Lec 11 : Dynamic Regression Models (Chapter 9)

Forecasting: principles and practice (2nd edition)

book by Rob Hyndman and George Athanasopoulos
Slides by Joseph ALBA

```r
library(fpp2)
library(tseries)
```

Multiple Regression model

In Chapter 5, the multiple regression has one variable to be forecast and several predictor variables. The general form of a multiple regression is

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t,$$

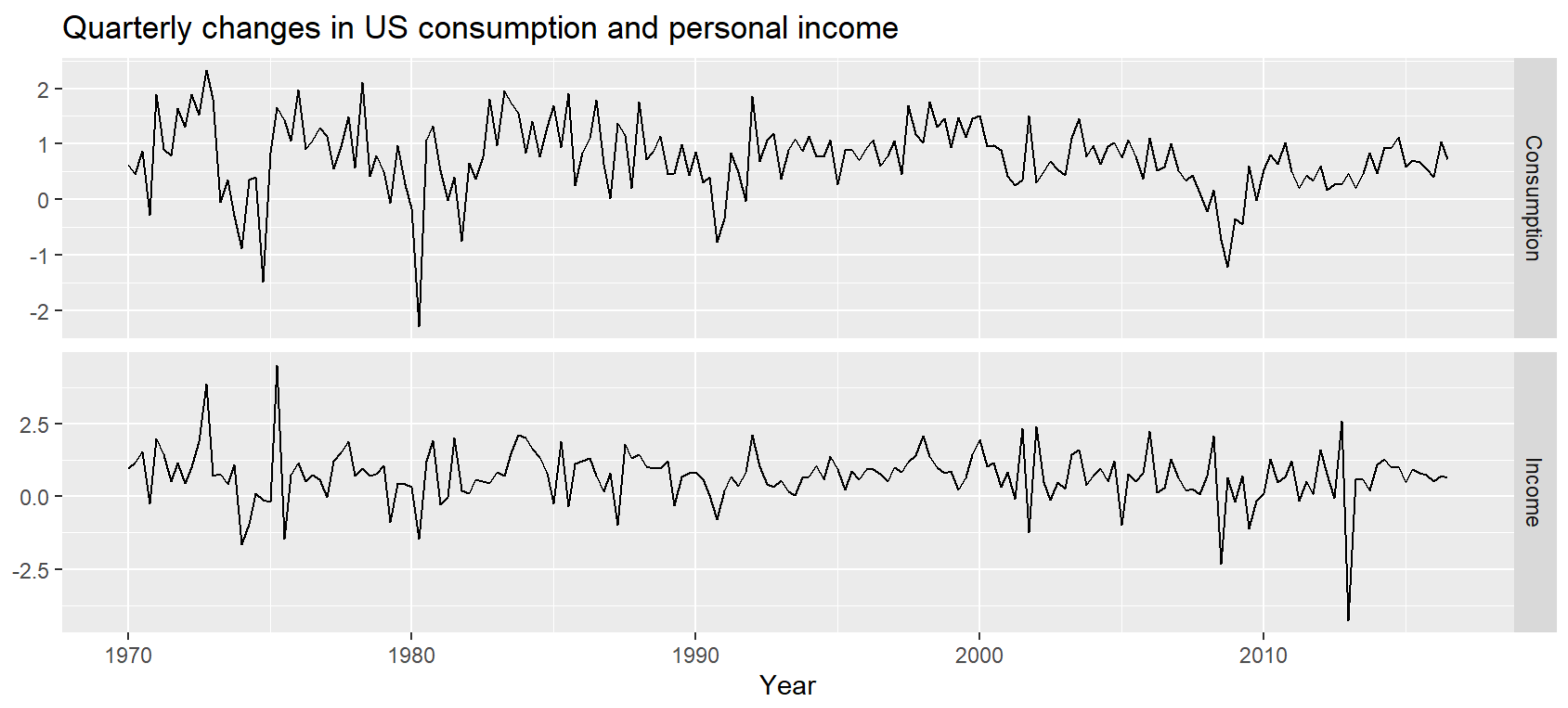
where y_t is the variable to be forecast and $x_{1,t}, \ldots, x_{k,t}$ are the k predictor variables.

The coefficients $\beta_1, \ldots, \beta_k$ measure the effect of each predictor after taking account of the effect of all other predictors in the model. ε_t is supposed to be uncorrelated error term (white noise).

However, ε_t may be autocorrelated as in our example in Chapter 5 shows

Example: US Personal Consumption and Income

```r
autoplot(uschange[,1:2], facets=TRUE) +
xlab("Year") + ylab("") +
ggtitle("Quarterly changes in US consumption and personal income")
```

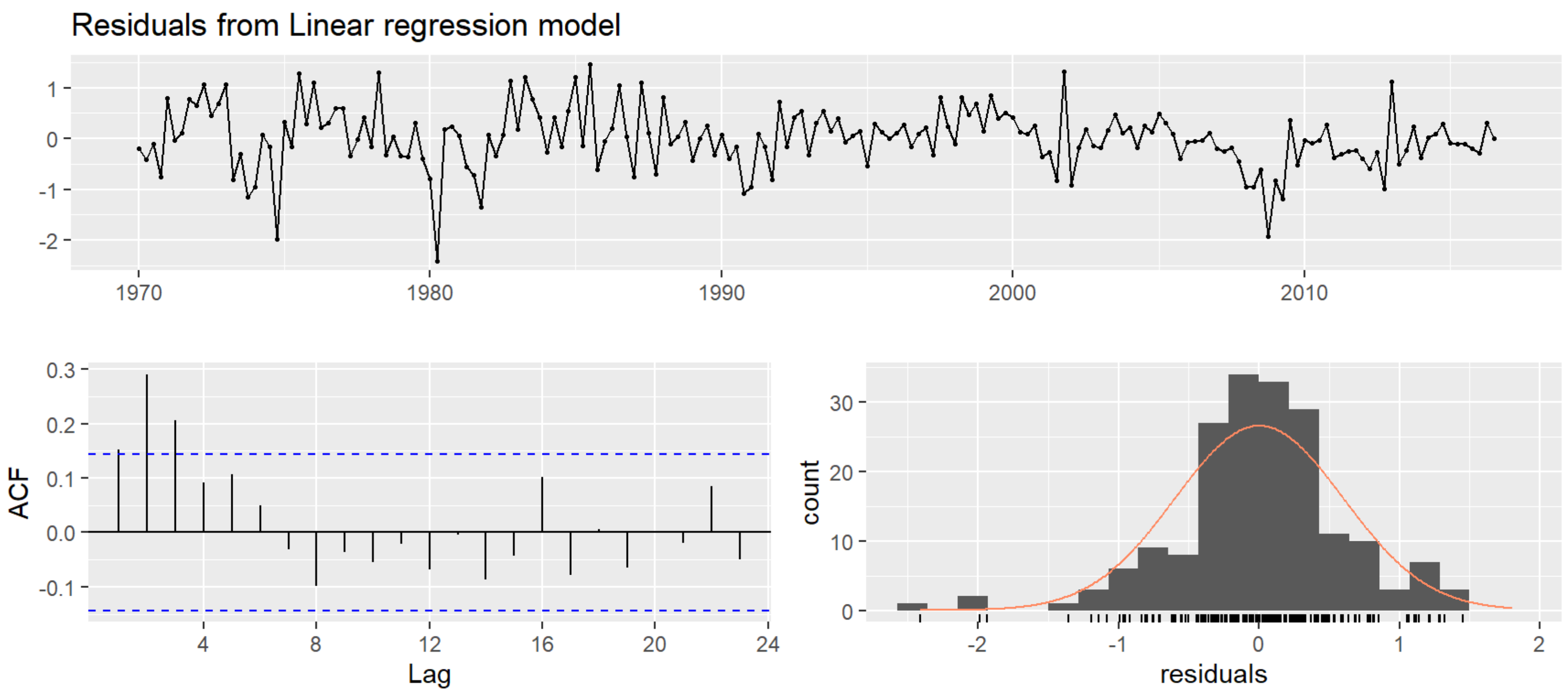


Regression without ARIMA errors (Chapter 5)

```r
fit.ci<-tslm(Consumption ~ Income, data=uschange);
summary(fit.ci)
```

```
##
## Call:
## tslm(formula = Consumption ~ Income, data = uschange)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.40845 -0.31816  0.02558  0.29978  1.45157
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.54510    0.05569   9.789  < 2e-16 ***
## Income       0.28060    0.04744   5.915 1.58e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6026 on 185 degrees of freedom
## Multiple R-squared:  0.159,  Adjusted R-squared:  0.1545
## F-statistic: 34.98 on 1 and 185 DF,  p-value: 1.577e-08
```

```
checkresiduals(fit.ci)
```



```
##
##  Breusch-Godfrey test for serial correlation of order up to 8
##
## data:  Residuals from Linear regression model
## LM test = 27.422, df = 8, p-value = 0.0005977
```

Dynamic Regression Models

Here, the errors from a regression are allowed to contain autocorrelation. To emphasise this change in perspective, ε_t will be replaced with η_t in the equation. The error series η_t is assumed to follow an ARIMA model e.g., if η_t follows an ARIMA(1,1,1) model, then:

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

where ε_t is the uncorrelated error term (white noise).

Estimation

When the parameters from the dynamic model are estimated, the sum of squared values ε_t needs to be minimized. If the sum of squared values η_t is minimized instead (which happens if the regression model is estimated while ignoring the autocorrelations in the errors), then several problems arise.

1. The estimated coefficients $\hat{\beta}_0, \cdots, \hat{\beta}_k$ are no longer the best estimates, as some information has been ignored in the calculation;
2. Any statistical tests associated with the model (e.g., t-tests on the coefficients) will be incorrect.
3. The AICc values of the fitted models are no longer a good guide as to which is the best model for forecasting.
4. In most cases, the p-values associated with the coefficients will be too small, and so some predictor variables will appear to be important when they are not.This is known as "spurious regression".

Estimation

Minimising the sum of squared ε_t values avoids these problems.

An important consideration when estimating a regression with ARMA errors is that all of the variables in the model must first be stationary. Thus, we first have to check that y_t and all of the predictors $x_{1,t}, \cdots, x_{k,t}$ appear to be stationary.

We therefore first difference the non-stationary variables in the model. It is often desirable to maintain the form of the relationship between y_t and the predictors, and consequently it is common to difference all of the variables if any of them need differencing. The resulting model is then called a “model in differences”, as distinct from a “model in levels”, which is what is obtained when the original data are used without differencing.

Estimation

If all of the variables in the model are stationary, then we only need to consider ARMA errors for the residuals. We can see that a regression model with ARIMA errors is equivalent to a regression model in differences with ARMA errors. For example, if the above regression model with ARIMA(1,1,1) errors is differenced we obtain the model

$$y'_t = \beta_0 + \beta_1 x'_{1,t} + \beta_2 x'_{2,t} + \cdots + \beta_k x'_{k,t} + \eta'_t,$$

$$(1 - \phi_1 B)\eta'_t = (1 + \theta_1 B)\varepsilon_t,$$

where $y'_t = y_t - y_{t-1}$, $x'_{i,t} = x_{i,t} - x_{i,t-1}$ and $\eta'_t = \eta_t - \eta_{t-1}$ which is a regression model in differences with ARMA errors.

Regression with ARIMA errors in R

The R function Arima() will fit a regression model with ARIMA errors if the argument xreg is used. The *order* argument specifies the order of the ARIMA error model. If differencing is specified, then the differencing is applied to all variables in the regression model before the model is estimated.

```
fit <- Arima(y, xreg=x, order=c(1,1,0))
```

will fit the model $y'_t = \beta_1 x'_t + \eta'_t$, where $\eta'_t = \phi_1 \eta'_{t-1} + \varepsilon_t$ is an AR(1) error. This is equivalent to the model:

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

where η_t is an ARIMA(1,1,0) error. The constant term β_0 disappears due to the differencing. To include a constant in the differenced model, specify *include.drift=TRUE*

The *auto.arima()* function will also handle regression terms via the *xreg* argument. The user must specify the predictor variables to include, but auto.arima() will select the best ARIMA model for the errors. If differencing is required, then all variables are differenced during the estimation process, although the final model will be expressed in terms of the original variables.

The AICc is calculated for the final model, and this value can be used to determine the best predictors. That is, the procedure should be repeated for all subsets of predictors to be considered, and the model with the lowest AICc value selected.

Example: US Personal Consumption and Income

Regression with ARIMA errors

```
fit <- auto.arima(uschange[,"Consumption"],xreg=uschange[,"Income"]);
summary(fit)
```

```
## Series: uschange[, "Consumption"]
## Regression with ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1     ma2  intercept    xreg
##       0.6922  -0.5758  0.1984     0.5990  0.2028
## s.e.  0.1159   0.1301  0.0756     0.0884  0.0461
##
## sigma^2 estimated as 0.3219:  log likelihood=-156.95
## AIC=325.91   AICc=326.37   BIC=345.29
##
## Training set error measures:
##                      ME      RMSE       MAE     MPE     MAPE      MASE
## Training set 0.001714366 0.5597088 0.4209056 27.4477 161.8417 0.6594731
##                     ACF1
## Training set 0.006299231
```

The data are clearly already stationary (as we are considering percentage changes rather than raw expenditure and income), so there is no need for any differencing. The fitted model is

$$\begin{gathered} y_t = 0.599 + 0.203x_t + \eta_t, \\ \eta_t = 0.692\eta_{t-1} + \varepsilon_t - 0.576\varepsilon_{t-1} + 0.198\varepsilon_{t-2}, \\ \varepsilon_t \sim NID(0, 0.322) \end{gathered}$$

We can recover estimates of both the η_t and the ε_t series using the residuals() function.

```
cbind("Regression Errors" = residuals(fit, type="regression"),
"ARIMA errors" = residuals(fit, type="innovation")) %>%autoplot(facets=TRUE)
```

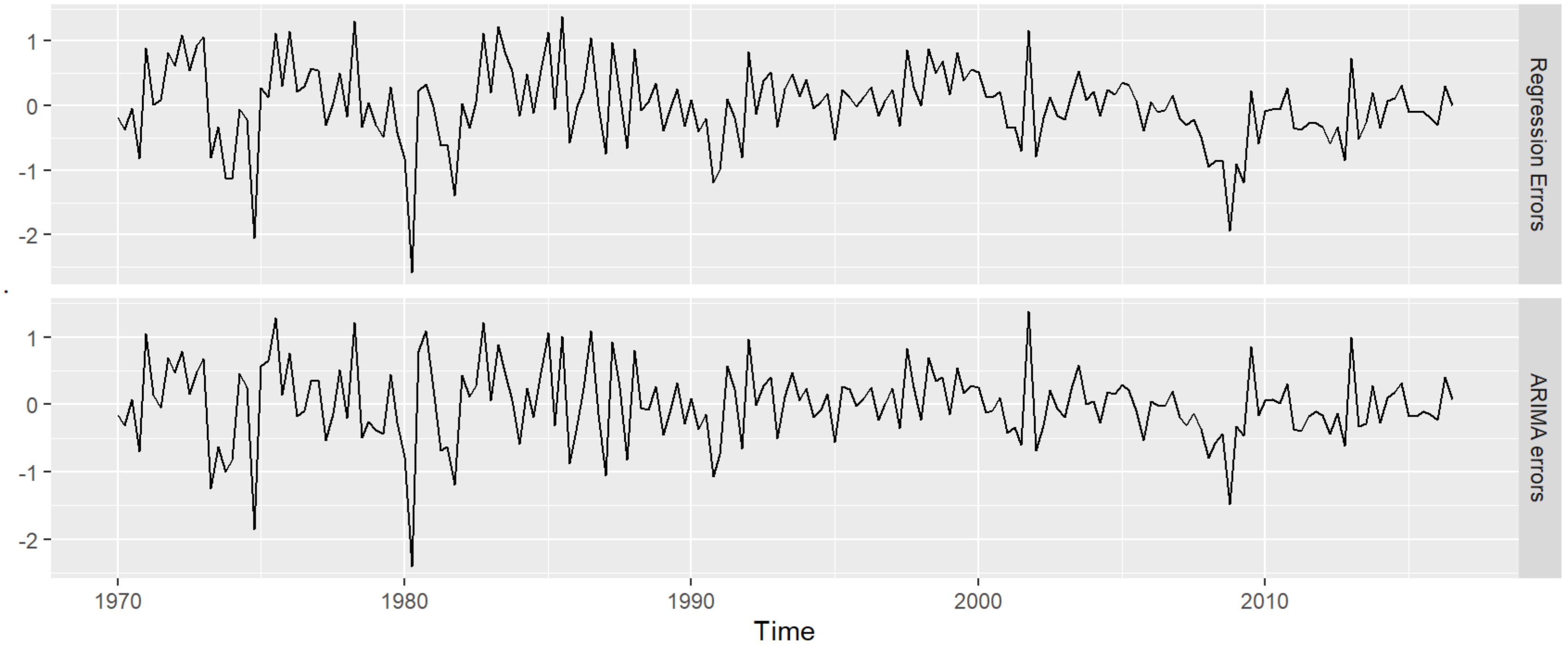
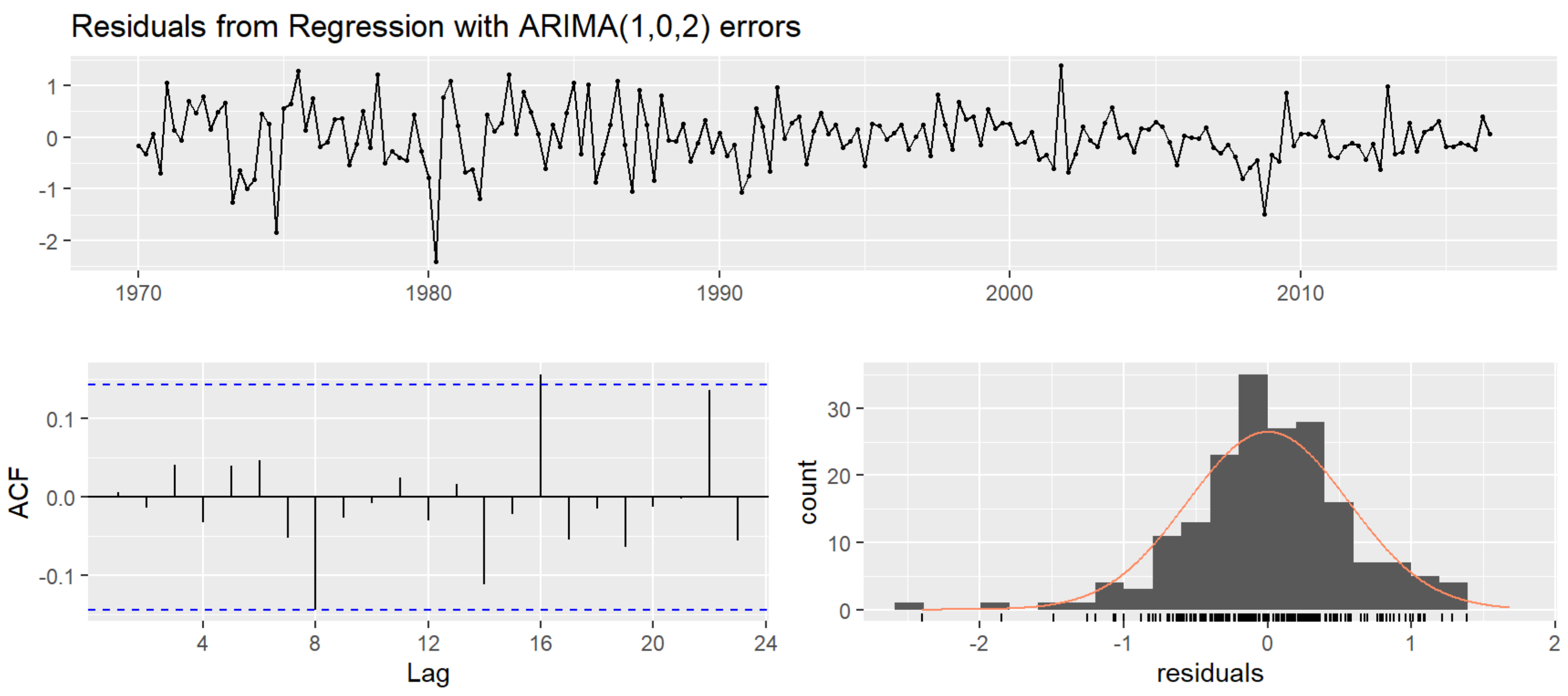


Figure 9.2: Regression errors (η_t) and ARIMA errors (ε_t) from the fitted model.

It is the ARIMA errors that should resemble a white noise series.

```
checkresiduals(fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(1,0,2) errors
## Q* = 5.8916, df = 3, p-value = 0.117
##
## Model df: 5.   Total lags used: 8
```

Forecasting

To forecast using a regression model with ARIMA errors,

- forecast the regression part of the model and the ARIMA part of the model, and combine the results.
- As with ordinary regression models, in order to obtain forecasts we first need to forecast the predictors. When the predictors are known into the future (e.g., calendar-related variables such as time, day-of-week, etc.)
- But when the predictors are themselves unknown, we must either model them separately, or use assumed future values for each predictor.

Example: US Personal Consumption and Income

```r
fcast <- forecast(fit, xreg=rep(mean(uschange[,2]),8))
autoplot(fcast) + xlab("Year") + ylab("Percentage change")
```

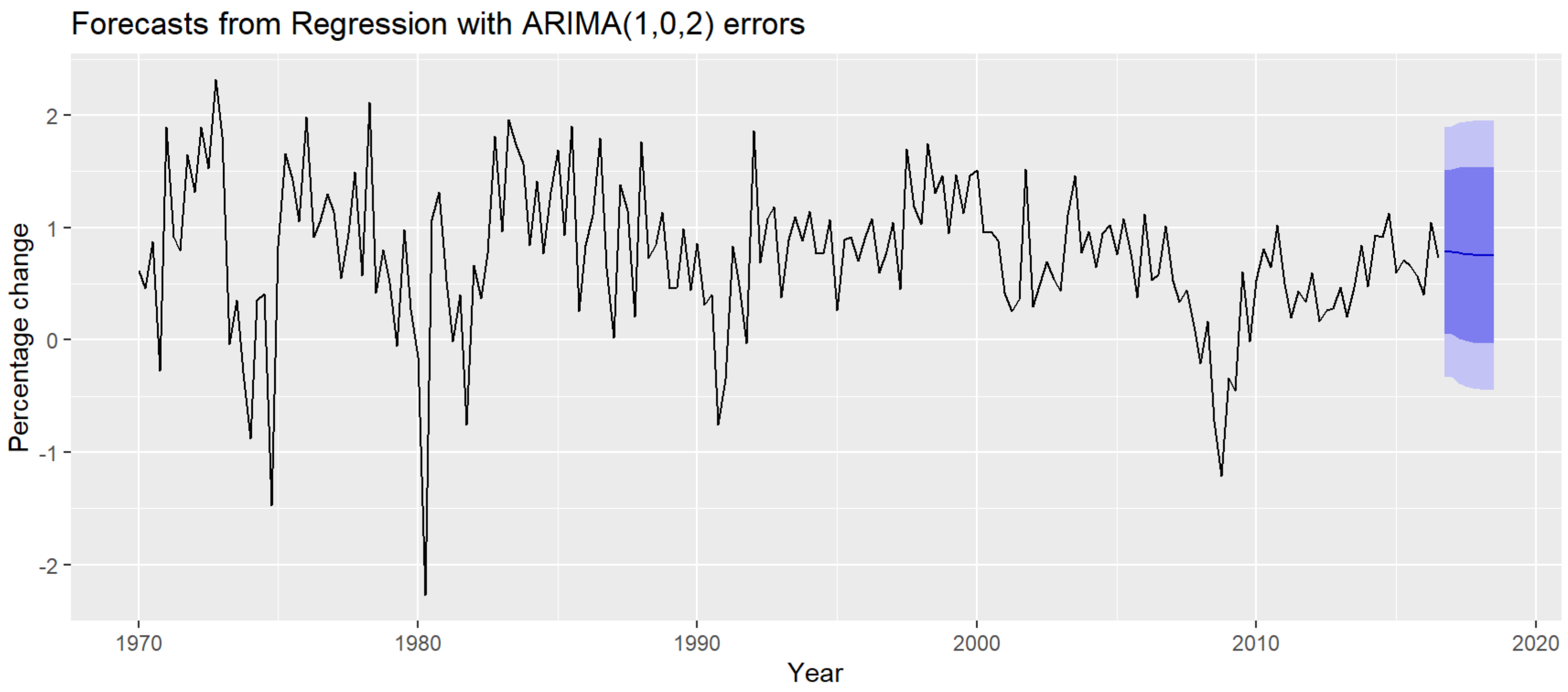


Figure 9.4: Forecasts obtained from regressing the percentage change in consumption expenditure on the percentage change in disposable income, with an ARIMA(1,0,2) error model.

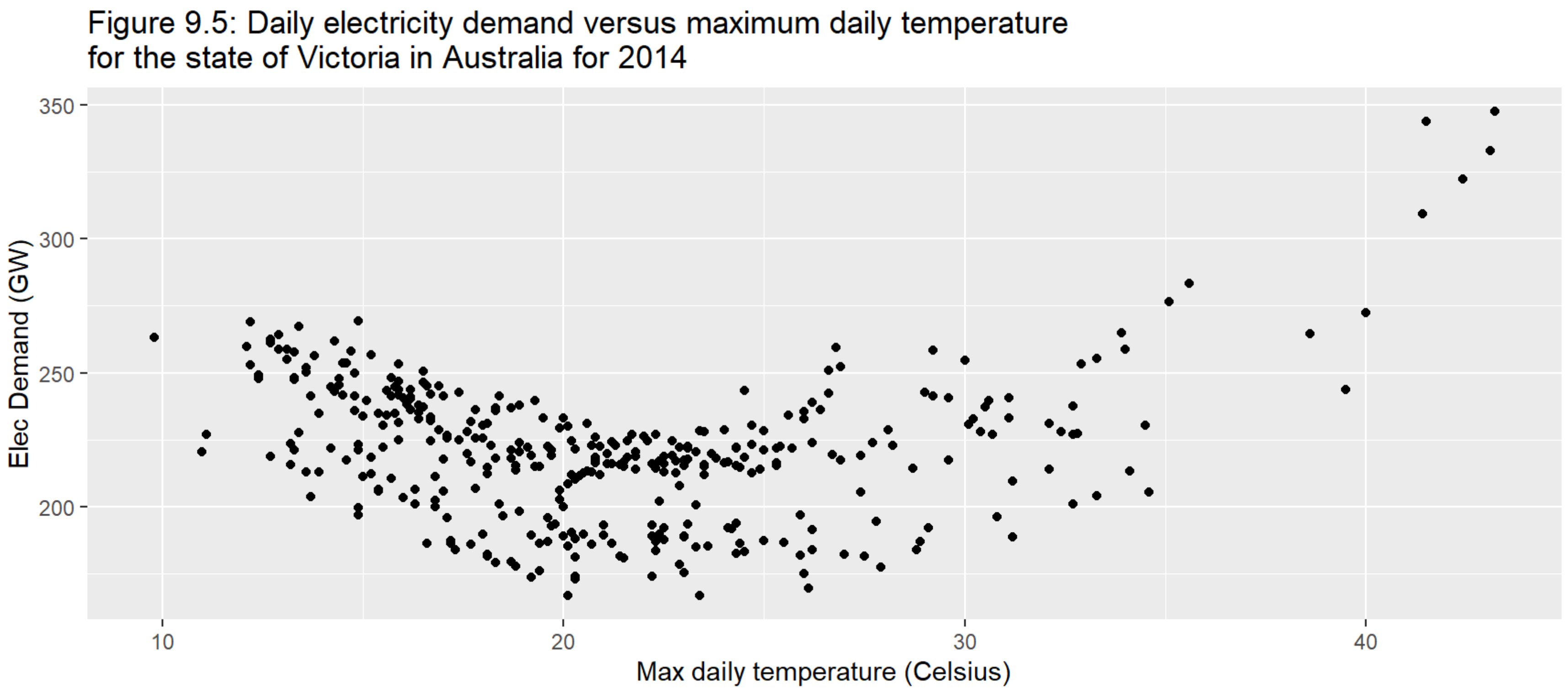
Note that:

-The prediction intervals for this model are narrower than those for the model developed in Section 8.5 because we are now able to explain some of the variation in the data using the income predictor.

-The prediction intervals from regression models (with or without ARIMA errors) do not take into account the uncertainty in the forecasts of the predictors. So they should be interpreted as being conditional on the assumed (or estimated) future values of the predictor variables.

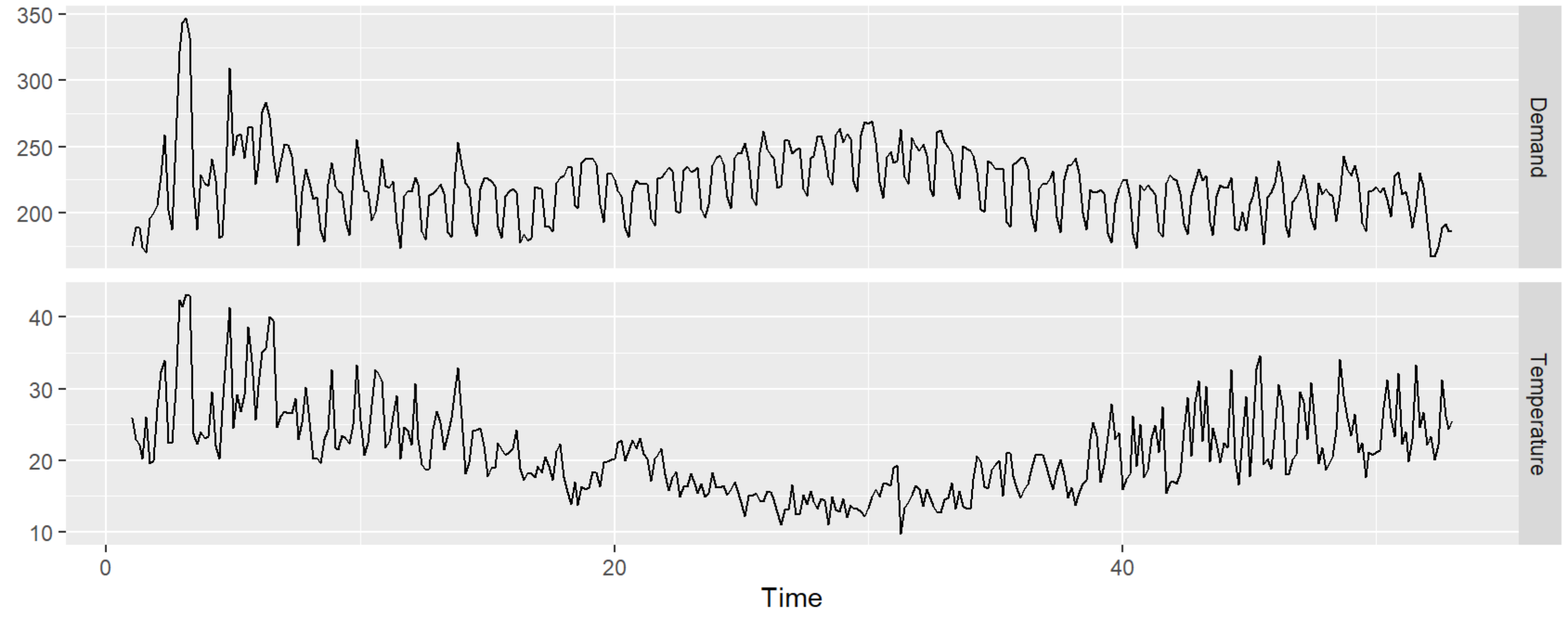
Example: Forecasting electricity demand

```
qplot(Temperature, Demand, data=as.data.frame(elecdaily)) +
ylab("Elec Demand (GW)") + xlab("Max daily temperature (Celsius)") + ggtitle("Figure 9.5: Daily electricity demand versus maximum daily temperature
for the state of Victoria in Australia for 2014")
```



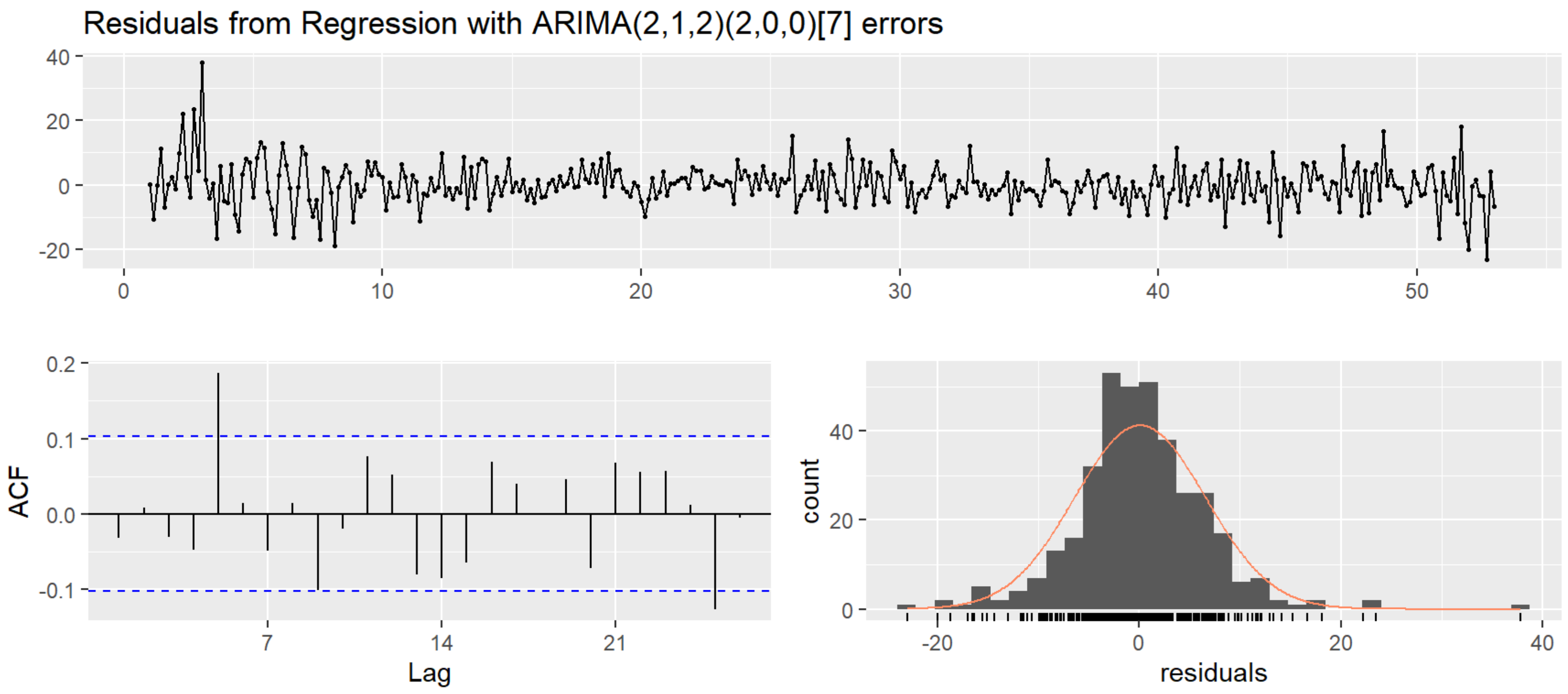
```
autoplot(elecdaily[,c(1,3)], facets=TRUE) + ggtitle("Figure 9.6: Daily electricity demand and maximum daily temperature for
the state of Victoria in Australia for 2014.") +
ylab("")
```

Figure 9.6: Daily electricity demand and maximum daily temperature for the state of Victoria in Australia for 2014.



In this example, we fit a quadratic regression model with ARMA errors using the *auto.arima()* function.

```r
xreg <- cbind(MaxTemp = elecdaily[, "Temperature"],
MaxTempSq = elecdaily[, "Temperature"]^2,
Workday = elecdaily[, "WorkDay"])
fit <- auto.arima(elecdaily[, "Demand"], xreg = xreg)
checkresiduals(fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,1,2)(2,0,0)[7] errors
## Q* = 28.229, df = 4, p-value = 1.121e-05
##
## Model df: 10.   Total lags used: 14
```

- Model has some significant autocorrelation in the residuals (the prediction intervals may not provide accurate coverage.)
- Histogram of the residuals shows one positive outlier, which will also affect the coverage of the prediction intervals.
- Using the estimated model, we forecast 14 days ahead starting from Thursday 1 January 2015 (a non-work-day being a public holiday for New Years Day).

Hence, we could obtain weather forecasts from the weather bureau for the next 14 days.

For illustration, the scenario based forecasting is used where the temperature is set for the next 14 days to a constant 26 degrees.

```
fcast <- forecast(fit,
xreg = cbind(MaxTemp=rep(26,14), MaxTempSq=rep(26^2,14),
Workday=c(0,1,0,0,1,1,1,1,1,0,0,1,1,1)))
autoplot(fcast) + ylab("Electricity demand (GW)")
```

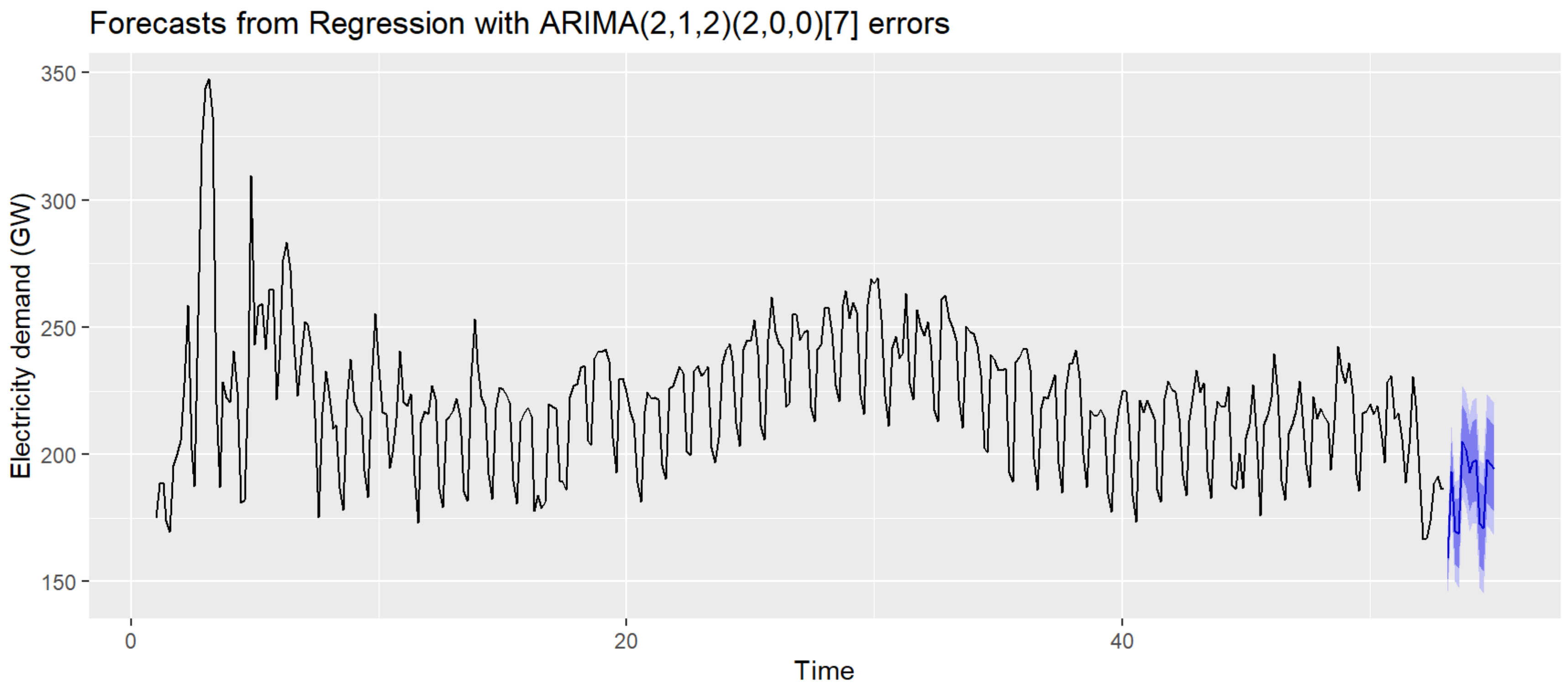


Figure 9.8: Forecasts from the dynamic regression model for daily electricity demand. All future temperatures set to 26 degrees, and working day dummy variable set to known future values.

The point forecasts look reasonable for the first two weeks of 2015. The slow down in electricity demand at the end of 2014 (due to many people taking summer vacations) has caused the forecasts for the next two weeks to show similarly low demand values.

Stochastic and deterministic trends

Two different ways of modelling a linear trend:

1. A deterministic trend is obtained using the regression model

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

where η_t is an ARMA process with $d = 0$.

2. stochastic trend is obtained using the model:

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

where η_t is an ARIMA process with $d = 1$.

In 2, both sides of the equation could be differenced so that $y'_t = \beta_1 + \eta'_t$, where η'_t is an ARMA process so that

$$y_t = y_{t-1} + \beta_1 + \eta'_t$$

This is a random walk with drift where the error term is an ARMA process rather than simply white noise.

Example: International visitors to Australia

In this example, we only wish compare deterministic and stochastic trends and ignore whether international visitors to Australia is a stationary or a non-stationary series.

First we plot the series in the next slide

```
autoplot(austa) + xlab("Year") +
ylab("millions of people") +
ggtitle("Total annual international visitors to Australia")
```

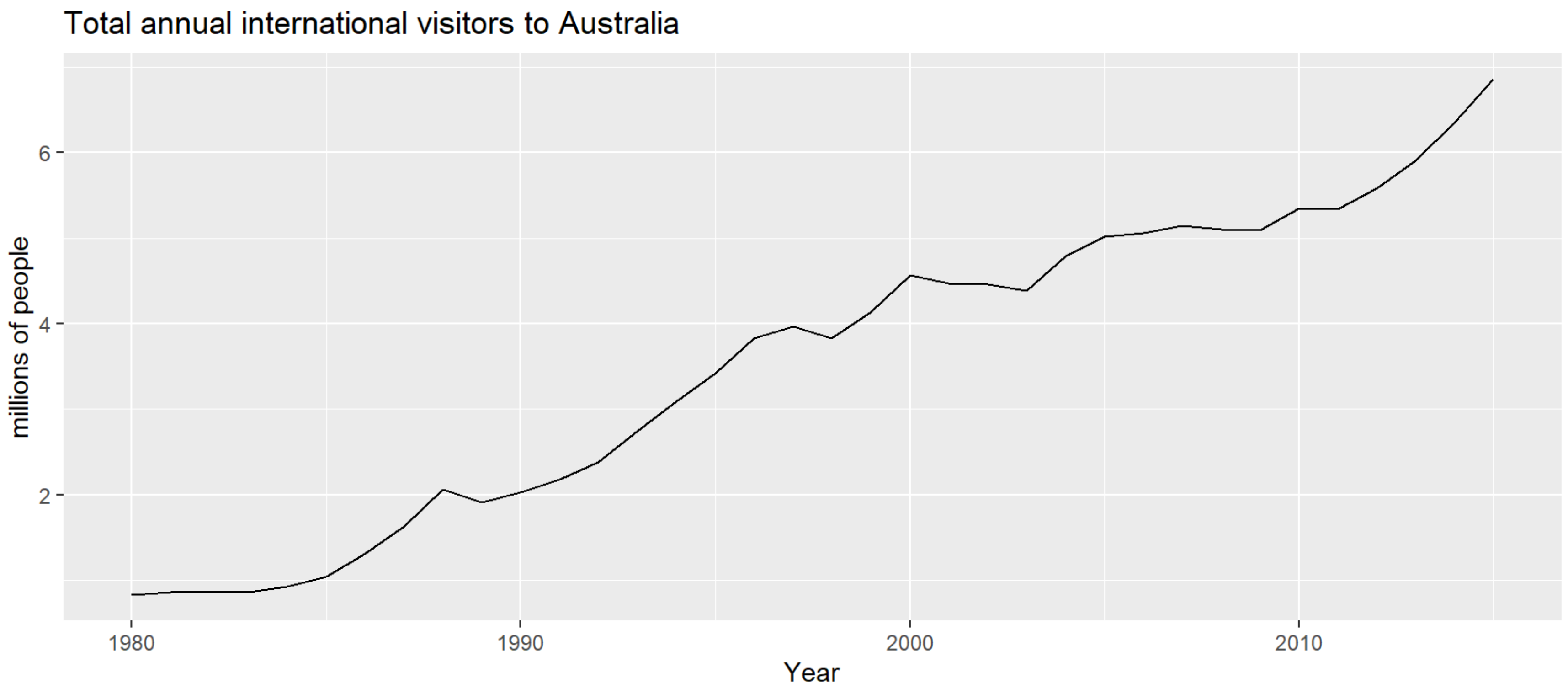


Figure 9.9 the total number of annual international visitors to Australia each year from 1980 to 2015.

The deterministic trend model is obtained as follows:

```
trend <- seq_along(austa)
(fit1 <- auto.arima(austa, d=0, xreg=trend))
```

```
## Series: austa
## Regression with ARIMA(2,0,0) errors
##
## Coefficients:
##          ar1      ar2  intercept    xreg
##       1.1127  -0.3805     0.4156  0.1710
## s.e.  0.1600   0.1585     0.1897  0.0088
##
## sigma^2 estimated as 0.02979:  log likelihood=13.6
## AIC=-17.2   AICc=-15.2   BIC=-9.28
```

This model can be written as:

$$
\begin{gathered}
y_t = 0.416 + 0.17t + \eta_t, \\
\eta_t = 1.113\eta_{t-1} - 0.380\eta_{t-2} + \varepsilon_t \\
\varepsilon_t \sim NID(0, 0.03)
\end{gathered}
$$

The estimated growth in visitor numbers is 0.17 million people per year

the stochastic trend model can be estimated as:

```r
(fit2 <- auto.arima(austa, d=1))
```

```
## Series: austa
## ARIMA(0,1,1) with drift
##
## Coefficients:
##          ma1   drift
##       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
```

This model can be written as:

$$y_t - y_{t-1} = 0.173 + \eta'_t,$$

or equivalently

$$\begin{aligned} y_t &= y_0 + 0.173t + \eta_t, \\ \eta_t &= \eta_{t-1} + 0.301\varepsilon_{t-1} + \varepsilon_t \\ \varepsilon_t &\sim NID(0, 0.034) \end{aligned}$$

the estimated growth in visitor numbers is also 0.17 million people per year.

Although the growth estimates are similar, the prediction intervals are not, as Figure 9.10 shows. In particular, stochastic trends have much wider prediction intervals because the errors are non-stationary.

```r
fc1 <- forecast(fit1,
xreg = length(austa) + 1:10)
fc2 <- forecast(fit2, h=10)
autoplot(austa) +
autolayer(fc2, series="Stochastic trend") +
autolayer(fc1, series="Deterministic trend") +
ggtitle("Forecasts from trend models") +
xlab("Year") + ylab("Visitors to Australia (millions)") +
guides(colour=guide_legend(title="Forecast"))
```

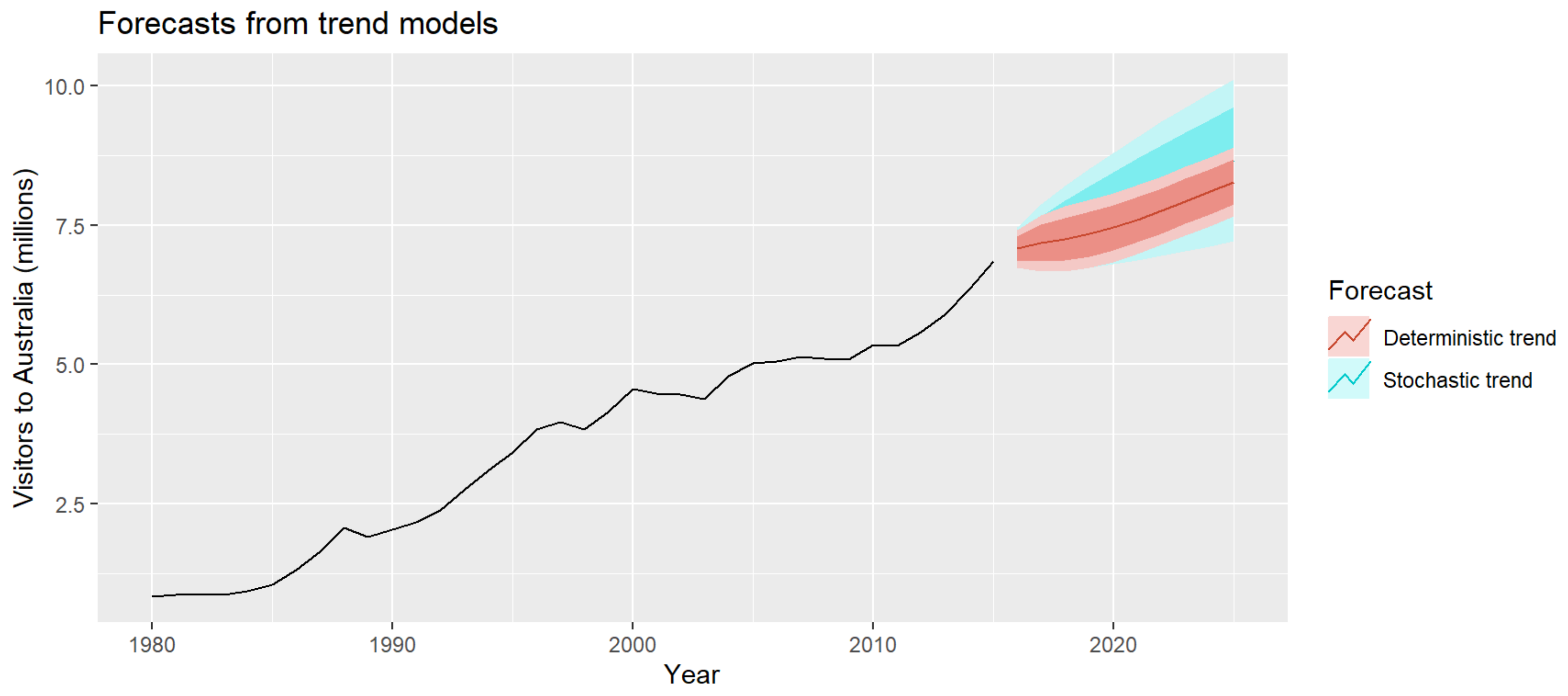


Figure 9.10: Forecasts of annual international visitors to Australia using a deterministic trend model and a stochastic trend model.

The implicit assumptions are:

1. With deterministic trends that the slope of the trend is not going to change over time.
2. Stochastic trends can change, and the estimated growth is only assumed to be the average growth over the historical period, not necessarily the rate of growth that will be observed into the future.

Whether we use 1 or 2 depends on whether y_t is stationary (use 1) or non-stationary (we difference y_t so we get 2)

Dynamic harmonic regression

For long seasonal periods, a dynamic regression with Fourier terms is often better.

Examples:

-daily data => annual seasonality of length 365;

-weekly data => seasonal period of approximately 52;

-half-hourly data => several seasonal periods with the shortest being a daily pattern of period 48

Seasonal versions of ETS and ARIMA models are designed for shorter periods such as 12 for monthly data or 4 for quarterly data.

- The ets() function restricts seasonality to be a maximum period of 24 to allow hourly data but not data with a larger seasonal frequency (The problem is that there are $m-1$ parameters to be estimated for the initial seasonal states where m is the seasonal period.)
- The Arima() and auto.arima() functions will allow a seasonal period up to $m = 350$, but in practice will usually run out of memory whenever the seasonal period is more than about 200.

Seasonal differencing of high order may not make a lot of sense - for daily data it involves comparing what happened today with what happened exactly a year ago and there is no constraint that the seasonal pattern is smooth.

For time series with long seasonal periods, a harmonic regression approach is preferred since the seasonal pattern is modelled using Fourier terms with short-term time series dynamics handled by an ARMA error.

Advantage of this approach is that it allows any length seasonality:

-for data with more than one seasonal period, Fourier terms of different frequencies can be included;

-the smoothness of the seasonal pattern can be controlled by K, the number of Fourier sin and cos pairs - the seasonal pattern is smoother for smaller values of K;

-the short-term dynamics are easily handled with a simple ARMA error.

Disadvantage (compared to ARIMA): the seasonality is assumed to be fixed, i.e., the seasonal pattern is not allowed to change overtime.

Example: Australian eating out expenditure

This example demonstrates combining Fourier terms for capturing seasonality with ARIMA errors capturing other dynamics in the data. Here, monthly data is used for simplicity. The authors use the same approach for weekly data in Section 12.1.

Notes on example:

-Time series is *auscafe*: the total monthly expenditure on cafes, restaurants and takeaway food services in Australia ($billion) from 2004 up to November 2016

-Forecast 24 months ahead.

-K, the number of Fourier sin and cos pairs, is varied from $K = 1$ to $K = 6$ (equivalent to including seasonal dummies).

-Figure 9.11 shows the seasonal pattern projected forward as K increases. As K increases the Fourier terms capture and project a more “wiggly” seasonal pattern while the simpler ARIMA models capture other dynamics.

-The AICc value is minimised for $K = 5$, with a significant jump going from $K = 4$ to $K = 5$ and then increases at $K = 6$. Hence, the forecasts generated from the $K = 5$ model would be the ones used.

```r
cafe04 <- window(auscafe, start=2004)
plots <- list()
for (i in seq(6)) {
fit <- auto.arima(cafe04, xreg = fourier(cafe04, K = i),
seasonal = FALSE, lambda = 0)
plots[[i]] <- autoplot(forecast(fit,
xreg=fourier(cafe04, K=i, h=24))) +
xlab(paste("K=",i," AICC=",round(fit[["aicc"]],2))) +
ylab("") + ylim(1.5,4.7)
}
gridExtra::grid.arrange(
plots[[1]],plots[[2]],plots[[3]],
plots[[4]],plots[[5]],plots[[6]], nrow=3)
```

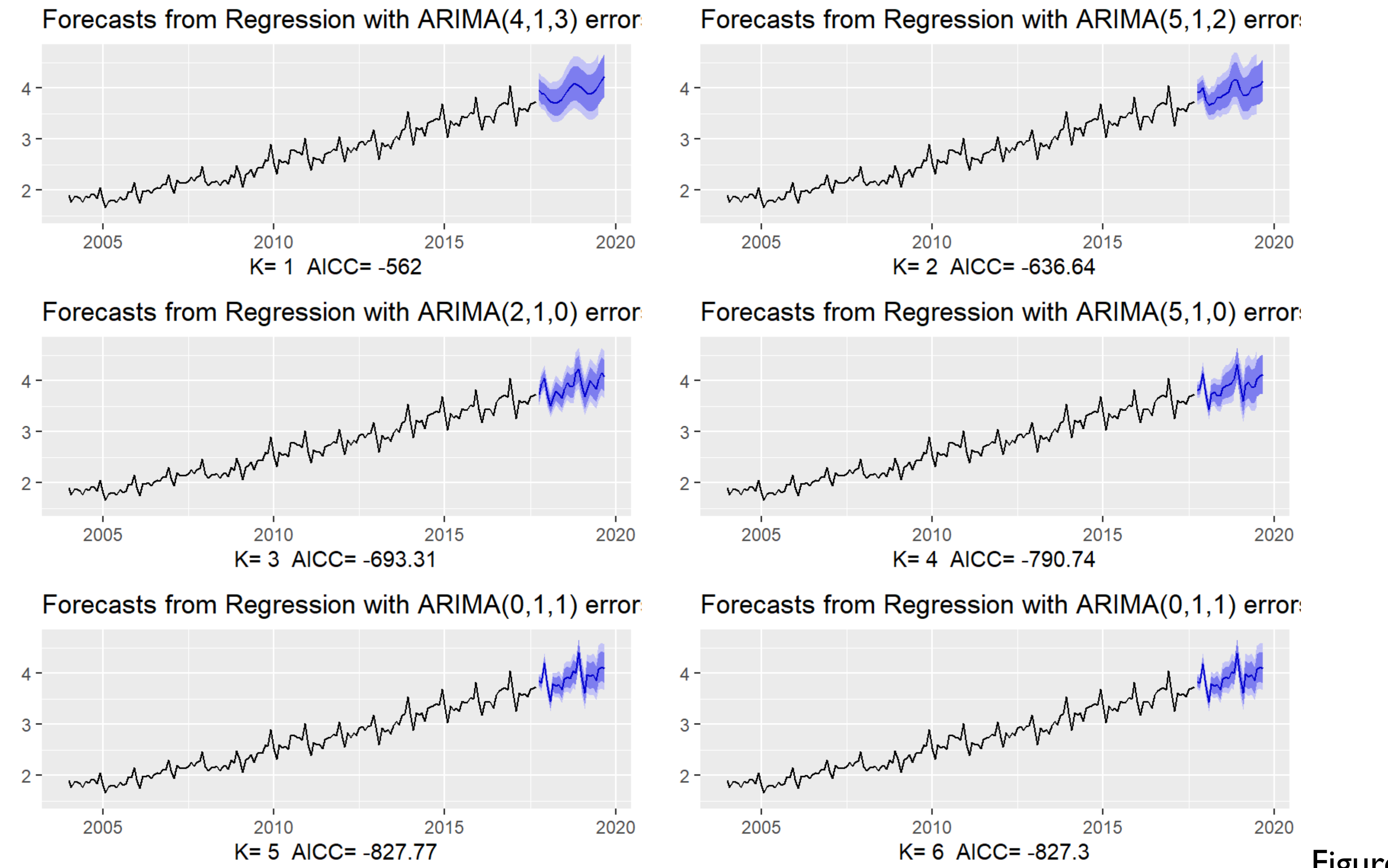


Figure 9.11: Using Fourier terms and ARIMA errors for forecasting monthly expenditure on eating out in Australia.

Lagged predictors

Sometimes, a predictor affects the forecast variable over several periods e.g., advertising campaign affects sales beyond the end of the sales campaign or a change in the company's safety policy reduces accidents immediately but have a diminishing effect overtime.

The appropriate model would allow for lagged effects of the predictor such as:

$$y_t = \beta_0 + \gamma_0 x_t + \gamma_1 x_{t-1} + \cdots + \gamma_k x_{t-k} + \eta_t,$$

where η_t is an ARIMA process. The value of k can be selected using the AICc, along with the values of p and q for the ARIMA error.

Example: TV advertising and insurance quotations

A US insurance company advertises on national television in an attempt to increase the number of insurance quotations provided (and consequently the number of new policies).

```r
autoplot(insurance, facets=TRUE) +
xlab("Year") + ylab("") +
ggtitle("Insurance advertising and quotations")
```

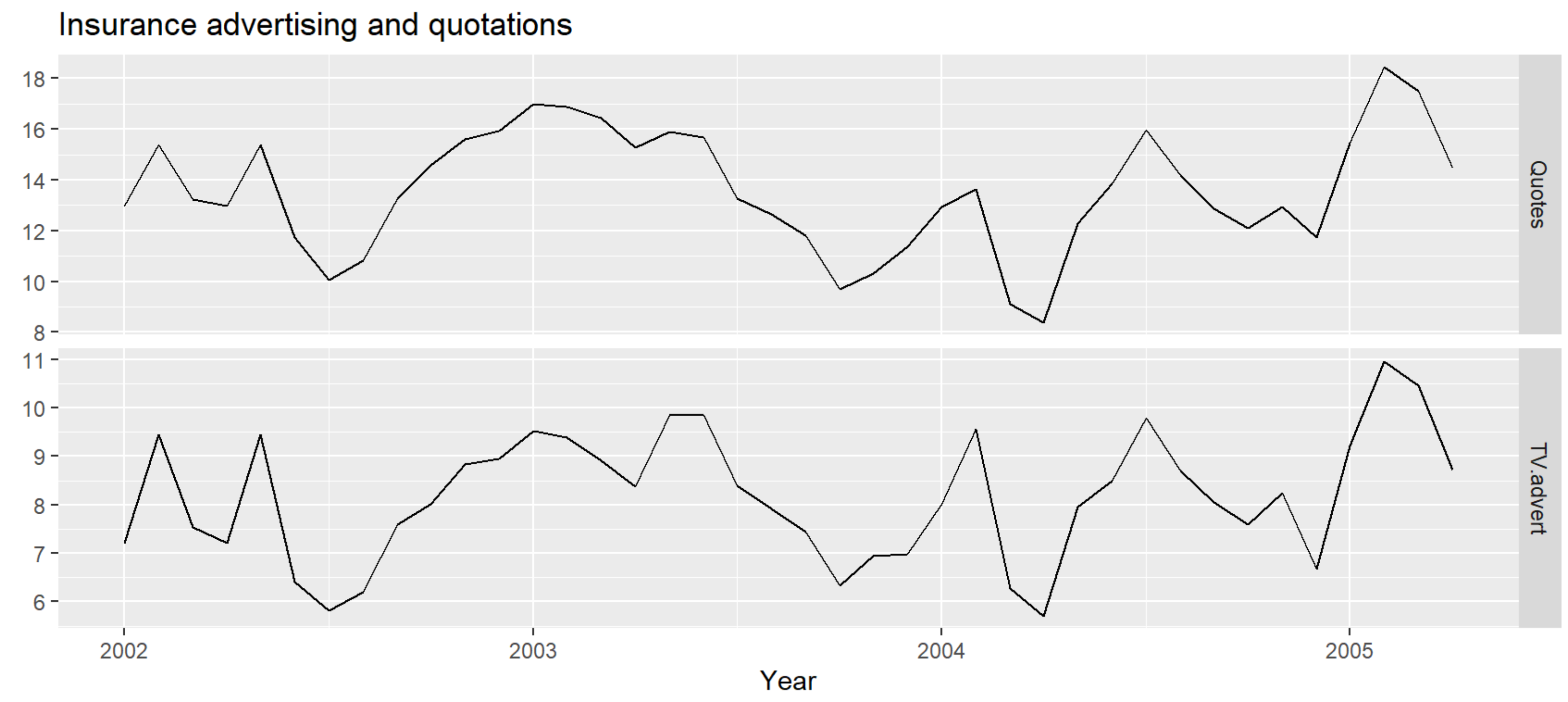


Figure 9.12: Numbers of insurance quotations provided per month and the expenditure on advertising per month.

Advertising expenditure for up to four months is included; that is, the model may include advertising expenditure in the current month, and the three months before that. When comparing models, all models must use the same training set. The following code excludes the first three months in order to make fair comparisons.

```r
# Lagged predictors. Test 0, 1, 2 or 3 lags.
Advert <- cbind(
AdLag0 = insurance[,"TV.advert"],
AdLag1 = stats::lag(insurance[,"TV.advert"],-1),
AdLag2 = stats::lag(insurance[,"TV.advert"],-2),
AdLag3 = stats::lag(insurance[,"TV.advert"],-3)) %>%
head(NROW(insurance))
```

```r
# Restrict data so models use same fitting period
fit1 <- auto.arima(insurance[4:40,1], xreg=Advert[4:40,1],
stationary=TRUE)
fit2 <- auto.arima(insurance[4:40,1], xreg=Advert[4:40,1:2],
stationary=TRUE)
fit3 <- auto.arima(insurance[4:40,1], xreg=Advert[4:40,1:3],
stationary=TRUE)
fit4 <- auto.arima(insurance[4:40,1], xreg=Advert[4:40,1:4],
stationary=TRUE)
```

Choose the optimal lag length for advertising based on the AICc.

```
c(fit1[["aicc"]],fit2[["aicc"]],fit3[["aicc"]],fit4[["aicc"]])
```

```
## [1] 68.49968 60.02357 62.83253 65.45747
```

The best model (with the smallest AICc value) has two lagged predictors; that is, it includes advertising only in the current month and the previous month. So we now re-estimate that model, but using all the available data.

```
(fit <- auto.arima(insurance[,1], xreg=Advert[,1:2],
stationary=TRUE))
```

```
## Series: insurance[, 1]
## Regression with ARIMA(3,0,0) errors
##
## Coefficients:
##          ar1      ar2     ar3  intercept  AdLag0  AdLag1
##       1.4117  -0.9317  0.3591     2.0393  1.2564  0.1625
## s.e.  0.1698   0.2545  0.1592     0.9931  0.0667  0.0591
##
## sigma^2 estimated as 0.2165:  log likelihood=-23.89
## AIC=61.78   AICc=65.4   BIC=73.43
```

The chosen model has AR(3) errors. The model can be written as

$$\begin{aligned} y_t &= 2.039 + 1.256x_t + 0.162x_{t-1} + \eta_t, \\ \eta_t &= 1.41\eta_{t-1} - 0.93\eta_{t-2} + 0.36\eta_{t-3} + \varepsilon_t \end{aligned}$$

where y_t is the number of quotations provided in month t, x_t is the advertising expenditure in month t and ε_t is white noise.

Forecasts can be calculated using this model if the future values for the advertising variable are assumed, to be some value, say, 8 units. The forecasts are shown in Figure 9.13.

```r
fc8 <- forecast(fit, h=20,
xreg=cbind(AdLag0 = rep(8,20),
AdLag1 = c(Advert[40,1], rep(8,19))))
autoplot(fc8) + ylab("Quotes") +
ggtitle("Forecast quotes with future advertising set to 8")
```

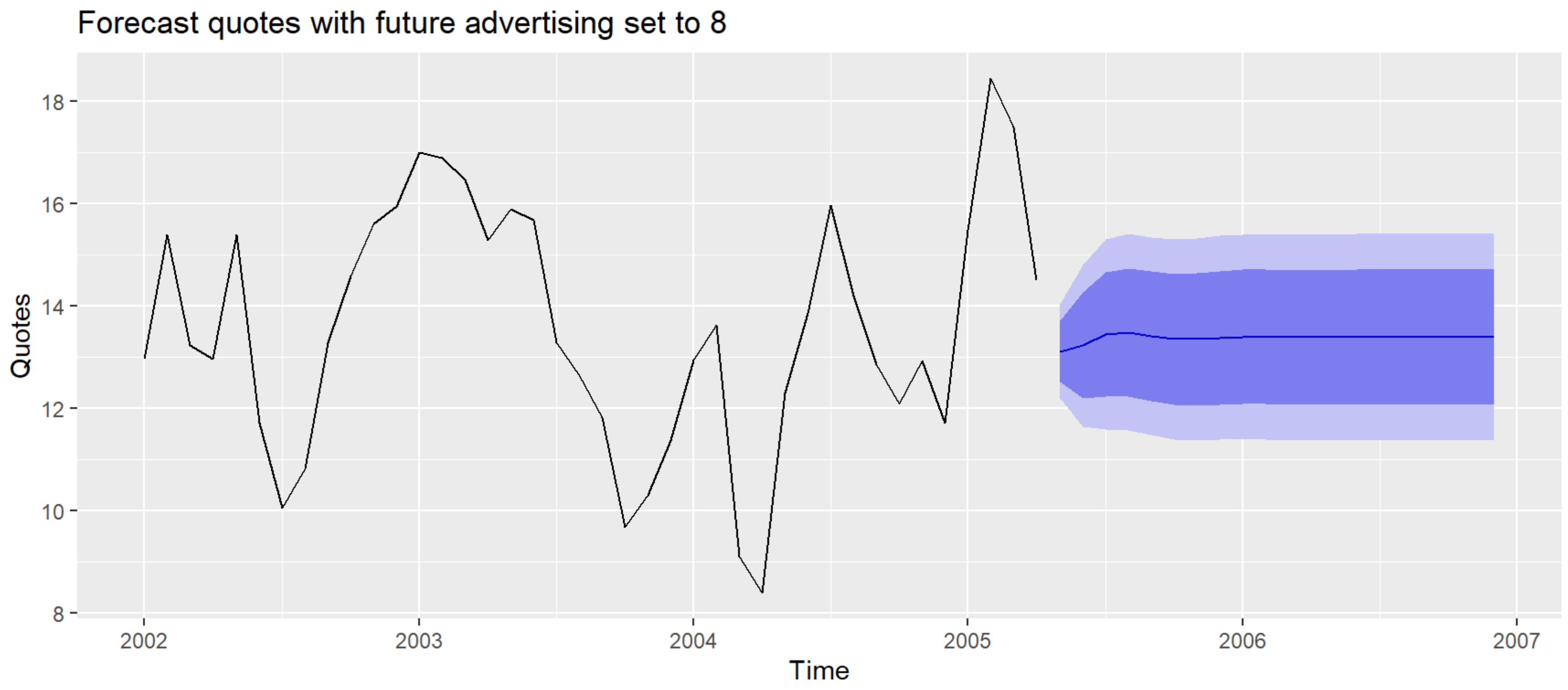


Figure 9.13: Forecasts of monthly insurance quotes, assuming that the future advertising expenditure is 8 units in each future month.

Thank you for your attention!