Dynamic regression models: Introduction

EC 361–001

Prof. Santetti
Spring 2024

Materials

Required readings:

- `Hyndman & Athanasopoulos, ch. 7`
    - sections 7.1—7.5.
- `Hyndman & Athanasopoulos, ch. 10`
    - section 10.1.

Motivation

Motivation

After studying **ARIMA** models, we have seen that we *can* (and *should*, when possible) include **information from past observations of a series** for modeling/forecasting purposes.

However, one **limitation** of such models is that they do not allow for the inclusion of **exogenous factors**.

By **exogenous** factors we mean including other **explanatory variables** that may be relevant to model and forecast a variable's behavior over time.

To this end, we turn our attention to **dynamic regression models**.

Time-series regression models

Time-series regression models

When applied to *time series data*, a **regression model** looks like the following:

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

In words, the **dependent** variable y_t is a **linear function** of k **predictor/independent** variables, as well of a **stochastic** error term (ε_t), which is assumed to be **white noise** and **uncorrelated** with the RHS variables.

However, when it comes to time series data, **residual autocorrelation** is a common *issue*.

To *overcome* that issue, one *alternative* is to incorporate such serial correlation into our residuals through **ARIMA modeling**.

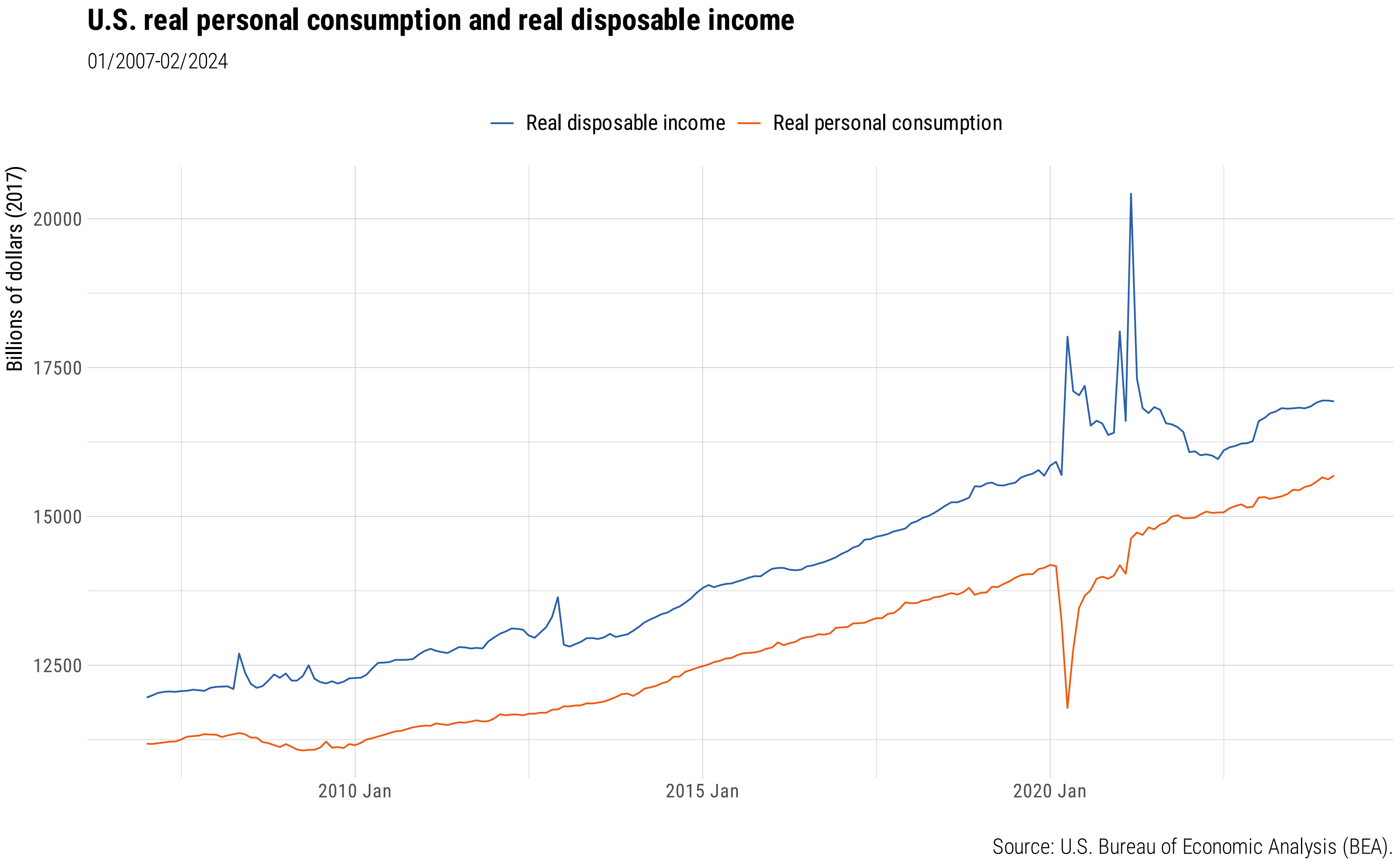
The basics

The basics

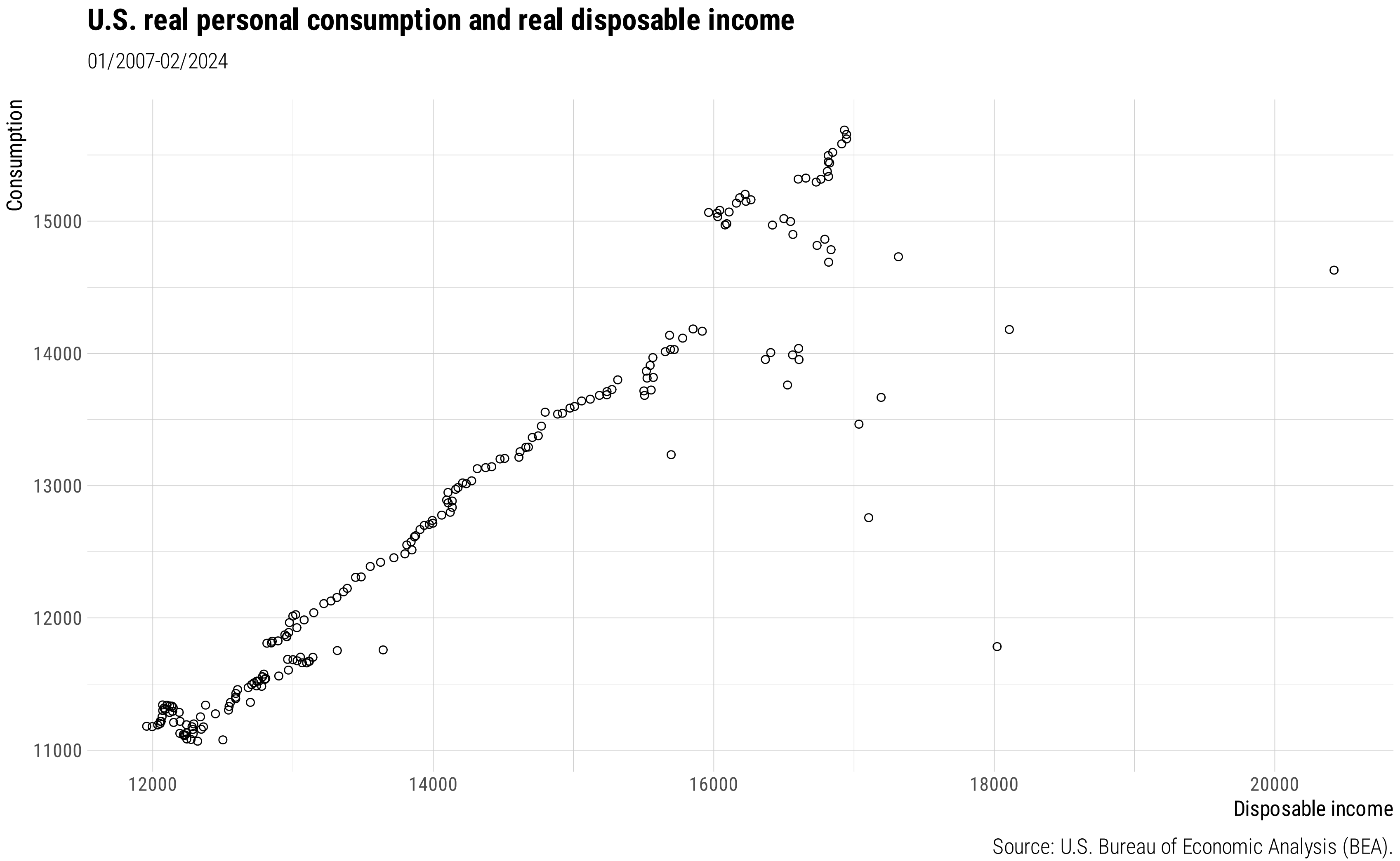
Starting from a **simple regression model**:

$$\text{Consumption}_t = \beta_0 + \beta_1 \text{Disposable Income}_t + \varepsilon_t$$

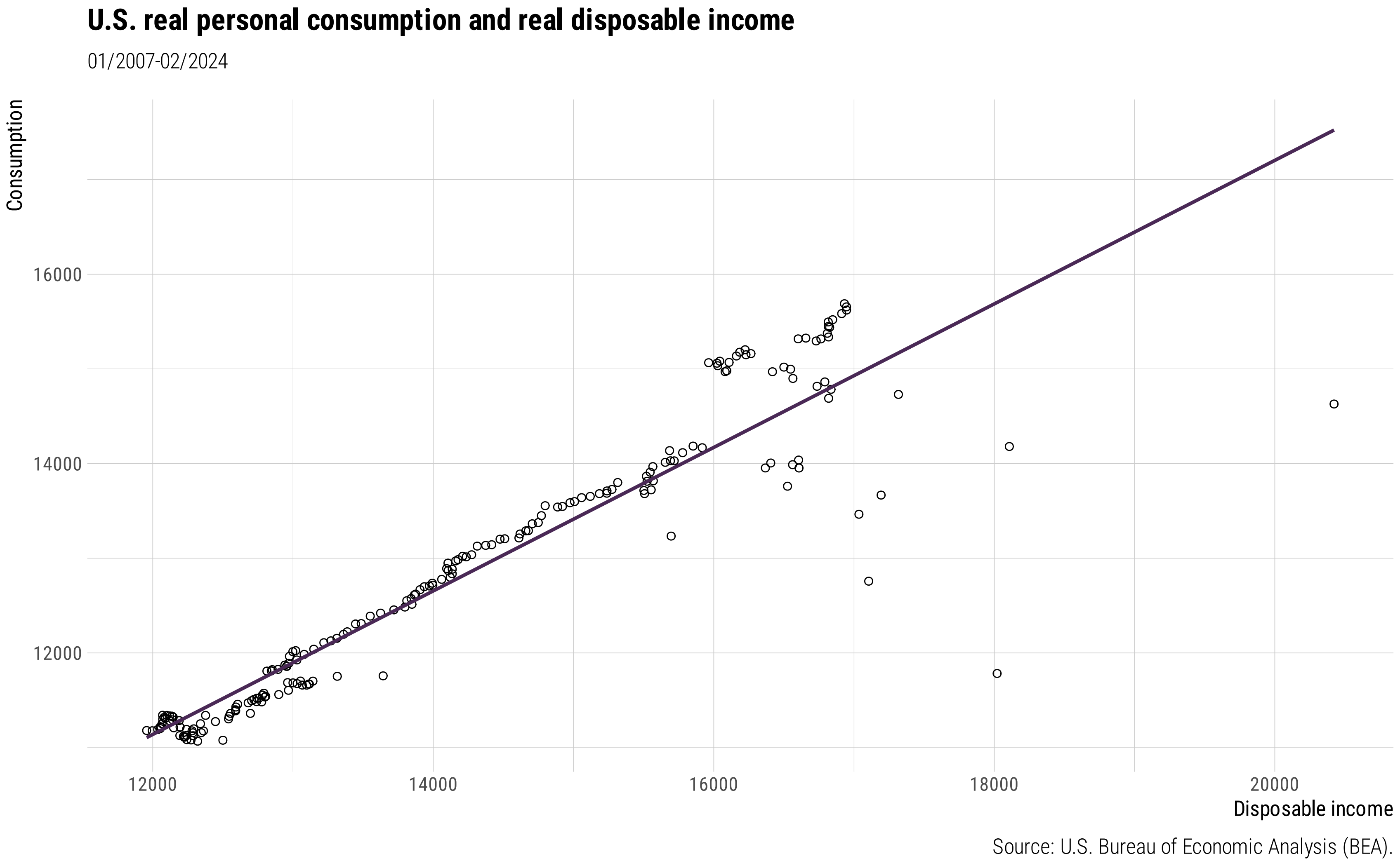
The basics


U.S. real personal consumption and real disposable income
01/2007-02/2024
Real disposable income
Real personal consumption
Billions of dollars (2017)
20000
17500
15000
12500
2010 Jan
2015 Jan
2020 Jan
Source: U.S. Bureau of Economic Analysis (BEA).

The basics


U.S. real personal consumption and real disposable income
01/2007-02/2024
Consumption
15000
14000
13000
12000
11000
12000
14000
16000
18000
20000
Disposable income
Source: U.S. Bureau of Economic Analysis (BEA).

The basics

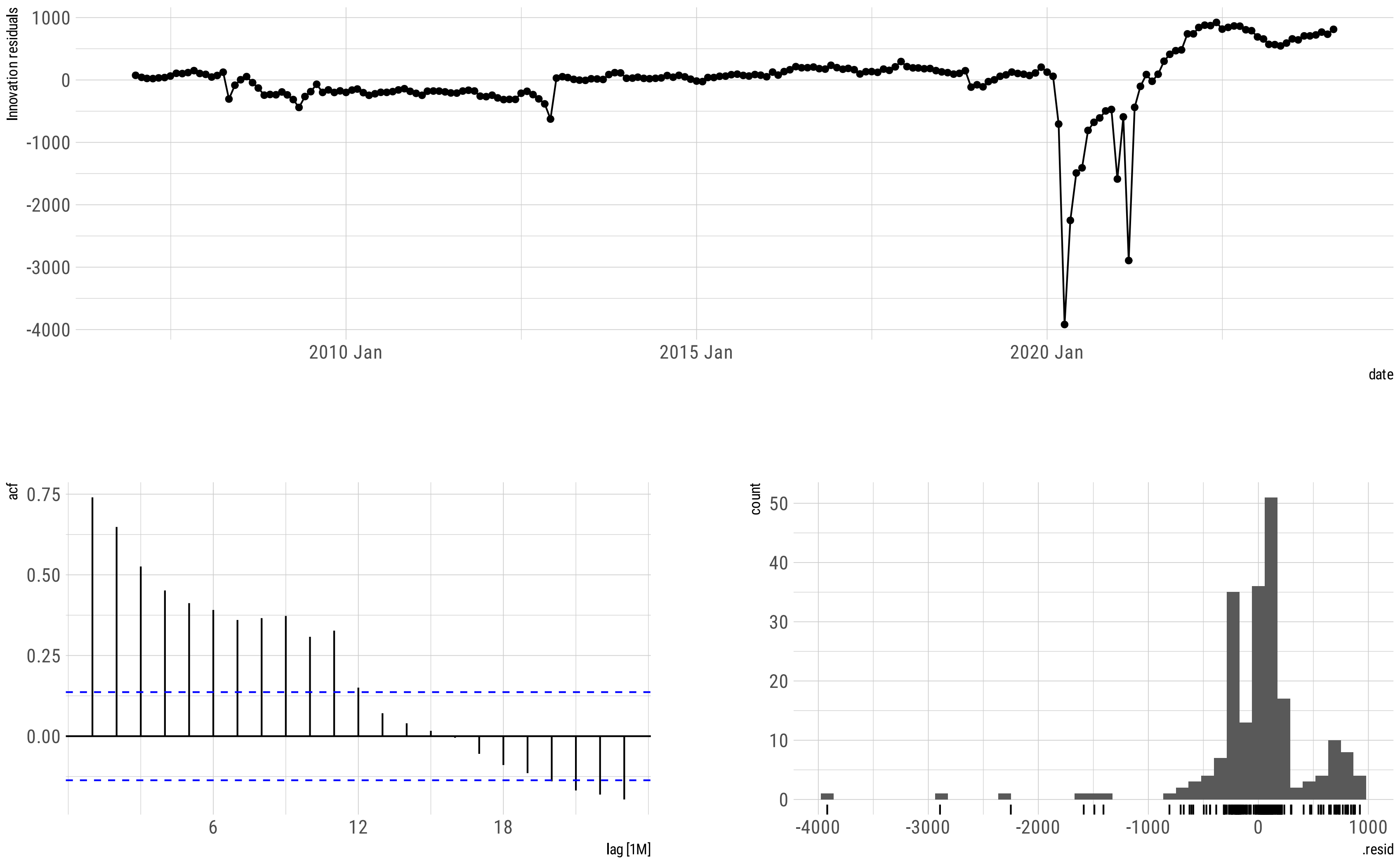

U.S. real personal consumption and real disposable income
01/2007-02/2024
Consumption
16000
14000
12000
12000
14000
16000
18000
20000
Disposable income
Source: U.S. Bureau of Economic Analysis (BEA).

In **R**, the `{fable}` package handles linear regression through the `TSLM()` function.

```r
dat_ts |>
  model(reg = TSLM(cons ~ inc)) |>
  report()
```

```
#> Series: cons
#> Model: TSLM
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -3918.8  -180.9    48.7   153.5   922.6
#>
#> Coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 2.044e+03  3.087e+02   6.622 3.08e-10 ***
#> inc         7.579e-01  2.154e-02  35.191  < 2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 535.2 on 204 degrees of freedom
#> Multiple R-squared: 0.8586,    Adjusted R-squared: 0.8579
#> F-statistic:  1238 on 1 and 204 DF, p-value: < 2.22e-16
```

```
reg_fit ▷ gg_tsresiduals()
```



The basics

Are the residuals **white noise**?

```
reg_fit ▷
  augment() ▷
  features(.innov, ljung_box, lag = 10)
```

```
#> # A tibble: 1 × 3
#>   .model lb_stat lb_pvalue
#>   <chr>    <dbl>     <dbl>
#> 1 reg       481.         0
```

Useful predictors

Useful predictors

When a time series shows **trend** and/or **seasonality**, one good *first step* may be **explicitly** incorporating these features on a regression's right-hand side.

A **linear trend** may be modeled in the following way:

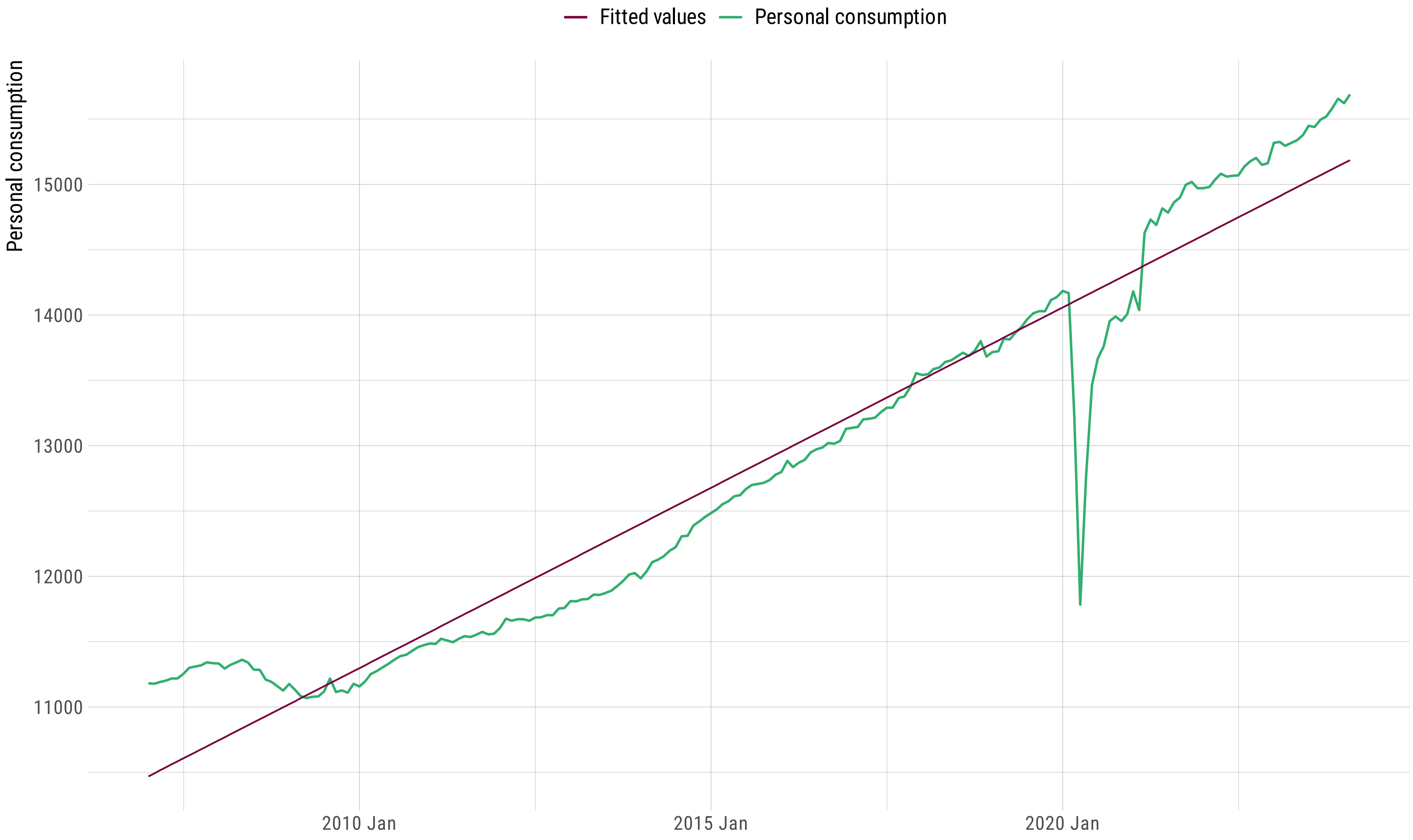
$$y_t = \beta_0 + \beta_1 t + \varepsilon_t$$

where $t = 1, 2, \ldots, T$.

```r
dat_ts |>
  model(reg_trend = TSLM(cons ~ trend())) |>
  report()
```

```
#> Series: cons
#> Model: TSLM
#>
#> Residuals:
#>      Min       1Q   Median       3Q      Max
#> -2343.74  -195.48   -77.25   311.25   712.36
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 10445.63      51.32  203.52   <2e-16 ***
#> trend()        23.01       0.43   53.51   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 367 on 204 degrees of freedom
#> Multiple R-squared: 0.9335,    Adjusted R-squared: 0.9332
#> F-statistic:  2864 on 1 and 204 DF, p-value: < 2.22e-16
```

Useful predictors



Useful predictors

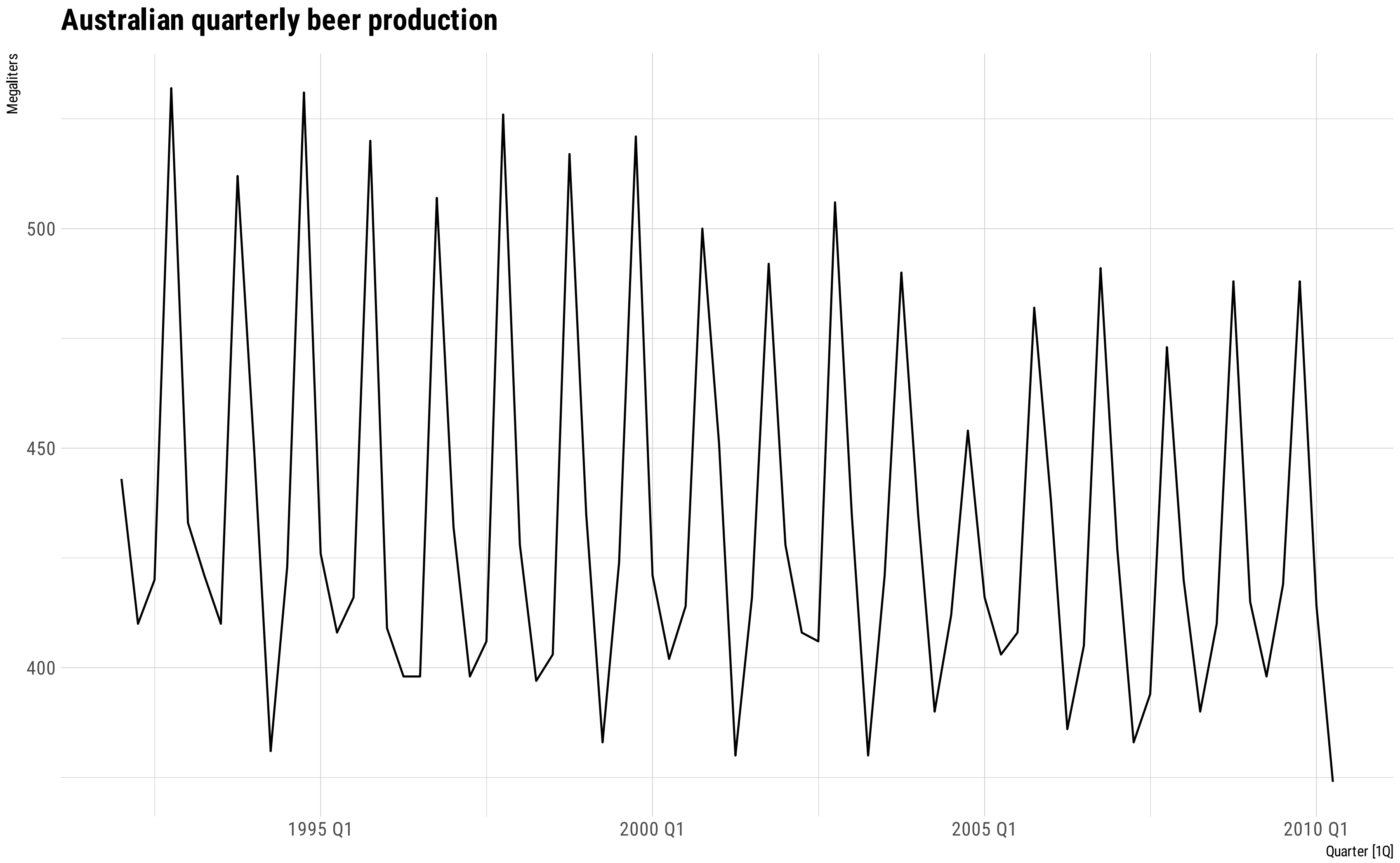
Seasonality can be easily handled with the use of **dummy (binary)** variables.

The idea is to **encode** specific seasonal periods with either *1* or *0* values.

- In R, the `TSLM()` function takes care of seasonal dummies for us.

In terms of **interpretation**, each of the coefficients associated with the dummy variables is a measure of the effect of *that category* **relative to** the *omitted* category.

Useful predictors



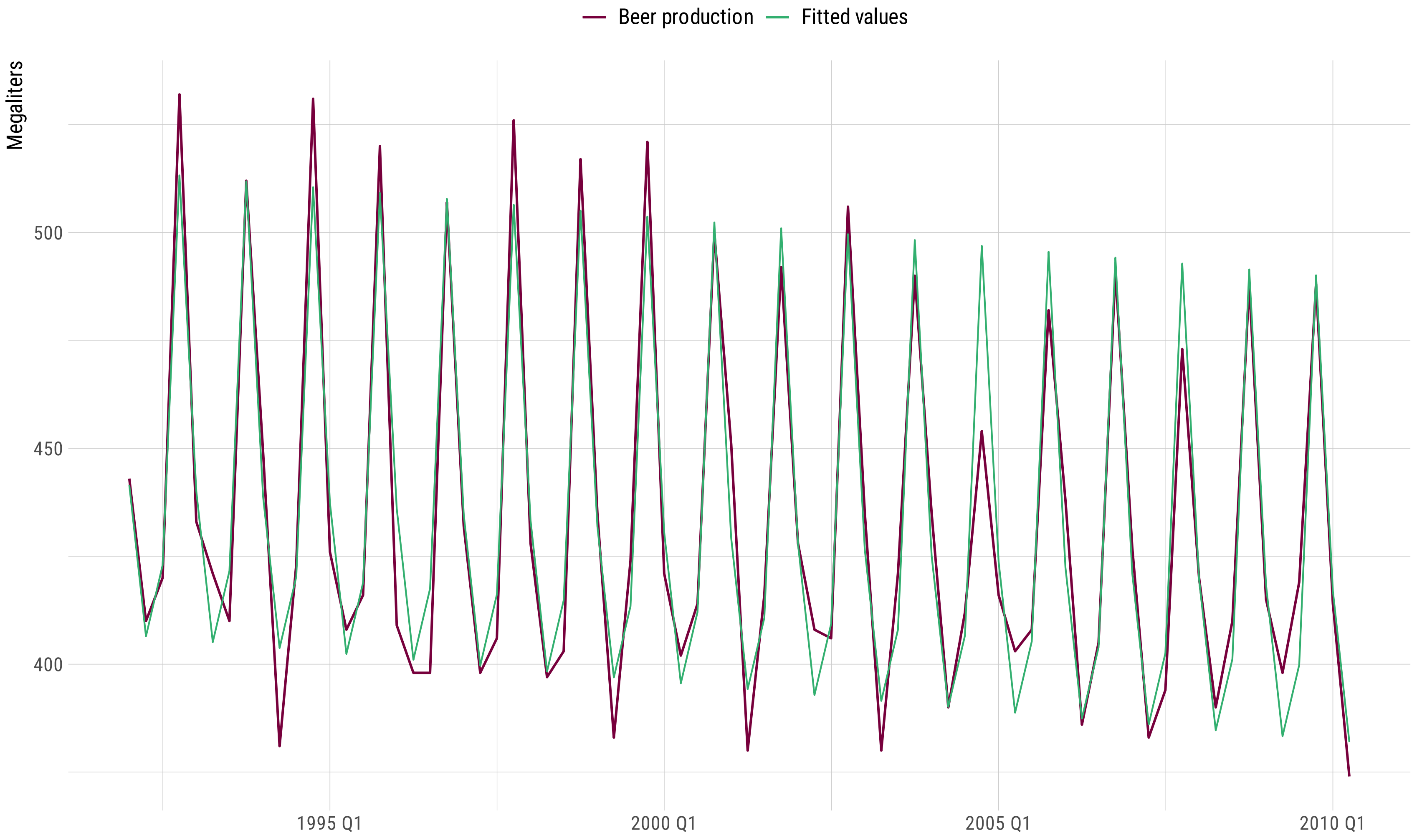
```r
recent_production |>
  model(reg_season = TSLM(Beer ~ season())) |>
  report()
```

```
#> Series: Beer
#> Model: TSLM
#>
#> Residuals:
#>      Min       1Q   Median       3Q      Max
#> -47.6667 -10.4167  -0.2997   8.7449  30.3333
#>
#> Coefficients:
#>               Estimate Std. Error t value Pr(>|t|)
#> (Intercept)    429.211      3.271 131.234  < 2e-16 ***
#> season()year2  -35.000      4.625  -7.567 1.14e-10 ***
#> season()year3  -17.822      4.689  -3.801 0.000305 ***
#> season()year4   72.456      4.689  15.452  < 2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 14.26 on 70 degrees of freedom
#> Multiple R-squared: 0.8957,    Adjusted R-squared: 0.8912
#> F-statistic: 200.3 on 3 and 70 DF, p-value: < 2.22e-16
```

```r
recent_production |>
  model(reg_season = TSLM(Beer ~ trend() + season())) |>
  report()
```

```
#> 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
```

Useful predictors



Useful predictors

```
recent_production ▷
  model(reg_season = TSLM(Beer ~ trend() + season())) ▷
  augment() ▷
  features(.innov, ljung_box, lag = 2 * 4)
```

```
#> # A tibble: 1 × 3
#>   .model     lb_stat lb_pvalue
#>   <chr>        <dbl>     <dbl>
#> 1 reg_season    10.4     0.240
```

Are residuals **white noise**?

Useful predictors

An alternative way of **modeling seasonality** is to incorporate **Fourier terms**.

These are *sine* and *cosine* terms used to approximate **periodic functions**.

As time series show *periodic* behavior, Fourier terms are well-suited for **seasonal series**.

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

$$x_{1,t} = \sin\left(\frac{2\pi t}{m}\right), x_{2,t} = \cos\left(\frac{2\pi t}{m}\right), x_{3,t} = \sin\left(\frac{4\pi t}{m}\right),$$

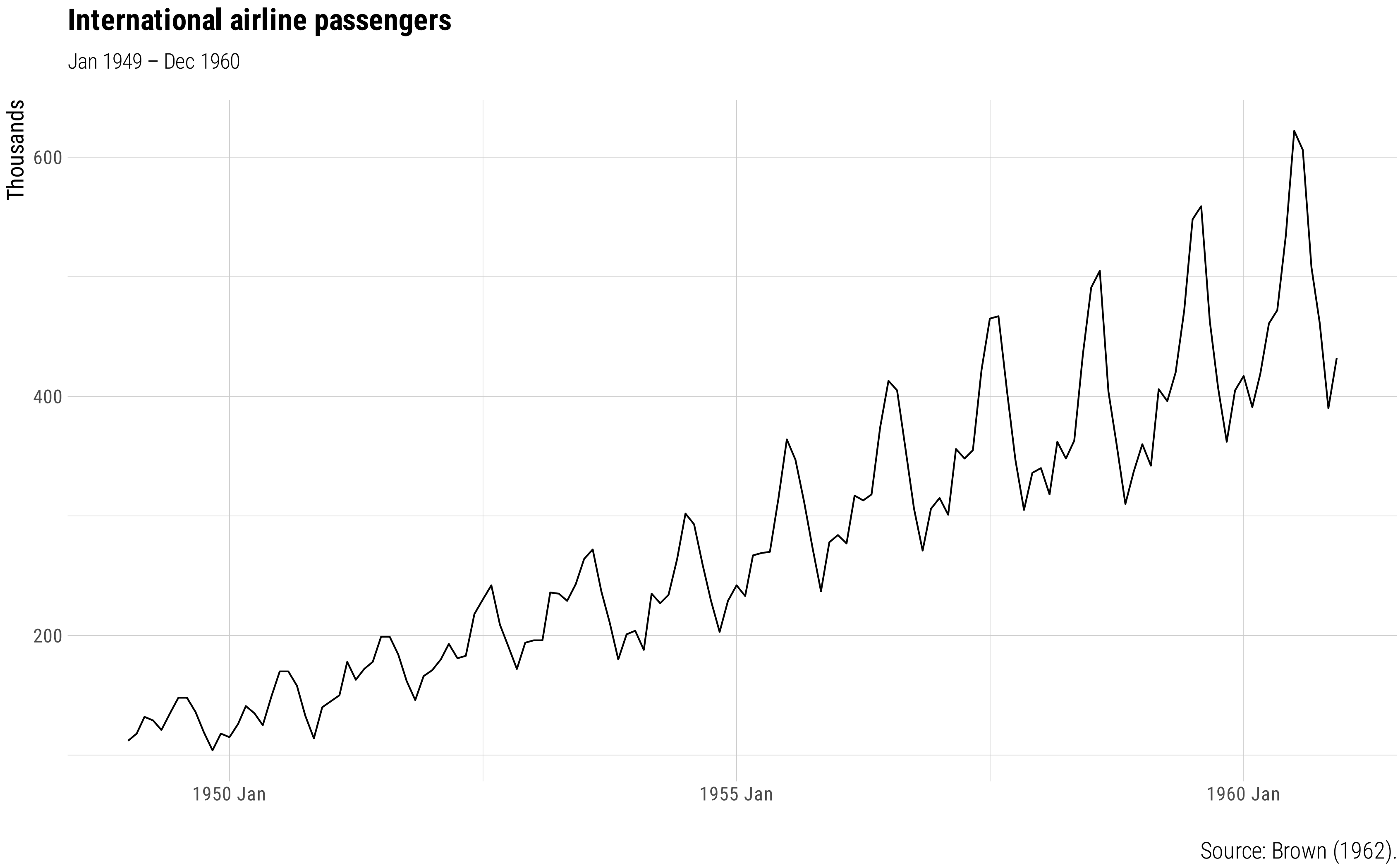
$$x_{4,t} = \cos\left(\frac{4\pi t}{m}\right), x_{5,t} = \sin\left(\frac{6\pi t}{m}\right), x_{6,t} = \cos\left(\frac{6\pi t}{m}\right),$$

Useful predictors

The **maximum** number of Fourier terms allowed is given by $K = m/2$, where m is the number of seasonal periods in a year.

A regression model containing Fourier terms is often called a **harmonic regression**.

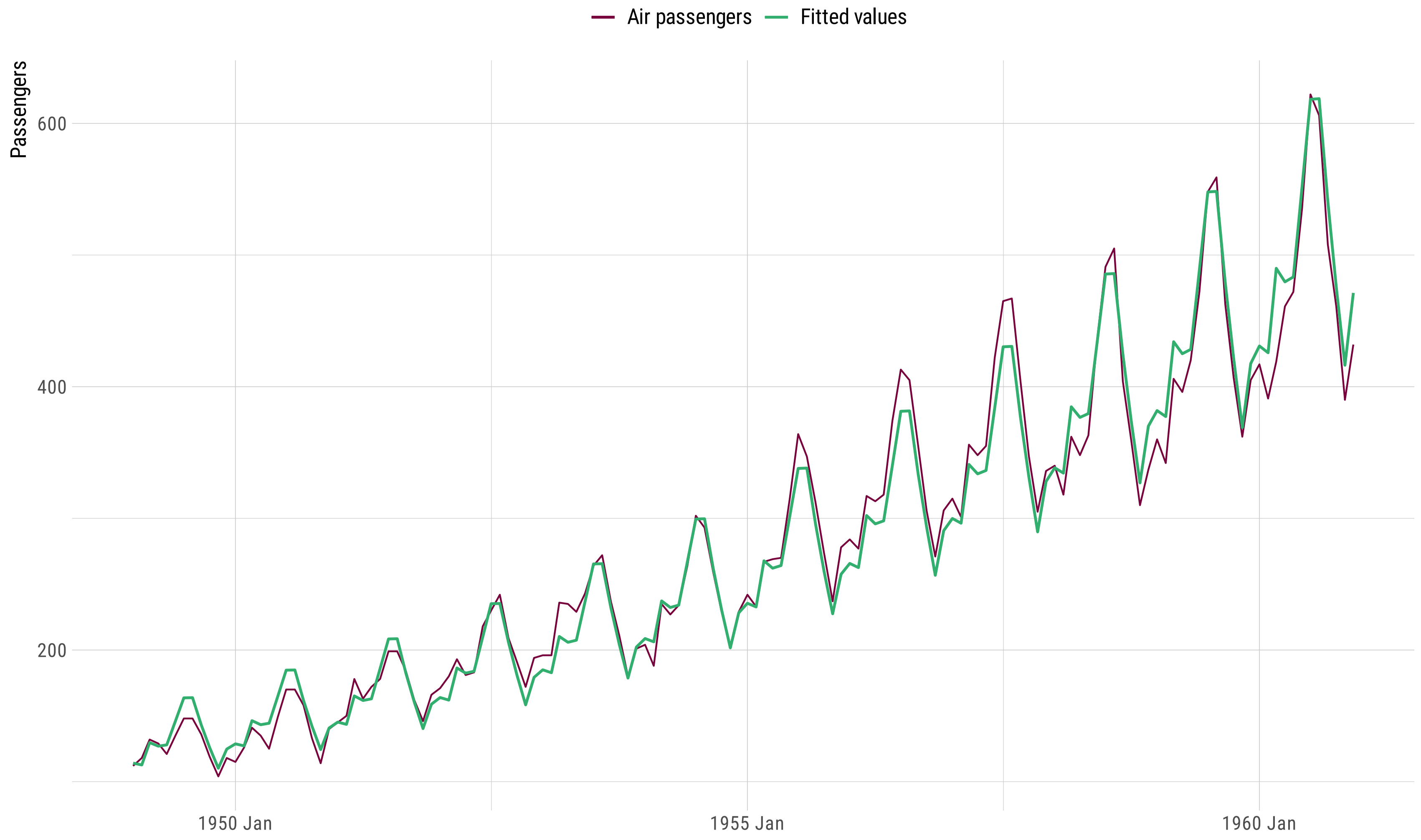
Useful predictors



```r
air_ts |>
  model(reg = TSLM(log(passengers) ~ trend() + season())) |>
  report()
```

```
#> Series: passengers
#> Model: TSLM
#> Transformation: log(passengers)
#>
#> Residuals:
#>       Min        1Q    Median        3Q       Max
#> -0.156370 -0.041016  0.003677  0.044069  0.132324
#>
#> Coefficients:
#>                  Estimate Std. Error t value Pr(>|t|)
#> (Intercept)     4.7267804  0.0188935 250.180  < 2e-16 ***
#> trend()         0.0100688  0.0001193  84.399  < 2e-16 ***
#> season()year2  -0.0220548  0.0242109  -0.911  0.36400
#> season()year3   0.1081723  0.0242118   4.468 1.69e-05 ***
#> season()year4   0.0769034  0.0242132   3.176  0.00186 **
#> season()year5   0.0745308  0.0242153   3.078  0.00254 **
#> season()year6   0.1966770  0.0242179   8.121 2.98e-13 ***
#> season()year7   0.3006193  0.0242212  12.411  < 2e-16 ***
#> season()year8   0.2913245  0.0242250  12.026  < 2e-16 ***
#> season()year9   0.1466899  0.0242294   6.054 1.39e-08 ***
#> season()year10  0.0085316  0.0242344   0.352  0.72537
#> season()year11 -0.1351861  0.0242400  -5.577 1.34e-07 ***
#> season()year12 -0.0213211  0.0242461  -0.879  0.38082
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.0593 on 131 degrees of freedom
#> Multiple R-squared: 0.9835,    Adjusted R-squared: 0.982
#> F-statistic: 649.4 on 12 and 131 DF, p-value: < 2.22e-16
```

Useful predictors

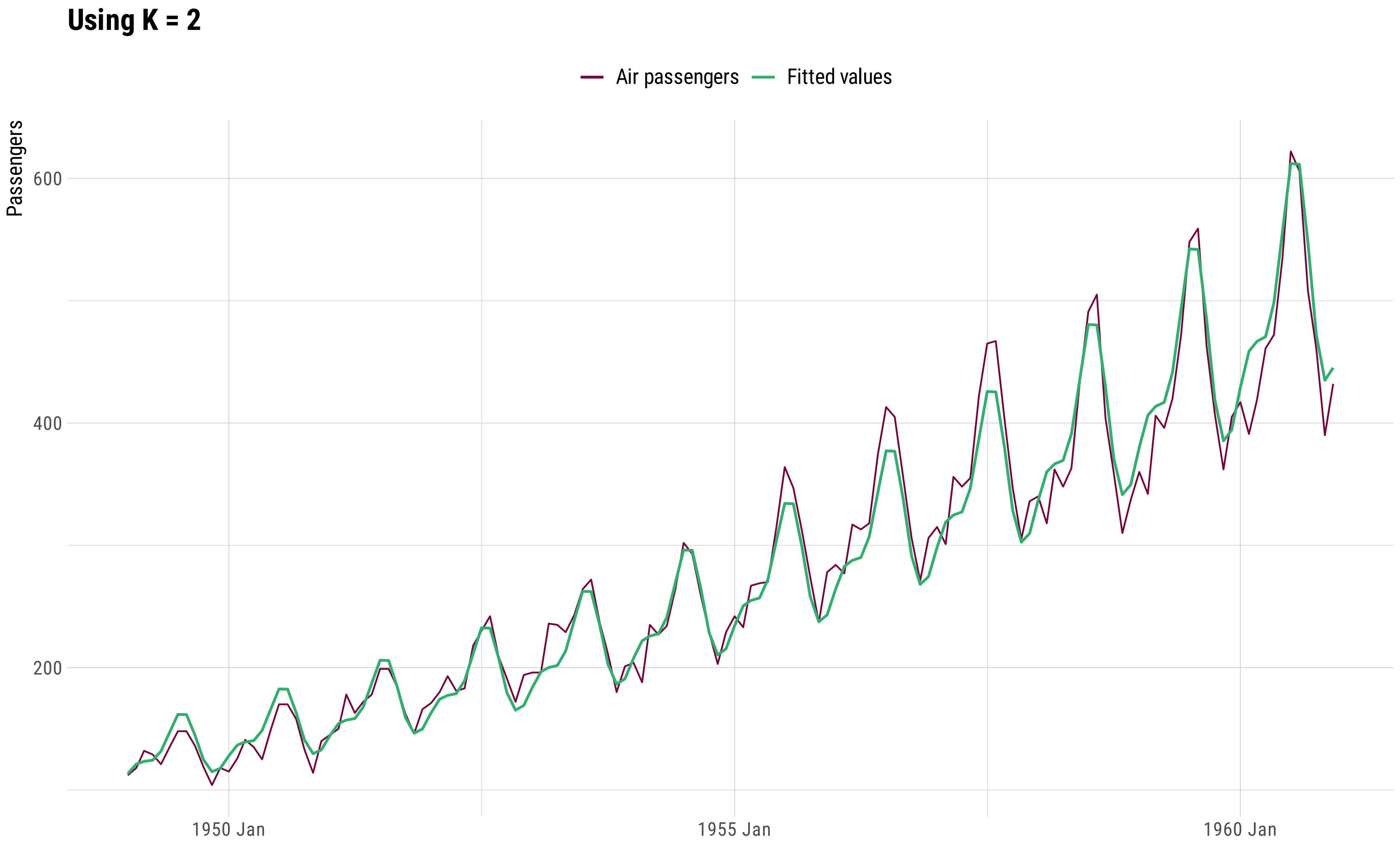


Useful predictors

```r
air_ts |>
  model(reg = TSLM(log(passengers) ~ trend() + fourier(K = 2))) |>
  report()
```

```
#> Series: passengers
#> Model: TSLM
#> Transformation: log(passengers)
#>
#> Residuals:
#>       Min        1Q    Median        3Q       Max
#> -0.172892 -0.040363  0.002417  0.046796  0.164906
#>
#> Coefficients:
#>                       Estimate Std. Error t value Pr(>|t|)
#> (Intercept)          4.8112150  0.0114209 421.262  < 2e-16 ***
#> trend()              0.0100822  0.0001368  73.725  < 2e-16 ***
#> fourier(K = 2)C1_12 -0.1474737  0.0080184 -18.392  < 2e-16 ***
#> fourier(K = 2)S1_12  0.0282074  0.0080334   3.511 0.000603 ***
#> fourier(K = 2)C2_12  0.0567457  0.0080184   7.077 6.79e-11 ***
#> fourier(K = 2)S2_12  0.0591195  0.0080207   7.371 1.41e-11 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.06803 on 138 degrees of freedom
#> Multiple R-squared: 0.9771,    Adjusted R-squared: 0.9763
#> F-statistic:  1177 on 5 and 138 DF, p-value: < 2.22e-16
```

Useful predictors

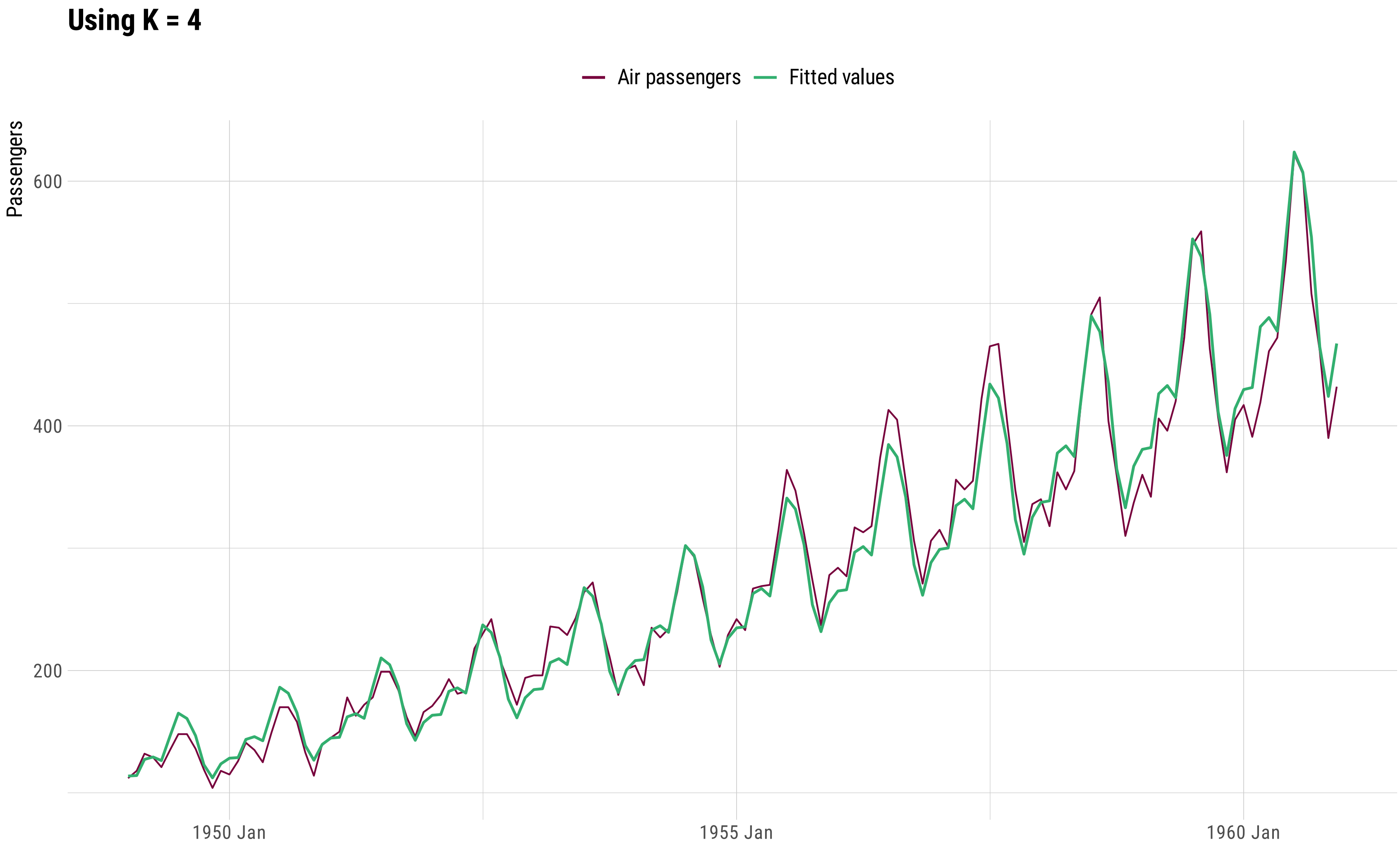


Useful predictors

```
air_ts |>
  model(reg = TSLM(log(passengers) ~ trend() + fourier(K = 4))) |>
  report()
```

```
#> Series: passengers
#> Model: TSLM
#> Transformation: log(passengers)
#>
#> Residuals:
#>        Min         1Q     Median         3Q        Max
#> -0.1379894 -0.0416537  0.0004086  0.0446304  0.1338178
#>
#> Coefficients:
#>                      Estimate Std. Error t value Pr(>|t|)
#> (Intercept)          4.8121301  0.0102284 470.467  < 2e-16 ***
#> trend()              0.0100696  0.0001225  82.208  < 2e-16 ***
#> fourier(K = 4)C1_12 -0.1474864  0.0071785 -20.546  < 2e-16 ***
#> fourier(K = 4)S1_12  0.0281603  0.0071920   3.916 0.000143 ***
#> fourier(K = 4)C2_12  0.0567331  0.0071785   7.903 8.84e-13 ***
#> fourier(K = 4)S2_12  0.0590977  0.0071806   8.230 1.46e-13 ***
#> fourier(K = 4)C3_12 -0.0087300  0.0071785  -1.216 0.226072
#> fourier(K = 4)S3_12 -0.0272914  0.0071785  -3.802 0.000217 ***
#> fourier(K = 4)C4_12  0.0111072  0.0071785   1.547 0.124154
#> fourier(K = 4)S4_12 -0.0319853  0.0071778  -4.456 1.75e-05 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.0609 on 134 degrees of freedom
#> Multiple R-squared: 0.9822,	Adjusted R-squared: 0.981
#> F-statistic: 819.9 on 9 and 134 DF, p-value: < 2.22e-16
```

Useful predictors

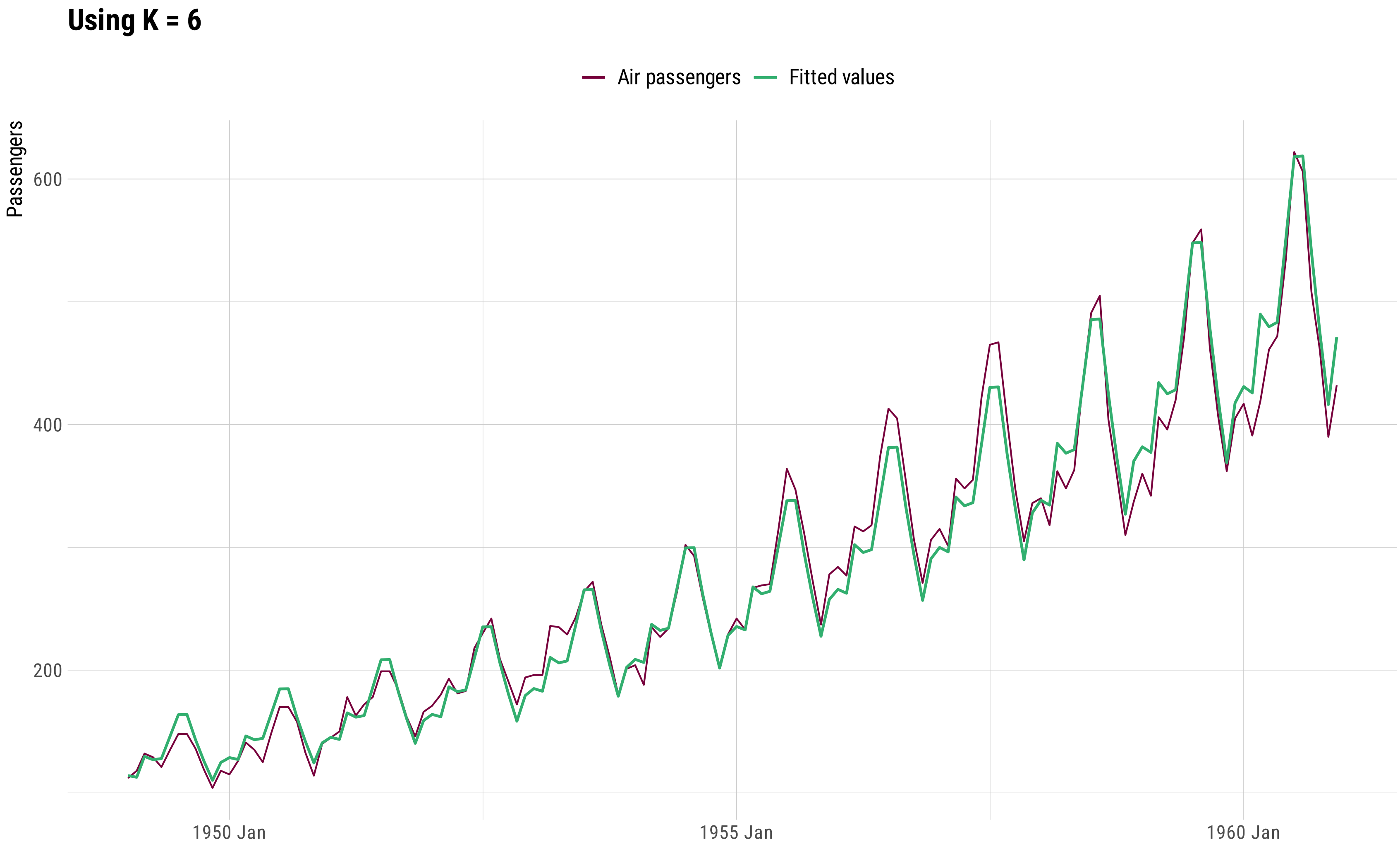


Useful predictors

```
air_ts |>
  model(reg = TSLM(log(passengers) ~ trend() + fourier(K = 6))) |>
  report()
```

```
#> Series: passengers
#> Model: TSLM
#> Transformation: log(passengers)
#>
#> Residuals:
#>       Min        1Q    Median        3Q       Max
#> -0.156370 -0.041016  0.003677  0.044069  0.132324
#>
#> Coefficients:
#>                        Estimate Std. Error t value Pr(>|t|)
#> (Intercept)           4.8121876  0.0099616 483.076  < 2e-16 ***
#> trend()               0.0100688  0.0001193  84.399  < 2e-16 ***
#> fourier(K = 6)C1_12 -0.1474871  0.0069900 -21.100  < 2e-16 ***
#> fourier(K = 6)S1_12  0.0281573  0.0070032   4.021 9.74e-05 ***
#> fourier(K = 6)C2_12  0.0567323  0.0069900   8.116 3.06e-13 ***
#> fourier(K = 6)S2_12  0.0590963  0.0069920   8.452 4.81e-14 ***
#> fourier(K = 6)C3_12 -0.0087308  0.0069900  -1.249  0.21388
#> fourier(K = 6)S3_12 -0.0272922  0.0069900  -3.904  0.00015 ***
#> fourier(K = 6)C4_12  0.0111064  0.0069900   1.589  0.11450
#> fourier(K = 6)S4_12 -0.0319857  0.0069893  -4.576 1.09e-05 ***
#> fourier(K = 6)C5_12  0.0059083  0.0069900   0.845  0.39951
#> fourier(K = 6)S5_12 -0.0212636  0.0069891  -3.042  0.00284 **
#> fourier(K = 6)C6_12 -0.0029362  0.0049423  -0.594  0.55347
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.0593 on 131 degrees of freedom
```

Useful predictors



ARIMA errors

ARIMA errors

In case we have evidence of residual serial correlation, we may write a time-series regression model as follows:

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

where η_t is assumed to be **autocorrelated**, and follows an **ARIMA** process.

For instance, if η_t follows an **ARIMA(1, 1, 1)** process, it can be expressed as

$$\eta'_t = c + \phi_1 \eta'_{t-1} + \theta_1 \varepsilon_{t-1} + \varepsilon_t$$

where ε_t follows a white-noise process.

ARIMA errors

Now, the model has **two error terms**:

1. The error from the *regression model*, which we denote by η_t; and
2. The error term from the *ARIMA* model, which we denote by ε_t.

Only ε_t is assumed to be white noise here.

ARIMA errors

Whenever a regression's **error term** shows **serial correlation**, several problems arise:

- As relevant information is left to the error term, the model is **not well specified**;
    - ***Autocorrelation is information!***
- Inference is unreliable;
    - *p-values*, *t-statistics* are **biased**.

Therefore, applying **Ordinary Least Squares** (OLS) estimation in models with residual autocorrelation is **problematic**.

Instead, we should **model** the autocorrelations in the residual term, so we incorporate such information into our modeling/forecasting.

ARIMA errors

An important **consideration** when estimating a regression with *ARIMA errors* is that all of the variables in the model must first be **stationary**.

Thus, running a unit-root test (such as KPSS) is **mandatory**.

One **common practice** is to difference ***all*** regression variables if ***any*** of them is *non-stationary*.

- The resulting model is then called a "*model in differences*."

On the other hand, a "*model in levels*" denotes a regression with all included variables being stationary *without* any transformation needed.

ARIMA errors

In R, the `{fable}` package handles dynamic regression models with the `ARIMA()` function.

For example, the code

```
ARIMA(y ~ x + pdq(1,1,0))
```

fits a dynamic regression for y_t, controlling for one exogenous variable (x_t), assuming that the residual term η_t follows an **ARIMA(1, 1, 0)** process.

- Let us write out this model.

Next time: More on dynamic regression