Copyright Notice

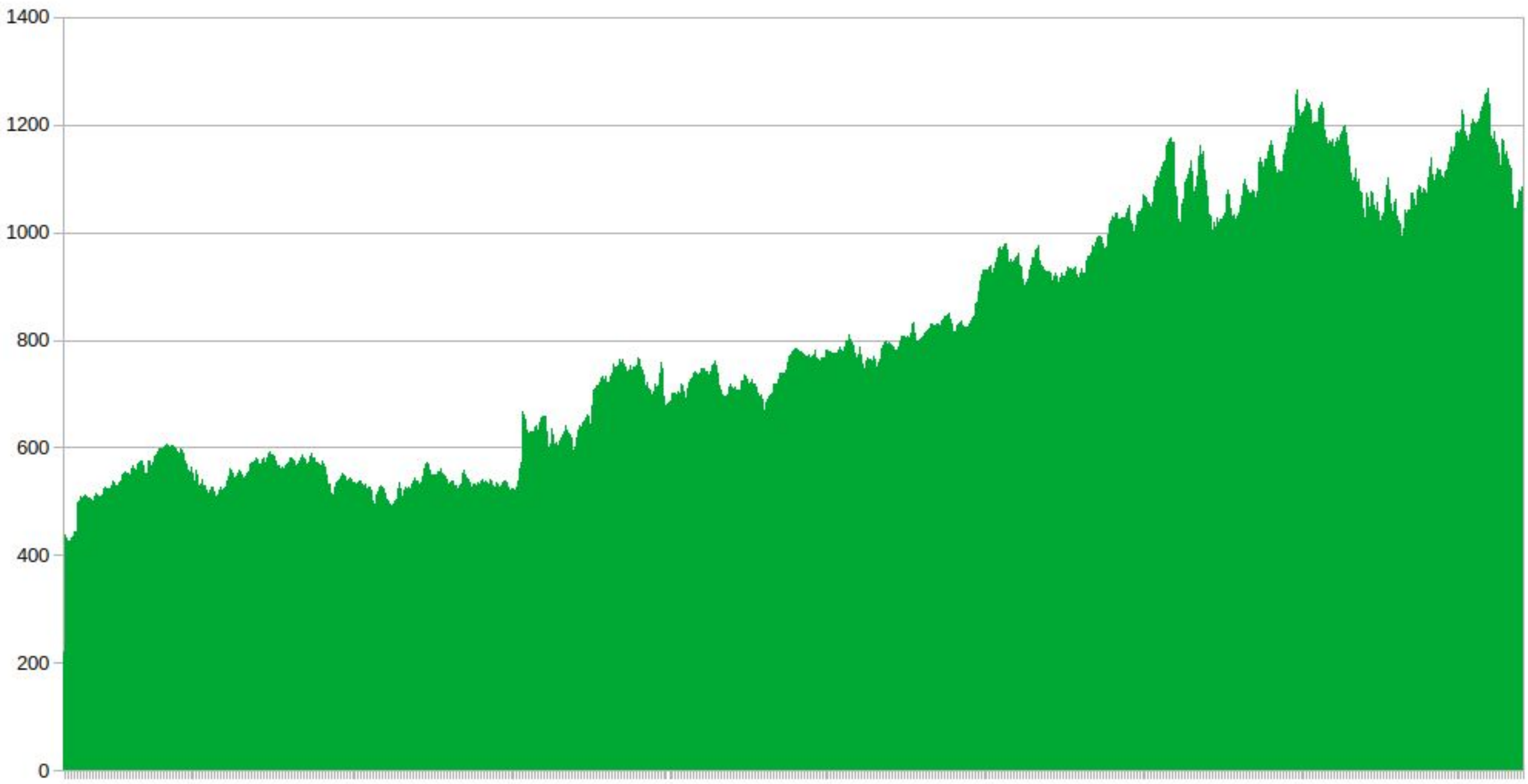


Chart Created by lmoroney@

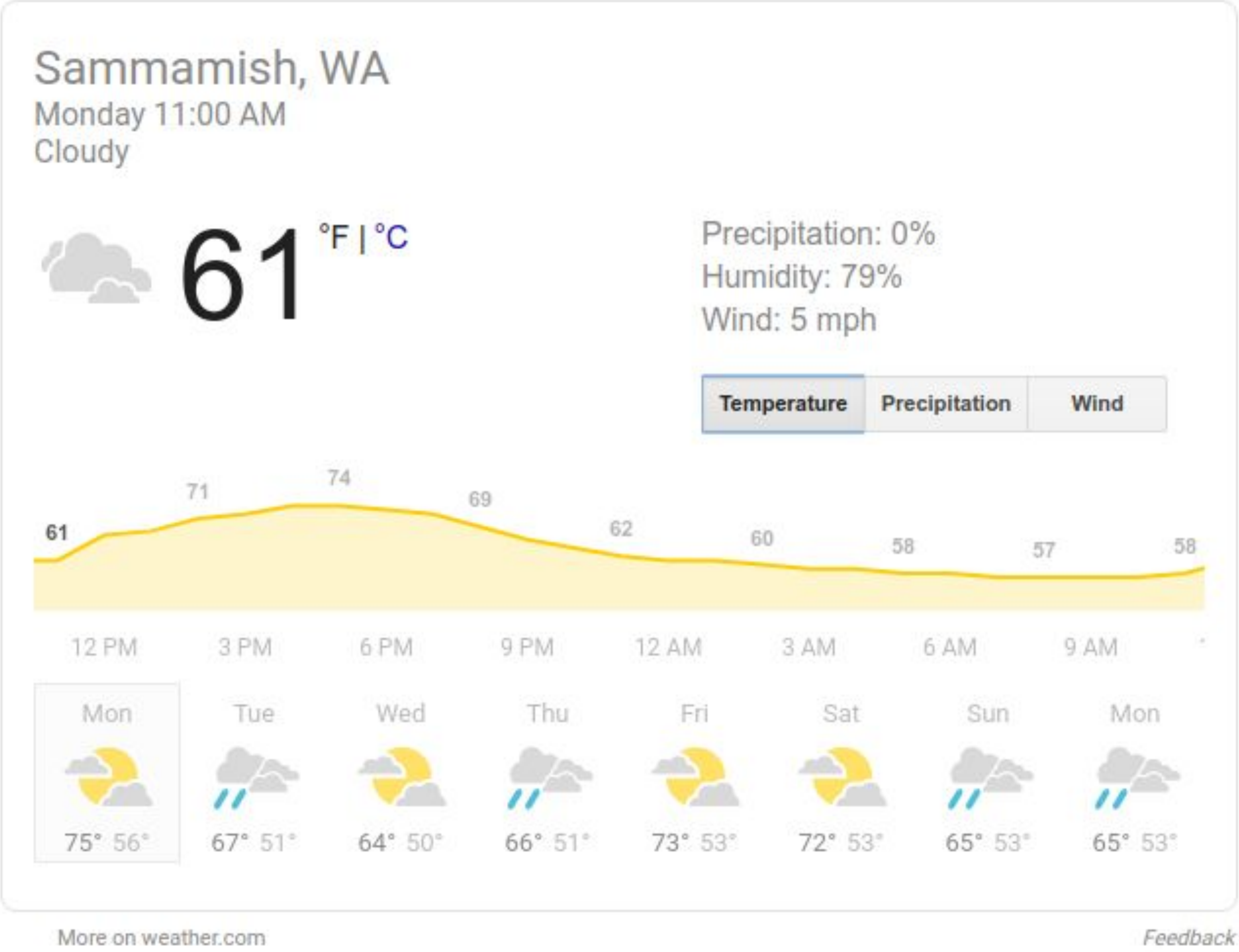


Chart From Google Search

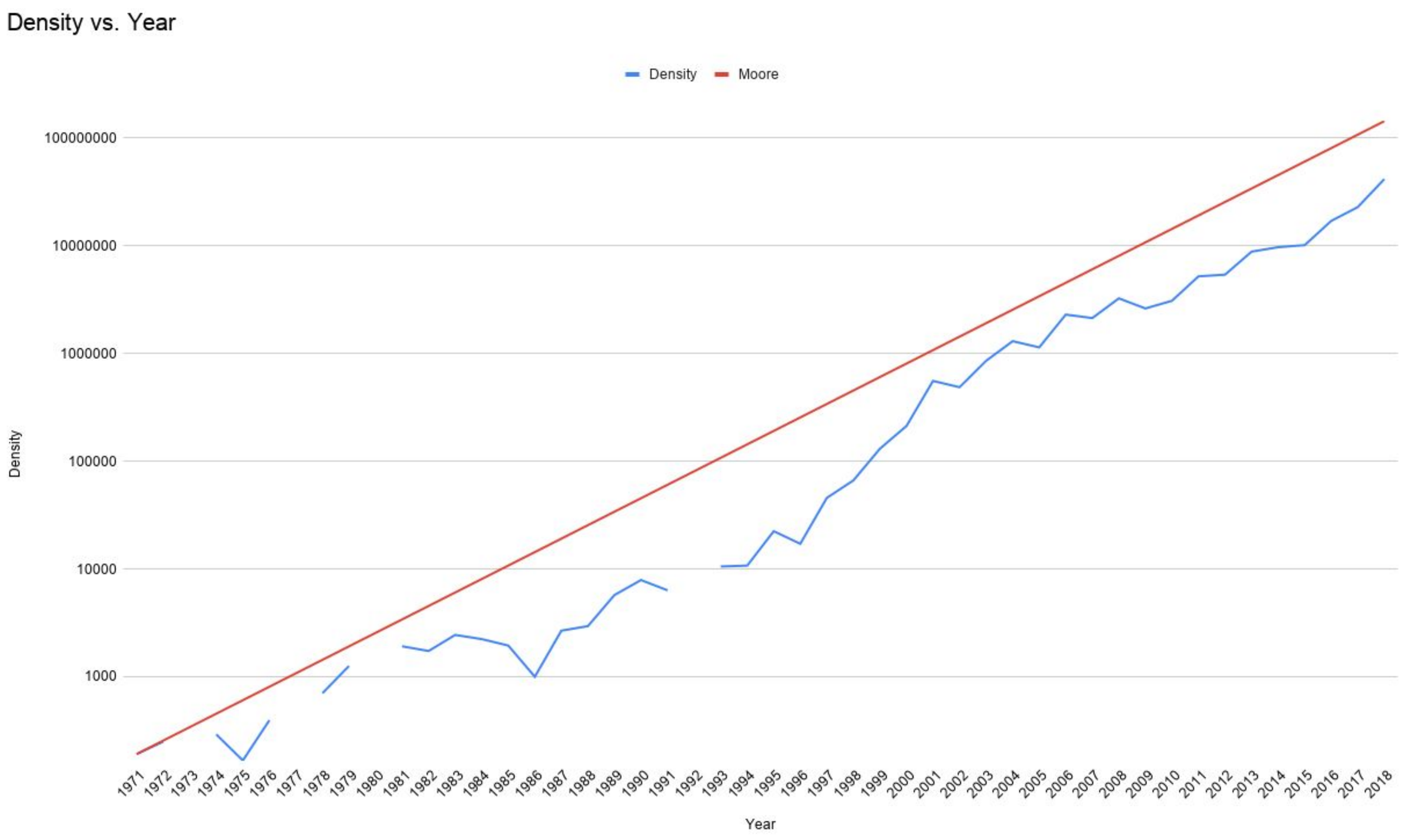
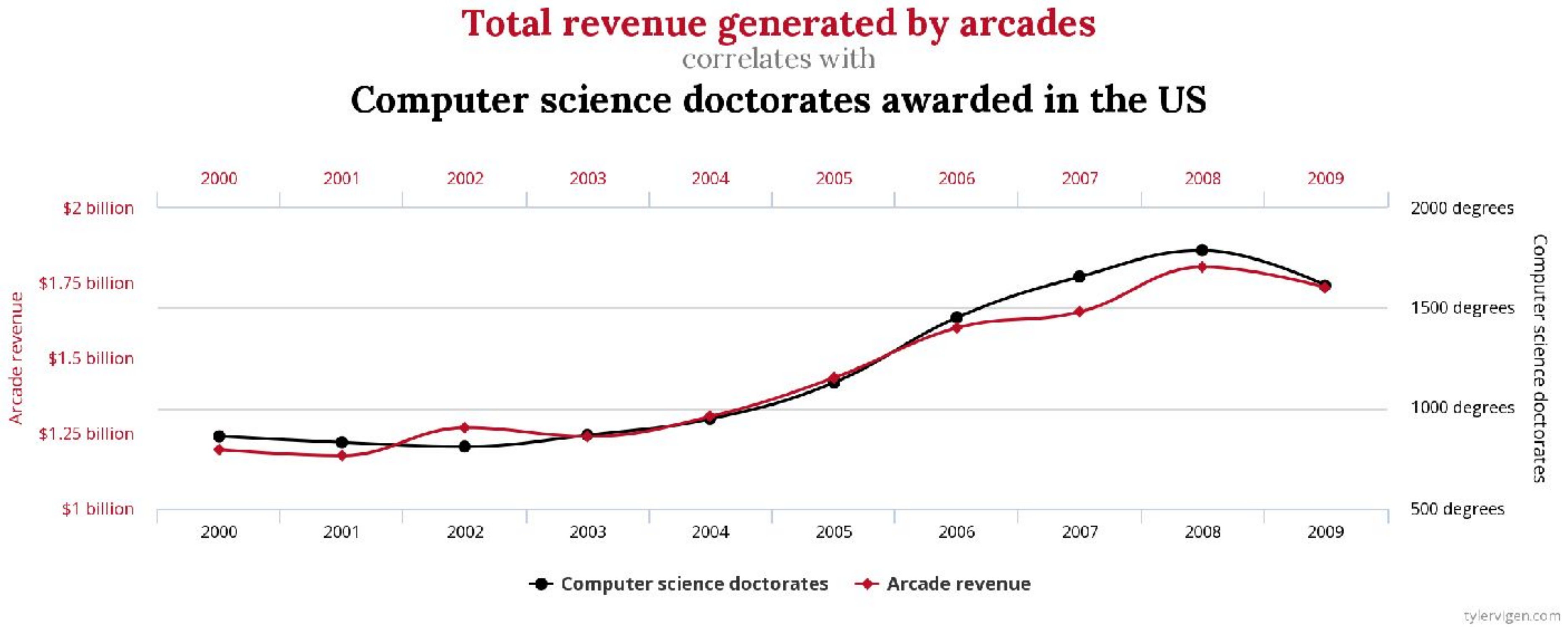
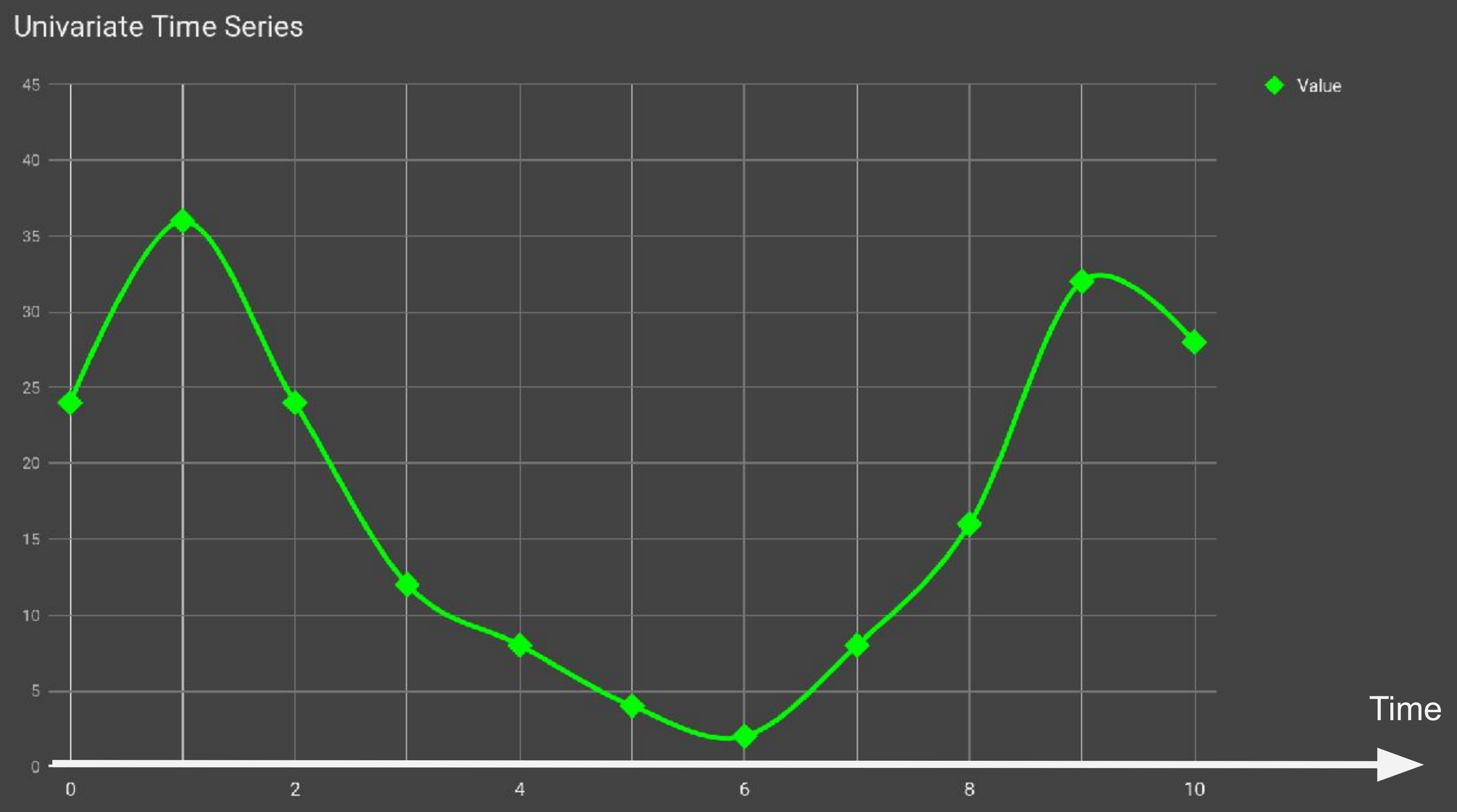


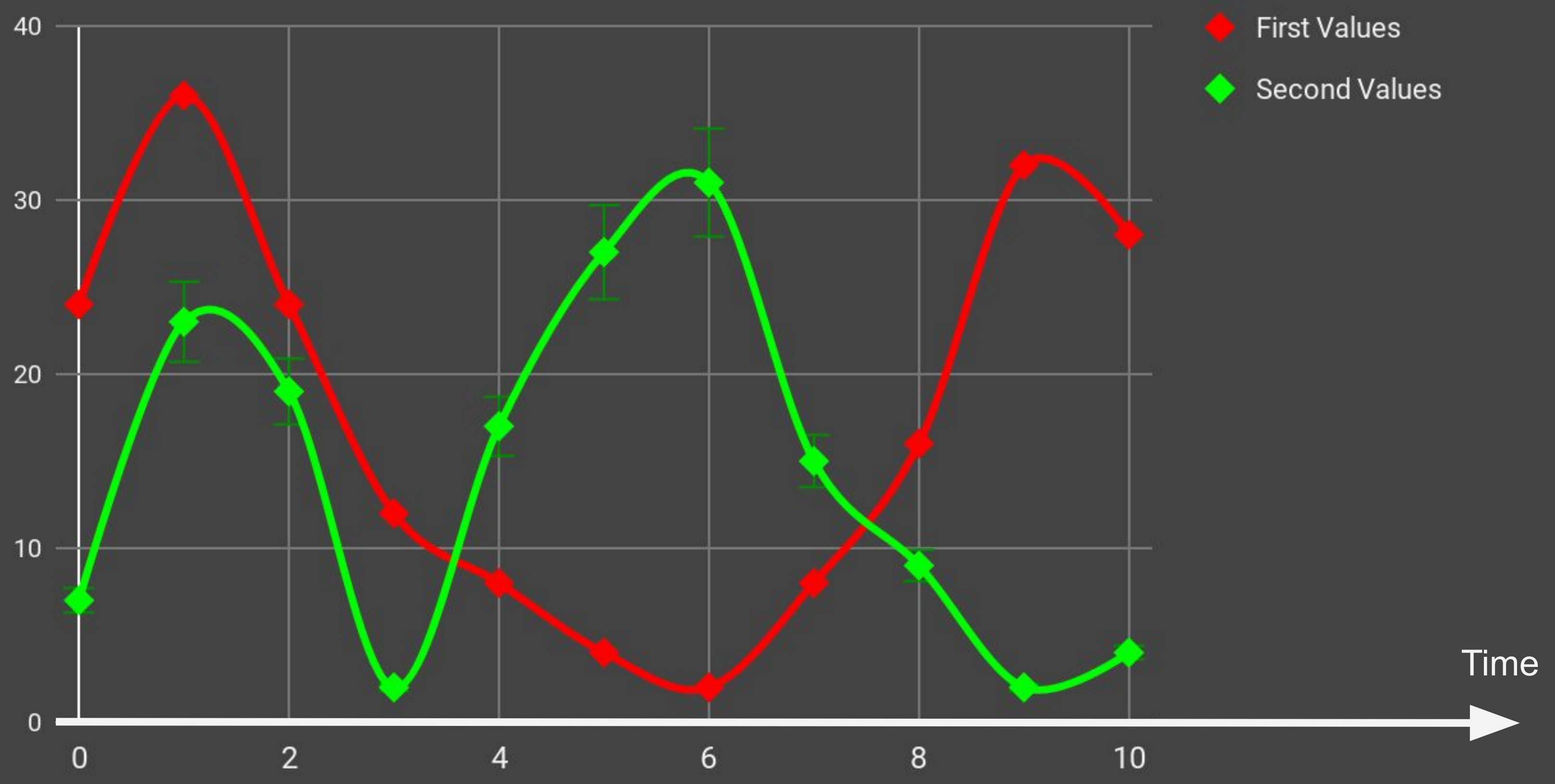
Chart Created by lmoroney@


Total revenue generated by arcades
correlates with
Computer science doctorates awarded in the US
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
$2 billion
$1.75 billion
$1.5 billion
$1.25 billion
$1 billion
Arcade revenue
2000 degrees
1500 degrees
1000 degrees
500 degrees
Computer science doctorates
Computer science doctorates
Arcade revenue
tylervigen.com

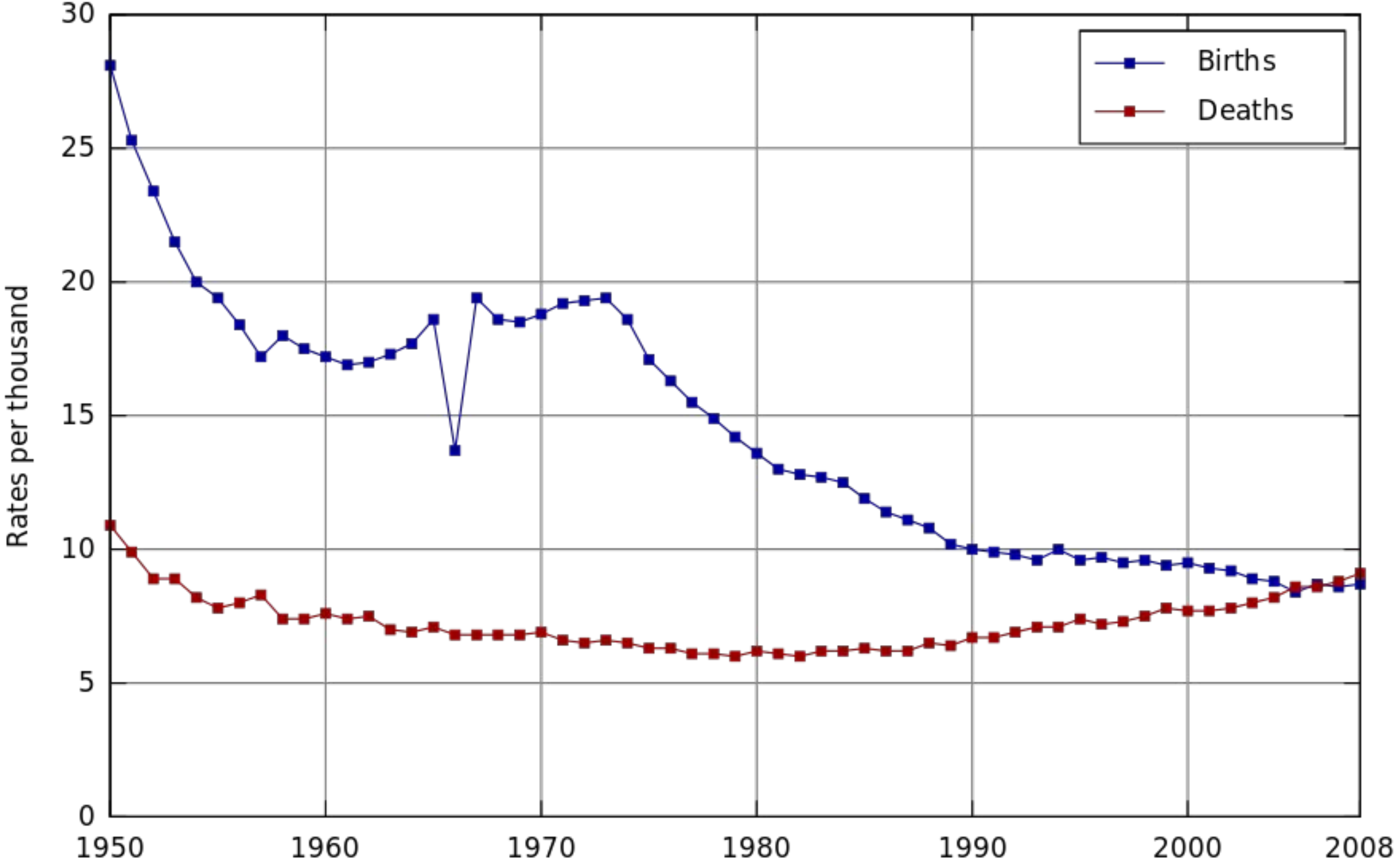


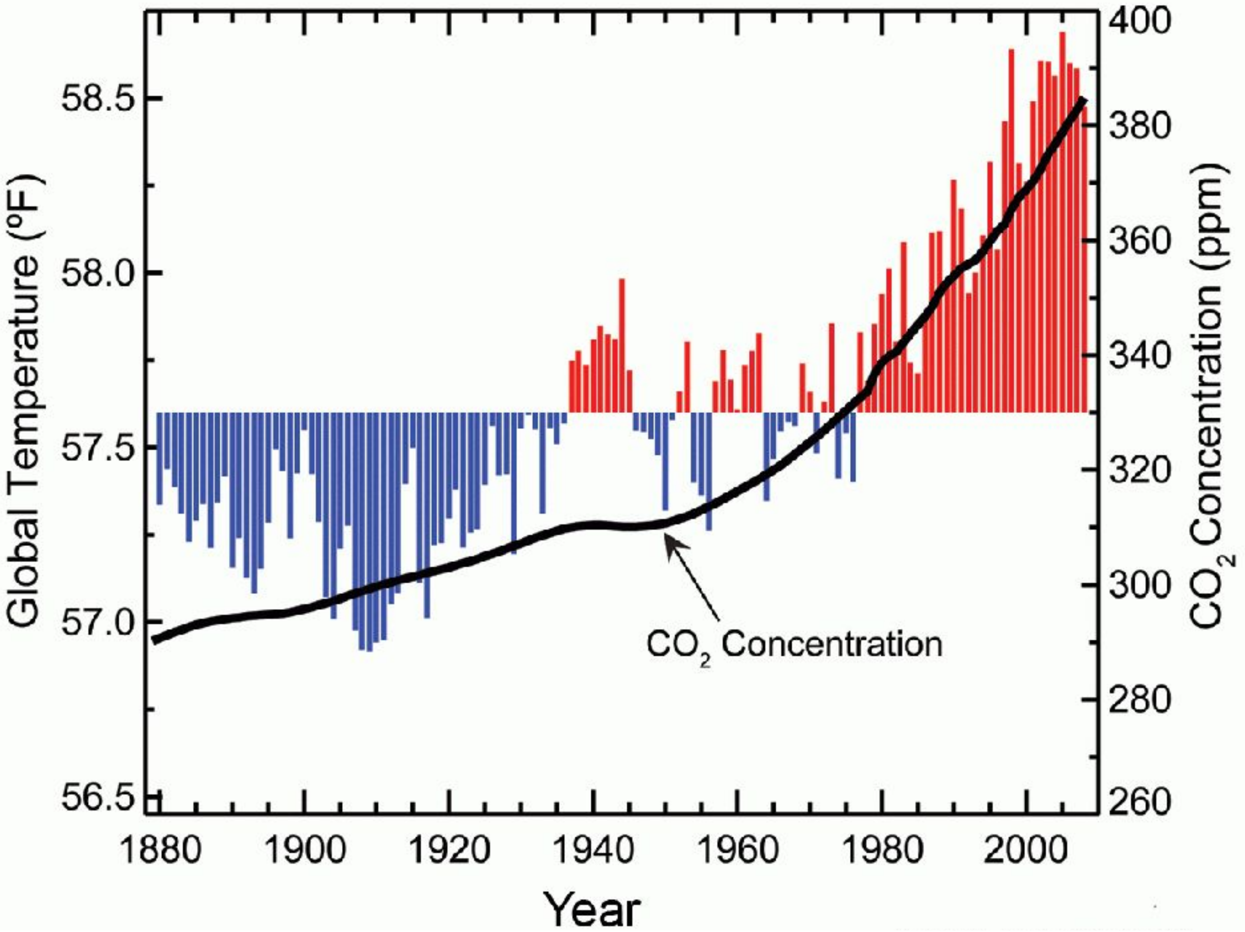
Univariate Time Series
Value
Time
45
40
35
30
25
20
15
10
5
0
0
2
4
6
8
10

Multivariate Time Series

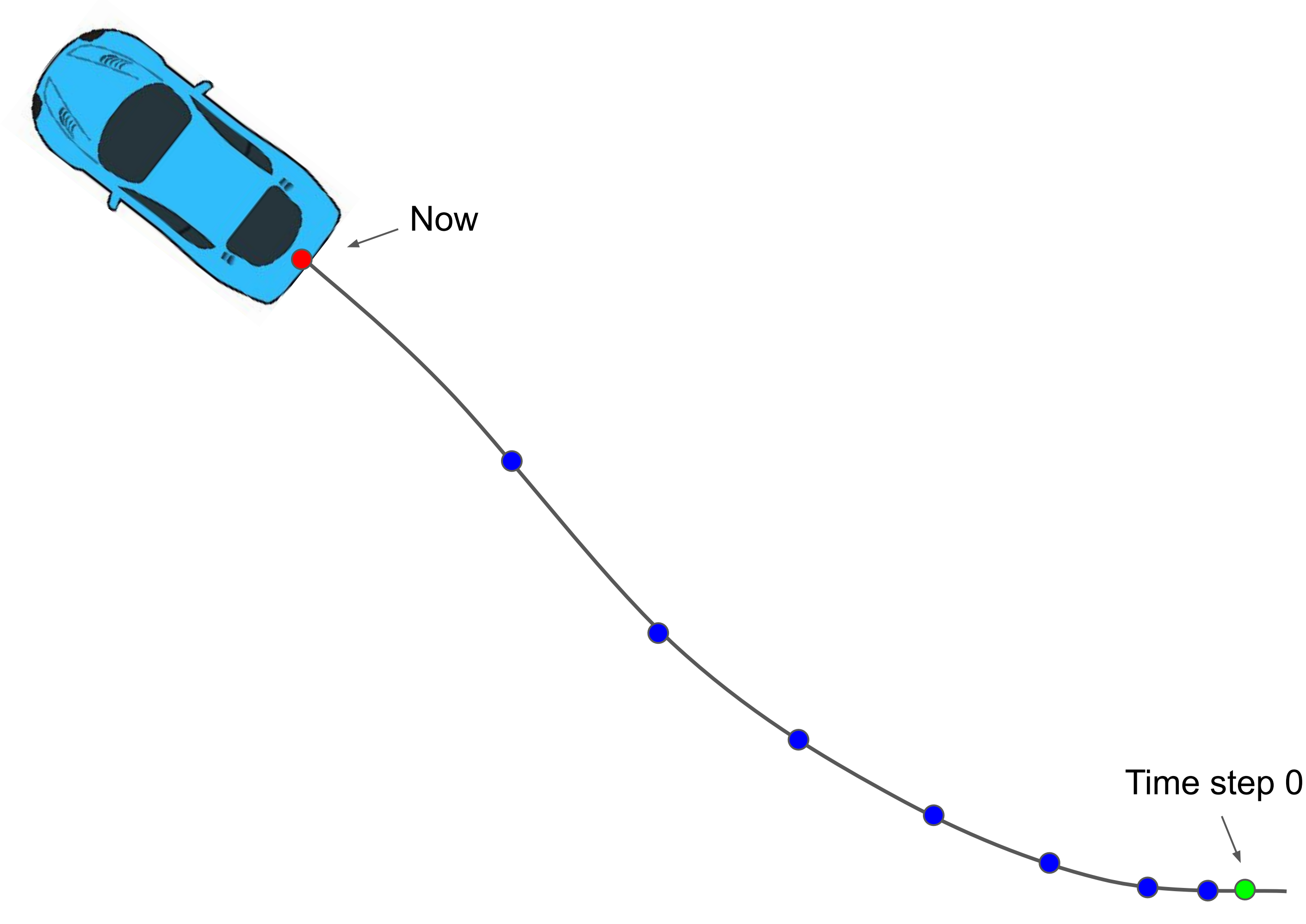


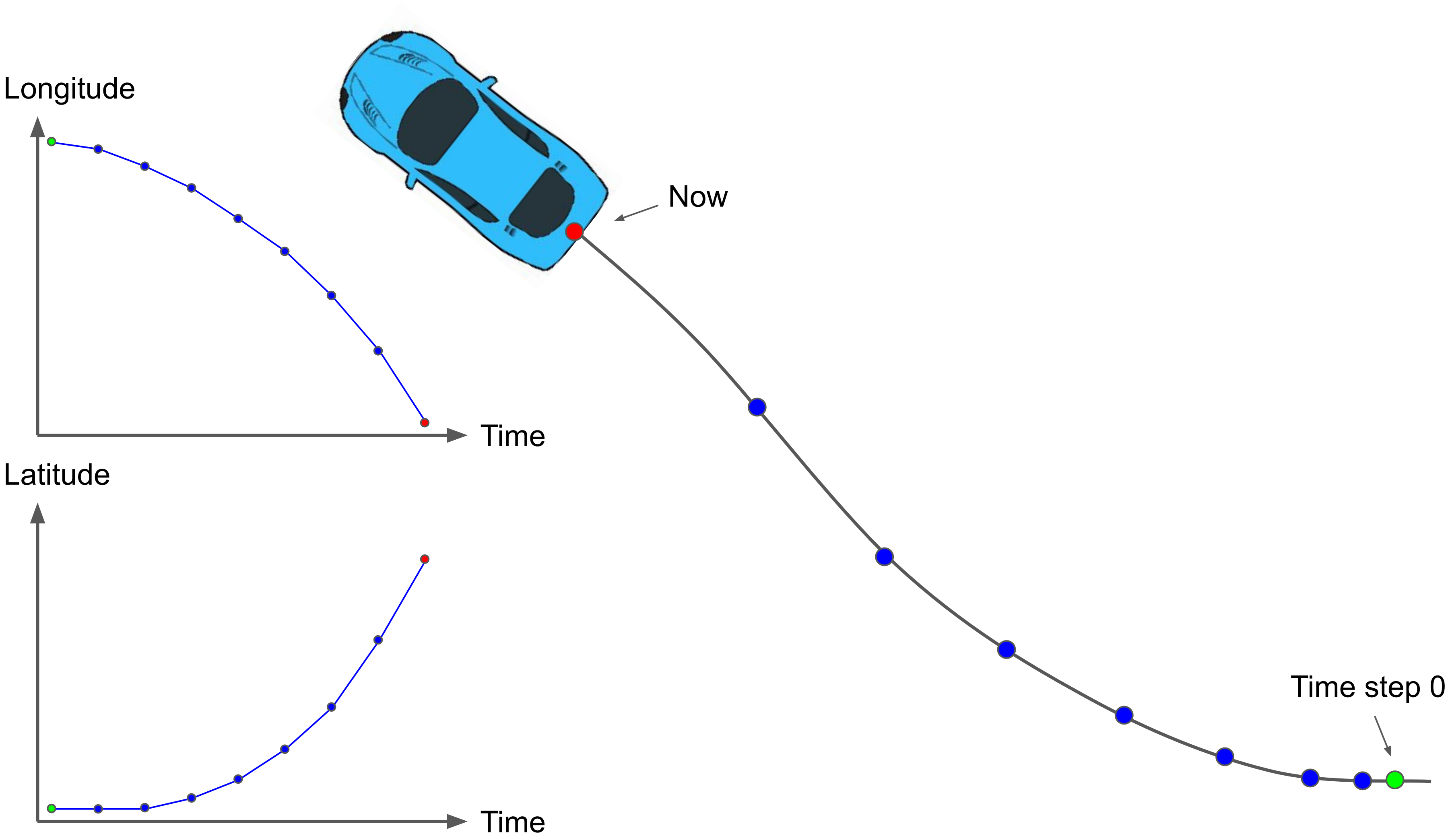
Birth and Death Rate in Japan



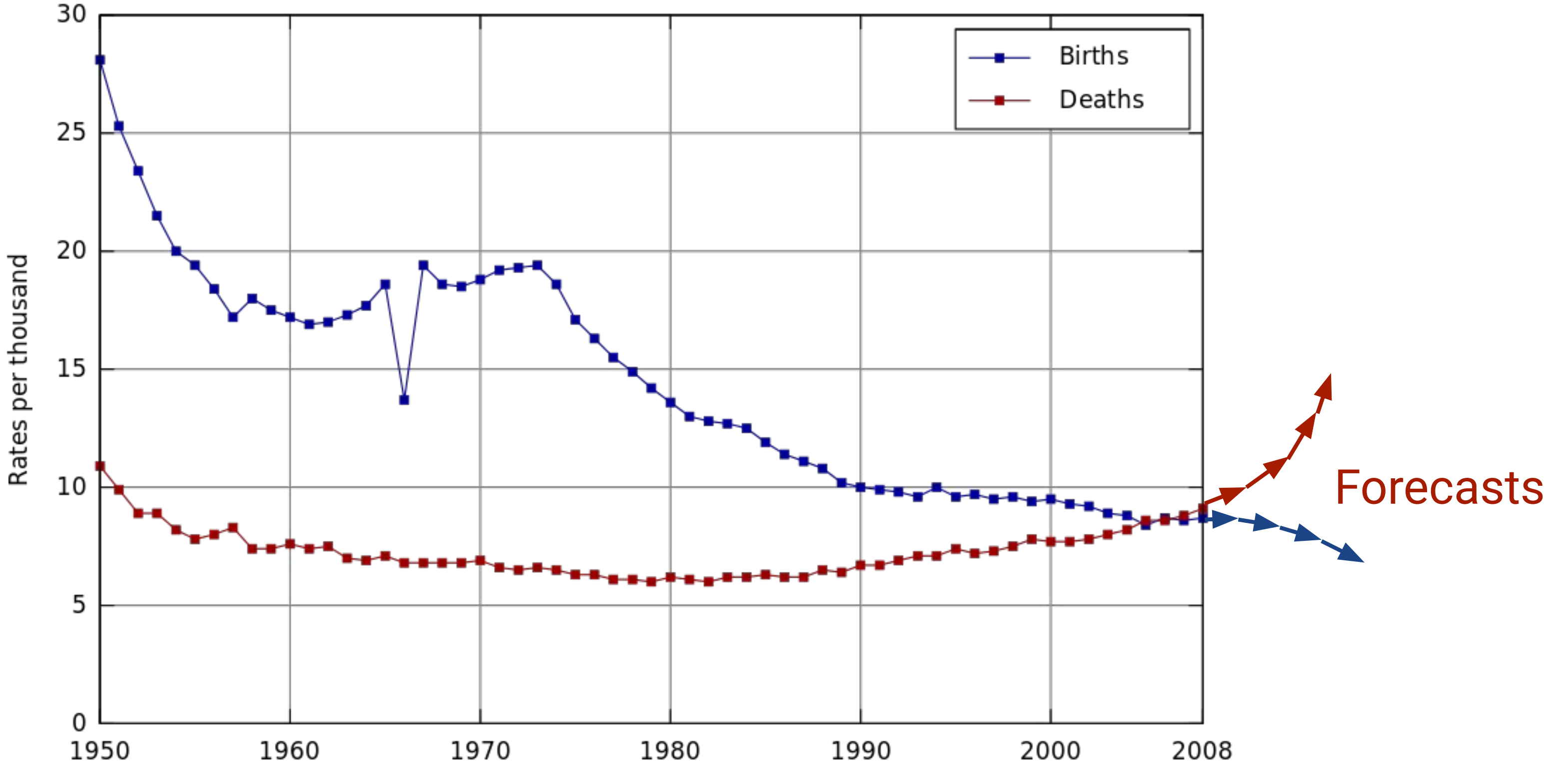


Global Temperature (°F)
58.5
58.0
57.5
57.0
56.5
CO_2 Concentration (ppm)
400
380
360
340
320
300
280
260
1880
1900
1920
1940
1960
1980
2000
Year
CO_2 Concentration


Now
Time step 0


Longitude
Time
Latitude
Time
Now
Time step 0

Birth and Death Rate in Japan



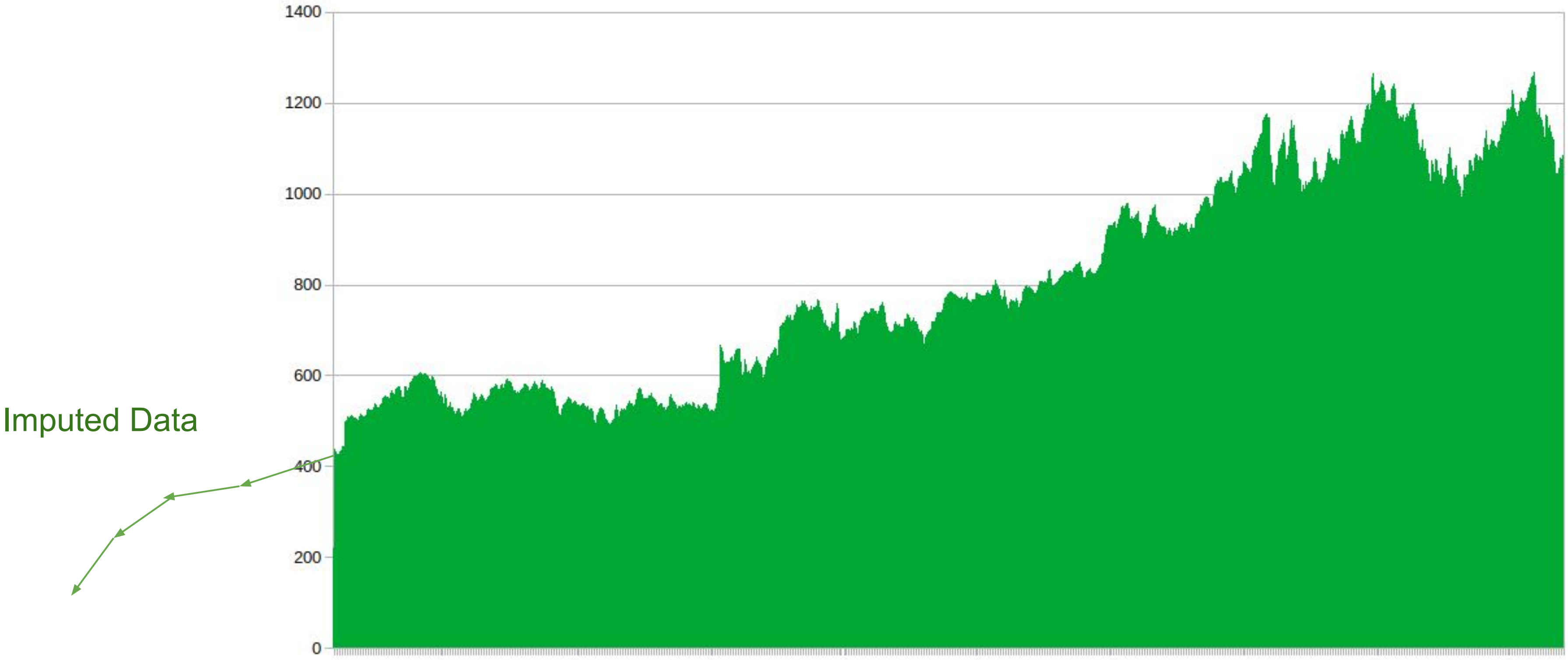


Chart Created by lmoroney@

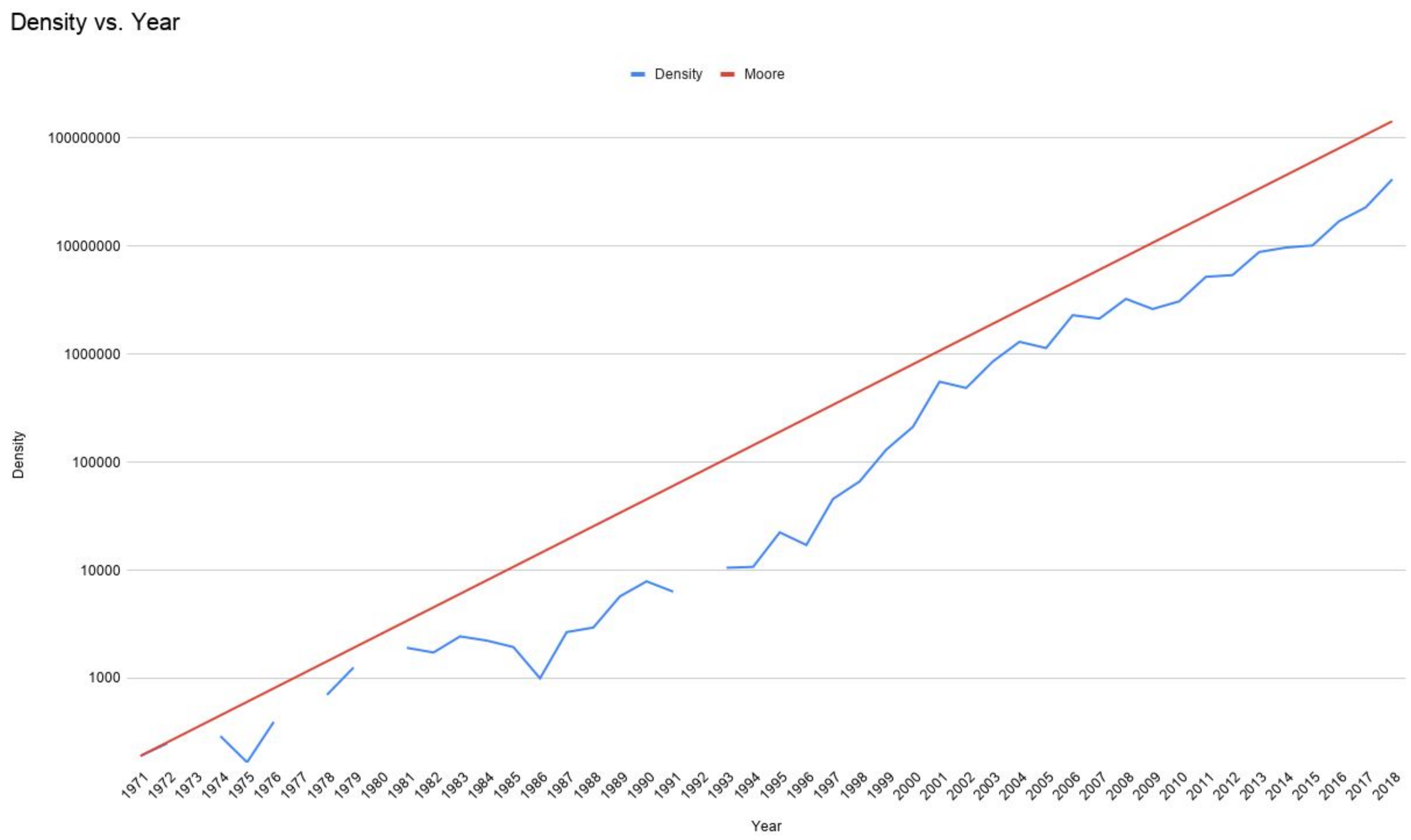


Chart Created by lmoroney@

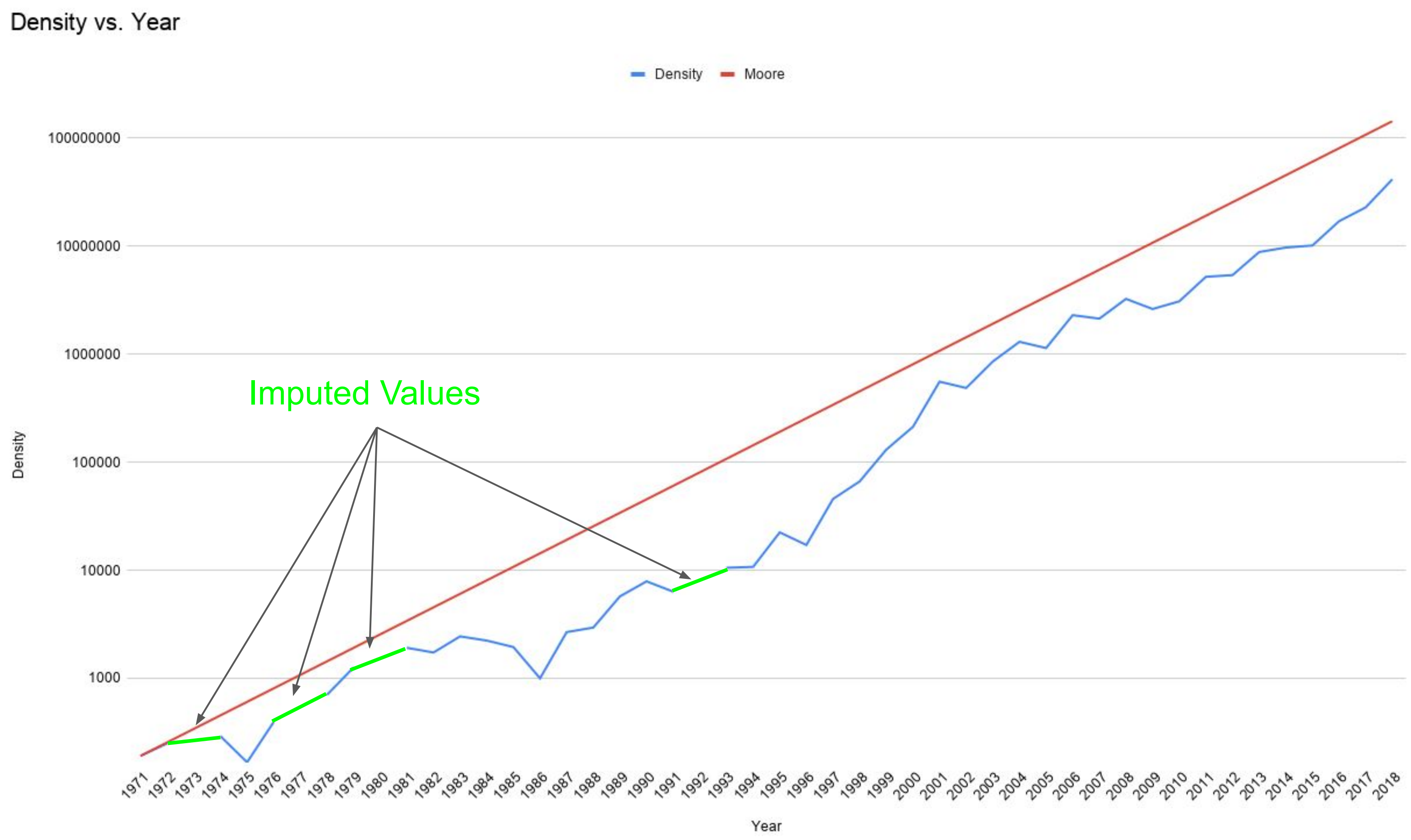
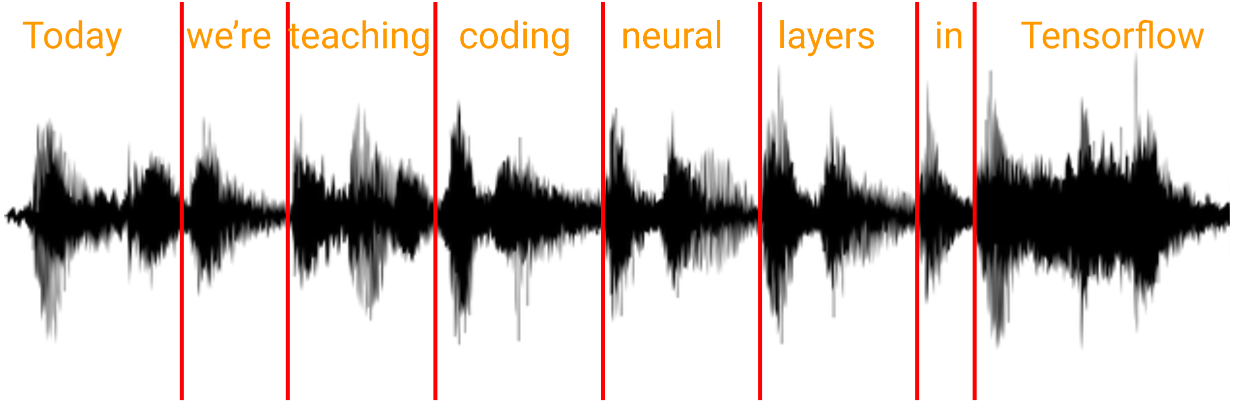


Chart Created by lmoroney@


150
140
130
120
110
100
90
0
50
100
150


Today
we're
teaching
coding
neural
layers
in
Tensorflow

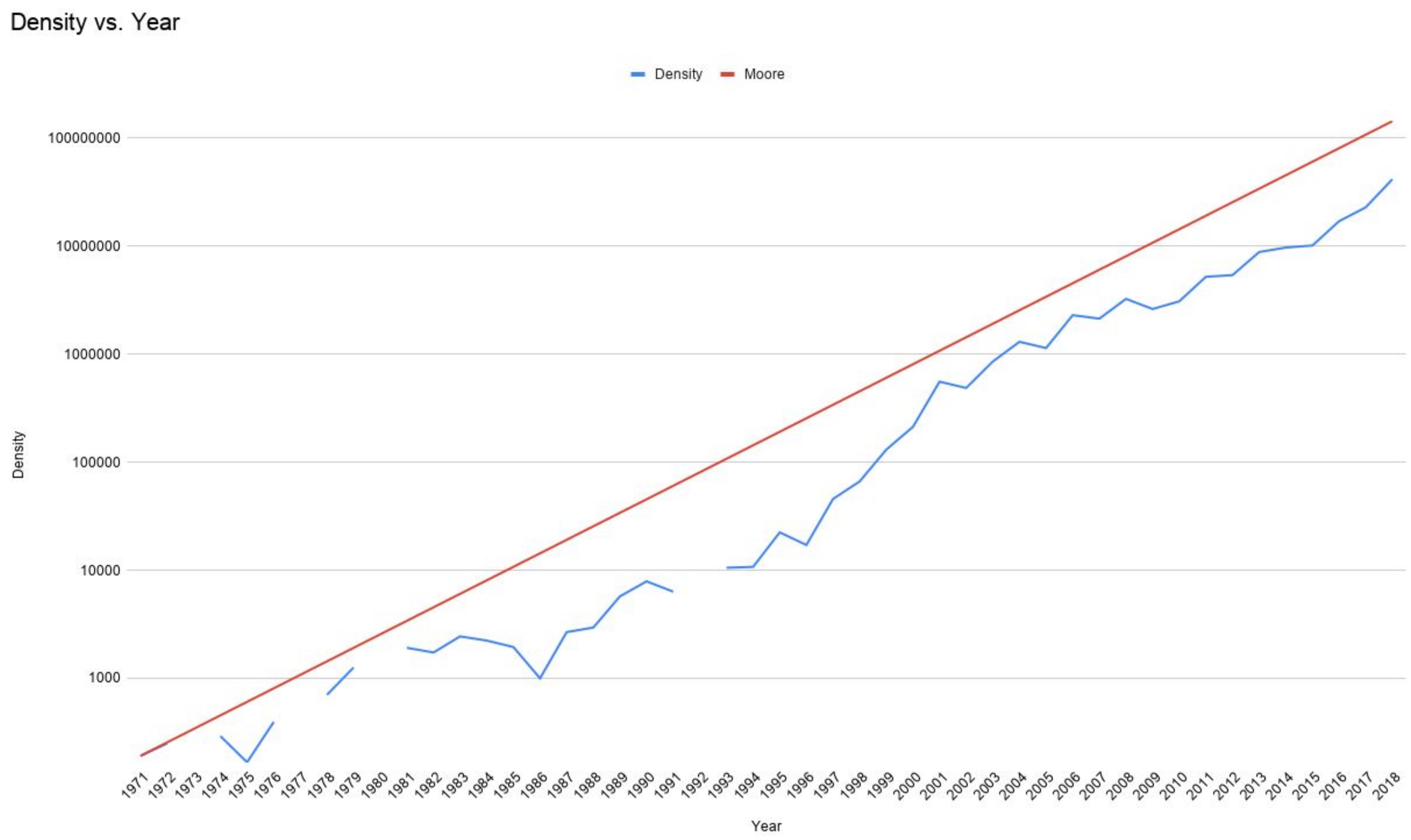
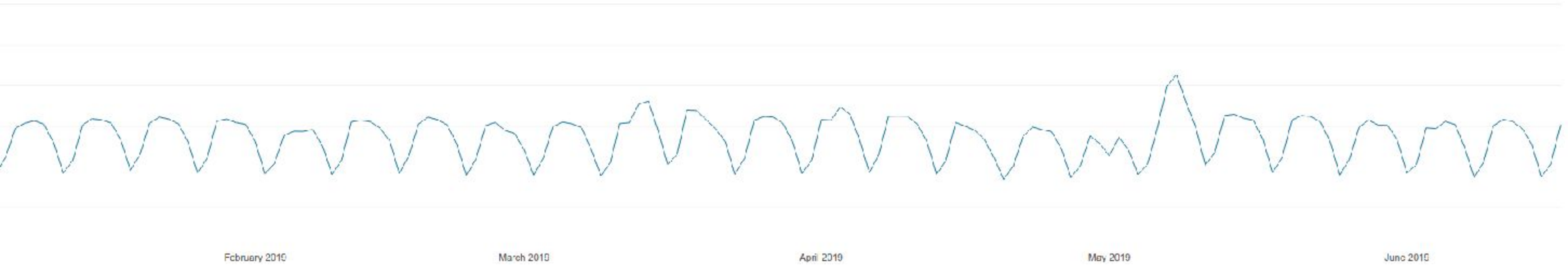
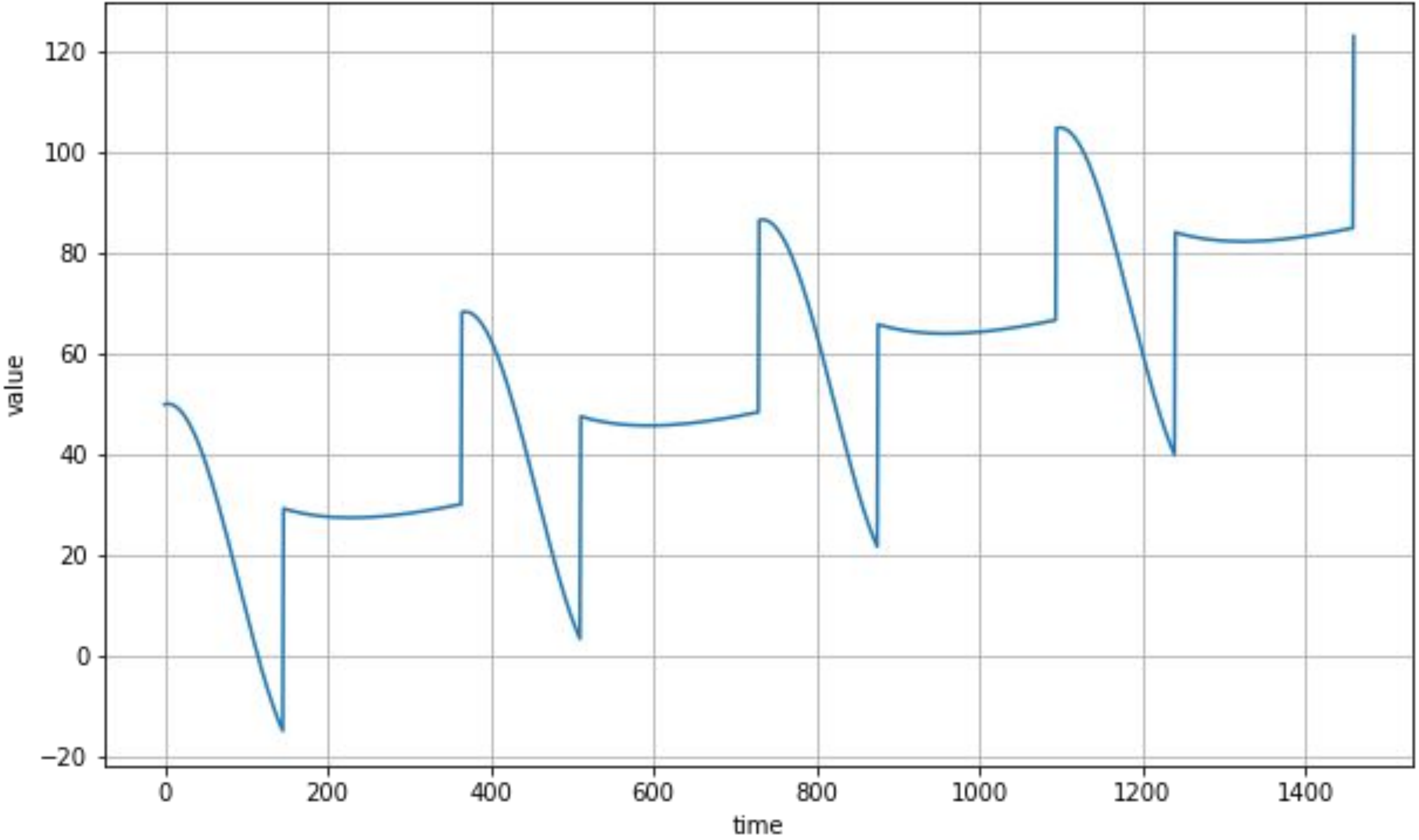
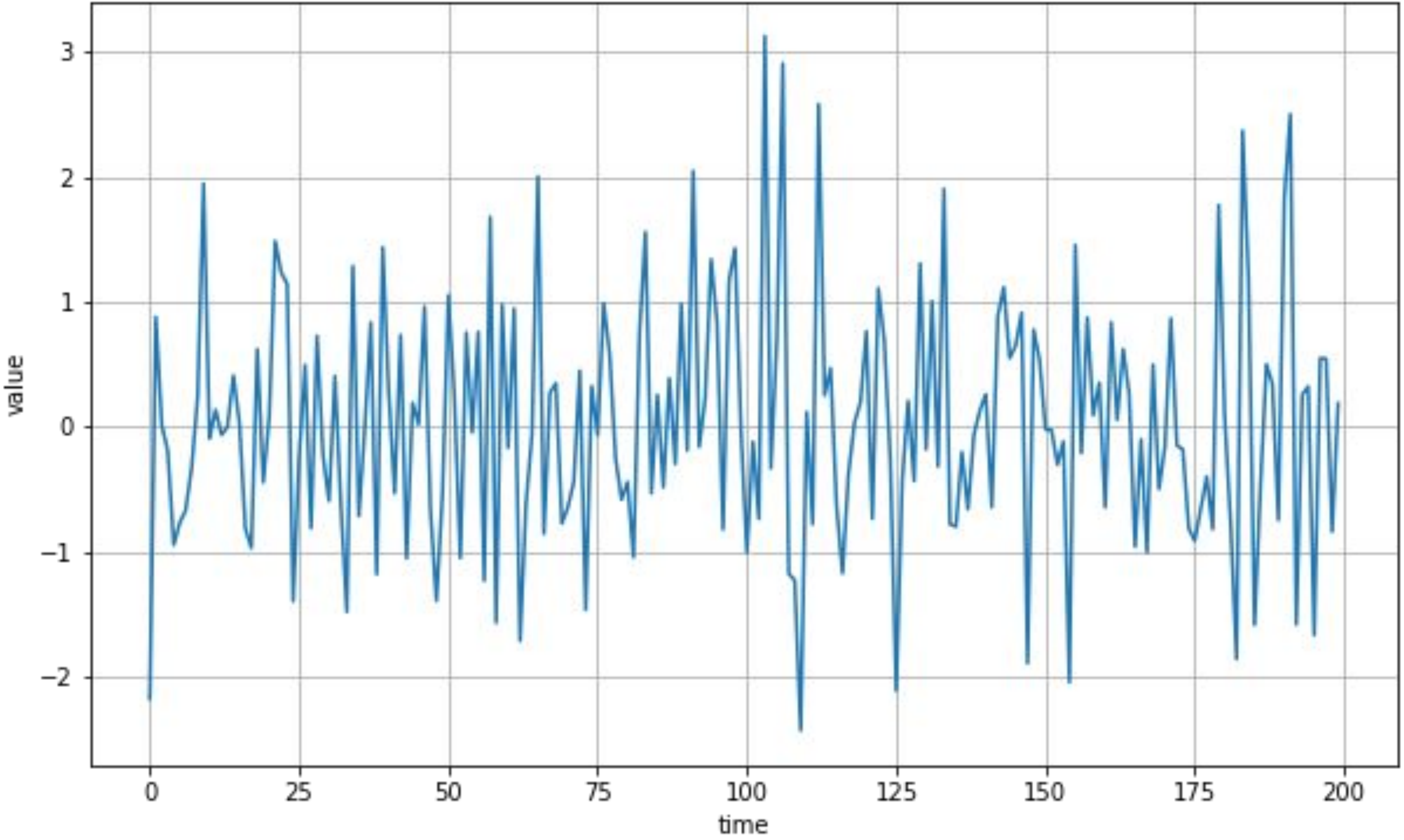


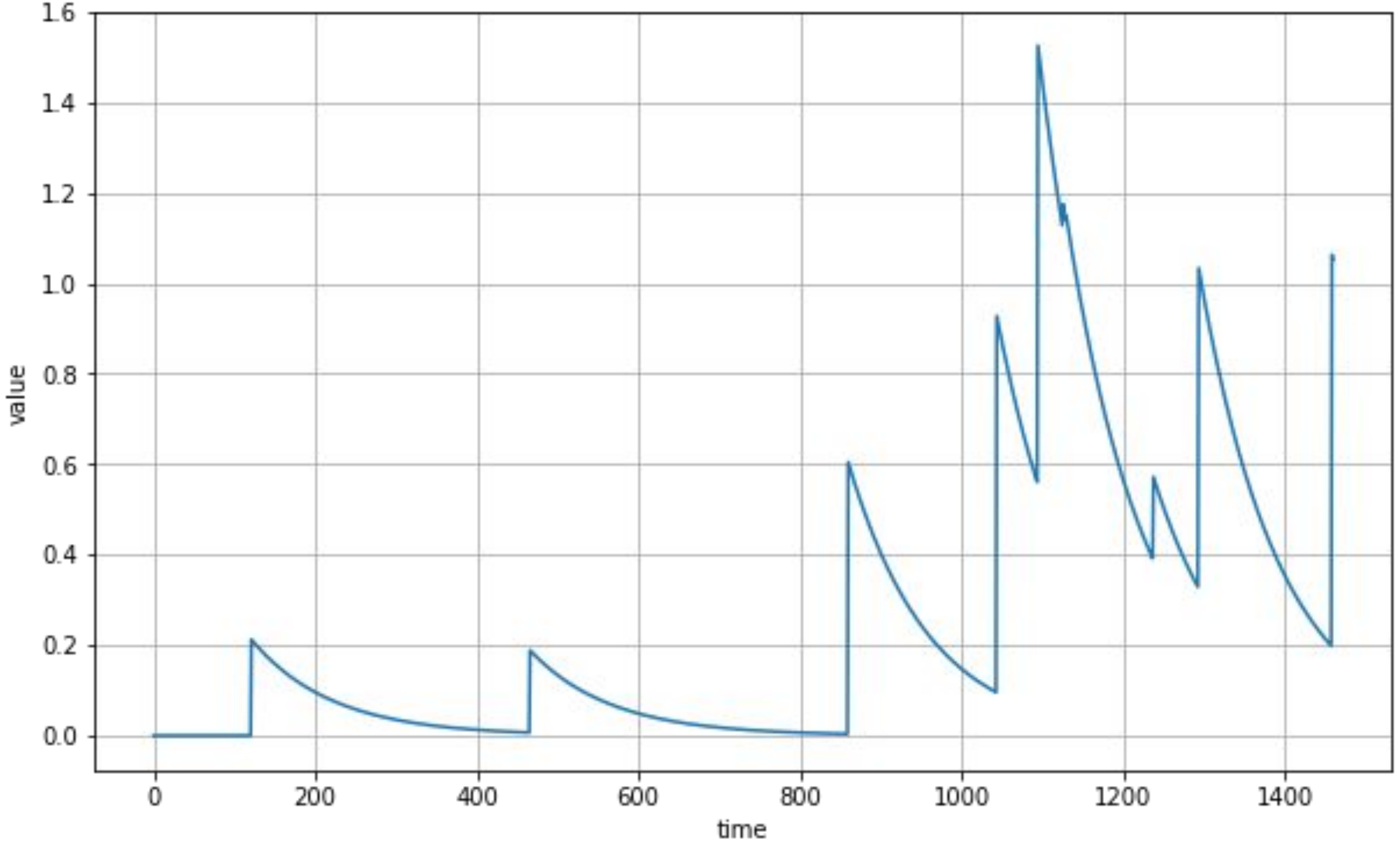
Chart Created by lmoroney@


February 2019
March 2019
April 2019
May 2019
June 2019

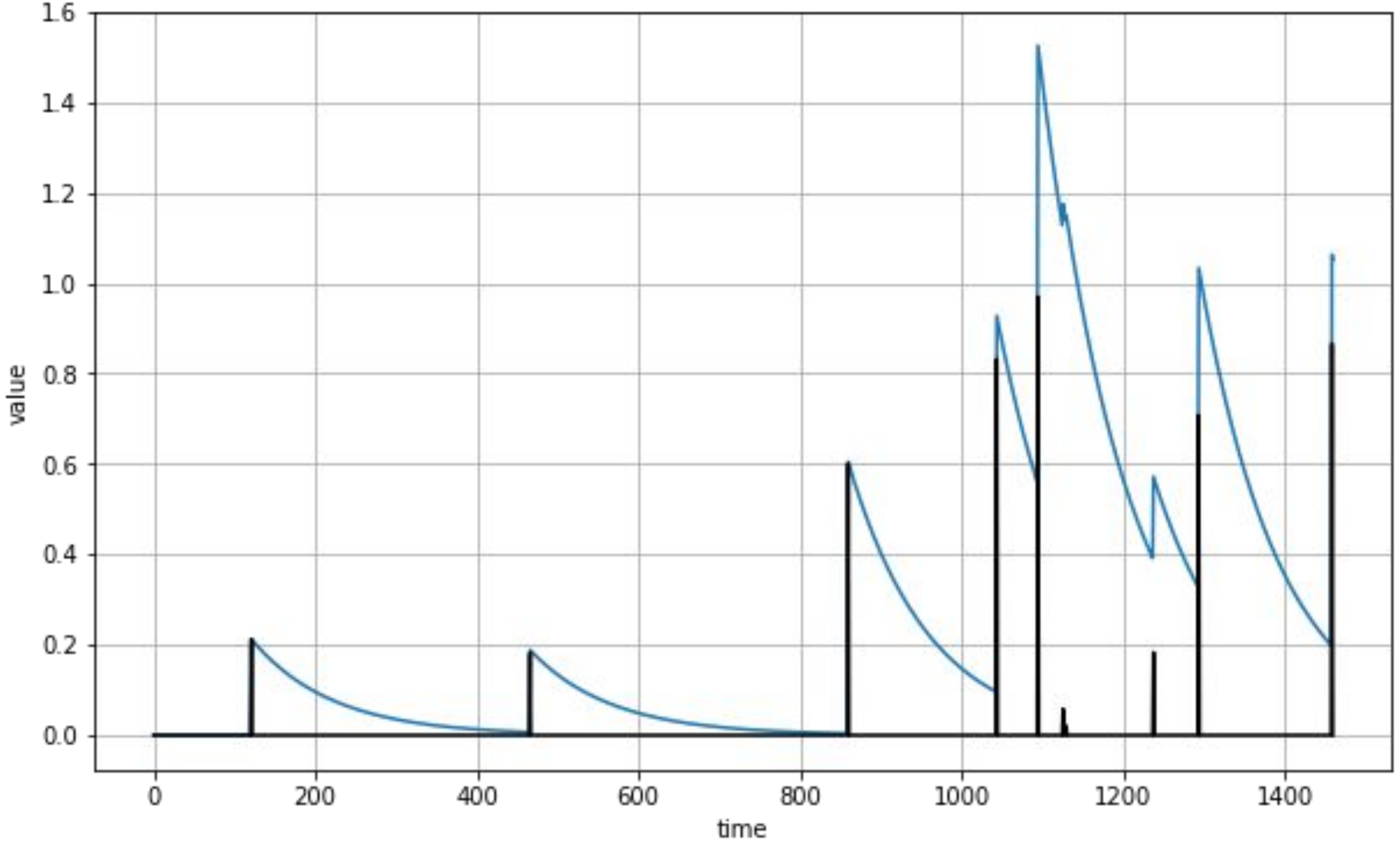

value
time
120
100
80
60
40
20
0
−20
0
200
400
600
800
1000
1200
1400


value
time
3
2
1
0
−1
−2
0
25
50
75
100
125
150
175
200

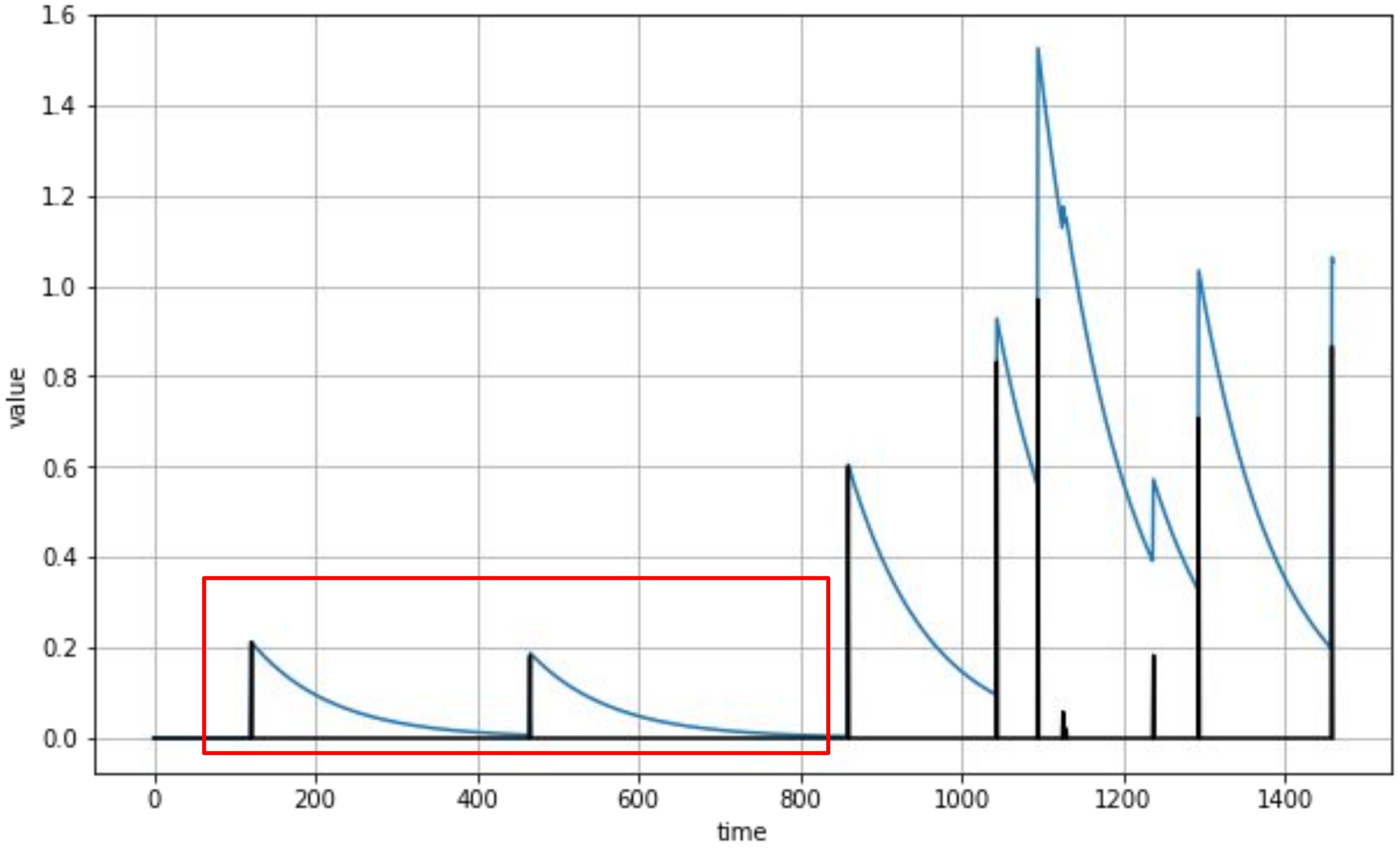
Autocorrelation



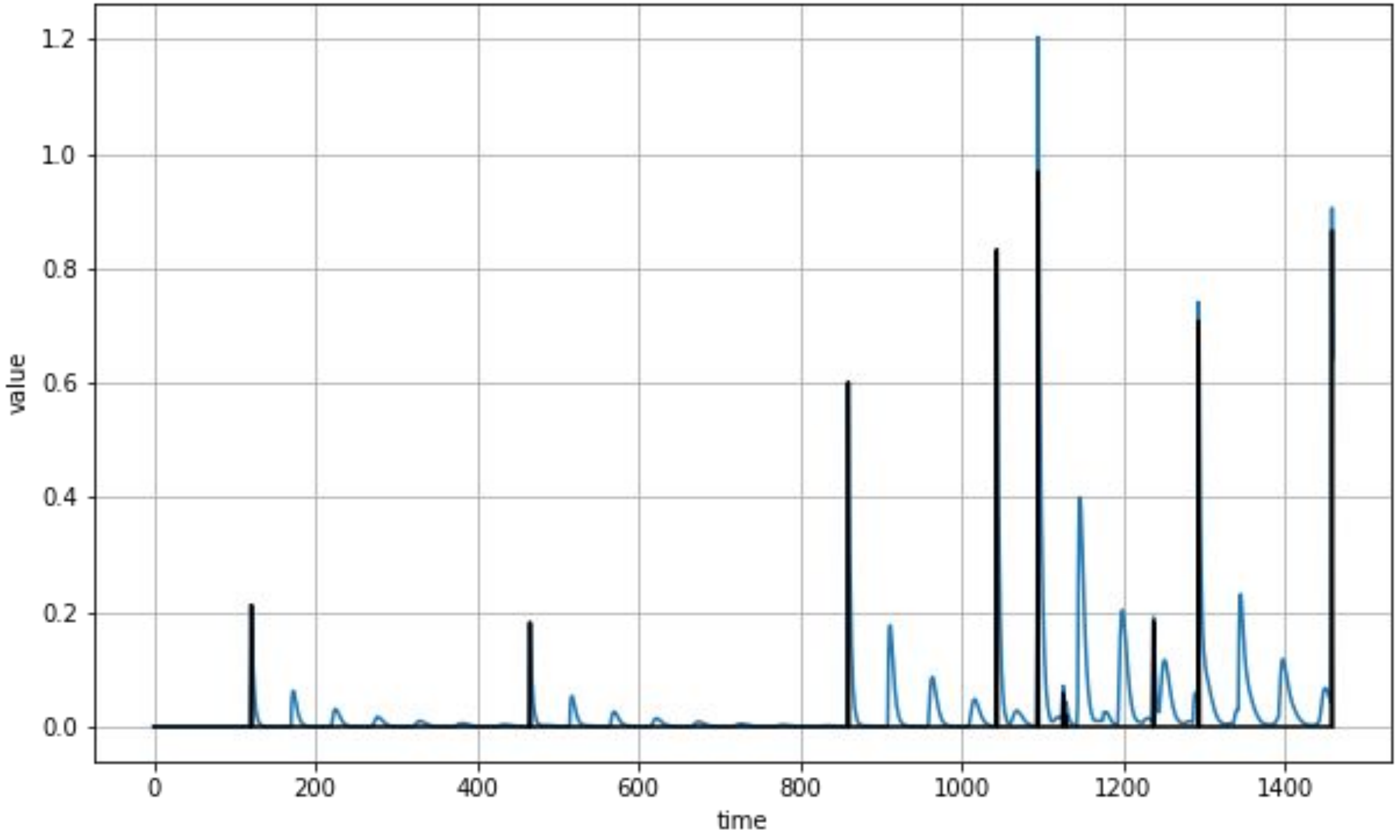
$v(t) = 0.99 \times v(t-1)$ + occasional spike



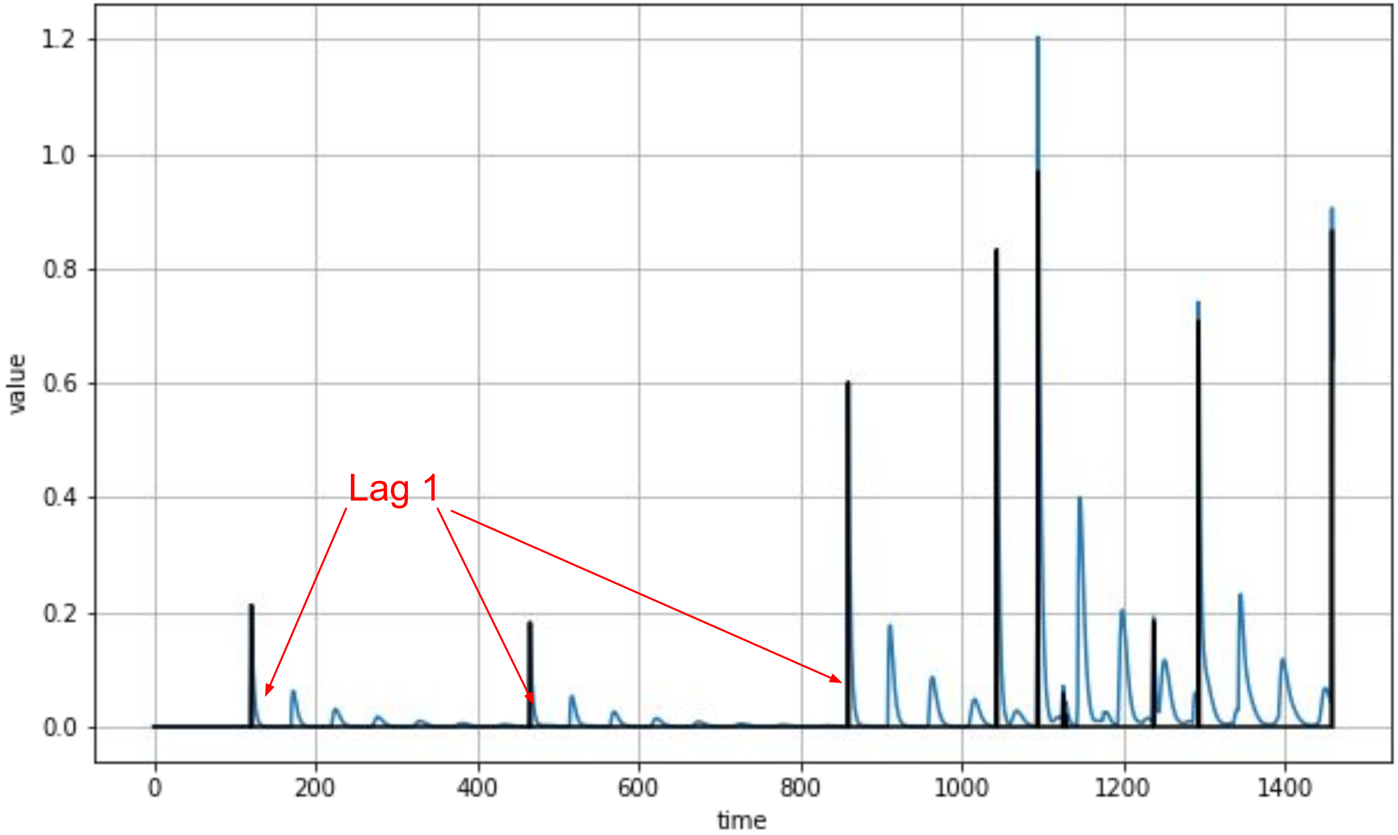
$v(t) = 0.99 \times v(t-1)$ + occasional spike



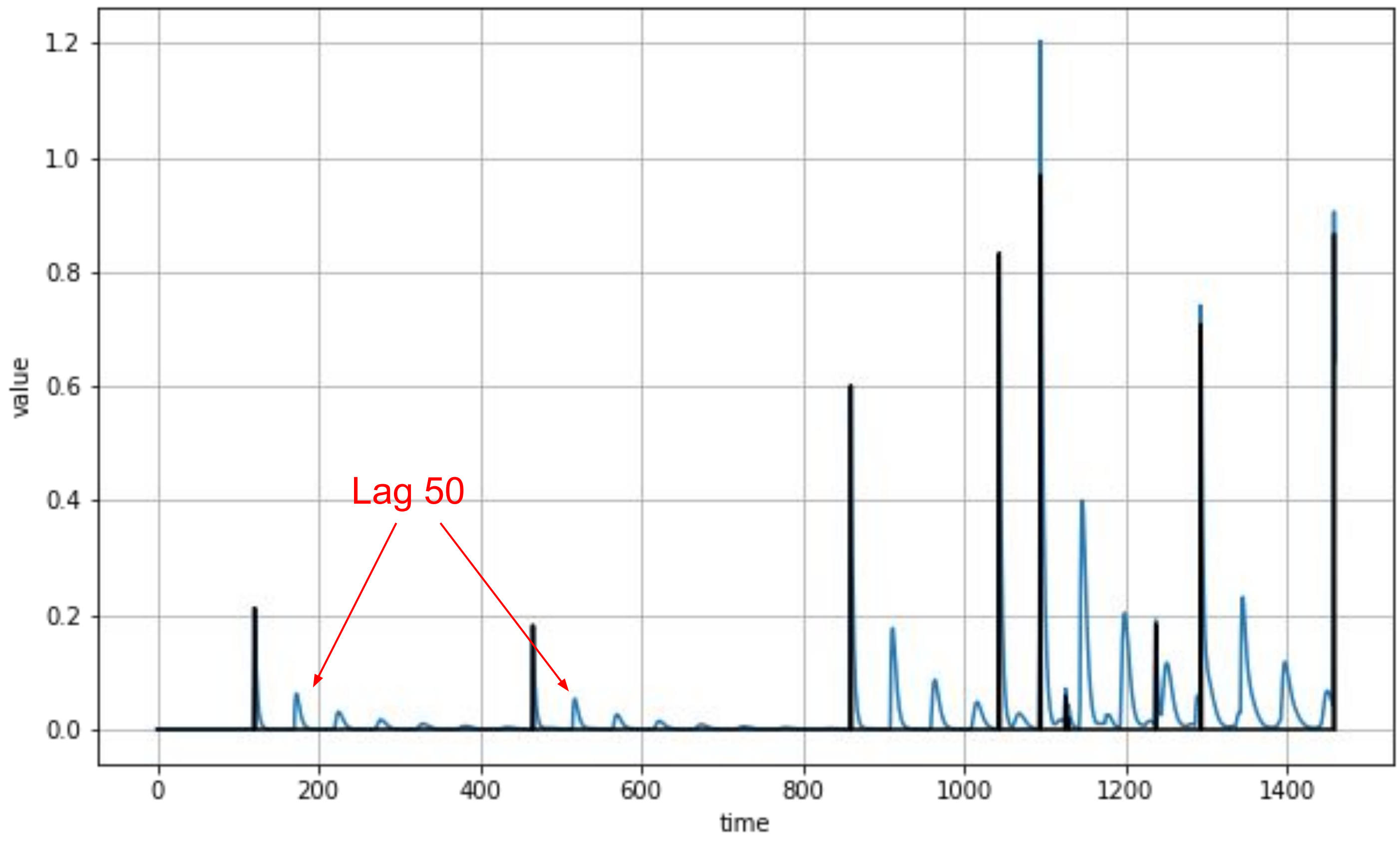
$v(t) = 0.7 \times v(t-1) + 0.2 \times v(t-50)$ + occasional spike



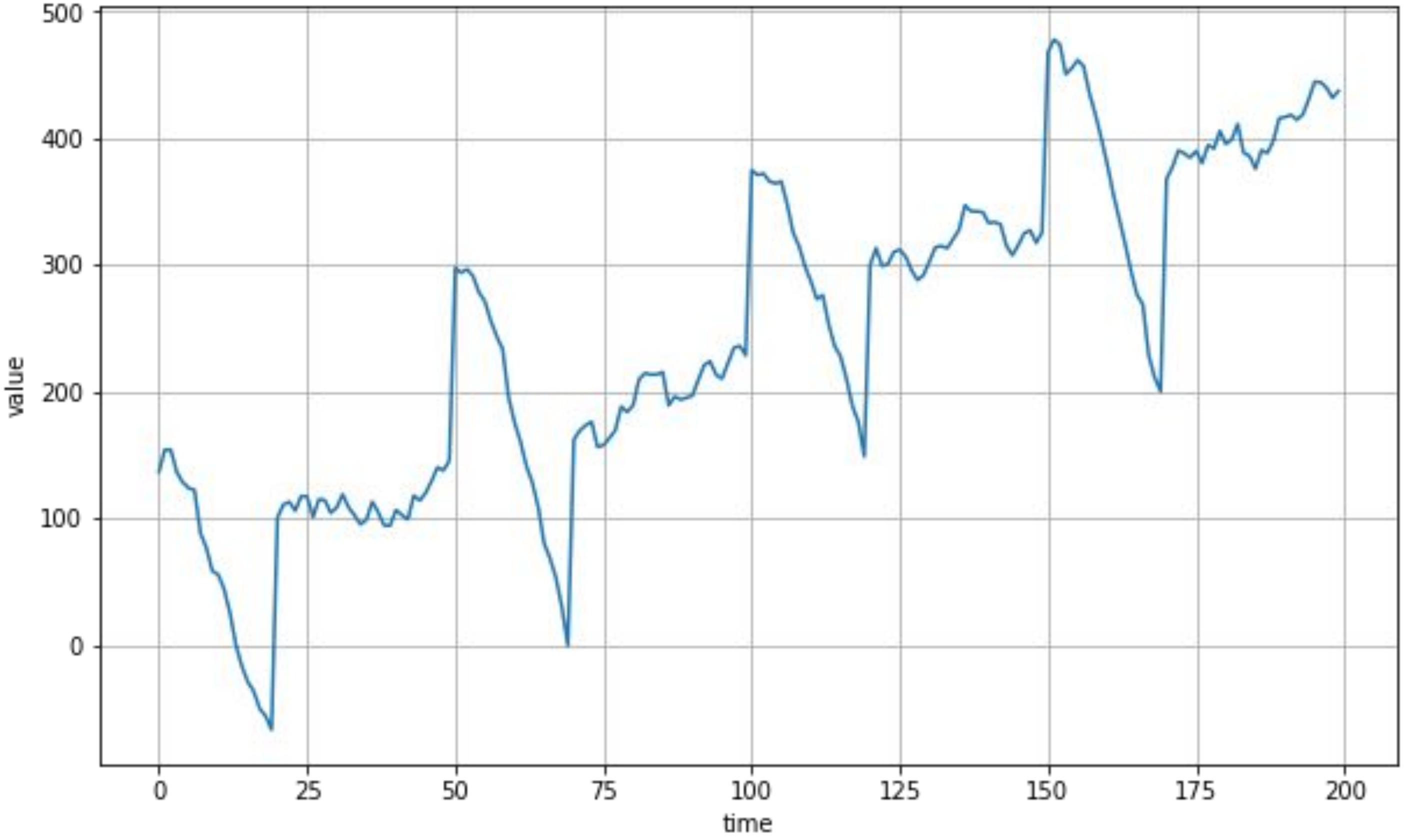
$v(t) = 0.7 \times v(t-1) + 0.2 \times v(t-50)$ + occasional spike



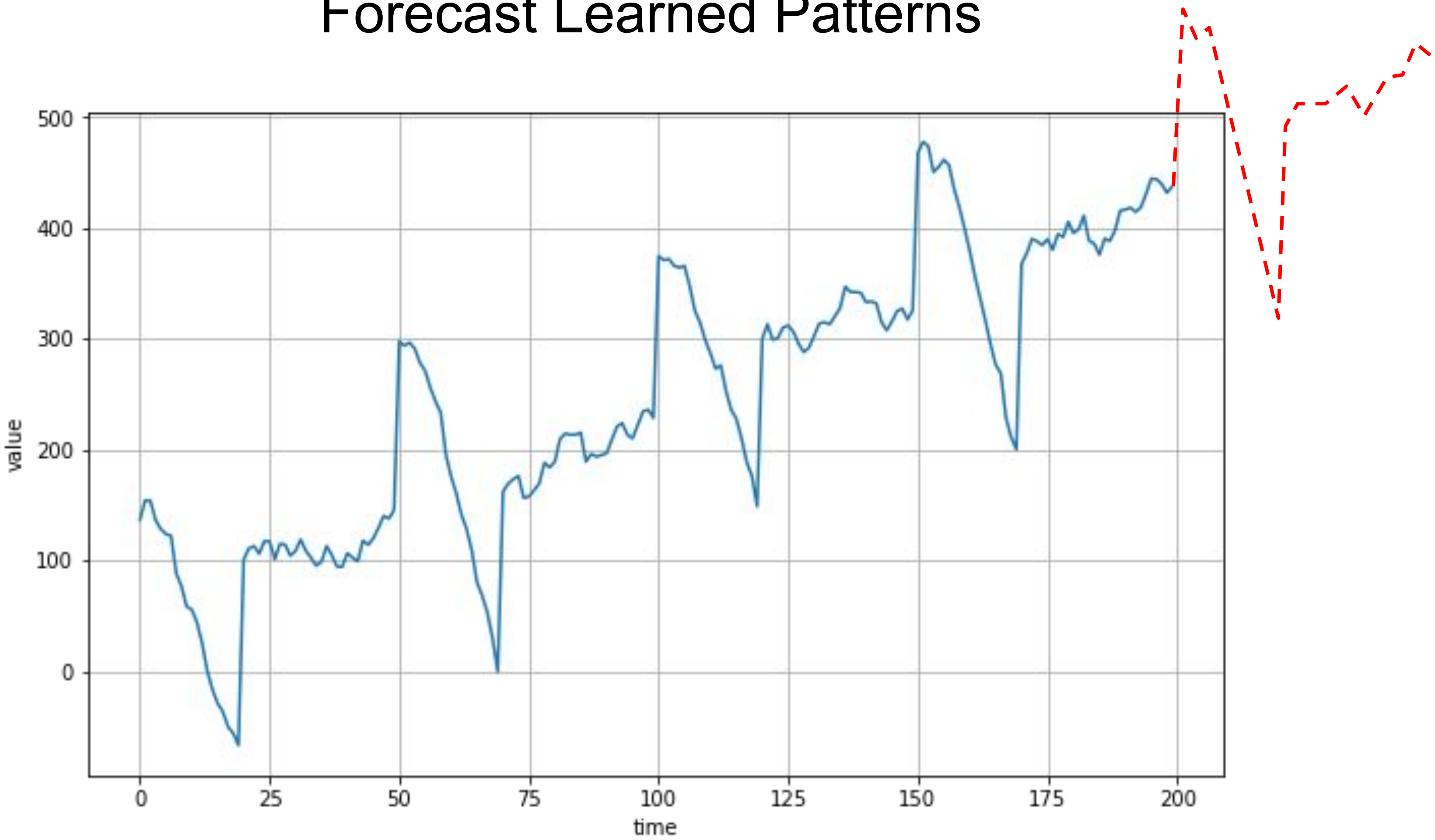
$v(t) = 0.7 \times v(t-1) + 0.2 \times v(t-50)$ + occasional spike



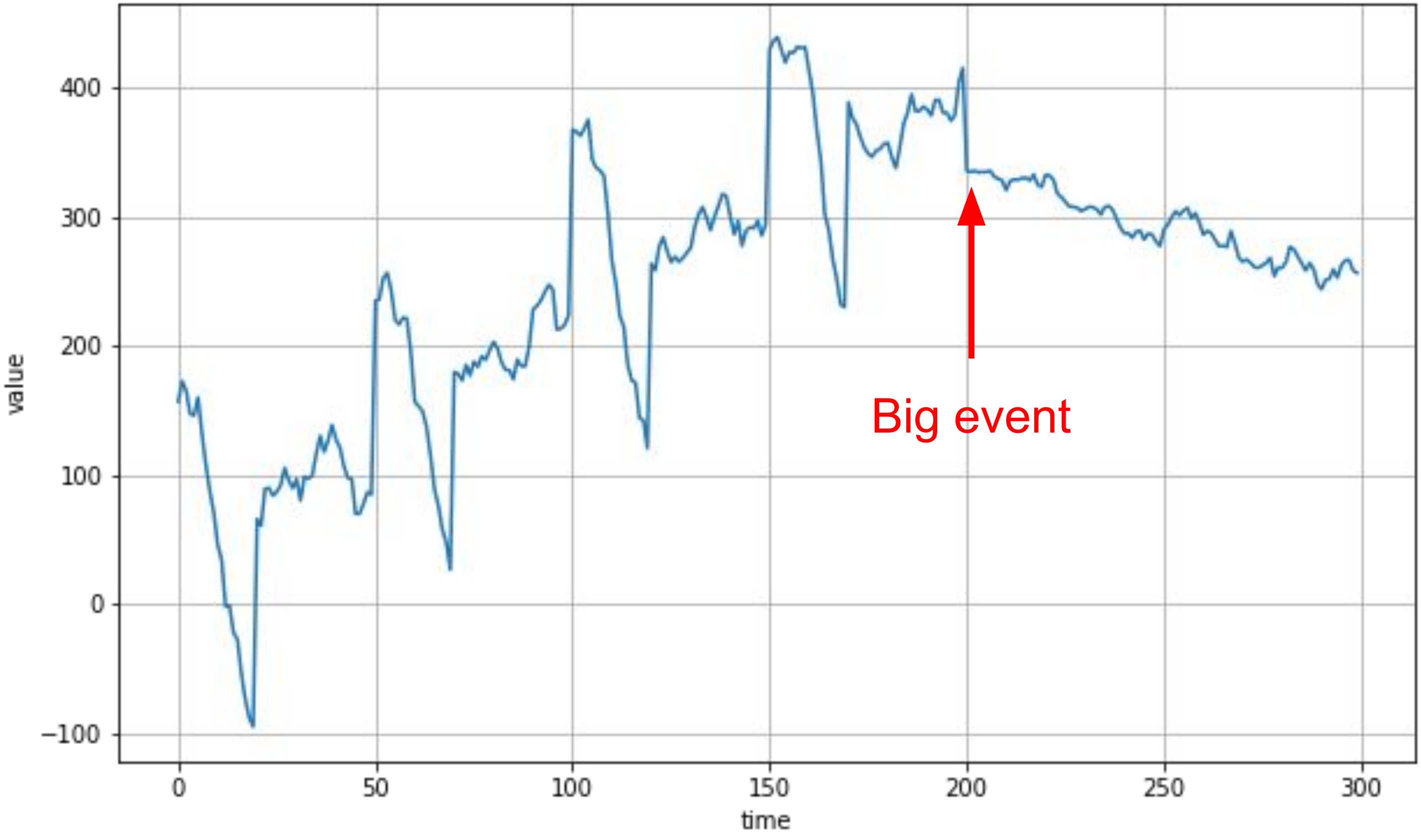
Trend + Seasonality + Autocorrelation + Noise



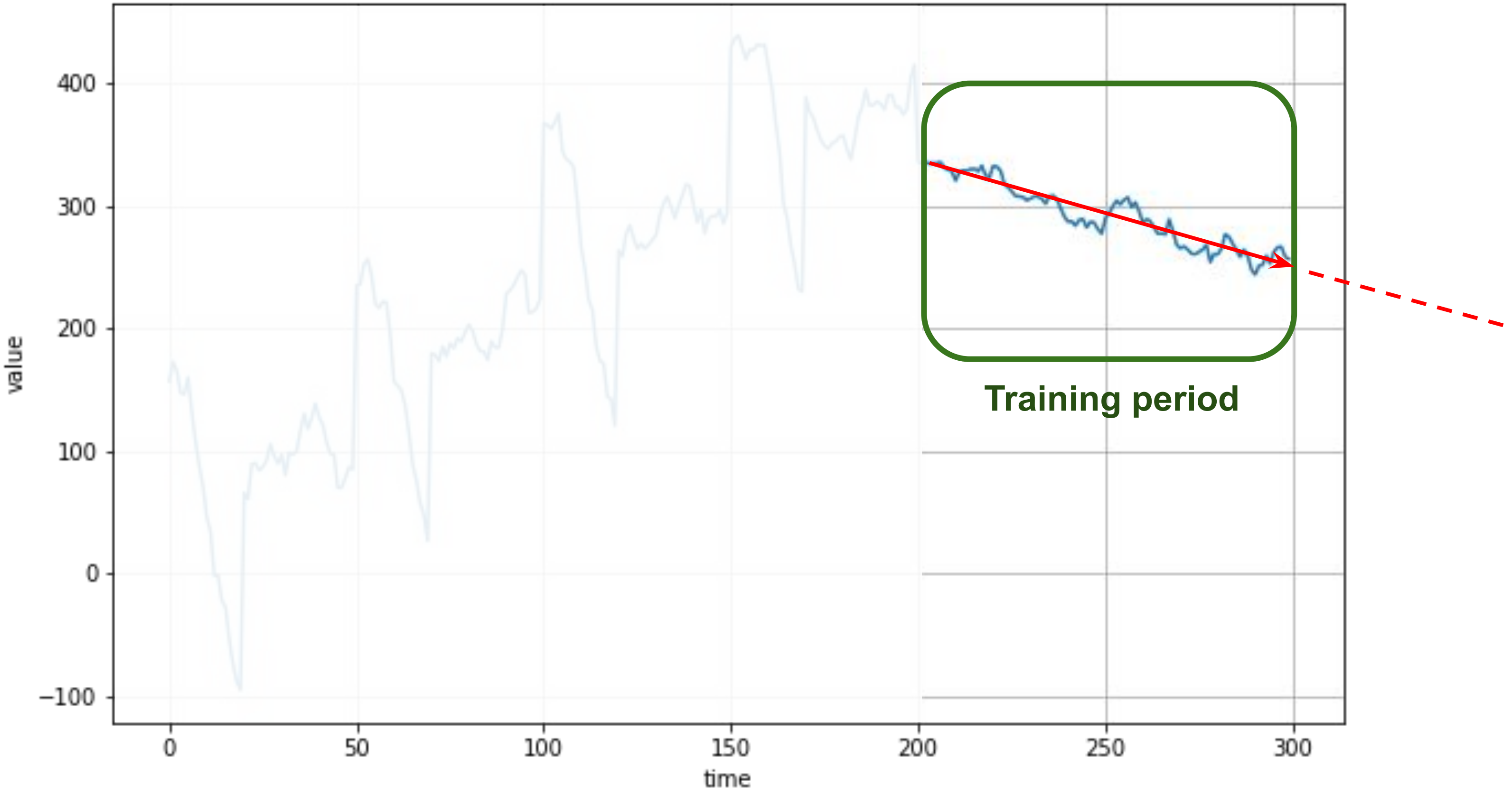
Forecast Learned Patterns



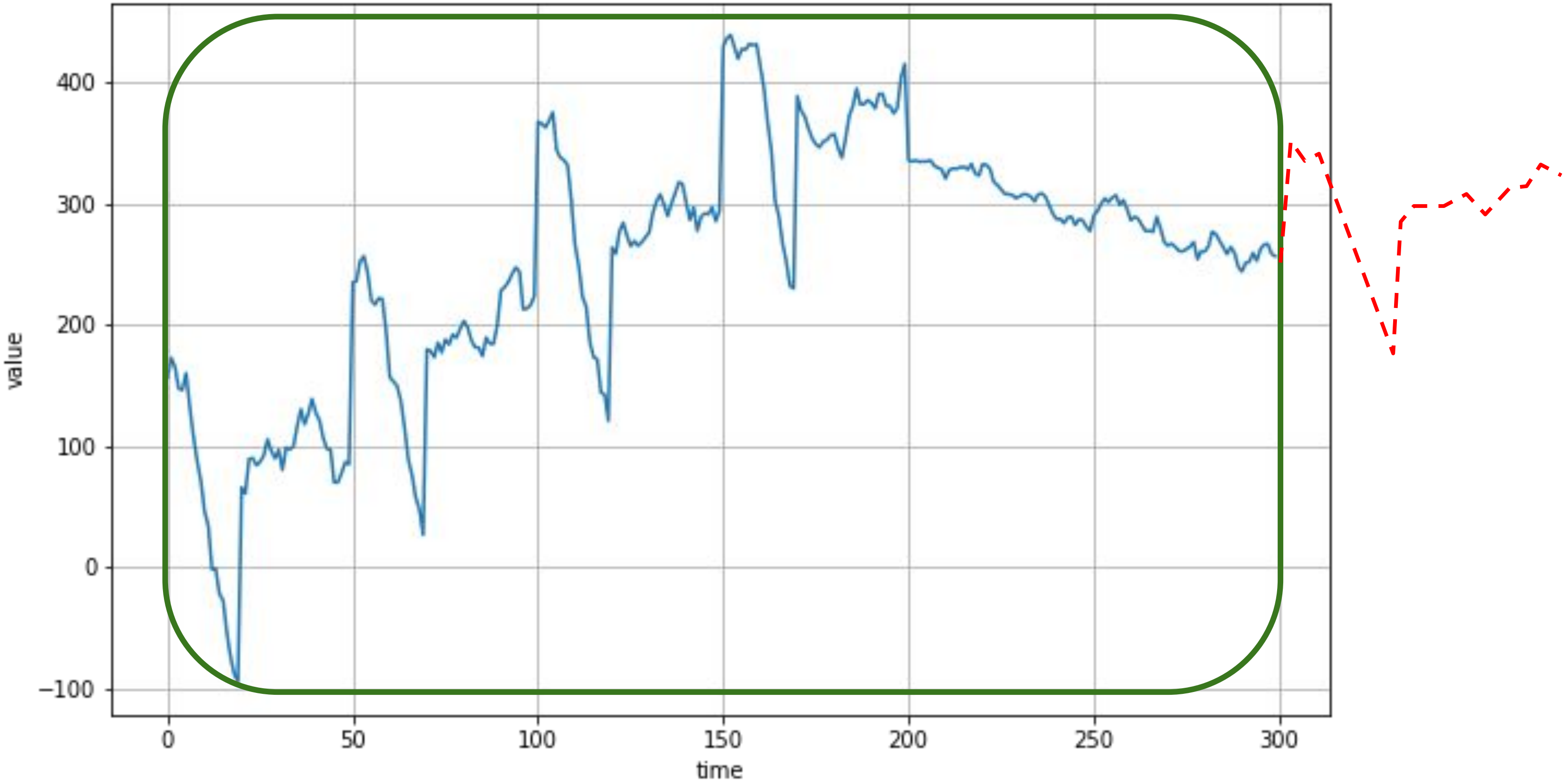
Non-Stationary Time Series



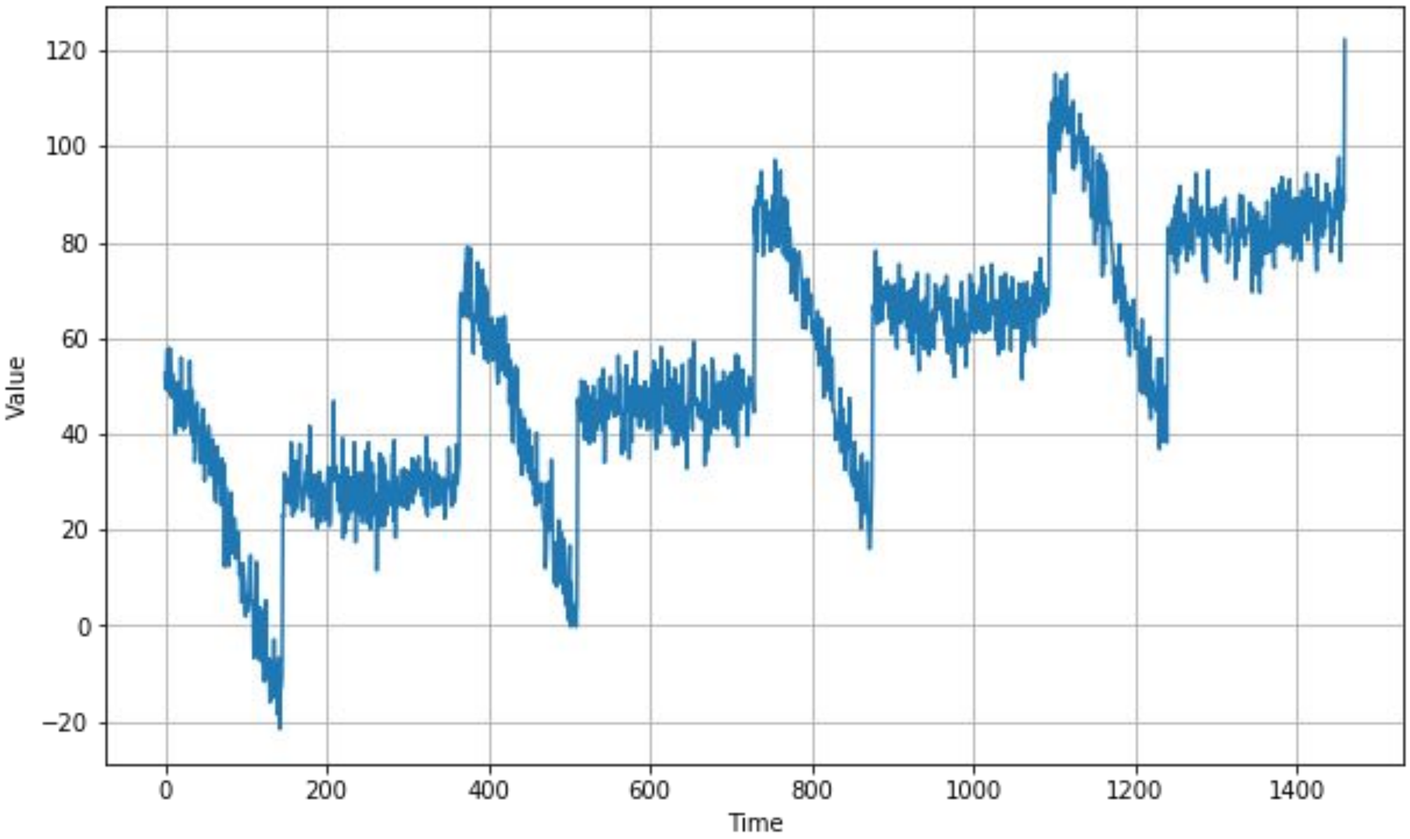
Non-Stationary Time Series



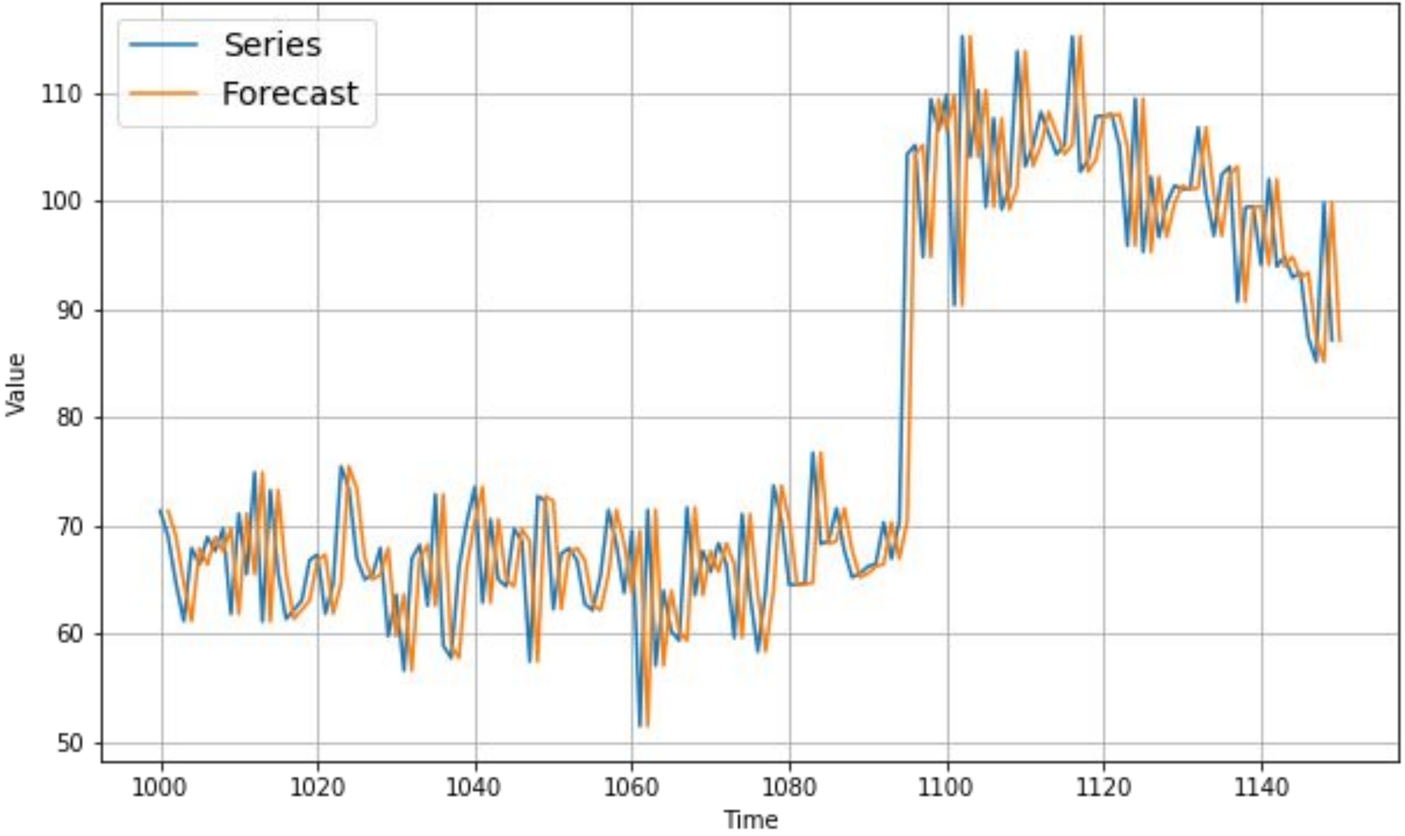
Non-Stationary Time Series



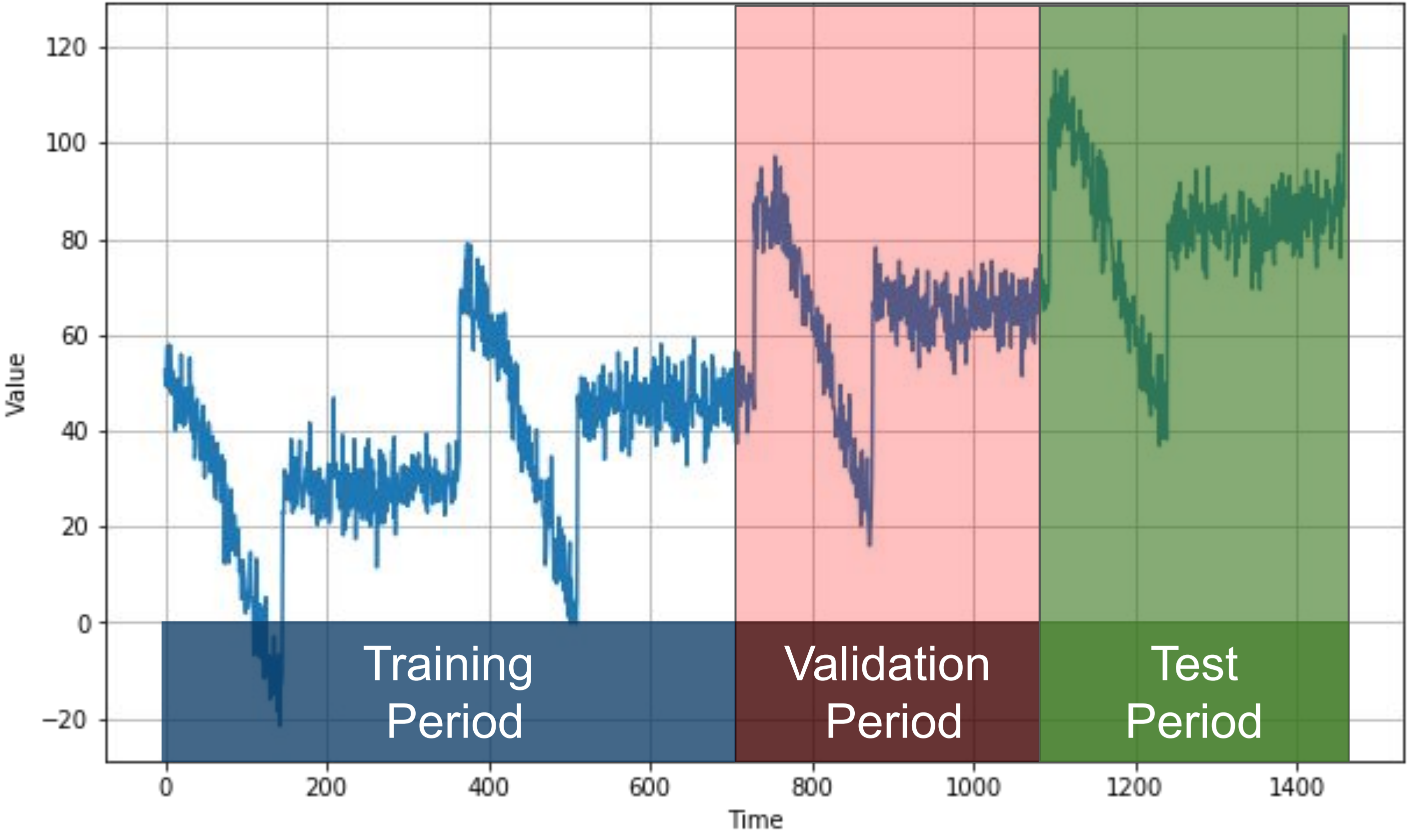
Trend + Seasonality + Noise



Naive Forecasting

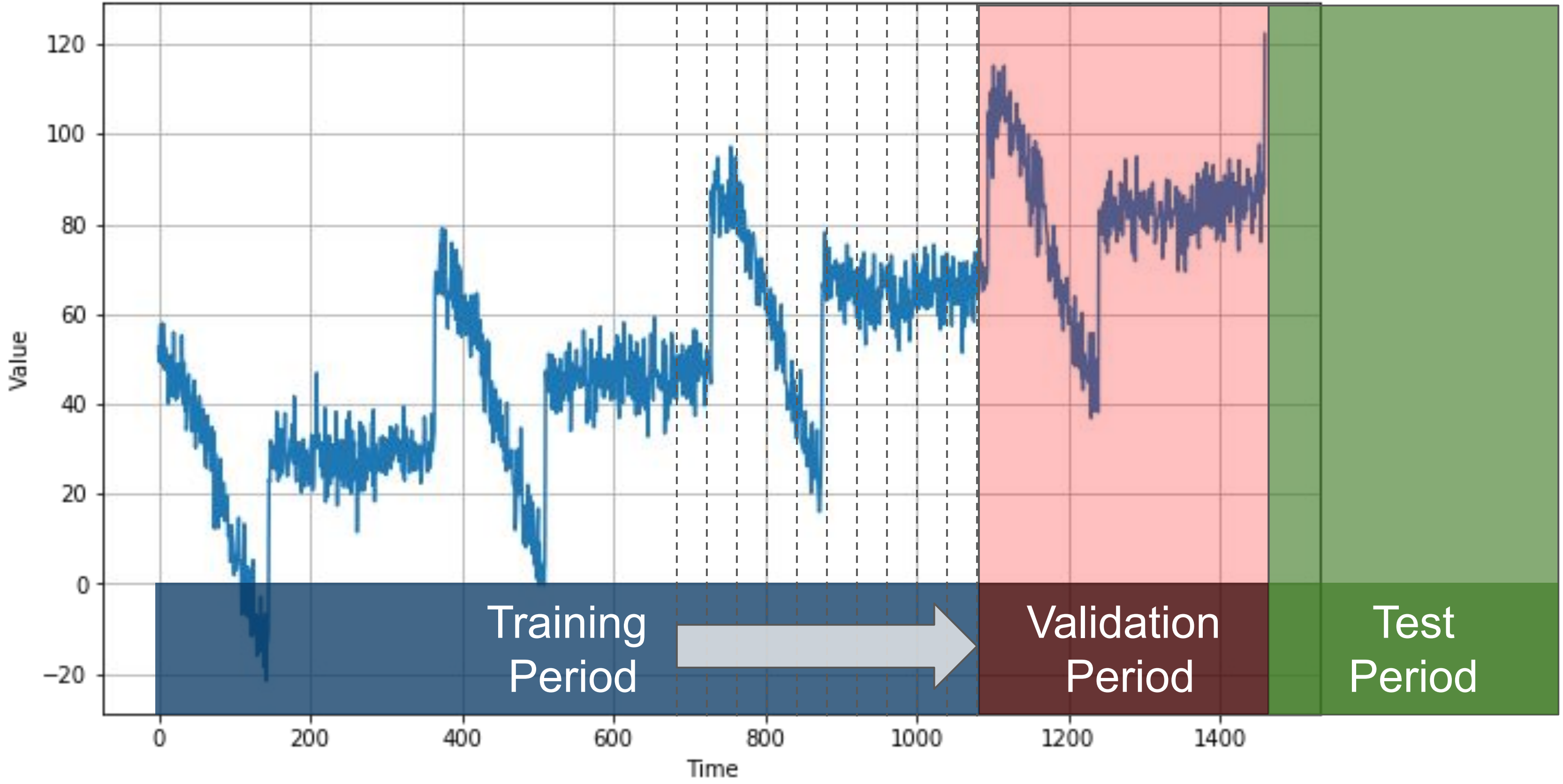


Fixed Partitioning



Fixed Partitioning

Roll-Forward Partitioning



```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

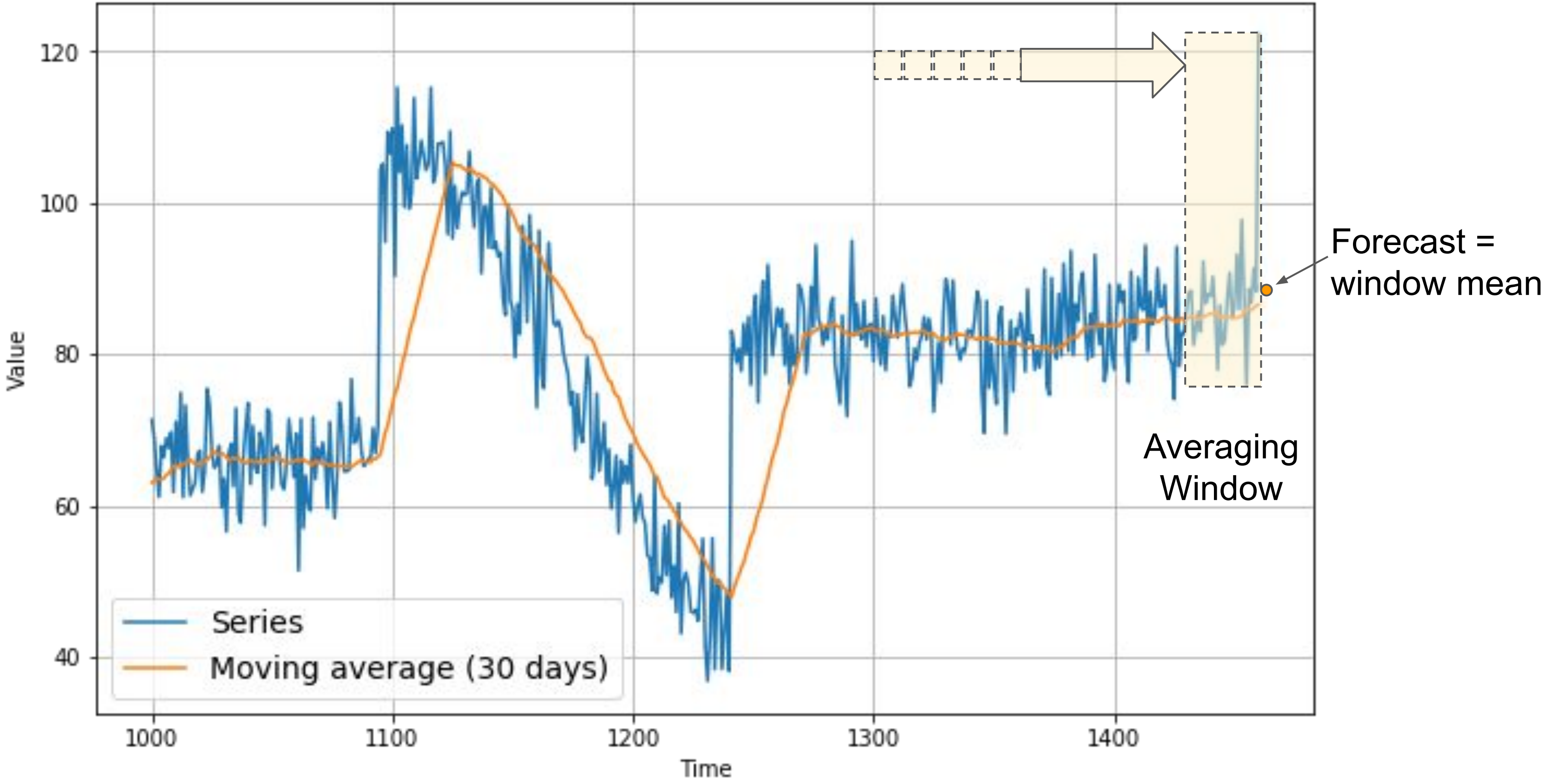
```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

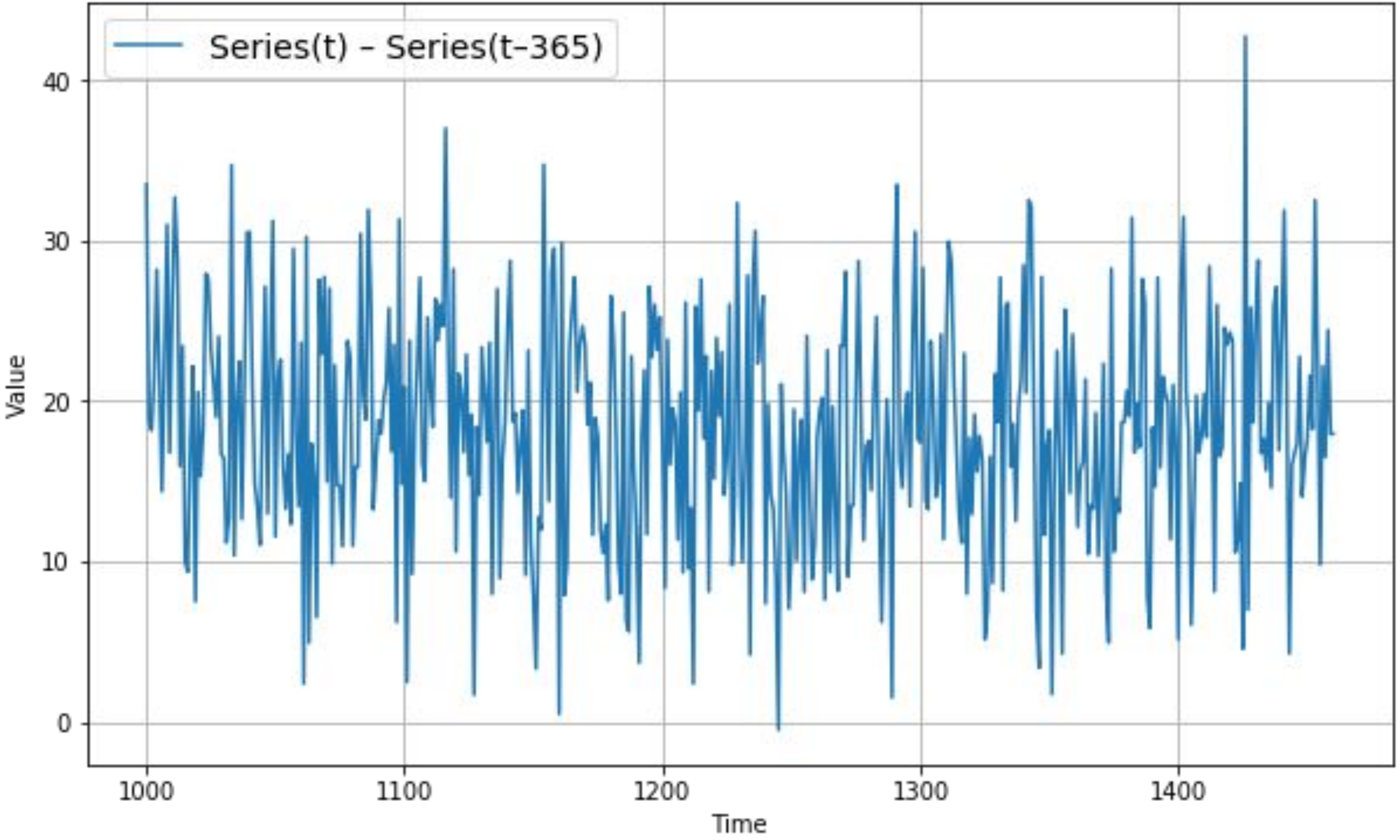
```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

```
mae = tf.keras.metrics.mae(x_valid, naive_forecast).numpy()

5.937908515321673
```

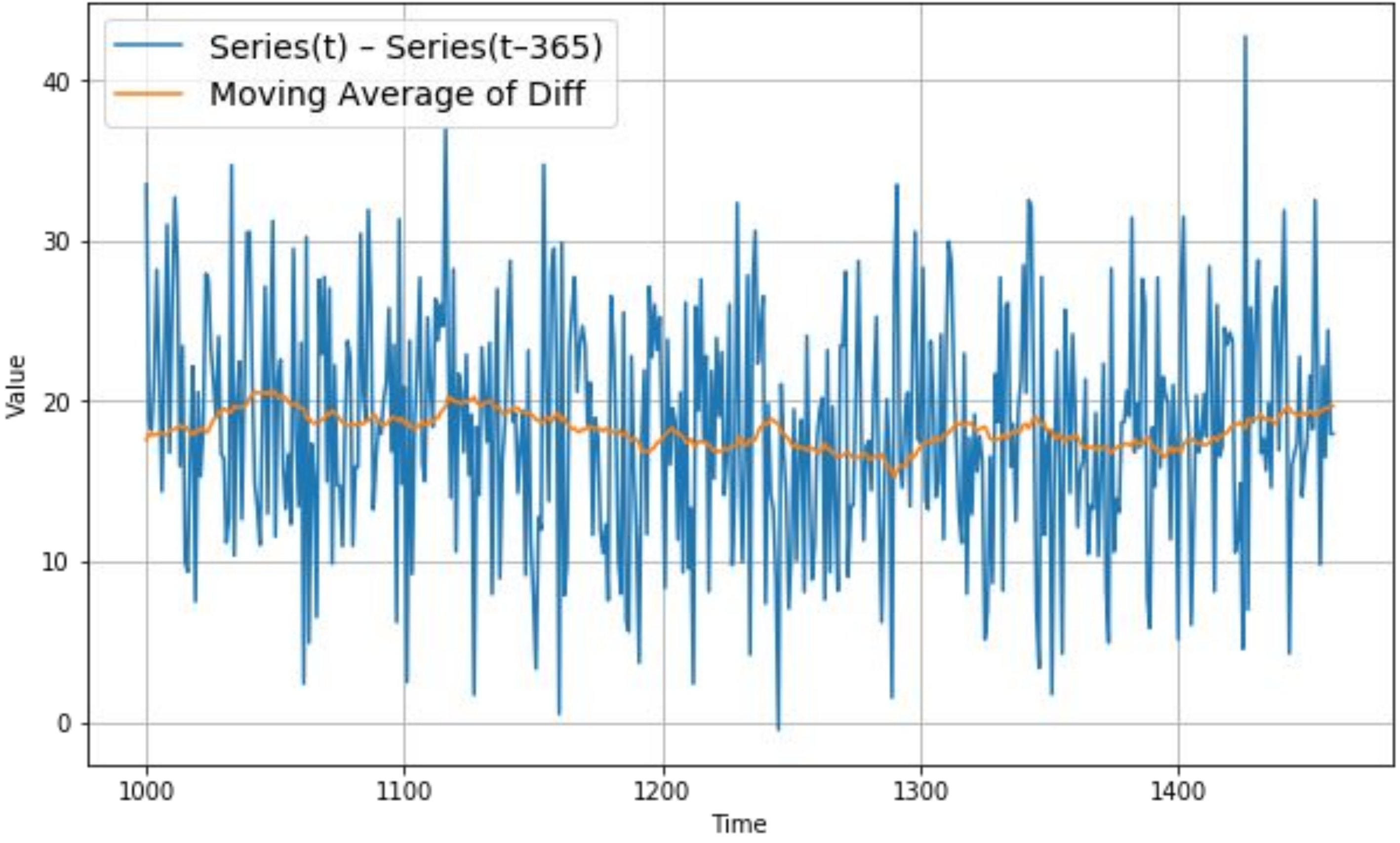
Moving Average



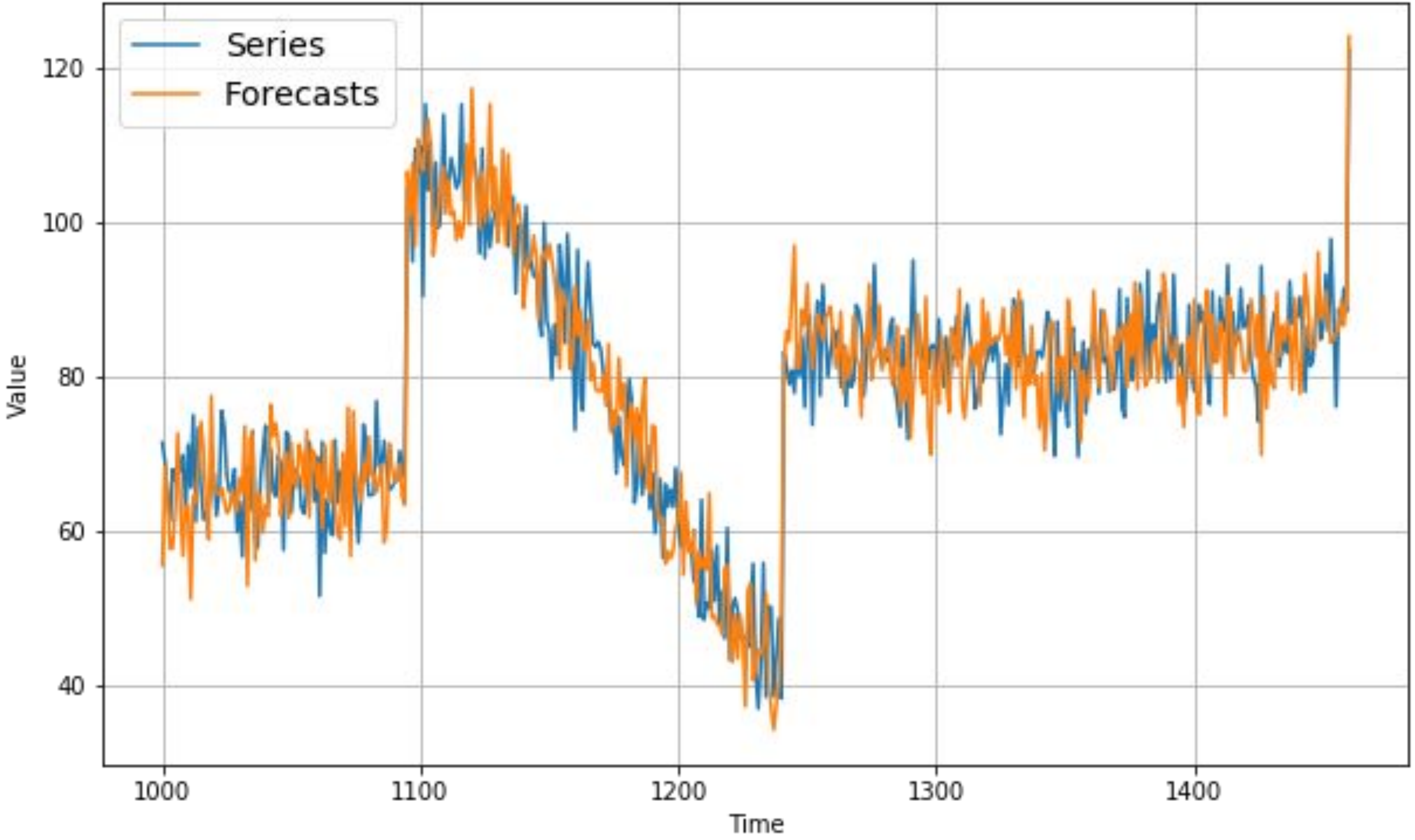
Differencing



Moving Average on Differenced Time Series

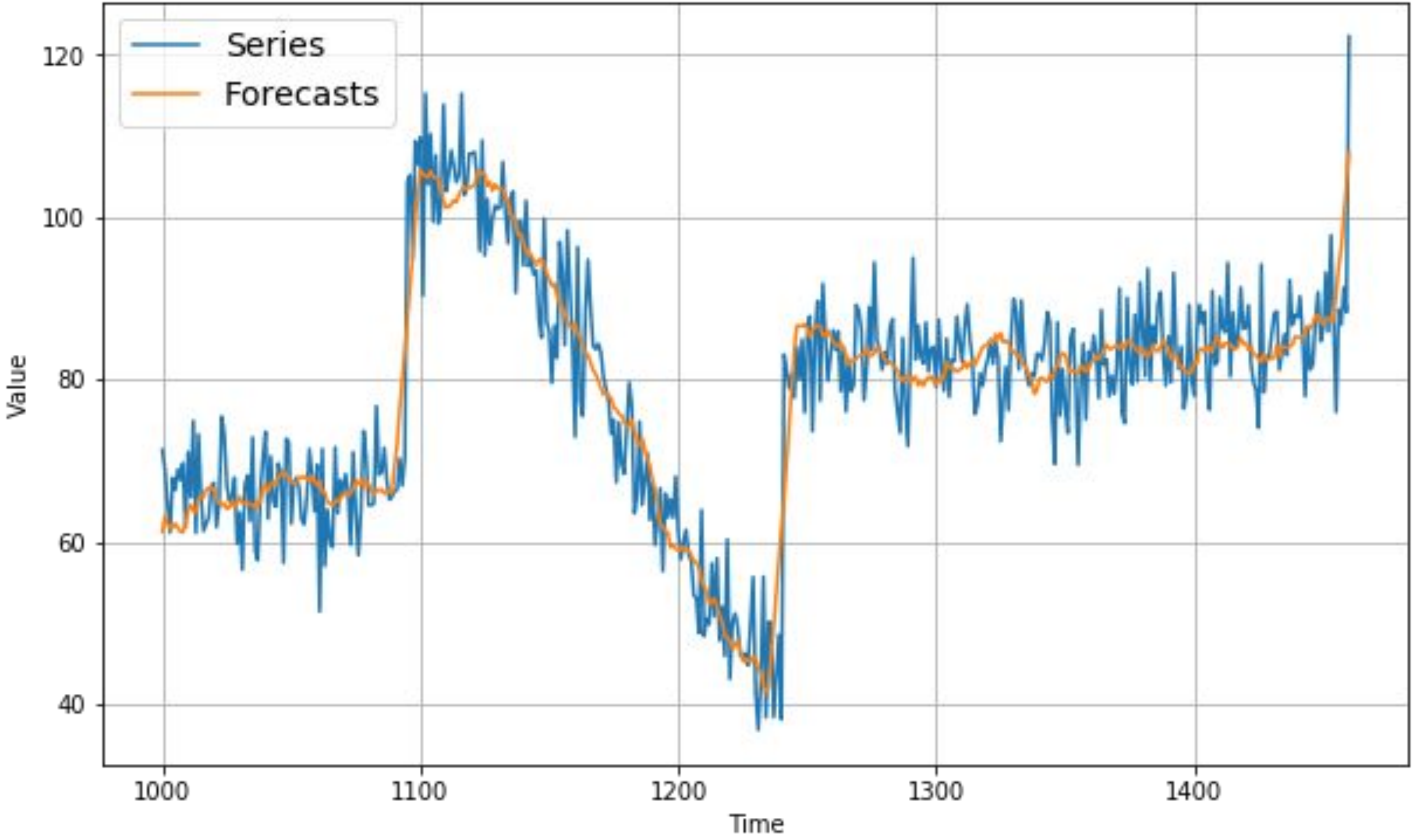


Restoring the Trend and Seasonality



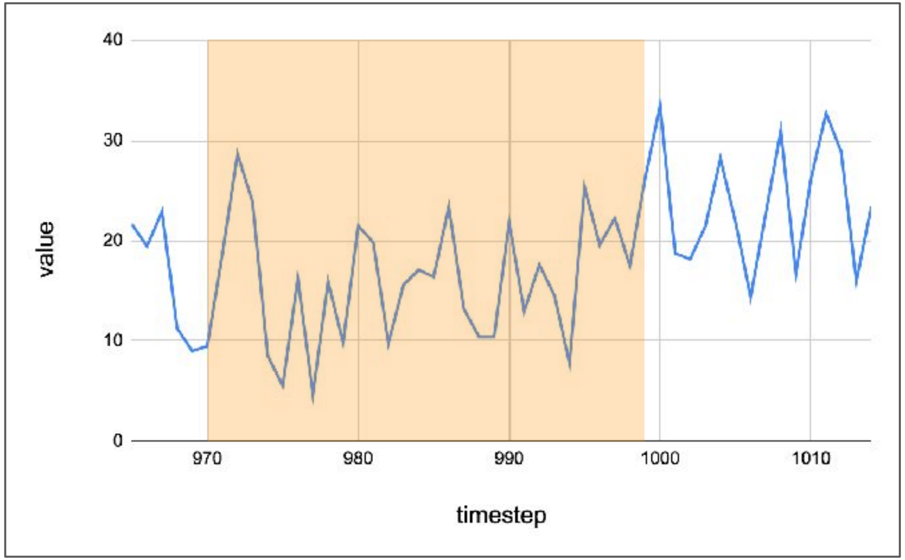
Forecasts = moving average of differenced series + series(t – 365)

Smoothing Both Past and Present Values

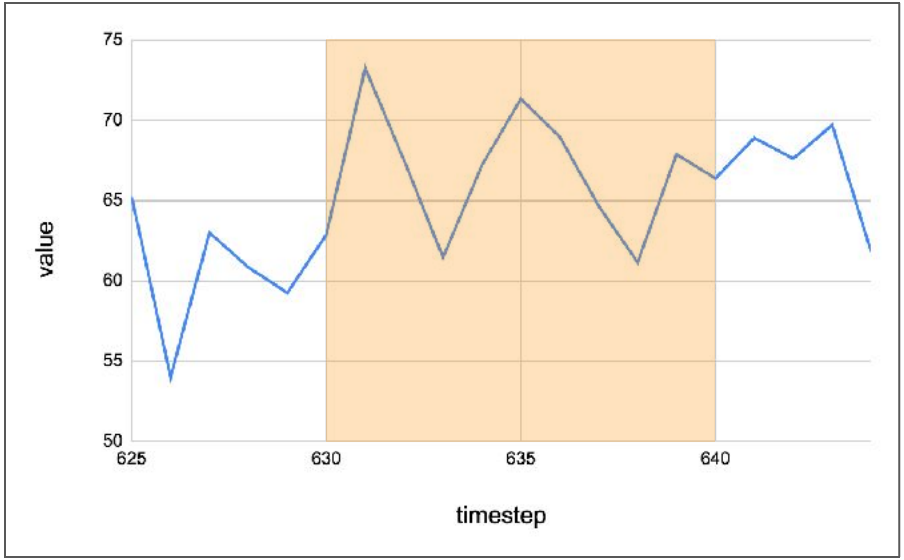


Forecasts = trailing moving average of differenced series + centered moving average of past series (t – 365)

Trailing Moving Average of Differenced Series
(zoomed at t_{1000} , window size = 30)



Centered Moving Average of Past Series (t - 365)
(zoomed at t_{635} , window size = 11)



$TMA_{t1000} = (v_{t970} + v_{t971} + v_{t972} + \ldots v_{t999}) / 30$

$CMA_{t635} = (v_{t630} + v_{t631} + v_{t632} + \ldots. v_{t640}) / 11$

forecast at $t_{1000} = TMA_{t1000} + CMA_{t635}$