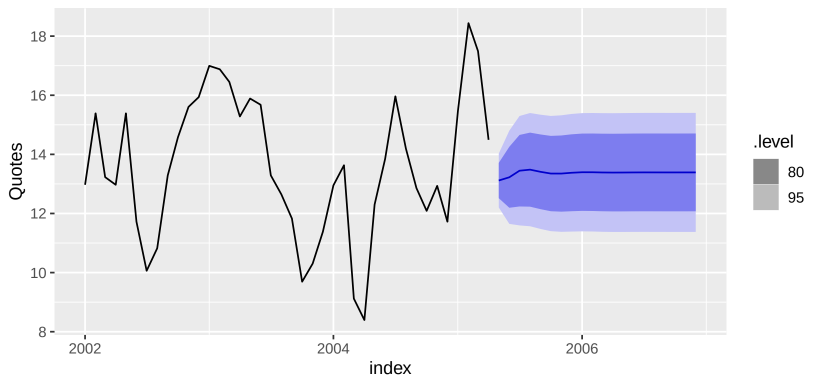
Example: Insurance quotes and TV adverts

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



Example: Insurance quotes and TV adverts

```r
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)
```