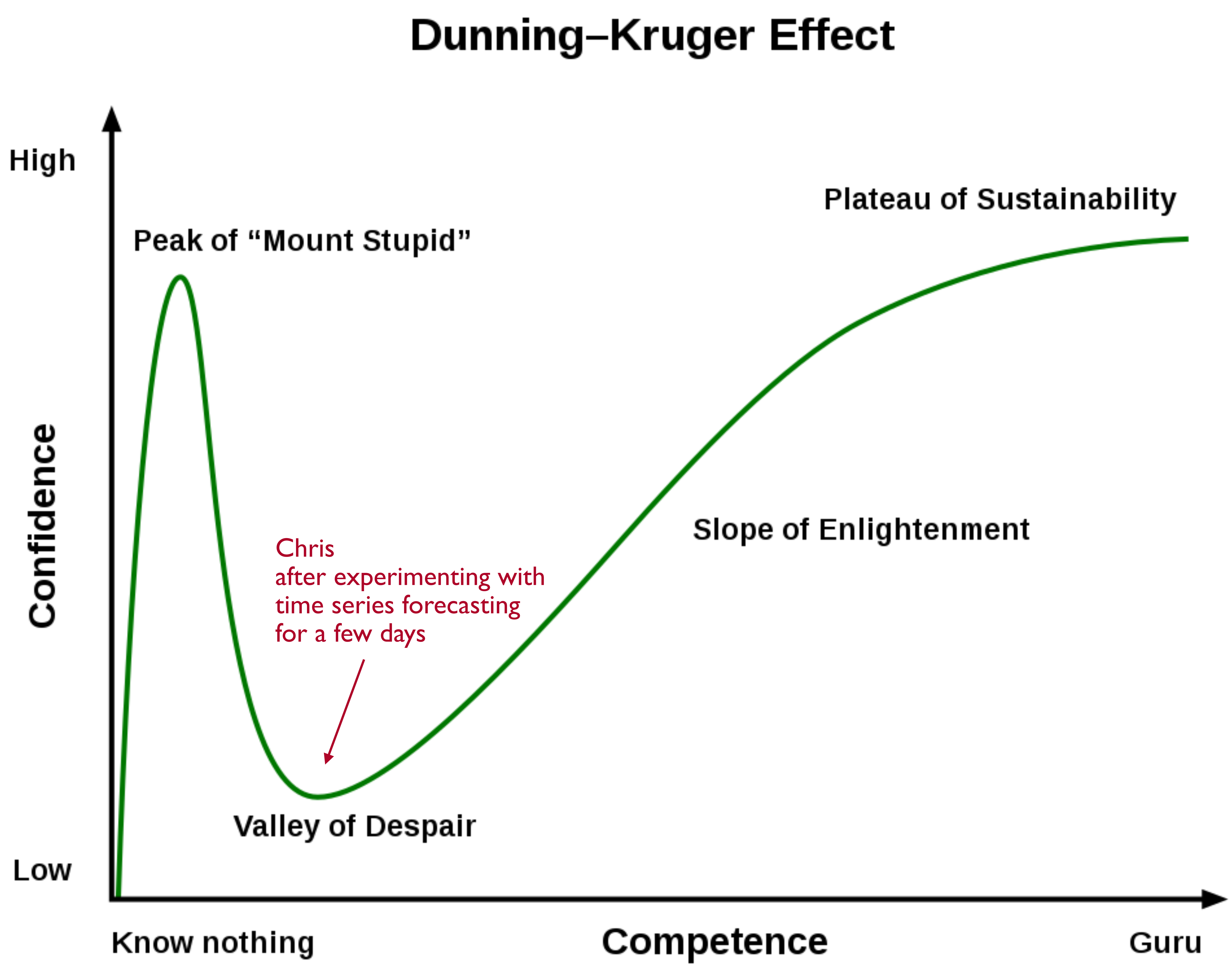
Some (humble) Experiments in Time Series Forecasting

Chris Mair

chris@1006.org
https://www.1006.org

ver. 2020-08-20


Dunning–Kruger Effect
High
Low
Confidence
Peak of “Mount Stupid”
Plateau of Sustainability
Chris
after experimenting with
time series forecasting
for a few days
Slope of Enlightenment
Valley of Despair
Know nothing
Competence
Guru

Time series forecasting

Time series methods

Time series methods use historical data as the basis of estimating future outcomes. They are based on the assumption that past data is a good indicator of future data.

Regression analysis (applied to forecasting)

Regression methods create models that can predict future values of one variable (observation) using information about other variables (predictors), including past observations.

Others

E.g. simulations, …

Ref:
https://en.wikipedia.org/wiki/Forecasting

Part 1: let's pick two time series methods

ARIMA

ARIMA = autoregressive integrated moving average

This is a 'classical' method. It is mostly used in econometrics.

Ref:
https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average and check out the the links to the sub-models ARMA, AR and MA

LSTM

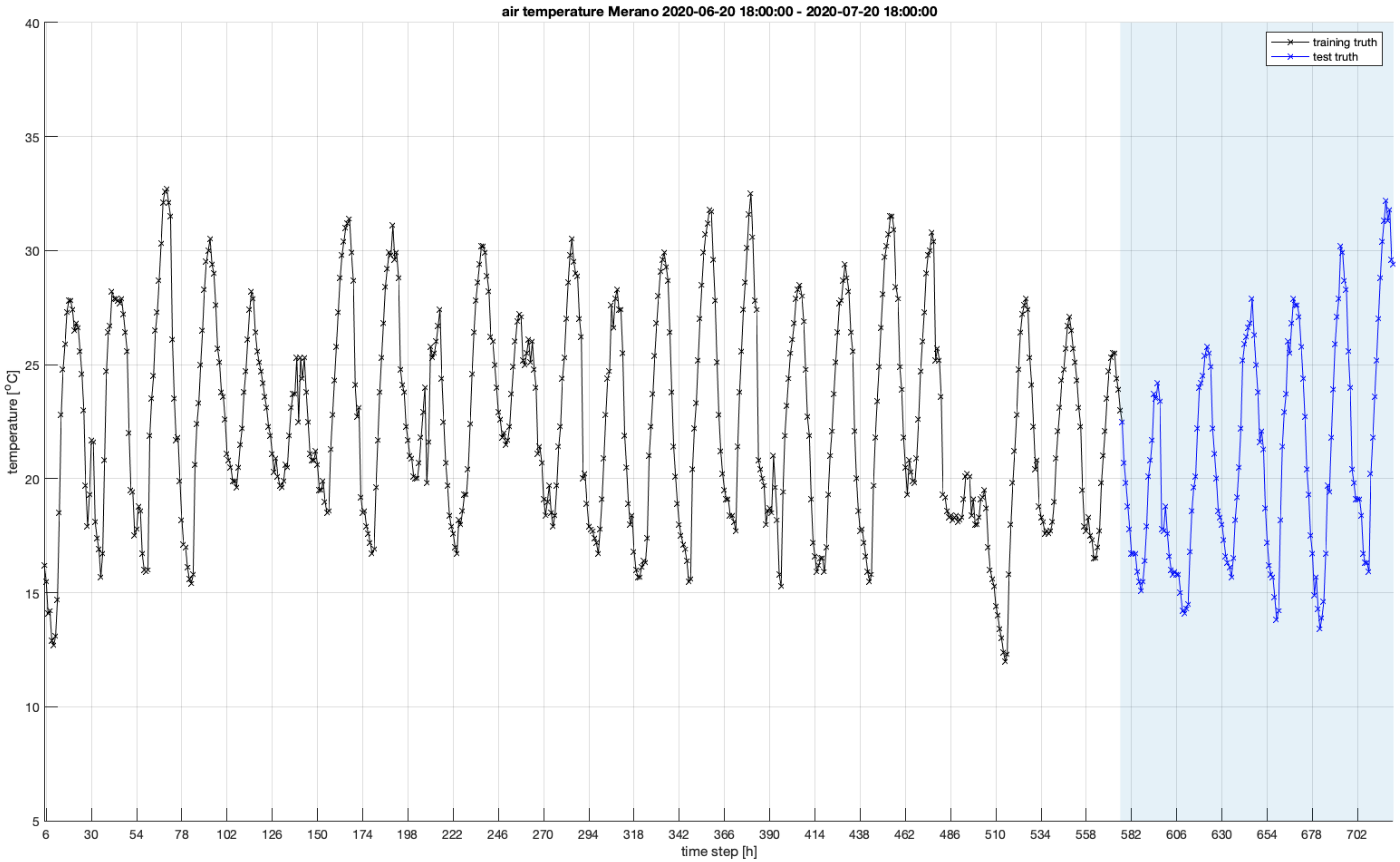
LSTM = long short-term memory, which is a type of RNN (recurrent neural network)

This is a machine learning ('deep learning') method. LSTMs are applied to sequences and typically used for classification (e.g. speech recognition or automatic classification of ECGs), but can also used for forecasting.

Ref:
http://www.deeplearningbook.org/contents/rnn.html and
https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (thx Tiziano :)

Test data: the last 30 days of air temperature in Merano

Can we predict the last 6 days?



Temperature data source: https://analytics.opendatahub.bz.it/tml

Tools

I used MATLAB® for the data handling, regressions, plotting and machine learning and R for ARIMA.

However, LSTMs are available also in the popular frameworks that can be used, such as TensorFlow 2.

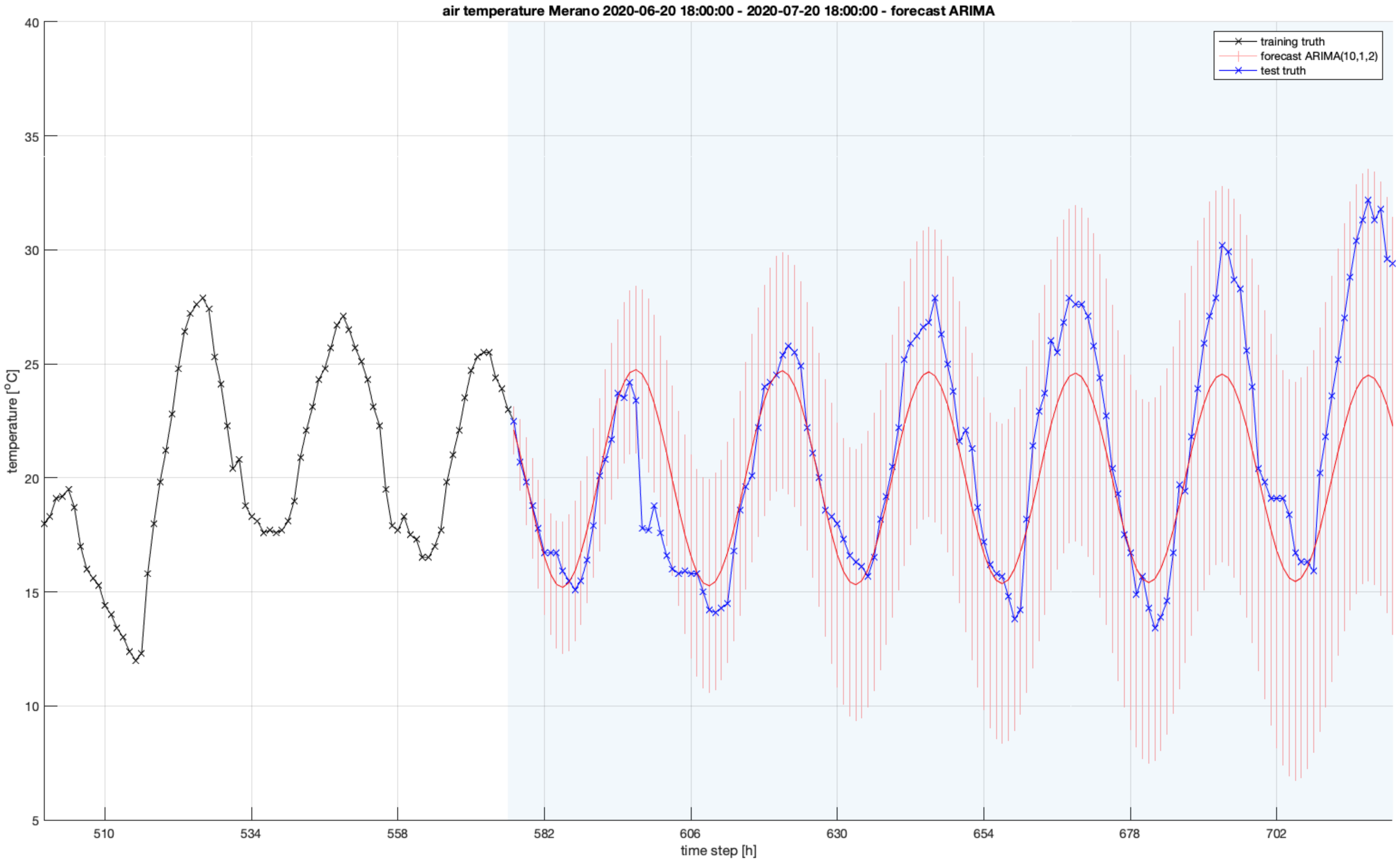
ARIMA

`arima.m` and `arima.r`

rmse = 2.70

Manually tuned parameters:
10, 1, 2

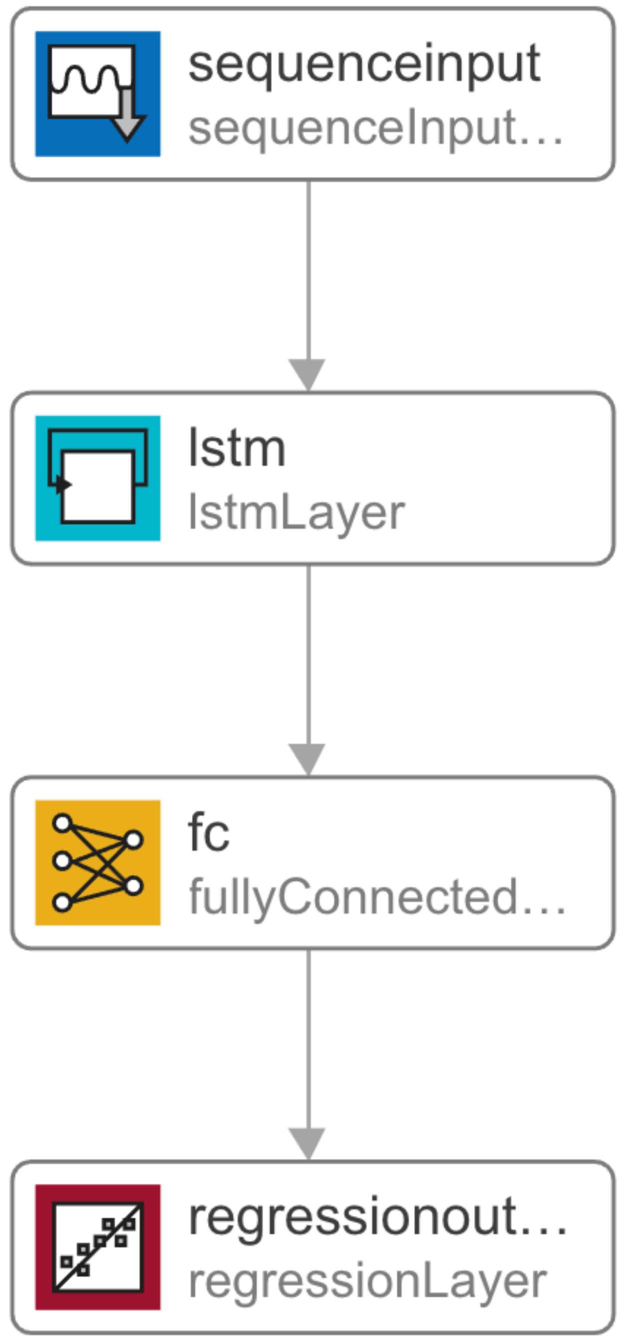
(could be improved? ;)

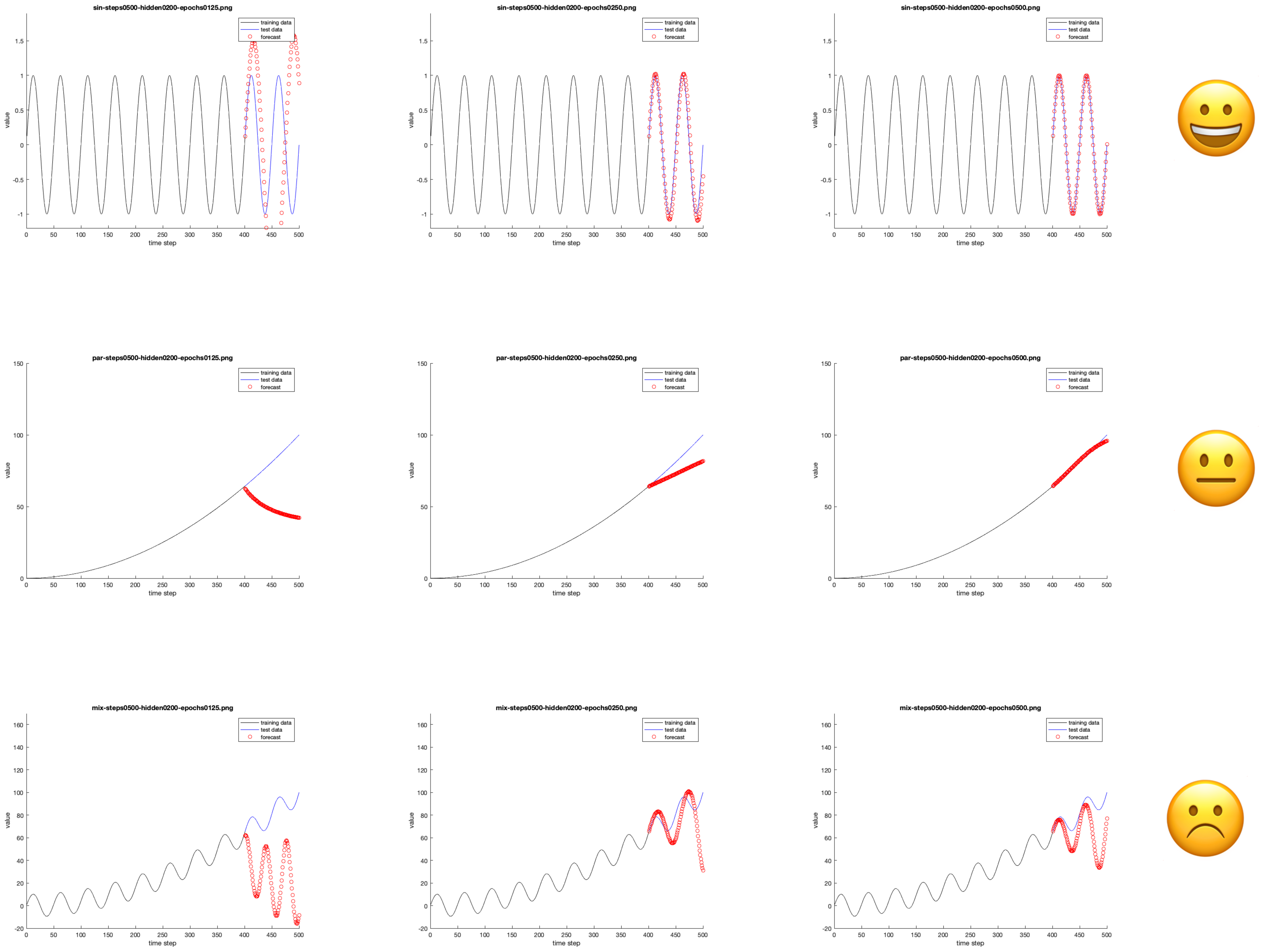


Playing with LSTMs

artificial_datasets.m

non_stationary.m



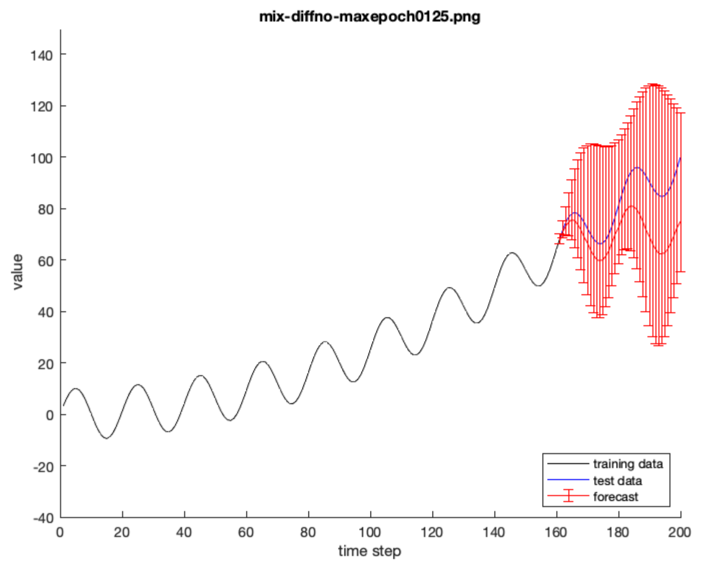


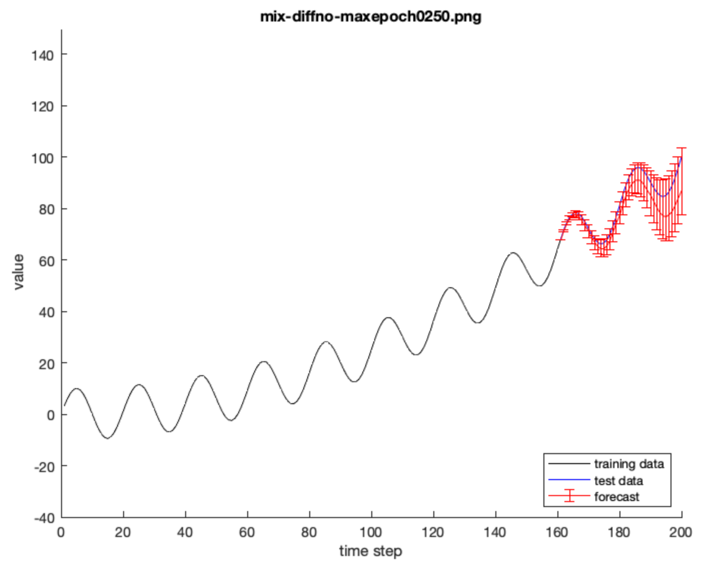
longer training

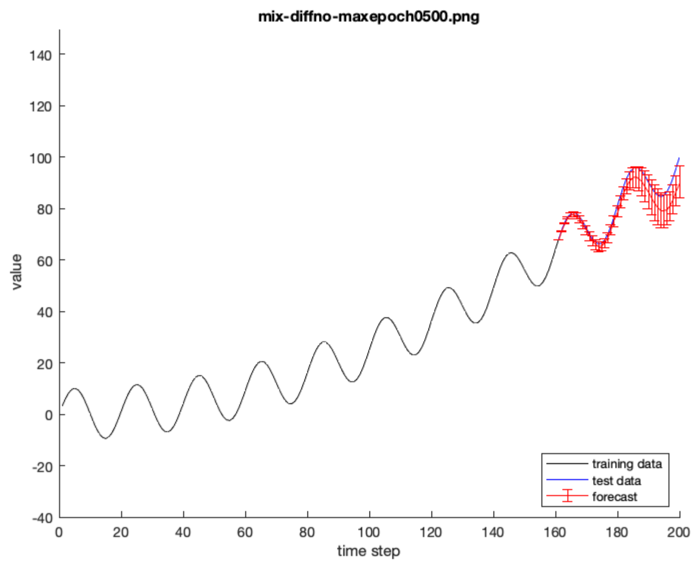
Playing with LSTMs

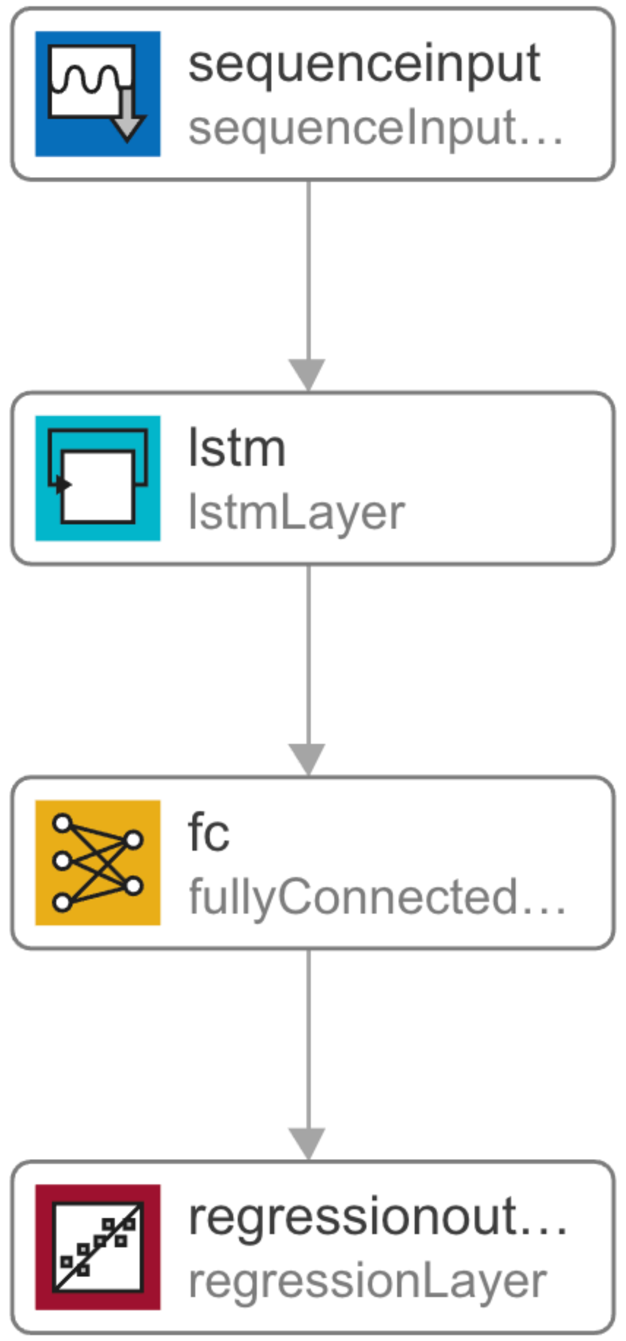
`artificial_datasets.m`

`non_stationary.m`

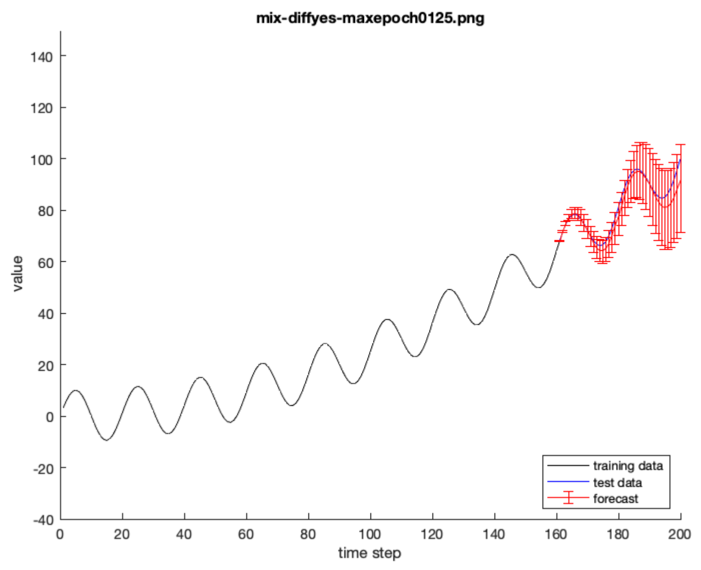


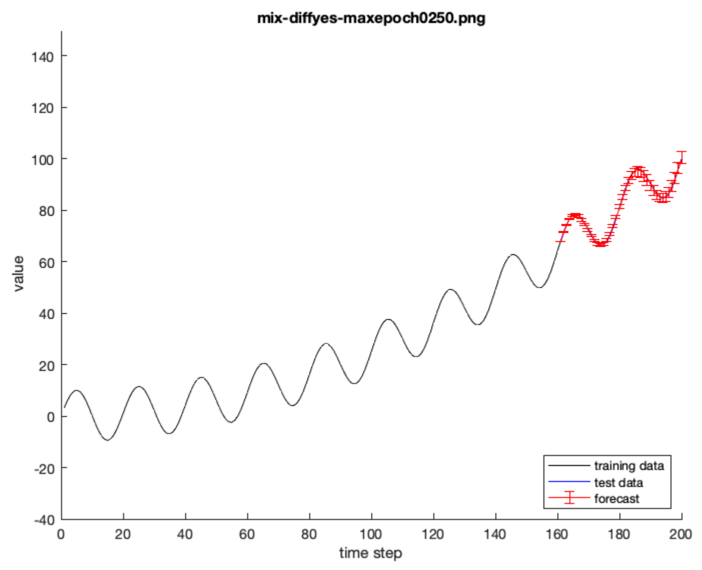


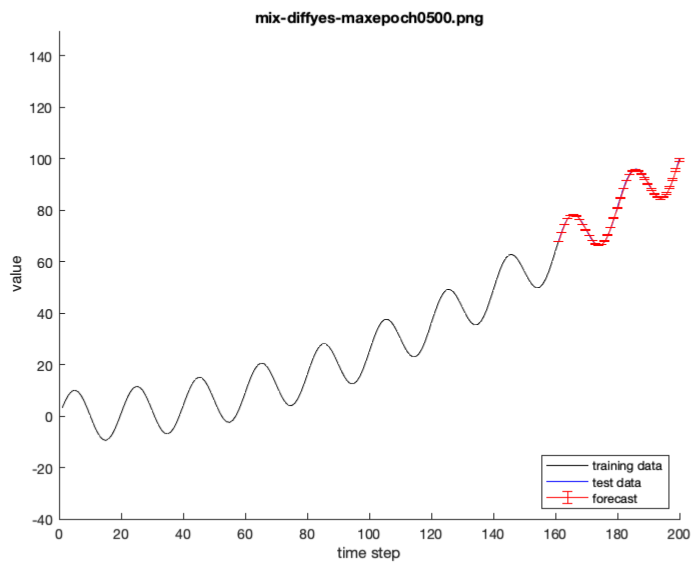




differentiation does the trick!
(also: more training runs to see the spread...)







longer training

LSTM

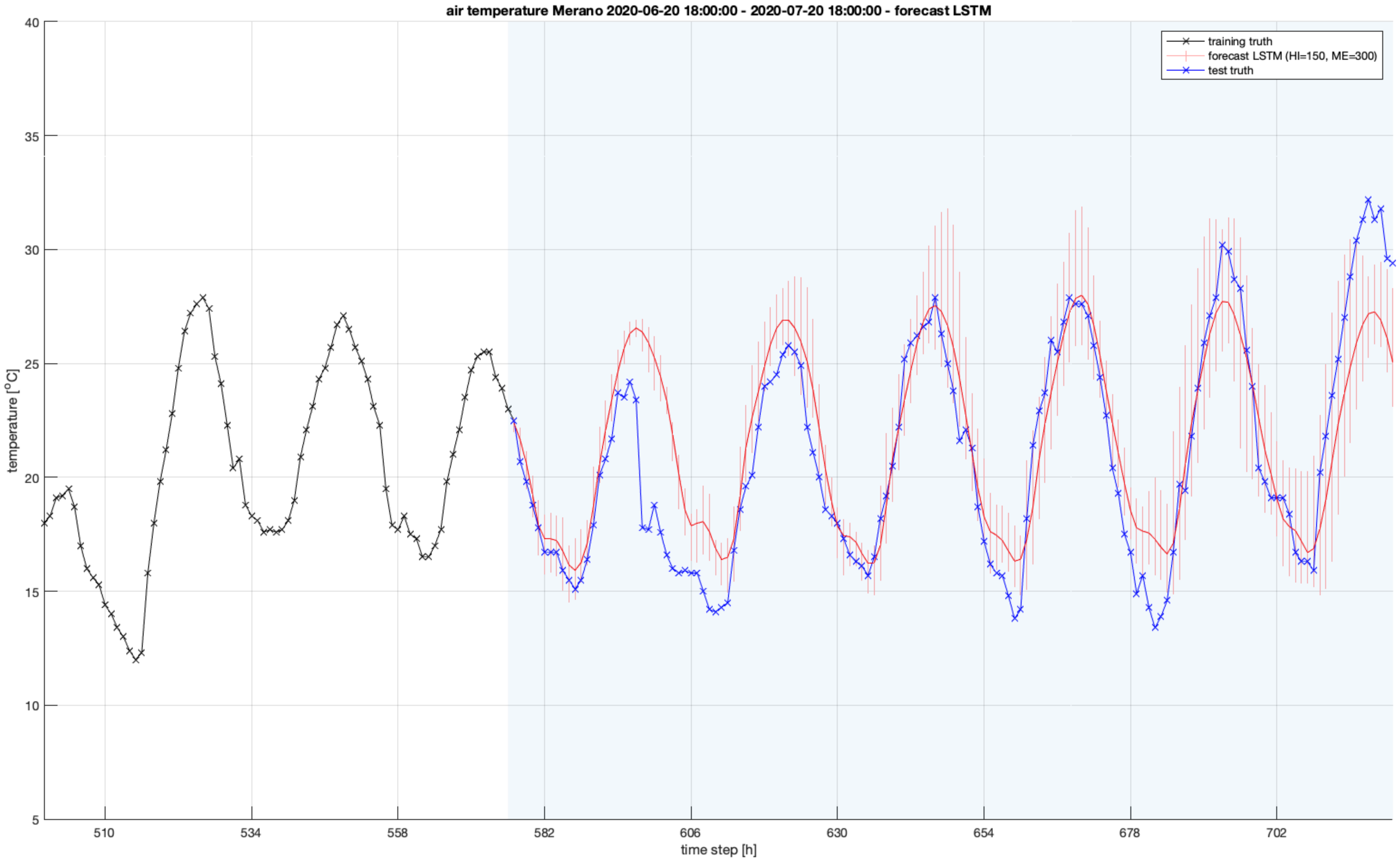
lstm.m

rmse = 2.88

for 150 hidden layers
and 300 epochs

error bars show spread
from 8 runs

each run takes about
1 min on my notebook



Part 2: let's try regression analysis

The sample dataset is open data from waste water plants in Trentino.

Source: http://adep.heidix.net/opendata/index.html

`get-data.js`
to download all the measurements (JSON)

`process-data.js`
to extract the timestamp *ts*, the average incoming waste water flow rate *q* and temperature *temp* (CSV)

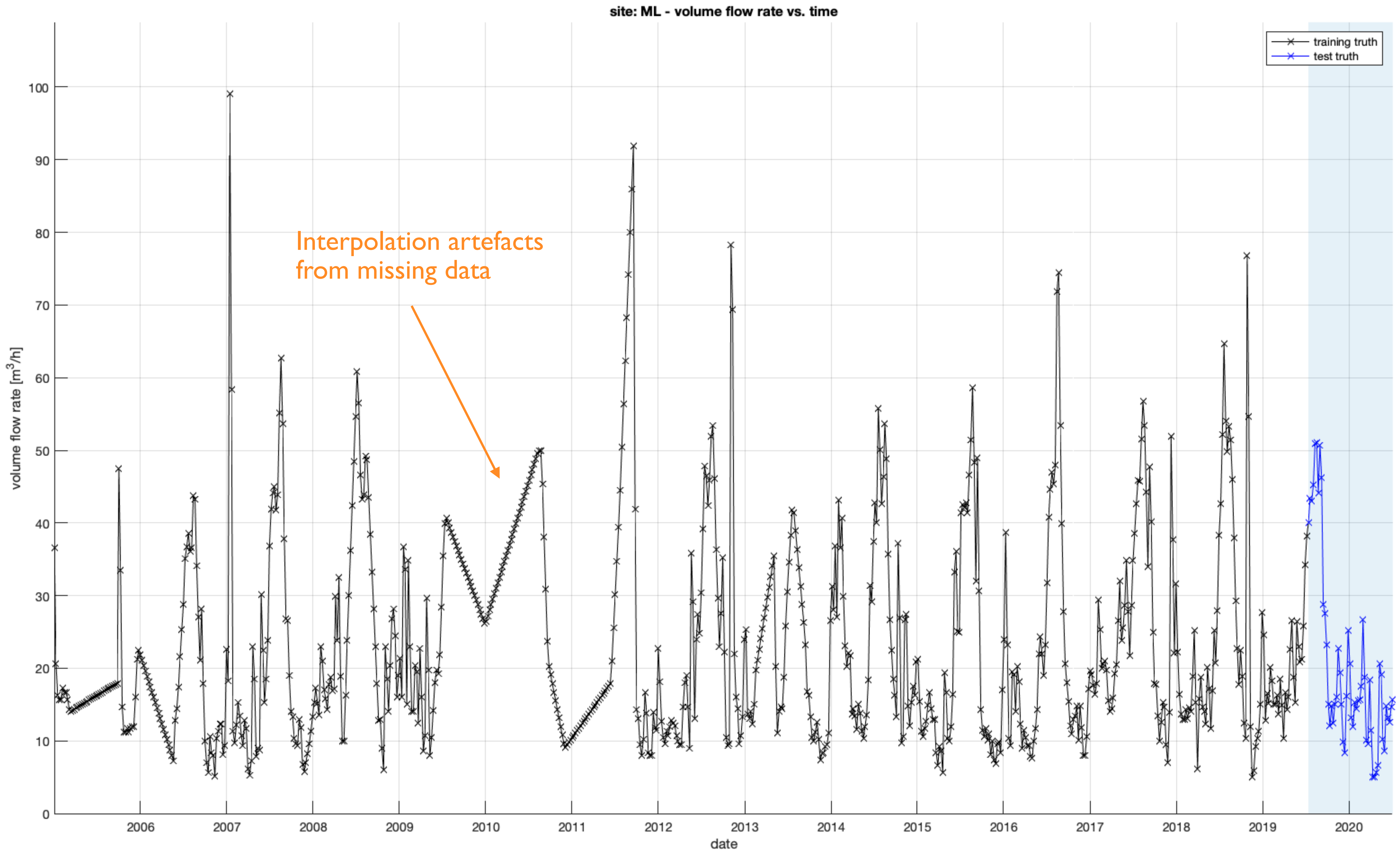
Manual cleanup:

- site PE and PR had duplicated data on 2012-06-19;
- one timestamp was 24:00 instead of the usual 00:00 and MATLAB doesn't like that;
- sites RM, TM and VT have too few data.

We got data from 67 plants (see `adep/*.csv`).

Test data: average incoming waste water volume flow rate (q)

Can we predict the last year? Sample site shown is Molveno (`adep_plot.m`).



Data source: http://adep.heidix.net/opendata/index.html

Regression

Regression methods create models that can predict future values of one variable (observation) using information about other variables (predictors), including past observations.

Let's pick the following predictors for `q`:

- day of year (1 - 365) `doy`
- flow rate of 1 week ago: `q1`
- flow rate of 52 weeks ago: `q52`
- temperature of 1 week ago: `temp1`
- temperature of 52 weeks ago: `temp52`

We use regression models to fit a function that gives `q` as a function of (`doy`, `q1`, `q52`, `temp1`, `temp52`).

Using this function we can predict the current `q` from past data. To iteratively predict future values of `q`, we need future values of the temperature too!

So we create a second model for the temperature with predictors:

- day of year (1 - 365) `doy`
- temperature of 1 week ago: `temp1`
- temperature of 52 weeks ago: `temp52`

Regression: linear model for Molveno (ML)

reg_lm.m

```
Linear regression model:
    q ~ 1 + doy + q1 + q52 + temp1 + temp52

Estimated Coefficients:
                    Estimate        SE         tStat        pValue
                   __________    _________    _______    __________

    (Intercept)         -1.93       1.0933    -1.7653      0.077925
    doy            -0.0044464    0.0028848    -1.5413       0.12366
    q1                0.71664     0.024492      29.26    3.0102e-126
    q52               0.06057     0.024177     2.5053      0.012445
    temp1             0.25244      0.13207     1.9114      0.056331
    temp52            0.30138       0.1338     2.2524      0.024583


Number of observations: 758, Error degrees of freedom: 752
Root Mean Squared Error: 7.9
R-squared: 0.705,  Adjusted R-Squared: 0.703
F-statistic vs. constant model: 360, p-value = 8.71e-197
```

```
Linear regression model:
    temp ~ 1 + doy + temp1 + temp52

Estimated Coefficients:
                    Estimate         SE         tStat       pValue
                   __________    __________    _______    __________

    (Intercept)       0.22417       0.13364     1.6774      0.093873
    doy            -0.0018937    0.00035168    -5.3847    9.7025e-08
    temp1             0.88316       0.01577     56.004    7.0562e-271
    temp52            0.12581      0.015961     7.8822    1.1263e-14

Number of observations: 758, Error degrees of freedom: 754
Root Mean Squared Error: 0.971
R-squared: 0.949,  Adjusted R-Squared: 0.948
F-statistic vs. constant model: 4.63e+03, p-value = 0
```

sort of works for *q*…

works well for `temp`

Regression: linear model for Trento Nord (TN)

reg_lm.m

```
Linear regression model:
    q ~ 1 + doy + q1 + q52 + temp1 + temp52

Estimated Coefficients:
                   Estimate        SE        tStat       pValue
                   _________    ________    ________    __________

    (Intercept)       285.55      48.027      5.9457     4.276e-09
    doy             0.040456    0.058401     0.69272       0.48871
    q1                0.6936    0.027944      24.821    5.9838e-99
    q52            0.0091534    0.028932     0.31638       0.75181
    temp1            -3.4879      2.7804     -1.2545       0.21008
    temp52            1.9074      2.5784     0.73978       0.45967


Number of observations: 732, Error degrees of freedom: 726
Root Mean Squared Error: 142
R-squared: 0.5,  Adjusted R-Squared: 0.497
F-statistic vs. constant model: 145, p-value = 9.28e-107
```

```
Linear regression model:
    temp ~ 1 + doy + temp1 + temp52

Estimated Coefficients:
                    Estimate           SE          tStat        pValue
                   __________    __________    ________    __________

    (Intercept)      0.010588       0.11325    0.093491       0.92554
    doy            -0.0034413    0.00026836     -12.823    4.3673e-34
    temp1             0.97033      0.012015      80.757             0
    temp52           0.064013      0.011129      5.7519    1.2991e-08

Number of observations: 732, Error degrees of freedom: 728
Root Mean Squared Error: 0.669
R-squared: 0.974,  Adjusted R-Squared: 0.973
F-statistic vs. constant model: 8.92e+03, p-value = 0
```

thinks are tricky for other sites (such as Trento Nord) where there is not so much a seasonal component…

Side-project: playing with the linear model for the temperature

Tiziano Lattisi has made a Jupyter Notebook where you can play with the linear model for the temperature.

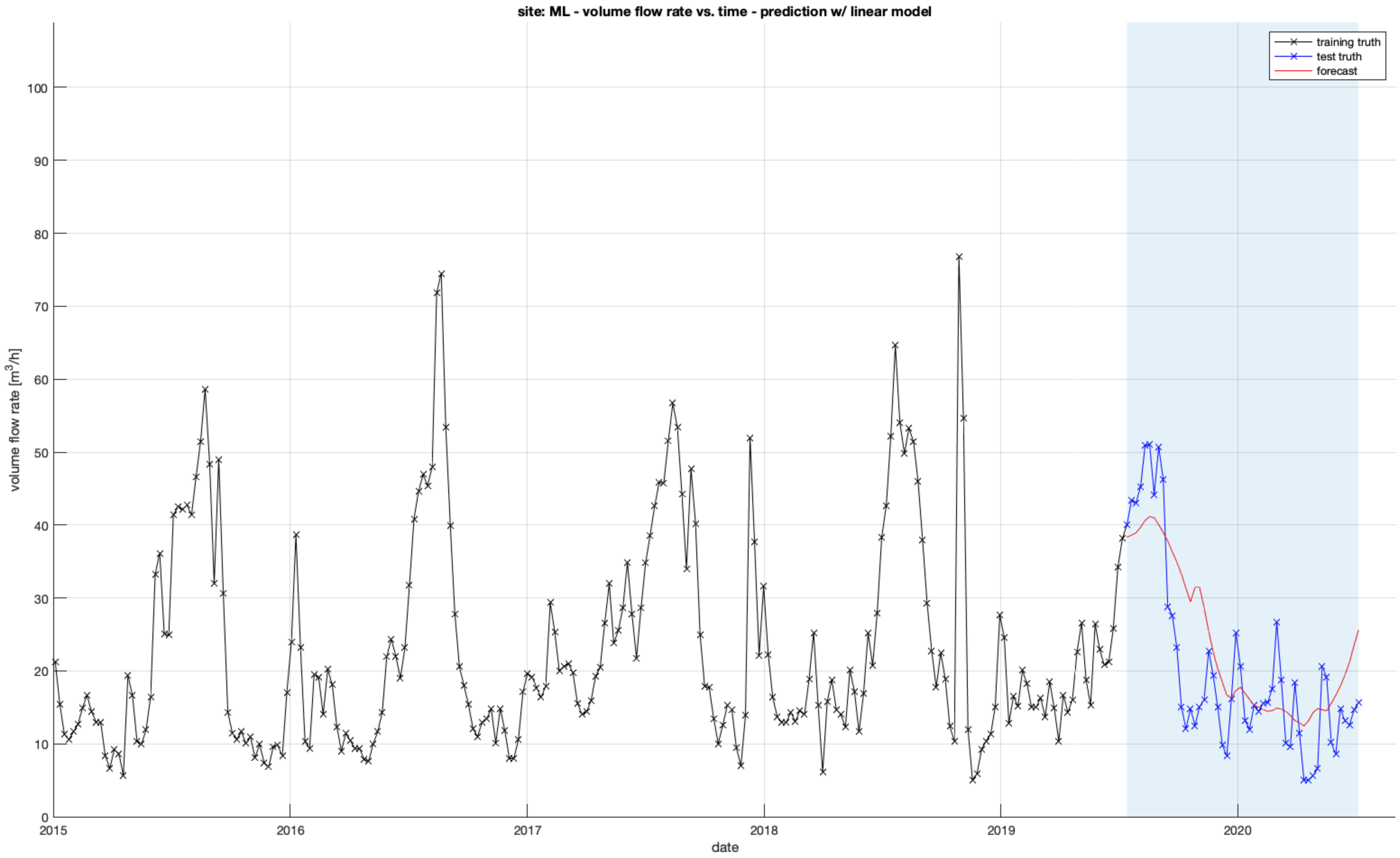
https://colab.research.google.com/drive/1UZY0EQEWS9BT5biCea5GIo1CJSUq1Zw6?usp=sharing

In particular he notes that instead of using the day of the year (`doy`) one should use 52 columns in a one-hot encoding to indicate which week a measurement was made.

He also notes that the strongest predictor is (not surprisingly) `temp1` alone.

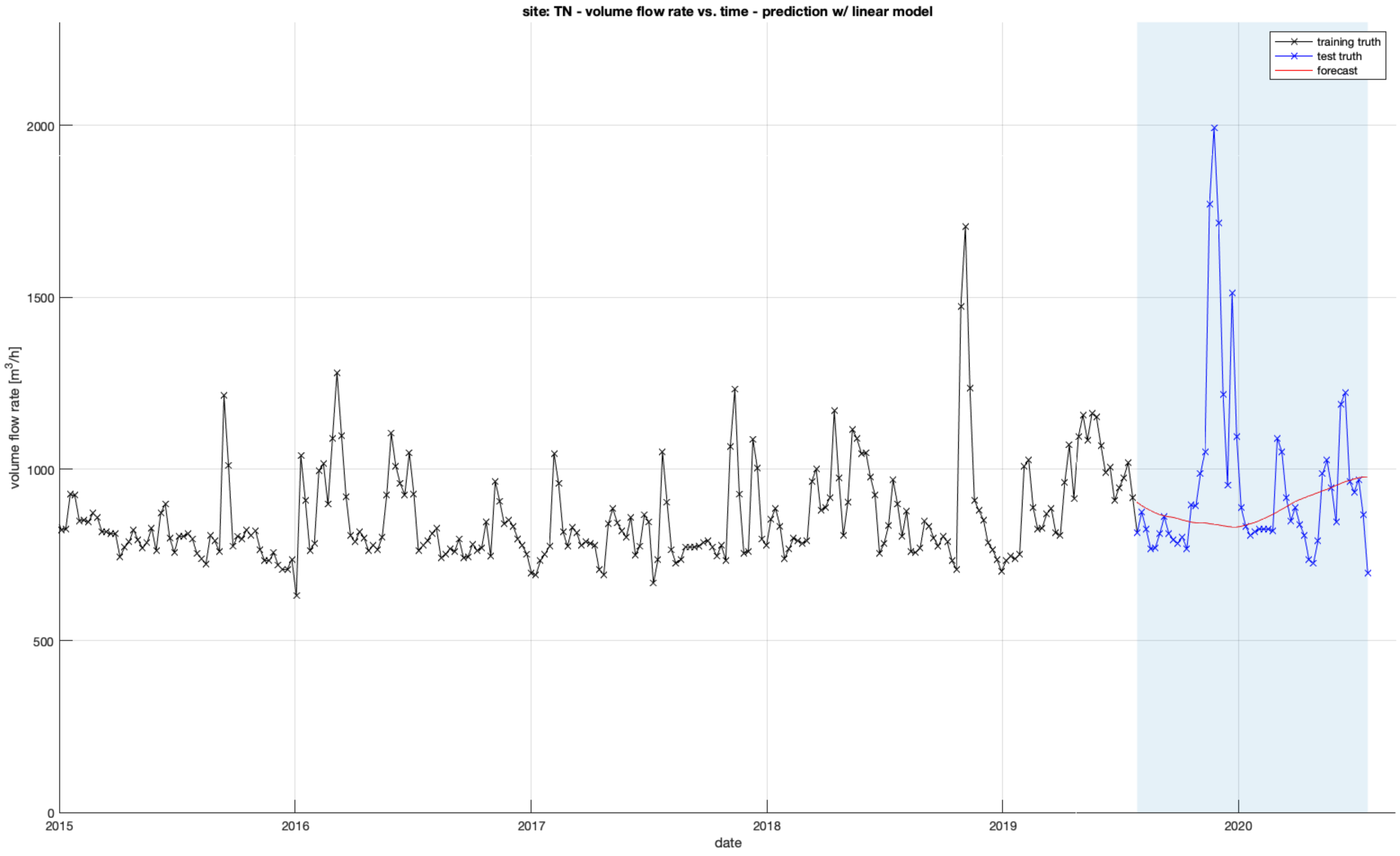
Regression: linear model for Molveno (ML) - results

`reg_lm.m`



Regression: linear model for Trento Nord (TN) - results

`reg_lm.m`



Regression: neural network model

`reg_nn.m`

We repeat the exact same forecasting algorithm with the model for q and $temp$, but instead of using a linear model, we pick a shallow neural network with three blocks having 1, 4 and 1 hidden layers in series.

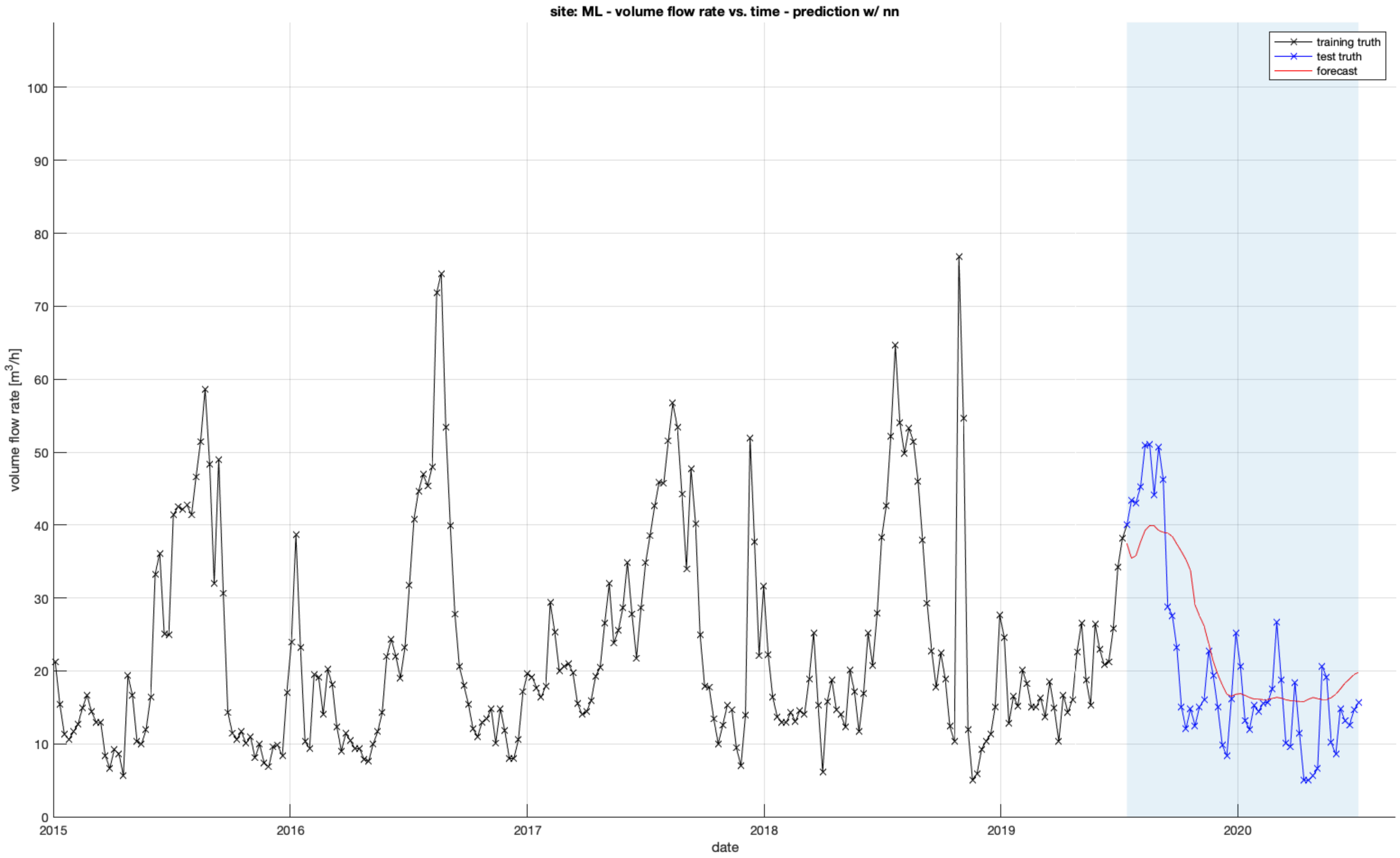
The training of this network is not reproducible (unless one seeds the random number generator with a constant), as the network weights are initialised randomly.

One could probably improve training a bit by not picking MATLAB's default parameters, but I didn't investigate this further. I just did the forecast 15 times and averaged the single runs.

As this is a shallow network, a run is a bit less than a second on my notebook.

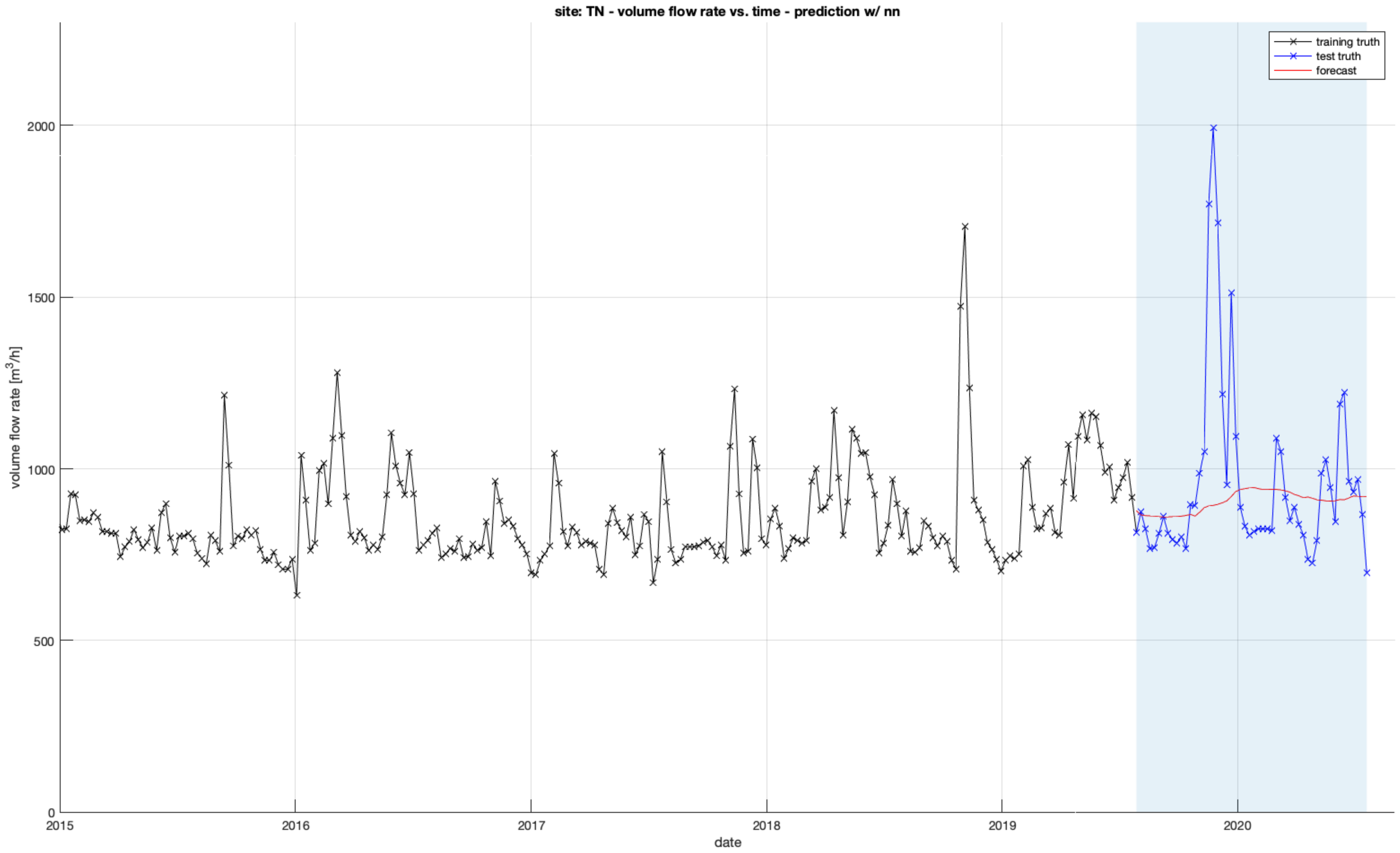
Regression: neural net model for Molveno (ML) - results

reg_nn.m



Regression: neural model for Trento Nord (TN) - results

`reg_nn.m`



Third party illustrations:

Dunning–Kruger Effect illustration:

https://commons.wikimedia.org/wiki/File:Dunning%E2%80%93Kruger_Effect_01.svg

Network Layer Blocks Illustration:

Screenshot from MATLAB®

Special thanks to Tiziano Lattisi.