

**Программа повышения конкурентоспособности**

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«Уральский федеральный университет имени**

**первого Президента России Б. Н. Ельцина»**

**ПРОГНОЗИРОВАНИЕ ВРЕМЕННЫХ РЯДОВ  
НА ОСНОВЕ НЕЙРОННЫХ СЕТЕЙ В МАТЛАВ**

**Методические указания к выполнению  
лабораторной работы № 2**

Екатеринбург

2019

## **Содержание**

Цель изучения материала .....	3
Список литературы .....	3
1. Введение.....	4
2. Задание на лабораторную работу .....	4

## Цель изучения материала

Целью данной лабораторной работы является изучение студентами методов прогнозирования временных рядов на основе нейронных сетей. В ходе выполнения работы студентами приобретаются навыки и умения по применению методик NAR и NARX. Для выполнения работы требуется пакет инструментов MATLAB *Neural Network Toolbox*.

## Список литературы

1. MathWorks Center. NARX Network. Design Time Series NARX Feedback Neural Networks. (<https://www.mathworks.com/help/nnet/ug/design-time-series-narx-feedback-neural-networks.html>)
2. Karter J. Time series analysis with MATLAB. ARIMA/VARMAX/GARCH/GJR Models. Functions and Examples. — 2016. — 422 с. ISBN 978-1-539-54638-2.
3. Садовникова Н.А., Шмойлова Р.А. Анализ временных рядов и прогнозирование. — М.: «Футурис», 2009. — 261 с.
4. Голяндина Н.Э. Метод «Гусеница»-SSA: прогноз временных рядов: Учеб. пособие. — СПб. — 2004. — 52 с.
5. Бокс Дж., Дженкинс Г. Анализ временных рядов: прогноз и управление, под ред. В.Ф. Писаренко. — М.: Мир, 1974. — 406 с.

## 1. Введение

В настоящее время нейронные сети проникают во все большее число самых различных областей науки и техники, поэтому было только дело времени, прежде чем были разработаны методы анализа и прогнозирования временных рядов на их основе. Особенно эффективно применение нейронных сетей в области прогнозирования ВР, так как, в отличие от других методов, нам не нужно строить модель для заданного ВР – за нас это делает как раз обученная нейронная сеть.

## 2. Задание на лабораторную работу

- 1) Загрузите из mat-файла **Fort.mat** ряд, содержащий отсчеты некоторого реального ВР, всего 174 отсчета в вектор-строке. Вид ВР представлен на рисунке 1. На глаз, в ряде видна явная сезонность, присутствует некоторый убывающий тренд. Был выбран короткий ряд, чтобы повысить скорость расчетов.

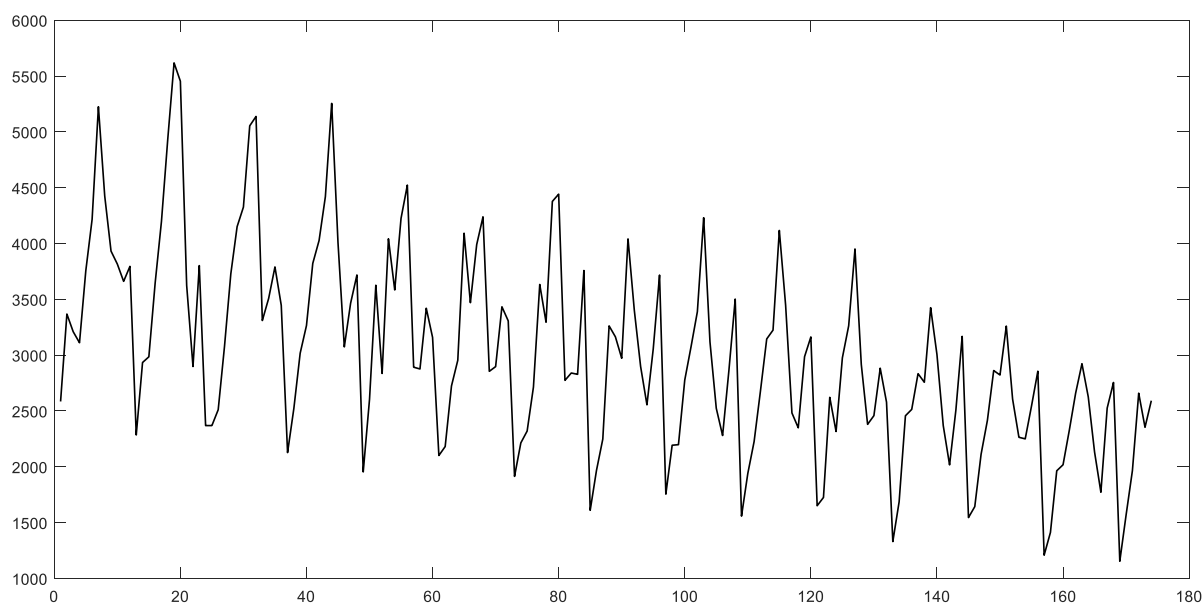
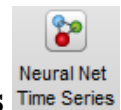


Рисунок 1 – Временной ряд *Fort*

- 2) Длина прогноза студентами выбирается самостоятельно, она должна быть **не меньше 24** отсчетов.
- 3) Начальная точка прогноза определяется студентом самостоятельно. У разных студентов они не должны совпадать.
- 4) Мы будем производить **ретроспективный прогноз**, подобно прогнозу из прошлой лабораторной работы. Графики исходного ряда и прогноза строятся вместе, так как они имеют малую длину и вполне могут поместиться рядом с достаточной точностью.
- 5) В среде MATLAB прогнозом временных рядов занимается средство из

*Neural Network Toolbox* называемое **Neural Net Time Series**.



- 6) Вызовите этот инструмент либо с панели приложений MATLAB, либо с помощью команды *ntstool*.
- 7) Откроется большое окно, как на рисунке 2.

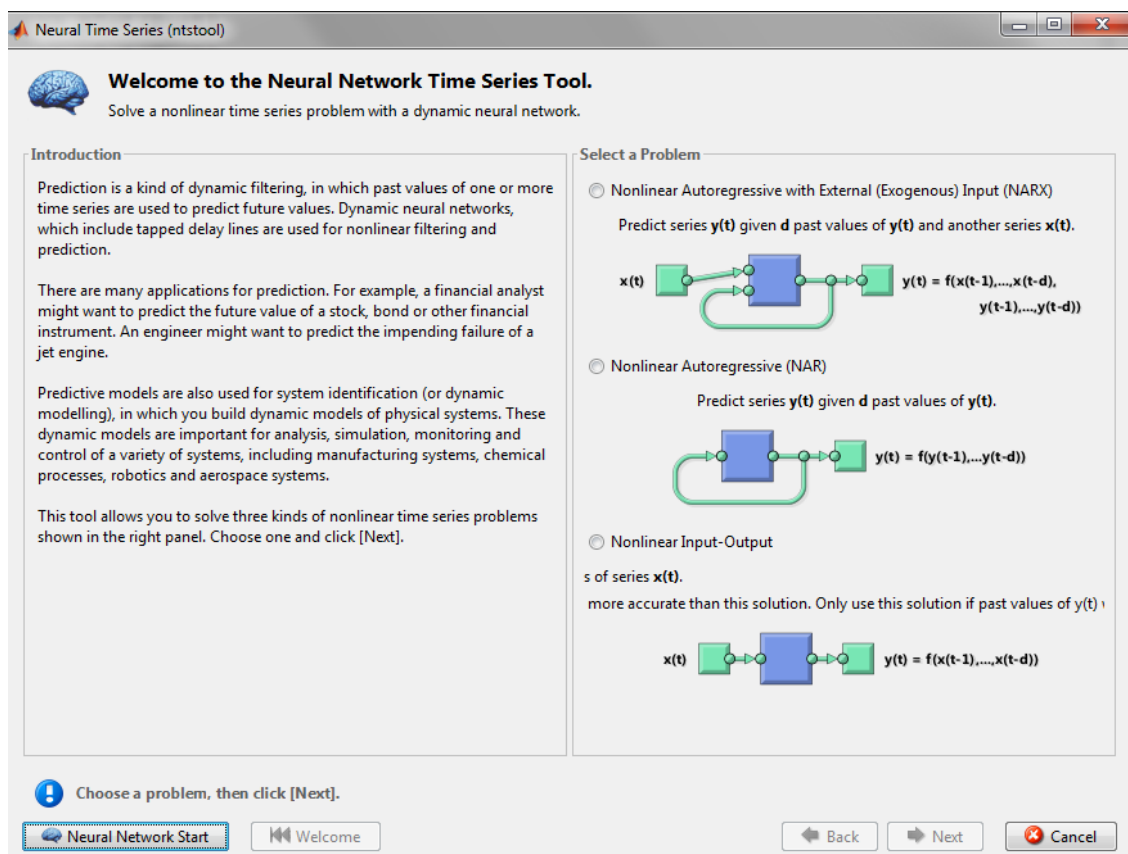


Рисунок 2 – Главное окно средства *Neural Net Time Series*

- 8) В данной лабораторной работе нас будут интересовать два типа моделей – это NAR и NARX. Начнем с более простой задачи – построения прогноза на основе самого ВР с помощью NAR.
- 9) Выберите пункт NAR в данном окне. Нажмите «Далее» (*Next*) внизу.
- 10) Откроется следующее окно выбора исходного ВР, для которого строится адаптивная модель прогноза (рис. 3). Если Вы уже загрузили данные Fort в *Workspace*, то их можно выбрать в выпадающем списке. Либо данные можно прочесть из внешнего файла. Наш ряд является вектором-строкой, поэтому выбираем соответствующий пункт *Matrix column*. Ошибиться здесь трудно – пока не будет выбран правильный вид ряда, следующий шаг не будет доступен.

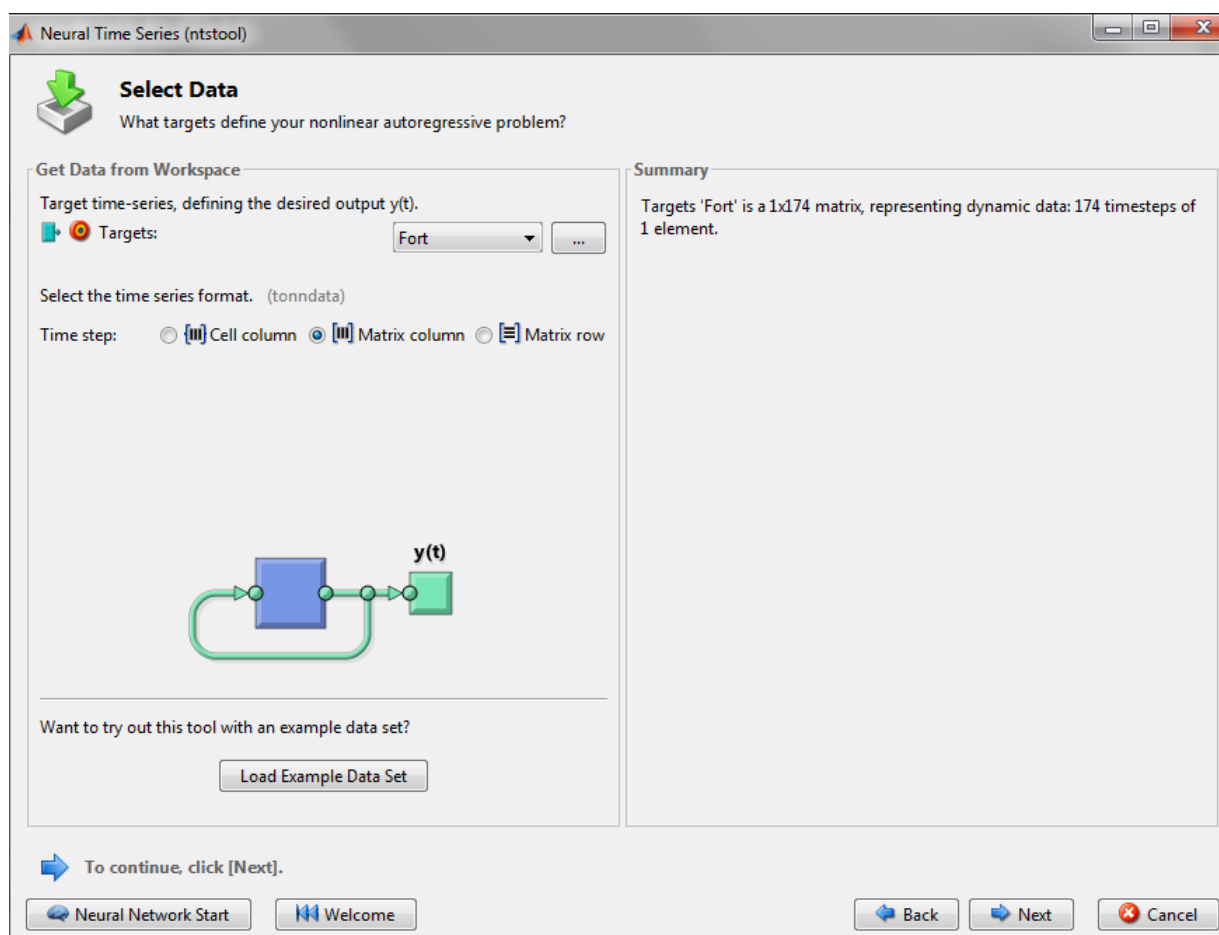


Рисунок 3 – Окно выбора исходного ВР

## Программа повышения конкурентоспособности

- 11) Переходите к следующему окну – к определению размеров обучающей выборки (*Training*), выборки валидации (*Validation*) и выборки тестирования (*Testing*) (рисунок 4).
- 12) На обучающей выборке сеть подстраивает модель под ВР. По выборке валидации при этом она отслеживает процесс обучения (принцип обучения с учителем) по нескольким параметрам. По выборке тестирования определяются оценки прогнозной модели NAR.
- 13) По умолчанию для нас уже выставили обычные размеры выборки в процентах от размера исходного ряда: **70% / 15% / 15%**.  
К сожалению, наш ряд очень короткий, поэтому разумным будет сделать обучающую выборку больше, выборку валидации оставить прежней, а тестирование свести к минимуму – мы сами будем проверять точность модели: **80% / 15% / 5%**.
- 14) Эти значения являются чистой начальной рекомендацией и могут быть изменены студентом по своему желанию. Результирующие проценты выборок должны быть указаны в отчете.

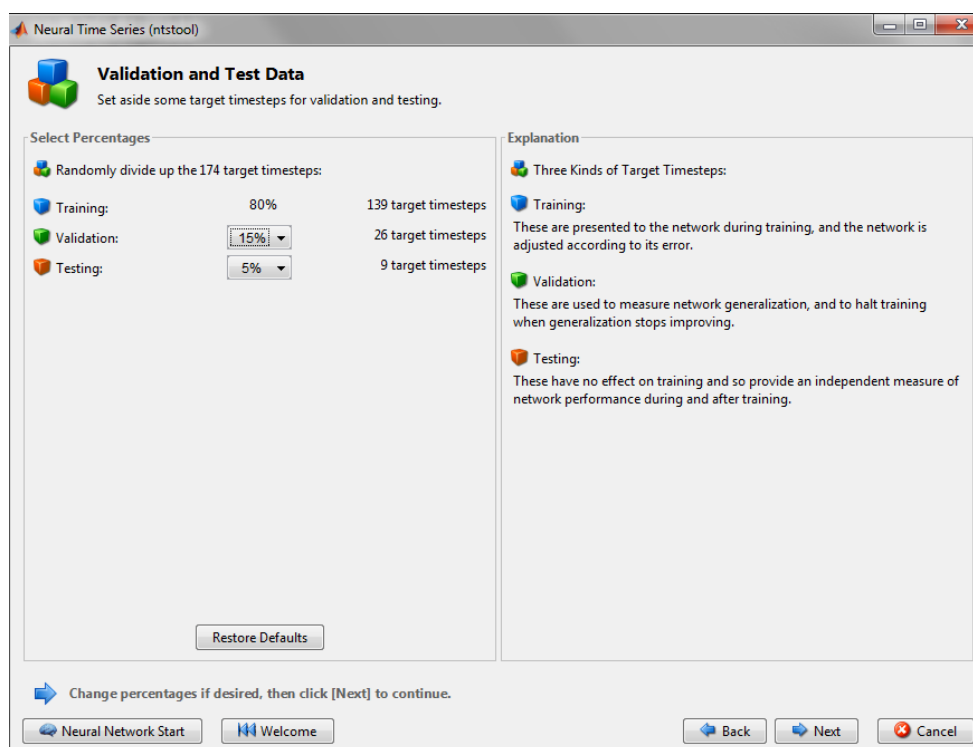


Рисунок 4 – Определение размера обучающей выборки сети

- 15) После определения размера выборок идет самый важный этап определения параметров нейронной сети – **число нейронов** скрытого слоя, и **порядок** авторегрессионной NAR модели. Именно эти параметры варьируются студентами для достижения высокой точности собственного прогноза (рисунок 5).
- 16) Число скрытых нейронов определяет **сложность модели** и **количество функциональных зависимостей**, которое может описать NAR модель. С ростом числа нейронов также растет вычислительная сложность обучения сети – чем их больше, тем дольше обучается сеть. Также, надо понимать, **большое число скрытых нейронов не равняется высокой точности прогноза**. В самом деле, допустим, исходные процессы ВР описываются приблизительно в 5 взаимозависимостей/параметров, но для нейронов выделено 50, то есть каждая характеристика словно «размазывается» по десятку нейронов с потерей точности реконструкции поведения ряда.
- 17) **Порядок модели  $d$  = число задержек** – есть величина, по скольким предыдущим точкам исходного ряда строится одноточечный прогноз. То есть, если поставить порядок равный трем, то модель предполагает, что одну точку ряда можно предсказать по трем предыдущим отсчетам ряда. Все точь-в-точь, как в АР моделях.
- 18) Итак, для примера поставим 24 нейрона скрытого слоя, и будем прогнозировать одну точку вперед по 24 предыдущим точкам.



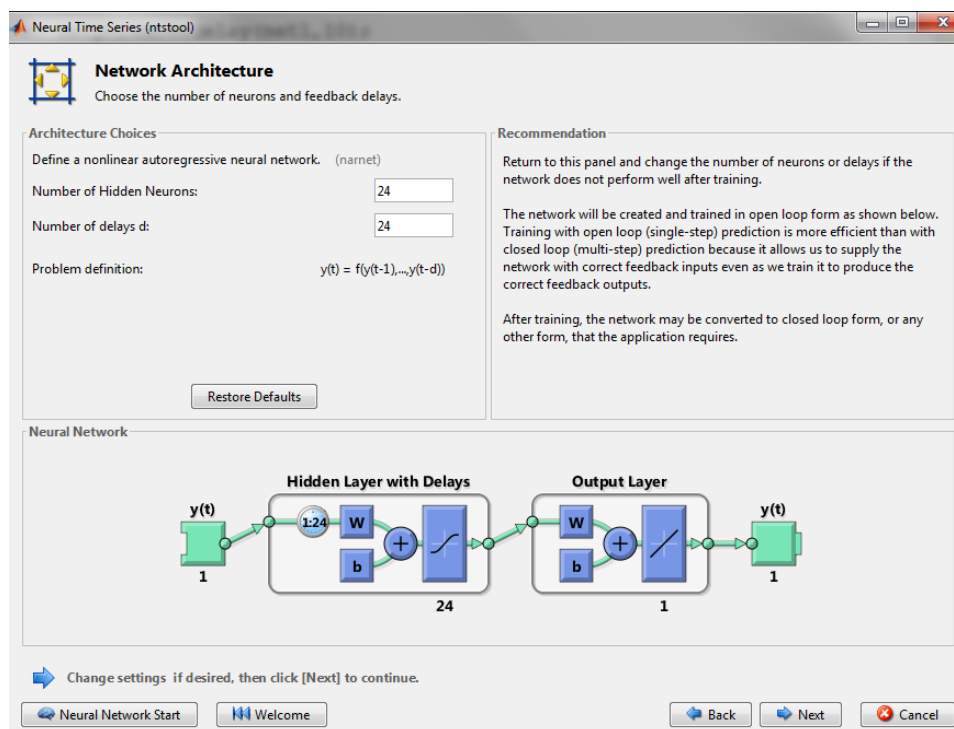


Рисунок 5 – Выбор параметров нейронной сети

- 19) Далее выбирается метод обучения модели – их здесь доступно три вида (рисунок 6). Метод **Levenberg-Marquardt** является самым быстрым, точность варьируется. Метод **Bayesian Regularization** является самым медленным и, зачастую, избыточным, но он в целом имеет высокую точность. Метод **Scaled Conjugate Gradient** является быстрым, эффективным по памяти, но имеет низкую точность. Выбор метода обучения тоже остается за студентами.
- 20) После нажатия на кнопку **Train** пойдет процесс обучения. Откроется окно, как на рисунке 7, где в режиме реального времени будет отображаться процесс обучения нейронной сети. Здесь можно построить огромное количество графиков, меняющихся прямо по ходу обучения, и отражающих разные параметры модели.
- 21) Разберемся кратко в этих графиках и зависимостях.

## Программа повышения конкурентоспособности

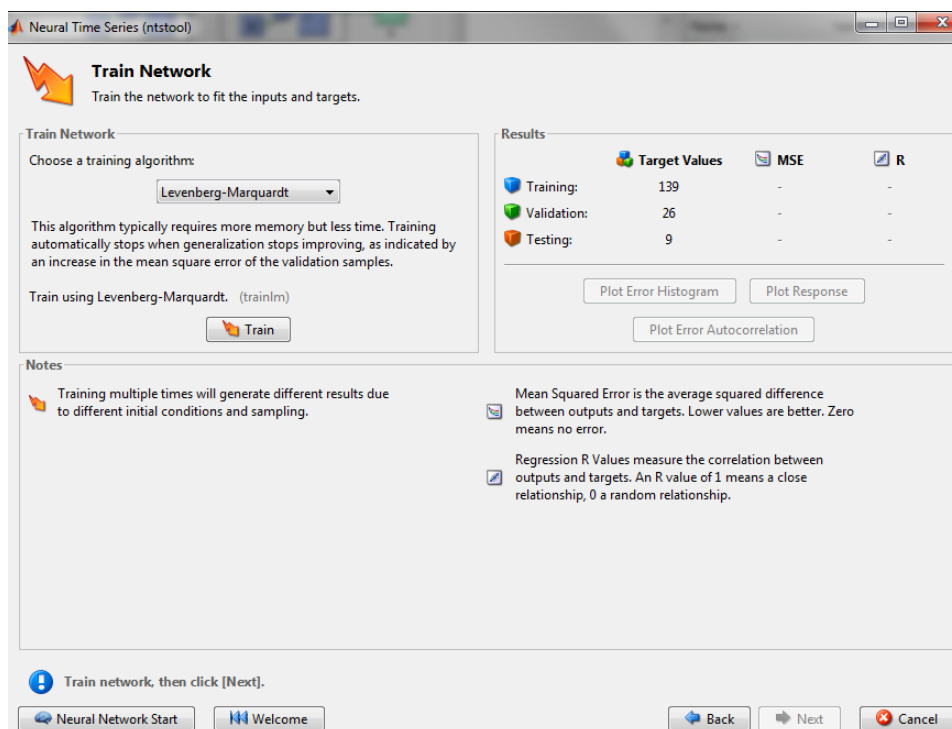


Рисунок 6 – Выбор метода обучения сети

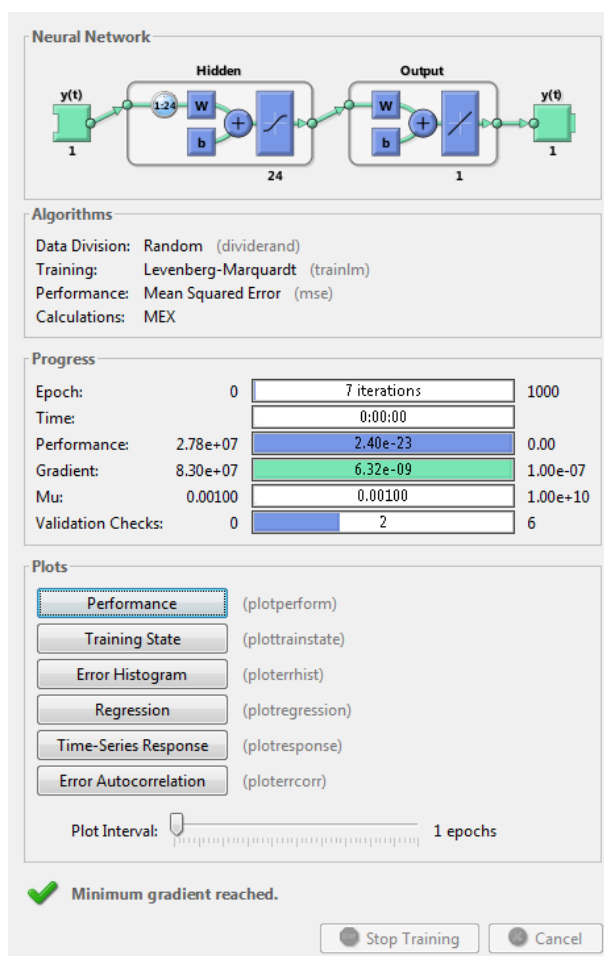
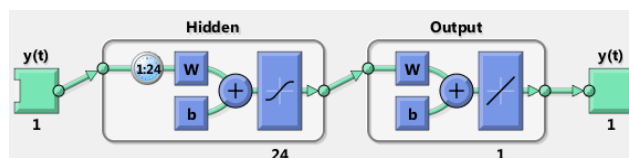


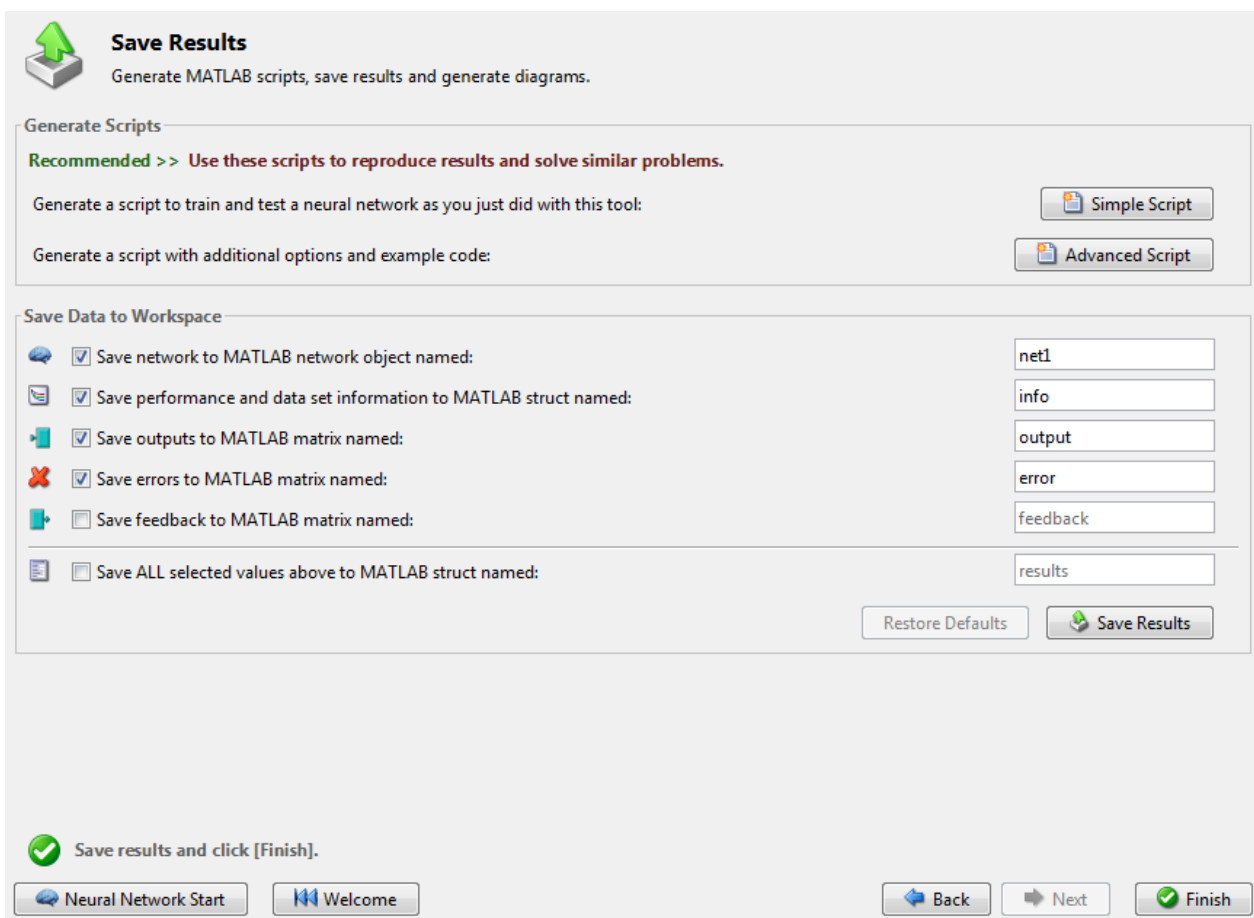
Рисунок 7 – Процесс обучения нейронной сети

- 22) Кнопка **Performance** открывает график, показывающий **скорость сходимости** минимизации среднеквадратичной ошибки **MSE** от итерации обучения сети. Как только ошибка почти не меняется, нейронную сеть вряд можно считать обученной (если ошибка и так близка к нулю – зачем что-то улучшать дальше?).
- 23) Кнопка **Training State** показывает параметры обучения нейронной сети от итерации (**эпохи/вехи**) обучения. Как только какая-либо из кривых перестает быть монотонной убывающей функцией, это обычно служит сигналом, что модель начинает «разбегаться», то есть вместо улучшения точности с ростом эпохи можно получить снижение точности NAR модели.
- 24) Кнопка **Error Histogram** отображает один из самых понятных графиков – оценку плотности распределения остаточного ряда, то есть ряда ошибок между моделью и ВР. **Лучше всего**, когда все ошибки возле нуля, а по краям распределения их почти нет. Еще замечательно, если ошибки распределены по нормальному закону – значит, нейронная сеть смогла декомпозировать ВР на детерминированную модель и случайный шум.
- 25) Кнопка **Regression** рисует регрессионные зависимости, которые служат оценкой корреляционного момента **R** между моделью и ВР. Здесь главное понимать, что чем ближе **R** к единице, тем лучше. Чем ближе нарисованные линии к диагональным прямым, тем лучше модель.
- 26) За кнопкой **Time-Series Response** скрывается **самый понятный график** – на нем изображены отсчеты исходного ряда и отсчеты, найденные по модели NAR. Внизу приведен остаточный ряд. Все очень просто – чем ближе друг к другу графики, тем лучше модель. Чем ближе к нулю остаточный ряд, тем лучше модель. Примерно

подобные графики требовались от студентов при построении прогноза на предыдущей лабораторной работе. Особенно прекрасно наблюдать за этим графиком по ходу обучения модели – на нем наглядно видно, как сеть постепенно приближается к желаемым отсчетам временного ряда.

- 27) Кнопка **Error Autocorrelation** строит функцию автокорреляции ряда остатков. Здесь все тоже очень просто трактуется – пик в нуле будет всегда, а вот на ненулевых сдвигах/лагах лучше не иметь больших выбросов. Напомним, что наличие всех значений функции автокорреляции меньше доверительного интервала (*Confidence Limit*) говорит о некоррелированности ряда остатков, то есть ряд остатков является случайной величиной (что для нас и нужно). Если же это не так, и у нас много выбросов выше уровней доверительного интервала, то в ряде остатков осталось нечто детерминированное, а, значит, модель может быть неточной. Если на протяжении всех сдвигов значения функции выходят за предел, то такая модель является плохой, так как ряд остатков имеет автокорреляционную зависимость, то есть сам по себе может быть описан некоторой NAR моделью.
- 28) Итак, сеть обучена, получена некоторая прогностическая модель. Если при рассмотрении графиков выше Вас что-то не устраивает, сеть можно перетренировать, изменив ее параметры и т.д. Все шаги можно откатить назад и изменить.
- 29) Окно **Evaluate Network** можно пропустить. В окне **Deploy Solution** Вам пригодится кнопка **Neural Network Diagram** – это рисунок, по которому видно параметры NAR нейронной сети. Нажмите на нее и полученный рисунок вставьте в отчет.





**Save Results**  
Generate MATLAB scripts, save results and generate diagrams.

**Generate Scripts**

**Recommended >>** Use these scripts to reproduce results and solve similar problems.

Generate a script to train and test a neural network as you just did with this tool: Simple Script

Generate a script with additional options and example code: Advanced Script

**Save Data to Workspace**

☒ Save network to MATLAB network object named:

☒ Save performance and data set information to MATLAB struct named:

☒ Save outputs to MATLAB matrix named:

☒ Save errors to MATLAB matrix named:

☐ Save feedback to MATLAB matrix named:

☐ Save ALL selected values above to MATLAB struct named:

Restore Defaults Save Results

☒ Save results and click [Finish].

Neural Network Start Welcome Back Next Finish

Рисунок 8 – Финальное окно для сохранения результатов

- 30) Следующий шаг последний. Здесь можно сохранить в *Workspace* все найденные оценки, ряд остатков, объект нейронной сети (для тонкой настройки), а также сокращенный и полный скрипты для повторения всего процесса обучения нейронной сети не вручную, а через команды MATLAB. Студенты могут **самостоятельно** разобраться в этих командах, реализовать функции быстрого перебора оптимальных параметров сети и т.д.
- 31) Для прогноза нам нужен либо скрипт, обучающий сеть и, самое главное, преобразующий ее в другие виды сетей – с замкнутой связью и с одиночным шагом; либо сама обученная сеть *net*. Поэтому обязательно сохраните **Advanced Script**, так как на его основе можно оперативно переобучать нейронную сеть с другими параметрами. В нем также есть примеры прогнозирования, в краткой форме.

- 32) Сохраните нейронную сеть с именем *net* в *Workspace*, поставив галочку напротив «Save network to MATLAB» и нажав кнопку **Save Results**. Теперь у нас есть нейронная модель прогноза (почти готовая).
- 33) Создайте м-сценарий, который и будет заниматься прогнозом.
- 34) Сначала создадим **замкнутую** модель прогноза, то есть, чтобы не прогнозировать по одной точке. В сценарии напишите следующее:  
**netc = closeloop(net);**  
**netc.name = [net.name ' - Closed Loop'];**  
**view(netc)**
- 35) Что здесь было сделано: создаем закрытую модель, меняем ей имя, рисуем ее (Вы должны увидеть на изображении обратную связь).
- 36) Пусть мы прогнозируем на **48** точек. Теперь задаем исходные точки прогноза и длину прогноза:  
**T = tonndata( [Fort(end-23:end) NaN(1,48)] , true, false);**  
 Конвертируем координаты:  
**[xc,xic,aic,tc] = preparets(netc,{}, {},T);**  
 И отдаем их закрытой модели:  
**yc = netc(xc,xic,aic);**
- 37) Строка **NaN(1,48)** создает **массив из 48 неизвестных точек** (Not-a-Number), которые нейронная сеть должна нам заполнить. Вместо нее, в принципе, можно использовать и чистые нули **zeros(1,48)**.
- 38) Теперь в этом массиве лежит прогноз. Чтобы его красиво построить, сделаем следующим образом, не забыв преобразовать ячеистый массив в обычный числовой через **cell2mat**:  
**plot(1:174, Fort, 174:174+48, [Fort(end) cell2mat(yc)])**
- 39) Будущий прогноз получился примерно таким, как изображено на рисунке 9. В целом, получилось что-то похожее на правду. Но точность такого прогноза мы пока не можем оценить – у нас нет новых

## Программа повышения конкурентоспособности

наблюдений/отсчетов ВР, чтобы с ними сравнить получившийся результат. Поэтому для лабораторной работы сделаем **ретроспективный прогноз**.

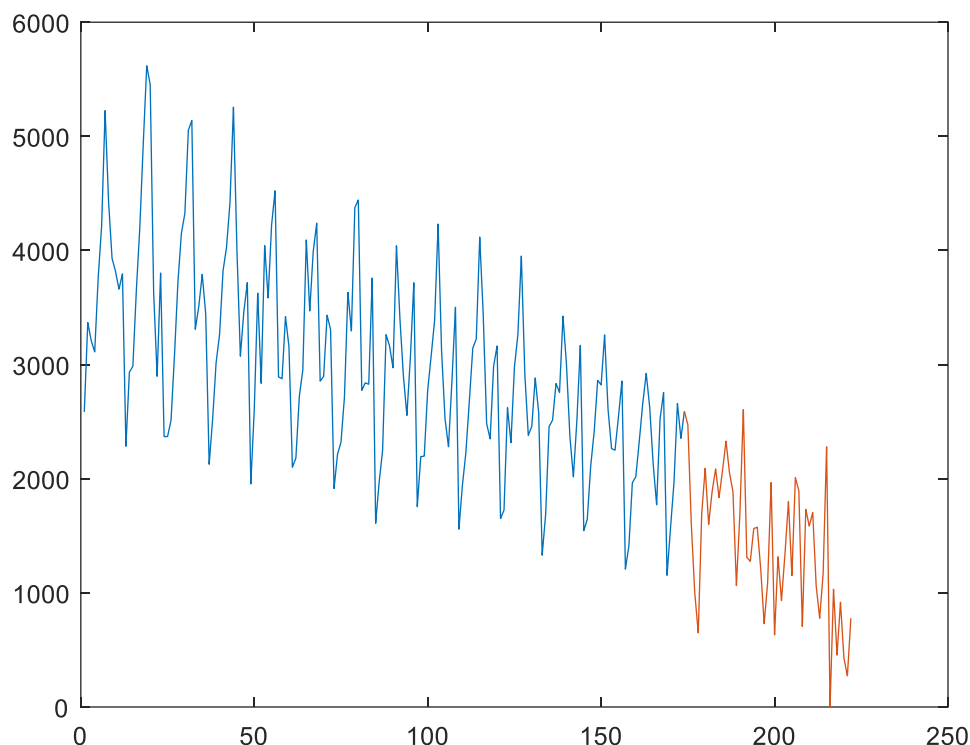


Рисунок 9 – Прогноз с помощью замкнутой NAR-модели

- 40) Ретроспективный прогноз делается по готовой модели аналогичным образом. Пусть мы хотим спрогнозировать **последние 24 точки ряда** (они нам известны). В нашей модели для их прогноза требуется еще 24 **пред-прогнозные** точки (порядок модели = 24). Тогда, получается, нам надо отступить на **48** шагов назад, дать **24** известные точки, и указать, сколько точек прогнозировать:

```
T = tonndata( [Fort(end-23-24:end-24) NaN(1,24)] , true, false);
```

Далее все остается также. Сдвигаем координаты:

```
[xc,xic,aic,tc] = preparets(netc,{},{},T);
```

И получаем прогноз нашей сетью.

```
yc = netc(xc,xic,aic);
```

Изобразим этот прогноз на 24 точки (рисунок 10):

```
plot(1:174, Fort, 174-24+1:174, [cell2mat(yc)])
```

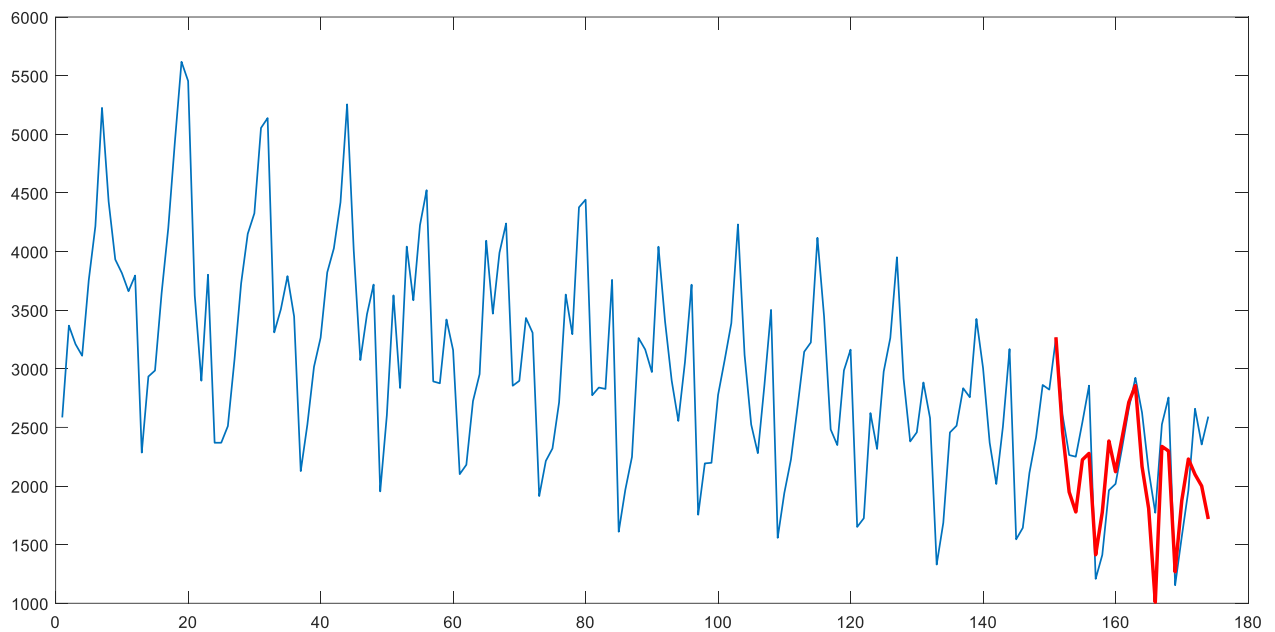


Рисунок 10 – Ретроспективный прогноз на 24 точки

- 41) Так как приведенная выше форма записи **T** довольно неудобна и сложна для понимания, есть **сокращенная форма записи ретроспективного прогноза**, в виде:

**T = tonndata( Fort(end-23-24:end) , true, false);**

где мы просто задаем **начальную точку** работы сети (от нее резервируют 24 (= порядок сети) пред-прогнозные точки) и **конечную точку** прогноза (до конца ряда = **end**).

- 42) Убедитесь самостоятельно, что получившийся прогноз совпадает (рисунок 10), хотя форма записи другая.
- 43) Теперь аналогичным образом выберите собственные параметры сети и обучите ее.
- 44) Постройте два прогноза, как в примерах выше: один прогноз **реальный**, после последней известной точки ряда; второй прогноз – **ретроспективный**, начиная с любой выбранной Вами точки и до любой выбранной точки (**точек прогноза должно быть не менее 24**).
- 45) Приведите в отчете получившиеся прогнозы, использованные модели и их параметры. Оцените точность прогноза численно.



- 46) Аналогичным образом попробуйте построить **усложненную нейронную сеть со входом – NARX-сеть** (см. рис. 2). Эта сеть отличается от NAR тем, что имеет еще вход  $x(t)$  который может служить **базисной моделью ряда**. На прошлых лабораторных работах мы изучили множество методов декомпозиции ВР на компоненты. Любая из этих компонент (или их комбинация), в принципе, может выступать в качестве базового ряда.
- 47) На основе методов декомпозиции и анализа временных рядов из прошлых лабораторных работ (EMD, SSA, WPD и другие) получите базисный ряд  $x(t)$  той же длины, что и ряд *Fort*. Затем по аналогичной методике натренируйте модель NARX-сети.
- 48) Постройте только **ретроспективный прогноз** с использованием NARX-сети. Учтите, что при наличии входа  $x(t)$  изменится выражение в пункте 40 на
- `[xc,xic,aic,tc] = preparets(netc, Xt, {}, T);`**
- так как нам потребуется передавать в качестве координат еще и входные данные базисного ряда.

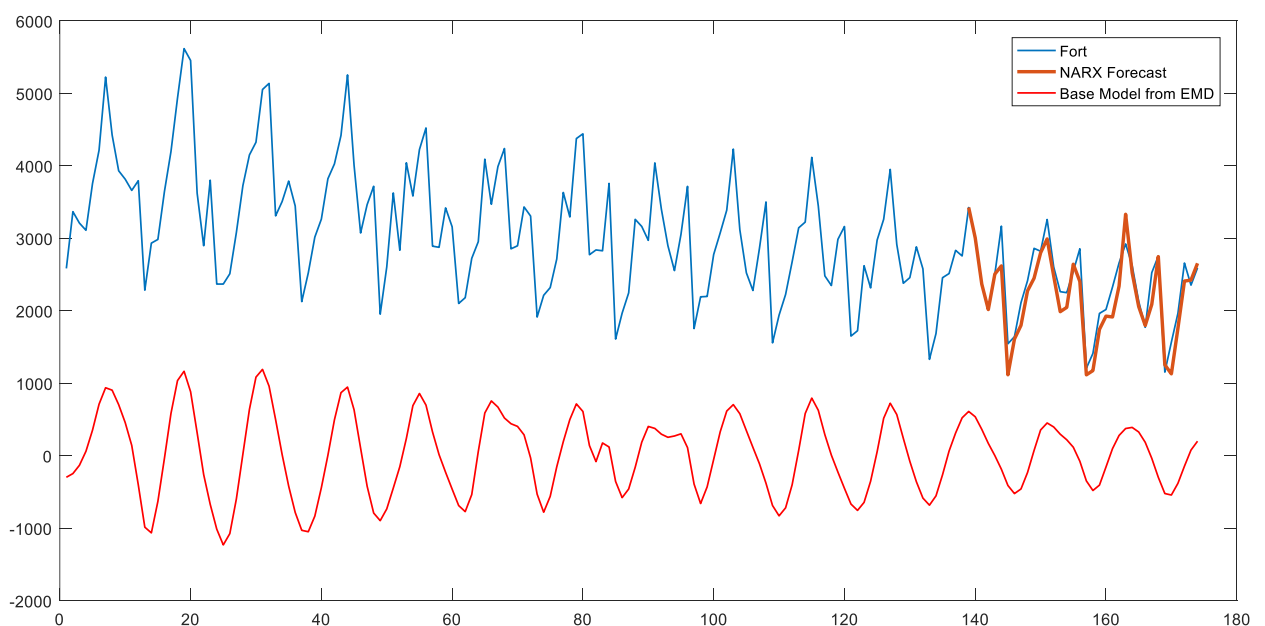


Рисунок 11 – Прогноз на основе базисного ряда и NARX-сети