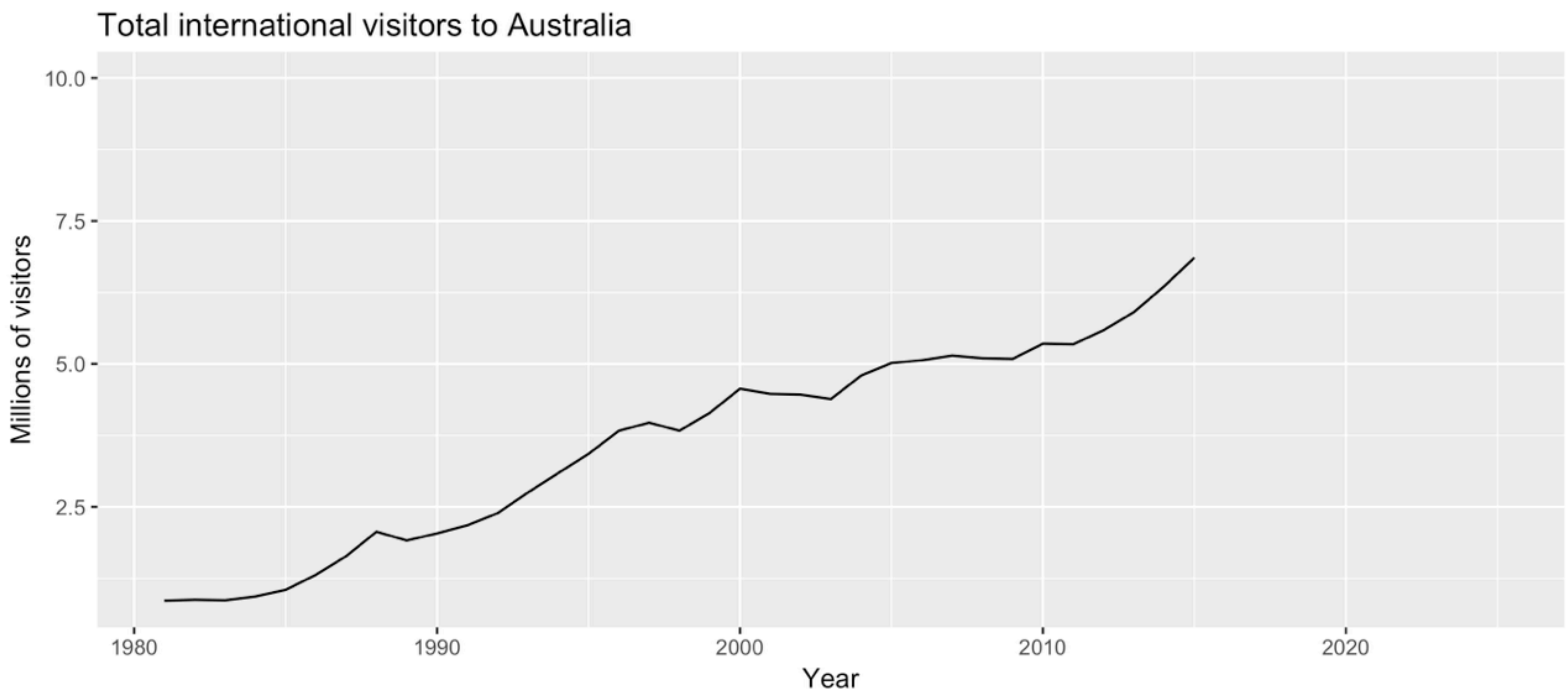
FORECASTING USING R

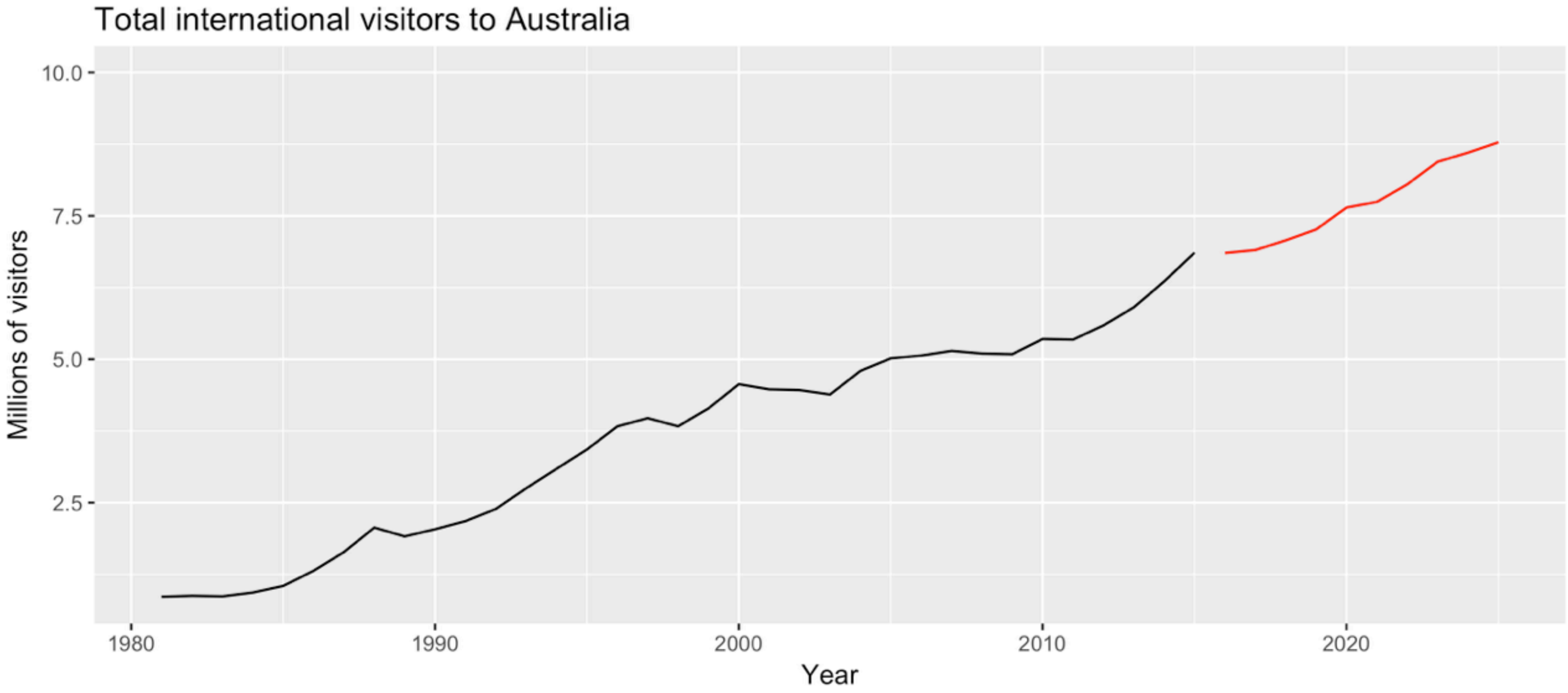
Forecasts and potential futures

Rob Hyndman
Author, forecast

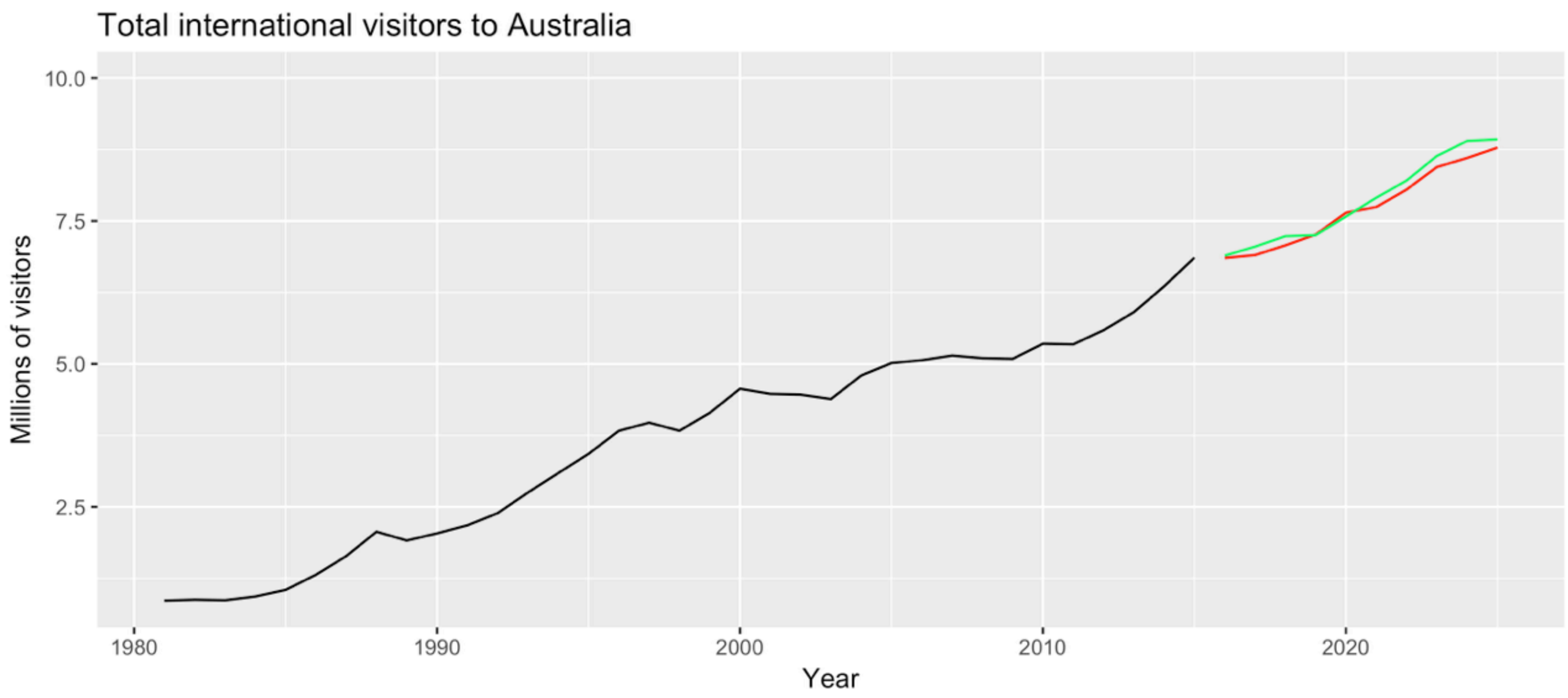
Sample futures



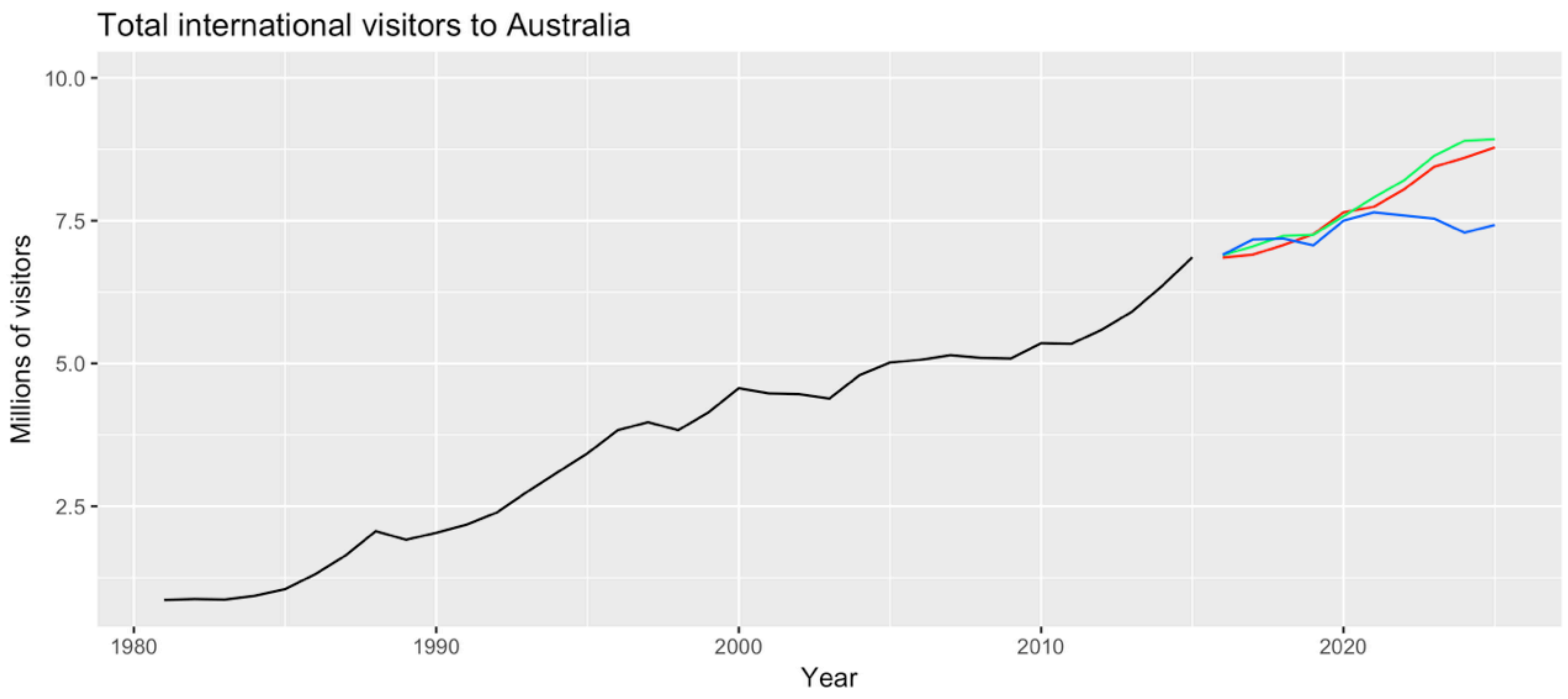
Sample futures



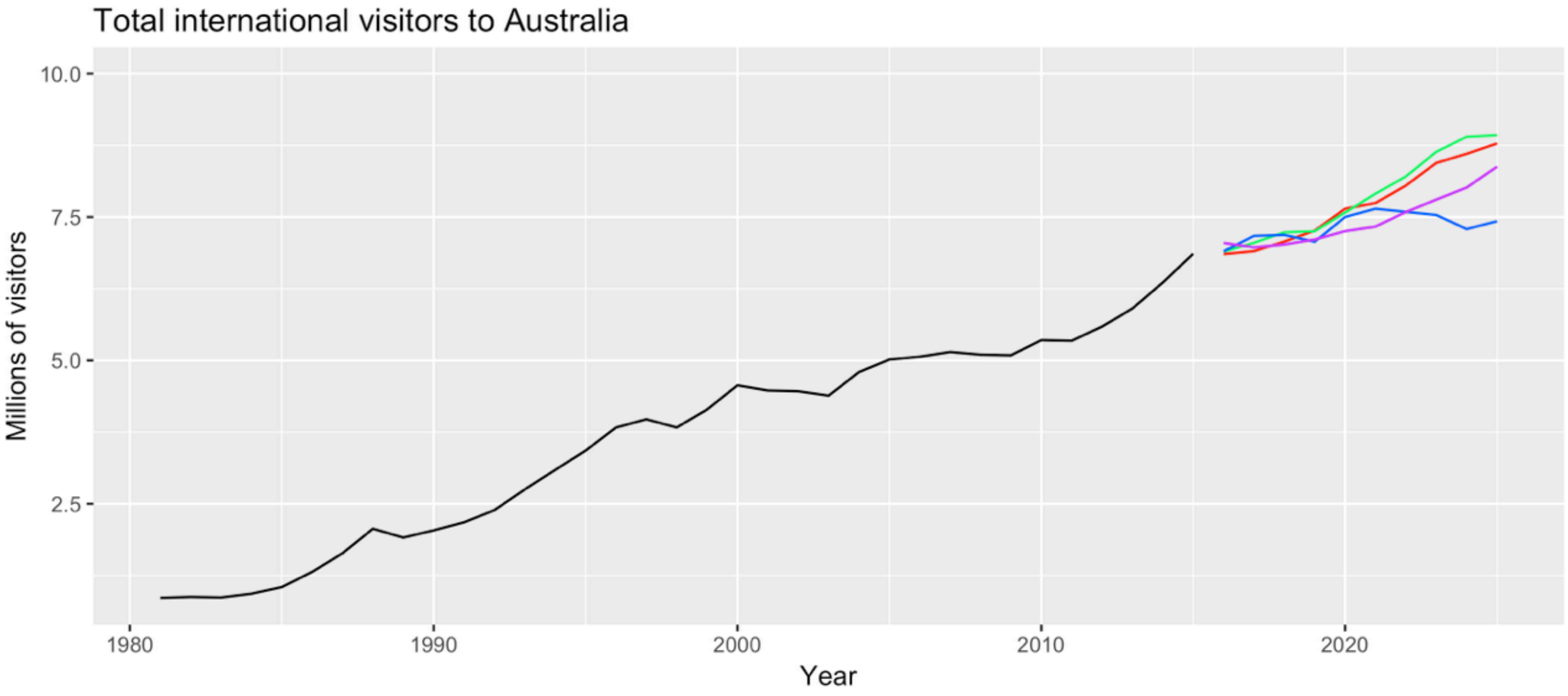
Sample futures



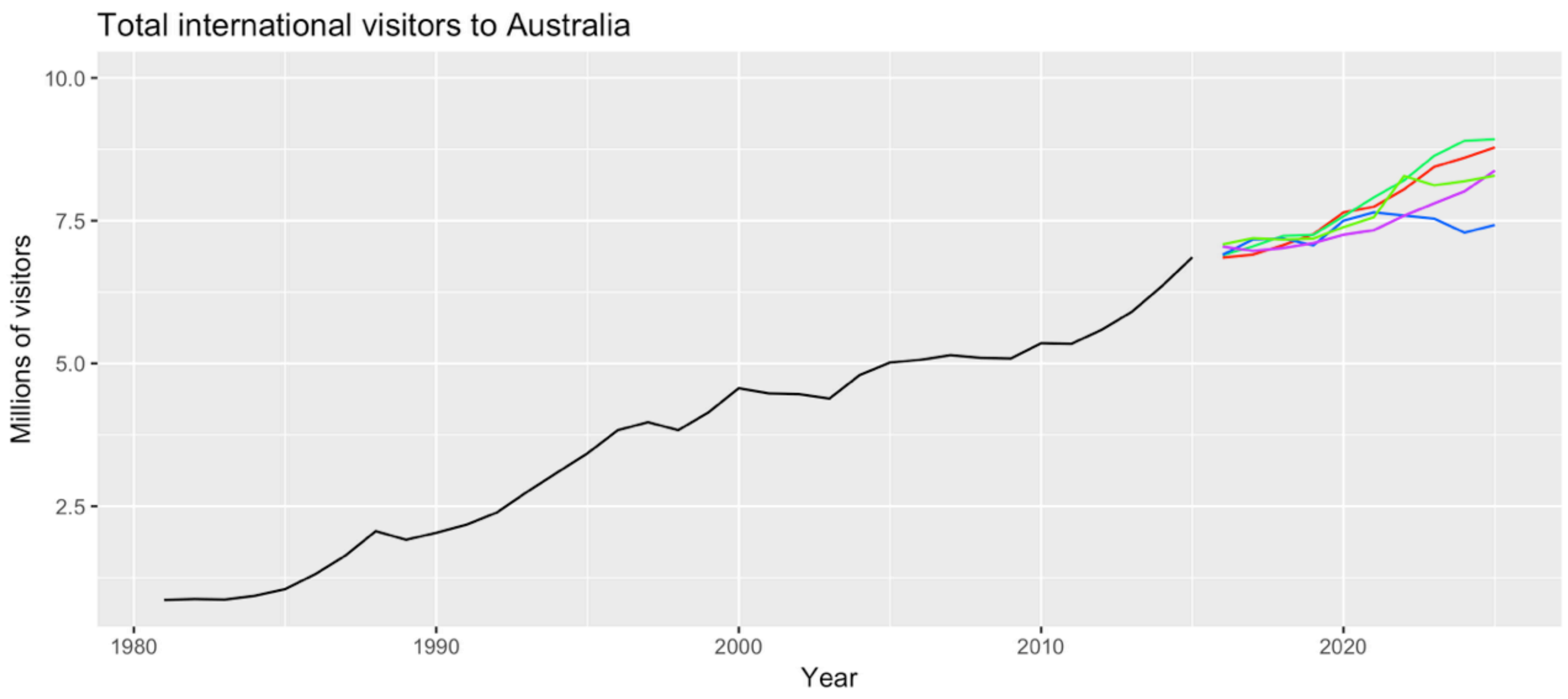
Sample futures



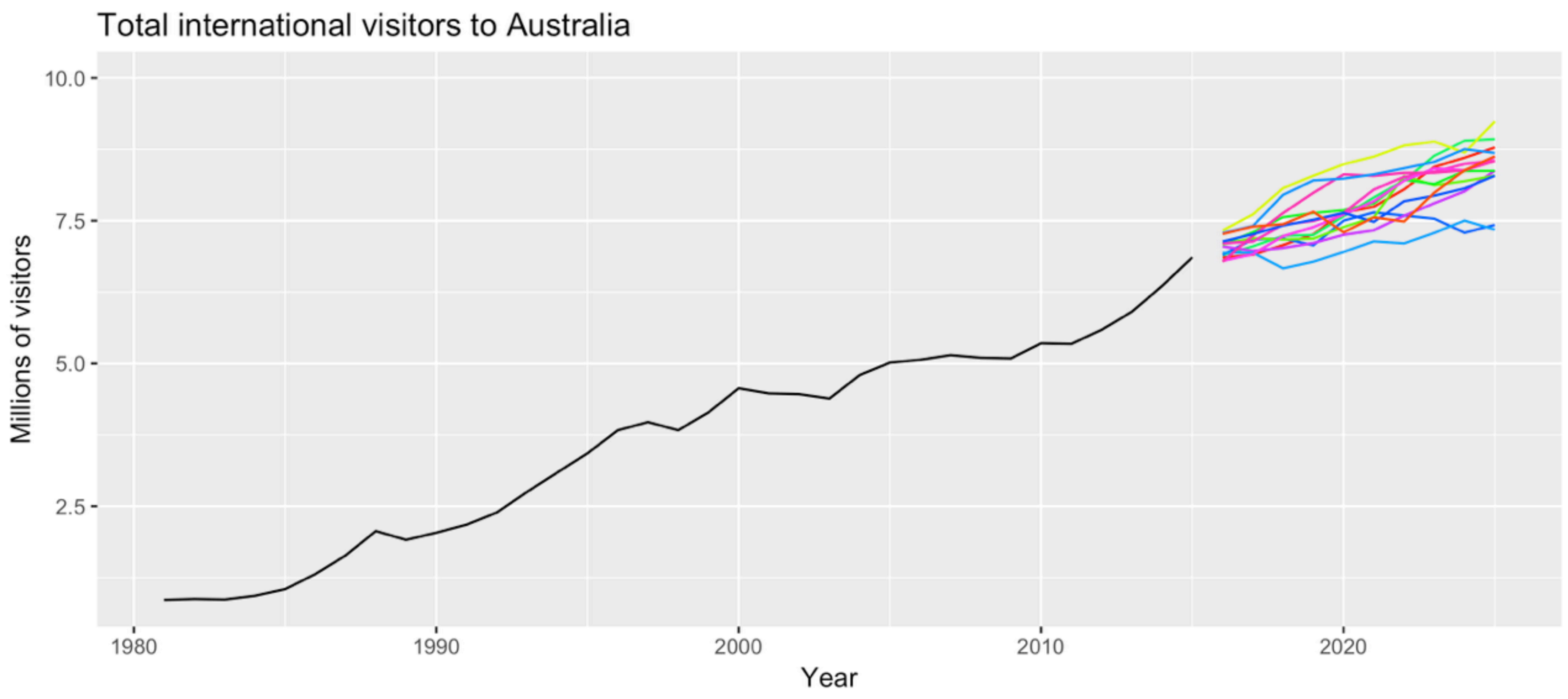
Sample futures



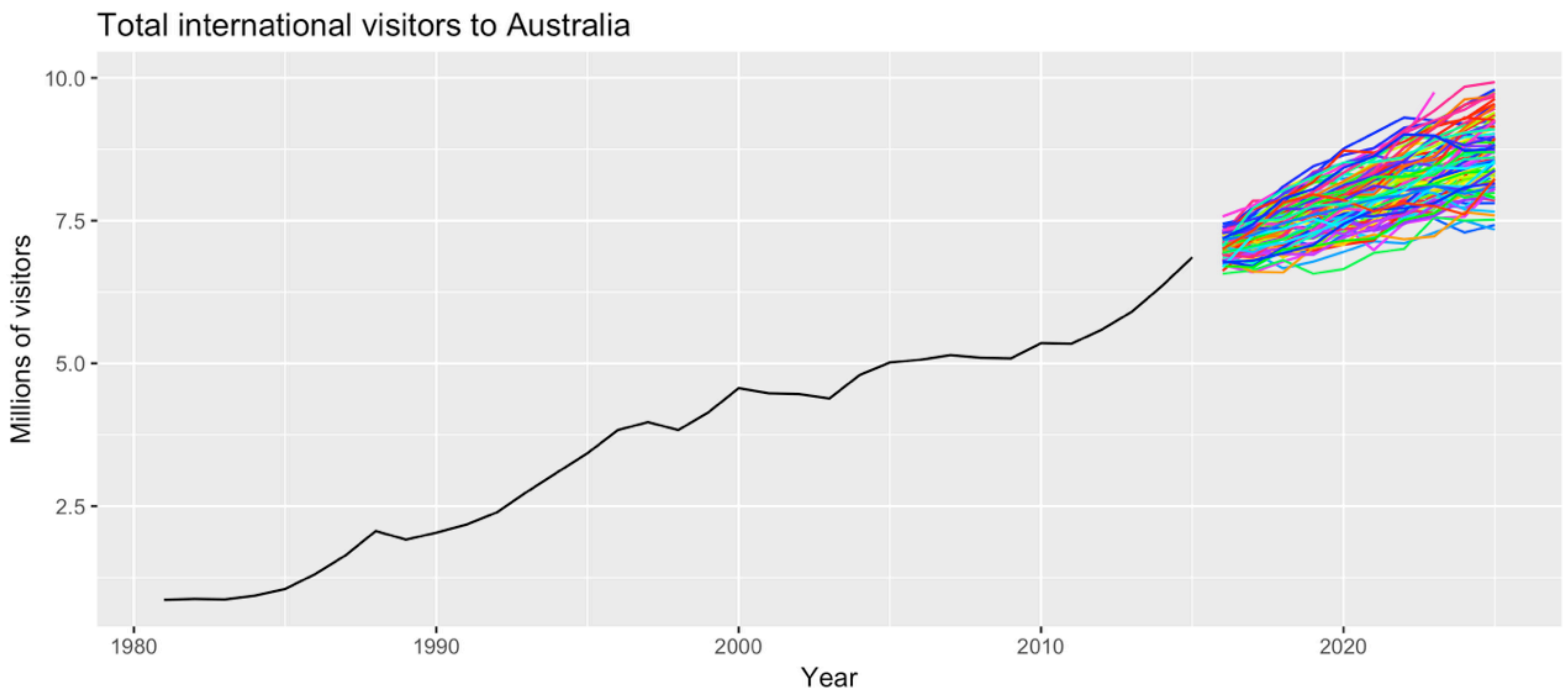
Sample futures



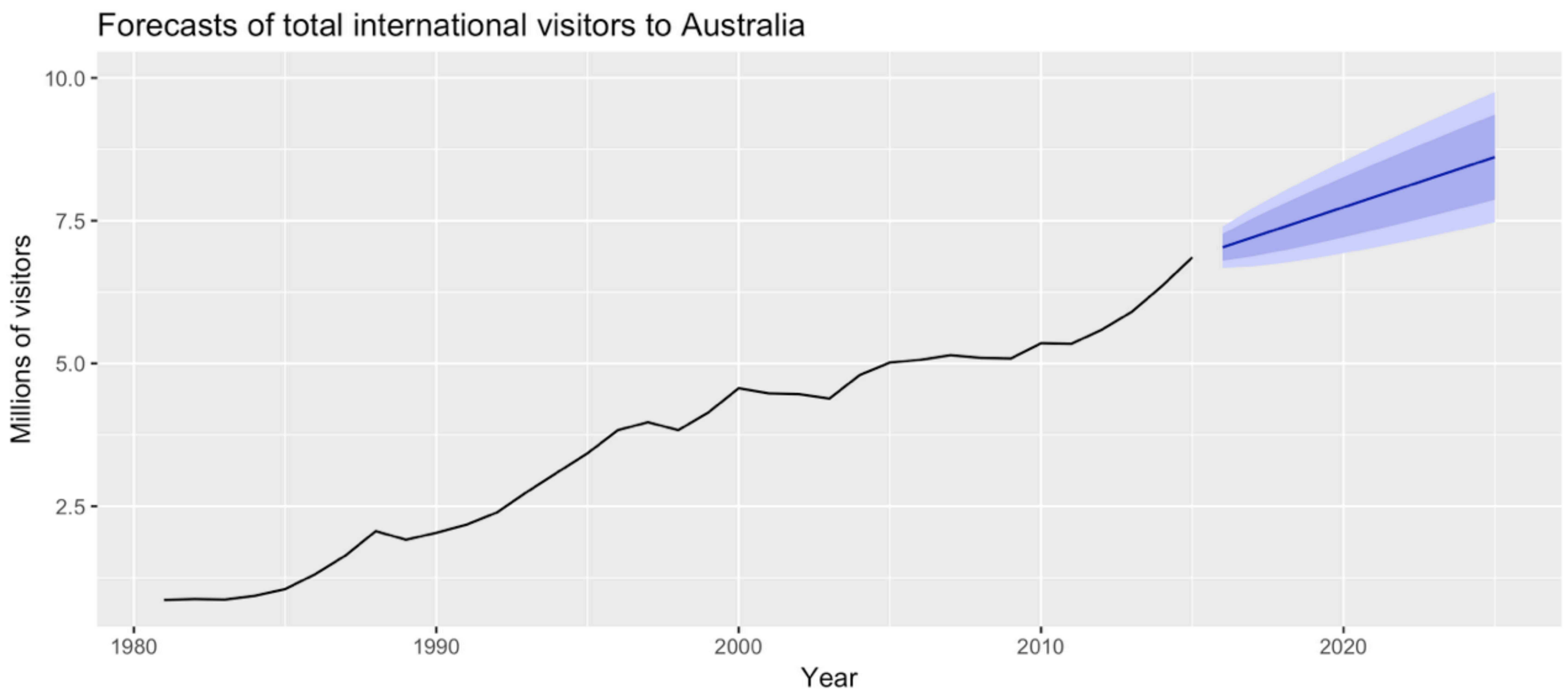
Sample futures



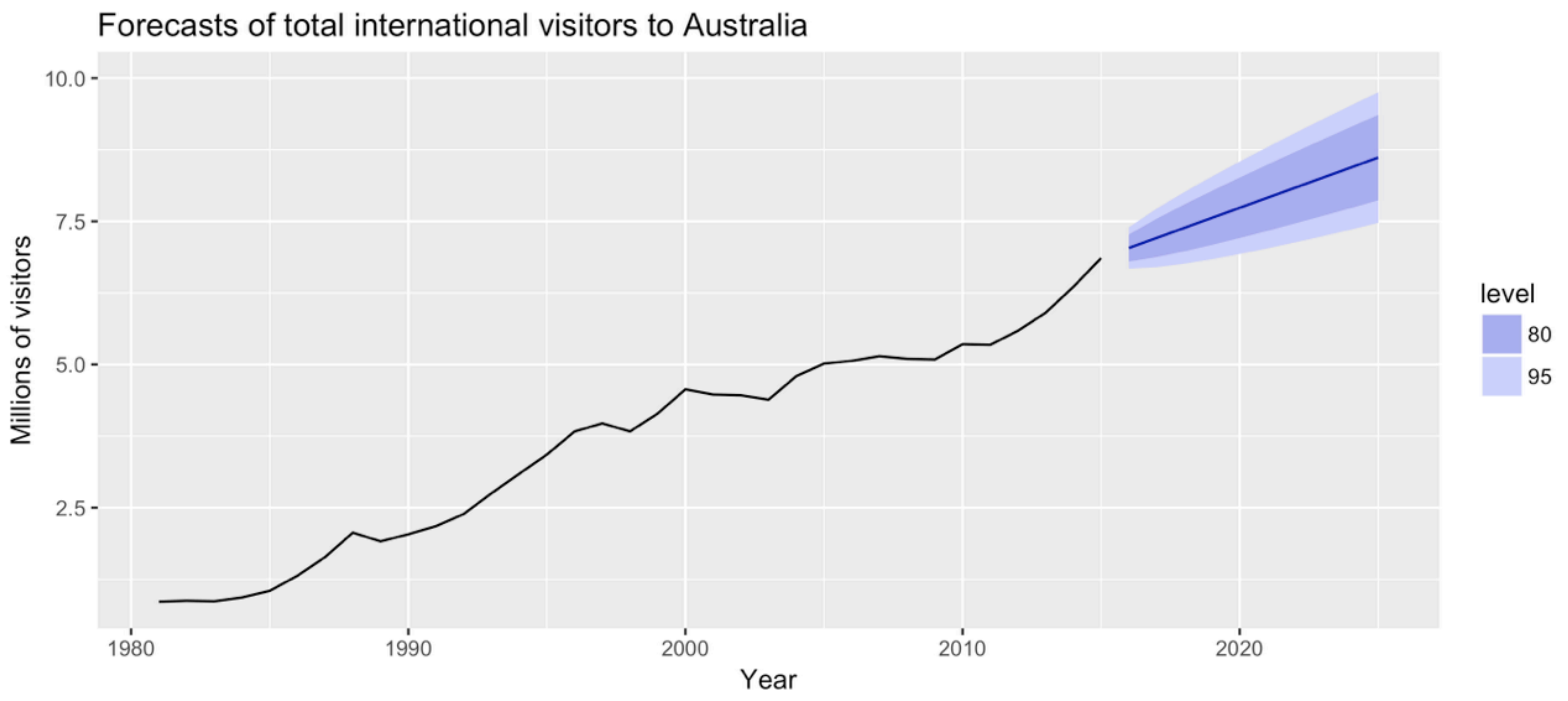
Sample futures



Forecast intervals



Forecast intervals



- The 80% forecast intervals should contain 80% of the future observations
- The 95% forecast intervals should contain 95% of the future observations

Let’s practice!

Fitted values and residuals

Fitted values and residuals

A *fitted value* is the forecast of an observation using all previous observations

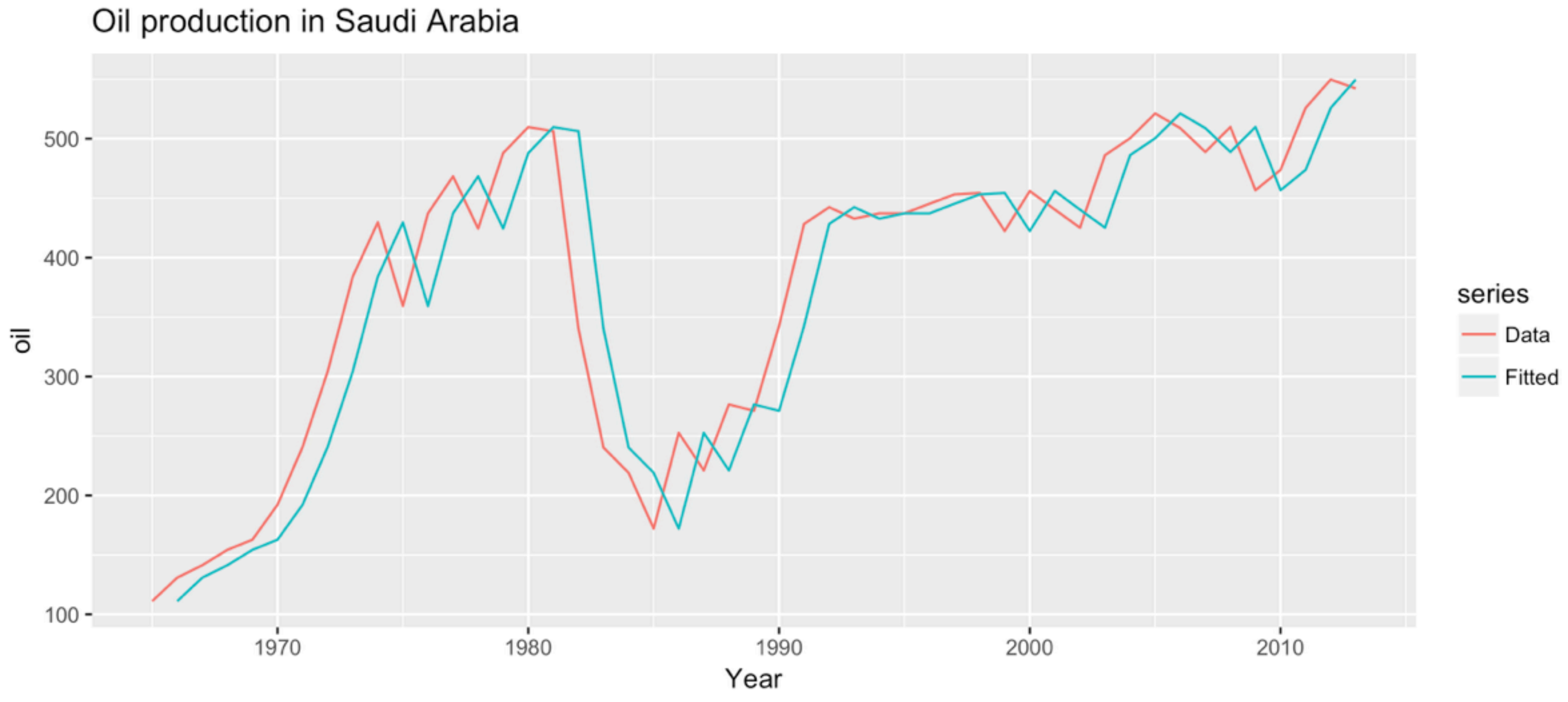
- That is, they are one-step forecasts
- Often not true forecasts since parameters are estimated on all data

A *residual* is the difference between an observation and its fitted value

- That is, they are one-step forecast errors

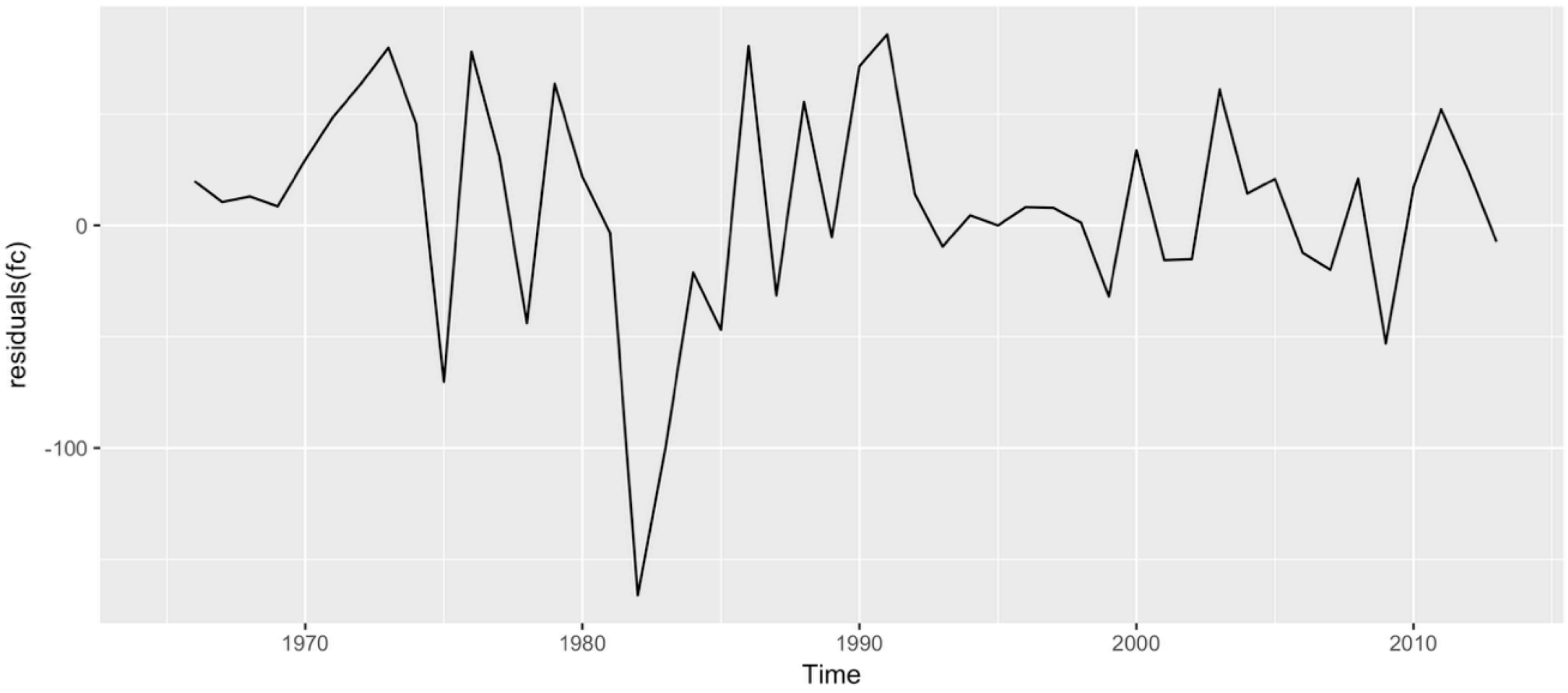
Example: oil production

```
> fc <- naive(oil)
> autoplot(oil, series = "Data") + xlab("Year") +
    autolayer(fitted(fc), series = "Fitted") +
    ggtitle("Oil production in Saudi Arabia")
```



Example: oil production

```
> autoplot(residuals(fc))
```



Residuals should look like white noise

Essential assumptions

- They should be uncorrelated
- They should have mean zero

Useful properties (for computing prediction intervals)

- They should have constant variance
- They should be normally distributed

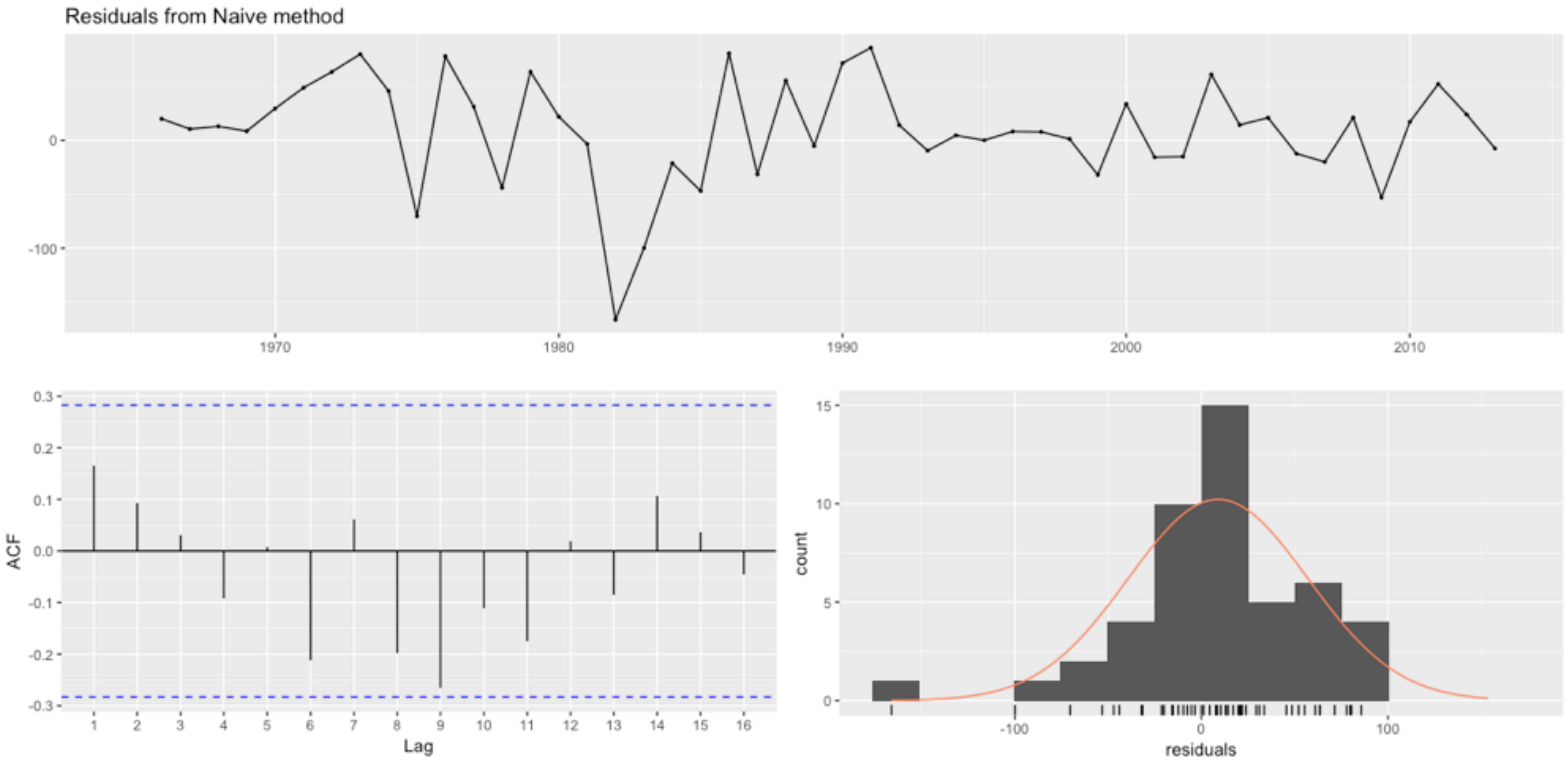
We can test these assumptions using the `checkresiduals()` function.

checkresiduals()

```
> checkresiduals(fc)
  Ljung-Box test

data:  residuals
Q* = 12.59, df = 10, p-value = 0.2475

Model df: 0.   Total lags used: 10
```



Let’s practice!

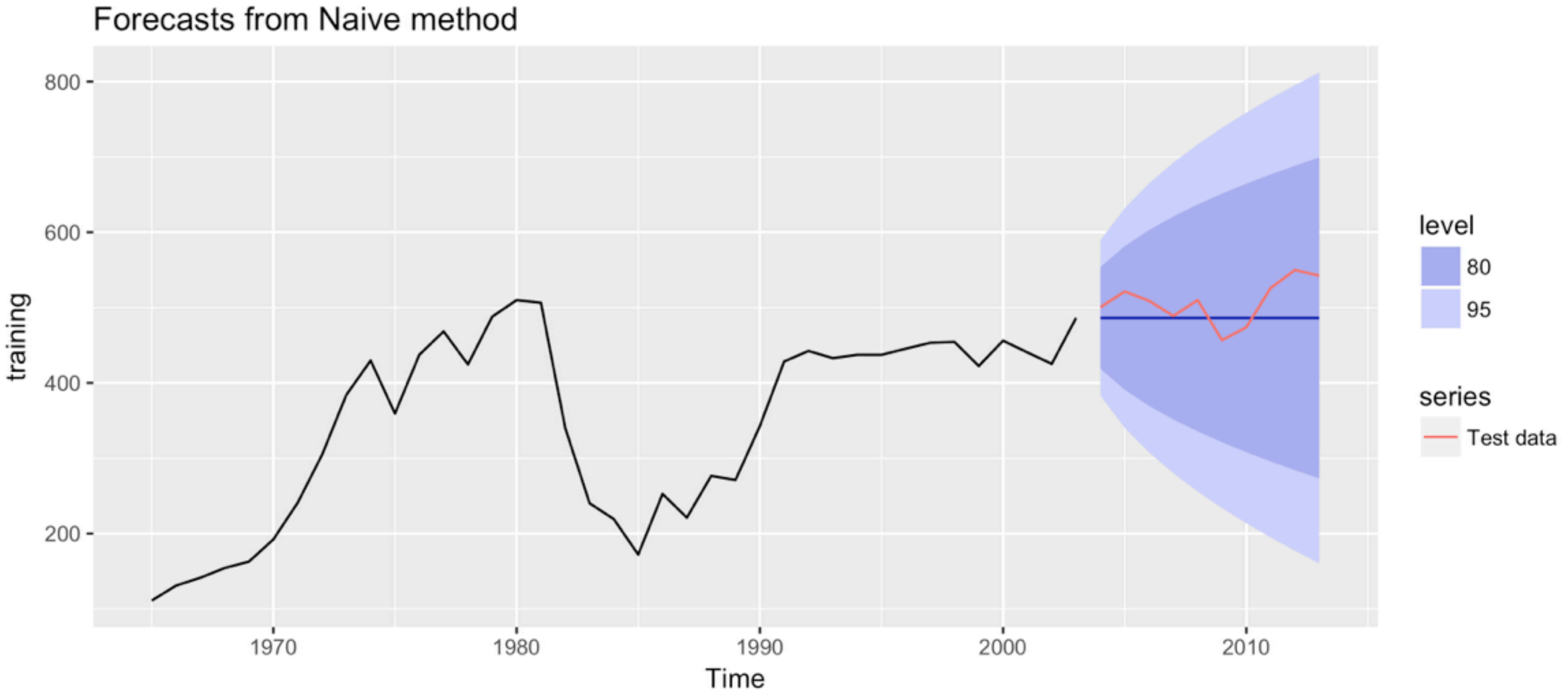
Training and test sets

Training and test sets

- The **test set** must **not** be used for *any* aspect of calculating forecasts
- Build forecasts using **training set**
- A model which fits the training data well will **not necessarily forecast well**

Example: Saudi Arabian oil production

```
> training <- window(oil, end = 2003)
> test     <- window(oil, start = 2004)
> fc       <- naive(training, h = 10)
> autoplot(fc) +
    autolayer(test, series = "Test data")
```



Forecast errors

Forecast "error" = the difference between observed value and its forecast in the test set.

≠ residuals

- which are errors on the **training set** (vs. **test set**)
- which are based on **one-step** forecasts (vs. **multi-step**)

Compute accuracy using forecast errors on test data

Measures of forecast accuracy

Definitions	Observation y_t	Forecast $\hat{y}_t$	Forecast error $e_t = y_t - \hat{y}_t$

Accuracy measure	Calculation
Mean Absolute Error	$MAE = average(\|e_t\|)$
Mean Squared Error	$MSE = average(e_t^2)$
Mean Absolute Percentage Error	$MAPE = 100 \times average(\|\frac{e_t}{y_t}\|)$
Mean Absolute Scaled Error	$MASE = MAE/Q$

* Where Q is a scaling constant.

The accuracy() command

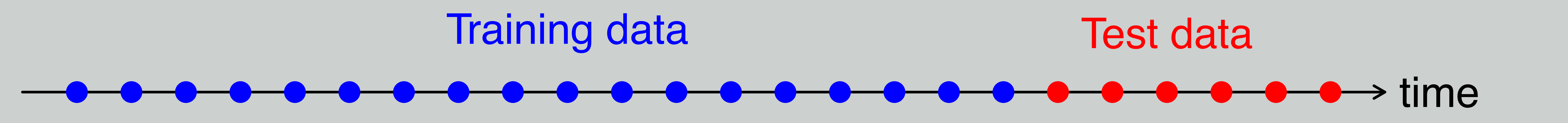
```
> accuracy(fc, test)
                 ME  RMSE   MAE   MPE   MAPE   MASE   ACF1 Theil's U
Training set  9.874 52.56 39.43 2.507 12.571 1.0000 0.1802        NA
Test set     21.602 35.10 29.98 3.964  5.778 0.7603 0.4030     1.185
```

Let’s practice!

Time series cross-validation

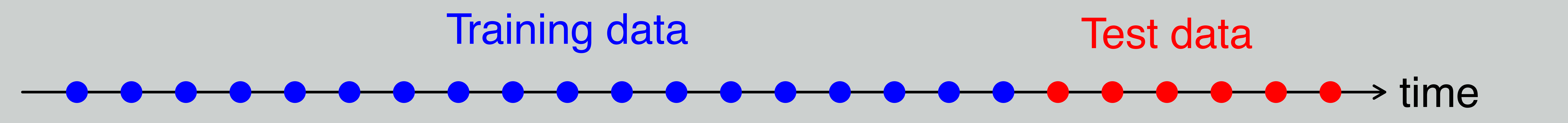
Time series cross-validation

Traditional evaluation

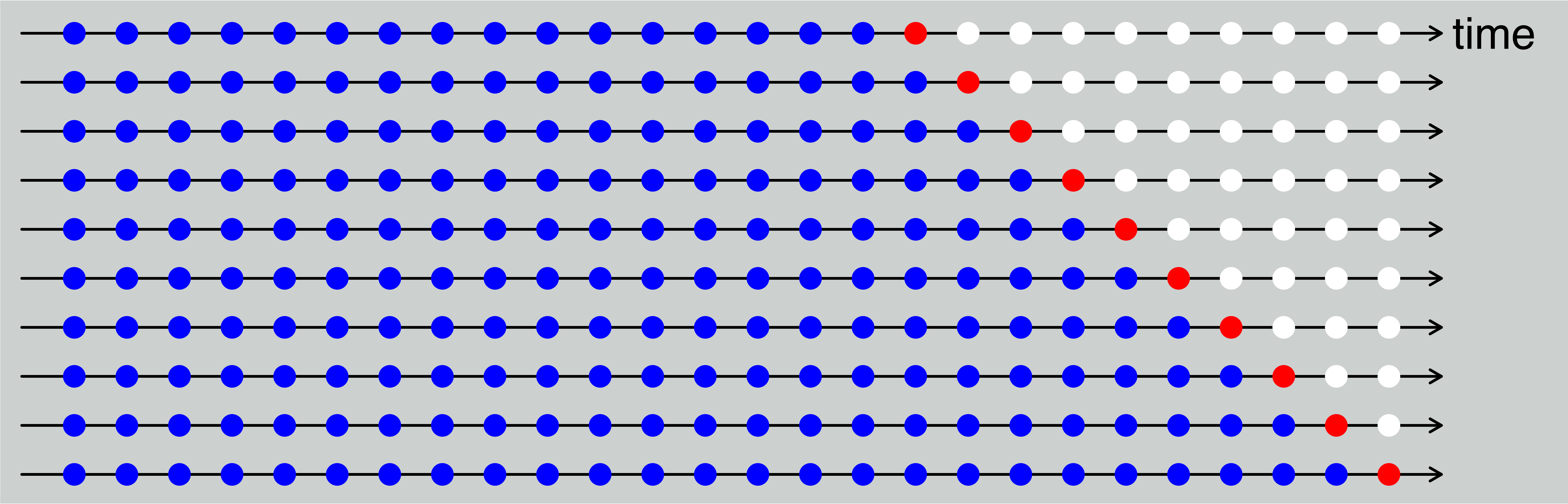


Time series cross-validation

Traditional evaluation

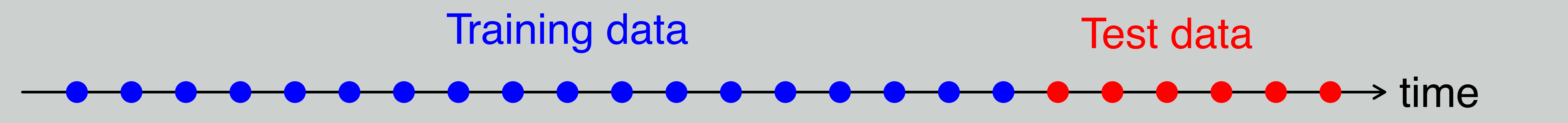


Time series cross-validation

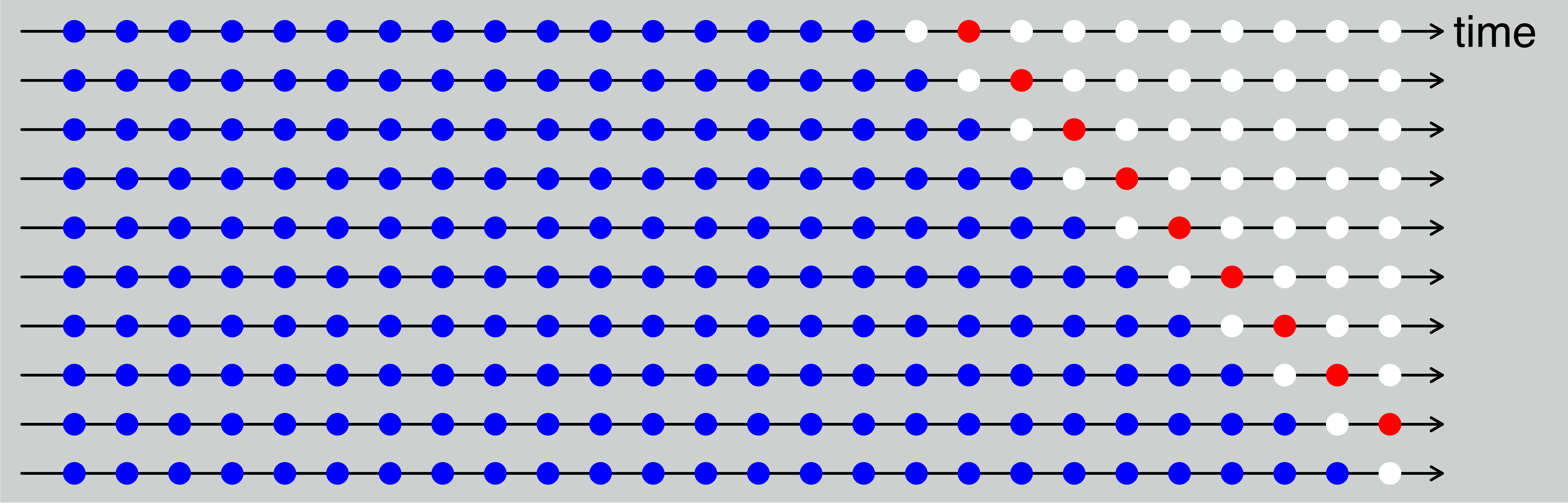


Time series cross-validation

Traditional evaluation

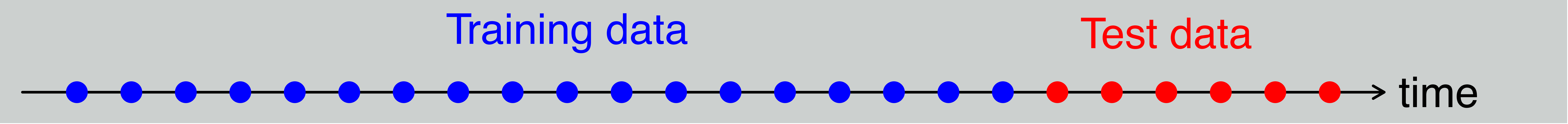


Time series cross-validation

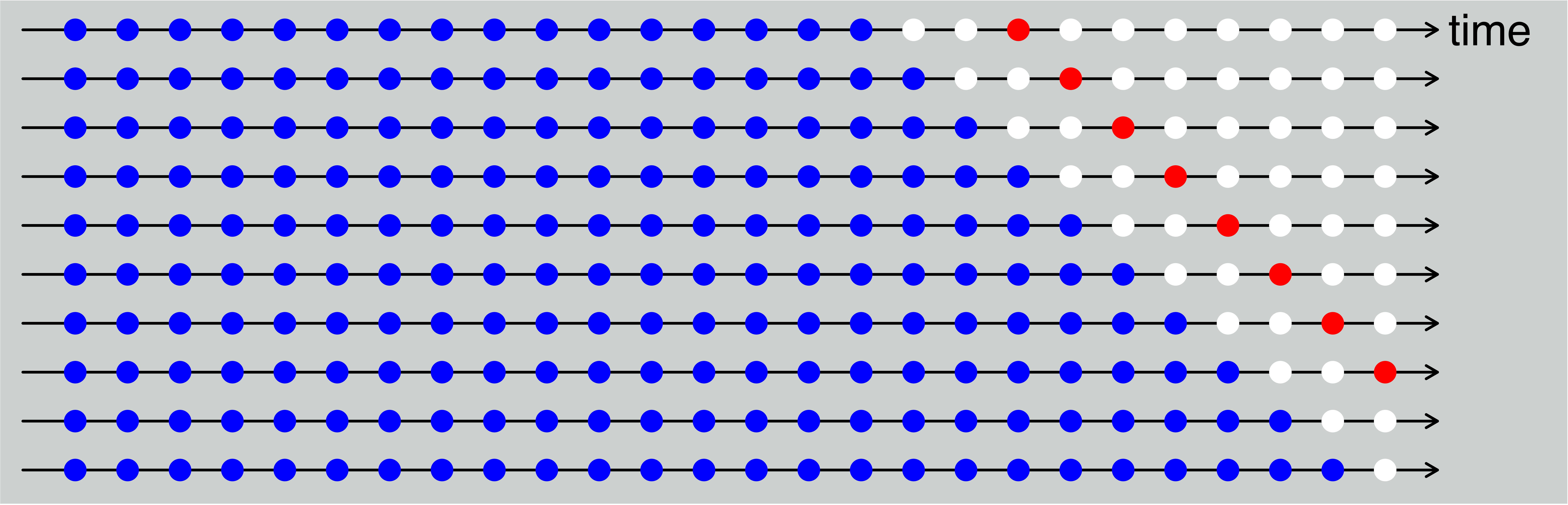


Time series cross-validation

Traditional evaluation



Time series cross-validation



tsCV function

MSE using time series cross-validation

```
>  e <- tsCV (oil, forecastfunction = naive, h = 1)
> mean(e^2 , na.rm = TRUE)
[1] 2355.753
```

When there are no parameters to be estimated, `tsCV` with `h=1` will give the same values as residuals

tsCV function

```
> sq <- function(u){u^2}
> for(h in 1:10)
+ {
+   oil %>% tsCV(forecastfunction = naive, h = h) %>%
+     sq() %>% mean(na.rm = TRUE) %>% print()
+ }
[1] 2355.753
[1] 5734.838
[1] 9842.239
[1] 14300
[1] 18560.89
[1] 23264.41
[1] 26932.8
[1] 30766.14
[1] 32892.2
[1] 32986.21
```

The MSE increases with the forecast horizon

tsCV function

- Choose the model with the smallest MSE computed using time series cross-validation
- Compute it at the forecast horizon of most interest to you

Let's practice!