ARIMA models: Modeling and forecasting

EC 361–001

Prof. Santetti
Spring 2024

Materials

Required readings:

- `Hyndman & Athanasopoulos, ch. 9`
    - sections 9.7—9.8.

Motivation

Motivation

Last time, we have joined the *AR*, *I*, and *MA* portions of **ARIMA** models.

When deciding on the **order** of our ARIMA models, a great starting point is to look at the *autocorrelation coefficient function* (**ACF**) and *partial autocorrelation function* (**PACF**) plots.

However, these can only do **so much** when deciding on which model to choose and perform our forecasts.

Therefore, we must move on to more **robust** procedures.

ARIMA at work

ARIMA at work

The `{fable}` R package handles ARIMA estimation following the **Hyndman-Khandakar algorithm**:

Hyndman–Khandakar algorithm for automatic ARIMA modelling

1. The number of differences $0 \le d \le 2$ is determined using repeated KPSS tests.
2. The values of p and q are then chosen by minimising the AICc after differencing the data d times. Rather than considering every possible combination of p and q, the algorithm uses a stepwise search to traverse the model space.

    a. Four initial models are fitted:
      - $\text{ARIMA}(0, d, 0)$,
      - $\text{ARIMA}(2, d, 2)$,
      - $\text{ARIMA}(1, d, 0)$,
      - $\text{ARIMA}(0, d, 1)$.

      A constant is included unless $d = 2$. If $d \le 1$, an additional model is also fitted:
      - $\text{ARIMA}(0, d, 0)$ without a constant.

    b. The best model (with the smallest AICc value) fitted in step (a) is set to be the "current model".

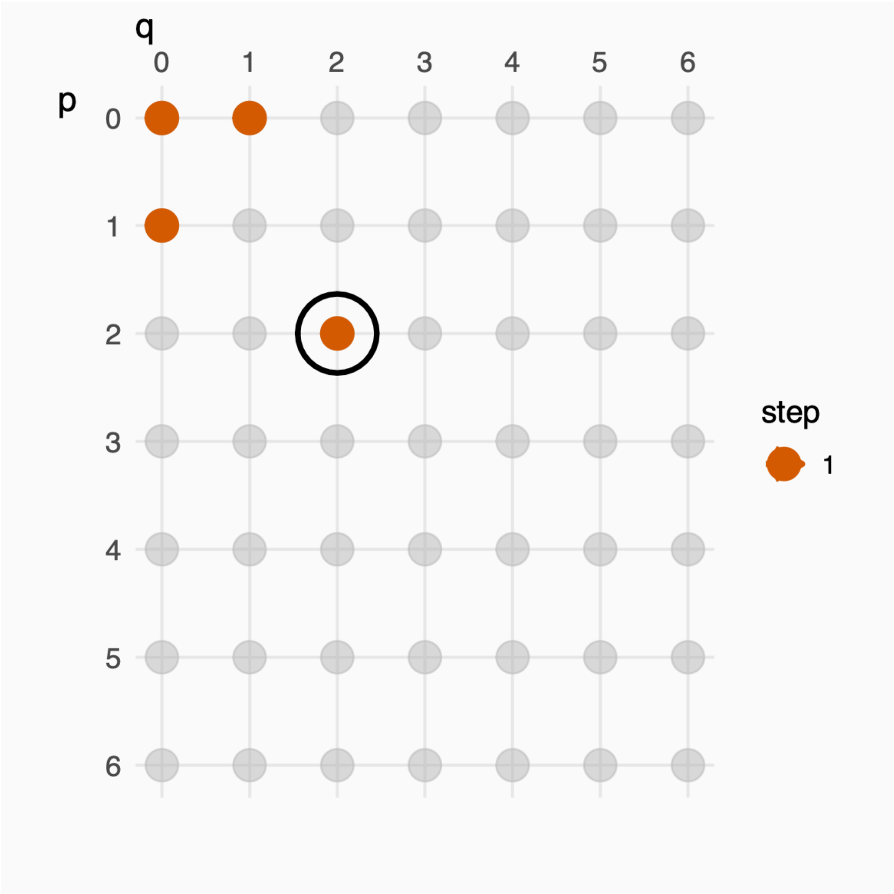
    c. Variations on the current model are considered:
      - vary p and/or q from the current model by ± 1;
      - include/exclude c from the current model.

      The best model considered so far (either the current model or one of these variations) becomes the new current model.

    d. Repeat Step 2(c) until no lower AICc can be found.

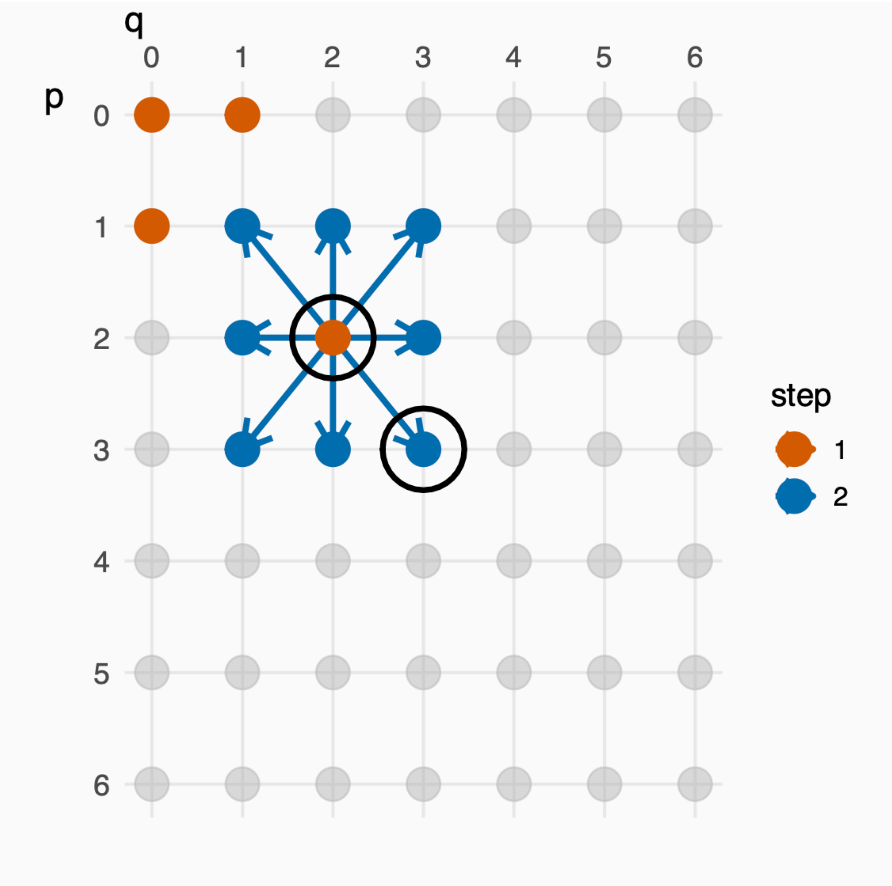
ARIMA at work

The **stepwise** procedure:



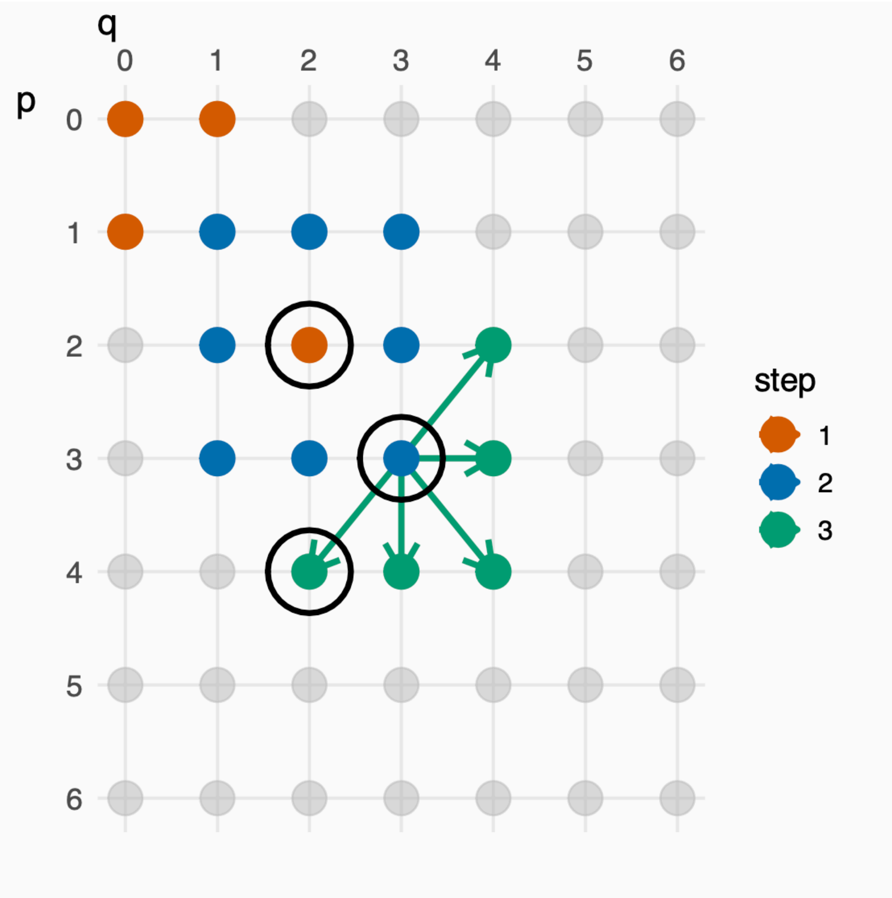
ARIMA at work

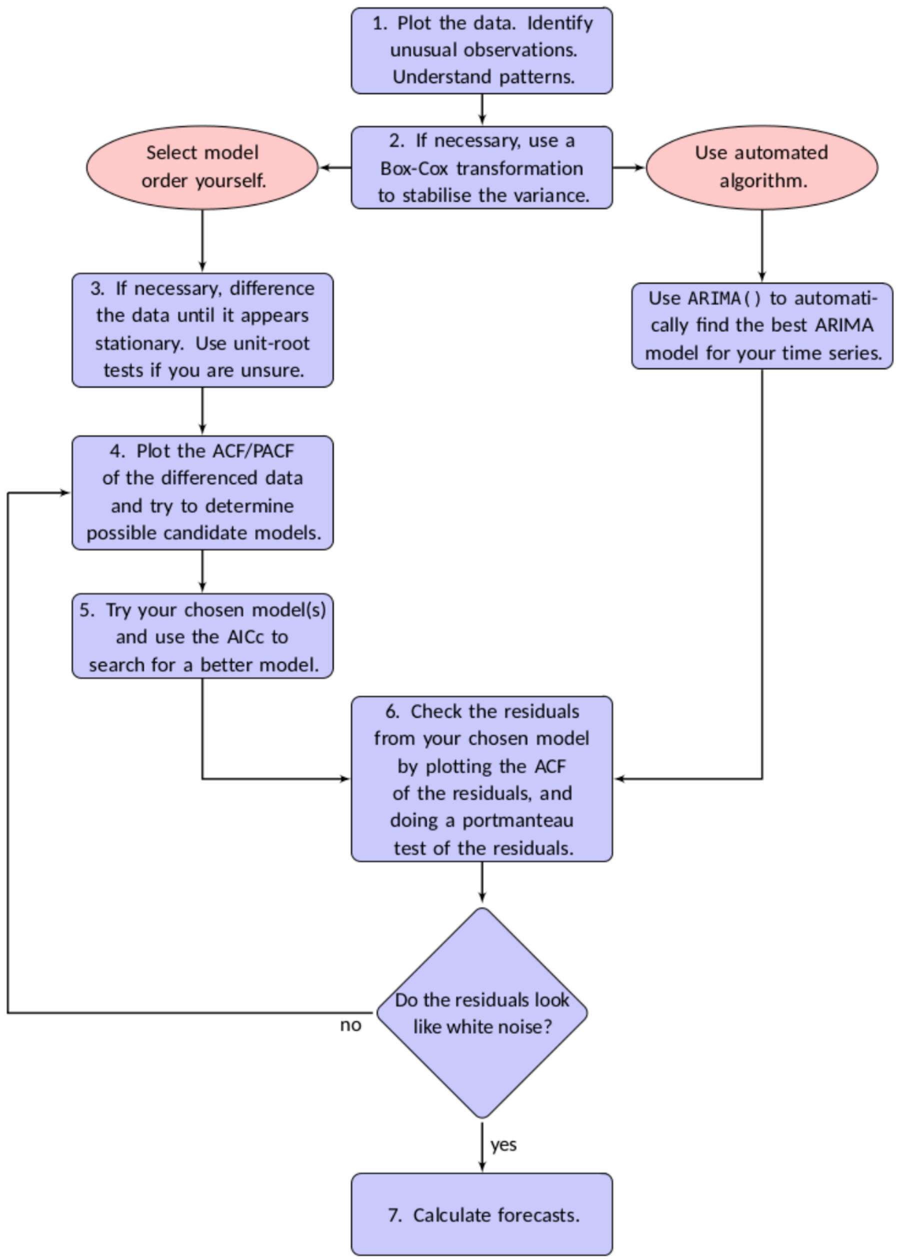
The **stepwise** procedure:



ARIMA at work

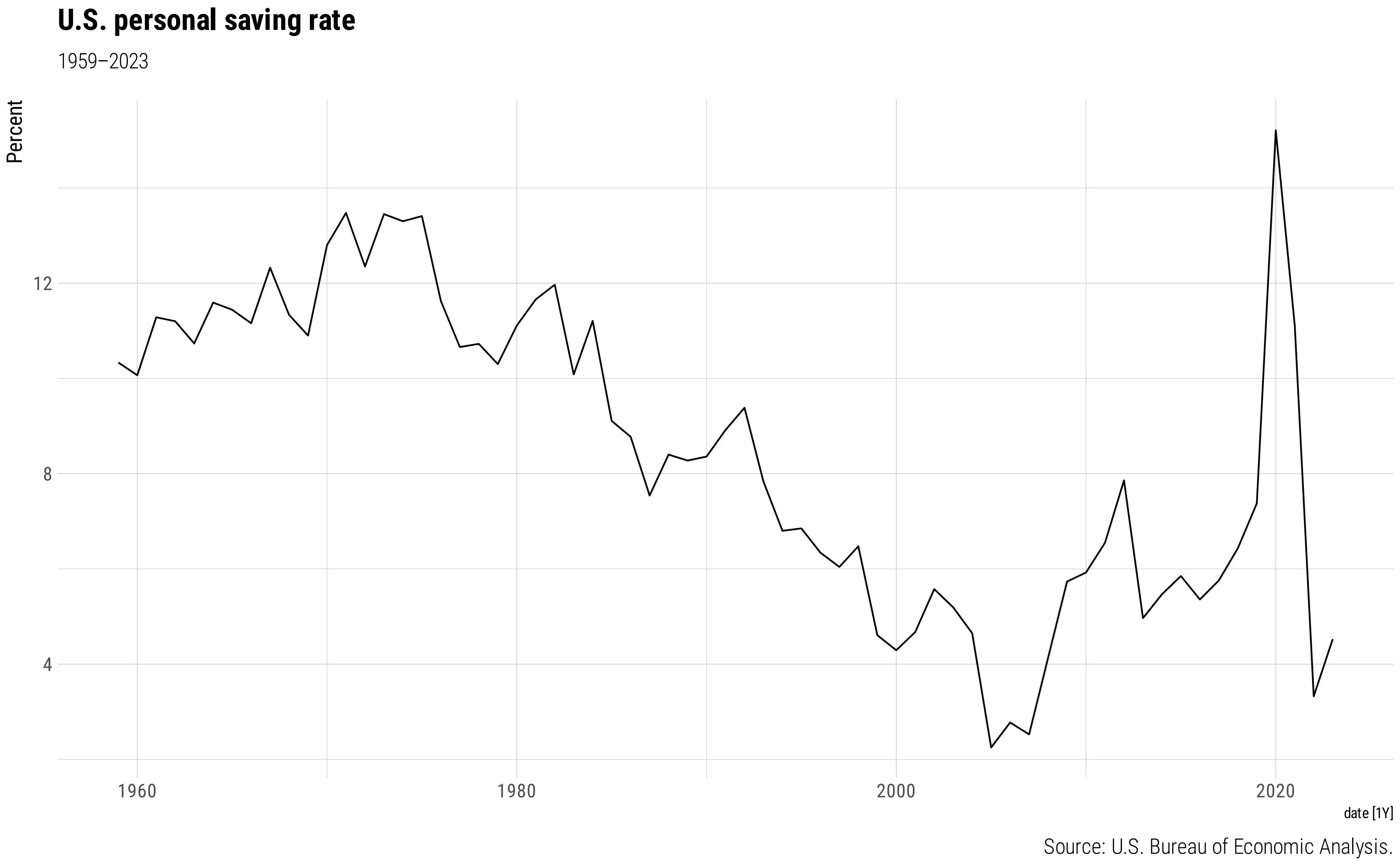
The **stepwise** procedure:




1. Plot the data. Identify unusual observations. Understand patterns.
2. If necessary, use a Box-Cox transformation to stabilise the variance.
Select model order yourself.
Use automated algorithm.
3. If necessary, difference the data until it appears stationary. Use unit-root tests if you are unsure.
Use ARIMA() to automatically find the best ARIMA model for your time series.
4. Plot the ACF/PACF of the differenced data and try to determine possible candidate models.
5. Try your chosen model(s) and use the AICc to search for a better model.
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals.
Do the residuals look like white noise?
no
yes
7. Calculate forecasts.

An example

An example



U.S. personal saving rate
1959–2023
Percent
12
8
4
1960
1980
2000
2020
date [1Y]
Source: U.S. Bureau of Economic Analysis.

An example

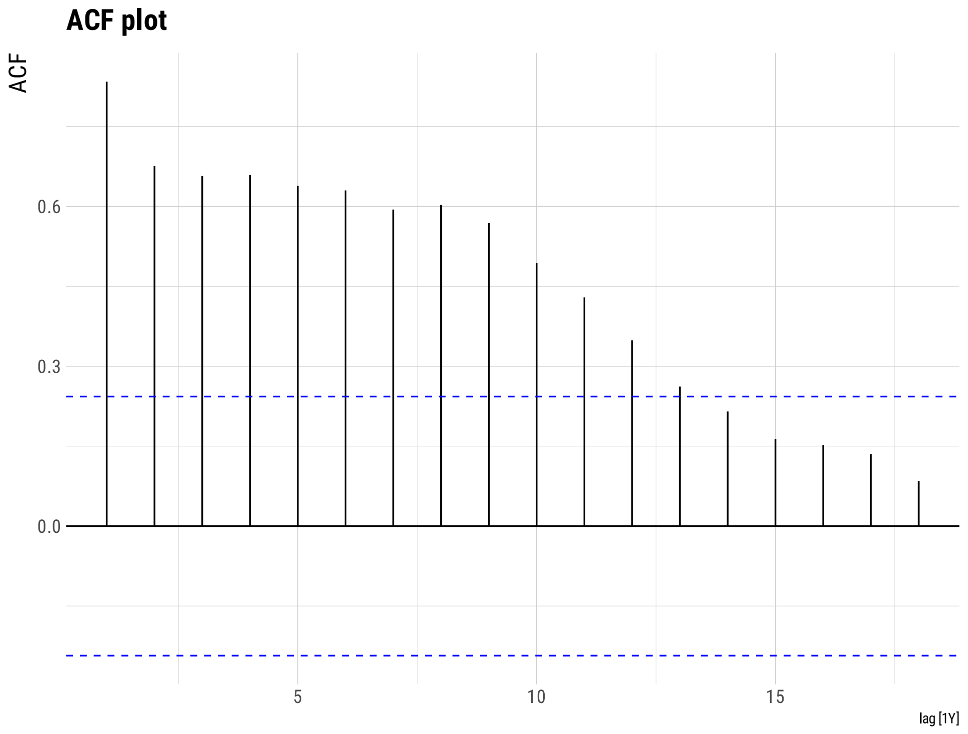
```
saving_ts |>
  features(psav, unitroot_kpss)
```

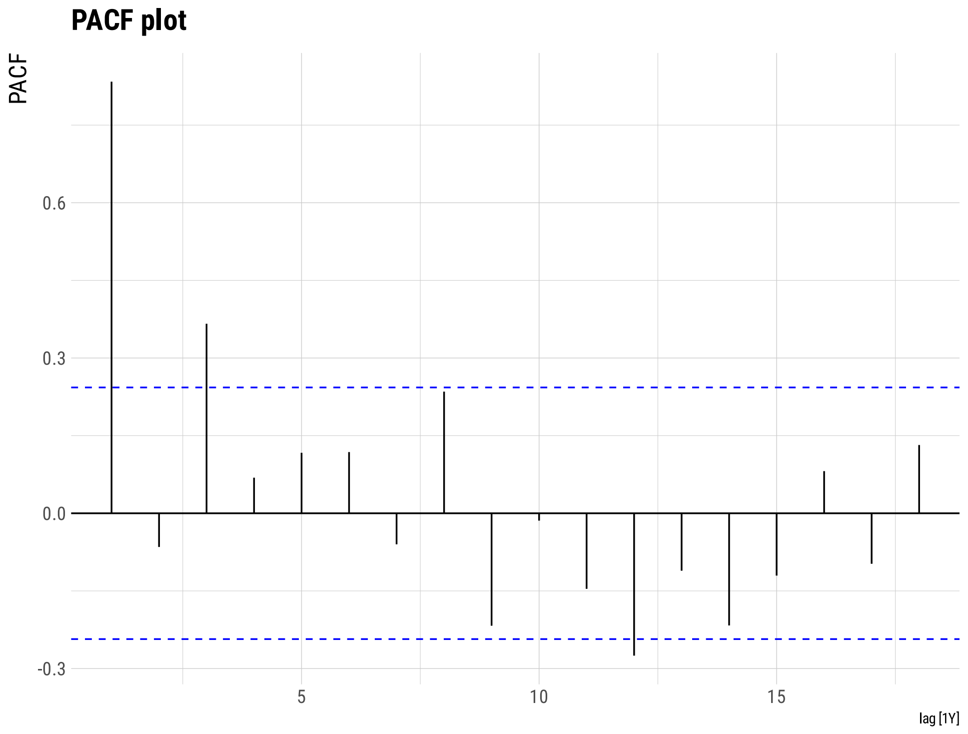
```
#> # A tibble: 1 × 2
#>   kpss_stat kpss_pvalue
#>       <dbl>       <dbl>
#> 1      1.22        0.01
```

```
saving_ts |>
  features(difference(psav), unitroot_kpss)
```

```
#> # A tibble: 1 × 2
#>   kpss_stat kpss_pvalue
#>       <dbl>       <dbl>
#> 1    0.0537         0.1
```

An example


ACF plot
ACF
0.6
0.3
0.0
5
10
15
lag [1Y]


PACF plot
PACF
0.6
0.3
0.0
-0.3
5
10
15
lag [1Y]

An example

```
saving_arima_fit ← saving_ts ▷
  model(arima310 = ARIMA(psav ~ 1 + pdq(3, 1, 0)), # "1" includes a constant (c).
        arima113 = ARIMA(psav ~ 1 + pdq(1, 1, 3)), # "1" includes a constant (c).
        arima_auto = ARIMA(psav)) # letting {fable} select the best model.

saving_arima_fit
```

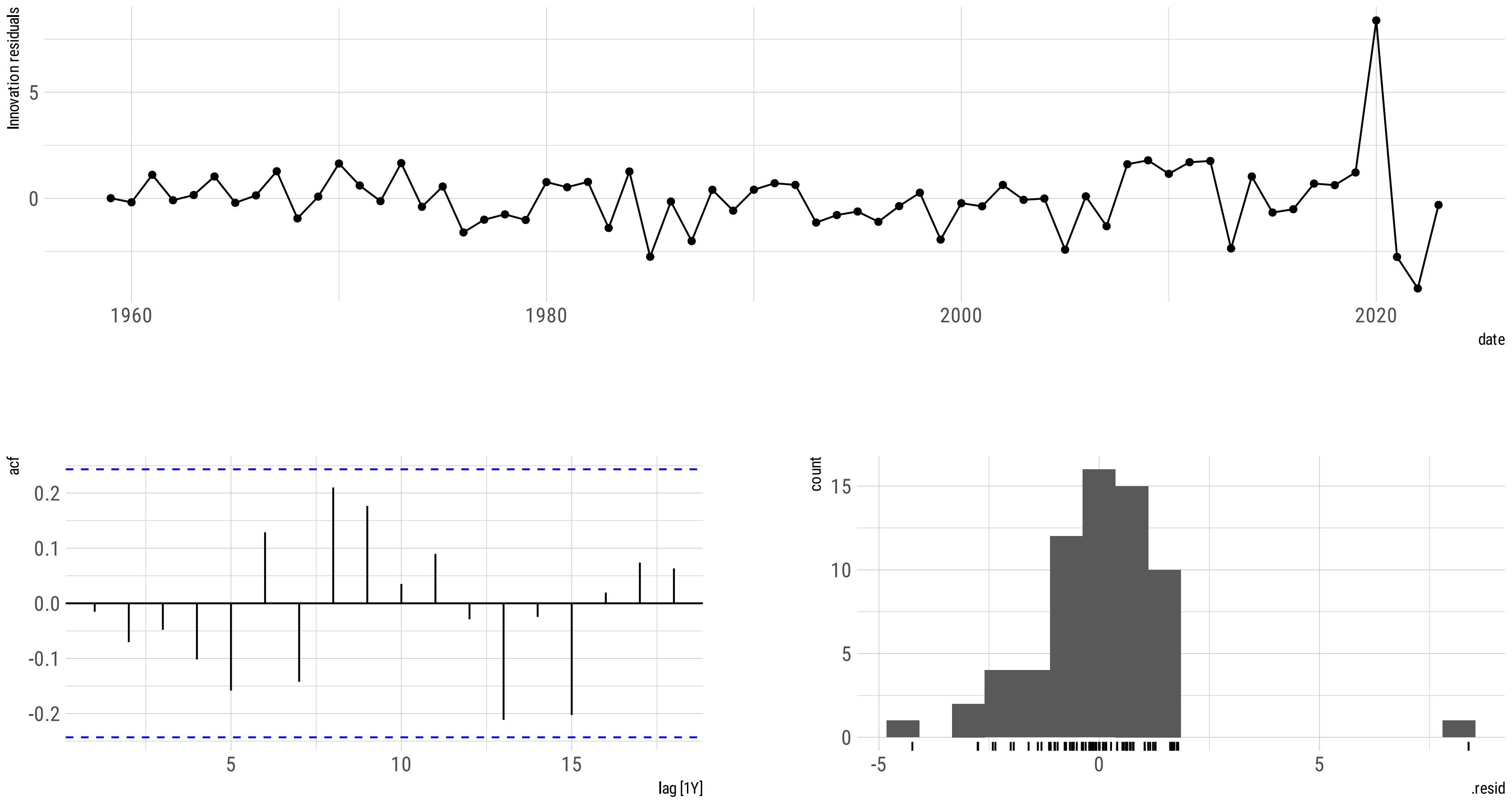
```
#> # A mable: 1 x 3
#>                  arima310                  arima113    arima_auto
#>                   <model>                   <model>       <model>
#> 1 <ARIMA(3,1,0) w/ drift> <ARIMA(1,1,3) w/ drift> <ARIMA(0,1,2)>
```

An example

```
saving_arima_fit ▷
  glance() ▷
  arrange(AICc) ▷
  select(.model, AIC, AICc)
```

```
#> # A tibble: 3 × 3
#>   .model        AIC  AICc
#>   <chr>       <dbl> <dbl>
#> 1 arima_auto   250.  250.
#> 2 arima113     249.  250.
#> 3 arima310     254.  255.
```

```r
saving_arima_fit |>
  select(arima310) |>
  gg_tsresiduals()
```

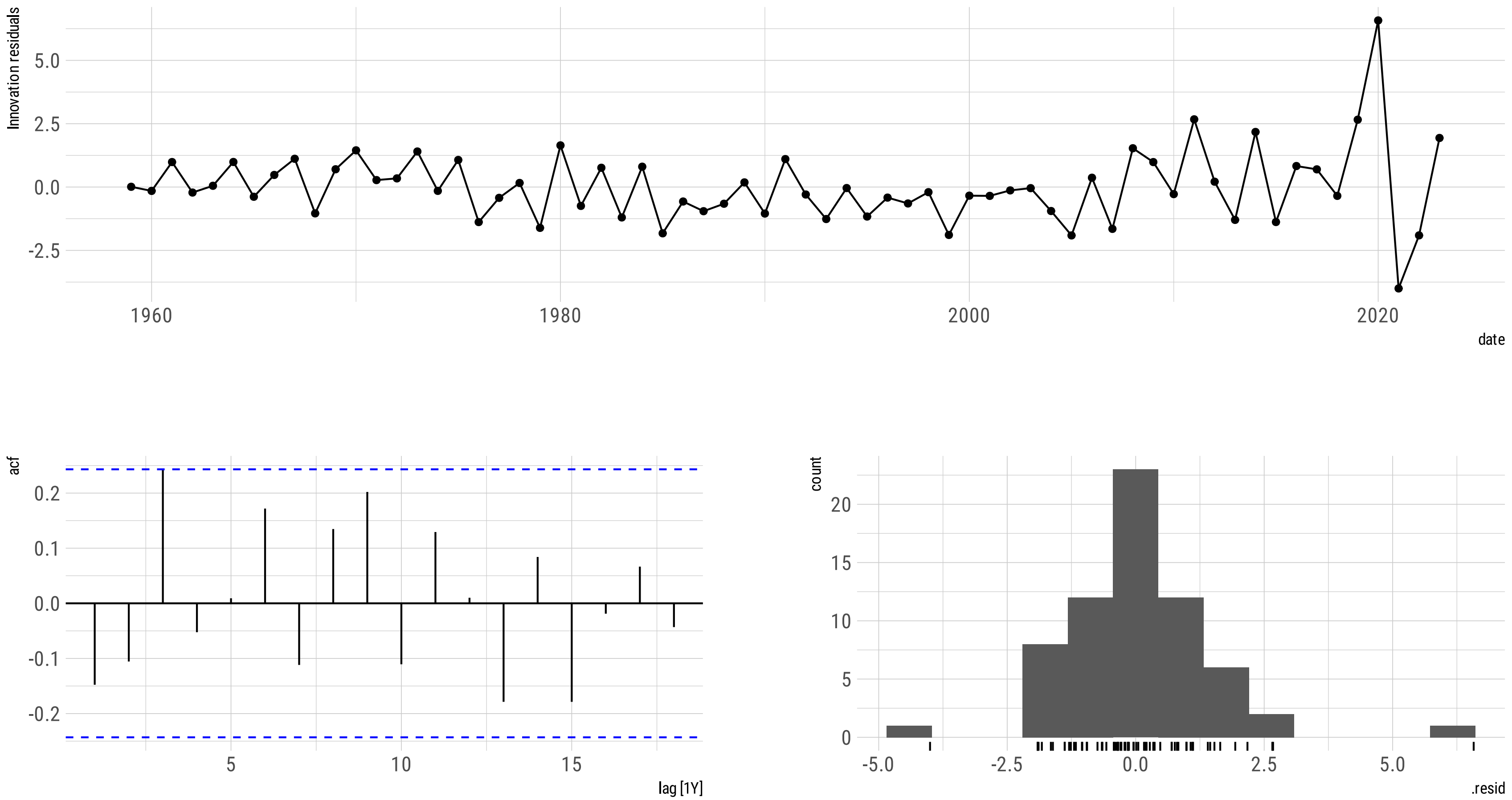


An example

```
saving_arima_fit |>
  augment() |>
  filter(.model == "arima310") |>
  features(.innov, ljung_box, lag = 10, dof = 3)
```

```
#> # A tibble: 1 × 3
#>   .model   lb_stat lb_pvalue
#>   <chr>      <dbl>     <dbl>
#> 1 arima310    11.7     0.110
```

```r
saving_arima_fit ▷
  select(arima113) ▷
  gg_tsresiduals()
```

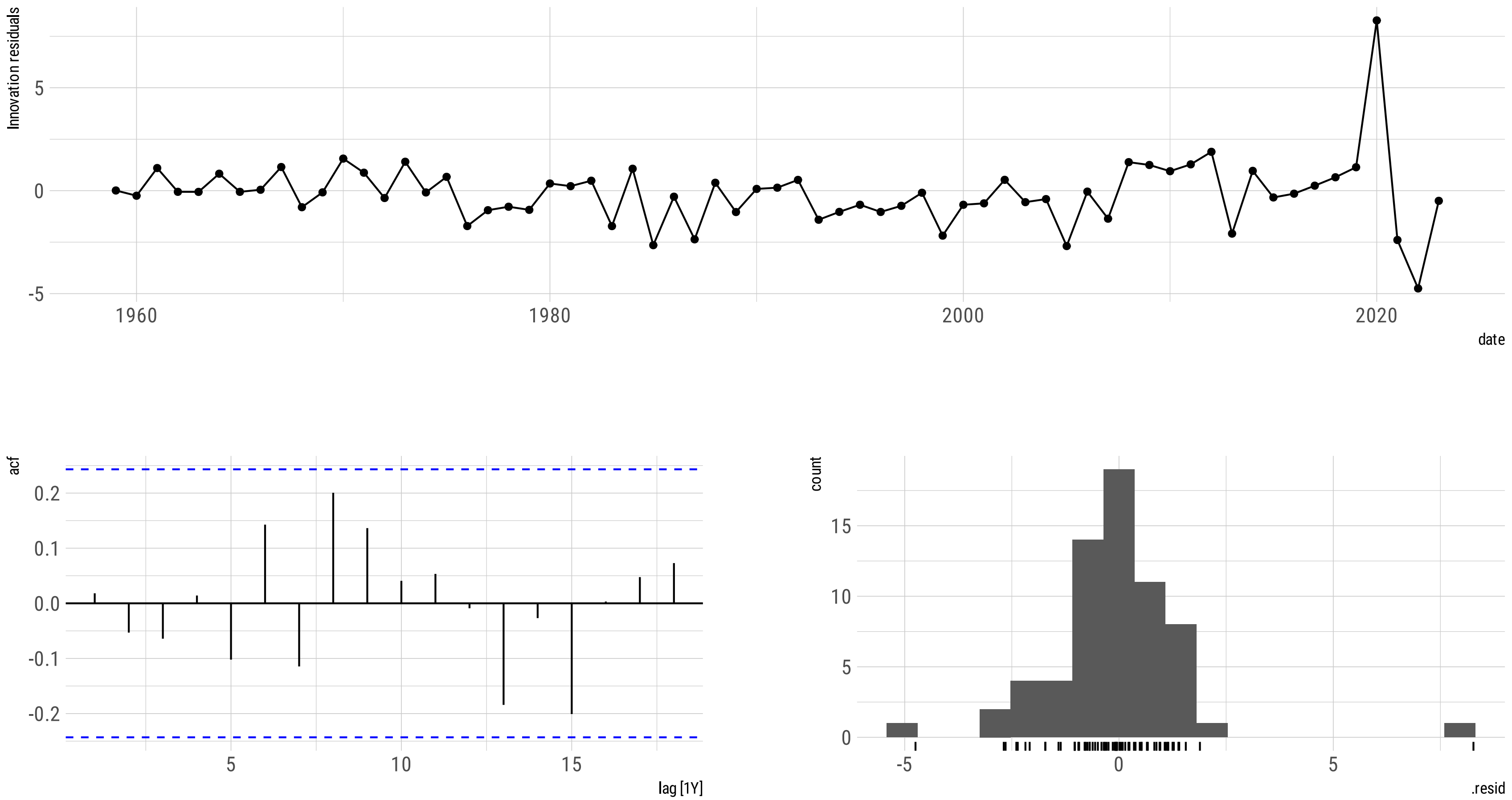


An example

```
saving_arima_fit ▷
  augment() ▷
  filter(.model == "arima113") ▷
  features(.innov, ljung_box, lag = 10, dof = 4)
```

```
#> # A tibble: 1 × 3
#>   .model   lb_stat lb_pvalue
#>   <chr>      <dbl>     <dbl>
#> 1 arima113    15.2    0.0186
```

```
saving_arima_fit ▷
  select(arima_auto) ▷
  gg_tsresiduals()
```



An example

```
saving_arima_fit ▷
  augment() ▷
  filter(.model == "arima_auto") ▷
  features(.innov, ljung_box, lag = 10, dof = 2)
```

```
#> # A tibble: 1 × 3
#>   .model     lb_stat lb_pvalue
#>   <chr>        <dbl>     <dbl>
#> 1 arima_auto    8.42     0.393
```

Forecasting with ARIMA

Forecasting with ARIMA

Time to explain **how** ARIMA forecasts are generated.

Suppose our model of choice is an **ARMA(2, 2)**.

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \varepsilon_t$$

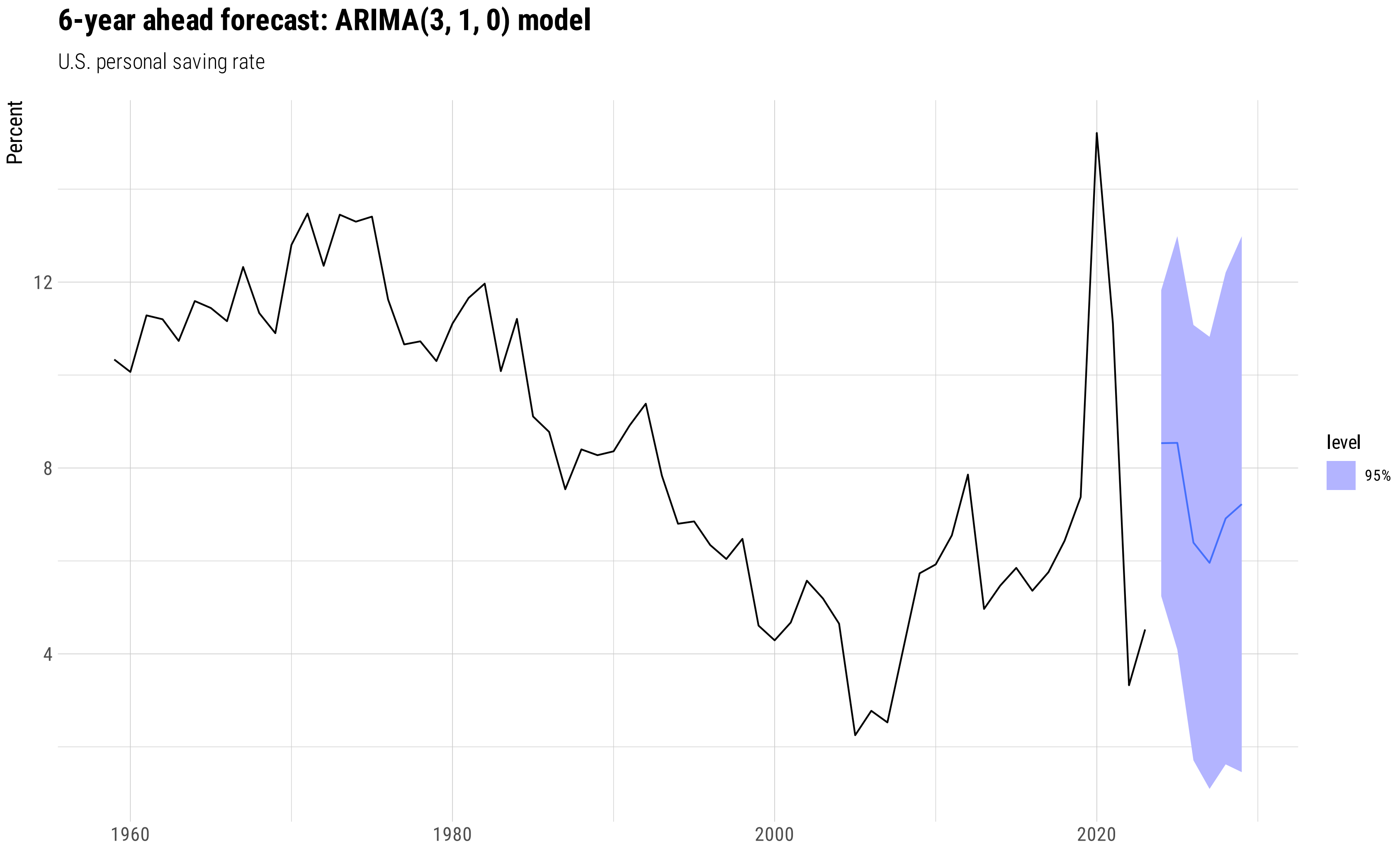
Let us derive the **point forecasts** for $T + 1$ and $T + 2$:

Forecasting with ARIMA

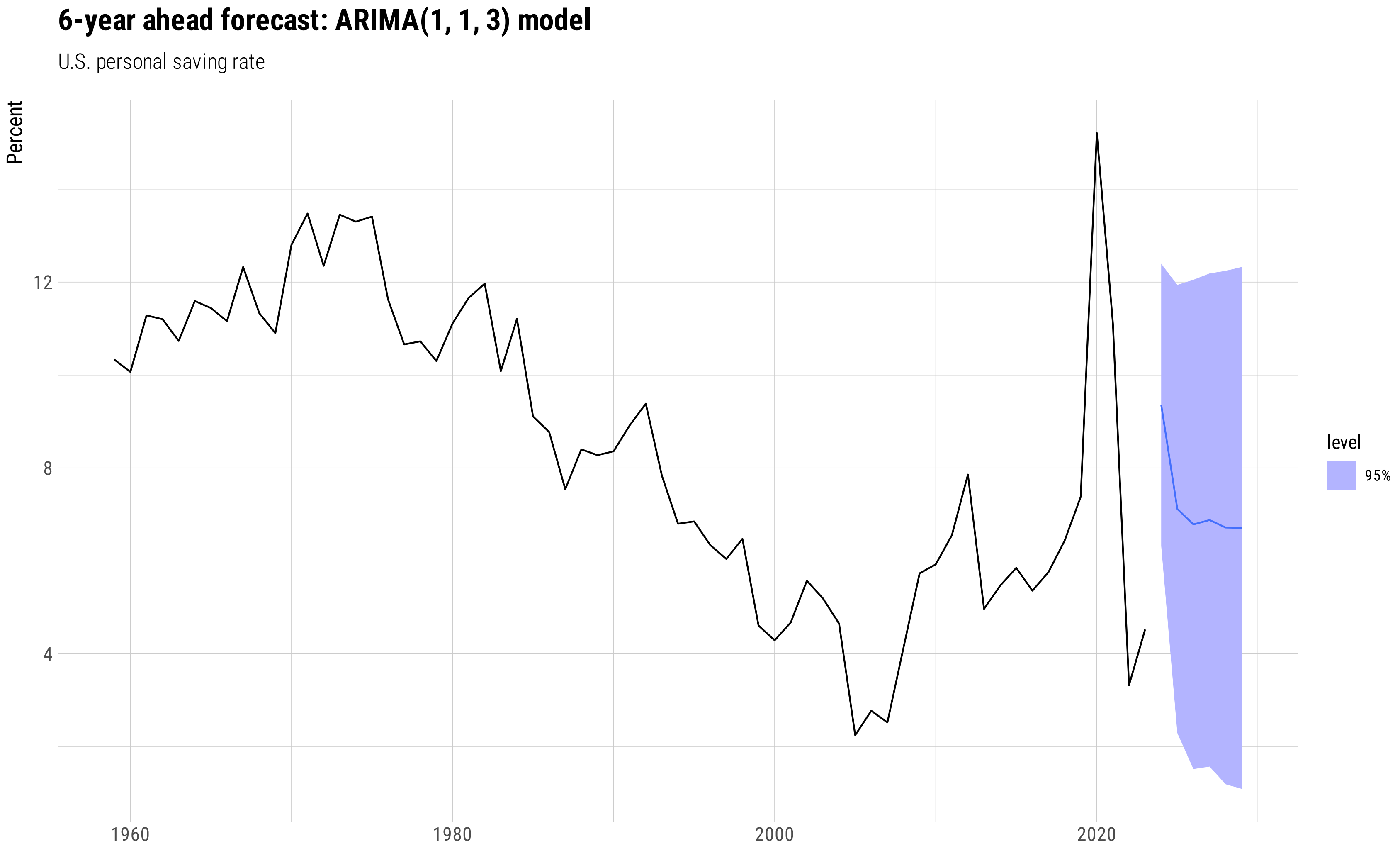
Coming back to our example...

```
saving_arima_fc <- saving_arima_fit |>
  forecast(h = 6)
```

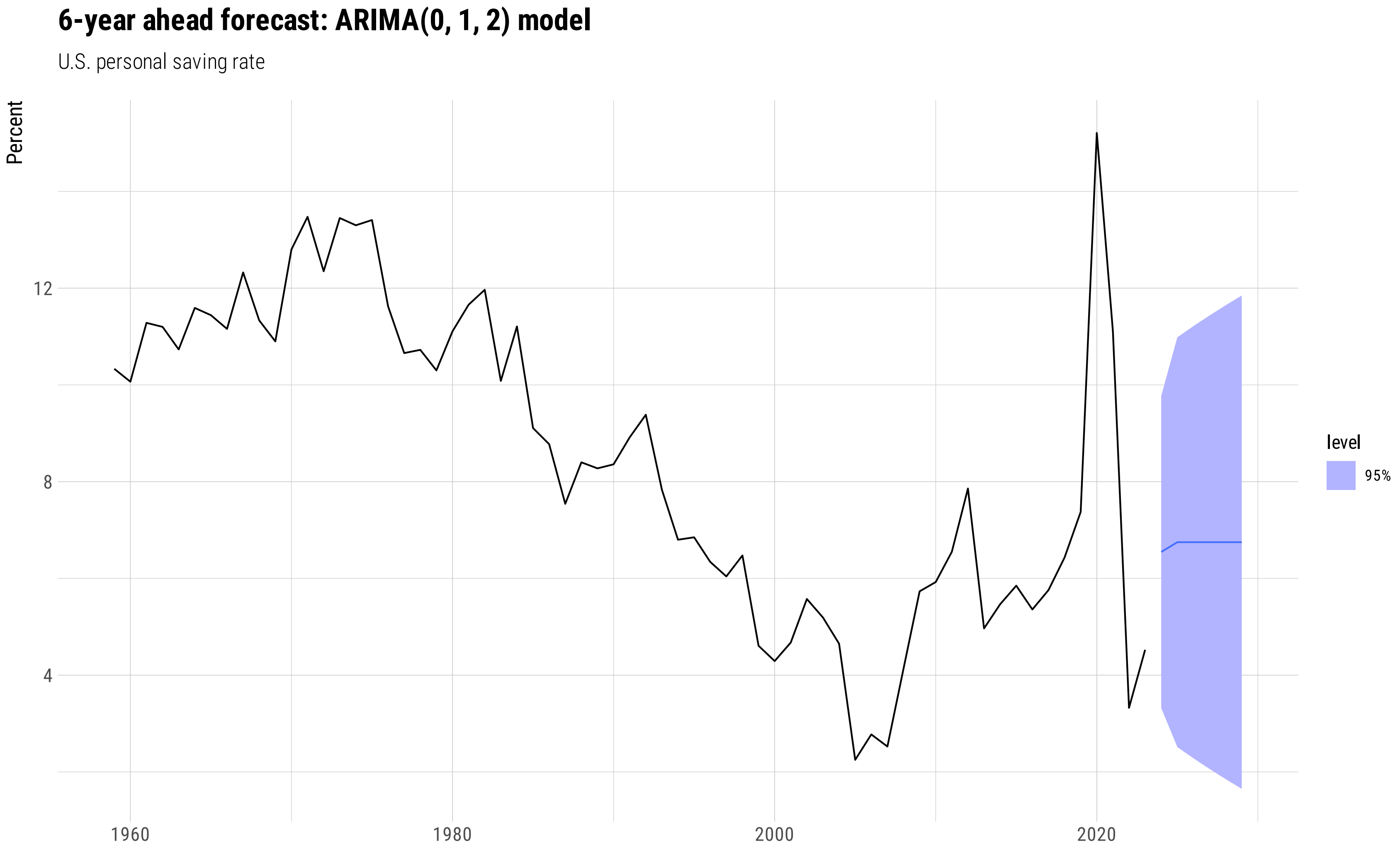
Forecasting with ARIMA



Forecasting with ARIMA



Forecasting with ARIMA



Next time: Seasonal ARIMA models