CS 229: Predicting Real GDP Growth Using Economic Indicators

Rachael Wang & Andrew Conkey, rachaelw@stanford.edu & aconkey@stanford.edu

June 3, 2021

Part 1: Abstract

Accurate prediction of GDP change and other country-level economic indicators is of importance to both policymakers and financial markets. We analyze IMF data for over 140+ countries between 2006-2014 to build a variety of models for GDP growth prediction.These include: (1) OLS regression, (2) Feed Forward Multi-layer Perception Neural Network, (3) OLS as an Input to a Feed Forward Neural Network and (4) Elman Recurrent Neural Network, as well as some regularization and feature selection refinements of a subset of these models. Our OLS model had a 10.344 MSE on the training set and 15.289 MSE on the test set, while our FFNN had a 15.26 MSE on the training set and 8.84 MSE on the test set. Variance in converged model parameters stemming from local maxima issues make it difficult to confidently recommend one model or another. A further, more computationally intensive study would attempt to run many more trials to better characterize the average and maximum accuracy of these models, and make use of more complex models to combat underfitting.

Part 2: Introduction

The growth rate of real GDP is used as an indicator of the health of an economy. An increase in real GDP is interpreted as a sign that the economy is doing well. When real GDP is growing strongly, employment is likely to be increasing, whereas when GDP is shrinking, employment often declines. It's hard to know how GDP will change from year to year. Through it's World Economic Outlook (WEO) database, the IMF provides one of the highest quality and most comprehensive sources of economic data. We will attempt to predict GDP growth based on this data source. Accurate prediction of GDP is important to both policymakers and financial markets, and machine learning applications to this area in the past have been sparse, and often only focused on a limited subset of countries. We instead aim to make predictions for a broad set of countries, including developing economies, in which more complex economic indicators are not always available.

Part 3: Related Work

There are two papers we referenced while working on this project. The first was *Improvements in the World Bank's Ease of Doing Business Rankings: Do They Translate into Greater Foreign Direct Investment Inflows?*, by Dinuk Jayasuriya. The second article we referenced was *Predicting Real GDP Growth* by Joohi Rana. Please see all other papers, packages and articles referenced in **Part 9: Resources and Index**.

Part 4: Dataset and Features

The dataset is sourced from the IMF World Economic Outlook database from April 2021 (`https://www.imf.org/en/Publications/WEO/weo-database/2021/April`). We narrowed our scope to only include data from 2006-2014, then dropped variables for which there were too many missing observations and also dropped clearly interdependent variables (example: we would drop one out of current account, GDP, and current account as a percentage of GDP). We then dropped countries for which there was still missing data among the remaining features. This left us with a total of 18 variables for 146 countries in each of 9 years, a reduction from 45 variables and 195 countries in the original dataset. We prioritized retaining a large number of countries over retaining a large number of variables, as otherwise we would have had to limit ourselves to only the most developed economies. Included in these features is GDP growth, also our outcome variable. See the appendix for a full list of countries and WEO subject codes for features. To these features we also added a constant feature.

We attempt to predict GDP in a given year based on variable values from the four years directly prior. To predict GDP growth in 2010, for example, we would use variable values from 2006-2009. For each country-year pair, therefore, we have $4 \cdot 18 = 72$ features (one per year-variable pair) from which to predict GDP growth in a given year. For the test set, we set aside GDP growth values for 2014. This leaves 4 years (2010-2013) with at least 4 years of preceding data that can be used in the training set. So we have $4 \cdot 146 = 584$ observations (one per country-year) in the training set and 146 in the test dataset.

Part 5: Methods

0.1 Linear Regression Model:

As an initial model, we fit an OLS regression to the data using the normal equations. The model and estimation method used are:

$$Y = X\theta$$
$$\hat{\theta} = (X^T X)^{-1} X^T Y.$$

See the CS229 notes for further information.

0.2 Feed Forward Multilayer Perceptron (Neural Network):

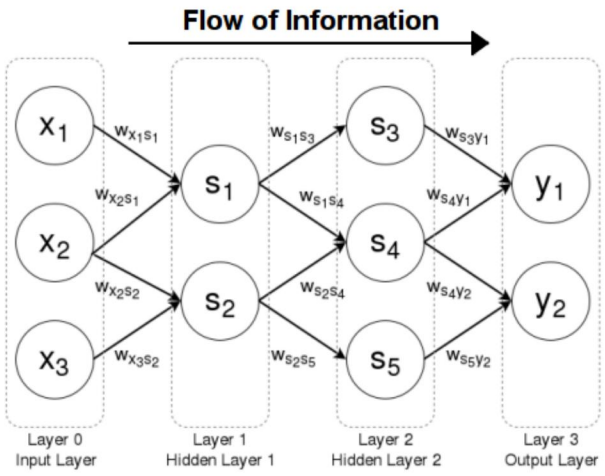
We implemented a fully-connected feed forward two-layer neural network that takes in as input $x_1, x_2, x_3, ...$ and outputs the activation function:

$$h_W, b(x) = f(W^T x) = f(\sum W_i x_i + b_i)$$

We are using the hyperbolic tangent, or tanh, function:

$$f(z) = tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

These activation functions from the hidden layer are then linearly combined into an output for the preiction.



0.3 Elman Recurrent Neural Networks:

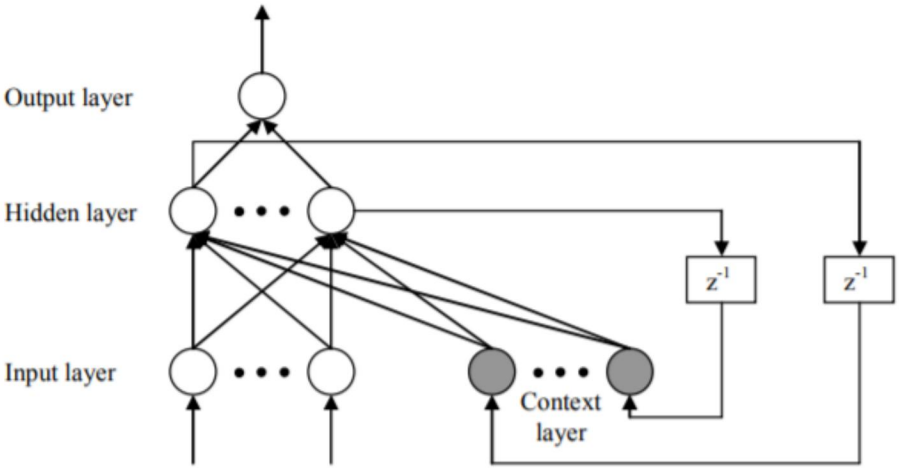


Figure 4. Architectural graph of Elman neural network

Elman neural network (ENN) is a recurrent neural network (RNN). Unlike our previous implementation of a neural network. ENN has additional inputs from a hidden layer, which forms a new layer-the context layer. The standard back-propagation (BP) algorithm used in ENN is called Elman back-propagation algorithm (EBP). ENN can be applied to solve prediction problems of discrete time sequences. In this case we are using it on input data from 2006 - 2012.

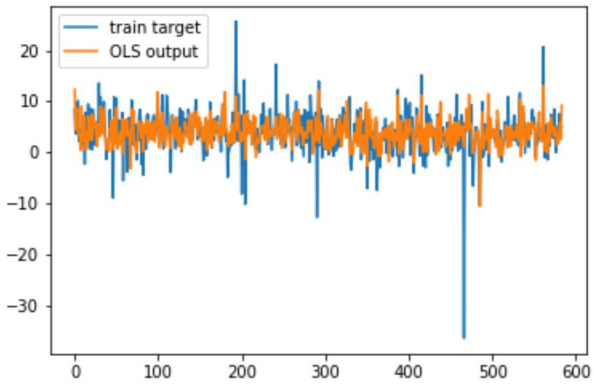
Part 6: Experiments

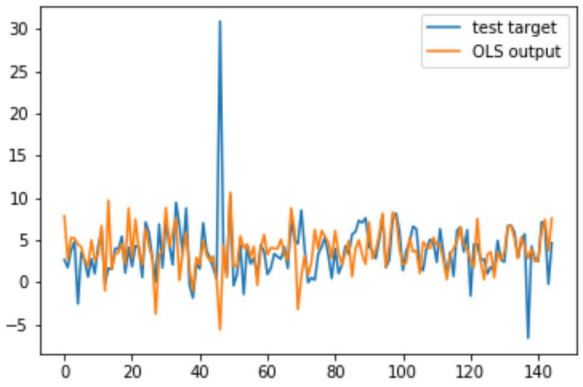
0.4 Linear Regression Model:

0.4.1 Implementation:

We used a linear regression for our original implementation, with features as described in part 4 and the appendix. Because we could use the normal equations, this was much more computationally efficient than the other methods. It should also provide a good baseline for performance.

0.4.2 Outcomes:





0.4.3 Analysis and Results:

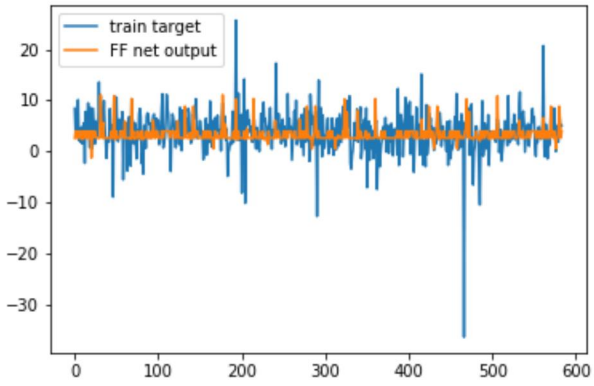
MSE on the training dataset: 10.344
MSE on the test dataset: 15.289
Squared error corresponding to a 3 percentage point difference in GDP growth is pretty bad, especially in sample. The error is worse in the test set (overfitting), understandable given the difficulty of out of sample GDP prediction, but this would suggest that regularization might be worthwhile. Note that a good portion of the error comes from outliers, however. Removing them might be one alternative path forward.

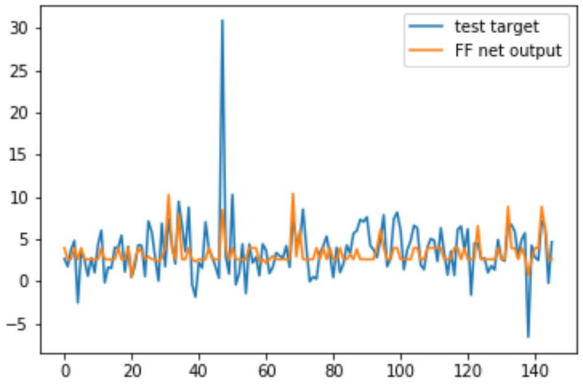
0.5 Feed Forward Multilayer Perceptron (Neural Network):

0.5.1 Implementation:

We implemented this neural network as described in Part 6, using the NeuroLab python package (`https://pythonhosted.org/neurolab/`). We used a number of nodes in the hidden layer equal to 2/3 the number of features (48 hidden nodes). We used the Broyden-Fletcher-Goldfarb-Shanno algorithm for optimization, as we found it more efficient than simple gradient descent, although we also experimented with momentum-based gradient descent and adaptive learning rates. Weights and biases were initialized from a uniform distribution between -0.1 and 0.1

0.5.2 Outcomes:





0.5.3 Analysis and Results:

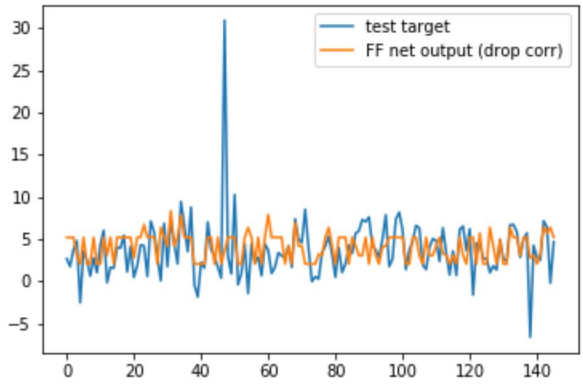
MSE on the training dataset: 15.268
MSE on the test dataset: 8.841 Here the MSE in the training dataset was actually worse than that of the linear regression, this was largely because of issues with local maxima. As can be seen from two other trials in the appendix, the model results varied significantly depending on initialization. This trial outperformed linear regression in the test set, but this was mostly just chance. With access to more computing power, we might run many trials and select the model with the best fit to the training set, to avoid local maxima.

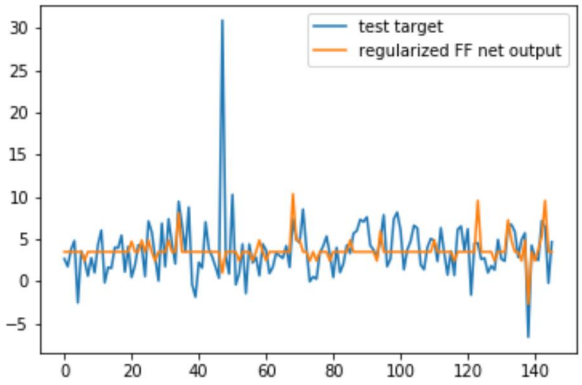
0.6 Approaches to feature selection:

0.6.1 Implementation:

We can attempt to resolve issues of over-fitting, local maxima, or correlation between features by removing some or reducing their influence. We approach this two ways: one by dropping one feature of any feature pair with a correlation above 0.95, and the other by regularizing using NeuroLab's built in regularization ratio functionality. We use a ratio equal to 0.1. Credit to Chris Albon for the code to drop correlated features.

0.6.2 Outcomes:



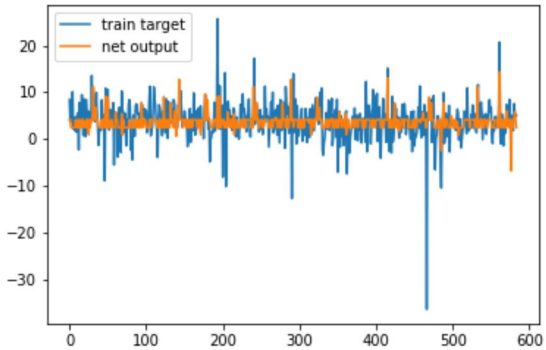


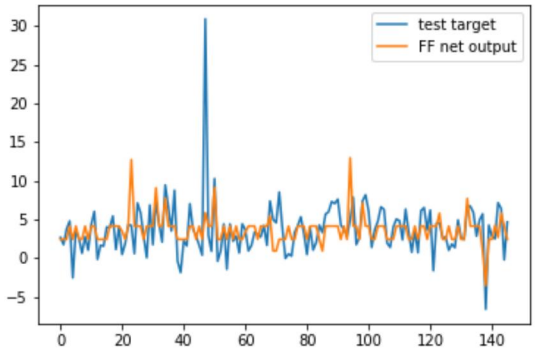
0.6.3 Analysis and Results:

(Both graphs above are on the test set). Dropping closely correlated features reduced our number of features from 72 to 50.
MSE on the training set: 13.737; MSE on the test set: 11.796
This is not out of line with the variation (from initialization differences) that we saw in the models that used the full feature set. However, it was much more computationally efficient. This makes sense: we would expect eliminating correlated variables to have little impact on prediction power but reduce the processing load.
Regularization: MSE on the training set: 15.314; MSE on the test set: 11.778
This did not affect accuracy or computation time. But that makes sense–if anything our model is underfit, and could have benefited from another layer given more computational resources. On the graph, regularization caused the predictions to often stick very close to the mean. Those predictions that it did make away from the mean did seem more confident, however. To fully assess either of these techniques, one would have to run many trials, given local maxima problems.

0.7 Linear Regression as an Input to a FFNN

In an attempt to avoid some inferior local maxima, we use the predicted GDP growths from the regression in part 6.1 as an input to our FFNN. Correspondingly, we drop one other feature to avoid collinearity. The results were fairly unremarkable, similar to what we obtained with other methods:
MSE on the training set: 14.217
MSE on the test set: 9.866





0.8 Elman Recurrent Neural Network:

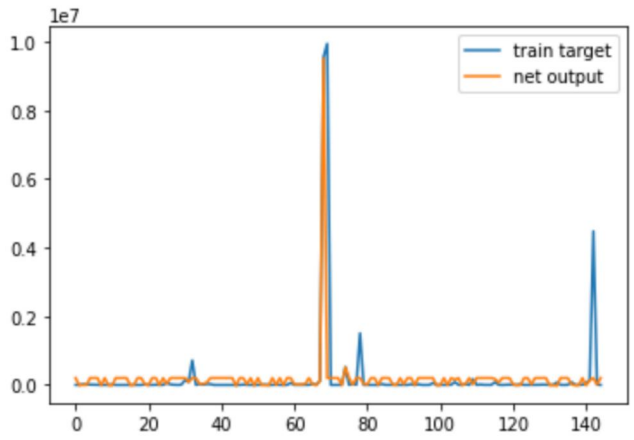
0.8.1 Implementation:

This implementation was done earlier, and met some hurdles, so the specifications are not the same as those for the other models. Here we train our model on 145 countries. For each country, we observe 22 features (GDP, current account balance, etc.) in each year. The goal of our model is to predict total GDP (**not GDP growth**) for each country in year t based on data obtained in year $t-1$ or before. We train our model on input data from 2006-2012, predicting GDP in 2013, and test it on input data from 2007-2013, predicting GDP in 2014. We treat the 22 features over 7 years as $22 \cdot 7 = 154$ features.

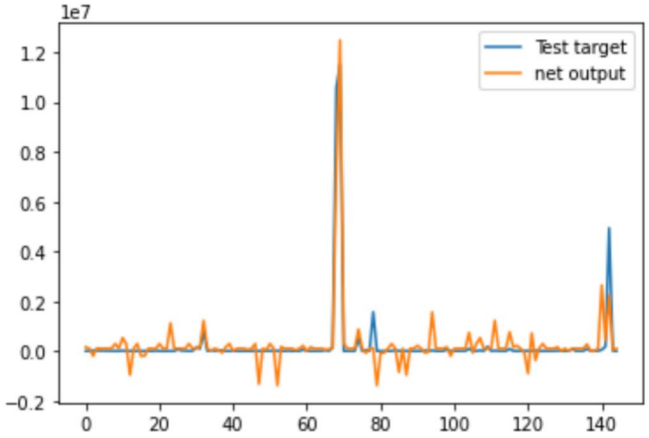
Implementation of RNN model using **NeuroLab**: https://pythonhosted.org/neurolab/ We implemented the Elman Recurrent network (newelm).

Access our code and google colab through this link:
https://colab.research.google.com/drive/1CoVBX0YKq76JWn9ir0rZq7U6ZN4LBiKV?usp=sharing

0.8.2 Outcomes:



Training Output Target vs Model Prediction



Testing Output Target vs Model Prediction

0.8.3 Analysis and Results:

Our accuracy stops improving around Epoch 3000. Training from 2013 data, our MSE for the training data is 8.456612e+11, our MSE for the testing data (2014 GDP) based on the 2013 model is 3.019396e+11, while the error for a model trained until 2014 data is 3.180453e+11. The difference between the errors can likely be explained by testing data predicting the increase around x = 140. The number of nodes or hidden layers used was $2 *$ the number of features/3. Different from our other models, we predicted GDP directly instead of GDP growth. The issue with this approach is that the model simply fits to the US's GDP, because it dwarfs that of the other countries in the sample.

Part 8: Conclusion

We predicted GDP change per year using a variety of models: (1) OLS regression, (2) Feed Forward Multi-layer Perceptron Neural Network, (3) OLS as Input to a Feed Forward Neural Network and (4) Elman Recurrent Neural Network, as well as regularization- and correlation-based refinements of some of these models Interestingly, the various FFNNs tended to outperform the OLS model because they underfit the training data. Typically one would expect the opposite, given that FFNNs are more complex. It is difficult to make comparison between the different FFNN approaches because of variance in MSE from trial to trial resulting from local maxima. Ideally, we would run many trials and compare the best or average cases, bu doing so for us was computationally infeasible.

Our results show that IMF Economic Outlook data can be used to build models that predict GDP change per year, but that predicting GDP change per year is difficult. Our predictions might be less accurate than those made by economists through more typical analytic means, but we do have the advantage of avoiding bias, a problem that often plagues professional projections (`https://voxeu.org/article/accuracy-long-term-growth-forecasts-economics-researchers`).

Going forward, we can continue the project by adopting more varied features and dataset, for example by including data from the World Bank's Ease of Doing Business Dataset. Including country fixed effects would probably improve accuracy, and we wrote code to do so, but the memory requirements to continue using the same algorithm were too great (from a computational perspective, including country fixed effects would essentially mean adding 146 further features). Because our model tended to underfit to the training data, it would be advisable to increase the number of features considered and find other ways to increase model complexity.

Part 9: Resources and Index

0.9 Database links

1. https://www.doingbusiness.org/en/data
2. https://www.mcc.gov/who-we-select/indicators

0.10 Relevant research links

1. https://link.springer.com/article/10.1007/s10614-020-10054-w
2. https://towardsdatascience.com/predicting-real-gdp-growth-85f34fdca97a
3. https://elibrary.worldbank.org/doi/abs/10.1596/1813-9450-5787
4. https://arxiv.org/abs/1905.07357
5. https://core.ac.uk/download/pdf/300382399.pdf

Part 10: Appendix

0.11 Country List

`https://docs.google.com/spreadsheets/d/1Xi2bFPUZI48hR24DL2WhnOi6WKvtF1nb6N3J6unmKkA/`

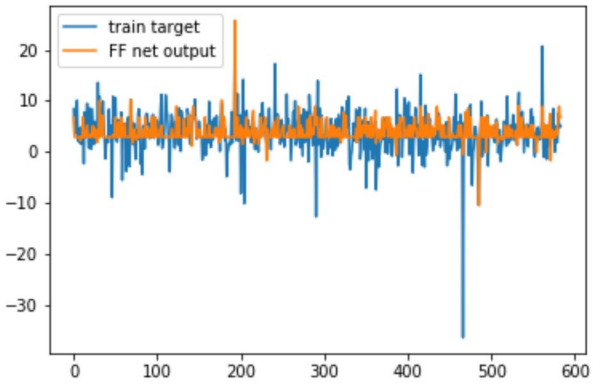
0.12 WEO Feature Codes

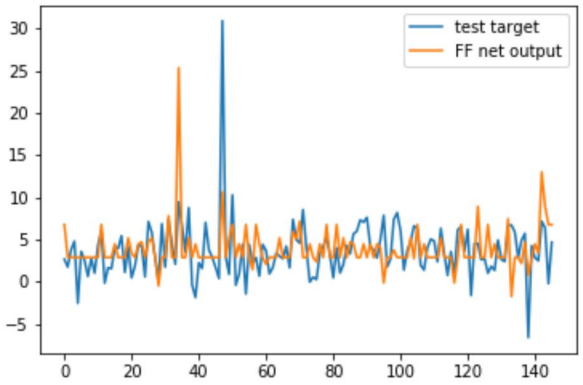
`https://docs.google.com/spreadsheets/d/1AxaAw4iD8IUXtEkvFEqIlQWWQVms14yEi42jA8NUks0/`

0.13 Further Trials of FFNN

0.13.1 Trial 1:

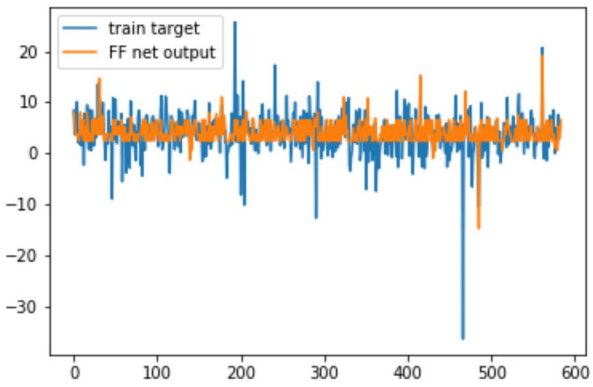
MSE on the training set: 12.944
MSE on the test set: 12.070

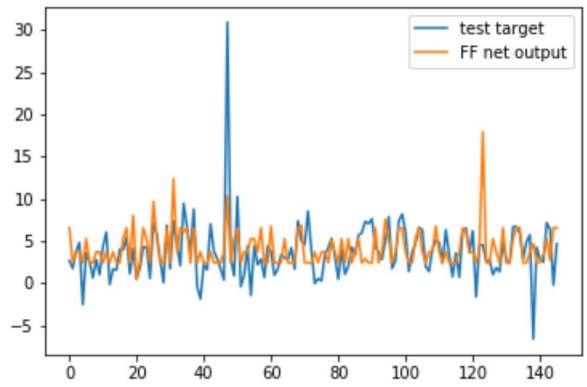




0.13.2 Trial 2:

MSE on the training set: 13.043
MSE on the test set: 10.298


train target
FF net output
20
10
0
−10
−20
−30
0
100
200
300
400
500
600


test target
FF net output
30
25
20
15
10
5
0
−5
0
20
40
60
80
100
120
140