# Chapter 7: Time series regression models

Ankit Gupta

06/01/2024

In this chapter we discuss regression models. The basic concept is that we forecast the time series of interest `y` assuming that it has a linear relationship with other time series `x`.

For example, we might wish to forecast monthly sales `y` using total advertising spend `x` as a predictor. Or we might forecast daily electricity demand `y` using temperature $x_1$ and the day of week $x_2$ as predictors.

The forecast variable $y$ is sometimes also called the regressand, dependent or explained variable. The predictor variables $x$ are sometimes also called the regressors, independent or explanatory variables. In this book we will always refer to them as the "forecast" variable and "predictor" variables.

## 7.1 The linear model

### Multiple Regression and forecasting

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

- $y_t$ is the variable we want to predict: the "response" variable
- Each $x_{j,t}$ is numerical and called "predictor". They are usually assumed to be known for all past and future times.
- The coefficients $\beta_1, ..., \beta_k$ measure the effect of each predictor after taking account of the effect of all other predictors in the model.
- That is, coefficients measure the *marginal effects*. For example effect of $x_1$ on $y_t$ is measure by taking the account of all other variables.
- $\epsilon_t$ is a white noise error term.

**Example: US Consumption Expenditure (only one predictor i.e. Simple regression)**

y=consumption and x=income

```
library(fpp3)
```

```
## -- Attaching packages ---------------------------------------------- fpp3 0.5 --

## v tibble      3.2.1     v tsibble     1.1.3
## v dplyr       1.1.3     v tsibbledata 0.4.1
## v tidyr       1.3.0     v feasts      0.3.1
## v lubridate   1.9.3     v fable       0.3.3
## v ggplot2     3.4.4     v fabletools  0.3.4

## -- Conflicts ------------------------------------------------- fpp3_conflicts --
## x lubridate::date()    masks base::date()
## x dplyr::filter()      masks stats::filter()
## x tsibble::intersect() masks base::intersect()
## x tsibble::interval()  masks lubridate::interval()
## x dplyr::lag()         masks stats::lag()
## x tsibble::setdiff()   masks base::setdiff()
## x tsibble::union()     masks base::union()
```

```
fit_cons <- us_change %>%
  model(lm=TSLM(Consumption ~ Income))
report(fit_cons)
```

```
## Series: Consumption
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.58236 -0.27777  0.01862  0.32330  1.42229
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.54454    0.05403  10.079  < 2e-16 ***
## Income       0.27183    0.04673   5.817  2.4e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5905 on 196 degrees of freedom
## Multiple R-squared: 0.1472,  Adjusted R-squared: 0.1429
## F-statistic: 33.84 on 1 and 196 DF, p-value: 2.4022e-08
```

Here, “lm” stands for the linear model and TSLM() is a built-in function in fable.

## Multiple Regression

Now x= Income, Production, Savings, Unemployment

Check the plot of each variable which is similar in time due to linear relationship. Also, check the correlation coefficients.

### Assumption for the linear model

For forecasting purposes, we require the following assumptions:

- $\epsilon_t$ have mean zero and are uncorrelated hence efficient means we don’t have any more signal left over in our errors that we should take advantage of.
- $\epsilon_t$ are uncorrelated with each $x_{j,t}$

It is also *useful* to have $\epsilon_t \sim N(0, \sigma^2)$ when producing prediction intervals or doing statistical tests.

Summary

##Simple linear regression

In the simplest case, the regression model allows for a linear relationship between the forecast variable $y$ and a single predictor variable $x : y_t = \beta_0 + \beta_1 x_t + \epsilon_t$.

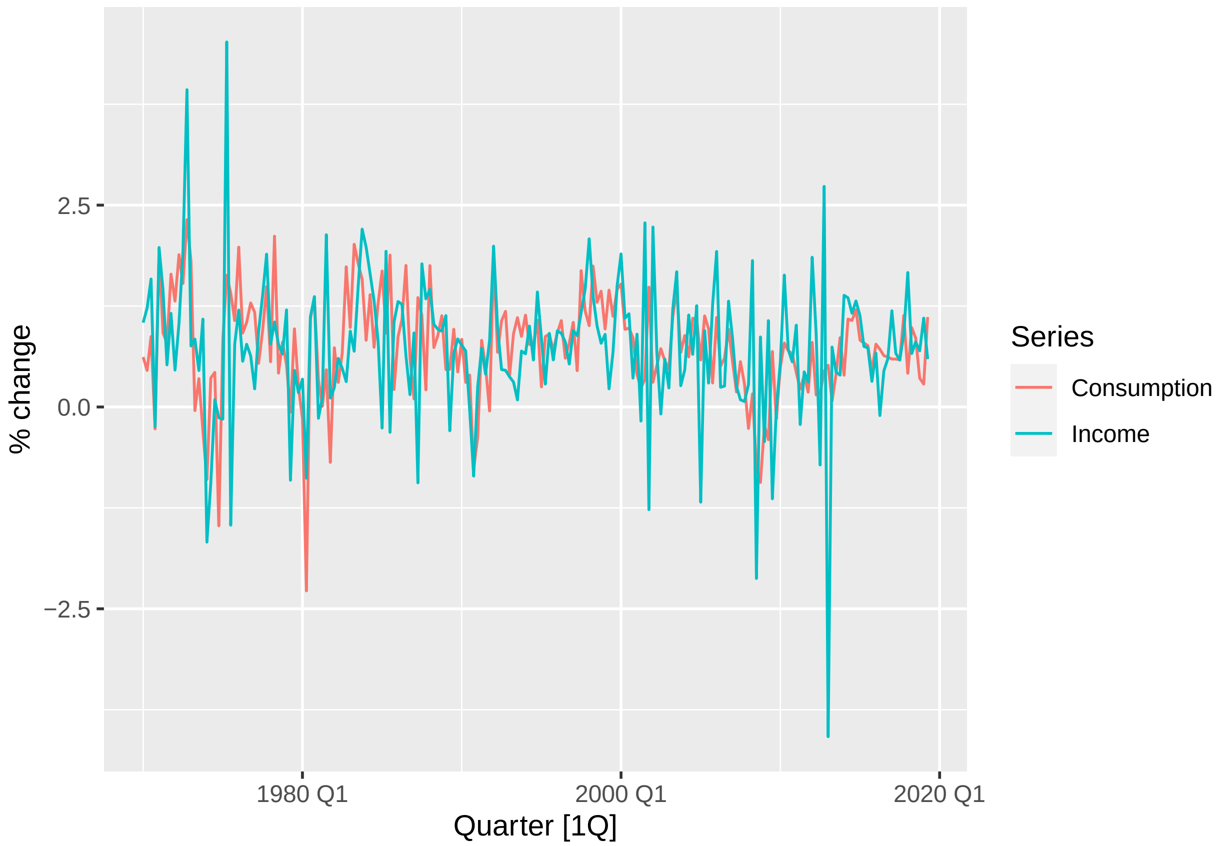
An artificial example of data from such a model is shown in Figure 7.1. The coefficients $\beta_0$ and $\beta_1$ denote the intercept and the slope of the line respectively. The intercept $\beta_0$ represents the predicted value of $y$ when $x = 0$. The slope $\beta_1$ represents the average predicted change in $y$ resulting from a one unit increase in $x$.

Notice that the observations do not lie on the straight line but are scattered around it. We can think of each observation $y_t$ as consisting of the systematic or explained part of the model, $\beta_0 + \beta_1 x_t$, and the random "error", $\epsilon_t$. The "error" term does not imply a mistake, but a deviation from the underlying straight line model. It captures anything that may affect $y_t$ other than $x_t$.

###Example: US consumption expenditure

Figure 7.2 shows time series of quarterly percentage changes (growth rates) of real personal consumption expenditure, $y$, and real personal disposable income, $x$, for the US from 1970 Q1 to 2019 Q2.

```r
us_change |>
  pivot_longer(c(Consumption, Income), names_to="Series") |>
  autoplot(value) +
  labs(y = "% change")
```

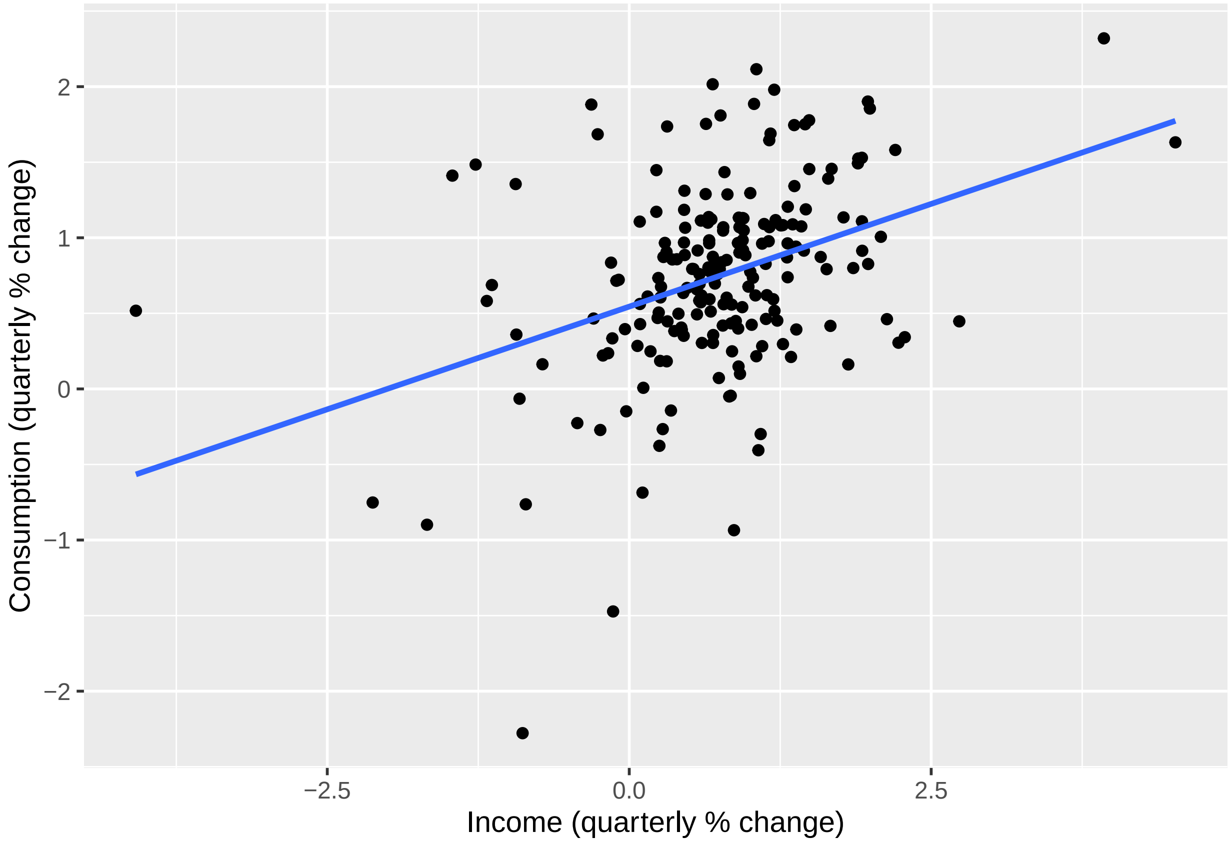


A scatter plot of consumption changes against income changes is shown in Figure 7.3 along with the estimated regression line$\widehat{y}_t = 0.54 + 0.27x_t$.

(We put a "hat" above $y$ to indicate that this is the value of $y$ predicted by the model.)

```r
us_change |>
  ggplot(aes(x = Income, y = Consumption)) +
  labs(y = "Consumption (quarterly % change)",
       x = "Income (quarterly % change)") +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



The equation is estimated using the TSLM() function:

```
us_change |>
  model(TSLM(Consumption ~ Income)) |>
  report()
```

```
## Series: Consumption
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.58236 -0.27777  0.01862  0.32330  1.42229
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.54454    0.05403  10.079  < 2e-16 ***
## Income       0.27183    0.04673   5.817  2.4e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5905 on 196 degrees of freedom
## Multiple R-squared: 0.1472,  Adjusted R-squared: 0.1429
## F-statistic: 33.84 on 1 and 196 DF, p-value: 2.4022e-08
```

We will discuss how TSLM() computes the coefficients in Section 7.2.

The fitted line has a positive slope, reflecting the positive relationship between income and consumption. The slope coefficient shows that a one unit increase in $x$ (a 1 percentage point increase in personal disposable income) results on average in 0.27 units increase in $y$ (an average increase of 0.27 percentage points in personal consumption expenditure). Alternatively the estimated equation shows that a value of 1 for $x$ (the percentage increase in personal disposable income) will result in a forecast value of $0.54 + 0.27 \times 1 = 0.82$ for $y$ (the percentage increase in personal consumption expenditure).

The interpretation of the intercept requires that a value of $x = 0$ makes sense. In this case when $x = 0$ (i.e., when there is no change in personal disposable income since the last quarter) the predicted value of $y$ is 0.54 (i.e., an average increase in personal consumption expenditure of 0.54%). Even when $x = 0$ does not make sense, the intercept is an important part of the model. Without it, the slope coefficient can be distorted unnecessarily. The intercept should always be included unless the requirement is to force the regression line "through the origin". In what follows we assume that an intercept is always included in the model.

### Multiple linear regression

Figure 7.4 shows additional predictors that may be useful for forecasting US consumption expenditure. These are quarterly percentage changes in industrial production and personal savings, and quarterly changes in the unemployment rate (as this is already a percentage). Building a multiple linear regression model can potentially generate more accurate forecasts as we expect consumption expenditure to not only depend on personal income but on other predictors as well.

```r
us_change |>
  select(-Consumption, -Income) |>
  pivot_longer(-Quarter) |>
  ggplot(aes(Quarter, value, colour = name)) +
  geom_line() +
  facet_grid(name ~ ., scales = "free_y") +
  guides(colour = "none") +
  labs(y="% change")
```

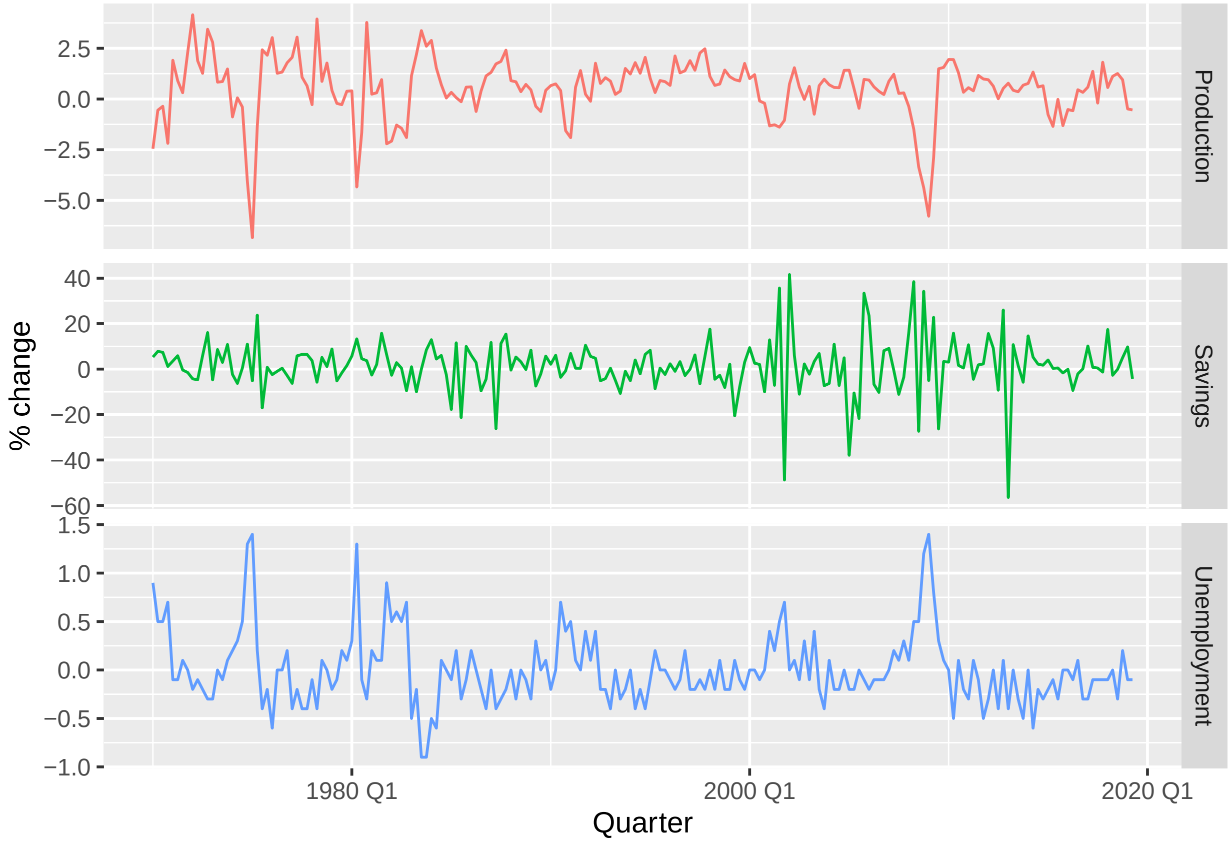
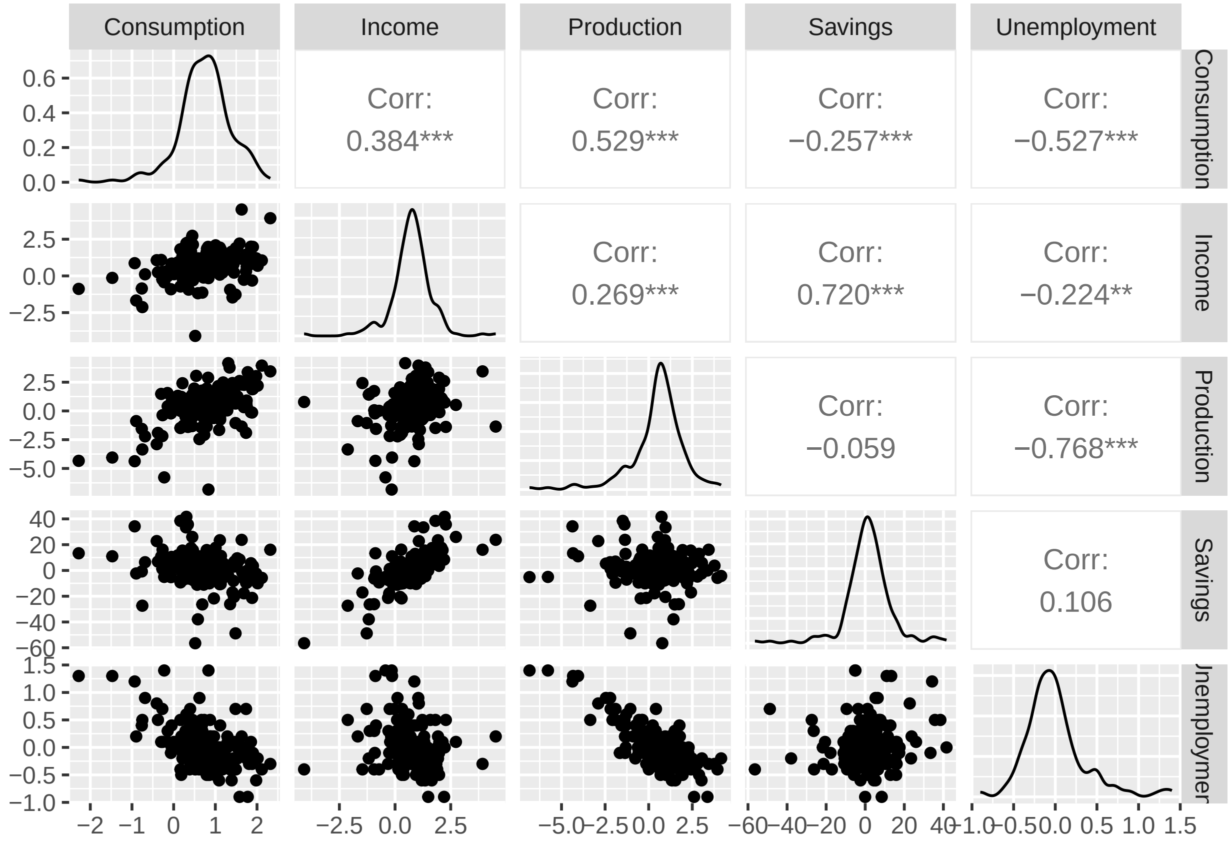


Figure 7.5 is a scatterplot matrix of five variables. The first column shows the relationships between the forecast variable (consumption) and each of the predictors. The scatterplots show positive relationships with income and industrial production, and negative relationships with savings and unemployment. The strength of these relationships are shown by the correlation coefficients across the first row. The remaining scatterplots and correlation coefficients show the relationships between the predictors.

```
us_change |>
  GGally::ggpairs(columns = 2:6)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```



## Assumptions

When we use a linear regression model, we are implicitly making some assumptions about the variables in Equation (7.1).

First, we assume that the model is a reasonable approximation to reality; that is, the relationship between the forecast variable and the predictor variables satisfies this linear equation.

Second, we make the following assumptions about the errors $(\epsilon_1, ..., \epsilon_T)$ :

- they have mean zero; otherwise the forecasts will be systematically biased.
- they are not autocorrelated; otherwise the forecasts will be inefficient, as there is more information in the data that can be exploited.
- they are unrelated to the predictor variables; otherwise there would be more information that should be included in the systematic part of the model.
- It is also useful to have the errors being normally distributed with a constant variance $\sigma^2$ in order to easily produce prediction intervals.

Another important assumption in the linear regression model is that each predictor $x$ is not a random variable. If we were performing a controlled experiment in a laboratory, we could control the values of each $x$ (so they would not be random) and observe the resulting values of $y$. With observational data (including most data in business and economics), it is not possible to control the value of $x$, we simply observe it. Hence we make this an assumption.

## 7.2 Least squares estimation

**Least Squares estimation:**

- In practice, we need to estimate the coefficients: $\beta_0, \beta_1, ..., \beta_k$
- Now, we need to minimize the following errors:

$ _{t=1}^{T} t^2 = {t=1}{T}(y_t- _0- 1x{1,t}- 2x{2,t}-...- kx{k,t})^2 $

- Here we use model(TSML($y \sim x_1 + x_2 + ...x_k$)) and it returns the set of estimated coefficients $\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_k$

**Example: US Consumption Expenditure**

```
fit_consMR <- us_change |>
  model(lm=TSLM(Consumption ~ Income + Production + Unemployment + Savings))
report(fit_consMR)
```

```
## Series: Consumption
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.90555 -0.15821 -0.03608  0.13618  1.15471
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.253105   0.034470   7.343 5.71e-12 ***
## Income        0.740583   0.040115  18.461  < 2e-16 ***
## Production    0.047173   0.023142   2.038   0.0429 *
## Unemployment -0.174685   0.095511  -1.829   0.0689 .
## Savings      -0.052890   0.002924 -18.088  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3102 on 193 degrees of freedom
## Multiple R-squared: 0.7683,  Adjusted R-squared: 0.7635
## F-statistic:   160 on 4 and 193 DF, p-value: < 2.22e-16
```

Here intercept is automatically estimated unless we use $y \sim 0 + ...$ and here we don't use the inferences like t-value or p-value etc.

So, we get the fitted values from $\hat{y}_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{12,t} + ... + \beta_k x_{k,t}$

**Goodness of fit**

Official ways of thinking about the fit of the model to the data.

**1. Coefficient of determination: R-squared**

$$R^2 = \frac{\Sigma(\hat{y}_t - \bar{y})^2}{(y_t - \bar{y})^2}$$

$R_2$ is defined as the ratio of the variance of the fitted values and variance of the data.

### 2. **Standard error of the regression**

$$\sigma_e = \sqrt{\frac{1}{T-k-1}\Sigma_{t=1}^{T} e_t^2}$$

Where $k$ is the number of predictors in the model. It is related to the size of the average error that the model produces. It also accounts for the degrees of freedom because it includes the number of predictors but $R^2$ does not.

Generally people use these 2 measures initially for the goodness of the fit.

```r
fit_consMR <- us_change |>
  model(lm=TSLM(Consumption ~ Income + Production + Unemployment + Savings))
report(fit_consMR)
```

```
## Series: Consumption
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.90555 -0.15821 -0.03608  0.13618  1.15471
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.253105   0.034470   7.343 5.71e-12 ***
## Income        0.740583   0.040115  18.461  < 2e-16 ***
## Production    0.047173   0.023142   2.038   0.0429 *
## Unemployment -0.174685   0.095511  -1.829   0.0689 .
## Savings      -0.052890   0.002924 -18.088  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3102 on 193 degrees of freedom
## Multiple R-squared: 0.7683,   Adjusted R-squared: 0.7635
## F-statistic:   160 on 4 and 193 DF, p-value: < 2.22e-16
```

Here, $R^2 = 0.768$ and average error produced is 0.3102. In this case, the model does an excellent job as it explains 76.8% of the variation in the consumption data. Compare that to the $R^2$ value of 0.15 obtained from the simple regression with the same data set in Section 7.1. Adding the three extra predictors has allowed a lot more of the variation in the consumption data to be explained.

In standard error, Notice that we divide by $T-k-1$ because we have estimated $k+1$ parameters (the intercept and a coefficient for each predictor variable) in computing the residuals.

The standard error is related to the size of the average error that the model produces. We can compare this error to the sample mean of $y$ or with the standard deviation of $y$ to gain some perspective on the accuracy of the model. The standard error will be used when generating prediction intervals, discussed in Section 7.6.

Summary

The least squares principle provides a way of choosing the coefficients effectively by minimising the sum of the squared errors. That is, we choose the values of $\beta_0, \beta_1, ..., \beta_k$ that minimise $\Sigma_{t=1}^{T}\epsilon_t^2$.

This is called least squares estimation because it gives the least value for the sum of squared errors. Finding the best estimates of the coefficients is often called “fitting” the model to the data, or sometimes “learning” or “training” the model.

The TSLM() function fits a linear regression model to time series data. It is similar to the lm() function which is widely used for linear models, but TSLM() provides additional facilities for handling time series standard error (i.e., the standard deviation which would be obtained from repeatedly estimating the coefficients on similar data sets). The standard error gives a measure of the uncertainty in the estimated coefficient.

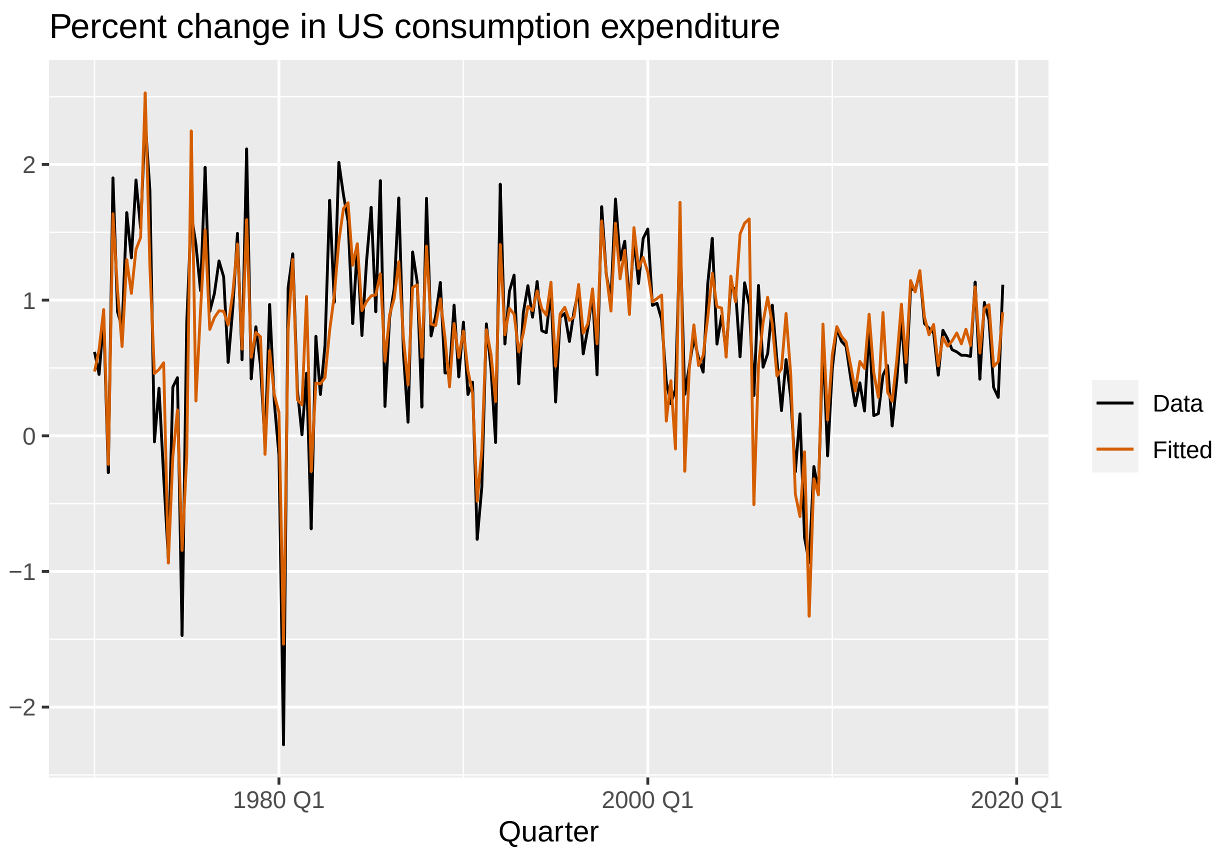
For forecasting purposes, the final two columns are of limited interest. The “t value” is the ratio of an estimated coefficient to its standard error and the last column gives the p-value: the probability of the estimated coefficient being as large as it is if there was no real relationship between consumption and the corresponding predictor. This is useful when studying the effect of each predictor, but is not particularly useful for forecasting.

Predictions of y can be obtained by using the estimated coefficients in the regression equation and setting the error term to zero.

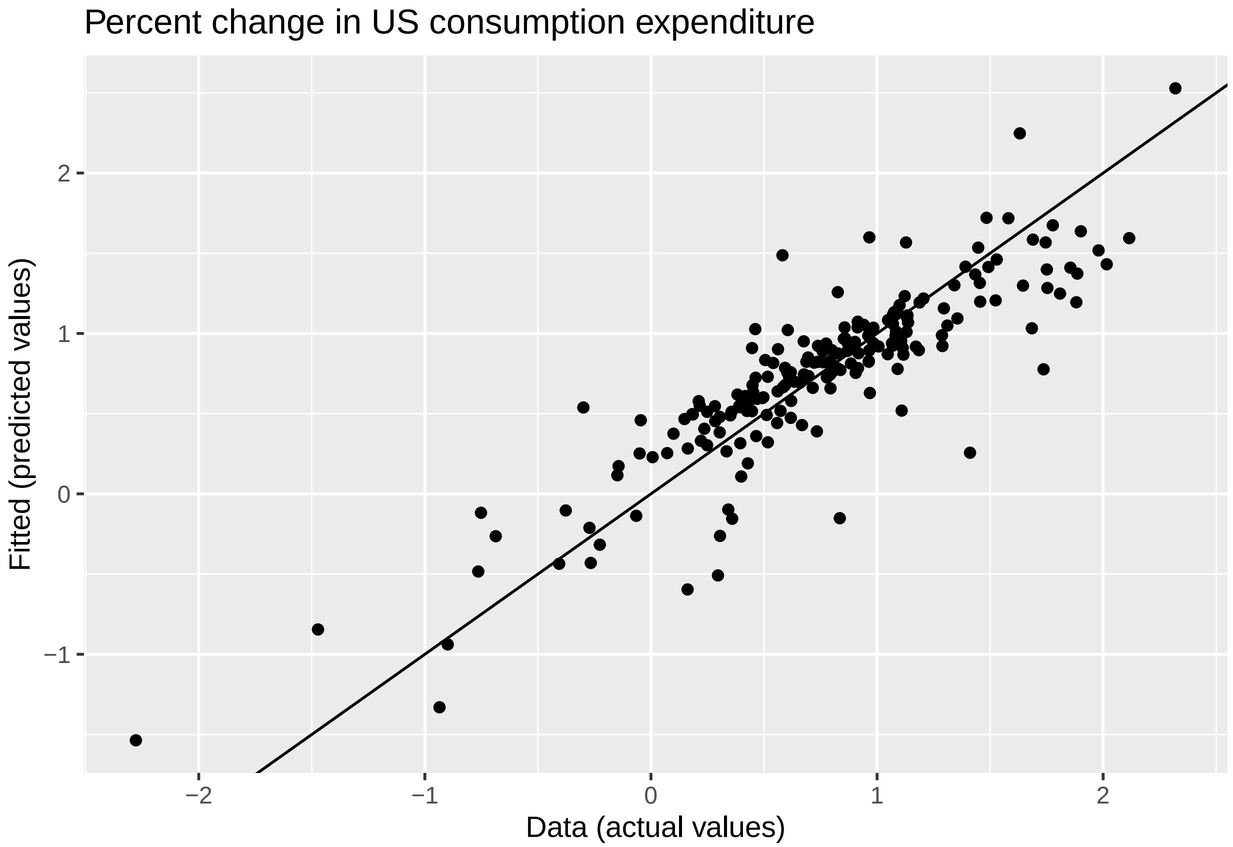
Plugging in the values of $x_{1,t}, ..., x_{k,t}$ for $t = 1, ..., T$ returns predictions of
$y_t$ within the training set, referred to as fitted values. Note that these are predictions of the data used to estimate the model, not genuine forecasts of future values of y.

The following plots show the actual values compared to the fitted values for the percentage change in the US consumption expenditure series. The time plot in Figure 7.6 shows that the fitted values follow the actual data fairly closely. This is verified by the strong positive relationship shown by the scatterplot in Figure 7.7.

```r
augment(fit_consMR) |>
  ggplot(aes(x = Quarter)) +
  geom_line(aes(y = Consumption, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  labs(y = NULL,
    title = "Percent change in US consumption expenditure"
  ) +
  scale_colour_manual(values=c(Data="black",Fitted="#D55E00")) +
  guides(colour = guide_legend(title = NULL))
```



```r
augment(fit_consMR) |>
  ggplot(aes(x = Consumption, y = .fitted)) +
  geom_point() +
  labs(
    y = "Fitted (predicted values)",
    x = "Data (actual values)",
    title = "Percent change in US consumption expenditure"
  ) +
  geom_abline(intercept = 0, slope = 1)
```



**Goodness-of-fit**

A common way to summarise how well a linear regression model fits the data is via the coefficient of determination, or $R^2$. This can be calculated as the square of the correlation between the observed y values and the predicted $\hat{y}$ values.

Alternatively, it can also be calculated as,

$$R^2 = \frac{\Sigma(\hat{y}_t - \bar{y})^2}{(y_t - \bar{y})^2}$$

where the summations are over all observations. Thus, it reflects the proportion of variation in the forecast variable that is accounted for (or explained) by the regression model.

In simple linear regression, the value of $R^2$ is also equal to the square of the correlation between y and x (provided an intercept has been included).

If the predictions are close to the actual values, we would expect $R^2$ to be close to 1. On the other hand, if the predictions are unrelated to the actual values, then
$R^2 = 0$ (again, assuming there is an intercept). In all cases, $R^2$ lies between 0 and 1.

The $R^2$ value is used frequently, though often incorrectly, in forecasting. The value of $R^2$ will never decrease when adding an extra predictor to the model and this can lead to over-fitting. There are no set rules for what is a good $R^2$ value, and typical values of $R^2$ depend on the type of data used. Validating a model's forecasting performance on the test data is much better than measuring the $R^2$ value on the training data.

## 7.3 Evaluating the regression model

### Regression Residuals

Residuals are defined as

$e_t = y_t - \hat{y}_t = y_t - \hat{\beta}_0 - \hat{\beta}_1 x_{1,t} - \hat{\beta}_2 x_{2,t} - \ldots - \hat{\beta}_k x_{k,t}$

Useful properties

(1) $\Sigma_{t=1}^{T} e_t = 0$ and

(2) $\Sigma_{t=1}^{T} x_{k,t} e_t = 0$ for all $k$

As a result of these properties, it is clear that the average of the residuals is zero, and that the correlation between the residuals and the observations for the predictor variable is also zero. (This is not necessarily true when the intercept is omitted from the model.)

After selecting the regression variables and fitting a regression model, it is necessary to plot the residuals to check that the assumptions of the model have been satisfied. There are a series of plots that should be produced in order to check different aspects of the fitted model and the underlying assumptions. We will now discuss each of them in turn.

### ACF plot of residuals

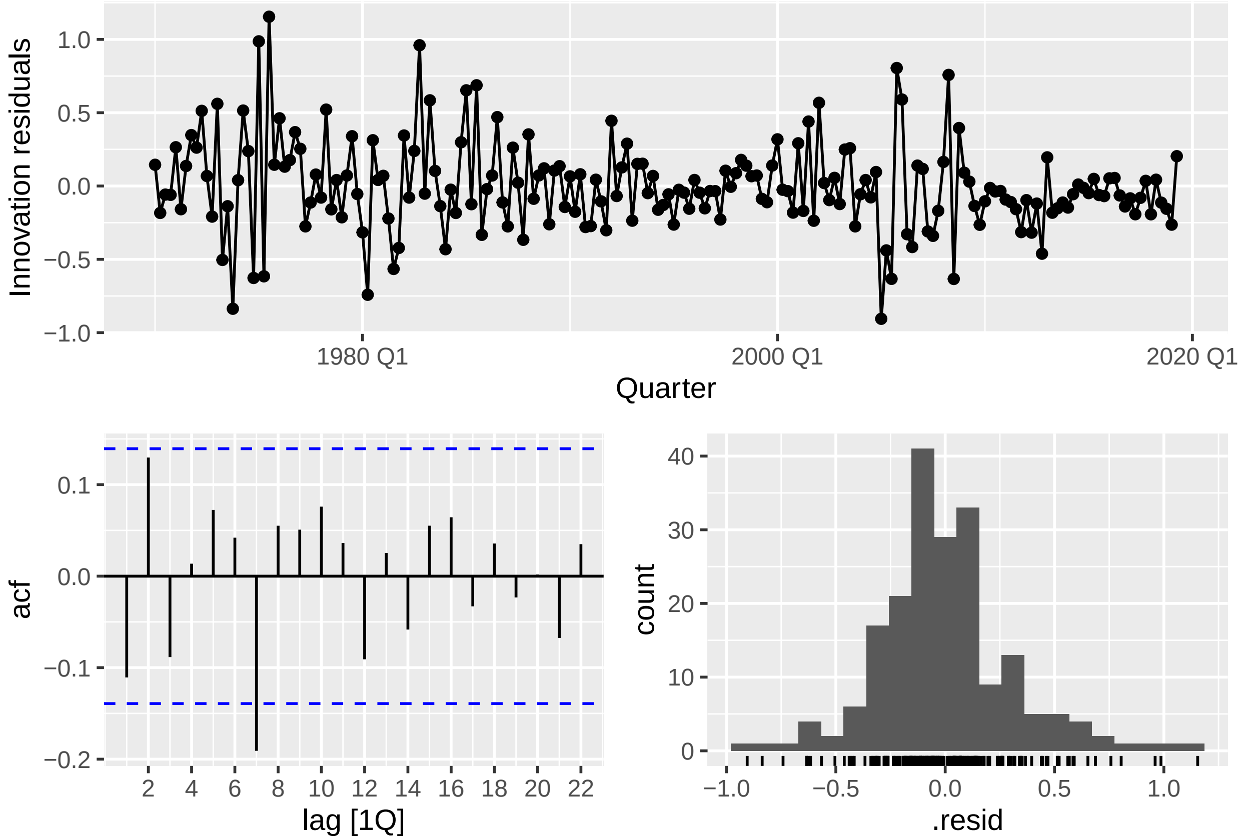
With time series data, it is highly likely that the value of a variable observed in the current time period will be similar to its value in the previous period, or even the period before that, and so on. Therefore when fitting a regression model to time series data, it is common to find autocorrelation in the residuals. In this case, the estimated model violates the assumption of no autocorrelation in the errors, and our forecasts may be inefficient — there is some information left over which should be accounted for in the model in order to obtain better forecasts. The forecasts from a model with autocorrelated errors are still unbiased, and so they are not "wrong", but they will usually have larger prediction intervals than they need to. Therefore we should always look at an ACF plot of the residuals.

### Histogram of residuals

It is always a good idea to check whether the residuals are normally distributed. As we explained earlier, this is not essential for forecasting, but it does make the calculation of prediction intervals much easier.

**Example** Using the `gg_tsresiduals()` function introduced in Section 5.3, we can obtain all the useful residual diagnostics mentioned above.

```
fit_consMR |> gg_tsresiduals()
```



```r
augment(fit_consMR) |>
  features(.innov, ljung_box, lag = 10)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 lm        18.9    0.0420
```

The time plot shows some changing variation over time, but is otherwise relatively unremarkable. This heteroscedasticity will potentially make the prediction interval coverage inaccurate.

The histogram shows that the residuals seem to be slightly skewed, which may also affect the coverage probability of the prediction intervals.

The autocorrelation plot shows a significant spike at lag 7, and a significant Ljung-Box test at the 5% level. However, the autocorrelation is not particularly large, and at lag 7 it is unlikely to have any noticeable impact on the forecasts or the prediction intervals. In Chapter 10 we discuss dynamic regression models used for better capturing information left in the residuals.

### Checking Assumptions

- $\epsilon_t$ have mean zero and are uncorrelated, NID$(0, \sigma^2)$.
- $\epsilon_t$ are uncorrelated with each $x_{j,t}$

i. Timeplot, ACF, Histogram(gg_tsresiduals())

ii. Against predictors (non-linearity)

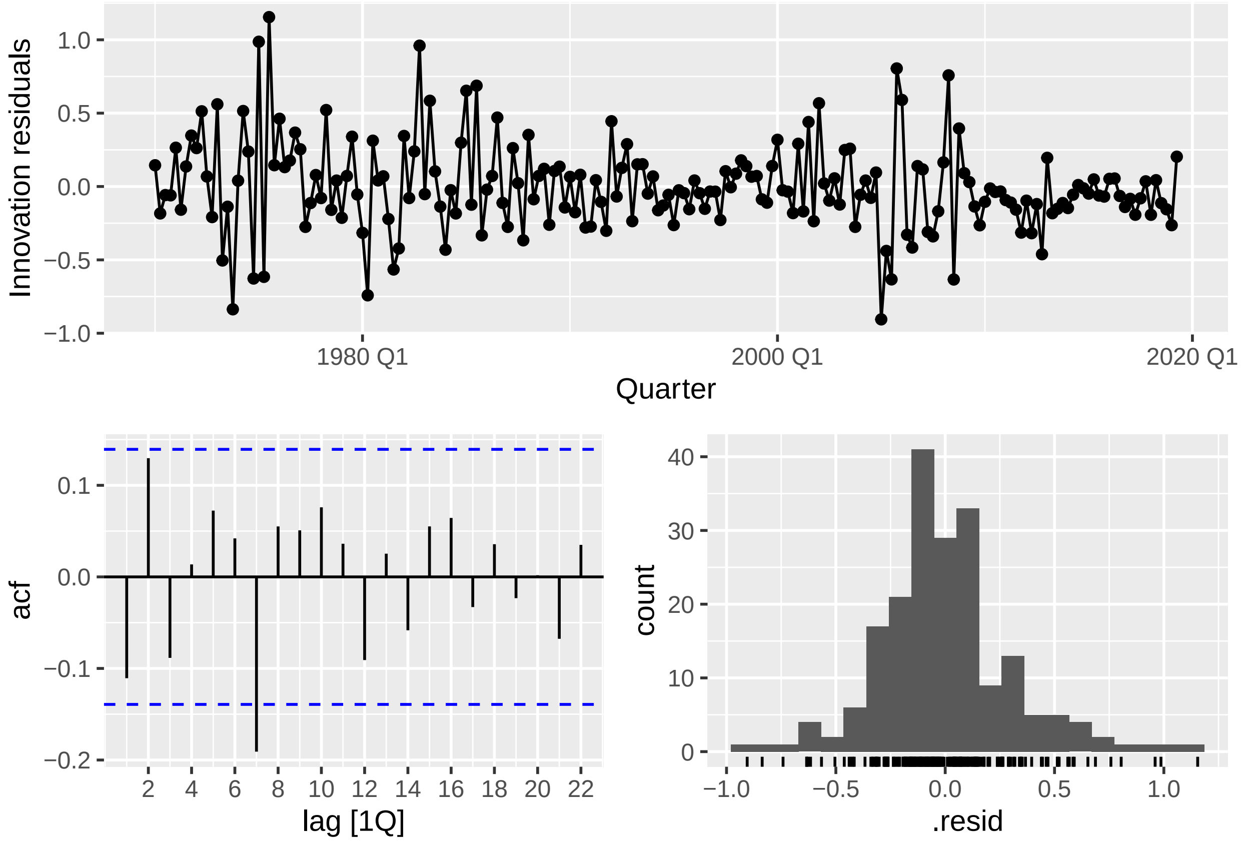
iii. Against fitted values (heteroscedasticity)

iv. Against predictors not in the model (include predictor in the model)

Expect to see scatterplots resembling a horizontal band with no values too far from the band and no patterns such as curvature or increasing spread.

### Example: US Consumption Expenditure

```
fit_consMR |> gg_tsresiduals()
```



### Residual plots against predictors

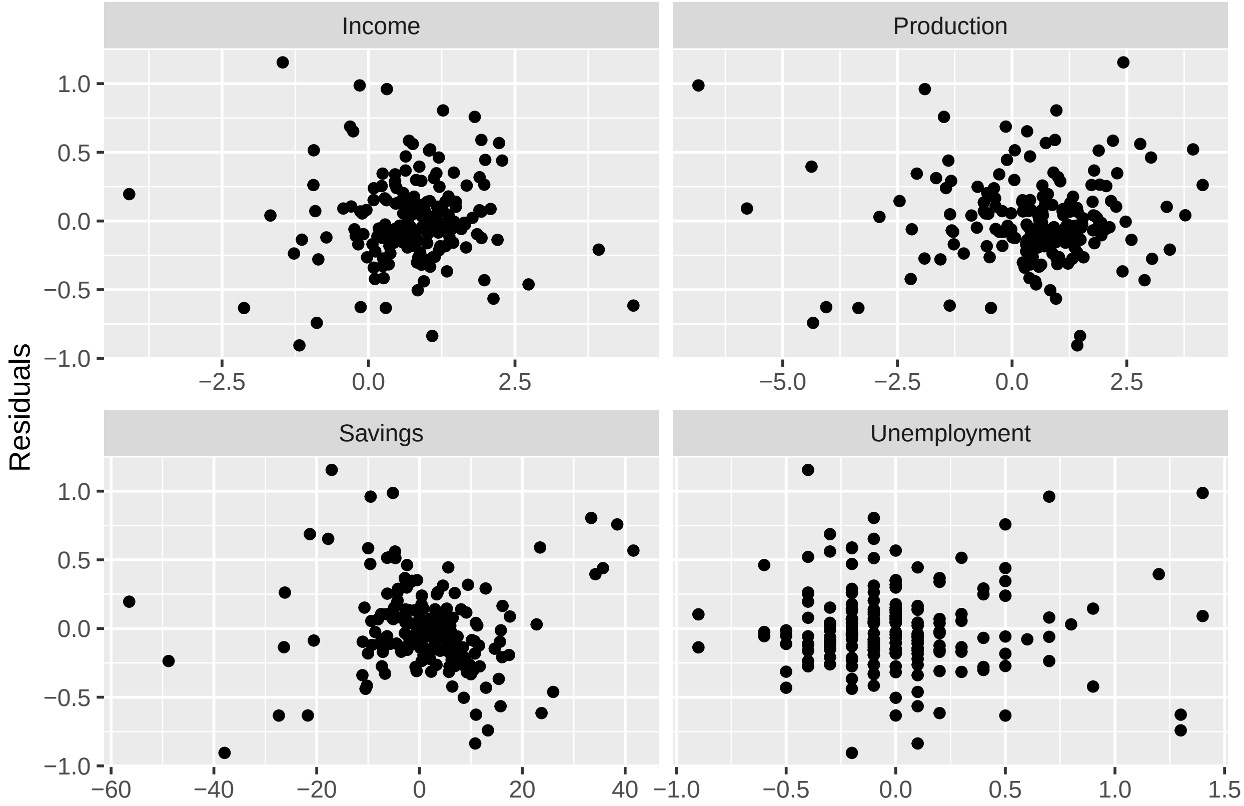
We would expect the residuals to be randomly scattered without showing any systematic patterns. A simple and quick way to check this is to examine scatterplots of the residuals against each of the predictor variables. If these scatterplots show a pattern, then the relationship may be nonlinear and the model will need to be modified accordingly. See Section 7.7 for a discussion of nonlinear regression.

It is also necessary to plot the residuals against any predictors that are not in the model. If any of these show a pattern, then the corresponding predictor may need to be added to the model (possibly in a nonlinear form).

### Example

The residuals from the multiple regression model for forecasting US consumption plotted against each predictor in Figure 7.9 seem to be randomly scattered. Therefore we are satisfied with these in this case.

```
us_change |>
  left_join(residuals(fit_consMR), by = "Quarter") |>
  pivot_longer(Income:Unemployment,
               names_to = "regressor", values_to = "x") |>
  ggplot(aes(x = x, y = .resid)) +
  geom_point() +
  facet_wrap(. ~ regressor, scales = "free_x") +
  labs(y = "Residuals", x = "")
```

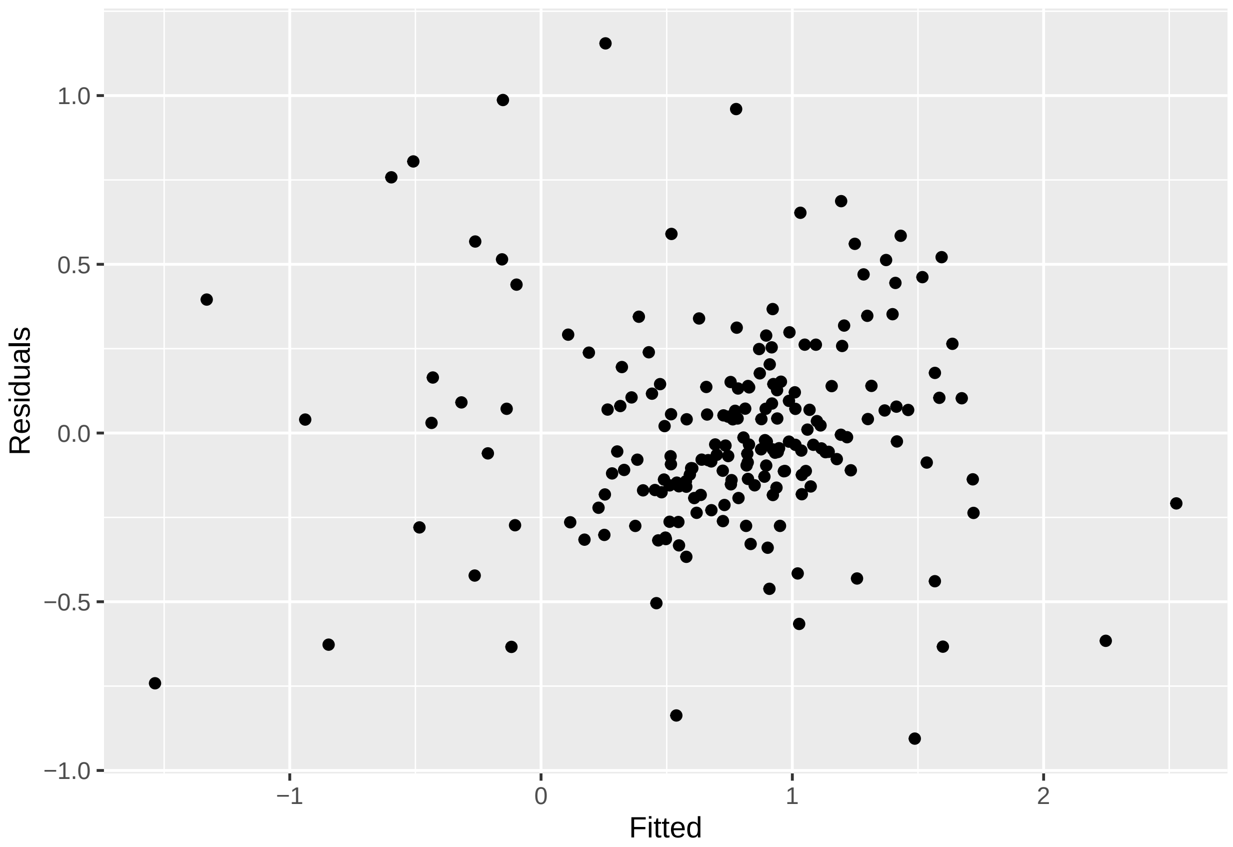


**Residual plots against fitted values** A plot of the residuals against the fitted values should also show no pattern. If a pattern is observed, there may be “heteroscedasticity” in the errors which means that the variance of the residuals may not be constant. If this problem occurs, a transformation of the forecast variable such as a logarithm or square root may be required (see Section 3.1).

**Example**

Continuing the previous example, Figure 7.10 shows the residuals plotted against the fitted values. The random scatter suggests the errors are homoscedastic.

```
augment(fit_consMR) |>
  ggplot(aes(x = .fitted, y = .resid)) +
  geom_point() + labs(x = "Fitted", y = "Residuals")
```



**Outliers and influential observations** Observations that take extreme values compared to the majority of the data are called outliers. Observations that have a large influence on the estimated coefficients of a regression model are called influential observations. Usually, influential observations are also outliers that are extreme in the $x$ direction.

There are formal methods for detecting outliers and influential observations that are beyond the scope of this textbook. As we suggested at the beginning of Chapter 2, becoming familiar with your data prior to performing any analysis is of vital importance. A scatter plot of $y$ against each $x$ is always a useful starting point in regression analysis, and often helps to identify unusual observations.

One source of outliers is incorrect data entry. Simple descriptive statistics of your data can identify minima and maxima that are not sensible. If such an observation is identified, and it has been recorded incorrectly, it should be corrected or removed from the sample immediately.

Outliers also occur when some observations are simply different. In this case it may not be wise for these observations to be removed. If an observation has been identified as a likely outlier, it is important to study it and analyse the possible reasons behind it. The decision to remove or retain an observation can be a challenging one (especially when outliers are influential observations). It is wise to report results both with and without the removal of such observations.

**Example** Figure 7.11 highlights the effect of a single outlier when regressing US consumption on income (the example introduced in Section 7.1). In the left panel the outlier is only extreme in the direction of y, as the percentage change in consumption has been incorrectly recorded as -4%. The orange line is the regression line fitted to the data which includes the outlier, compared to the black line which is the line fitted to the data without the outlier. In the right panel the outlier now is also extreme in the direction of x with the 4% decrease in consumption corresponding to a 6% increase in income. In this case the outlier is extremely influential as the orange line now deviates substantially from the black line.

### Spurious regression

More often than not, time series data are “non-stationary”; that is, the values of the time series do not fluctuate around a constant mean or with a constant variance. We will deal with time series stationarity in more detail in Chapter 9, but here we need to address the effect that non-stationary data can have on regression models.

For example, consider the two variables plotted in Figure 7.12. These appear to be related simply because they both trend upwards in the same manner. However, air passenger traffic in Australia has nothing to do with rice production in Guinea.

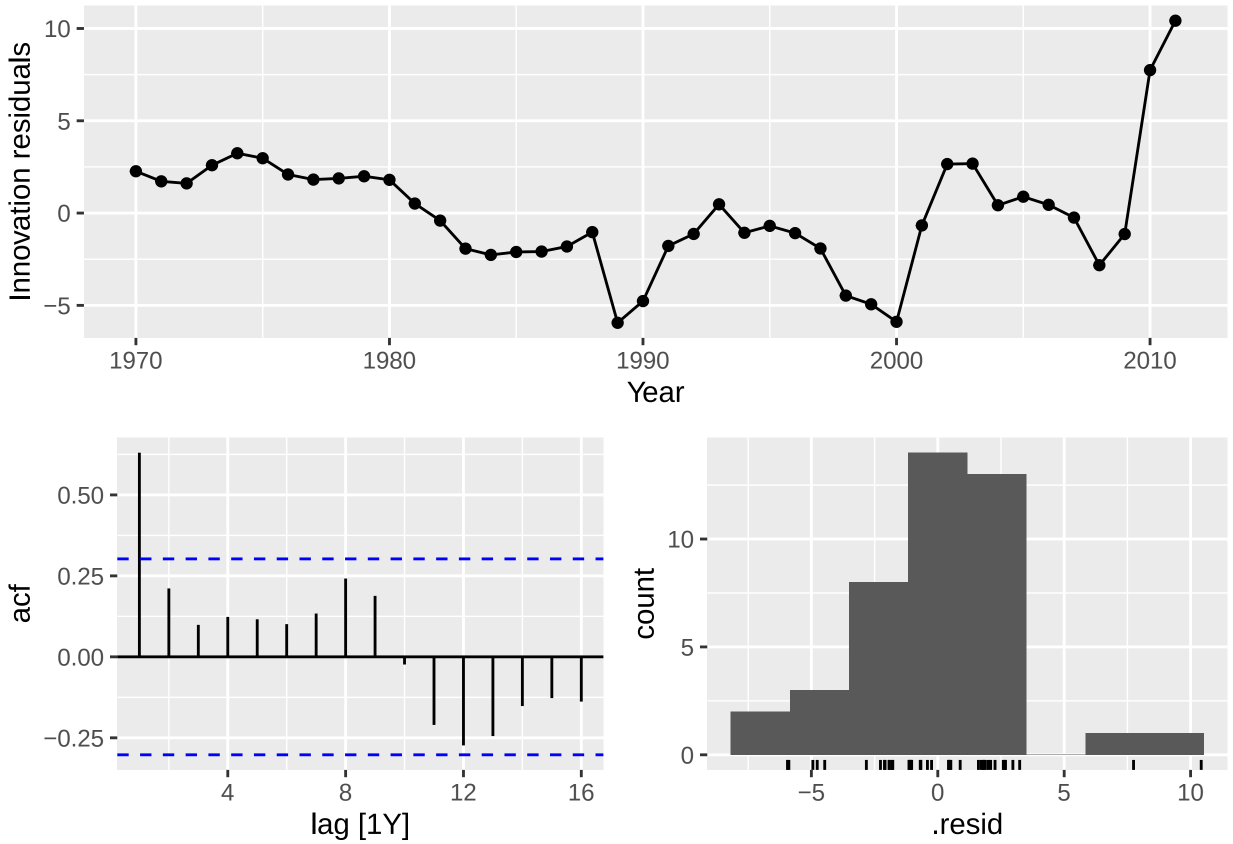
Regressing non-stationary time series can lead to spurious regressions. The output of regressing Australian air passengers on rice production in Guinea is shown in Figure 7.13. High $R^2$ and high residual autocorrelation can be signs of spurious regression. Notice these features in the output below. We discuss the issues surrounding non-stationary data and spurious regressions in more detail in Chapter 10.

Cases of spurious regression might appear to give reasonable short-term forecasts, but they will generally not continue to work into the future.

```r
fit <- aus_airpassengers |>
  filter(Year <= 2011) |>
  left_join(guinea_rice, by = "Year") |>
  model(TSLM(Passengers ~ Production))
report(fit)
```

```
## Series: Passengers
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9448 -1.8917 -0.3272  1.8620 10.4210
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -7.493      1.203  -6.229 2.25e-07 ***
## Production    40.288      1.337  30.135  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.239 on 40 degrees of freedom
## Multiple R-squared: 0.9578,  Adjusted R-squared: 0.9568
## F-statistic: 908.1 on 1 and 40 DF, p-value: < 2.22e-16
```

```r
fit |> gg_tsresiduals()
```



## 7.4 Some useful predictors

### Trend

This is most obvious one.

### (1) Linear Trend

$$x_t = t$$

- t=1,2,3,...,T
- It simply set the predictor to be equal to the number of observations since the start of the series.
- Strong assumption that trend will continue.

**(2) Dummy variables** If categorical variable takes only two values (e.g. “Yes”, or “No”) then an equivalent numerical variable can be constructed taking values 1 if yes and 0 if no. This is called a *dummy variable*.

If there are more than two categories then the variable can be coded using several dummy variables (one fewer than the total number of categories).

### Beware of dummy variable trap

- Using one dummy for each category gives too many dummy variables.
- Since due to rank deficiency, matrix becomes singular and so the regression will then be singular and inestimable.
- Either omit the constant, or omit the dummy for one category.

- The coefficients of the dummies are relative to the omitted category.

## Uses of dummy variables

### Seasonal dummies

- For quarterly data: use 3 dummies
- For monthly data: use 11 dummies
- For daily data: use 6 dummies
- What to do with weekly data ? – It is problematic because there are not a integer number of weeks in a year so treating weeks as a 52 category variable is actually not correct because you are going to get the days at the end that are not part of any of the first 52 weeks of the year and so we need another solution.

### Outliers

- If there is an outlier, you can use a dummy variable to remove its effect. Suppose you have an outlier, you can put in a dummy variable where it's 1 on the time when outlier occurs and zero everywhere else.

### Public holidays

- For daily data: if it is a public holiday, dummy=1, otherwise dummy=0.

#### Example: Beer Production

```
#fit_beer <- recent_production |> model(TSLM(Beer ~ trend()+season()))
#report(fit_beer)
```

Here, our dataset does not actually contains trend or seasonal dummy variables but we can create them on the fly using trend() and season() functions. It will automatically create a trend variable and included in the regression and season() will also create the seasonal variable, it will see the data for example, if it is quarterly data and so it will create the 3 seasonal variables which can be seen in above output. In above output, trend() has negative coefficient which means beer production is decreasing by 0.34 every quarter on average over the series. season() year2 means quarter 2 for every year and similarly season() year3 means quarter 3 for every year. So, say for quarter 3 , if it is -17.0215 it means it is 17.0215 is less than quarter 1.

Now, let's look at the fitted values for this example.

```
#augment(fit_beer) |>
#  ggplot(aes(x=Quarter))+
#  geom_line(aes(y=Beer, colour="Data"))+
#  geom_line(aes(y=.fitted,colour="Fitted"))+
#  labs(y="Megalitres",title = "Australian quarterly beer production")+
#  scale_colour_manual(values=c(Data="black",Fitted="#D55E00"))
```

It looks like it fitted very well. For last part of the data, peaks have overestimate the data whereas initially peaks underestimate the data. So, it's a suggestion that model may not be right.

```
#augment(fit_beer) |>
#  ggplot(aes(x=Beer, y=.fitted,colour=factor(quarter(Quarter))))+
#  geom_point()+
#  labs(y="Fitted",x="Actual Values",title = "Quarterly beer production")+
#  scale_color_brewer(palette = "Dark2", name="Quarter")+
#  geom_abline(intercept = 0,slope=1)
```

```
#fit_beer |> gg_tsresiduals()
```

Here, ACF and the histogram looks okay.

Now comes to the forecast.

```
#fit_beer |>
#  forecast() |>
#  autoplot(recent_production)
```

Here, notice that when we are doing the forecasting, we don't have to tell what future values of the trend are and what future values of the dummy seasonal values are. It will create them on the fly because we created them using functions like trend() and season().

## Intervention Variables

It means something external has changed the output for a particular period. It might generate a spike.

### Spikes

- Equivalent to a dummy variable for handling an outlier.

### Steps:

- Variables take values 0 before the intervention and 1 afterwards.

### Change of slope

We might have the trend and we might want the trend to change the slope

- Variables take values 0 before the intervention and values $1, 2, 3, \ldots$ afterwards.

## Holidays variables

### For monthly data

If holiday moves and may not be in the same month then we have to use the dummy variable.

- Christmas: always in december, so part of monthly seasonal effect and no need of dummy variable.
- Easter: use a dummy variable $v_t = 1$ if any part of Easter is in that month, $v_t = 0$ otherwise
- Ramadan and Chinese new year work similarly.

## Distributed lags

Lagged values of a predictor

Example: $x$ is advertising which has a delayed effect

$x_1 = $ advertising for previous month

$x_2 = $ advertising for two months previously

....

$x_m = $ advertising for $m$ months previously

Here, we distribute the effect of the predictor over a number of lags.

### Fourier Series

It is particularly useful when we got long periods or if we have got non-integer periods.

Periodic seasonality can be handled using pairs of Fourier terms:

$s_k(t) = \sin(\frac{2\pi kt}{m})$ and $c_k(t) = \cos(\frac{2\pi kt}{m})$

where $m$ is the length of the period. So, if we have got hourly data then $m = 24$ so number of times we observe something before it starts to repeat.$K$ is an index. It can be 1,2 or 3 etc.

$y_t = a + bt + \Sigma_{k=1}^{K}[\alpha_k s_k(t) + \beta_k c_k(t)] + \epsilon_t$

Here, in the above regression, we have a linear trend $a + bt$ and we need certain pair of sin and cos terms. It can be 1,2 or 3 etc.

- Every periodic function can be approximated by sums of sin and cos terms for large enough $K$. So we have to pick large enough $K$ to handle the pattern in the data.
- Choose $K$ by minimizing AICc.
- Called "harmonic regression" because it is modelling harmonics so each pair of sin and cos gives you another harmonics. It is analogous to what happens with sound.
- TSLM(y ~ trend() + fourier(K))

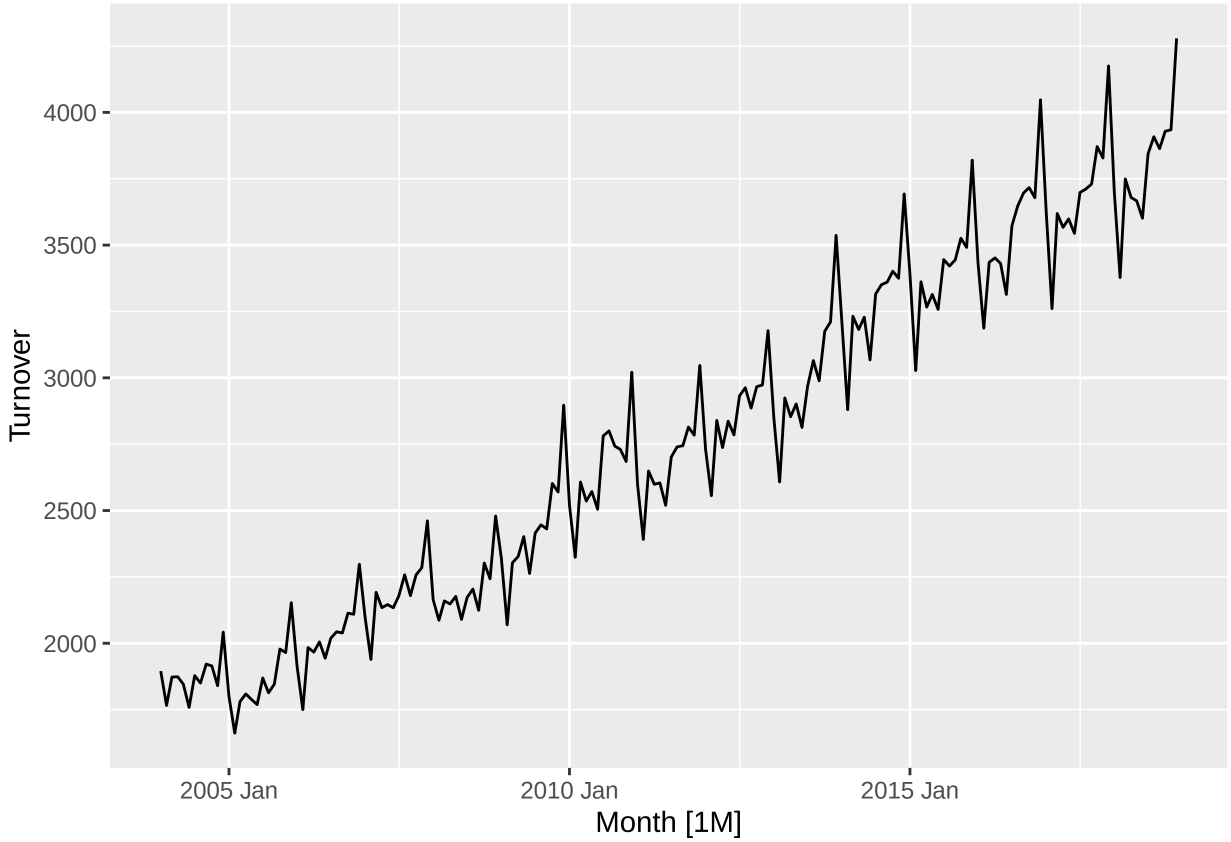
### Example: Harmonic Regression: Beer Production

```
#fourier_beer <- recent_production |> model(TSLM(Beer ~ trend()+fourier(K=2)))
#report(fourier_beer)
```

Here we are using $K = 2$ so we got 4 extra coefficients but since we have got 4 quarters and so we have to drop 1 sin coeffcient i.e. $S_2$ in the output of the above code, in fact it is equal to zero. Here forecast will be identical to the forecast using seasonal dummies.

### Example 2: Harmonic Regression: 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)
```



It's trended and here seasonality is relatively complicated. Now, let's see how to handle this using sin and cos terms.

```
fit <- aus_cafe  |>
  model(
    K1 = TSLM(log(Turnover) ~ trend() + fourier(K=1)),
    K2 = TSLM(log(Turnover) ~ trend() + fourier(K=2)),
    K3 = TSLM(log(Turnover) ~ trend() + fourier(K=3)),
    K4 = TSLM(log(Turnover) ~ trend() + fourier(K=4)),
    K5 = TSLM(log(Turnover) ~ trend() + fourier(K=5)),
    K6 = TSLM(log(Turnover) ~ trend() + fourier(K=6))
  )
```

Here, largest K=12 because there are 12 months in a year so by taking K=6, we get the 12 coefficients and we need 11 coefficients to handle it and so we drop the last coefficient.

So, various terms in fourier series helps to handle seasonal patterns and we can approximate any periodic function by the sums of sine and cosine terms.

So,

- Periodic seasonality can be handled using pairs of Fourier terms.
- Every periodic function can be approximated by sums of sin and cos terms for large enough $K$
- $K \leq m/2$
- $m$ can be non-integer
- Particularly useful for large $m$.

- More importantly, we have non-integer $m$, so we have weekly data where $m = 52.18$ which is little more than 52 weeks in a year. So we put the correct $m$ i.e. average number of weeks in a year and it will work fine, so it is a way to handle the weekly data and none of the other ways can handle weekly data properly because in previous models, $m$ has to be an integer.

Summary

## Dummy variables

So far, we have assumed that each predictor takes numerical values. But what about when a predictor is a categorical variable taking only two values (e.g., "yes" and "no")? Such a variable might arise, for example, when forecasting daily sales and you want to take account of whether the day is a public holiday or not. So the predictor takes value "yes" on a public holiday, and "no" otherwise.

This situation can still be handled within the framework of multiple regression models by creating a "dummy variable" which takes value 1 corresponding to "yes" and 0 corresponding to "no". A dummy variable is also known as an "indicator variable".

A dummy variable can also be used to account for an outlier in the data. Rather than omit the outlier, a dummy variable removes its effect. In this case, the dummy variable takes value 1 for that observation and 0 everywhere else. An example is the case where a special event has occurred. For example when forecasting tourist arrivals to Brazil, we will need to account for the effect of the Rio de Janeiro summer Olympics in 2016.

If there are more than two categories, then the variable can be coded using several dummy variables (one fewer than the total number of categories). TSLM() will automatically handle this case if you specify a factor variable as a predictor. There is usually no need to manually create the corresponding dummy variables.

Notice that only six dummy variables are needed to code seven categories. That is because the seventh category (in this case Sunday) is captured by the intercept, and is specified when the dummy variables are all set to zero.

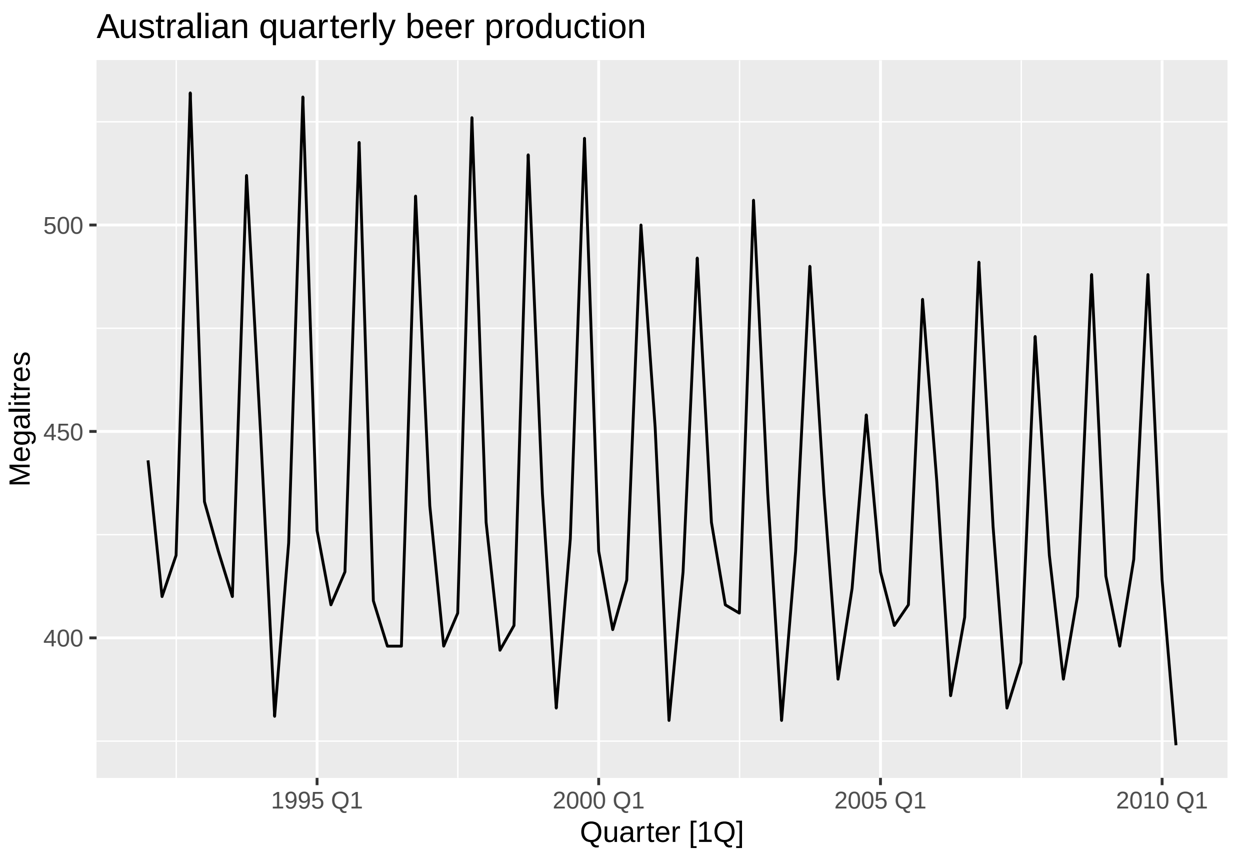
Many beginners will try to add a seventh dummy variable for the seventh category. This is known as the "dummy variable trap", because it will cause the regression to fail. There will be one too many parameters to estimate when an intercept is also included. The general rule is to use one fewer dummy variables than categories. So for quarterly data, use three dummy variables; for monthly data, use 11 dummy variables; and for daily data, use six dummy variables, and so on.

The interpretation of each of the coefficients associated with the dummy variables is that it is a measure of the effect of that category relative to the omitted category. In the above example, the coefficient of $d_{1,t}$ associated with Monday will measure the effect of Monday on the forecast variable compared to the effect of Sunday. An example of interpreting estimated dummy variable coefficients capturing the quarterly seasonality of Australian beer production follows.

**Example:**

Recall the Australian quarterly beer production data shown again in Figure 7.14.

```r
recent_production <- aus_production |>
  filter(year(Quarter) >= 1992)
recent_production |>
  autoplot(Beer) +
  labs(y = "Megalitres",
       title = "Australian quarterly beer production")
```



We want to forecast the value of future beer production. We can model this data using a regression model with a linear trend and quarterly dummy variables,

$y_t = \beta_0 + \beta_1 t + \beta_2 d_{2,t} + \beta_3 d_{3,t} + \beta_4 d_{4,t} + \varepsilon_t,$

where $d_{i,t} = 1$ if $t$ is in quarter $i$ and 0 otherwise. The first quarter variable has been omitted, so the coefficients associated with the other quarters are measures of the difference between those quarters and the first quarter.

The `TSLM()` function will automatically handle this situation if you specify the special `season()`

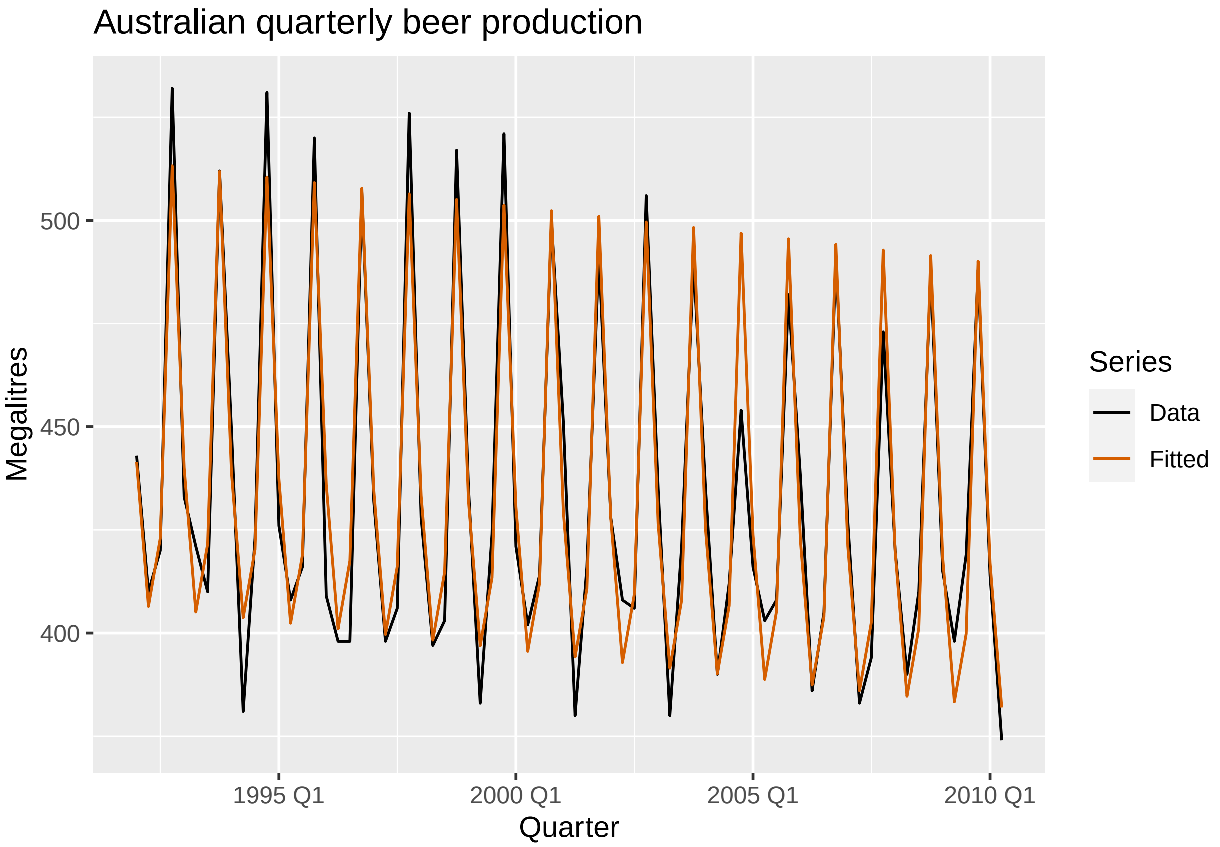
```r
fit_beer <- recent_production |>
  model(TSLM(Beer ~ trend() + season()))
report(fit_beer)
```

```
## Series: Beer
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -42.9029  -7.5995  -0.4594   7.9908  21.7895
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   441.80044    3.73353 118.333  < 2e-16 ***
## trend()        -0.34027    0.06657  -5.111 2.73e-06 ***
## season()year2 -34.65973    3.96832  -8.734 9.10e-13 ***
## season()year3 -17.82164    4.02249  -4.430 3.45e-05 ***
## season()year4   72.79641    4.02305  18.095  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.23 on 69 degrees of freedom
## Multiple R-squared: 0.9243,  Adjusted R-squared: 0.9199
## F-statistic: 210.7 on 4 and 69 DF, p-value: < 2.22e-16
```

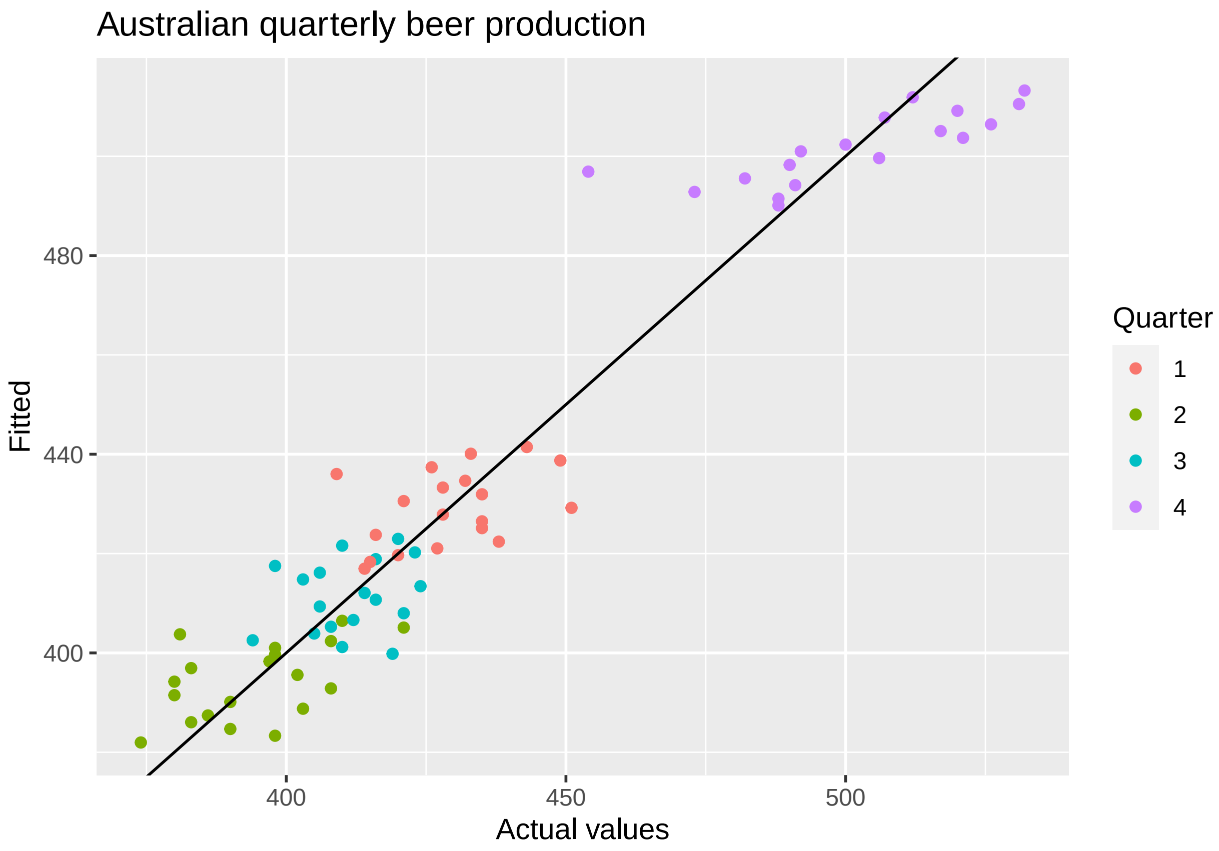
Note that trend() and season() are not standard functions; they are “special” functions that work within the TSLM() model formulae.

There is an average downward trend of -0.34 megalitres per quarter. On average, the second quarter has production of 34.7 megalitres lower than the first quarter, the third quarter has production of 17.8 megalitres lower than the first quarter, and the fourth quarter has production of 72.8 megalitres higher than the first quarter.

```
augment(fit_beer) |>
  ggplot(aes(x = Quarter)) +
  geom_line(aes(y = Beer, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  scale_colour_manual(
    values = c(Data = "black", Fitted = "#D55E00")
  ) +
  labs(y = "Megalitres",
       title = "Australian quarterly beer production") +
  guides(colour = guide_legend(title = "Series"))
```



```
augment(fit_beer) |>
  ggplot(aes(x = Beer, y = .fitted,
             colour = factor(quarter(Quarter)))) +
  geom_point() +
  labs(y = "Fitted", x = "Actual values",
       title = "Australian quarterly beer production") +
  geom_abline(intercept = 0, slope = 1) +
  guides(colour = guide_legend(title = "Quarter"))
```



##Intervention variables

It is often necessary to model interventions that may have affected the variable to be forecast. For example, competitor activity, advertising expenditure, industrial action, and so on, can all have an effect.

When the effect lasts only for one period, we use a “spike” variable. This is a dummy variable that takes value one in the period of the intervention and zero elsewhere. A spike variable is equivalent to a dummy variable for handling an outlier.

Other interventions have an immediate and permanent effect. If an intervention causes a level shift (i.e., the value of the series changes suddenly and permanently from the time of intervention), then we use a “step” variable. A step variable takes value zero before the intervention and one from the time of intervention onward.

Another form of permanent effect is a change of slope. Here the intervention is handled using a piecewise linear trend; a trend that bends at the time of intervention and hence is nonlinear. We will discuss this in Section 7.7.

## Trading days

The number of trading days in a month can vary considerably and can have a substantial effect on sales data. To allow for this, the number of trading days in each month can be included as a predictor.

An alternative that allows for the effects of different days of the week has the following predictors:

$x_1$ = number of Mondays in month;

$x_2$ =number of Tuesdays in month;

$x_7$ = number of Sundays in month.

## Distributed lags

It is often useful to include advertising expenditure as a predictor. However, since the effect of advertising can last beyond the actual campaign, we need to include lagged values of advertising expenditure.

Thus, the following predictors may be used.

$x_1$ = advertising for previous month;

$x_2$ = advertising for two months previously;

$x_m$ = advertising for m months previously.

It is common to require the coefficients to decrease as the lag increases, although this is beyond the scope of this book.

###Easter

Easter differs from most holidays because it is not held on the same date each year, and its effect can last for several days. In this case, a dummy variable can be used with value one where the holiday falls in the particular time period and zero otherwise.

With monthly data, if Easter falls in March then the dummy variable takes value 1 in March, and if it falls in April the dummy variable takes value 1 in April. When Easter starts in March and finishes in April, the dummy variable is split proportionally between months.

## Fourier series

An alternative to using seasonal dummy variables, especially for long seasonal periods, is to use Fourier terms. Jean-Baptiste Fourier was a French mathematician, born in the 1700s, who showed that a series of sine and cosine terms of the right frequencies can approximate any periodic function. We can use them for seasonal patterns.

If $m$ is the seasonal period, then the first few Fourier terms are given by

$x_{1,t}=sin(2 t/m),x_{2,t}=cos(2 t/m),x_{3,t}=sin(4 t/m),x_{4,t}=cos(4 t/m),x_{5,t}=sin(6 t/m),x_{6,t}=cos(6 t/m

and so on. If we have monthly seasonality, and we use the first 11 of these predictor variables, then we will get exactly the same forecasts as using 11 dummy variables.

With Fourier terms, we often need fewer predictors than with dummy variables, especially when $m$ is large. This makes them useful for weekly data, for example, where $m \approx 52$. For short seasonal periods (e.g., quarterly data), there is little advantage in using Fourier terms over seasonal dummy variables.

These Fourier terms are produced using the fourier() function. For example, the Australian beer data can be modelled like this.

```r
fourier_beer <- recent_production |>
  model(TSLM(Beer ~ trend() + fourier(K = 2)))
report(fourier_beer)
```

```
## Series: Beer
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -42.9029  -7.5995  -0.4594   7.9908  21.7895
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         446.87920    2.87321 155.533  < 2e-16 ***
## trend()              -0.34027    0.06657  -5.111 2.73e-06 ***
## fourier(K = 2)C1_4    8.91082    2.01125   4.430 3.45e-05 ***
## fourier(K = 2)S1_4 -53.72807    2.01125 -26.714  < 2e-16 ***
## fourier(K = 2)C2_4 -13.98958    1.42256  -9.834 9.26e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.23 on 69 degrees of freedom
## Multiple R-squared: 0.9243,  Adjusted R-squared: 0.9199
## F-statistic: 210.7 on 4 and 69 DF, p-value: < 2.22e-16
```

The K argument to fourier() specifies how many pairs of sin and cos terms to include. The maximum allowed is $K = m/2$ where $m$ is the seasonal period. Because we have used the maximum here, the results are identical to those obtained when using seasonal dummy variables.

If only the first two Fourier terms are used ( $x_{1,t}$ and $x_{2,t}), the seasonal pattern will follow a simple sine wave. A regression model containing Fourier terms is often called a harmonic regression because the successive Fourier terms represent harmonics of the first two Fourier terms.

## 7.5 Selecting predictors

The common in regressing modelling is to select a subset of the available predictors.

### Comparing regression models

Computer output for regression will always give the $R^2$ value. $R^2$ is useful for the summary of the model but not useful for selecting predictors. $R^2$ is also called the *"coefficient of determination"* and it is the proportion of variance accounted for (explained) by the predictors. It is the ratio of variation in the fitted values to the variation in the data. If you explain lots of variation in the data then it is close to 1. So it is useful for the summary but not useful for selecting the predictors.

- $R^2$ does not allow for the "degrees of freedom". It does not allow for the extra coefficients that might be attached to not very useful predictors.
- Adding "any" variable tends to increase the value of $R^2$ even if that variable is not relevant.

To overcome this problem, we can use "adjusted $R^2$":

$\bar{R}^2 = 1 - (1 - R^2)\frac{T-1}{T-k-1}$

where $k =$ no. of predictors and $T =$ no. of observations.

It adjusts the no. of terms in the model. Observe the term $\frac{T-1}{T-k-1}$, if we add the extra predictor then denominator is going to get smaller.

Maximizing $\bar{R}^2$ is equivalent to minimizing the unbiased estimate of the residual variance where we just sum the squared residuals and divide by $T-k-1$

$$\hat{\sigma}^2 = \frac{1}{T-k-1}\Sigma_{t=1}^{T}\epsilon_t^2$$

So far it became sort of a way that people use to select predictors but it's also not a best way to do it.

In 1970s, Japanese statistician Akaiki came up the following way to do it which is now called the "AIC".

### Akaiki's Information Criterian

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

Where $L$ is likelihood and $k$ is the number of predictors in the model.

- AIC penalizes terms more heavily that $\bar{R}^2$.
- Minimizing the AIC is asymptotically equivalent to minimizing MSE via *leave-one-out cross-validation* (for any linear regression). In regression context, leave one out cross-validation is the more interesting measure.

### Corrected AIC

For small values of $T$, the AIC tends to select too many predictors and so, a bias-corrected version of the AIC has been developed.

$AIC_c = AIC + \frac{2(k+2)(k+3)}{T-k-3}$

As with the $AIC$, $AIC_c$ should be minimized. These days we use AICc statistic when we choose a model.

### Bayesian Information Criterian

There is another statistic that is sometimes called BIC which has even stronger penalties so when we add more terms to the model then it penalized more strongly with a multiple of $\log(T)$.

$$BIC = -2\log(L) + (k+2)\log(T)$$

where $L$ is the likelihood and $k$ is the no. of predictors in the model.

- BIC penalizes terms more heavily than AIC.
- Also called SBIC (Swartchz) and SC(Swarctz criterian)
- Minimizing BIC is asymptotically equivalent to leave-v-out cross-validation when $v = T[1 - \frac{1}{\log T - 1}]$ here we leave "v" number of observations and we have provided the expression for $v$.
- BIC tends to popular in econometrics but it is not so interesting in forecasting because we don't believe that data comes from a correct model. It comes from some complicated real world phenomenon.
- AIC is a much better measure for the forecast accuracy than BIC.

### Leave-one-out-cross validation

For regression, leave-one-out cross validation is faster and more efficient than time series cross validation.

- Select one observation in test test and use *remaining* observations in the training set. Compute error on test observation. If T is the number of observations then we can fit $T$ different models.
- Repeat using each possible observation as the test set.

- Compute accuracy measure over all errors.

(Check figures)

## Choosing Regression Variables

- Fit all possible regression models using one or more of the predictors.
- Choose the best model based on one of the measures of predictive ability (CV, AIC, AICc) they are asymptotically equivalent to each other. So minimizing the CV or minimizing the AIC or minimizing the AICc will lead to the same model.

The problem with this strategy is that there are too many models to compare. If there are large no. of predictors then it's not possible to fit all of the possible regression models. For example, 50 predictors leads to over 1 quadrillion possible models. So we need an alternative strategy.

### (1) Backward stepwise regression

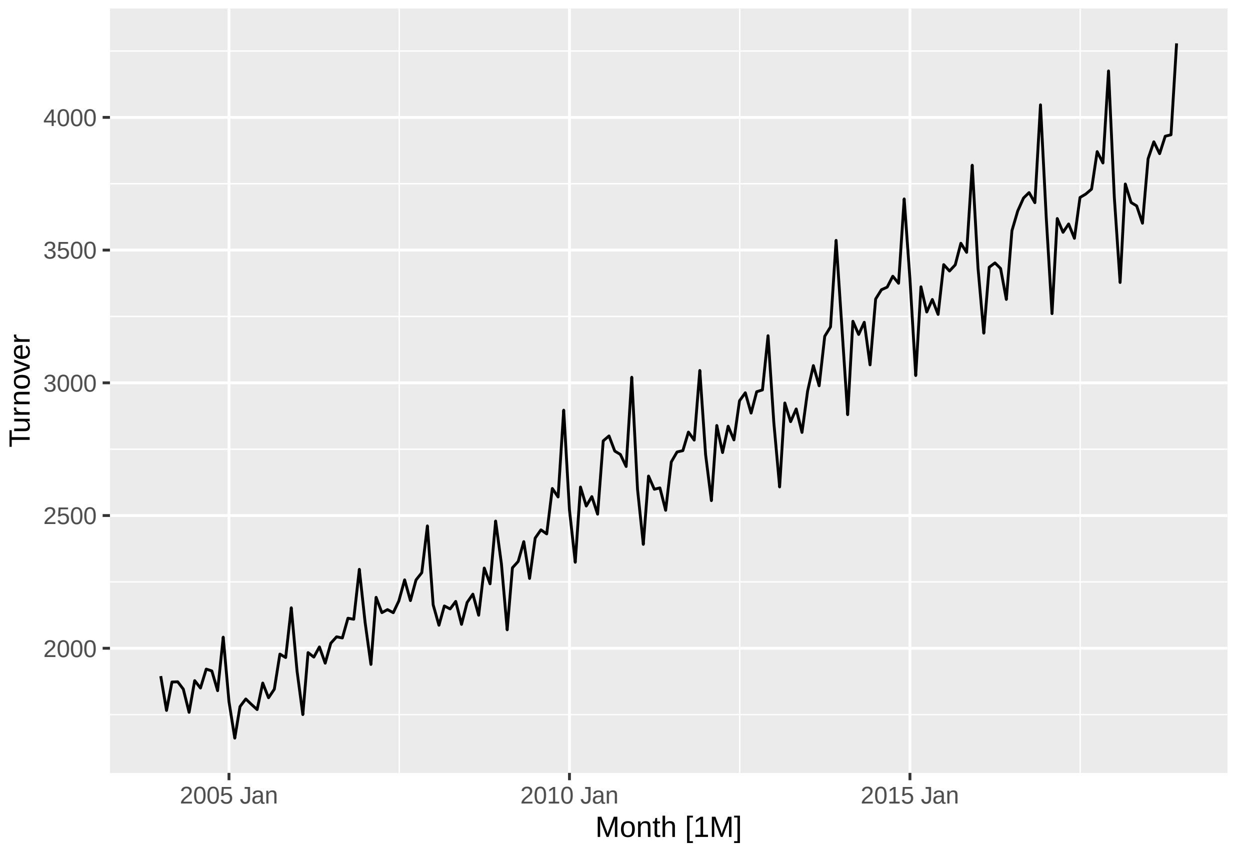
- Start with a model containing all the variables.
- Try subtracting one variable at a time. Keep the model if it has lower CV or AICc.
- Iterate until no further improvement.

*Note:*

- Stepwise regression is not guaranteed to lead to the best model.
- Inference on coefficients of final model will be wrong.

#### Example: Harmonic regression: eating-out expenditure

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



```
fit <- aus_cafe  |>
  model(
    K1 = TSLM(log(Turnover) ~ trend() + fourier(K=1)),
    K2 = TSLM(log(Turnover) ~ trend() + fourier(K=2)),
    K3 = TSLM(log(Turnover) ~ trend() + fourier(K=3)),
    K4 = TSLM(log(Turnover) ~ trend() + fourier(K=4)),
    K5 = TSLM(log(Turnover) ~ trend() + fourier(K=5)),
    K6 = TSLM(log(Turnover) ~ trend() + fourier(K=6))
  )
glance(fit) |> select(.model,r_squared, adj_r_squared, CV, AICc)
```

```
## # A tibble: 6 x 5
##   .model r_squared adj_r_squared      CV   AICc
##   <chr>      <dbl>         <dbl>   <dbl>  <dbl>
## 1 K1         0.962         0.962 0.00238 -1085.
## 2 K2         0.966         0.965 0.00220 -1099.
## 3 K3         0.976         0.975 0.00157 -1160.
## 4 K4         0.980         0.979 0.00138 -1183.
## 5 K5         0.985         0.984 0.00104 -1234.
## 6 K6         0.985         0.984 0.00105 -1232.
```

For AICc, 5th model is best and slightly better than the 6th model. And CV statistic for 5th model is best and slightly better than the 6th model. So it tells that we have to use 5 pairs of Fourier terms to fit the model and describe the seasonality.

——————————- Summary ——————————

When there are many possible predictors, we need some strategy for selecting the best predictors to use in a regression model.

A common approach that is not recommended is to plot the forecast variable against a particular predictor and if there is no noticeable relationship, drop that predictor from the model. This is invalid because it is not always possible to see the relationship from a scatterplot, especially when the effects of other predictors have not been accounted for.

Another common approach which is also invalid is to do a multiple linear regression on all the predictors and disregard all variables whose p-values are greater than 0.05. To start with, statistical significance does not always indicate predictive value. Even if forecasting is not the goal, this is not a good strategy because the p-values can be misleading when two or more predictors are correlated with each other (see Section 7.8).

Instead, we will use a measure of predictive accuracy. Five such measures are introduced in this section. They can be shown using the glance() function, here applied to the model for US consumption:

```
glance(fit_consMR) |>
  select(adj_r_squared, CV, AIC, AICc, BIC)
```

```
## # A tibble: 1 x 5
##   adj_r_squared    CV   AIC  AICc   BIC
##           <dbl> <dbl> <dbl> <dbl> <dbl>
## 1         0.763 0.104 -457. -456. -437.
```

```
#> # A tibble: 1 × 5
#>   adj_r_squared    CV   AIC  AICc   BIC
#>           <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1         0.763 0.104 -457. -456. -437.
```

We compare these values against the corresponding values from other models. For the CV, AIC, AICc and BIC measures, we want to find the model with the lowest value; for Adjusted $R^2$, we seek the model with the highest value.

## Adjusted $R^2$

Computer output for a regression will always give the $R^2$ value, discussed in Section 7.2. However, it is not a good measure of the predictive ability of a model. It measures how well the model fits the historical data, but not how well the model will forecast future data.

In addition, $R^2$ does not allow for "degrees of freedom". Adding any variable tends to increase the value of $R^2$, even if that variable is irrelevant. For these reasons, forecasters should not use $R^2$ to determine whether a model will give good predictions, as it will lead to overfitting.

An equivalent idea is to select the model which gives the minimum sum of squared errors (SSE), given by

$$SSE = \Sigma_{t=1}^{T} e_t^2$$

Minimising the SSE is equivalent to maximising $R^2$ and will always choose the model with the most variables, and so is not a valid way of selecting predictors.

An alternative which is designed to overcome these problems is the adjusted $R^2$ (also called "R-bar-squared").

where $T$ is the number of observations and k is the number of predictors. This is an improvement on $R^2$, as it will no longer increase with each added predictor. Using this measure, the best model will be the one with the largest value of $\bar{R}^2$. Maximising $\bar{R}^2$ is equivalent to minimising the standard error
$\hat{\sigma}_e$ given in Equation (7.3).

Maximising $\bar{R}^2$ works quite well as a method of selecting predictors, although it does tend to err on the side of selecting too many predictors.

## Cross-validation

Time series cross-validation was introduced in Section 5.8 as a general tool for determining the predictive ability of a model. For regression models, it is also possible to use classical leave-one-out cross-validation to select predictors (Bergmeir et al., 2018). This is faster and makes more efficient use of the data. The procedure uses the following steps:

(1) Remove observation $t$ from the data set, and fit the model using the remaining data. Then compute the error ( $e^*_t = y_t - \hat{y}_t$) for the omitted observation. (This is not the same as the residual because the $t^{th}$ observation was not used in estimating the value of $\hat{y}_t$.)

(2) Repeat step 1 for $t = 1, ..., T$.

(3) Compute the MSE from $e^∗_1,…,…,e*∗_T. We shall call this the CV.

Although this looks like a time-consuming procedure, there are fast methods of calculating CV, so that it takes no longer than fitting one model to the full data set. The equation for computing CV efficiently is given in Section 7.9. Under this criterion, the best model is the one with the smallest value of CV.

## Akaike's Information Criterion

A closely-related method is Akaike's Information Criterion, which we define as

$AIC = T\log(\frac{SSE}{T}) + 2(k+2)$

where T is the number of observations used for estimation and k is the number of predictors in the model. Different computer packages use slightly different definitions for the AIC, although they should all lead to the same model being selected. The k+2 part of the equation occurs because there are k+2 parameters in the model: the k coefficients for the predictors, the intercept and the variance of the residuals. The idea here is to penalise the fit of the model (SSE) with the number of parameters that need to be estimated.

The model with the minimum value of the AIC is often the best model for forecasting. For large values of T, minimising the AIC is equivalent to minimising the CV value.

## Corrected Akaike's Information Criterion

For small values of T, the AIC tends to select too many predictors, and so a bias-corrected version of the AIC has been developed,

$$AIC_c = AIC + \frac{2(k+2)(k+3)}{T-k-3}$$

As with the AIC, the AICc should be minimised.

## Schwarz's Bayesian Information Criterion

A related measure is Schwarz's Bayesian Information Criterion (usually abbreviated to BIC, SBIC or SC):

$$BIC = T\log(\frac{SSE}{T}) + (k+2)\log(T)$$

As with the AIC, minimising the BIC is intended to give the best model. The model chosen by the BIC is either the same as that chosen by the AIC, or one with fewer terms. This is because the BIC penalises the number of parameters more heavily than the AIC. For large values of T ,minimising BIC is similar to leave-v-out cross-validation when v=T[1−1/(log(T)−1)].

### Which measure should we use?

While $\bar{R}^2$ is widely used, and has been around longer than the other measures, its tendency to select too many predictor variables makes it less suitable for forecasting.

Many statisticians like to use the BIC because it has the feature that if there is a true underlying model, the BIC will select that model given enough data. However, in reality, there is rarely, if ever, a true underlying model, and even if there was a true underlying model, selecting that model will not necessarily give the best forecasts (because the parameter estimates may not be accurate).

Consequently, we recommend that one of the AICc, AIC, or CV statistics be used, each of which has forecasting as their objective. If the value of T is large enough, they will all lead to the same model. In most of the examples in this book, we use the AICc value to select the forecasting model.

In the multiple regression example for forecasting US consumption we considered four predictors. With four predictors, there are $2^4=16 possible models. Now we can check if all four predictors are actually useful, or whether we can drop one or more of them. All 16 models were fitted and the results are summarised in Table 7.1. A " " indicates that the predictor was included in the model. Hence the first row shows the measures of predictive accuracy for a model including all four predictors.

The results have been sorted according to the AICc. Therefore the best models are given at the top of the table, and the worst at the bottom of the table.

(check figure)

The best model contains all four predictors. However, a closer look at the results reveals some interesting features. There is clear separation between the models in the first four rows and the ones below. This indicates that Income and Savings are both more important variables than Production and Unemployment. Also, the first three rows have almost identical values of CV, AIC and AICc. So we could possibly drop either the Production variable, or the Unemployment variable, and get similar forecasts. Note that Production and Unemployment are highly (negatively) correlated, as shown in Figure 7.5, so most of the predictive information in Production is also contained in the Unemployment variable.

## Best subset regression

Where possible, all potential regression models should be fitted (as was done in the example above) and the best model should be selected based on one of the measures discussed. This is known as "best subsets" regression or "all possible subsets" regression.

### Stepwise regression

If there are a large number of predictors, it is not possible to fit all possible models. For example, 40 predictors leads to $2^40>1$ trillion possible models! Consequently, a strategy is required to limit the number of models to be explored.

An approach that works quite well is backwards stepwise regression:

Start with the model containing all potential predictors. Remove one predictor at a time. Keep the model if it improves the measure of predictive accuracy. Iterate until no further improvement. If the number of potential predictors is too large, then the backwards stepwise regression will not work and forward stepwise regression can be used instead. This procedure starts with a model that includes only the intercept. Predictors are added one at a time, and the one that most improves the measure of predictive accuracy is retained in the model. The procedure is repeated until no further improvement can be achieved.

Alternatively for either the backward or forward direction, a starting model can be one that includes a subset of potential predictors. In this case, an extra step needs to be included. For the backwards procedure we should also consider adding a predictor with each step, and for the forward procedure we should also consider dropping a predictor with each step. These are referred to as hybrid procedures.

It is important to realise that any stepwise approach is not guaranteed to lead to the best possible model, but it almost always leads to a good model. For further details see James et al. (2014).

### Beware of inference after selecting predictors

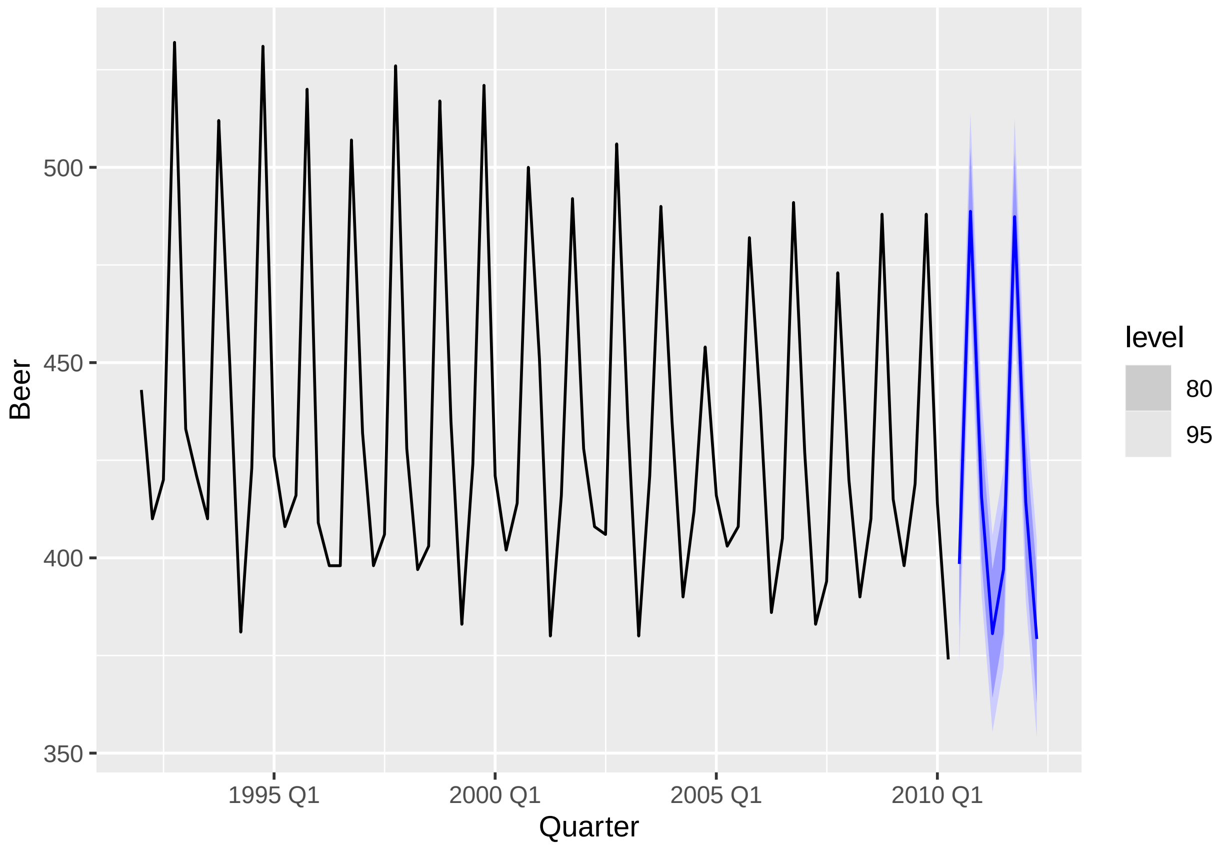
We do not discuss statistical inference of the predictors in this book (e.g., looking at p-values associated with each predictor). If you do wish to look at the statistical significance of the predictors, beware that any procedure involving selecting predictors first will invalidate the assumptions behind the p-values. The procedures we recommend for selecting predictors are helpful when the model is used for forecasting; they are not helpful if you wish to study the effect of any predictor on the forecast variable.

## 7.6 Forecasting with regression

### Ex-ante versus Ex-post forecasts

- *Ex-ante forecasts* are made using only information available in advance. Hence I need to generate some forecasts for the $x's$ and use these forecasts to predict $y's$. This is one of the challenges in regression to predict x's and people come up with strategies like autoregressions and so on.
- *Ex-post forecasts* are made using later information on the predictors. It is useful for studying behaviour of forecasting models.
- trend, seasonal and calendar variables are known in advance, so these don't need to be forecast.

```
recent_production <- aus_production |> filter(year(Quarter) >= 1992)
recent_production |> model(TSLM(Beer ~ trend()+season())) |>
  forecast() |> autoplot(recent_production)
```



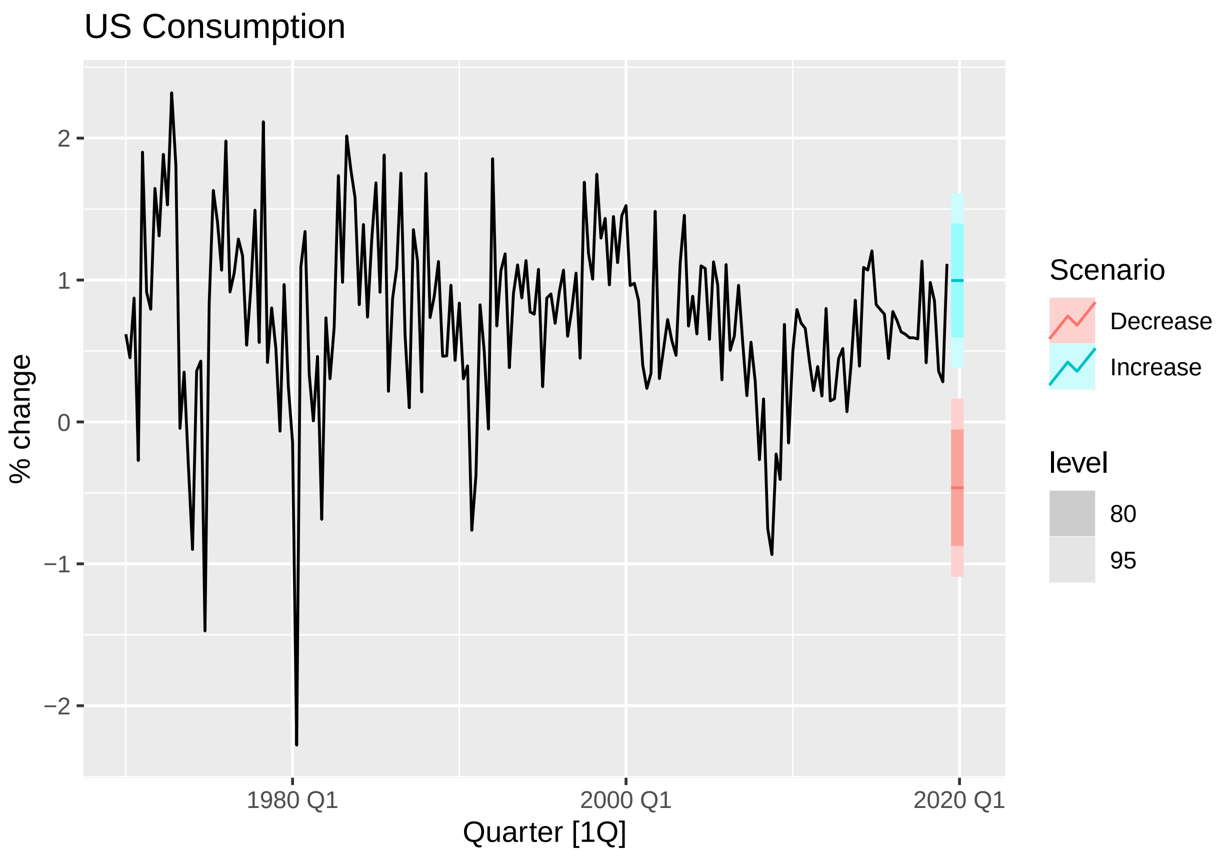
### Scenario based forecasting

- Assumes possible scenarios for the predictor variables
- Prediction intervals for scenario based forecasts do not include the uncertainty associated with the future values of the predictor variables.

```
fit_consBest <- us_change |>
  model(TSLM(Consumption ~ Income + Savings +Unemployment))
future_scenarios <- scenarios(
  Increase = new_data(us_change,4) |>
    mutate(Income=1,Savings=0.5,Unemployment=0),
  Decrease=new_data(us_change,4) |>
    mutate(Income=-1,Savings=-0.5,Unemployment=0),
  names_to = "Scenario"
)
fc <- forecast(fit_consBest,new_data=future_scenarios)
```

here, us_change is 4 steps ahead, income is 1% increased in next 4 quarters, savings is 0.5% increased.

```
us_change |> autoplot(Consumption) +
  labs(y= "% change in US Consumption")+
  autolayer(fc)+
  labs(title = "US Consumption", y="% change")
```



### Building a predictive regression model

if we don't want to deal with the future values of our predictors and alternative is to build predictive regression model.

- If getting forecasts of predictors is difficult, you can use lagged predictors instead.

$y_{t+h} = \beta_0 + \beta_1 x_{1,t} + ... + \beta_k x_{k,t} + \epsilon_{t+h}$

Here, my predictors are lagged values of the predictors.

- A different model for each forecast horizon h.

Summary

Recall that predictions of y can be obtained using

$\hat{y}_t = \hat{\beta}_0 + \hat{\beta}_1 x_{1,t} + \hat{\beta}_2 x_{2,t} + \cdots + \hat{\beta}_k x_{k,t},$

which comprises the estimated coefficients and ignores the error in the regression equation. Plugging in the values of the predictor variables \$x_{1,t},...,x_{k,t} for $t = 1, ..., T$ returns the fitted (training set) values of y. What we are interested in here, however, is forecasting future values of $y$.

## Ex-ante versus ex-post forecasts

When using regression models for time series data, we need to distinguish between the different types of forecasts that can be produced, depending on what is assumed to be known when the forecasts are computed.

Ex-ante forecasts are those that are made using only the information that is available in advance. For example, ex-ante forecasts for the percentage change in US consumption for quarters following the end of the sample, should only use information that was available up to and including 2019 Q2. These are genuine forecasts, made in advance using whatever information is available at the time. Therefore in order to generate ex-ante forecasts, the model requires forecasts of the predictors. To obtain these we can use one of the simple methods introduced in Section 5.2 or more sophisticated pure time series approaches that follow in Chapters 8 and 9. Alternatively, forecasts from some other source, such as a government agency, may be available and can be used.

Ex-post forecasts are those that are made using later information on the predictors. For example, ex-post forecasts of consumption may use the actual observations of the predictors, once these have been observed. These are not genuine forecasts, but are useful for studying the behaviour of forecasting models.

The model from which ex-post forecasts are produced should not be estimated using data from the forecast period. That is, ex-post forecasts can assume knowledge of the predictor variables (the x variables), but should not assume knowledge of the data that are to be forecast (the y variable).

A comparative evaluation of ex-ante forecasts and ex-post forecasts can help to separate out the sources of forecast uncertainty. This will show whether forecast errors have arisen due to poor forecasts of the predictor or due to a poor forecasting model.

###Example: Australian quarterly beer production

Normally, we cannot use actual future values of the predictor variables when producing ex-ante forecasts because their values will not be known in advance. However, the special predictors introduced in Section 7.4 are all known in advance, as they are based on calendar variables (e.g., seasonal dummy variables or public holiday indicators) or deterministic functions of time (e.g. time trend). In such cases, there is no difference between ex-ante and ex-post forecasts.

## Scenario based forecasting

In this setting, the forecaster assumes possible scenarios for the predictor variables that are of interest. For example, a US policy maker may be interested in comparing the predicted change in consumption when there is a constant growth of 1% and 0.5% respectively for income and savings with no change in the employment rate, versus a respective decline of 1% and 0.5%, for each of the four quarters following the end of the sample. The resulting forecasts are calculated below and shown in Figure 7.18. We should note that prediction intervals for scenario based forecasts do not include the uncertainty associated with the future values of the predictor variables. They assume that the values of the predictors are known in advance.

## Building a predictive regression model

The great advantage of regression models is that they can be used to capture important relationships between the forecast variable of interest and the predictor variables. However, for ex ante forecasts, these models require future values of each predictor, which can be challenging. If forecasting each predictor is too difficult, we may use scenario-based forecasting instead, where we assume specific future values for all predictors.

An alternative formulation is to use as predictors their lagged values. Assuming that we are interested in generating a h-step ahead forecast we write $y_{t+h} = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \varepsilon_{t+h}$

for h=1,2…. The predictor set is formed by values of the x's, that are observed h time periods prior to observing y. Therefore when the estimated model is projected into the future, i.e., beyond the end of the sample T, all predictor values are available.

Including lagged values of the predictors does not only make the model operational for easily generating forecasts, it also makes it intuitively appealing. For example, the effect of a policy change with the aim of increasing production may not have an instantaneous effect on consumption expenditure. It is most likely that this will happen with a lagging effect. We touched upon this in Section 7.4 when briefly introducing distributed lags as predictors. Several directions for generalising regression models to better incorporate the rich dynamics observed in time series are discussed in Section 10.

## Prediction intervals

With each forecast for the change in consumption in Figure 7.18, 95% and 80% prediction intervals are also included. The general formulation of how to calculate prediction intervals for multiple regression models is presented in Section 7.9. As this involves some advanced matrix algebra we present here the case for calculating prediction intervals for a simple regression, where a forecast can be generated using the equation,

$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$

Assuming that the regression errors are normally distributed, an approximate 95% prediction interval associated with this forecast is given by

$\hat{y} \pm 1.96\hat{\sigma}_e\sqrt{1+\frac{1}{T}} + (x-\bar{x})^2/(T-1)(s_x)^2,$

where T is the total number of observations, $\bar{x}$ is the mean of the observed x values, $s_x$ is the standard deviation of the observed x values and $\hat{\sigma}e$ is the standard error of the regression given by Equation (7.3). Similarly, an 80% prediction interval can be obtained by replacing 1.96 by 1.28. Other prediction intervals can be obtained by replacing the 1.96 with the appropriate value given in Table 5.1. If the fable package is used to obtain prediction intervals, more exact calculations are obtained (especially for small values of T) than what is given by Equation (7.4).

Equation (7.4) shows that the prediction interval is wider when x is far from $\bar{x}$. That is, we are more certain about our forecasts when considering values of the predictor variable close to its sample mean.

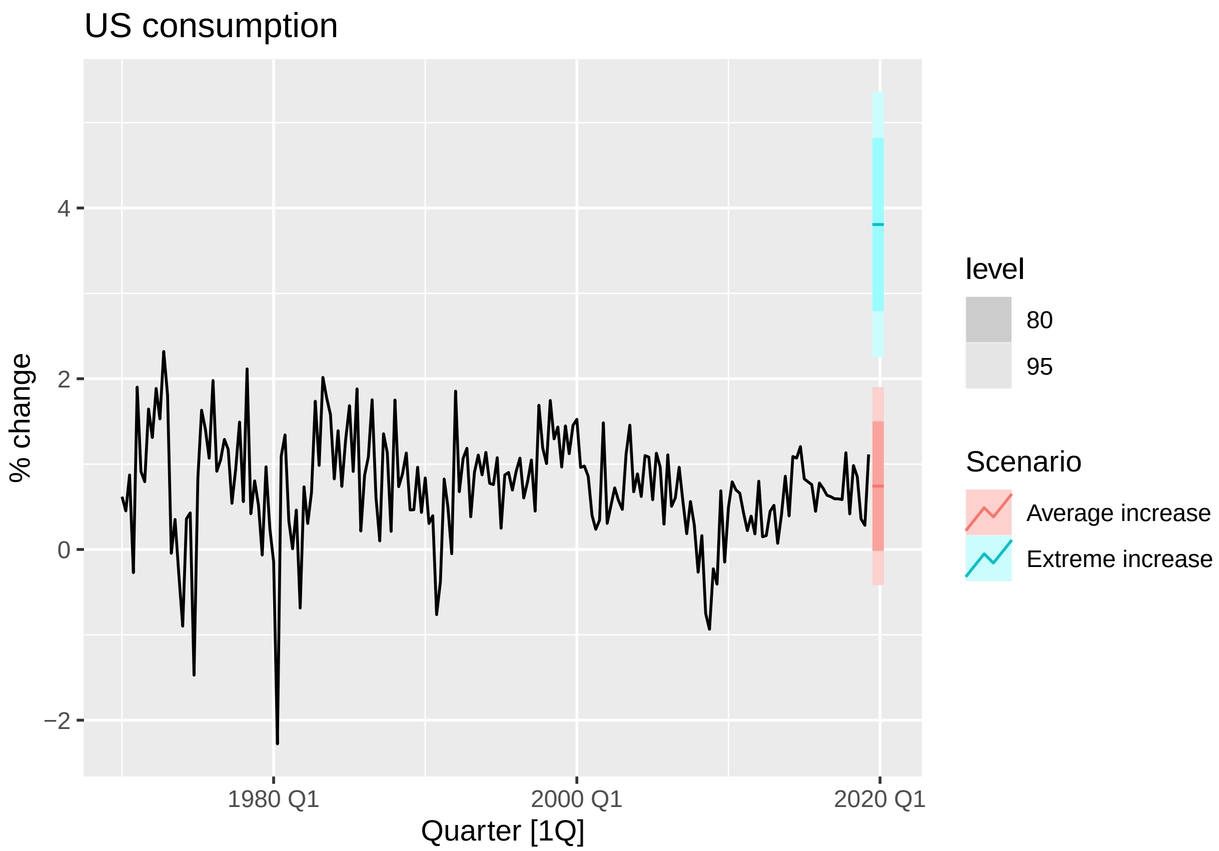
###Example

The estimated simple regression line in the US consumption example is $\hat{y}_t = 0.54 + 0.27x_t$.

Assuming that for the next four quarters, personal income will increase by its historical mean value of $\bar{x} = 0.73$ consumption is forecast to increase by
0.74% and the corresponding 80% and 95% prediction intervals are [−0.02,1.5] and [−0.42,1.9] respectively (calculated using R). If we assume an extreme increase of 12% in income, then the prediction intervals are considerably wider as shown in Figure 7.19.

```
fit_cons <- us_change |>
  model(TSLM(Consumption ~ Income))
new_cons <- scenarios(
  "Average increase" = new_data(us_change, 4) |>
    mutate(Income = mean(us_change$Income)),
  "Extreme increase" = new_data(us_change, 4) |>
    mutate(Income = 12),
  names_to = "Scenario"
)
fcast <- forecast(fit_cons, new_cons)

us_change |>
  autoplot(Consumption) +
  autolayer(fcast) +
  labs(title = "US consumption", y = "% change")
```



## 7.7 Nonlinear regression

**Non-linear regression**

A *log-log* functional form

$\log y = \beta_0 + \beta_1 \log x + \epsilon$

where $\beta_1$ is interpreted as an elasticity (the average percentage change in $y$ resulting from a $1$ increase in $x$).

- alternative specifications: log-linear, linear-log.
- use $\log(x + 1)$ if required because it will resolve the issue when $x = 0$.

**Piecewise linear and regression splines** $y = f(x) + \epsilon$

where $f$ is a non-linear function

If a log transformation is not good enough and we want to see a little bit more flexibility, we can think about having a non-linear function for the $x's$. A commonly used non-linear function will be to have a piecewise linear function

- For *piecewise linear* let $x_1 = x$ and

$x_2 = (x - c)_+ = 0 \; if \; x < c$ and

$x_2 = (x - c)_+ = x - c \; if \; x \geq c$

“c” is a point where trend is bended at that point.

- In general, *linear regression splines*

$x_1 = x$ and $x_2 = (x - c_1)_+$ and so on till $x_k = (x - c_{k-1})_+$

where $c_1, c_2, ..., c_{k-1}$ are knots.

- Need to select knots: can be difficult and arbitrary.
- Automatic knot selection algorithma very slow.
- Using piecewise cubics achieves a smoother result.

*Warning:* Better fit but forecasting outside the range of the historical data is even more unreliable.

### Non-linear trends

Piecewise linear trend with bend at $\tau$

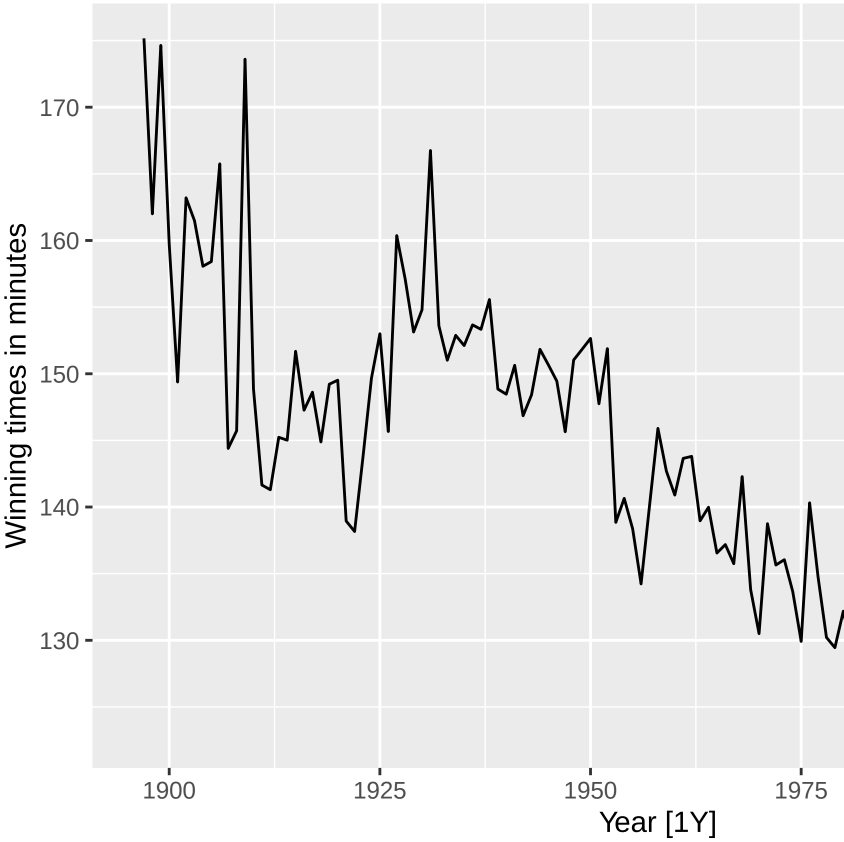
$x_{1,t} = t$

$x_{2,t} = 0 \; if \; t < \tau$

$x_{2,t} = (t - \tau) \; if \; t \geq \tau$

(check fig: screenshot-169)

```
marathon <- boston_marathon |>
  filter(Event == "Men's open division") |>
  select(-Event) |>
  mutate(Minutes = as.numeric(Time)/60)
marathon |> autoplot(Minutes) +labs(y="Winning times in minutes")
```



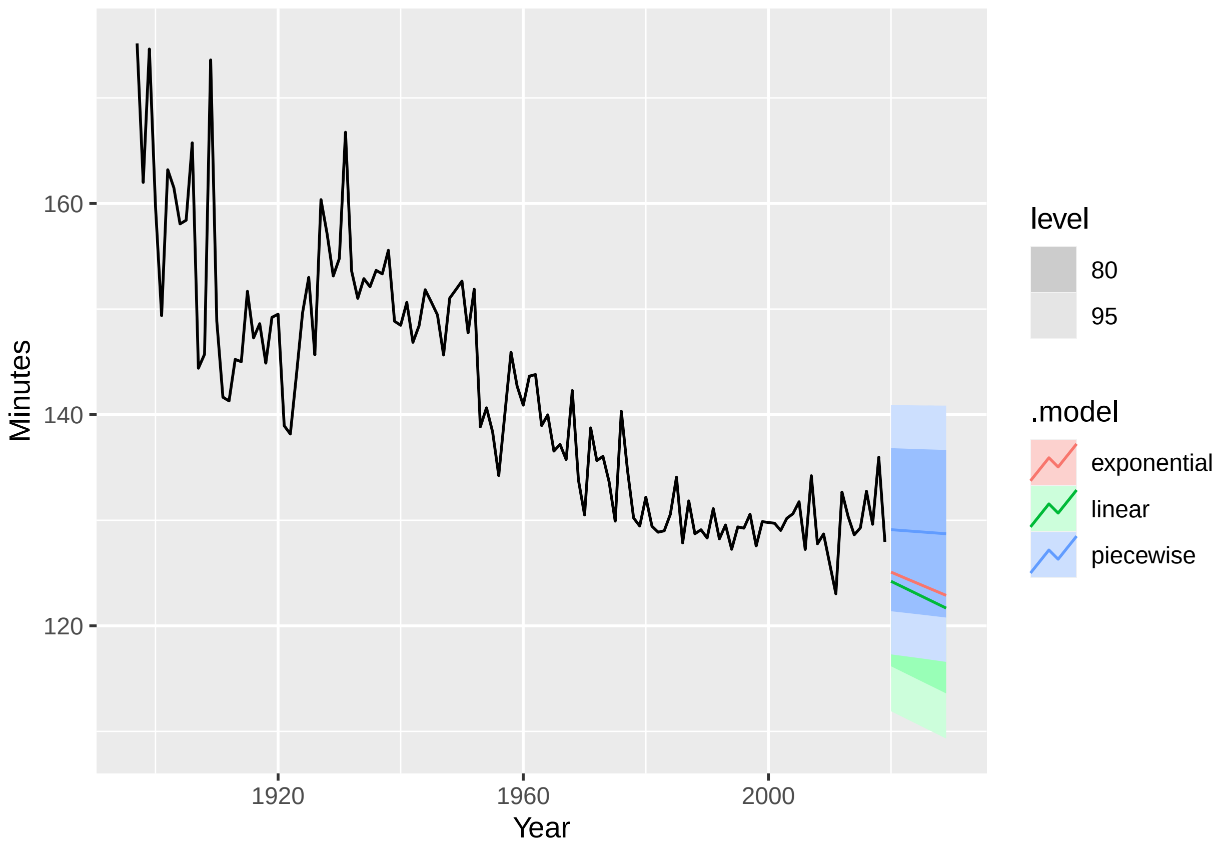
**Example: Boston Marathon winning times**

It is a non-linear trend so we are going to estimate a couple of alternative models. We take the exponential trend so that y would never be zero for large x.

```
fit_trends <- marathon |>
  model(
    # Linear trend
    linear = TSLM(Minutes ~ trend()),
    # Exponential trend
    exponential = TSLM(log(Minutes) ~ trend()),
    # Piecewise linear trend
    piecewise =TSLM(Minutes ~ trend(knots=c(1940,1980)))
  )
fit_trends
```

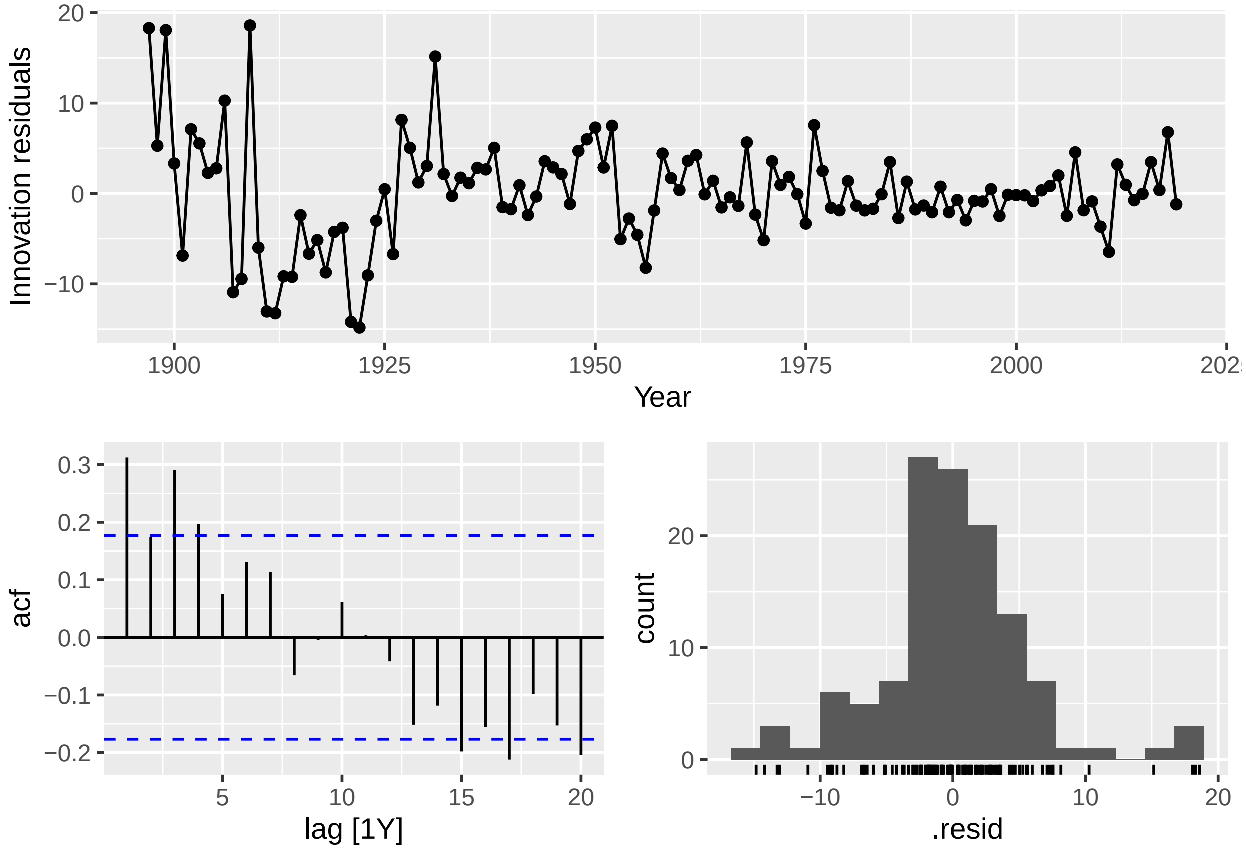
```
## # A mable: 1 x 3
##    linear exponential piecewise
##   <model>     <model>   <model>
## 1  <TSLM>      <TSLM>    <TSLM>
```

```
fit_trends |>
  forecast(h=10) |>
  autoplot(marathon)
```



Here, exponential model is completely unrealistic. Piecewise linear trend is going to do a better job. They are decreasing but decreasing at a much flatter rate. Now, if we check the residuals then

```
fit_trends |>
  select(piecewise) |>
  gg_tsresiduals()
```



Here, we are not capturing the autocorrelation well that is reflected in the acf plot. So, there is some significant spikes.

Summary

Although the linear relationship assumed so far in this chapter is often adequate, there are many cases in which a nonlinear functional form is more suitable. To keep things simple in this section we assume that we only have one predictor x.

The simplest way of modelling a nonlinear relationship is to transform the forecast variable y and/or the predictor variable x before estimating a regression model. While this provides a non-linear functional form, the model is still linear in the parameters. The most commonly used transformation is the (natural) logarithm (see Section 3.1).

A log-log functional form is specified as

$\log y = \beta_0 + \beta_1 \log x + \epsilon$

In this model, the slope $\beta_1$ can be interpreted as an elasticity: $\beta_1$ is the average percentage change in y resulting from a 1% increase in $x$. Other useful forms can also be specified. The log-linear form is specified by only transforming the forecast variable and the linear-log form is obtained by transforming the predictor.

Recall that in order to perform a logarithmic transformation to a variable, all of its observed values must be greater than zero. In the case that variable $x$ contains zeros, we use the transformation $\log(x + 1)$; i.e., we add one to the value of the variable and then take logarithms. This has a similar effect to taking logarithms but avoids the problem of zeros. It also has the neat side-effect of zeros on the original scale remaining zeros on the transformed scale.

There are cases for which simply transforming the data will not be adequate and a more general specification may be required. Then the model we use is

$y = f(x) + \epsilon$

where $f$ is a nonlinear function. In standard (linear) regression, $f(x) = \beta_0 + \beta_1 x$. In the specification of nonlinear regression that follows, we allow $f$ to be a more flexible nonlinear function of x,compared to simply a logarithmic or other transformation.

One of the simplest specifications is to make $f$ piecewise linear. That is, we introduce points where the slope of $f$ can change. These points are called knots. This can be achieved by letting $x_1 = x$ and introducing variable $x_2$ such that

$x_2 = (x - c)_+ = 0 \ if \ x < c$ and

$x_2 = (x - c)_+ = x - c \ if \ x \geq c$

The notation $(x-c)_+$ means the value $x-c$ if it is positive and 0 otherwise. This forces the slope to bend at point $c$. Additional bends can be included in the relationship by adding further variables of the above form.

Piecewise linear relationships constructed in this way are a special case of regression splines. In general, a linear regression spline is obtained using

$x_1 = x$ and $x_2 = (x - c_1)_+$ and so on till $x_k = (x - c_{k-1})_+$

where $c_1, ..., c_{k-1}$ are the knots (the points at which the line can bend). Selecting the number of knots (k−1) and where they should be positioned can be difficult and somewhat arbitrary. Some automatic knot selection algorithms are available, but are not widely used.

## Forecasting with a nonlinear trend

In Section 7.4 fitting a linear trend to a time series by setting $x = t$ was introduced. The simplest way of fitting a nonlinear trend is using quadratic or higher order trends obtained by specifying

$x_{1,t} = t, x_{2,t} = t^2, ....$

However, it is not recommended that quadratic or higher order trends be used in forecasting. When they are extrapolated, the resulting forecasts are often unrealistic.

A better approach is to use the piecewise specification introduced above and fit a piecewise linear trend which bends at some point in time. We can think of this as a nonlinear trend constructed of linear pieces. If the trend bends at time $ , then it can be specified by simply replacing x=t and $c = \tau$above such that we include the predictors,

$x_{1,t} = t$

$x_{2,t} = 0 \ if \ t < \tau$

$x_{2,t} = (t - \tau) \ if \ t \geq \tau$

in the model. If the associated coefficients of $x_{1,t}$and x_{2,t}$ are $\beta_1$ and $\beta_2$, then $\beta_1$ gives the slope of the trend before time $\tau$, while the slope of the line after time $\tau$ is given by $\beta_1 + \beta_2$. Additional bends can be included in the relationship by adding further variables of the form $(t - \tau)_+$where $\tau$ is the “knot” or point in time at which the line should bend.

## Example: Boston marathon winning times

We will fit some trend models to the Boston marathon winning times for men. First we extract the men’s data and convert the winning times to a numerical value. The course was lengthened (from 24.5 miles to 26.2 miles) in 1924, which led to a jump in the winning times, so we only consider data from that date onwards.

```
boston_men <- boston_marathon |>
  filter(Year >= 1924) |>
  filter(Event == "Men's open division") |>
  mutate(Minutes = as.numeric(Time)/60)
```

The top panel of Figure 7.20 shows the winning times since 1924. The time series shows a general downward trend as the winning times have been improving over the years. The bottom panel shows the residuals from fitting a linear trend to the data. The plot shows an obvious nonlinear pattern which has not been captured by the linear trend.

Fitting an exponential trend (equivalent to a log-linear regression) to the data can be achieved by transforming the y variable so that the model to be fitted is, $\log y_t = \beta_0 + \beta_1 t + \varepsilon_t$.

The fitted exponential trend and forecasts are shown in Figure 7.21. Although the exponential trend does not seem to fit the data much better than the linear trend, it perhaps gives a more sensible projection in that the winning times will decrease in the future but at a decaying rate rather than a fixed linear rate.

The plot of winning times reveals three different periods. There is a lot of volatility in the winning times up to about 1950, with the winning times barely declining. After 1950 there is a clear decrease in times, followed by a flattening out after the 1980s, with the suggestion of an upturn towards the end of the sample. To account for these changes, we specify the years 1950 and 1980 as knots. We should warn here that subjective identification of knots can lead to over-fitting, which can be detrimental to the forecast performance of a model, and should be performed with caution

```r
fit_trends <- boston_men |>
  model(
    linear = TSLM(Minutes ~ trend()),
    exponential = TSLM(log(Minutes) ~ trend()),
    piecewise = TSLM(Minutes ~ trend(knots = c(1950, 1980)))
  )
fc_trends <- fit_trends |> forecast(h = 10)

boston_men |>
  autoplot(Minutes) +
  geom_line(data = fitted(fit_trends),
            aes(y = .fitted, colour = .model)) +
  autolayer(fc_trends, alpha = 0.5, level = 95) +
  labs(y = "Minutes",
       title = "Boston marathon winning times")
```

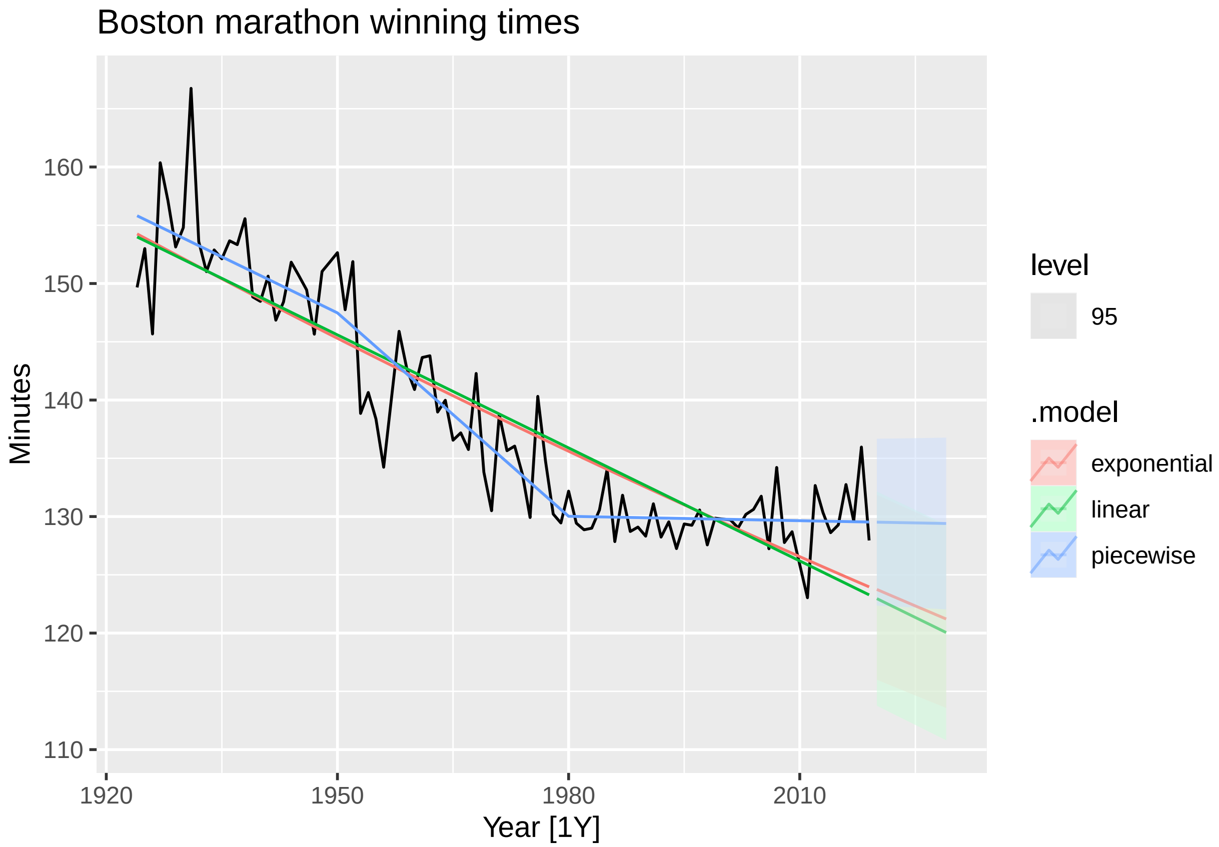


Figure 7.21 shows the fitted lines and forecasts from linear, exponential and piecewise linear trends. The best forecasts appear to come from the piecewise linear trend.

## 7.8 Correlation, causation and forecasting

You need to be careful when interpreting the model about two things:

(1) Correlation is not same as causation.

(2) Beware of multicollinearity

In forecasting these two things are treated differently

### Correlation is not causation

- When $x$ is useful for predicting $y$, it is not necessarily causing $y$. Because two things are correlated does not mean that one is causing the other. When you are doing a regression model for forecasting, it does not really much. Your predictor $x$ may not be causing $y$ but it can be still a good predictor, it can still be a good model.
- e.g. predict number of swimmers that swim every day $y$ using number of ice-creams sold $x$.Here, $y$ is highly correlated with $x$ but $x$ is not causing $y$.
- Correlations are useful for forecasting, even when there is no causality.
- Better models usually involve causal relationship (e.g. temperature $x$ and people $z$ to predict swimmers $y$). You can predict the number of swimmers if you know the temperatures and it will be a good model because here relationship is causal and causal relationship is stronger than non-causal relationships.

**Multicollinearity**

- Two predictors are highly correlated (i.e. the correlation between them is close to ±1)
- A linear combination of some of the predictors is highly correlated with another predictor. That happens in a dummy variable trap where we might put in a dummy variable for every seasonal period and we learned that we need less than 1 but if we put all in then we might fall into the problem of multicolinearity because last dummy is the sum of all - 1.
- A linear combination of one subset f predictors is highly correlated with a linear combination of another subset of predictors.

If there is an exact multicolinearity then we actually can't fit the regression. You should never use microsoft excel for statistical work. Use R for statistical work.

*if multicolinearity exists*

- the numerical estimates of the coefficients may be wrong (worse in excel than in a statistical package)
- don't rely on the p-values to determine significance of some predictors if multicolinearity is present and so these tests are very unreliable but in forecasting we are not doing it, we are just trying to find models that give good forecasts and there is no problem actually with the predictions from the model provided that the value of the predictors in the forecasts are within the range that you used in the training in the fitting. So, if your predictors normally goes between 0 and 100 and we are going to forecast ahead and the future values of predictors are also between 0 and 100 then it is colinear or close to colinear and other variable should not matter but if future values are outside the range of 0 and 100 then forecast can become quite unreliable.
- there is no problem with model predictions provided the predictors used for forecasting are within the range used for fitting.
- omitting variables can help
- combining variables can help. SOmetimes it is useful to take average of two variables or difference of the two variables and you don't lose any information and avoid the problem of collinearity.

——————————- Summary ——————————

## Correlation is not causation

It is important not to confuse correlation with causation, or causation with forecasting. A variable x may be useful for forecasting a variable y, but that does not mean x is causing y. It is possible that x is causing y, but it may be that
y is causing x, or that the relationship between them is more complicated than simple causality.

For example, it is possible to model the number of drownings at a beach resort each month with the number of ice-creams sold in the same period. The model can give reasonable forecasts, not because ice-creams cause drownings, but because people eat more ice-creams on hot days when they are also more likely to go swimming. So the two variables (ice-cream sales and drownings) are correlated, but one is not causing the other. They are both caused by a third variable (temperature). This is an example of "confounding" — where an omitted variable causes changes in both the response variable and at least one predictor variable.

We describe a variable that is not included in our forecasting model as a confounder when it influences both the response variable and at least one predictor variable. Confounding makes it difficult to determine what variables are causing changes in other variables, but it does not necessarily make forecasting more difficult.

Similarly, it is possible to forecast if it will rain in the afternoon by observing the number of cyclists on the road in the morning. When there are fewer cyclists than usual, it is more likely to rain later in the day. The model can give reasonable forecasts, not because cyclists prevent rain, but because people are more likely to cycle when the published weather forecast is for a dry day. In this case, there is a causal relationship, but in the opposite direction to our forecasting model. The number of cyclists falls because there is rain forecast. That is, y(rainfall) is affecting x(cyclists).

It is important to understand that correlations are useful for forecasting, even when there is no causal relationship between the two variables, or when the causality runs in the opposite direction to the model, or when there is confounding.

However, often a better model is possible if a causal mechanism can be determined. A better model for drownings will probably include temperatures and visitor numbers and exclude ice-cream sales. A good forecasting model for rainfall will not include cyclists, but it will include atmospheric observations from the previous few days.

## Forecasting with correlated predictors

When two or more predictors are highly correlated it is always challenging to accurately separate their individual effects. Suppose we are forecasting monthly sales of a company for 2012, using data from 2000–2011. In January 2008, a new competitor came into the market and started taking some market share. At the same time, the economy began to decline. In your forecasting model, you include both competitor activity (measured using advertising time on a local television station) and the health of the economy (measured using GDP). It will not be possible to separate the effects of these two predictors because they are highly correlated.

Having correlated predictors is not really a problem for forecasting, as we can still compute forecasts without needing to separate out the effects of the predictors. However, it becomes a problem with scenario forecasting as the scenarios should take account of the relationships between predictors. It is also a problem if some historical analysis of the contributions of various predictors is required.

## Multicollinearity and forecasting

A closely related issue is multicollinearity, which occurs when similar information is provided by two or more of the predictor variables in a multiple regression.

It can occur when two predictors are highly correlated with each other (that is, they have a correlation coefficient close to +1 or -1). In this case, knowing the value of one of the variables tells you a lot about the value of the other variable. Hence, they are providing similar information. For example, foot size can be used to predict height, but including the size of both left and right feet in the same model is not going to make the forecasts any better, although it won't make them worse either.

Multicollinearity can also occur when a linear combination of predictors is highly correlated with another linear combination of predictors. In this case, knowing the value of the first group of predictors tells you a lot about the value of the second group of predictors. Hence, they are providing similar information.

An example of this problem is the dummy variable trap discussed in Section 7.4. Suppose you have quarterly data and use four dummy variables, $d_1, d_2, d_3 and d_4$. Then $d_4 = 1 - d_1 - d_2 - d_3$, so there is perfect correlation between $d_4$ and $d_1 + d_2 + d_3$.

In the case of perfect correlation (i.e., a correlation of +1 or -1, such as in the dummy variable trap), it is not possible to estimate the regression model.

If there is high correlation (close to but not equal to +1 or -1), then the estimation of the regression coefficients is computationally difficult. In fact, some software (notably Microsoft Excel) may give highly inaccurate estimates of the coefficients. Most reputable statistical software will use algorithms to limit the effect of multicollinearity on the coefficient estimates, but you do need to be careful. The major software packages such as R, SPSS, SAS and Stata all use estimation algorithms to avoid the problem as much as possible.

When multicollinearity is present, the uncertainty associated with individual regression coefficients will be large. This is because they are difficult to estimate. Consequently, statistical tests (e.g., t-tests) on regression coefficients are unreliable. (In forecasting we are rarely interested in such tests.) Also, it will not be possible to make accurate statements about the contribution of each separate predictor to the forecast.

Forecasts will be unreliable if the values of the future predictors are outside the range of the historical values of the predictors. For example, suppose you have fitted a regression model with predictors $x_1$ and $x_2$ which are highly correlated with each other, and suppose that the values of $x_1$ in the training data ranged between 0 and 100. Then forecasts based on $x_1 > 100$ or $x_1 < 0$ will be unreliable. It is always a little dangerous when future values of the predictors lie much outside the historical range, but it is especially problematic when multicollinearity is present.

Note that if you are using good statistical software, if you are not interested in the specific contributions of each predictor, and if the future values of your predictor variables are within their historical ranges, there is nothing to worry about — multicollinearity is not a problem except when there is perfect correlation.

## 7.9 Matrix formulation

(check html page)