Dynamic regression models: Further thoughts

EC 361–001

Prof. Santetti
Spring 2024

Materials

Required readings:

- `Hyndman & Athanasopoulos, ch. 10`
    - sections 10.3; 10.5.

Motivation

Motivation

After studying the **essentials** of *dynamic regression* models, let us wrap up this topic with a few more **practical issues**.

Harmonic regression with ARIMA errors

Harmonic regression with ARIMA errors

We have already seen that **harmonic regression** (i.e., using *Fourier* terms) performs well for modeling **seasonality**.

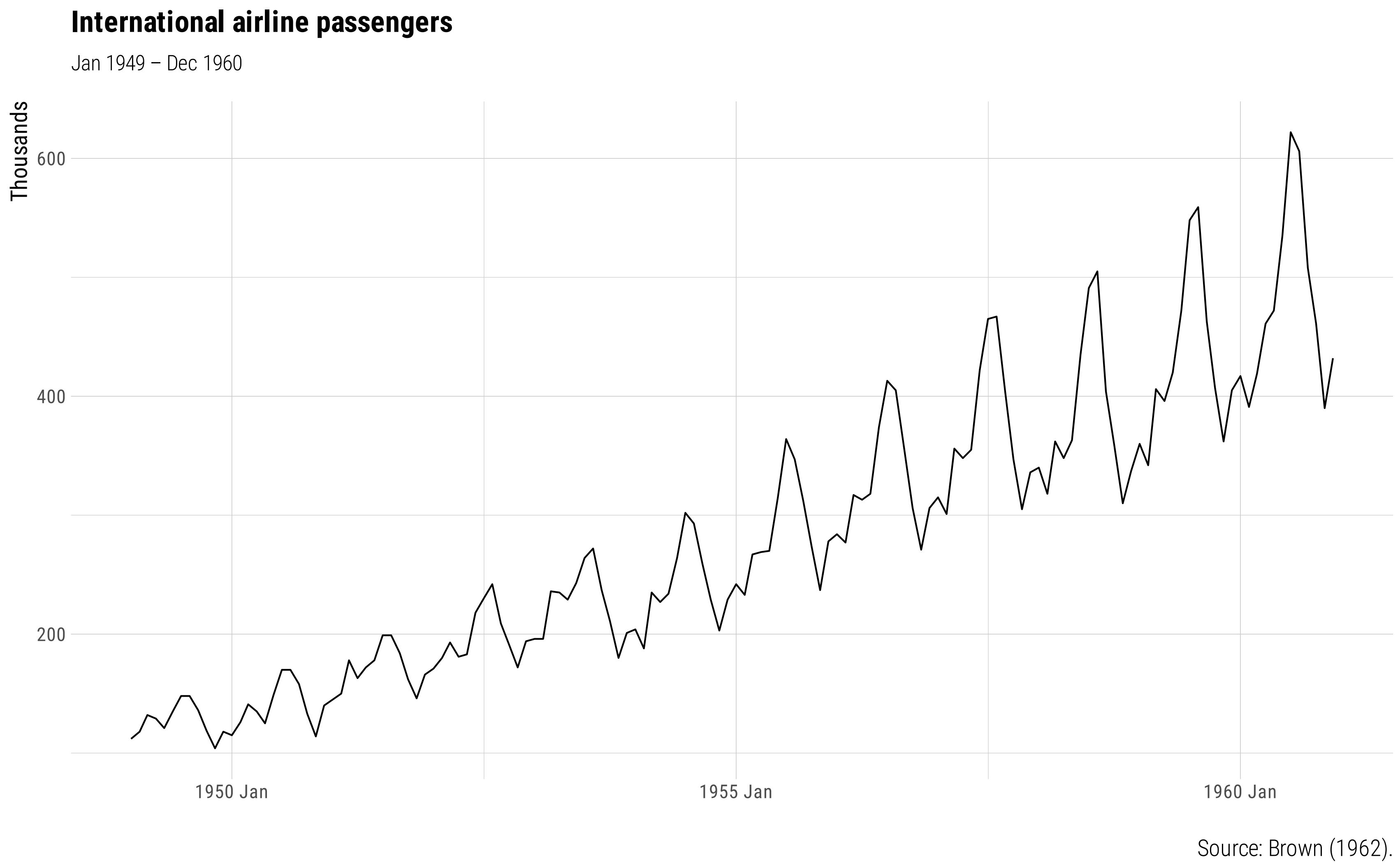
- Especially *long* seasonal periods (*weekly* and *monthly* seasonal data, for example).

One **limitation**, however, is that harmonic regression assumes that the seasonal component is **fixed** over time (i.e., its variance is constant).

With this in mind, harmonic regression is a *useful tool* for seasonal time series.

Fourier terms can be used **along with** *ARIMA errors* to capture several important dynamics of the time series at hand.

Harmonic regression with ARIMA errors



Harmonic regression with ARIMA errors

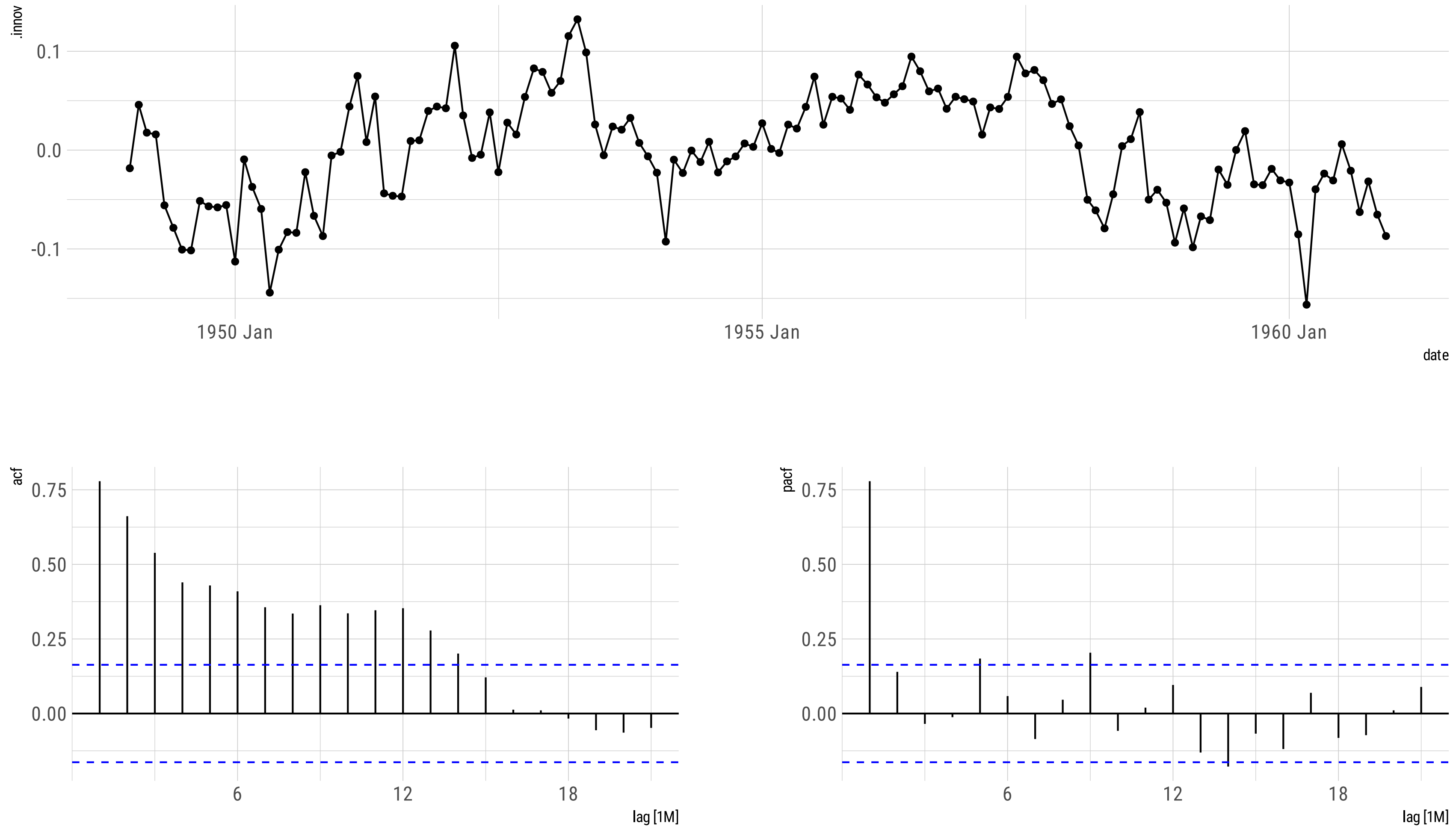
- Let us start a forecasting exercise for the **air passengers** data set with a *straightforward* regression model.

```
air_tslm ← air_ts ▷
  model(reg = TSLM(log(passengers) ~ trend() + season()))
```

```
air_tslm ▷
  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
```

```r
air_tslm ▷
  augment() ▷
  gg_tsdisplay(.innov, plot_type = "partial")
```



Harmonic regression with ARIMA errors

- Testing for **residual serial correlation**:

```
air_tslm ▷
  augment() ▷
  features(.innov, ljung_box, lag = 2 * 12)
```

```
#> # A tibble: 1 × 3
#>   .model lb_stat lb_pvalue
#>   <chr>    <dbl>     <dbl>
#> 1 reg       417.         0
```

What do we conclude?

Harmonic regression with ARIMA errors

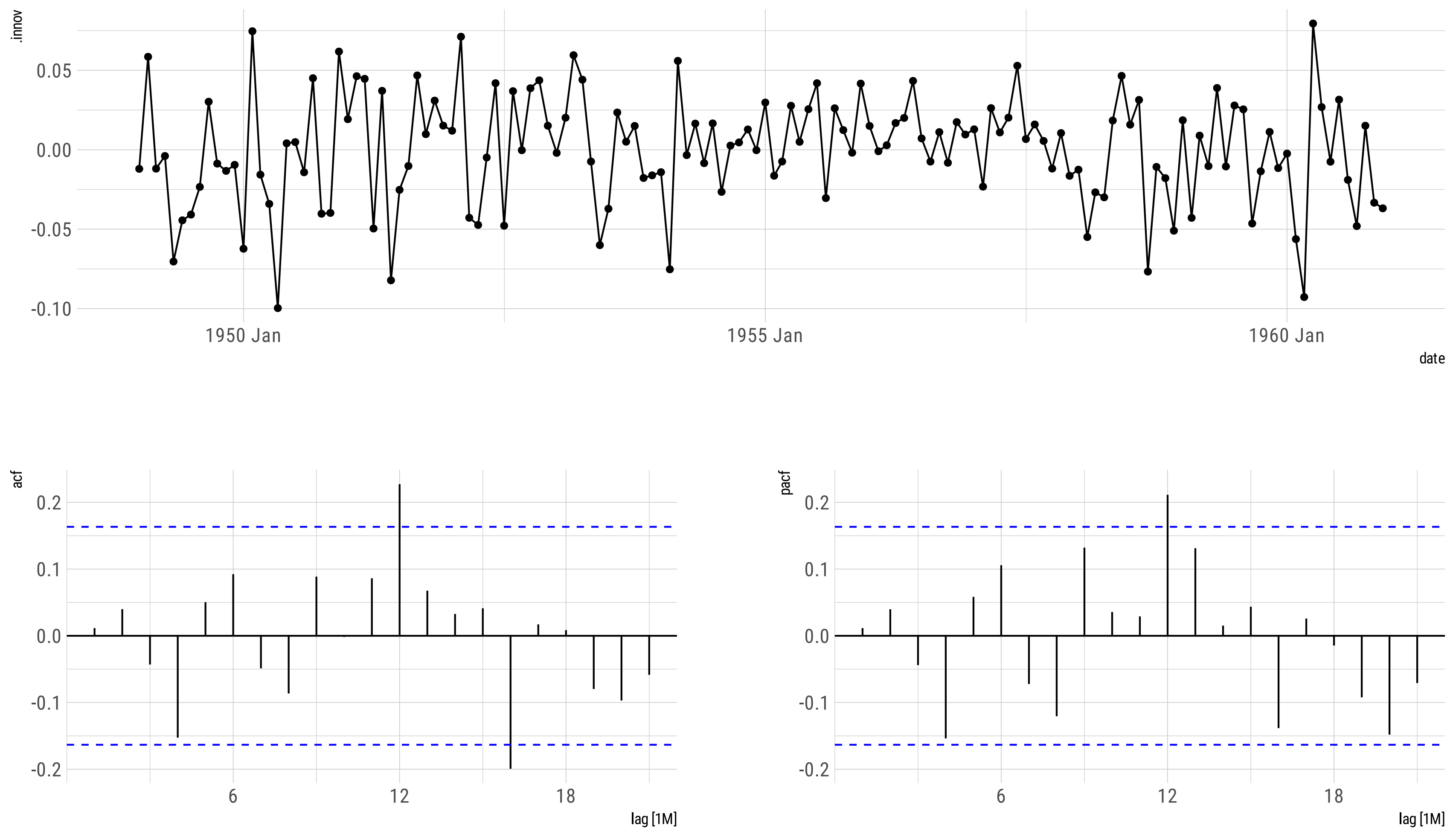
Moving on to a model with **ARIMA errors**:

```r
air_arima <- air_ts |>
  model(arima_errors = ARIMA(log(passengers) ~ trend() + season() + PDQ(0, 0, 0))) # setting PDQ() with zeros, as
                                                                                    # the "season()" function is
                                                                                    # taking care of seasonality.
```

```
air_arima ▷
  report()
```

```
#> Series: passengers
#> Model: LM w/ ARIMA(2,0,0) errors
#> Transformation: log(passengers)
#>
#> Coefficients:
#>          ar1     ar2  trend()  season()year2  season()year3  season()year4  season()year5  season()year6  season()year7  season()year8  season()year9  seas
#>       0.6746  0.1438    1e-02        -0.0211         0.1099         0.0794         0.0777         0.2004         0.3047         0.2958         0.1515
#> s.e.  0.0829  0.0836    4e-04         0.0106         0.0128         0.0145         0.0155         0.0161         0.0163         0.0162         0.0156
#>       season()year11  season()year12  intercept
#>              -0.1299         -0.0158     4.7282
#> s.e.          0.0131          0.0109     0.0306
#>
#> sigma^2 estimated as 0.001337:  log likelihood=279.55
#> AIC=-527.11   AICc=-522.82   BIC=-479.59
```

```
air_arima |>
  augment() |>
  gg_tsdisplay(.innov, plot_type = "partial")
```



Harmonic regression with ARIMA errors

- Testing for **residual serial correlation**:

```
air_arima ▷
  augment() ▷
  features(.innov, ljung_box, lag = 2 * 12)
```

```
#> # A tibble: 1 × 3
#>   .model       lb_stat lb_pvalue
#>   <chr>          <dbl>     <dbl>
#> 1 arima_errors    36.6    0.0478
```

What do we conclude?

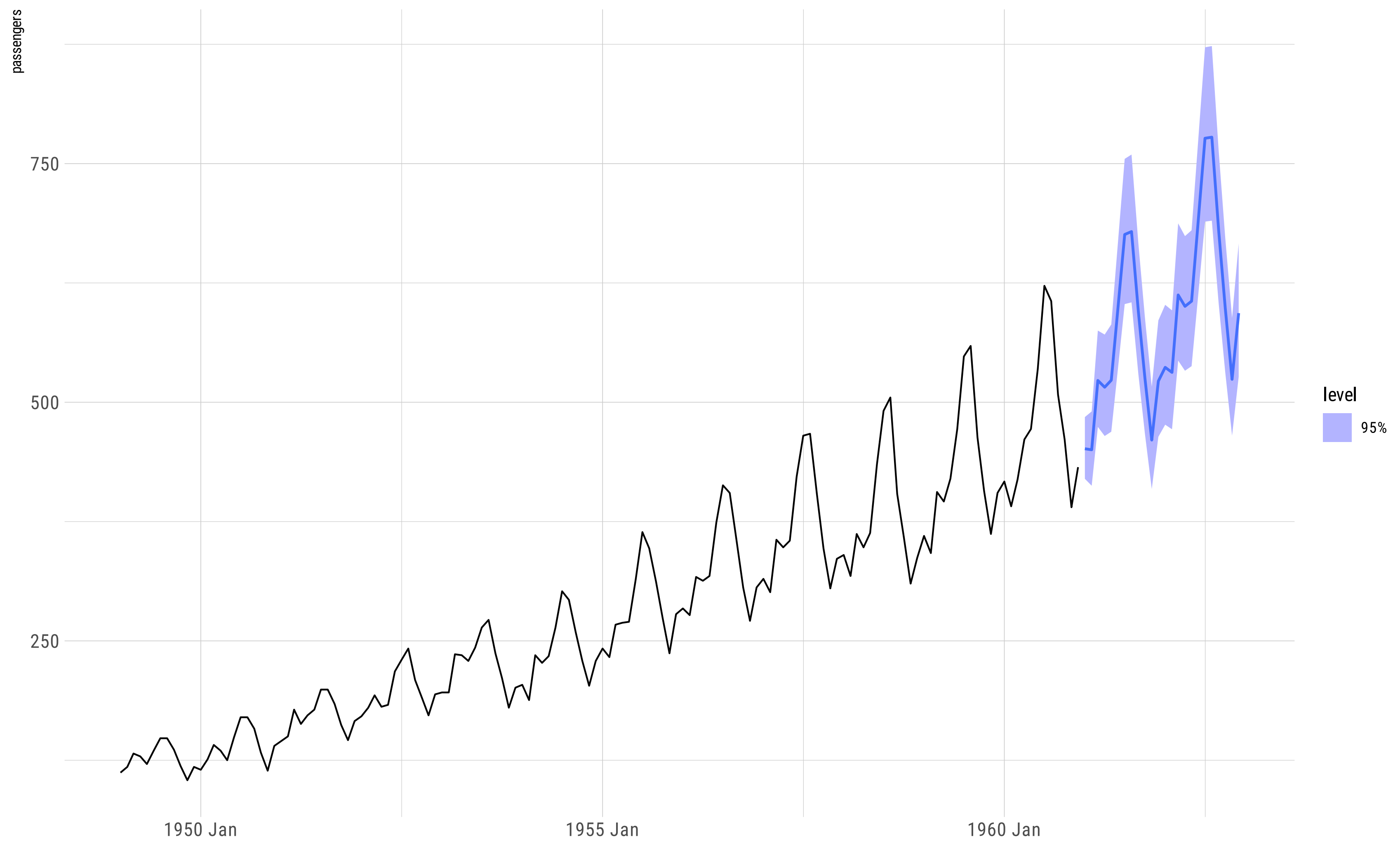
Harmonic regression with ARIMA errors

In case we forecast the time series at hand based on a model *with residual serial correlation*, **prediction intervals** will be **unreliable**.

```
air_arima ▷
  forecast(h = 24) ▷
  head(6)
```

```
#> # A fable: 6 x 4 [1M]
#> # Key:     .model [1]
#>   .model           date        passengers .mean
#>   <chr>           <mth>            <dist> <dbl>
#> 1 arima_errors 1961 Jan t(N(6.1, 0.0013))  451.
#> 2 arima_errors 1961 Feb t(N(6.1, 0.0019))  450.
#> 3 arima_errors 1961 Mar t(N(6.3, 0.0024))  523.
#> 4 arima_errors 1961 Apr t(N(6.2, 0.0028))  516.
#> 5 arima_errors 1961 May  t(N(6.3, 0.003))  523.
#> 6 arima_errors 1961 Jun t(N(6.4, 0.0032))  600.
```

```r
air_arima ▷ forecast(h = 24) ▷ autoplot(air_ts, level = 95, linewidth = .8) + labs(x = "")
```



Harmonic regression with ARIMA errors

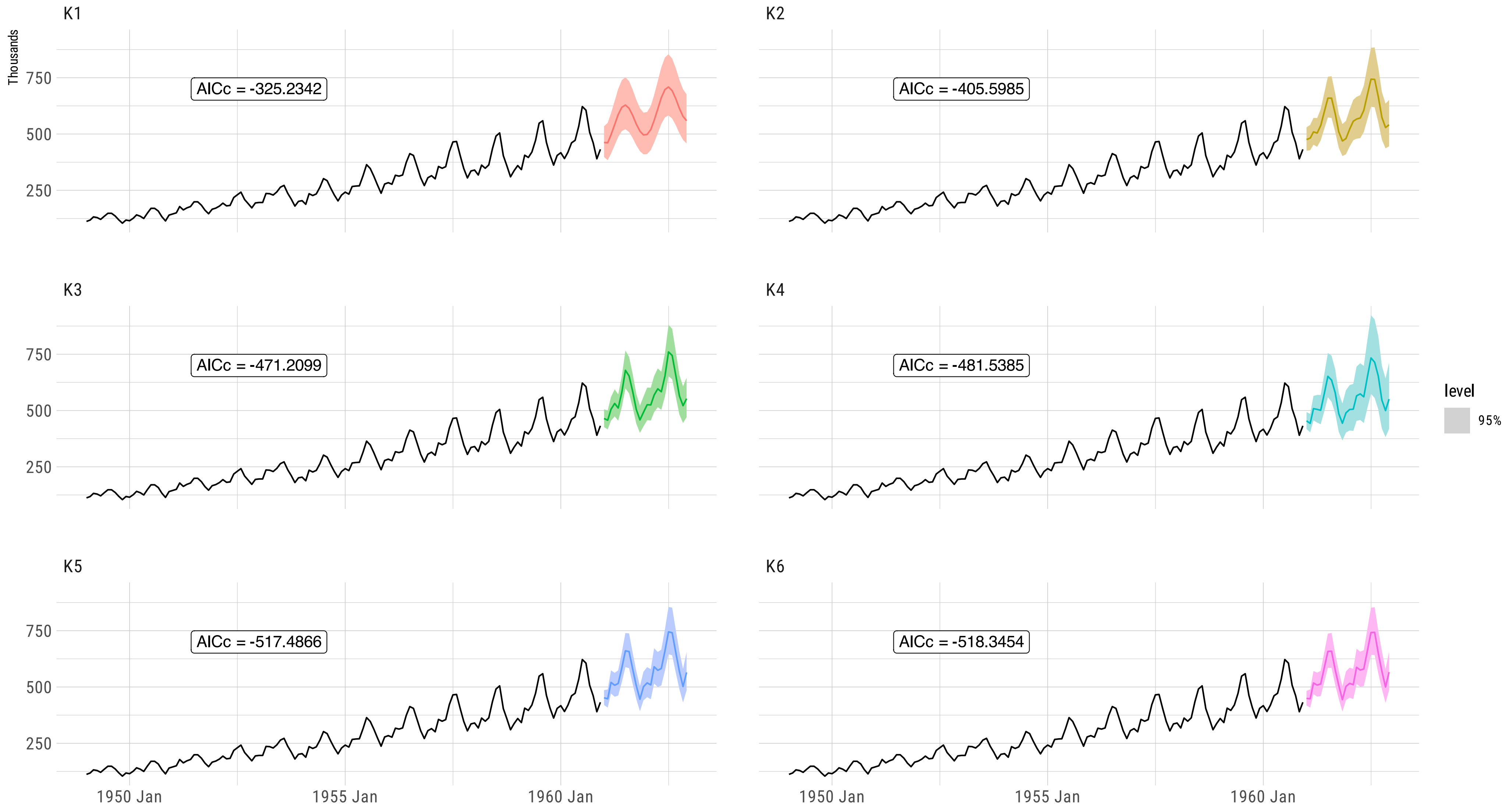
So far, we have not been successful in coming up with a *reliable* **regression forecast model** for the air passengers data set.

An *alternative* for modeling seasonality is using **harmonic regression with ARIMA errors**.

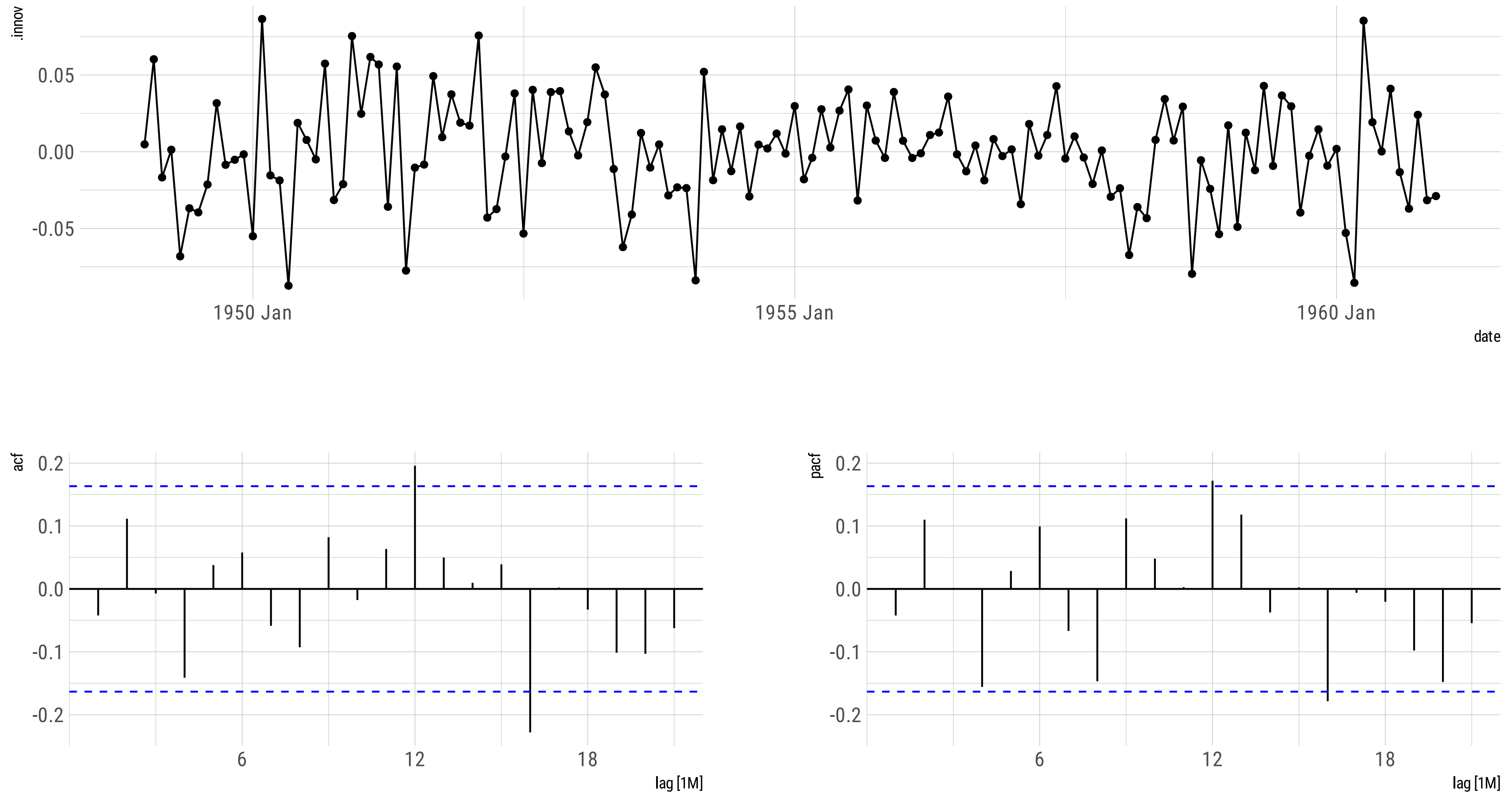
```
air_harmonic_fit ← model(air_ts,
              K1 = ARIMA(log(passengers) ~ fourier(K = 1) + PDQ(0, 0, 0)),
              K2 = ARIMA(log(passengers) ~ fourier(K = 2) + PDQ(0, 0, 0)),
              K3 = ARIMA(log(passengers) ~ fourier(K = 3) + PDQ(0, 0, 0)),
              K4 = ARIMA(log(passengers) ~ fourier(K = 4) + PDQ(0, 0, 0)),
              K5 = ARIMA(log(passengers) ~ fourier(K = 5) + PDQ(0, 0, 0)),
              K6 = ARIMA(log(passengers) ~ fourier(K = 6) + PDQ(0, 0, 0)))
```

Recall that K, the number of Fourier sine and cosine pairs, can vary from $K = 1$ up to $K = m/2$.

U.S. air passengers: 24-month ahead forecast



```
air_harmonic_fit ▷
  select(K6) ▷
  augment() ▷
  gg_tsdisplay(.innov, plot_type = "partial")
```



Harmonic regression with ARIMA errors

```
air_harmonic_fit ▷
  select(K6)
```

```
#> # A mable: 1 x 1
#>                               K6
#>                          <model>
#> 1 <LM w/ ARIMA(1,1,1) errors>
```

```
air_harmonic_fit ▷
  select(K5)
```

```
#> # A mable: 1 x 1
#>                               K5
#>                          <model>
#> 1 <LM w/ ARIMA(1,1,1) errors>
```

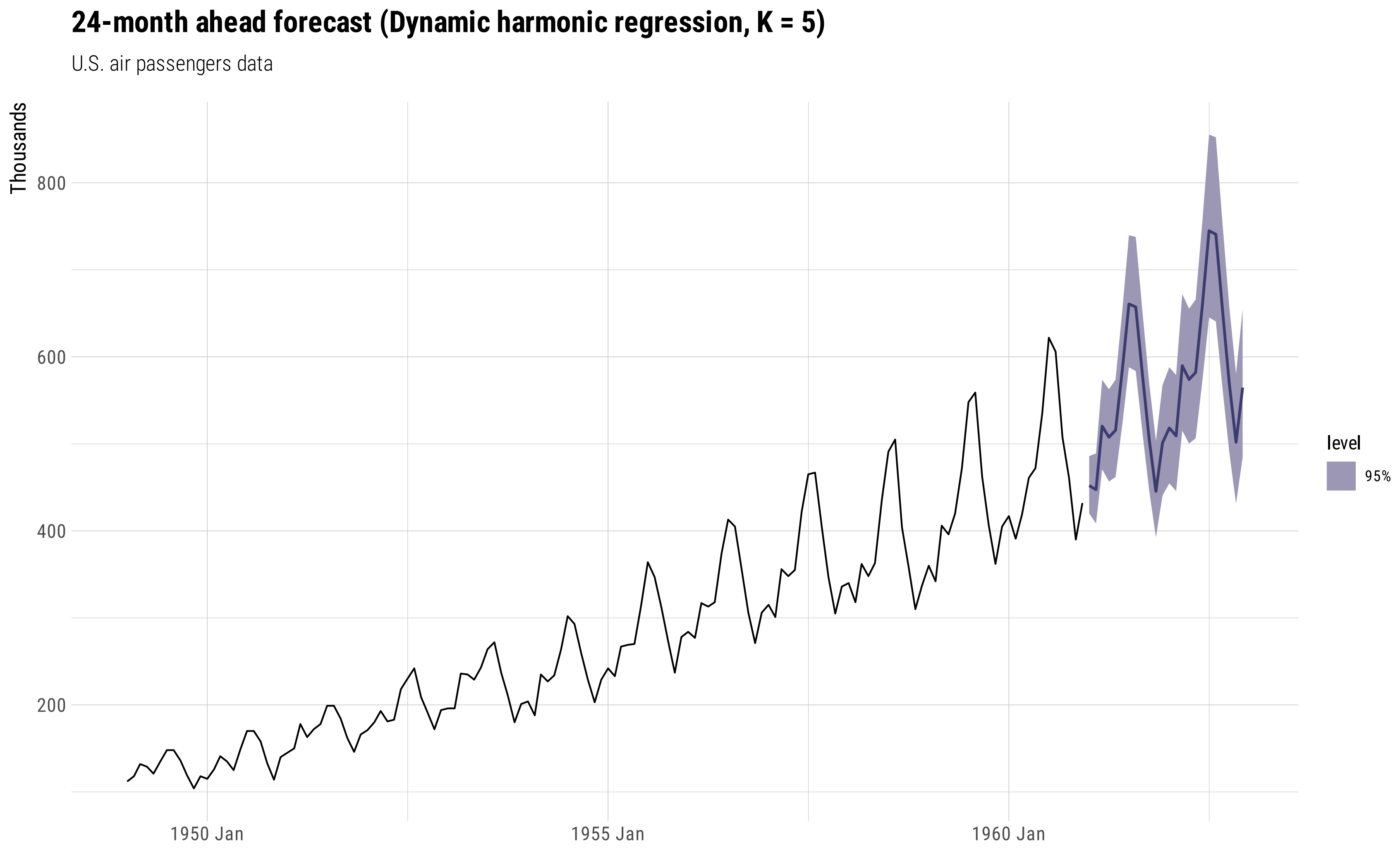
```
air_harmonic_fit ▷
  select(K6) ▷
  augment() ▷
  features(.innov, ljung_box, lag = 2 * 12, dof = 2)
```

```
#> # A tibble: 1 × 3
#>   .model lb_stat lb_pvalue
#>   <chr>    <dbl>     <dbl>
#> 1 K6        35.6    0.0335
```

```
air_harmonic_fit ▷
  select(K5) ▷
  augment() ▷
  features(.innov, ljung_box, lag = 2 * 12, dof = 2)
```

```
#> # A tibble: 1 × 3
#>   .model lb_stat lb_pvalue
#>   <chr>    <dbl>     <dbl>
#> 1 K5        33.9    0.0503
```

Harmonic regression with ARIMA errors



Scenario-based forecasting

Scenario-based forecasting

When using **regression** models for forecasting, the time series we are interested in forecasting is a **function** of one or more **predictor variables**.

Excluding variables we **do know** their future values (e.g., *trend* and *seasonal dummy* variables), we must **make assumptions** about other predictors for which we **do not have** information.

This allows us to assume different **scenarios** for the *predictor variables* that are of interest.

Let us come back to our **Phillips curve** example.

Scenario-based forecasting

```
phillips_arima ← phillips_ts ▷
  model(arima_reg = ARIMA(delta_infrate ~ unrate))
```

We will assume **3** possible **forecasting scenarios**:

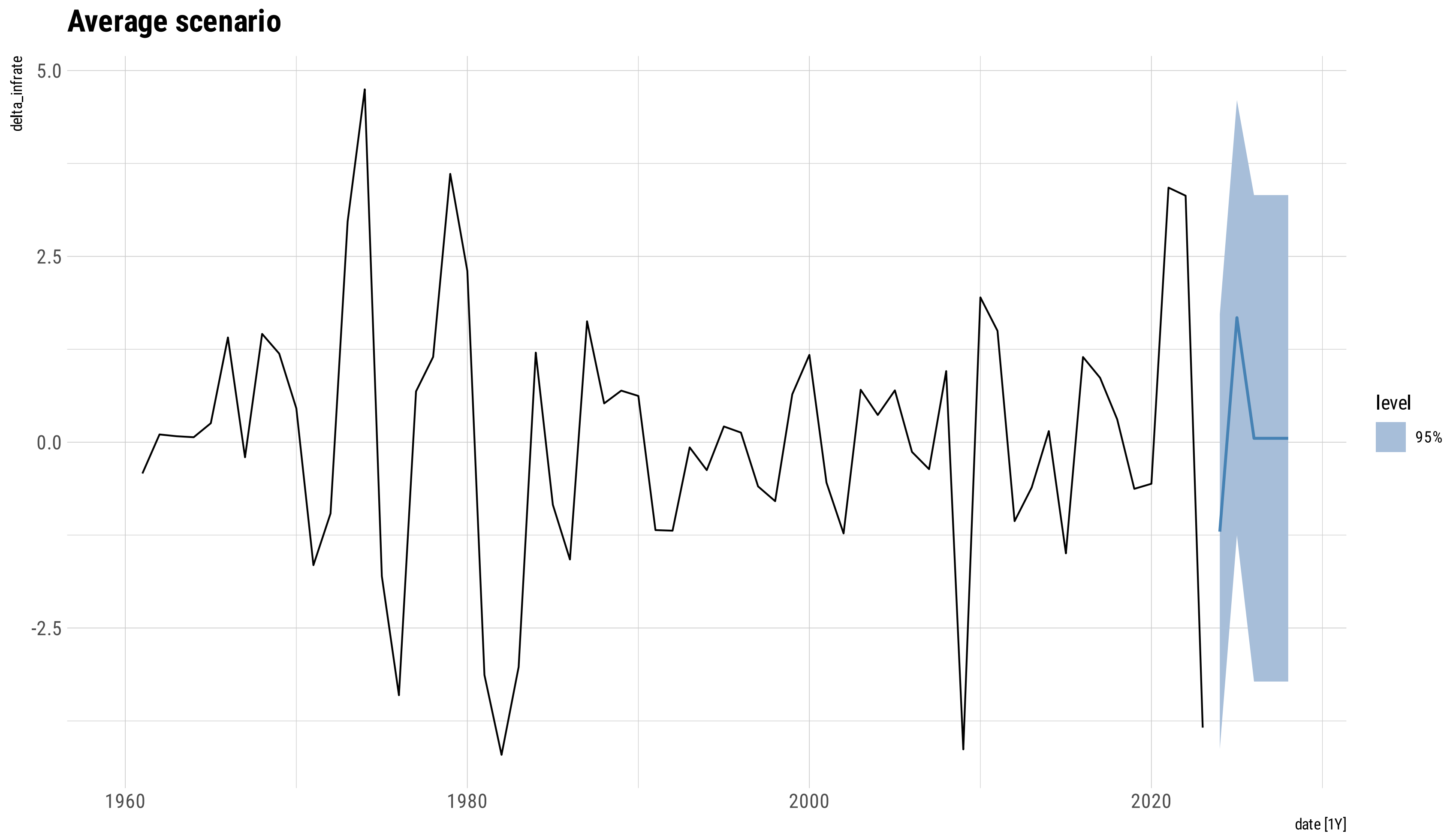
1. The unemployment rate follows its historical *average value* (5.9%);
2. The unemployment rate *increases* to 10%;
3. The unemployment rate follows its *current value* of 3.36%.

Scenario-based forecasting

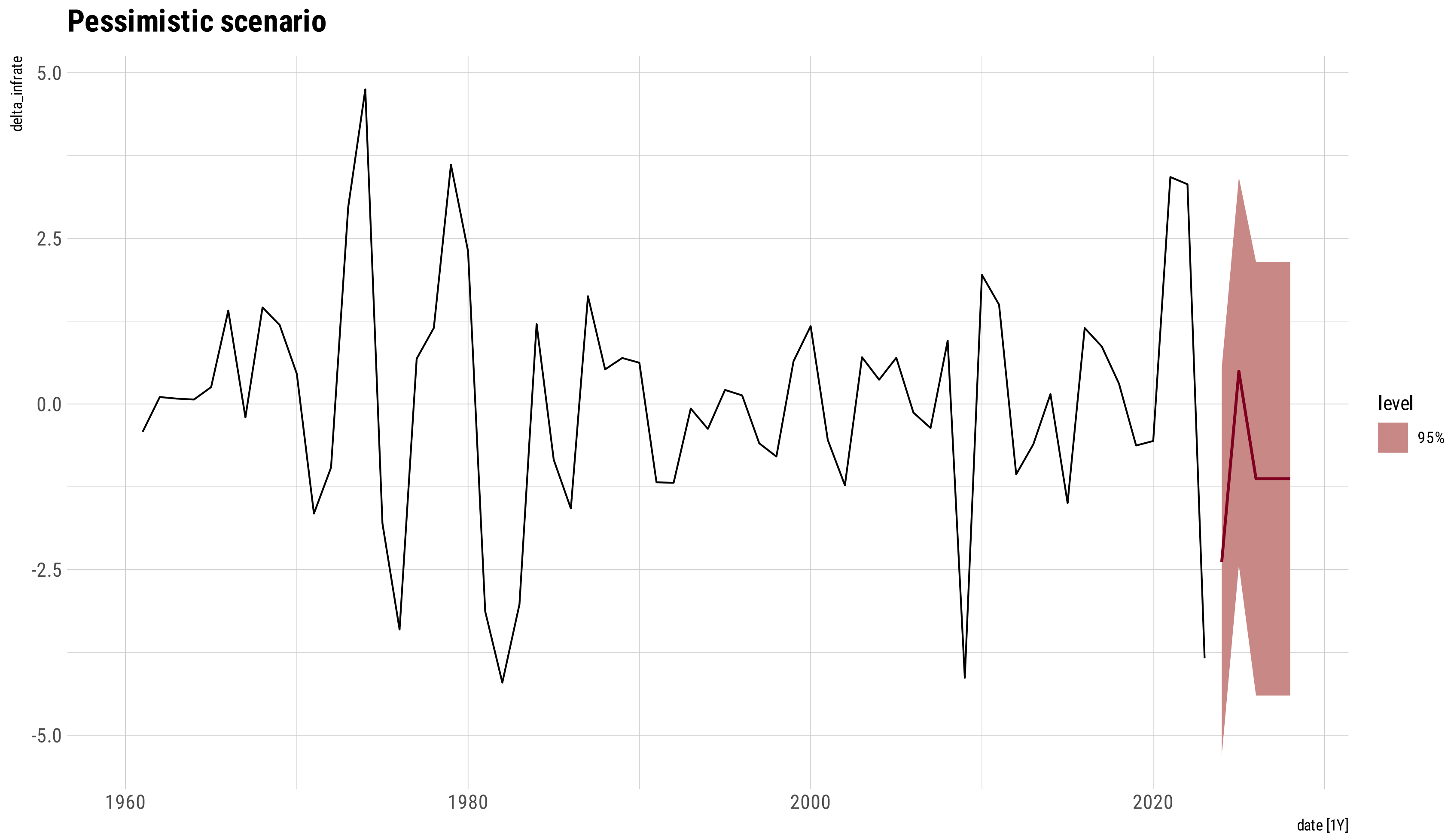
```
future_scenarios ← scenarios(average = new_data(phillips_ts, n = 5) ▷
                                mutate(unrate = mean(phillips_ts$unrate)),
                              pessimistic = new_data(phillips_ts, n = 5) ▷
                                mutate(unrate = 10),
                              optimistic = new_data(phillips_ts, n = 5) ▷
                                mutate(unrate = 3.36))
```

```
phillips_scen_fc ← phillips_arima ▷
  forecast(h = 5, new_data = future_scenarios)
```

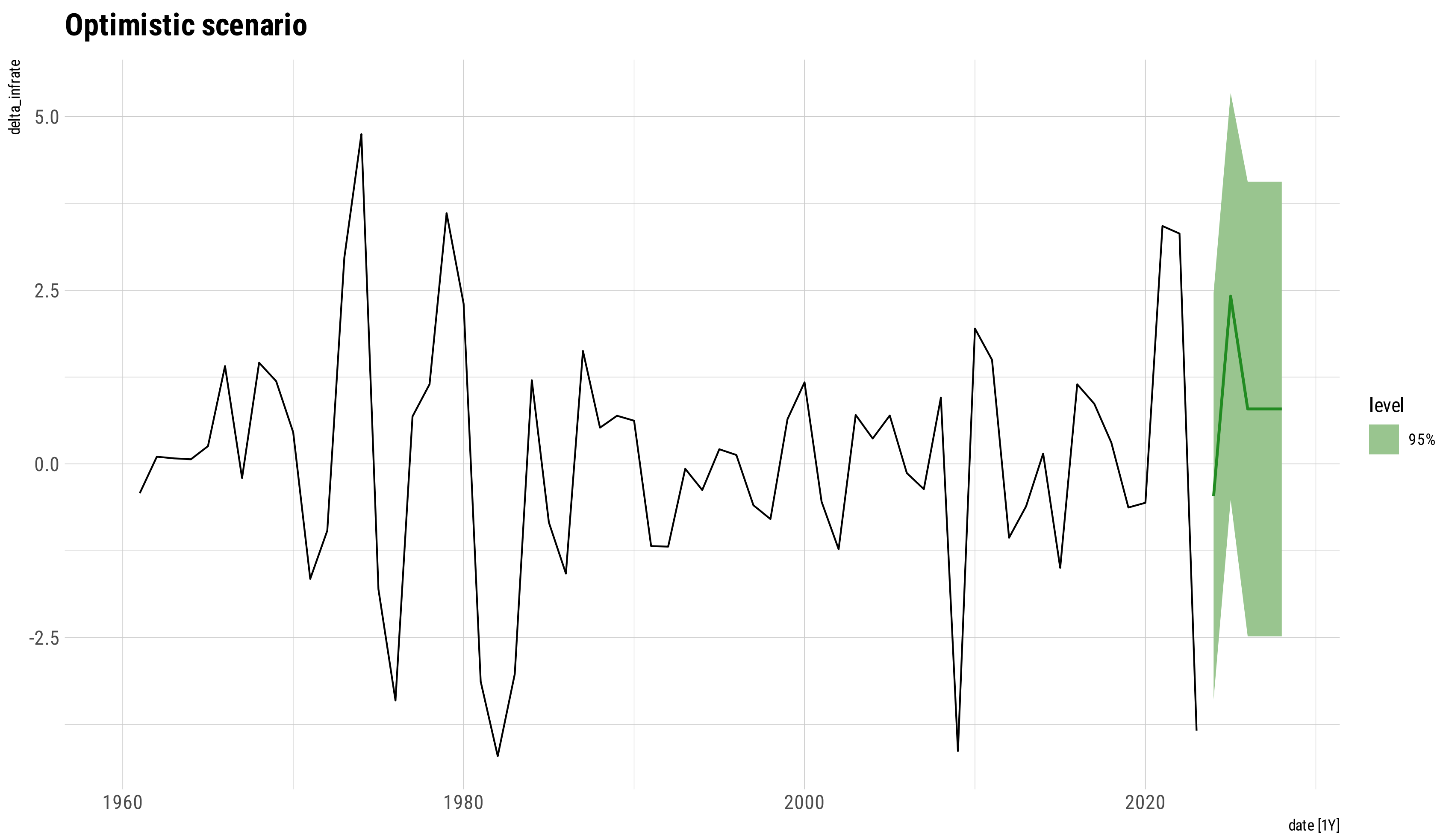
```r
phillips_ts ▷
  autoplot(delta_infrate) +
  autolayer(phillips_scen_fc ▷ filter(.scenario == "average"), level = 95, color = "#4682b4", linewidth = .8) +
  labs(title = "Average scenario")
```



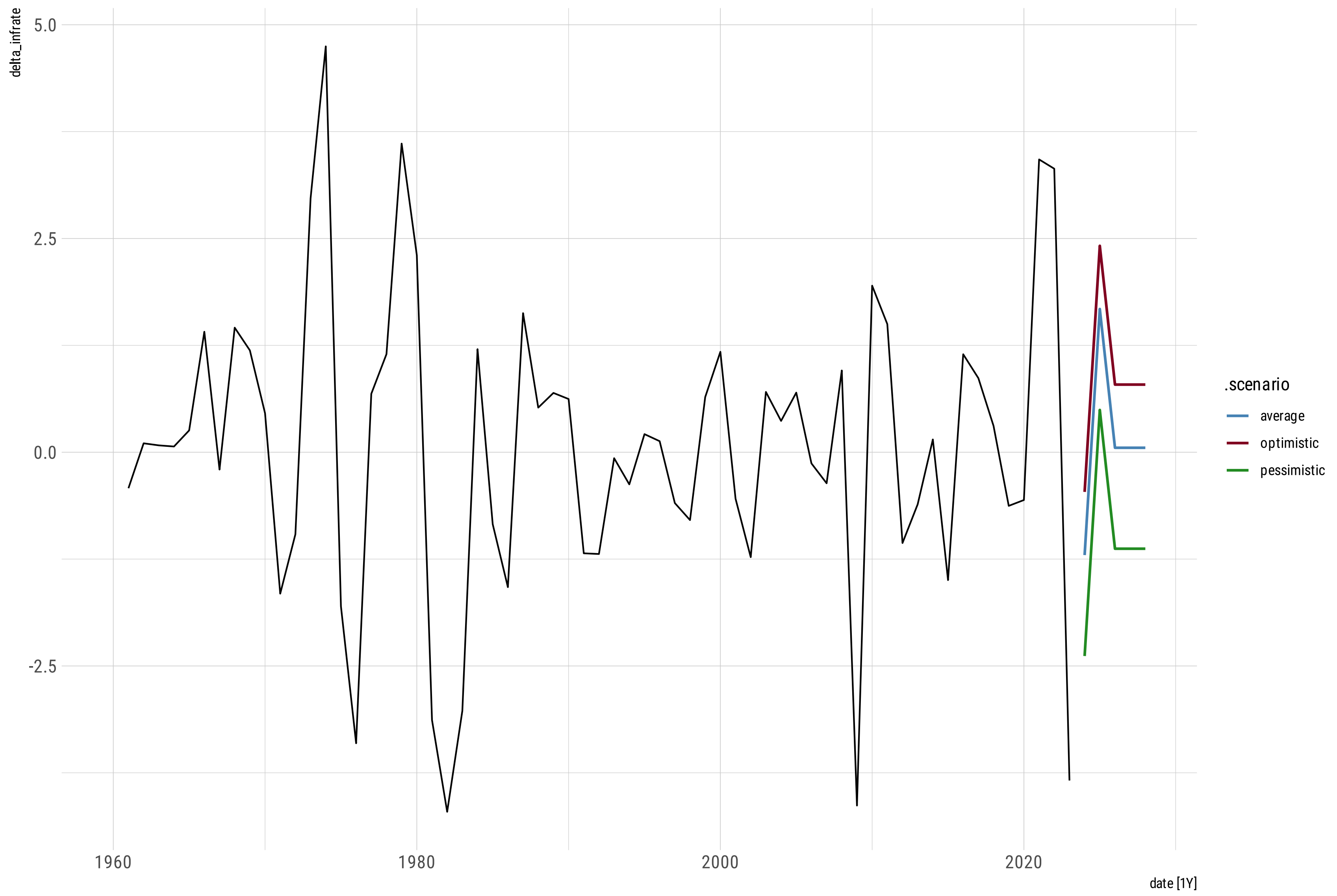
```r
phillips_ts |>
  autoplot(delta_infrate) +
  autolayer(phillips_scen_fc |> filter(.scenario == "pessimistic"), level = 95, color = "#800020", linewidth = .8) +
  labs(title = "Pessimistic scenario")
```



```r
phillips_ts |>
  autoplot(delta_infrate) +
  autolayer(phillips_scen_fc |> filter(.scenario == "optimistic"), level = 95, color = "#228b22", linewidth = .8) +
  labs(title = "Optimistic scenario")
```



All 3 scenarios



Next time: Applications