

Time Series Deep Learning

Matt Dancho
Founder & CEO, Business Science
business-science.io

April 19, 2018



About Business Science



- We are **applications** people that build tools to solve tough problems
- We **serve** the data science **community**
- We **empower** organizations
 - Expert plan-based coaching, training, & consultation
 - Specialize in R Package Development, Shiny Apps, and more!
 - NEW!! Business Science University (BSU)



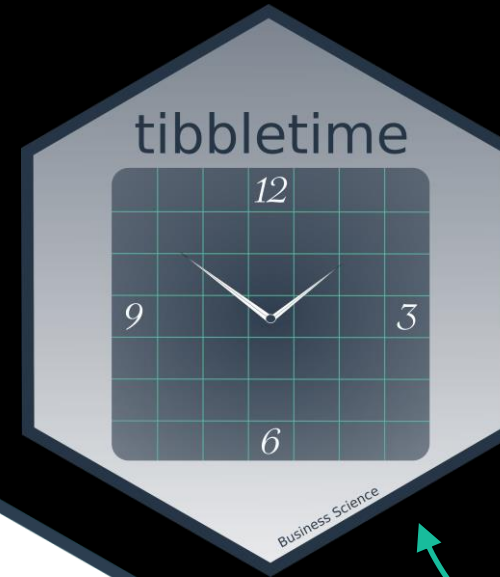
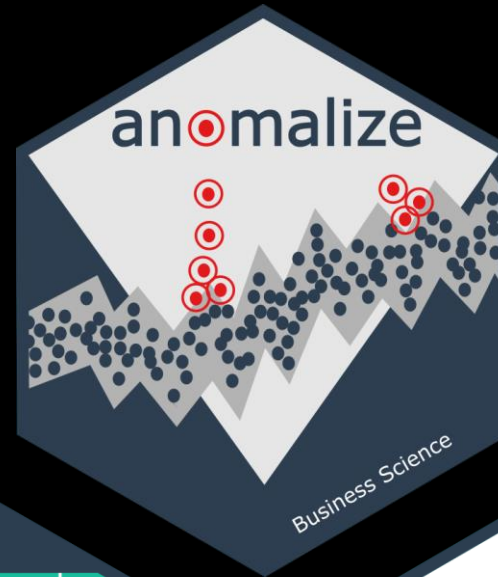
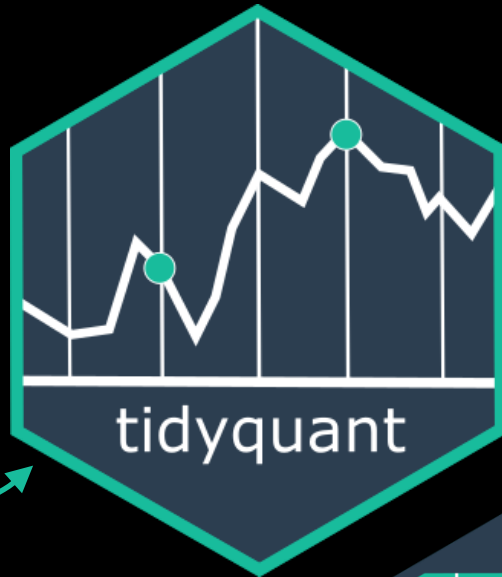
We Love Time Series

Open Source Packages

NEW!!
Anomaly
Detection



Most Popular
Financial Data
Stock/Portfolio
Analysis



Helps Do
Time-Based
Forecasting
w/ forecast



Time-Aware
tibbles
(dplyr)

Time Series
ML

Objectives



- Importance of Time Series
- Challenges & Opportunities
- Deep Learning for Time Series



Importance of Time Series

Importance of Time Series

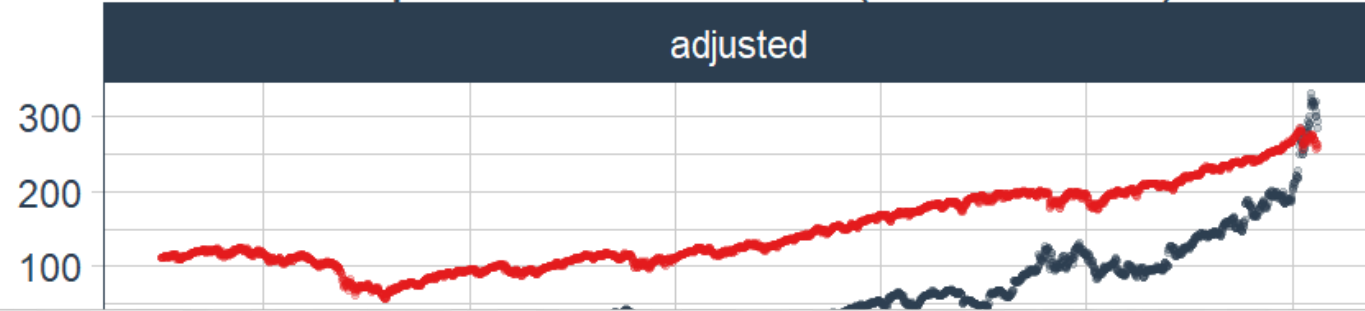
Netflix vs S&P500

Gain vs Pain

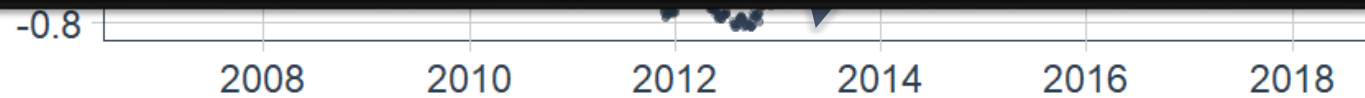


Importance of Time Series

Netflix Compared To S&P500 (Benchmark)



Prediction Is Critical



symbol • NFLX • SP500



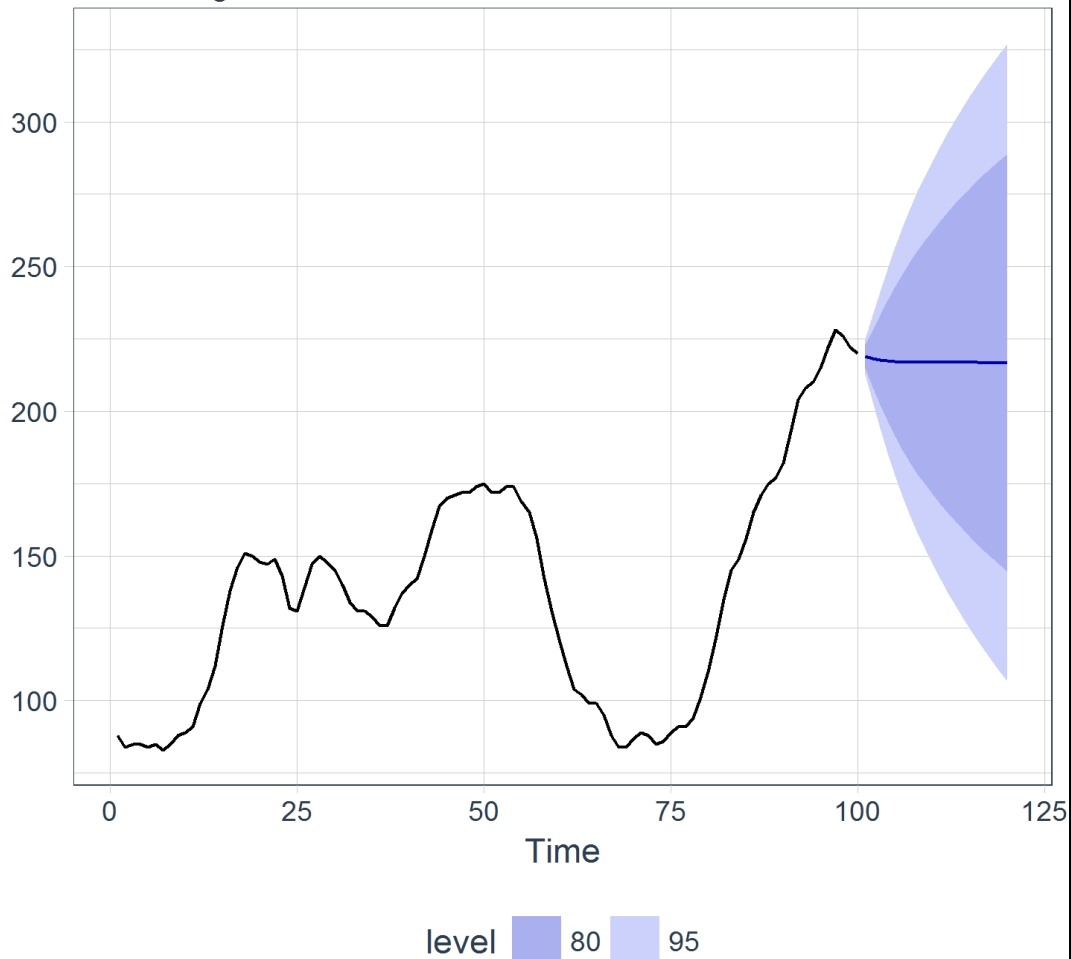
New Challenges

Data Is Changing



Forecasts from ARIMA(1,1,1)

WWWusage dataset



How Data Used to Be:

Short Time Series

Viewed As Univariate

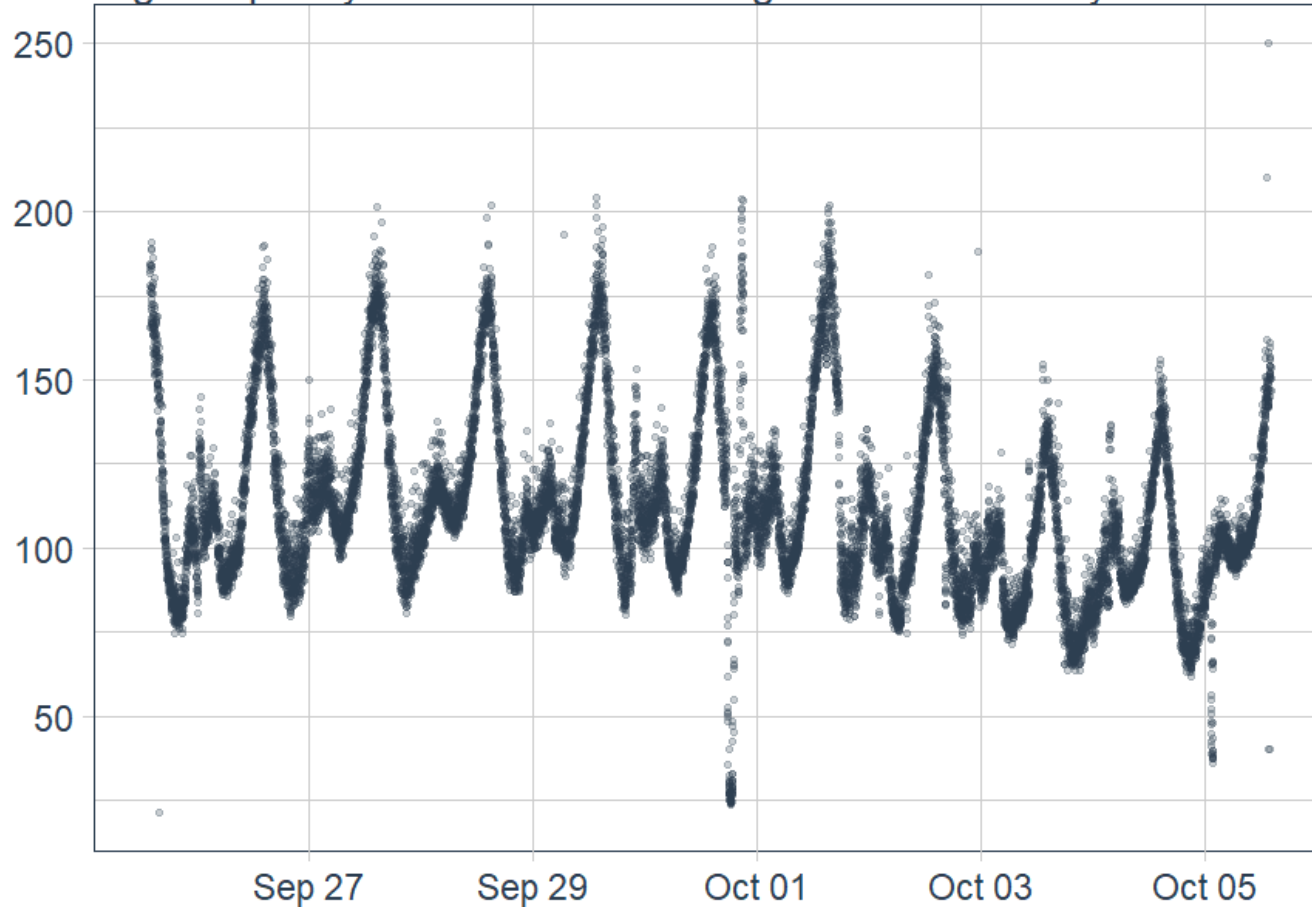
Wide Forecast Margins

Today's Data Is Different



Twitter Data: By Minute

High frequency with anomalies and significant seasonality



Source: <https://github.com/twitter/AnomalyDetection>

High Frequencies

Time-Based Pattern

Multivariate Systems

Sparse Data

Event Driven

Anomalies



New Opportunities For Time Series Prediction

New Techniques Available



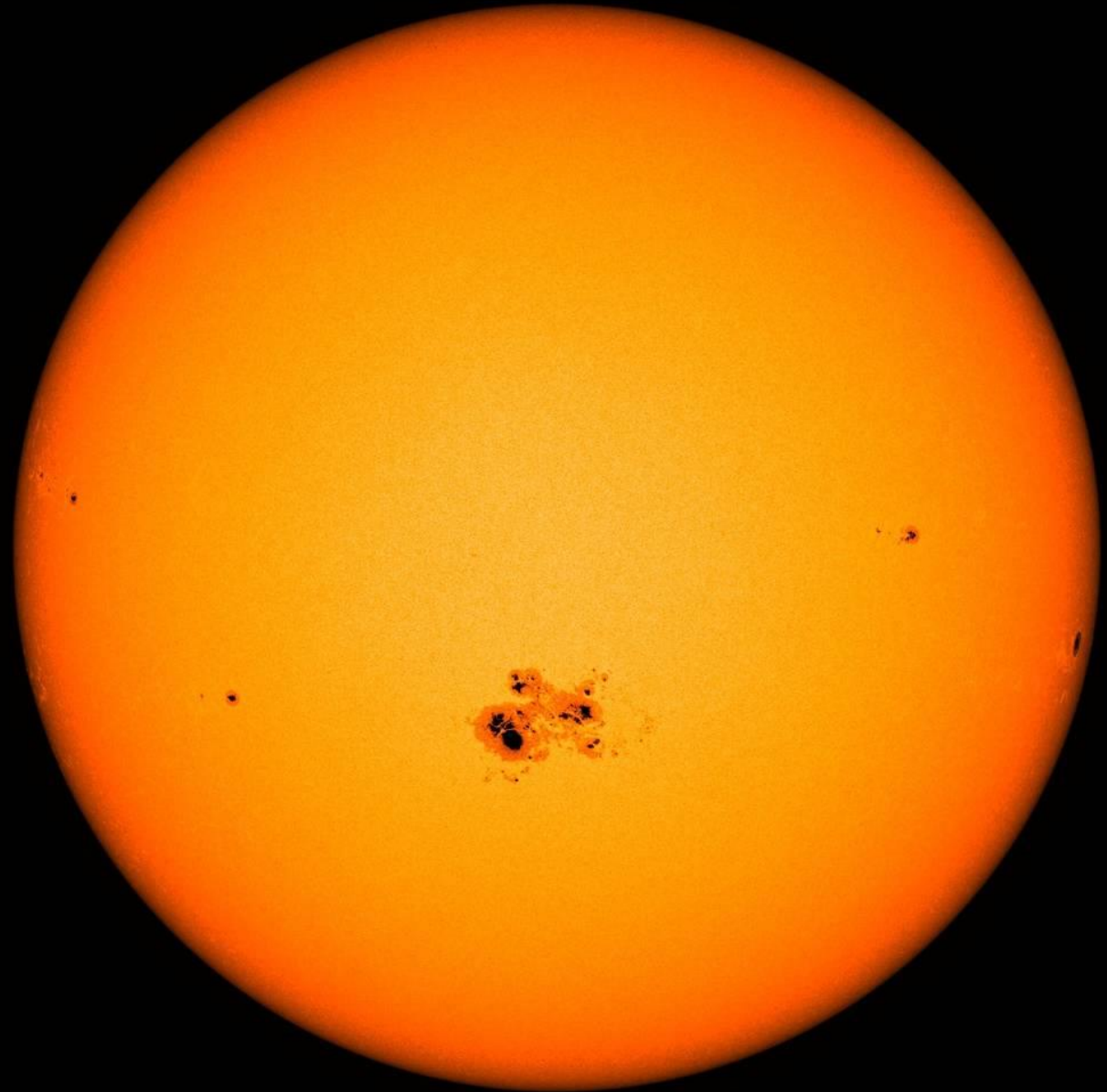
- Machine Learning:
 - Time Series Signature Pattern Recognition
 - Deep Learning:
 - Keras Stateful LSTMs
- Our Focus
Today
-



Deep Learning with Keras Stateful LSTM

Sunspots Data

- Solar Phenomenon
- Great Example of ML Tool-Application Fit

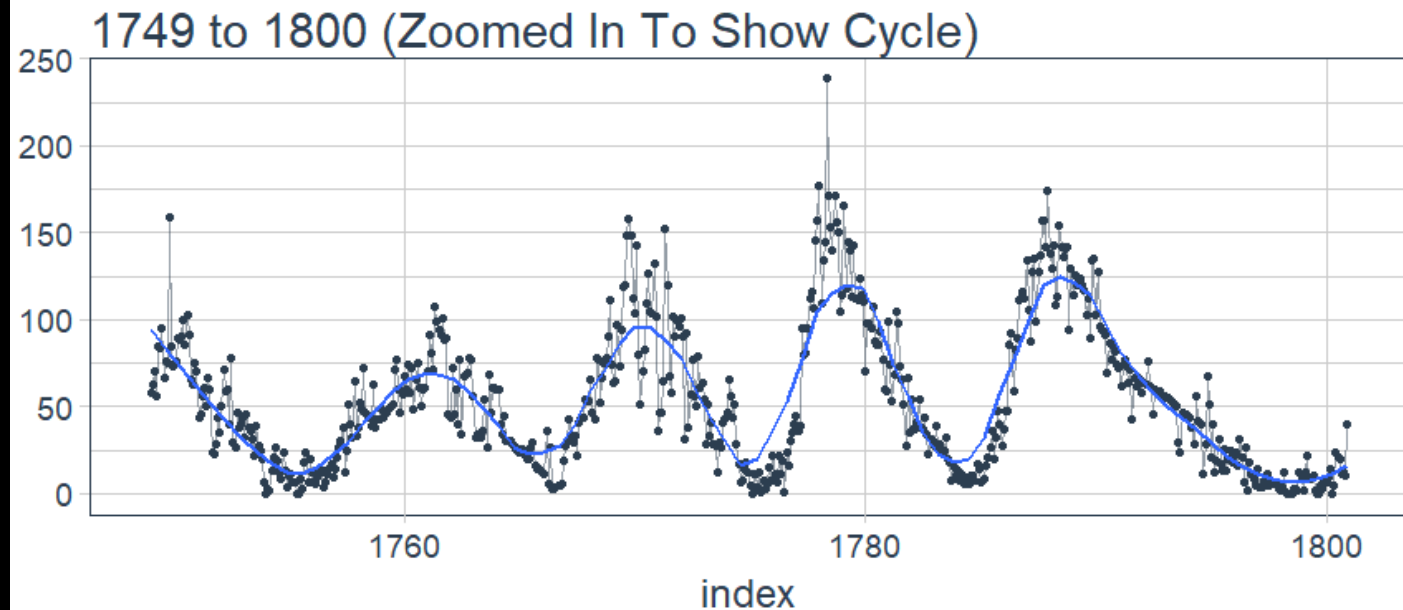
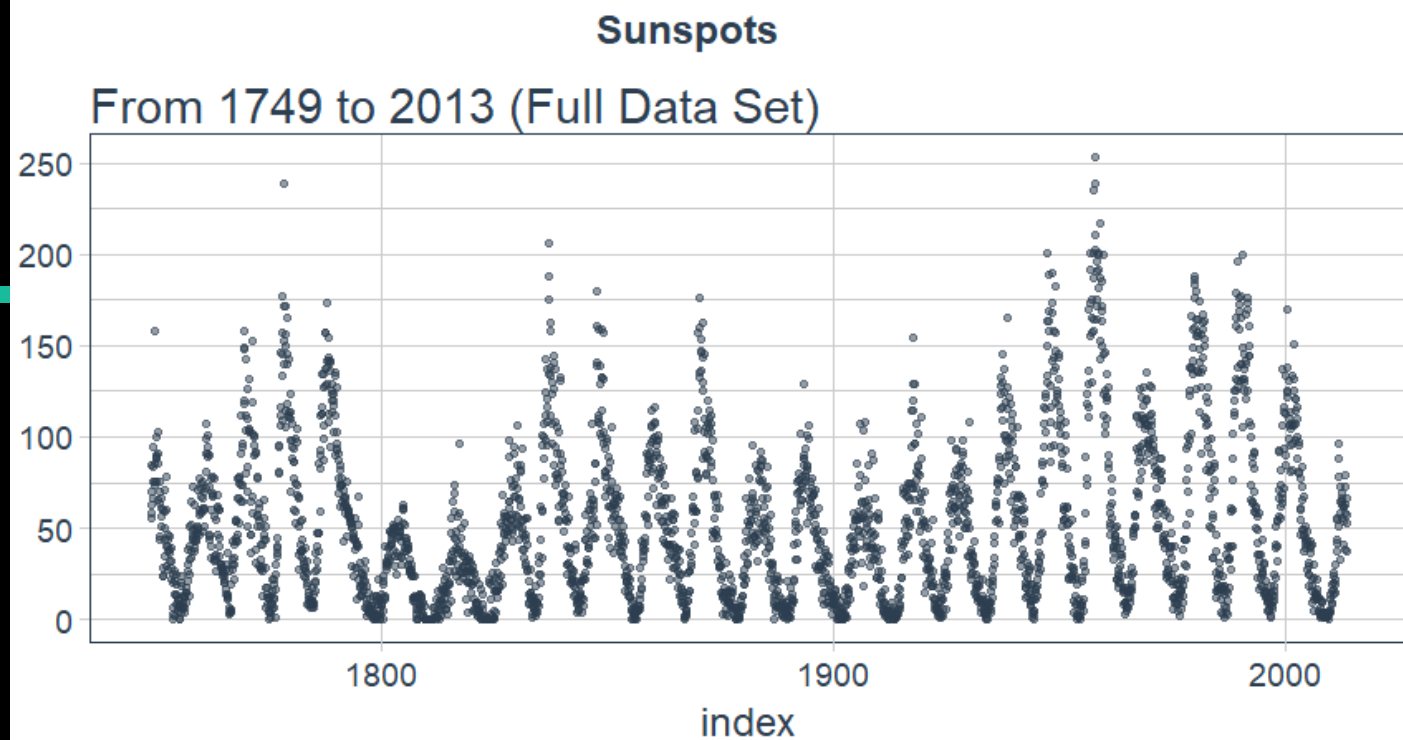


SDO/HMI Quick-Look Continuum: 20141023_131500

Source: <https://www.nasa.gov/content/goddard/largest-sunspot-of-solar-cycle>

Sunspots Data

- Looks easy to predict
- Difficult
- Cycle & amplitude changes





Objective:
Predict
Next 10 Years
Using Keras Stateful LSTM

Learning Path



- What is an LSTM?
- Is LSTM a **good candidate** for the problem?
- Develop a **Stateful LSTM Model** using **Keras**
- Time Series Cross Validation using **Backtesting**

What is an LSTM?

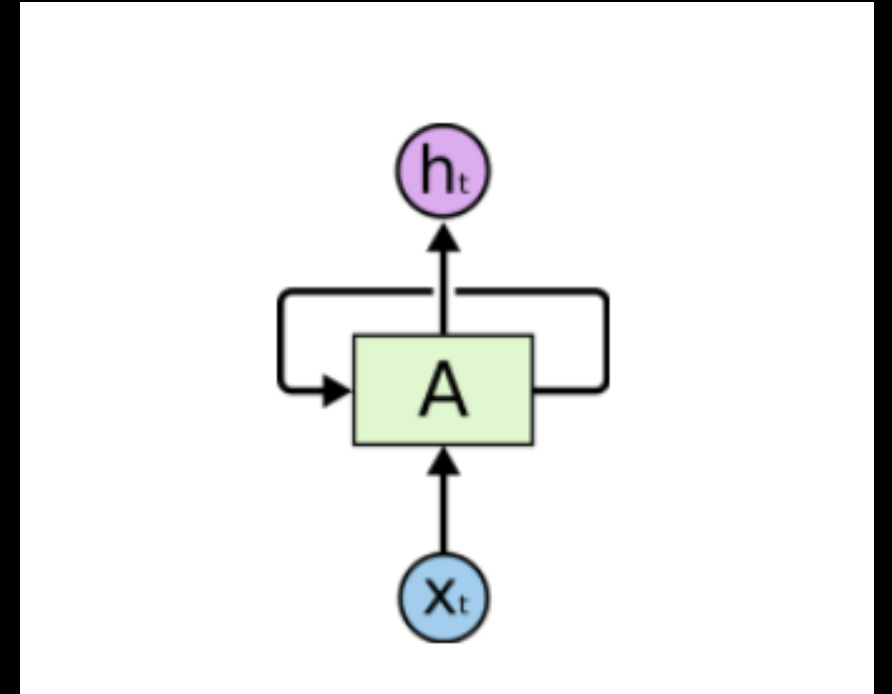


- Special Type of Recurrent Neural Network (RNN)
- Long-Short Term Memory
- Models sequence data
- “*Understanding LSTM Networks*” by Christopher Olah

RNN

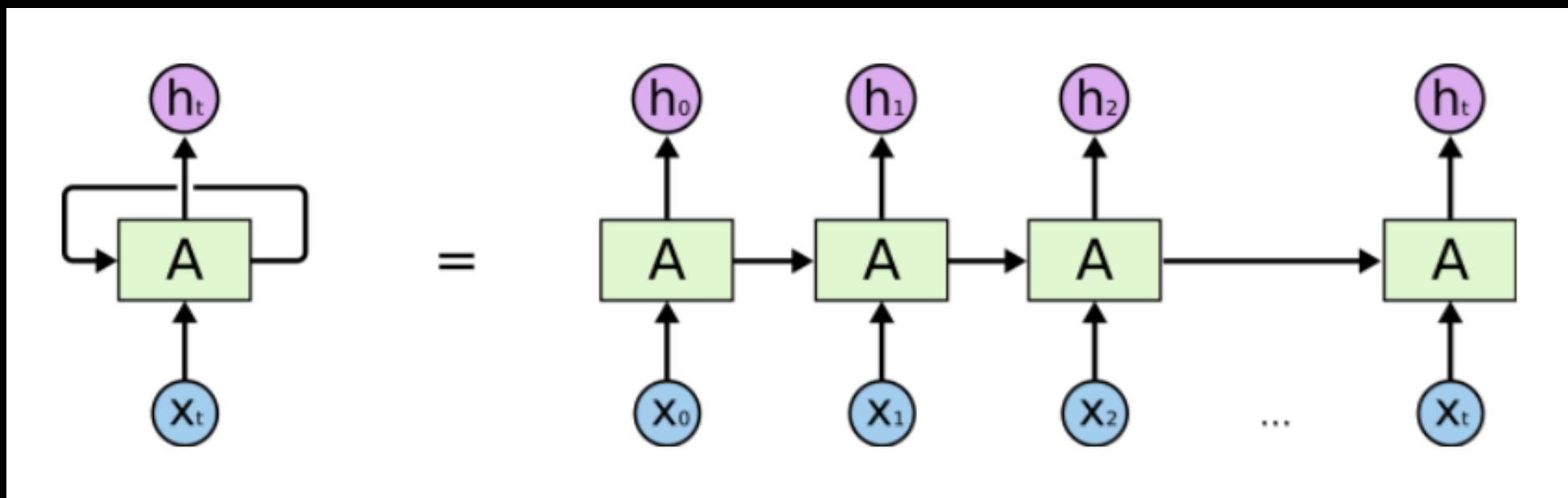


- RNNs have loops
- Enables persistence
- Strengths: Sequences
 - Learning context based on what happened previously
 - Speech recognition, image classifying, etc

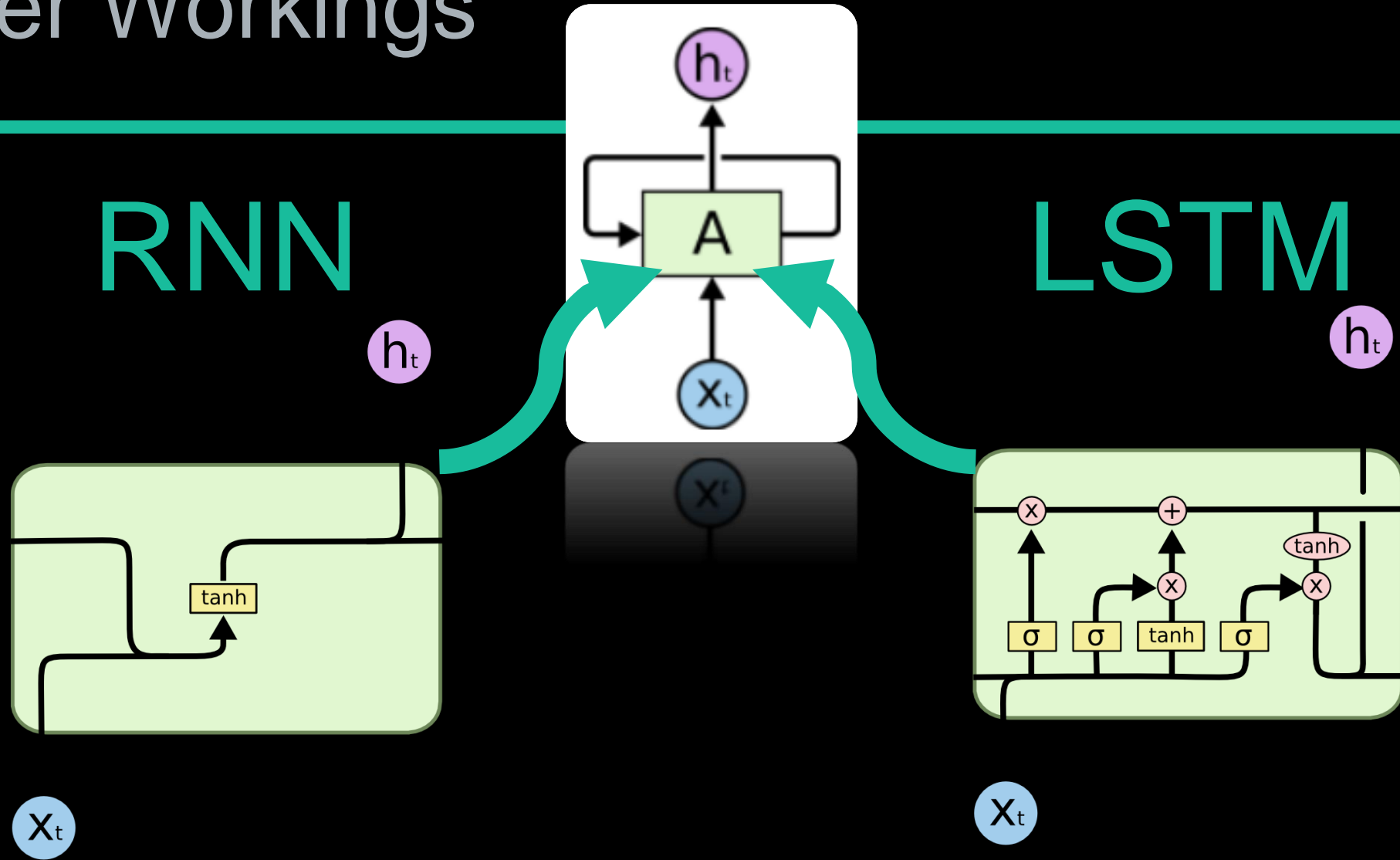




Unrolling The RNN Loop



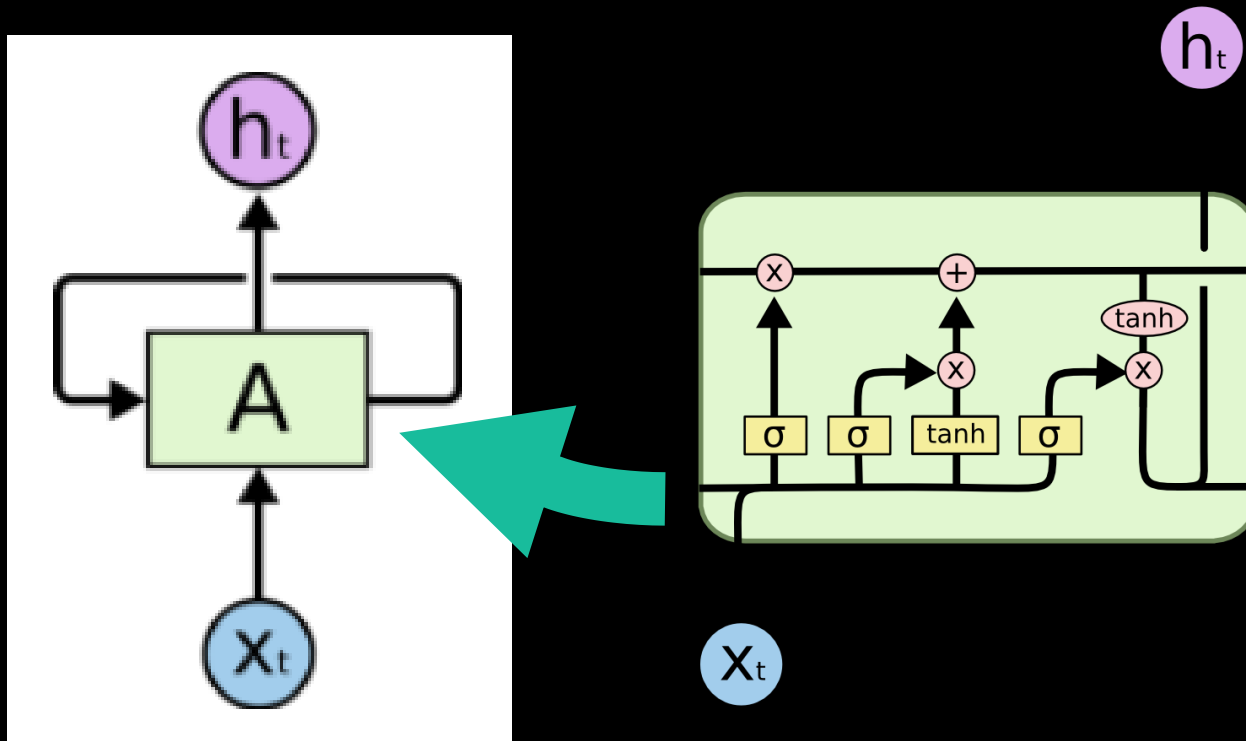
Inner Workings



LSTM



LSTM



**Internal Cell Complexity
Solves problem with
basic RNNs:**

**Long Term
Dependencies**



Is LSTM A
Good Candidate?

Is LSTM A Good Candidate?

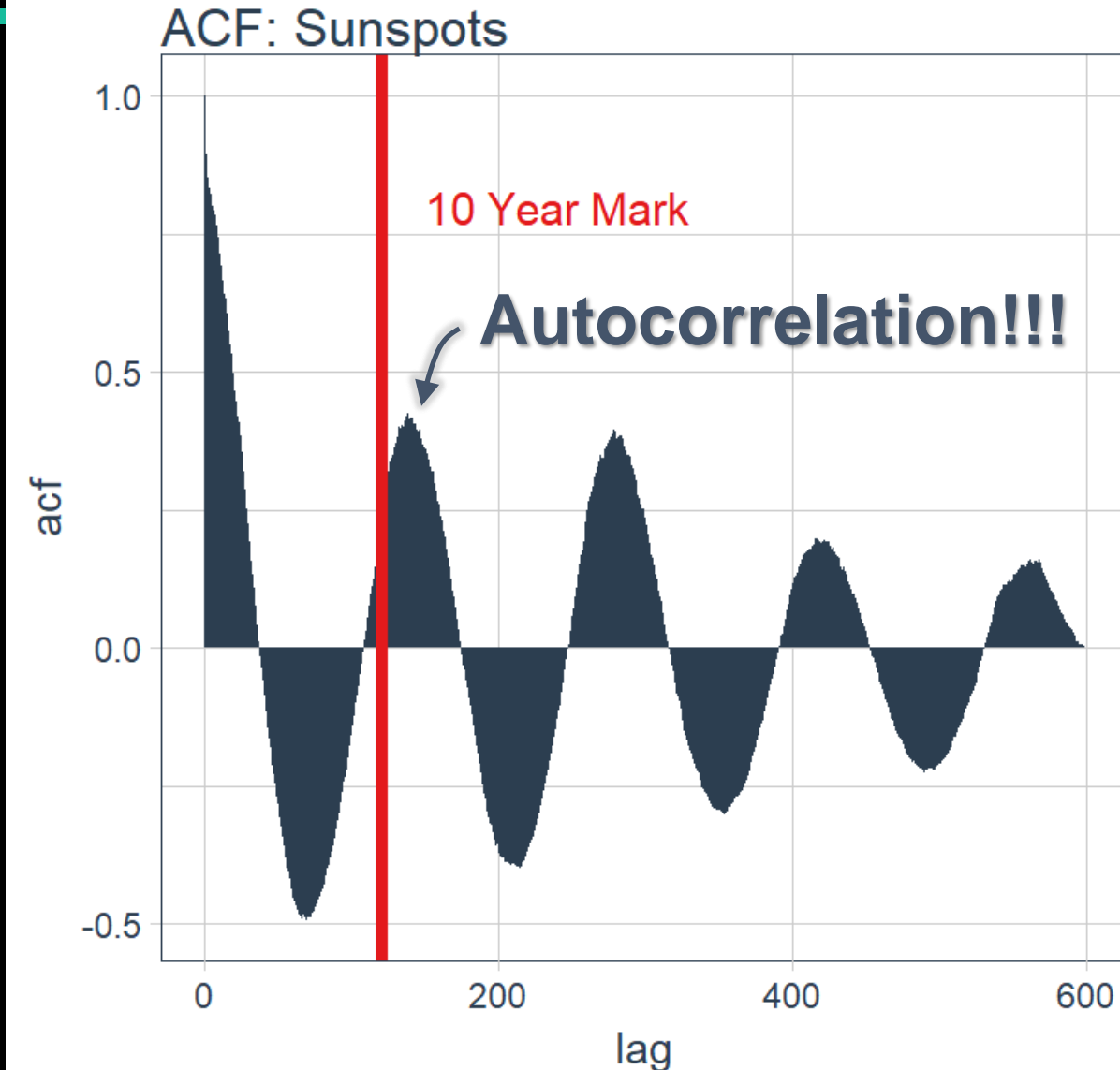


LSTM's take advantage of autocorrelation

Is LSTM A Good Candidate?



- Autocorrelation
- Batch prediction
- Lag at 120 months (10 years) to predict next 10 years!
- LSTM is good candidate!





Developing A Stateful LSTM With Keras

Keras Stateful LSTM Terminology



- **Batch Size:**
 - The batch size is the number of training examples in one forward/backward pass of a RNN before a weight update
- **Time Steps:**
 - A time step is the number of lags included in the training/testing set
- **Epochs:**
 - The epochs are the total number of forward/backward pass iterations



Keras Input Setup

```
# setup inputs  
lag_setting  <- 120 # = nrow(df_tst)  
batch_size  <- 40  
train_length <- 440  
tsteps      <- 1  
epochs      <- 300
```



Keras Stateful Model Setup

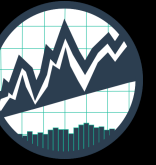
- **Stateful:** No reshuffling between batches
- **Time dependency** is preserved
- **Higher accuracy** than stateless

Modeling With Keras



Stacking Bricks

Keras Stateful LSTM



Stacking Bricks

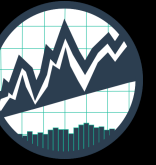
```
model <- keras_model_sequential()
```

```
model %>%  
  layer_lstm(units      = 50,  
             input_shape = c(tsteps, 1),  
             batch_size  = batch_size,  
             return_sequences = TRUE,  
             stateful    = TRUE) %>%  
  layer_lstm(units      = 50,  
             return_sequences = FALSE,  
             stateful    = TRUE) %>%  
  layer_dense(units = 1)
```


```
model %>%  
  compile(loss = 'mae', optimizer = 'adam')
```

```
model
```

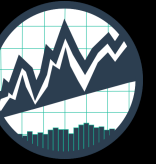
```
## Model  
##  
## Layer (type)                Output Shape          Param #  
## =====  
## lstm_1 (LSTM)                (40, 1, 50)           10400  
##  
## lstm_2 (LSTM)                (40, 50)              20200  
##  
## dense_1 (Dense)              (40, 1)               51  
## =====  
## Total params: 30,651  
## Trainable params: 30,651  
## Non-trainable params: 0  
##
```

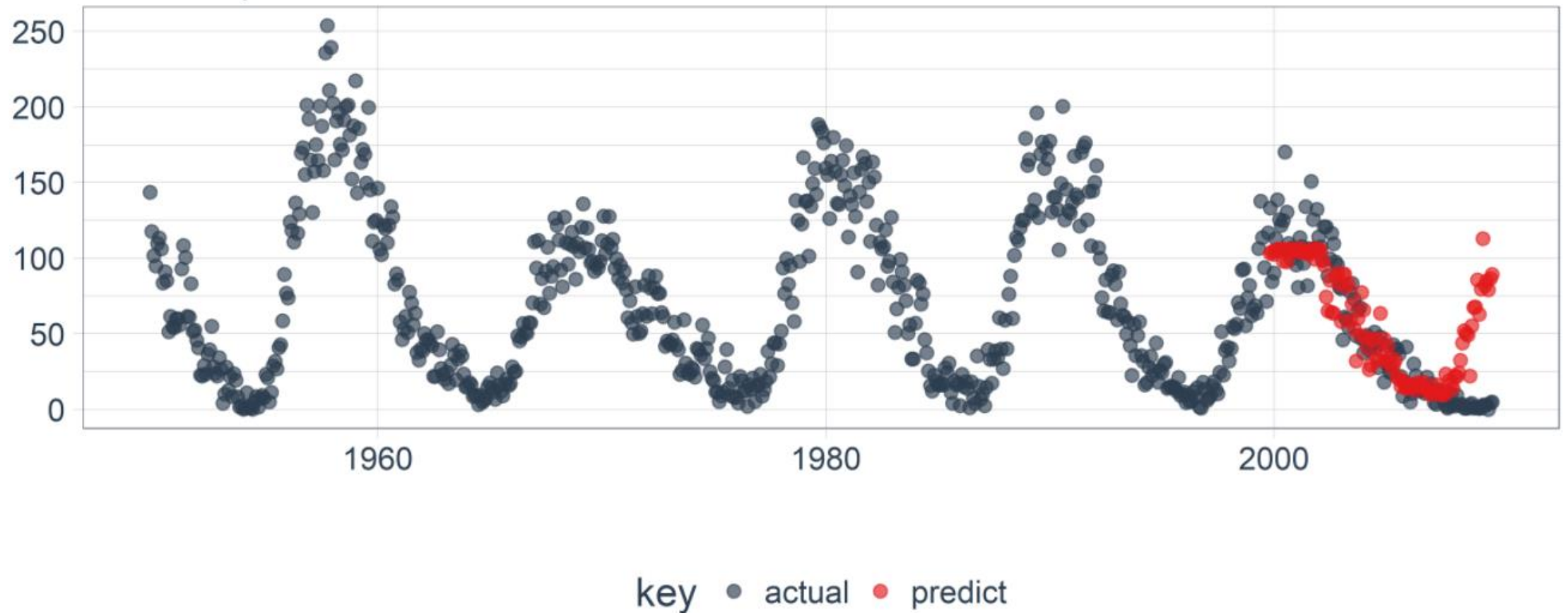
Fitting The Model

```
for (i in 1:epochs) {  
  model %>% fit(x      = x_train_arr,  
               y      = y_train_arr,  
               batch_size = batch_size,  
               epochs   = 1,  
               verbose   = 1,  
               shuffle   = FALSE)  
    
  model %>% reset_states()  
  cat("Epoch: ", i)  
}
```

Keras Stateful LSTM



Slice11, RMSE: 31.8





Backtesting

Backtesting Strategy



rsample::rolling_origin()

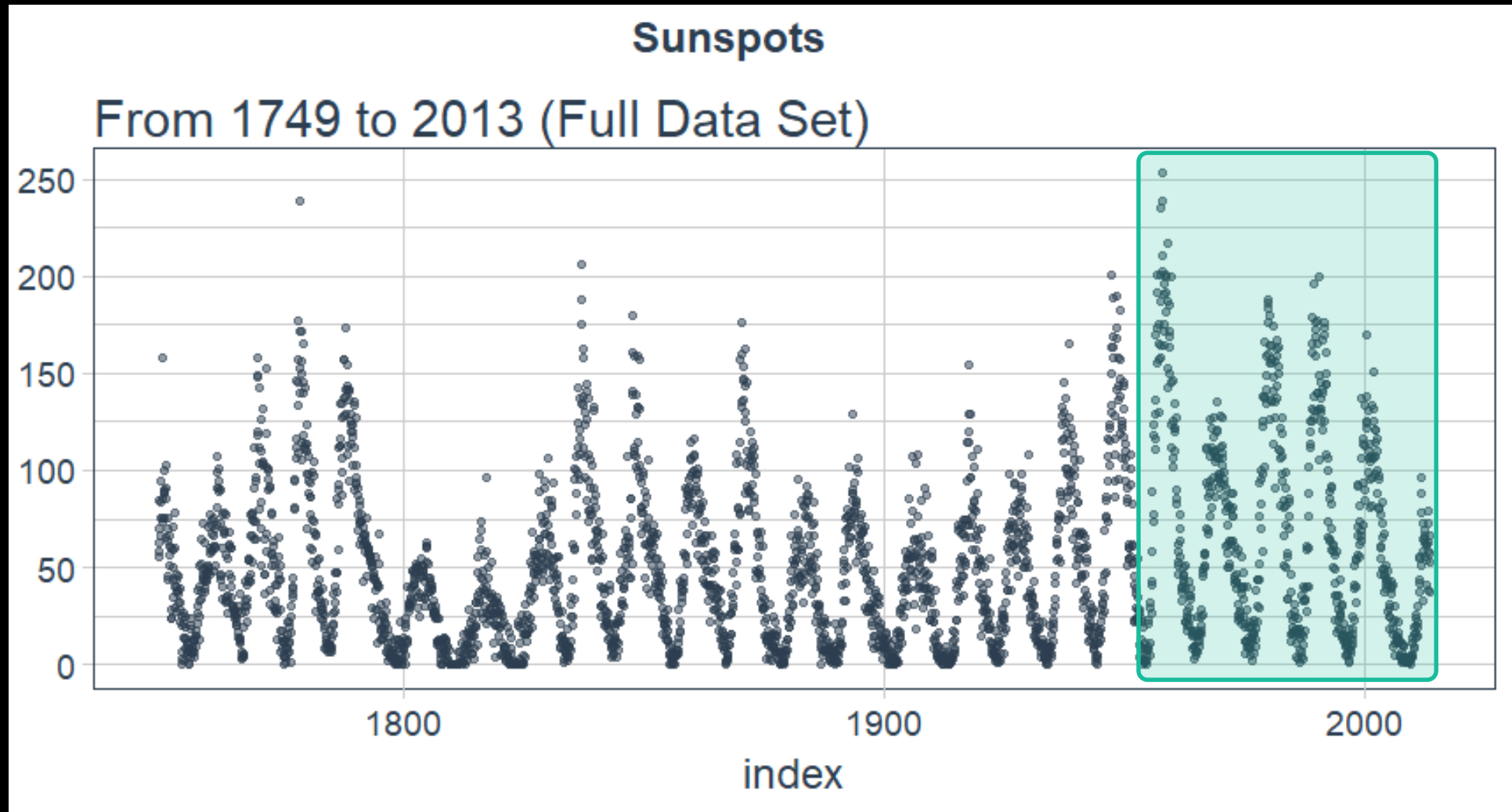
```
periods_train <- 12 * 50
periods_test  <- 12 * 10
skip_span     <- 12 * 20
```

```
rolling_origin_resamples <- rolling_origin(
  sun_spots,
  initial      = periods_train,
  assess      = periods_test,
  cumulative  = FALSE,
  skip        = skip_span
)
```

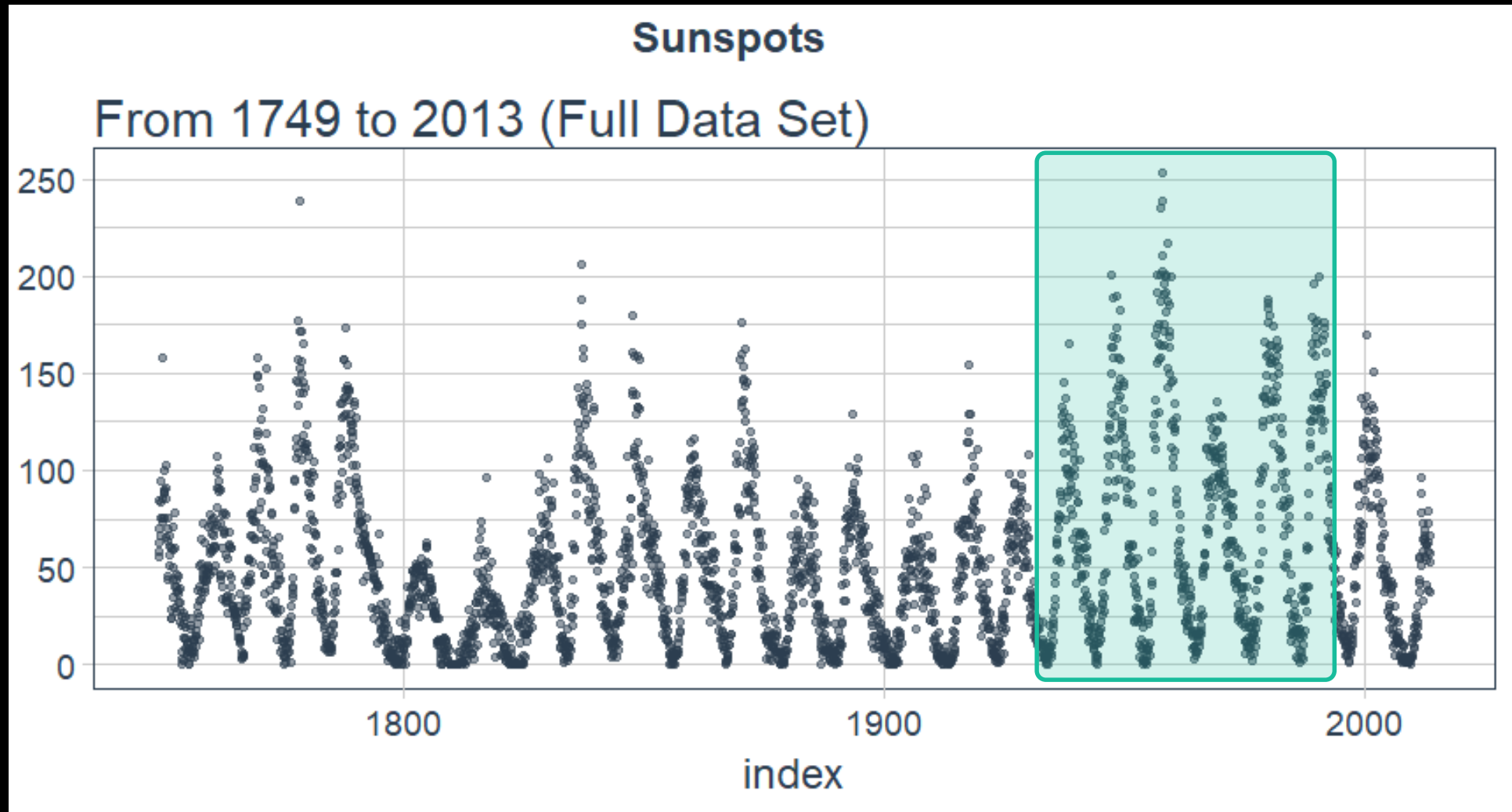
```
rolling_origin_resamples
```

```
## # Rolling origin forecast resampling
## # A tibble: 11 x 2
##   splits      id
##   <list>      <chr>
## 1 <S3: rsplit> Slice01
## 2 <S3: rsplit> Slice02
## 3 <S3: rsplit> Slice03
## 4 <S3: rsplit> Slice04
## 5 <S3: rsplit> Slice05
## 6 <S3: rsplit> Slice06
## 7 <S3: rsplit> Slice07
## 8 <S3: rsplit> Slice08
## 9 <S3: rsplit> Slice09
## 10 <S3: rsplit> Slice10
## 11 <S3: rsplit> Slice11
```

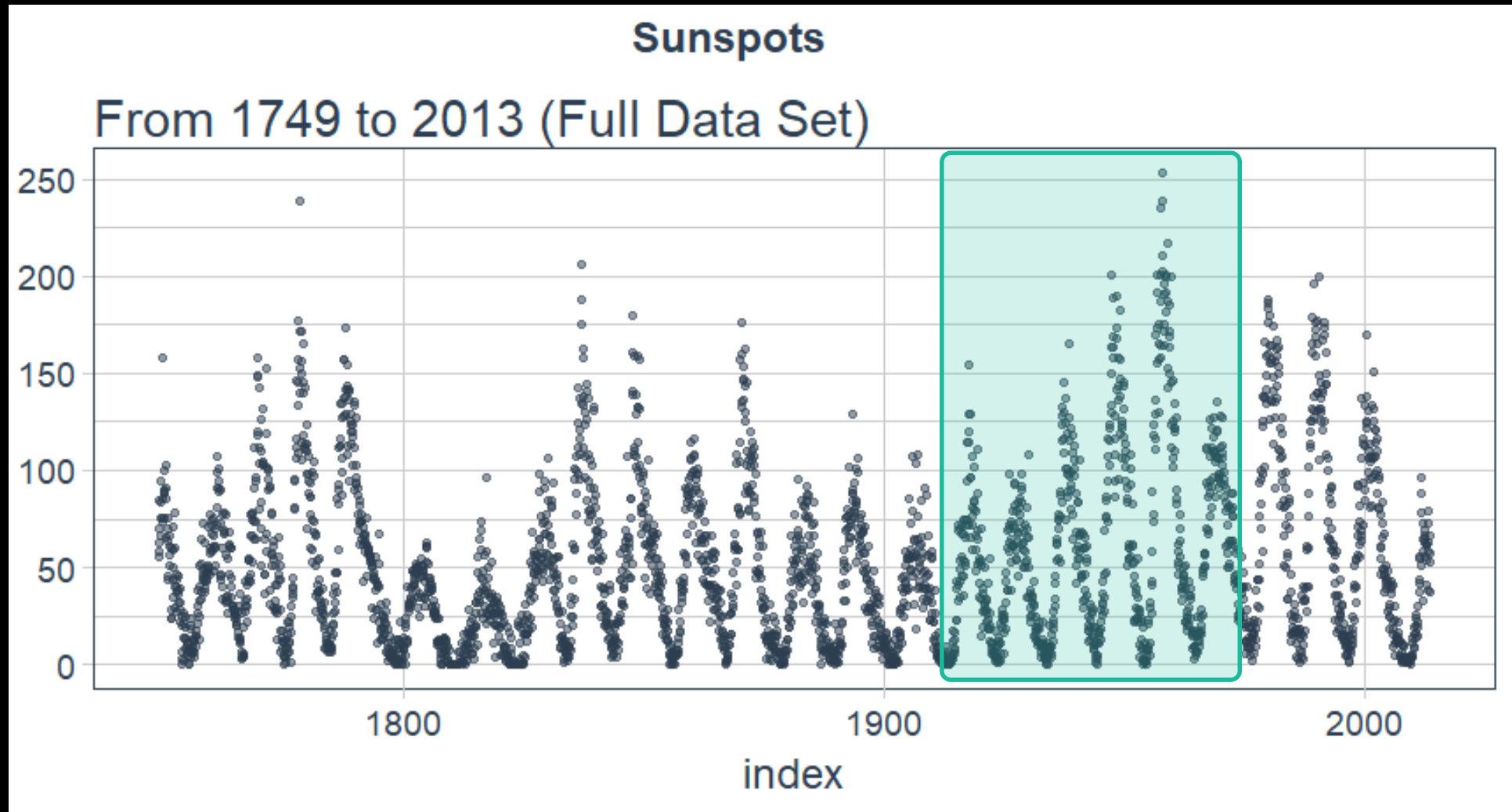
rsample::rolling_origin()



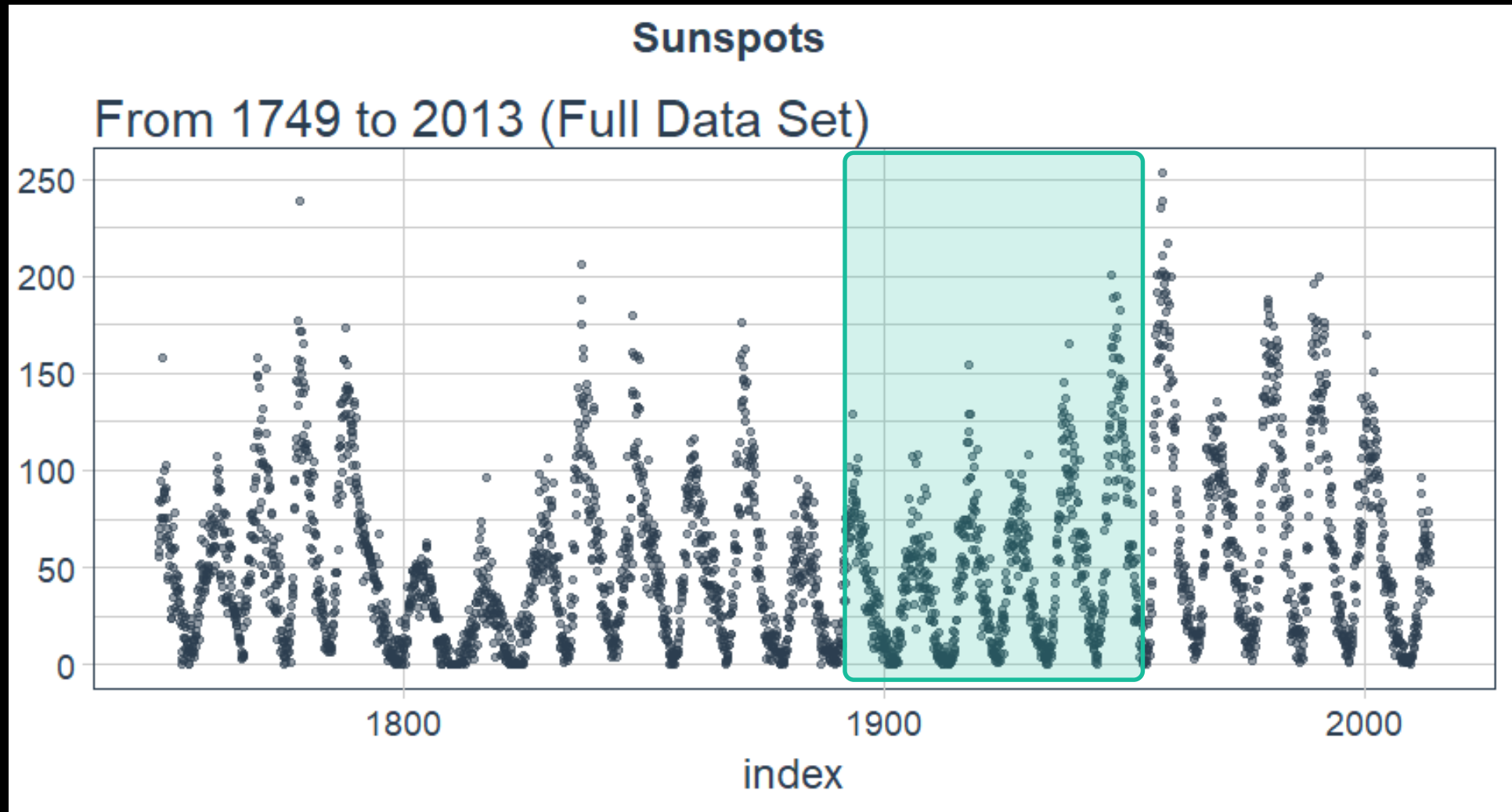
rsample::rolling_origin()



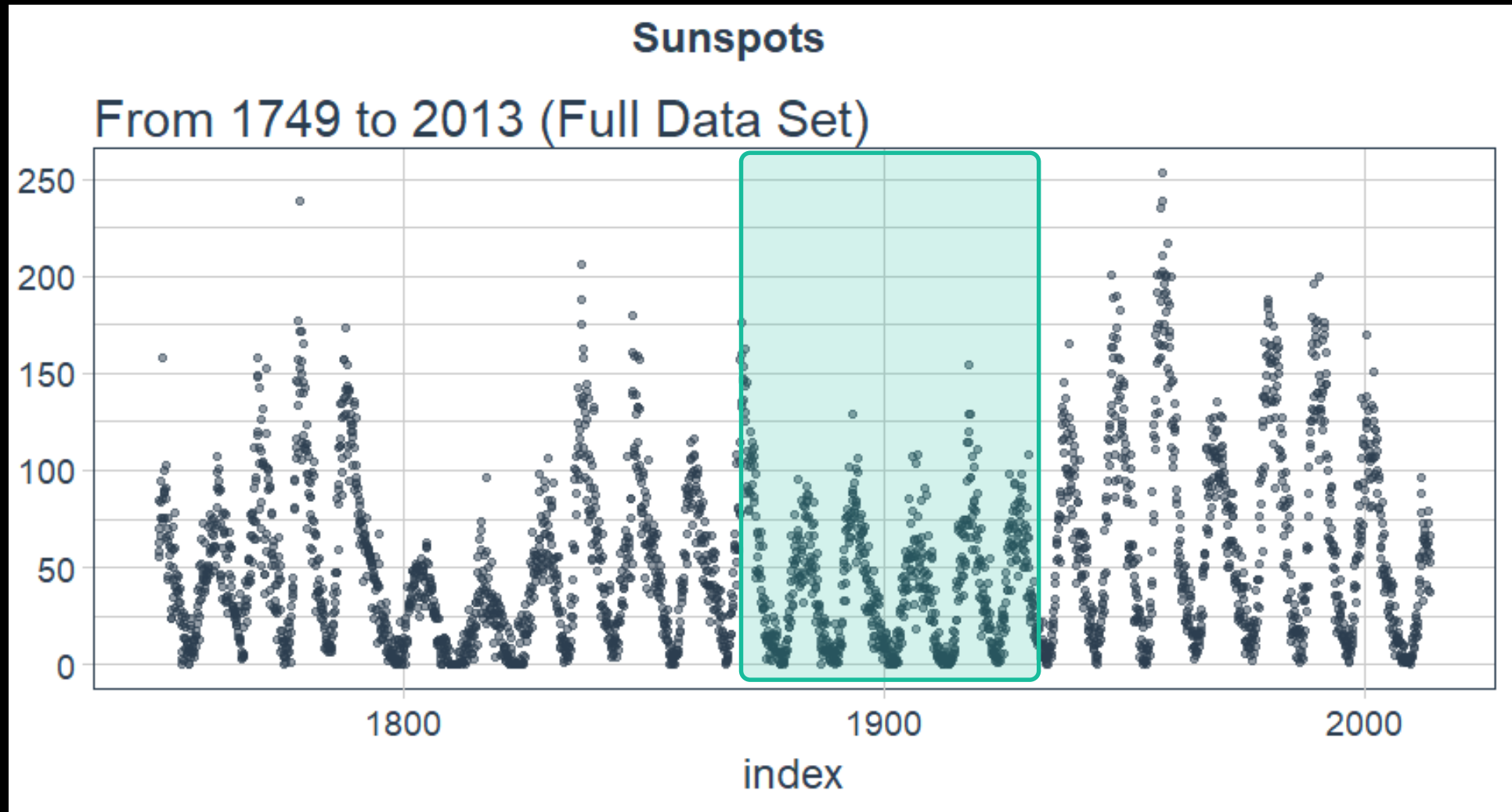
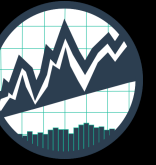
rsample::rolling_origin()



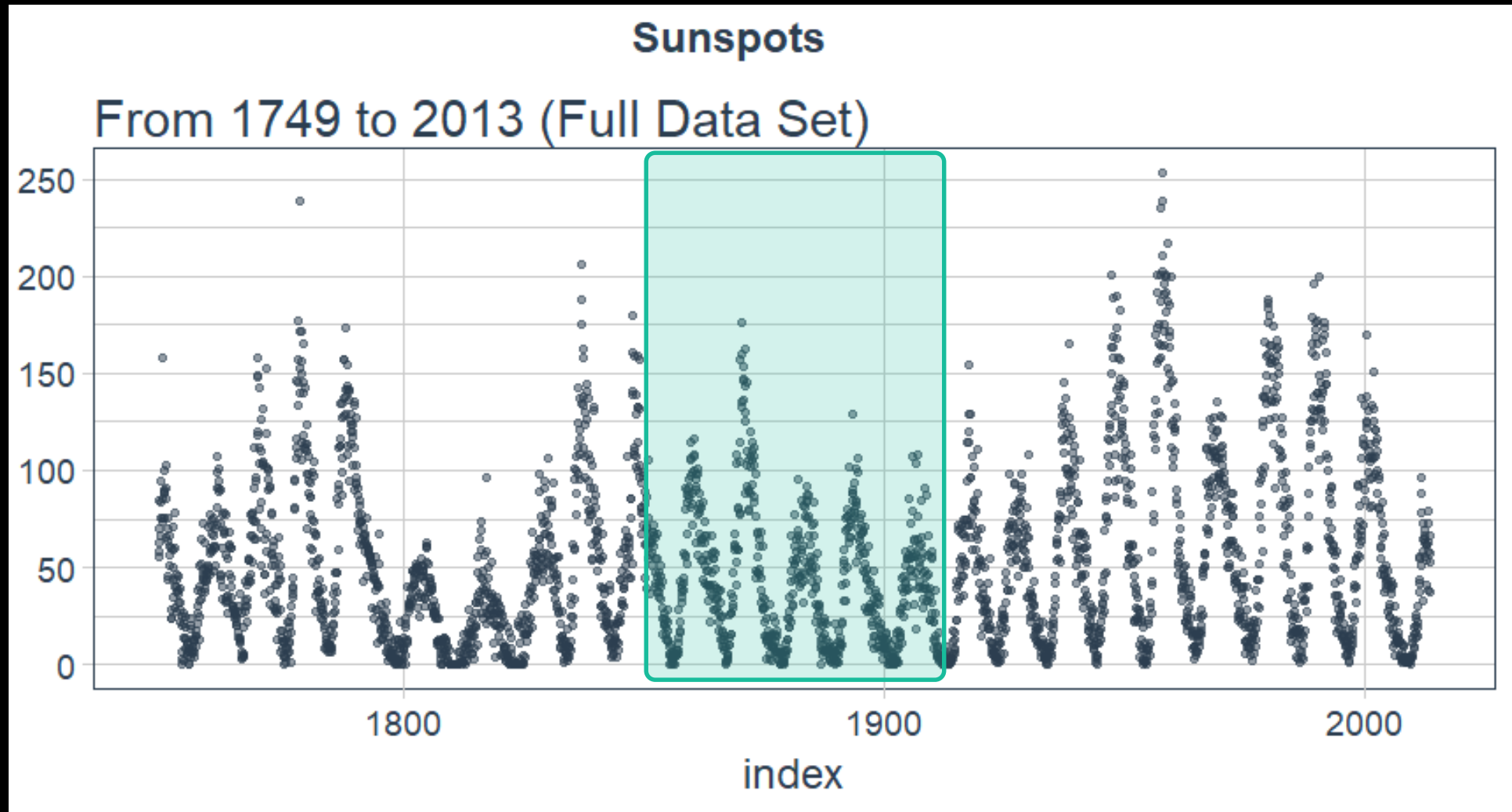
rsample::rolling_origin()



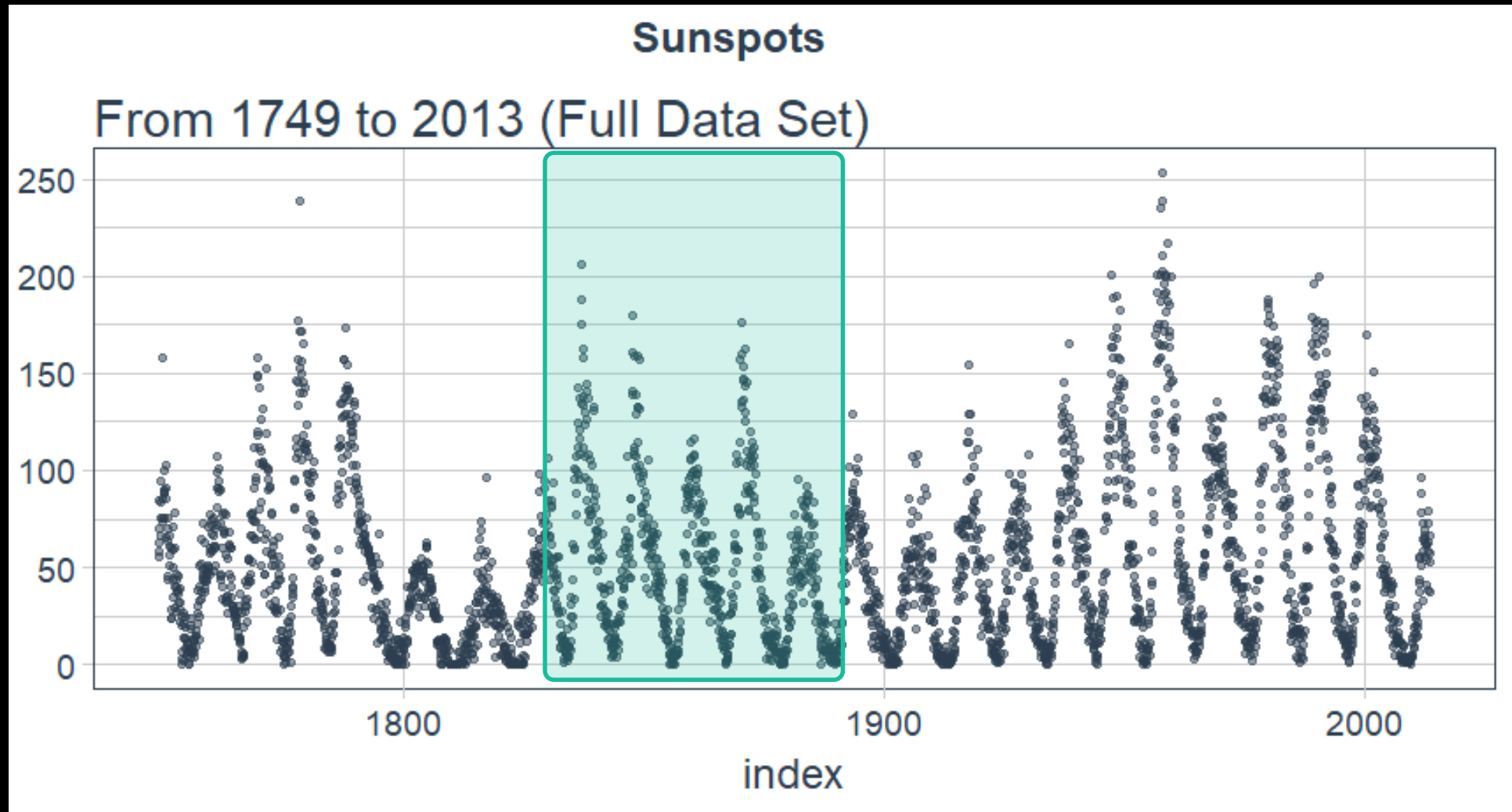
rsample::rolling_origin()



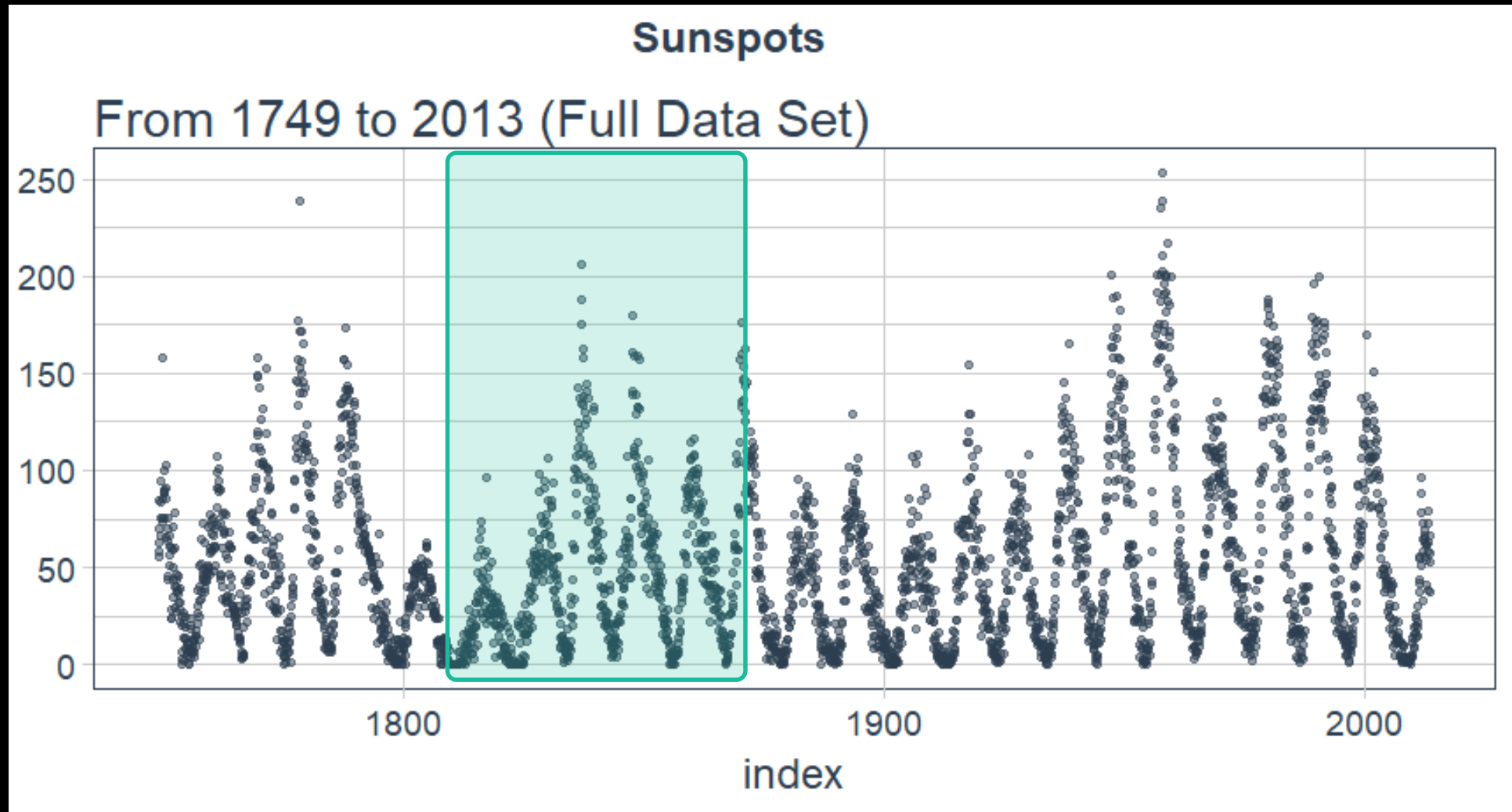
rsample::rolling_origin()



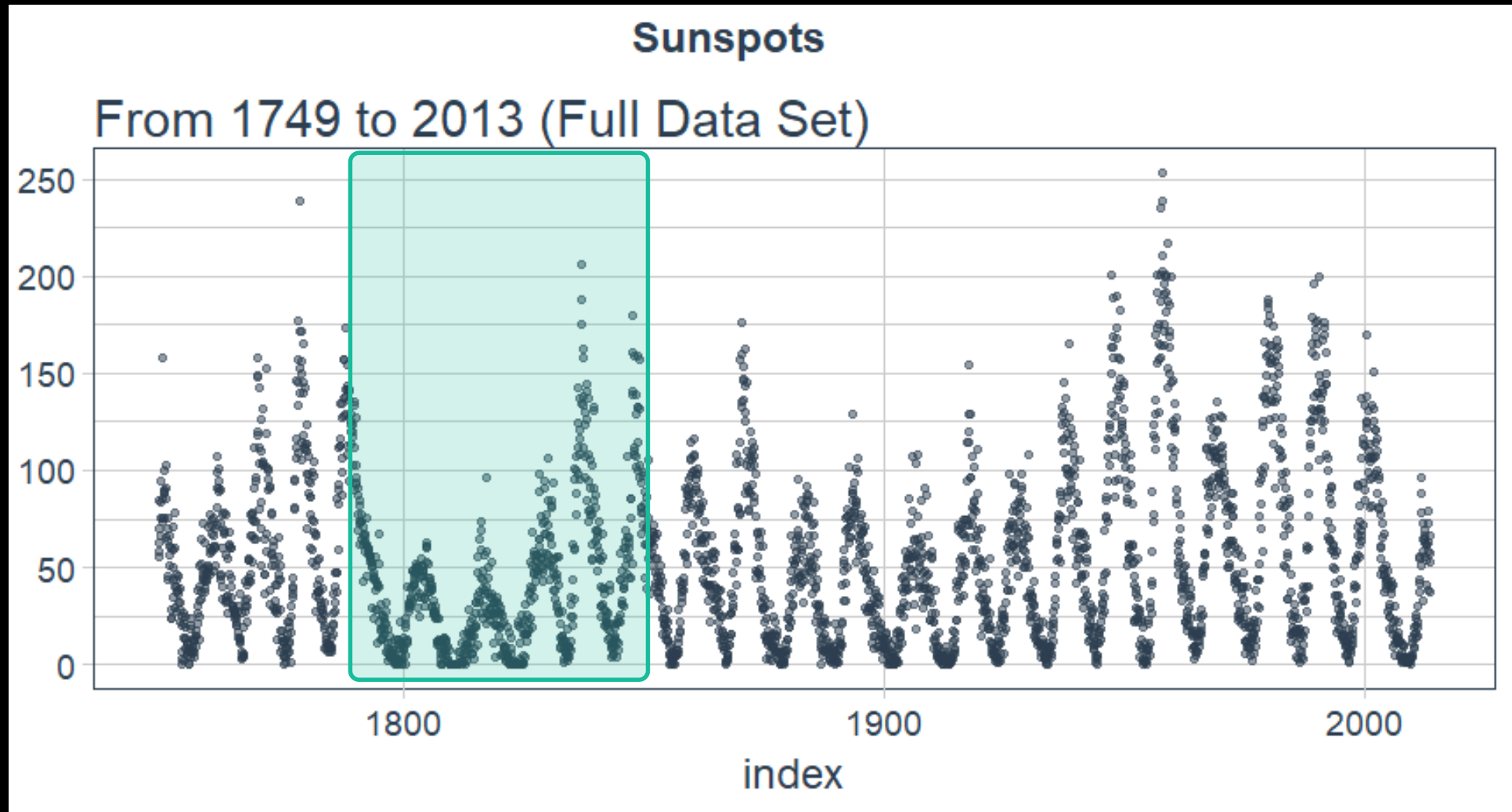
rsample::rolling_origin()



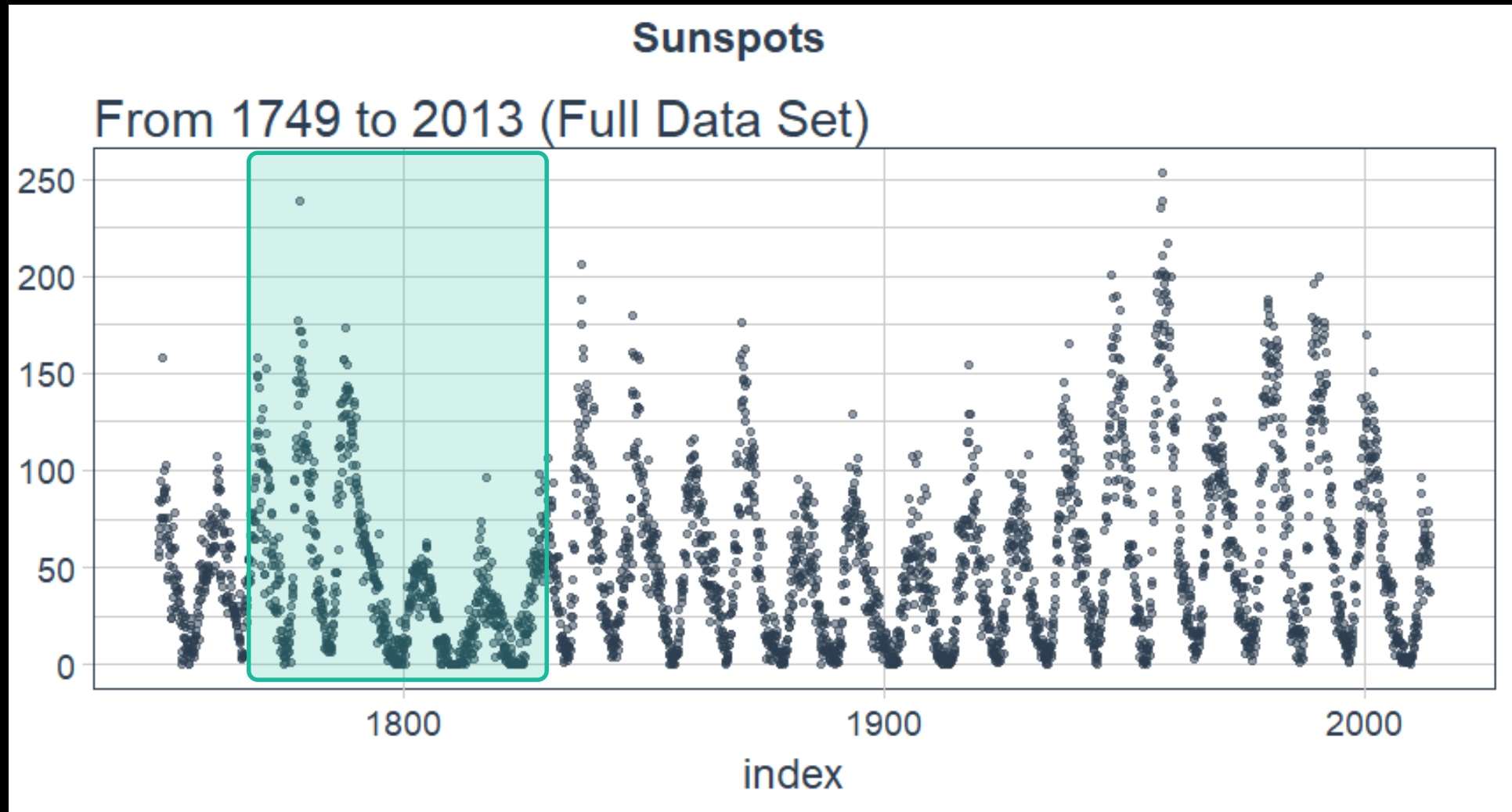
rsample::rolling_origin()



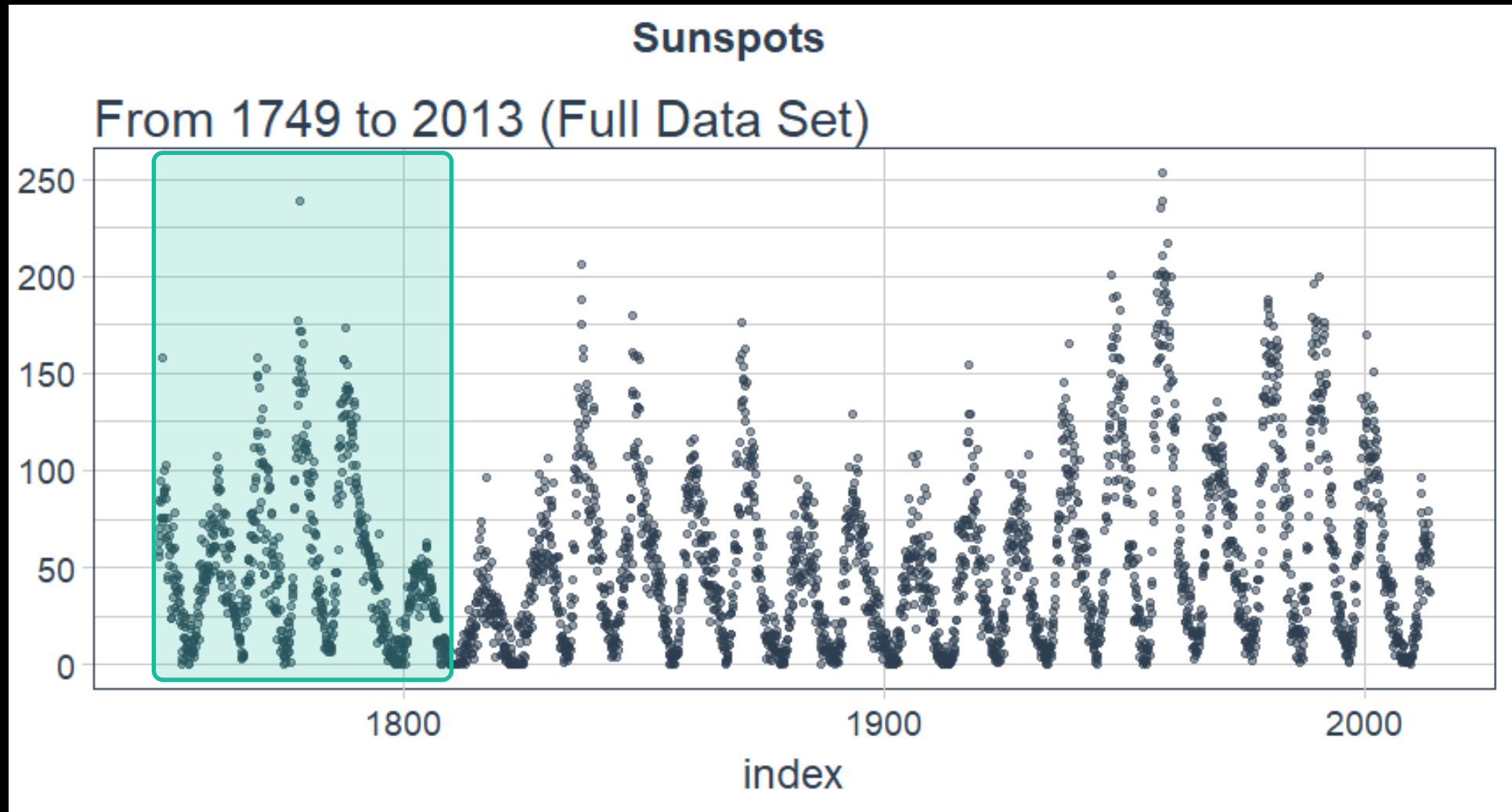
rsample::rolling_origin()



rsample::rolling_origin()



rsample::rolling_origin()



Backtesting Strategy

- `rolling_origin()`
- 1 Time Series
- 11 Samples →

Backtesting Strategy: Zoomed In



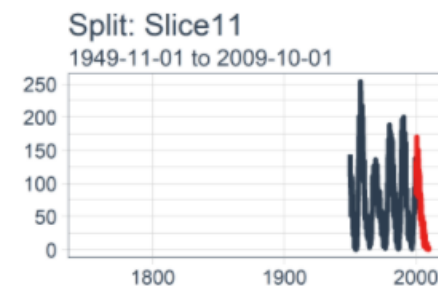
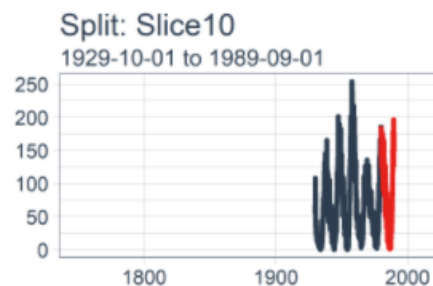
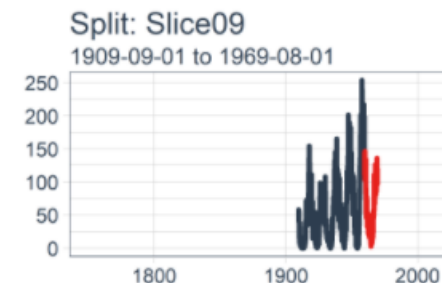
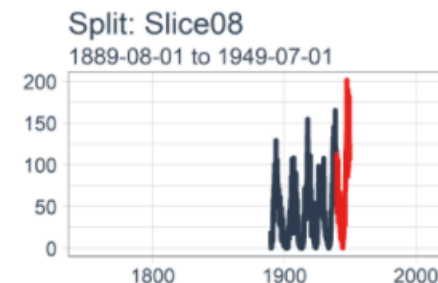
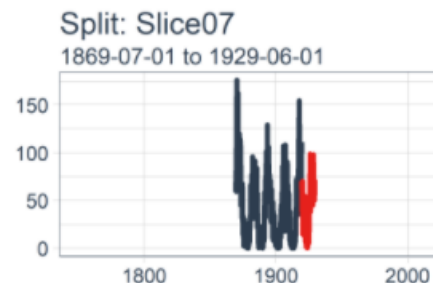
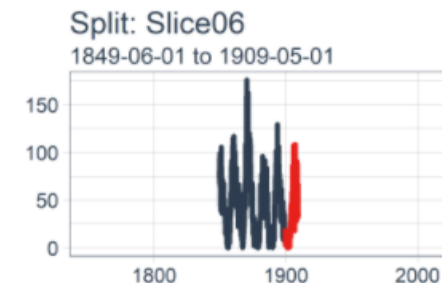
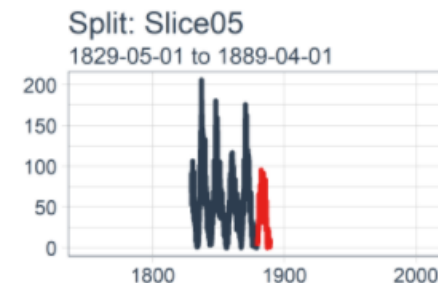
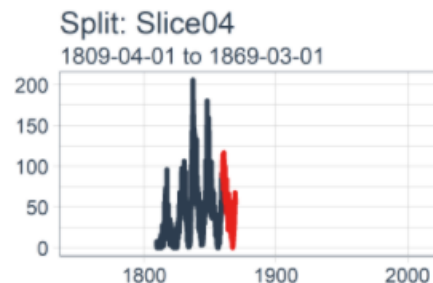
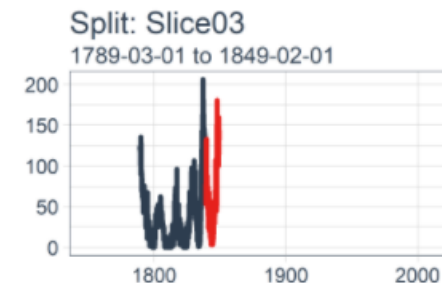
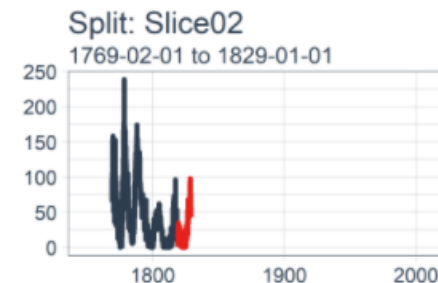
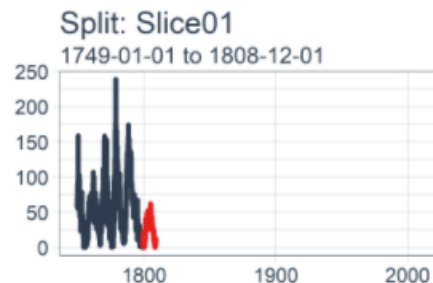
Key:
Training
Testing

Backtesting Strategy

Zoomed Out

Train: 50 years
Test: 10 years
Gap: 20 years

Backtesting Strategy: Rolling Origin Sampling Plan



key — training — testing

Key:
Training
Testing



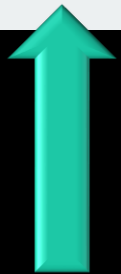
Results

Keras LSTM Results



map(): custom keras prediction function

```
sample_predictions_lstm_tbl <- rolling_origin_resamples %>%  
  mutate(predict = map(splits, predict_keras_lstm, epochs = 300))
```



Keras LSTM Results



map(): custom keras prediction function

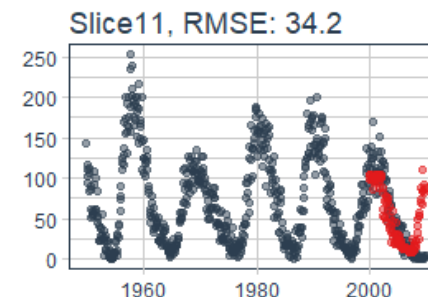
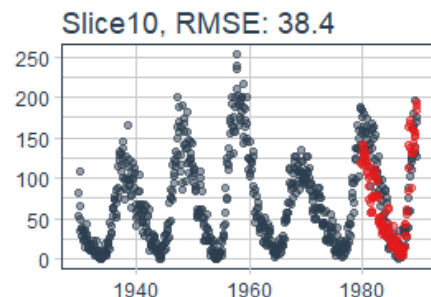
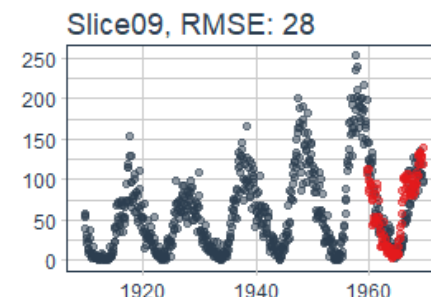
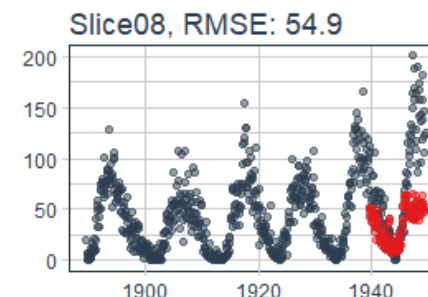
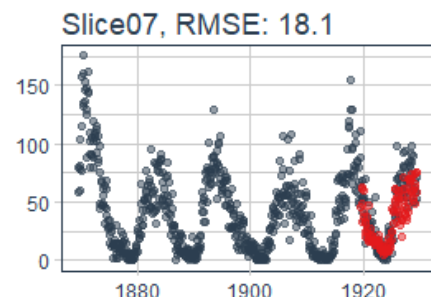
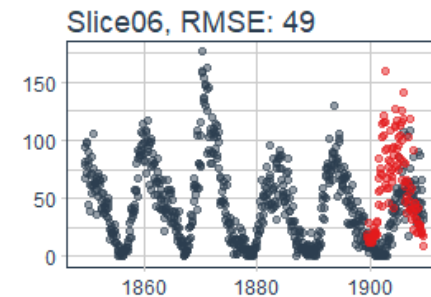
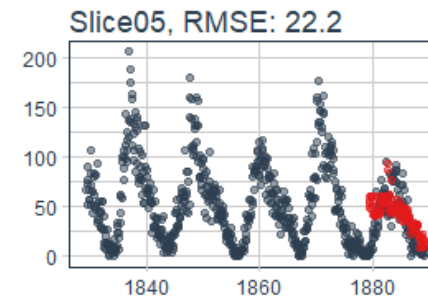
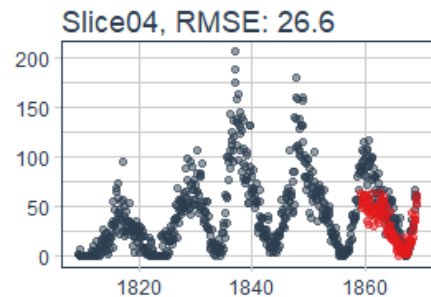
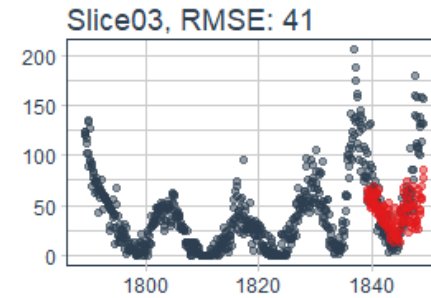
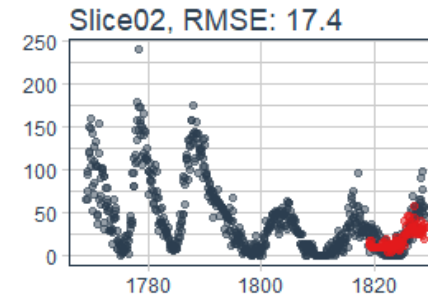
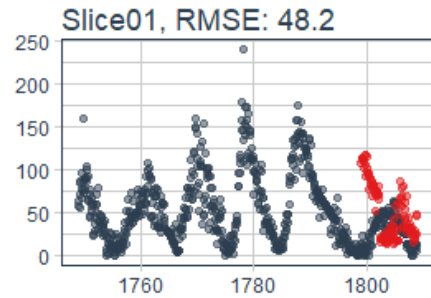
```
## # Rolling origin forecast resampling
## # A tibble: 11 x 3
##   splits      id predict
## * <list>    <chr> <list>
## 1 <S3: rsplit> Slice01 <tibble [840 x 3]>
## 2 <S3: rsplit> Slice02 <tibble [840 x 3]>
## 3 <S3: rsplit> Slice03 <tibble [840 x 3]>
## 4 <S3: rsplit> Slice04 <tibble [840 x 3]>
## 5 <S3: rsplit> Slice05 <tibble [840 x 3]>
## 6 <S3: rsplit> Slice06 <tibble [840 x 3]>
## 7 <S3: rsplit> Slice07 <tibble [840 x 3]>
## 8 <S3: rsplit> Slice08 <tibble [840 x 3]>
## 9 <S3: rsplit> Slice09 <tibble [840 x 3]>
## 10 <S3: rsplit> Slice10 <tibble [840 x 3]>
## 11 <S3: rsplit> Slice11 <tibble [840 x 3]>
```

Keras LSTM Results

yardstick::rmse()

```
## # Rolling origin forecast resampling
## # A tibble: 11 x 2
##   id      rmse
##   * <chr>   <dbl>
## 1 Slice01  48.2
## 2 Slice02  17.4
## 3 Slice03  41.0
## 4 Slice04  26.6
## 5 Slice05  22.2
## 6 Slice06  49.0
## 7 Slice07  18.1
## 8 Slice08  54.9
## 9 Slice09  28.0
## 10 Slice10 38.4
## 11 Slice11 34.2
```

Keras Stateful LSTM: Backtested Predictions, Rolling Origin



Technique Comparison



ARIMA

Prophet

Keras LSTM

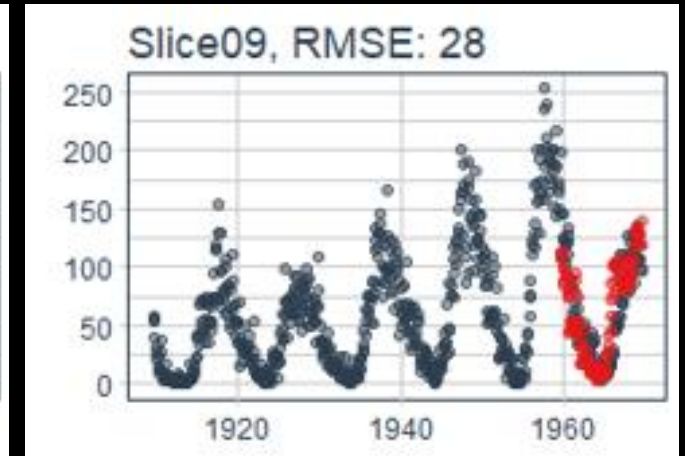
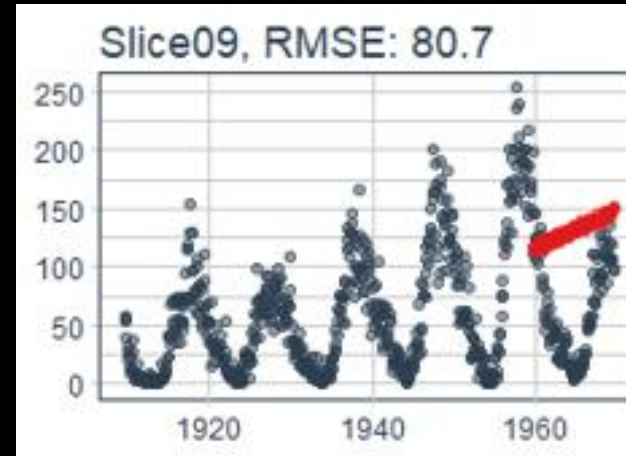
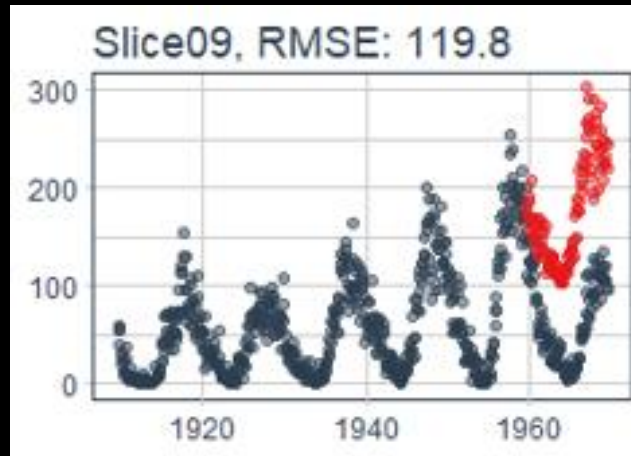
**RMSE
(Average)**

50.5

45.3

34.4

**Typical Model
(Slice09)**

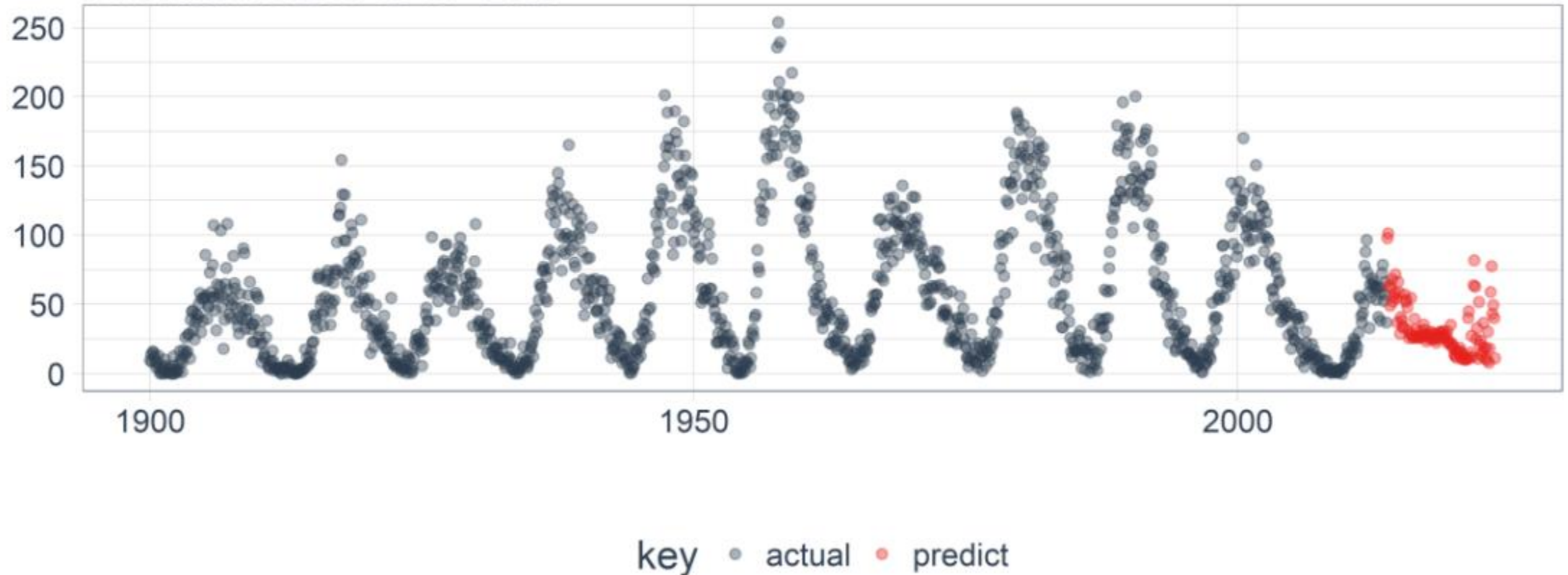


Keras Stateful LSTM: 32% Lower RMSE

Forecasting Next 10 Years



Sunspots: Ten Year Forecast
Forecast Horizon: 2013 - 2023



Conclusions



- Built a Stateful LSTM with Keras using Lag 120
- Benchmarked by Backtesting
- Compared to other techniques, 32% better RMSE
- Good Tool-Application Fit

Code Available



Business Science Blog



[SERVICES](#) [UNIVERSITY](#) [SOFTWARE](#) [BLOG](#) [ABOUT](#) [CONTACT](#)



TIME SERIES DEEP LEARNING: FORECASTING SUNSPOTS WITH KERAS STATEFUL LSTM IN R

Written by Matt Dancho on April 18, 2018

Categories: Timeseries-Analysis

Tags: R-Project, R, Time Series, Deep Learning, Keras, TensorFlow, Backtesting, tidyverse, tibbletime, timetk, keras, rsample, recipes, yardstick

business-science.io/blog

Presentation Available



Business Science GitHub



This repository

Search

Pull requests


Issues

Marketplace

Explore



 [business-science](#) / [presentations](#)

 Unwatch ▾

8

★ Unstar

17

 Fork

16

↔ Code

⚠ Issues 0

🔗 Pull requests 0

📁 Projects 0

📖 Wiki

📊 Insights

⚙ Settings

A central repository of Business Science presentations <http://www.business-science.io/>

Edit

[business-science](#)

[tidyquant](#)

[timetk](#)

[sweep](#)

[tibbletime](#)

[Manage topics](#)

<https://github.com/business-science/presentations>

Business + Data Science



business-science.io