Дана функция, заданная в виде системы трёх дифференциальных уравнений первого порядка:

# ГЕНЕРИРОВАНИЕ ОБУЧАЮЩЕЙ ВЫБОРКИ

Из-за невозможности решения системы (1.1) необходимо использовать

приближенные численные методы.

Для численного решения данной системы дифференциальных уравнений будем применять алгоритм Рунге-Кутты.

Метод Рунге-Кутты заключается в последовательном применении следующих формул:

 \textbf{y}_{n+1} = \textbf{y}_n + {h \over 6}(\textbf{k}_1 + 2\textbf{k}_2 + 2\textbf{k}_3 + \textbf{k}_4) 

 \textbf{k}_1 = \textbf{f} \left( x_n, \textbf{y}_n \right), 

 \textbf{k}_2 = \textbf{f} \left( x_n + {h \over 2}, \textbf{y}_n + {h \over 2} \textbf{k}_1 \right), 

 \textbf{k}_3 = \textbf{f} \left( x_n + {h \over 2}, \textbf{y}_n + {h \over 2} \textbf{k}_2 \right), 

 \textbf{k}_4 = \textbf{f} \left( x_n + h, \textbf{y}_n + h\ \textbf{k}_3 \right). 

Этот метод имеет четвёртый порядок точности. Это значит, что ошибка на одном шаге имеет порядок O(h^5), а суммарная ошибка на конечном интервале интегрирования имеет порядок O(h^4) .

В результате решения системы (1.1) методом Рунге-Кутта были получены эталонные значения *X(t), Y(t)* и *Z(t)* с шагом дискретизации *h = 0,1*. Для обучения нейронных сетей будем использовать 750 полученных значений, а на оставшихся значениях будет исследоваться обобщающая способность сетей.

По рассчитанным точкам построим следующие графики функции:

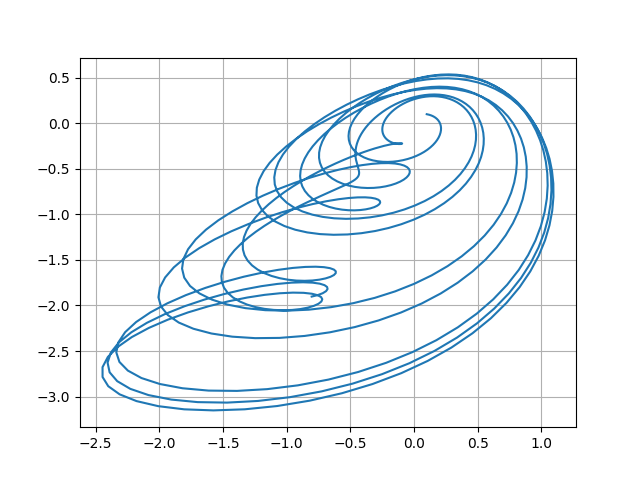


Рис. 2.1. Проекция функции на плоскость XY

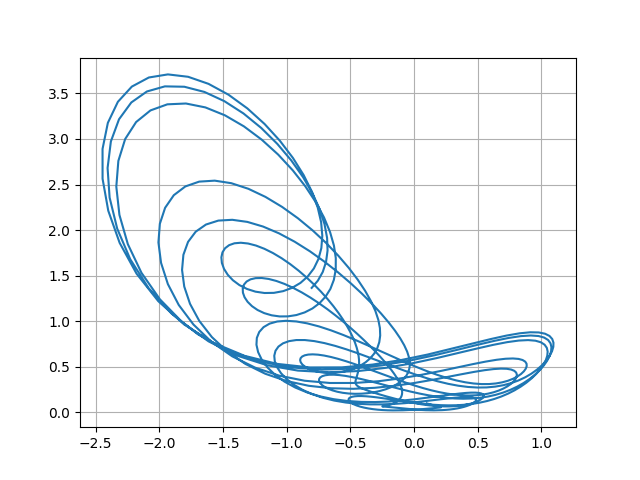


Рис. 2.2. Проекция функции на плоскость XZ

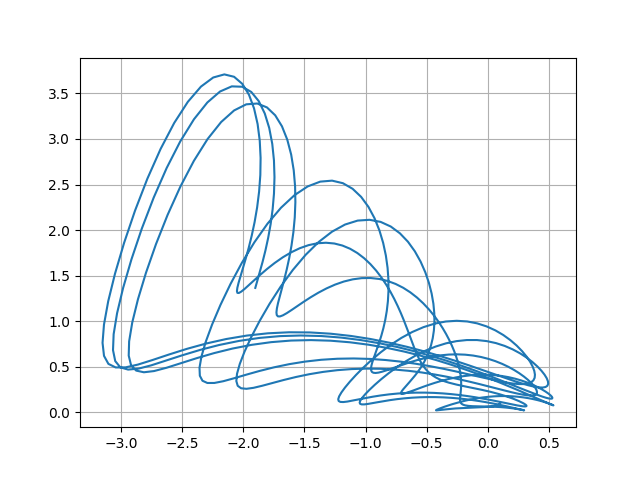


Рис. 2.3. Проекция функции на плоскость YZ

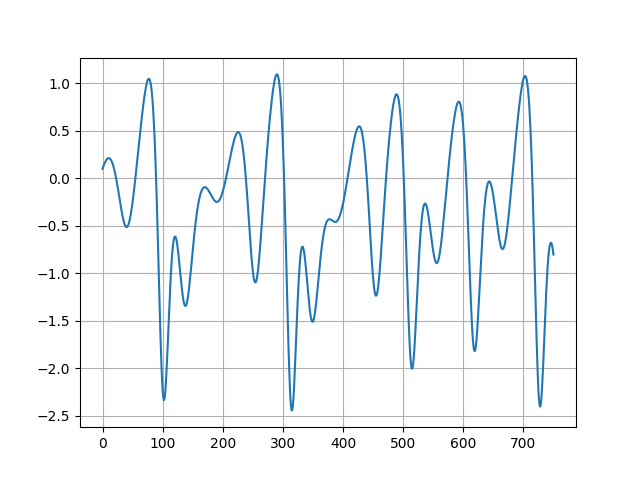


Рис. 2.4. Зависимость X(t)

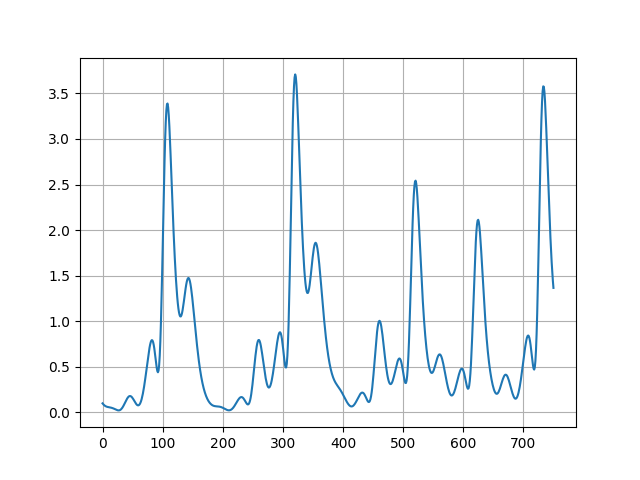


Рис. 2.5. Зависимость Y(t)

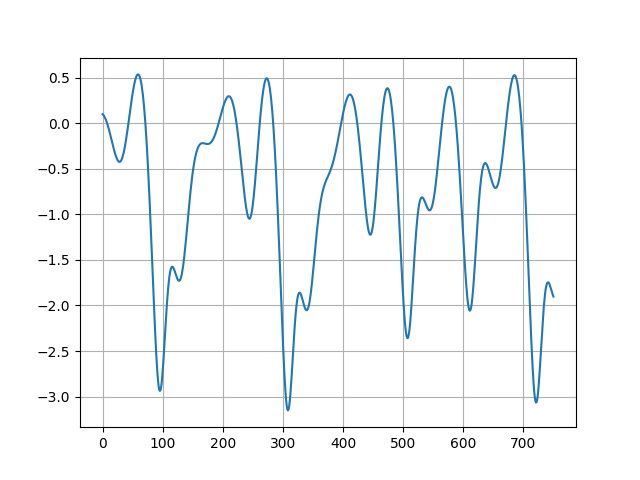


Рис. 2.6. Зависимость Z(t)

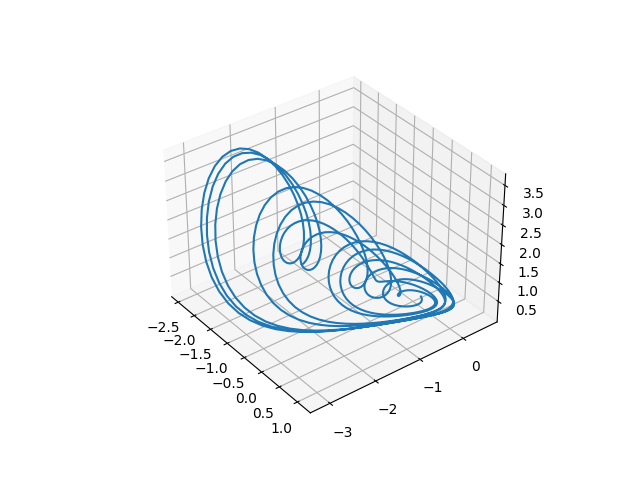


Рис. 2.7. Трёхмерное изображение функции

Таким образом, обучающая выборка сформирована, можно переходить к следующему этапу – проектированию нейронных сетей и алгоритмов обучения.

# ПРОЕКТИРОВАНИЕ НЕЙРОННЫХ СЕТЕЙ

Для решения поставленной задачи будем использовать многослойную нейронную сеть, а именно многослойный персептрон с одним скрытым слоем, линейной функцией активации в выходном слое и нелинейной в скрытом.

В курсовом проекте необходимо рассмотреть следующие нейронные сети для аппроксимации и прогнозирования функции:

1. Многослойный персептрон с тремя нейронными элементами в распределительном и тремя нейронными элементами в выходном слое (рис. 3.1):

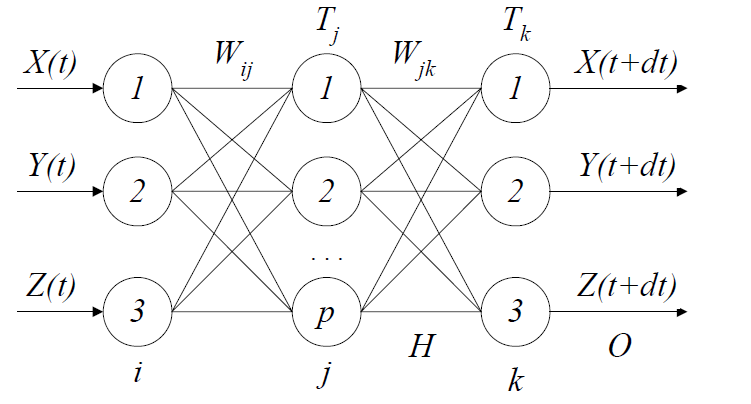


Рис. 3.1. Многослойный персептрон (3/*р*/3)

Число нейронов в скрытом слое *j* определяется экспериментально с целью достижения наилучших результатов аппроксимации и прогнозирования.

На вход сети подаются значения *X(t)*, *Y(t)* и *Z(t)*, а выходными значениями сети будут значения *X*, *Y* и *Z* в следующий момент времени – *X(t+Δt)*, *Y(t+Δt)* и *Z(t+Δt)*.

2. Многослойный персептрон с шестью нейронными элементами в распределительном и тремя нейронными элементами в выходном слое (рис. 3.2):

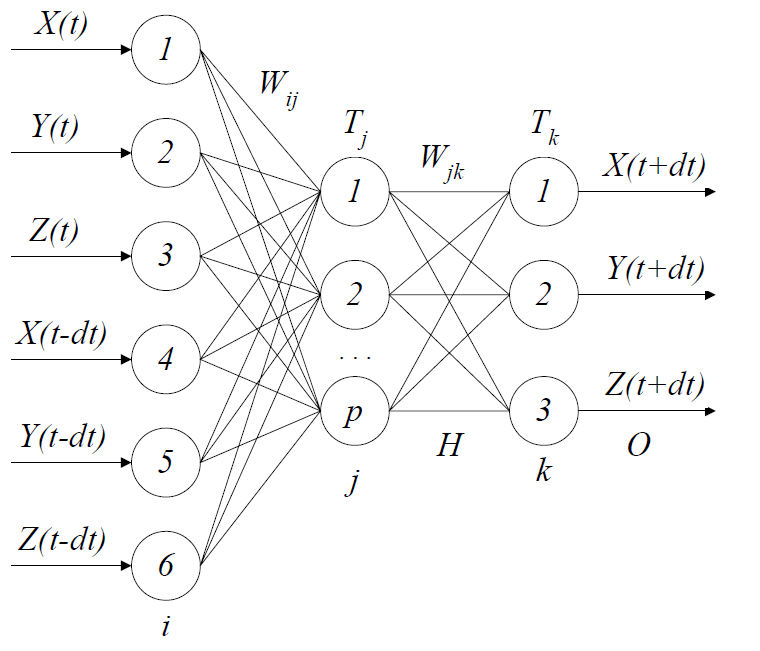


Рис. 3.2. Многослойный персептрон (6/*р*/3)

Здесь количество нейронных элементов в скрытом слое *j* также изначально неизвестно, оптимальное их количество определяется экспериментальным путем.

На вход сети помимо значений *X(t)*, *Y(t)* и *Z(t)* будем подавать значения

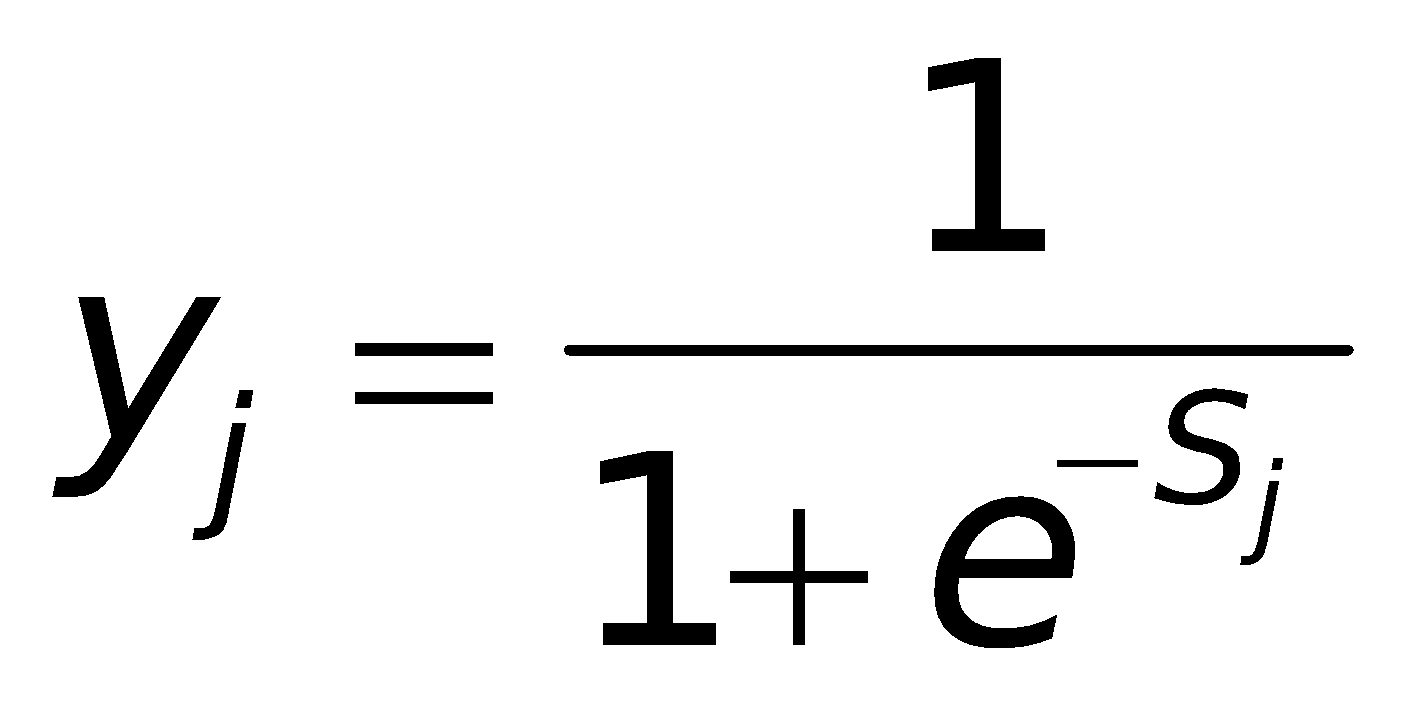
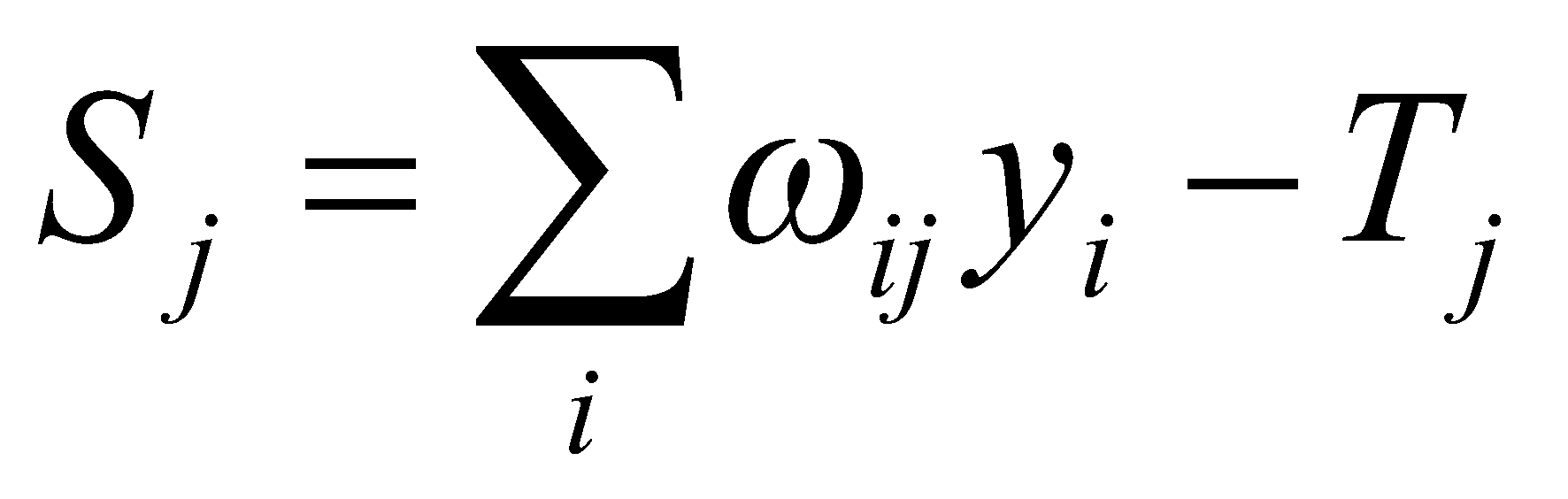
*X(t-Δt)*, *Y(t-Δt)* и *Z(t-Δt)*. На выходе сети будем получать значения *X*, *Y* и *Z* в момент времени *t+Δt*: *X(t+Δt)*, *Y(t+Δt)* и *Z(t+Δt)*.

Число нейронов во входном слое, а также размерность промежуточного слоя подбираются так, чтобы погрешность аппроксимации и прогнозирования была минимальной.

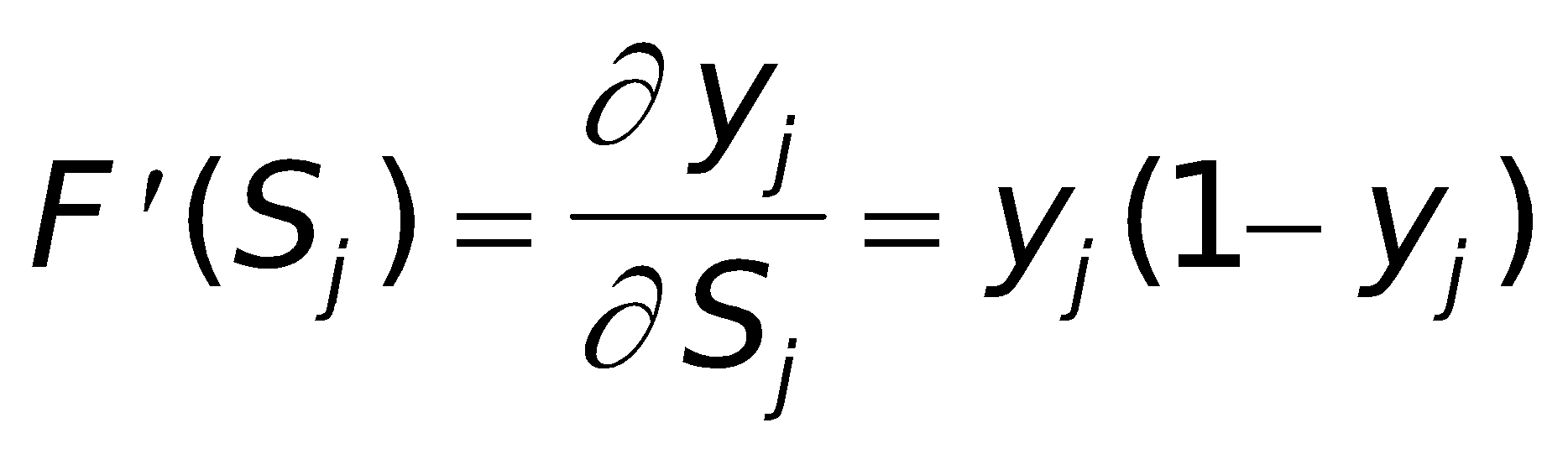
Для обучения таких сетей используется метод «скользящего окна». Суть метода «скользящего окна» состоит в том, что в каждый момент времени на вход сети подаются значения временного ряда *A(t+(1-n)Δt)..A(t)*, при этом полученное на выходе значение сравнивается со значением *A(t+Δt)*, где *n* – размерность «окна».

Для программной реализации использовалась сигмоидная функция активации в скрытом слое.

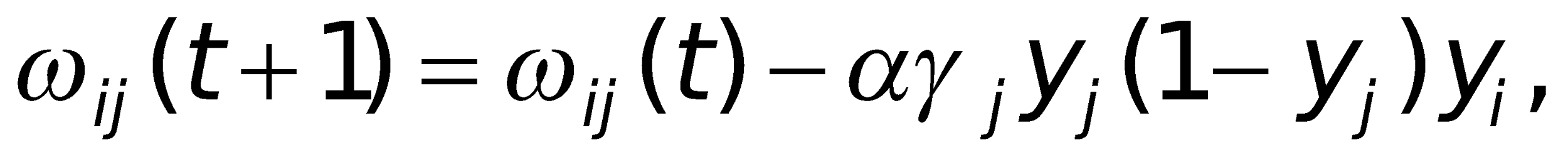
Выходное значение *j*-го нейронного элемента определяется следующим   
образом:

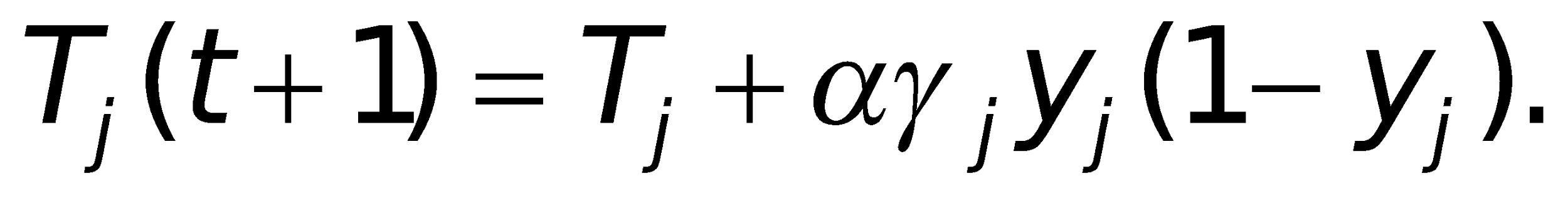
*, *

Тогда

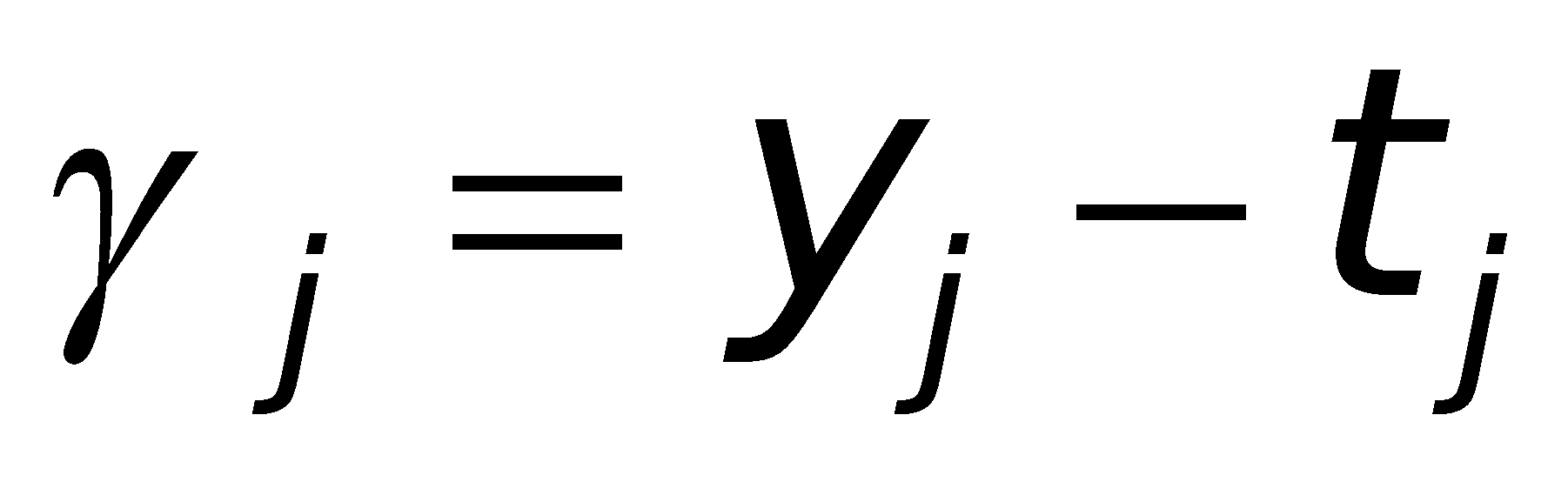
**

В результате обобщенное дельта правило для сигмоидной функции активации можно представить в следующем виде:

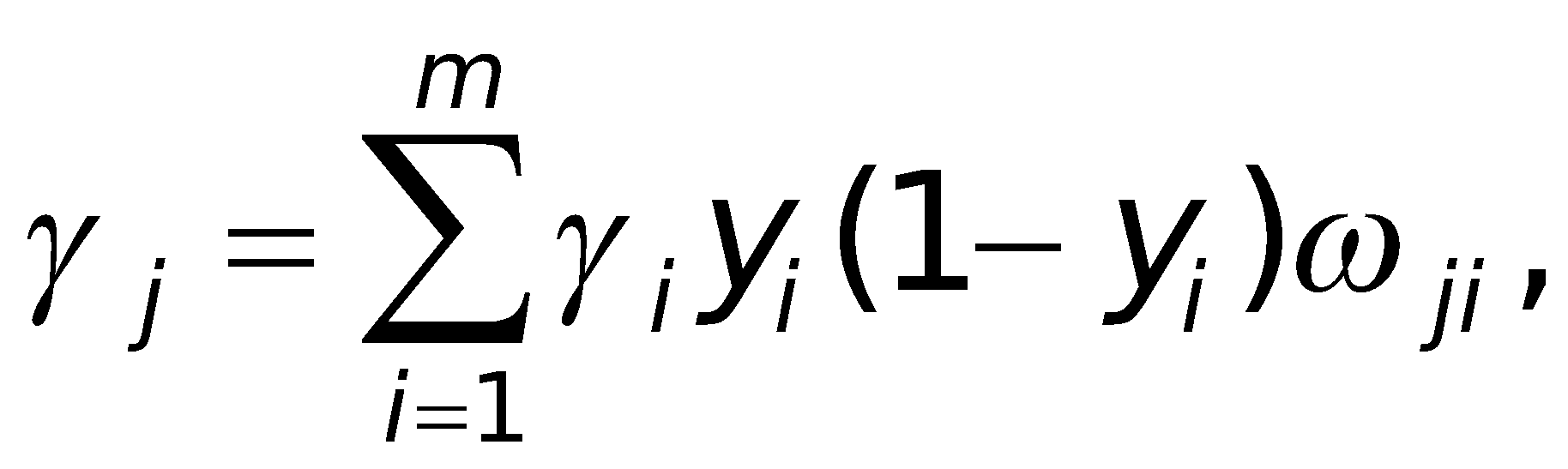
**

**

Ошибка для *j*-го нейрона выходного слоя определяется, как

*.*

Для *j*-го нейронного элемента скрытого слоя:



# ПРОЕКТИРОВАНИЕ АЛГОРИТМОВ ОБУЧЕНИЯ

Для обучения сети будем использовать алгоритм обратного распространения ошибки. Он является эффективным средством обучения нейронных сетей и представляет собой следующую последовательность шагов:

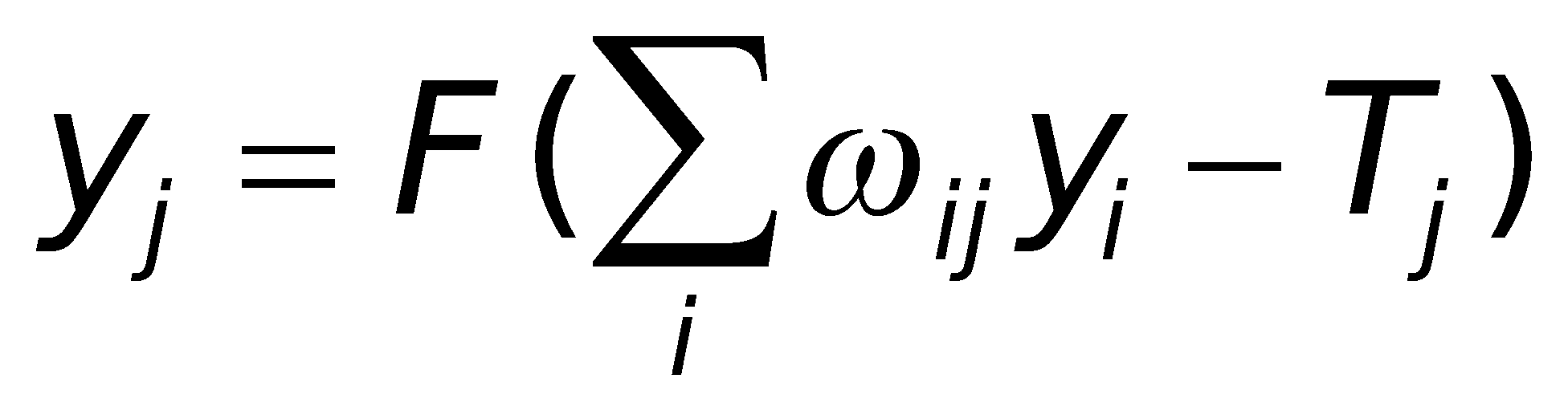
1. Задается шаг обучения α (0> α>1) и желаемая среднеквадратичная

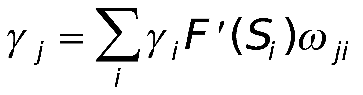
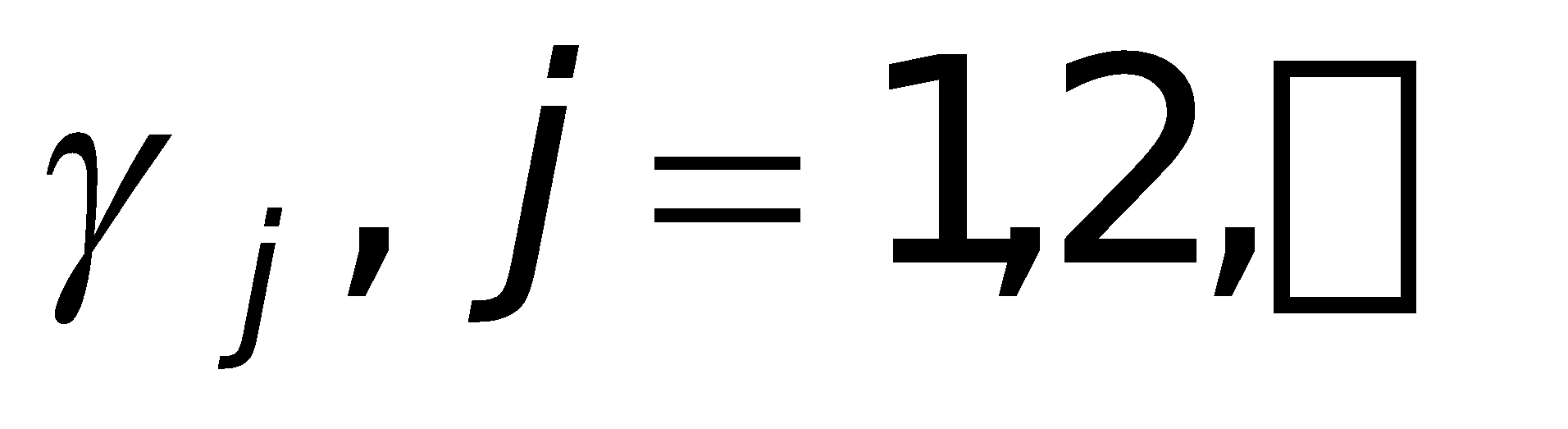
ошибка нейронной сети .

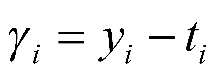
2. Случайным образом инициализируются весовые коэффициенты и пороговые значения нейронной сети.

3. Последовательно подаются образы из обучающей выборки на вход нейронной сети. При этом для каждого входного образа выполняются следующие действия:

a) производится фаза прямого распространения входного образа по нейронной сети. При этом вычисляется выходная активность всех нейронных элементов сети:

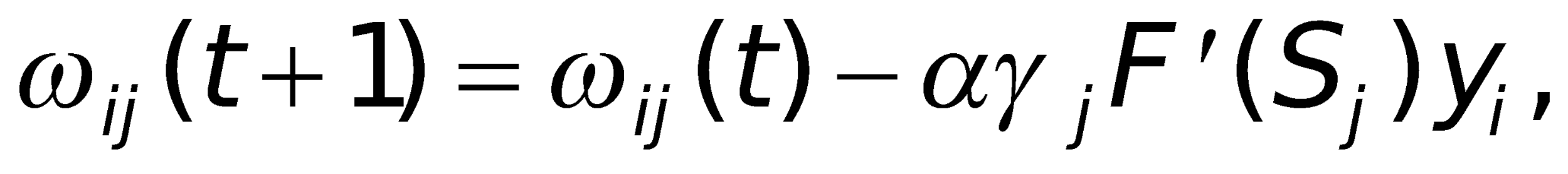
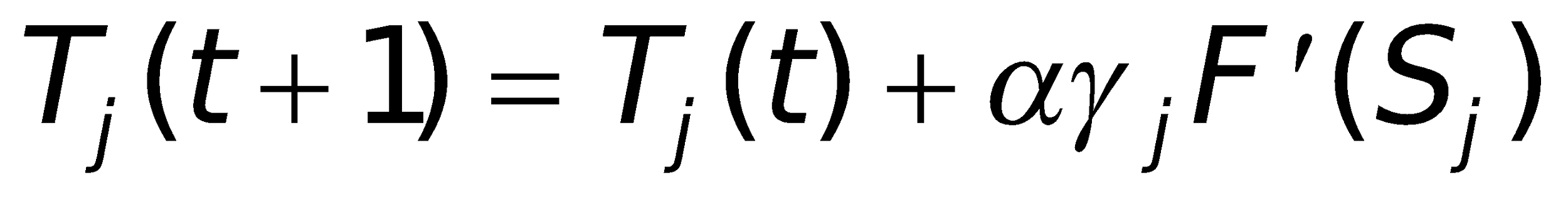
,   
где индекс j характеризует нейроны следующего слоя по отношению к слою i.

b) производится фаза обратного распространения сигнала, в результате которой определяется ошибка нейронных элементов для всех слоев сети. При этом соответственно для выходного и скрытого слоев:

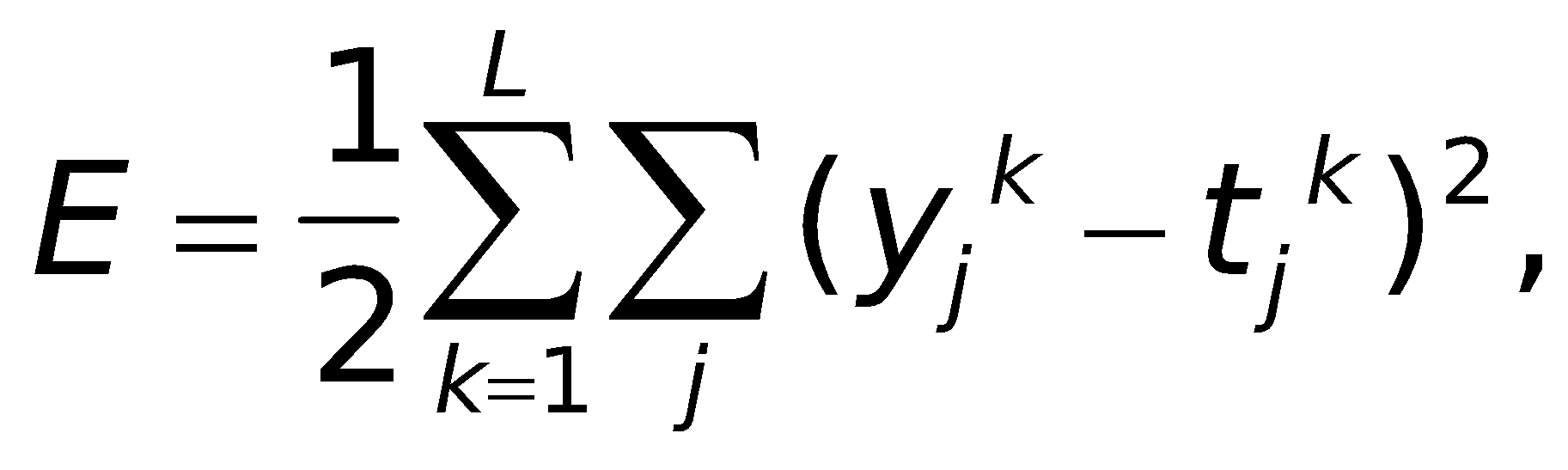


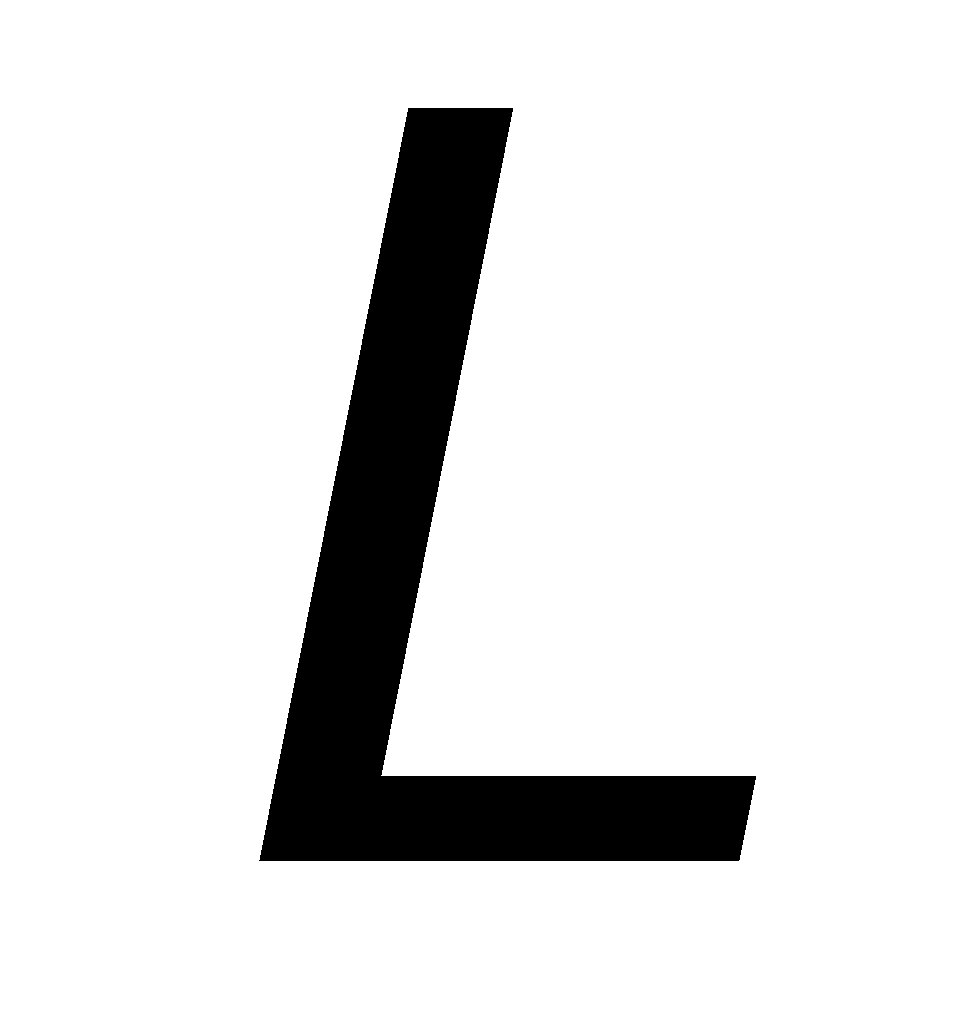
В последнем выражении индекс i характеризует нейронные элементы  
следующего слоя по отношению к слою j.

c) для каждого слоя нейронной сети происходит изменение весовых коэффициентов и порогов нейронных элементов:

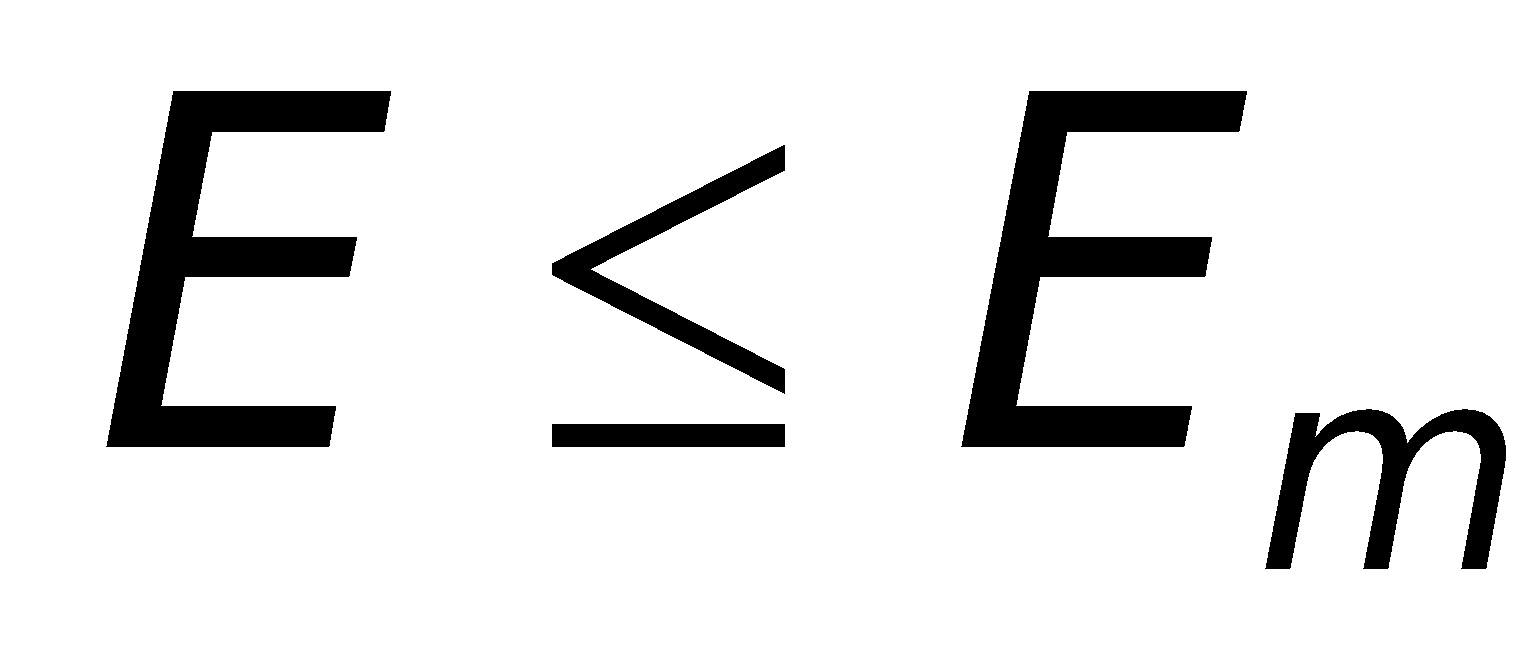
 .

4. Вычисляется суммарная среднеквадратичная ошибка нейронной сети:



где — размерность обучающей выборки.

5. Если E. > . то происходит переход к шагу 3 алгоритма. В противном случае алгоритм обратного распространения ошибки заканчивается.

Таким образом, данный алгоритм функционирует до тех пор, пока суммарная среднеквадратичная ошибка сети не станет меньше заданной, т. е. .

# ТЕСТИРОВАНИЕ И РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

Необходимо, чтобы нейронная сеть обучалась прогнозировать функцию с достаточной точностью, при этом обучение происходило за приемлемый промежуток времени.

Как уже отмечалось, размерности выборок для обучения и тестирования равны 750 точкам.

Проанализировав результаты прогнозирования для нейронной сети с различными комбинациями нейронов входного и скрытого слоев, было принято решение использовать двенадцать нейронов в скрытом слое.

Лучшие сочетания точности и времени были достигнуты при шаге обучения . При этом допустимая точность, которая достигается за достаточный промежуток времени равна .

Целью этого этапа курсового проекта является определение тех параметров нейронных сетей, которые не были определены на этапе проектирования. Для заданных архитектур (3/p/3, 6/p/3) необходимо определить какое количество нейронов на входном слое является оптимальным для решения задачи.

Главным критерием, на основании которого будем определять эти параметры, является ошибка аппроксимации и прогнозирования функции (1.1), причем нейронная сеть должна обладать как достаточной точностью аппроксимации функции, так и приемлемой точностью прогнозирования. Точность аппроксимации и прогнозирования функции (1.1) нейронной сетью должна определяться как визуально по степени сходства оригинального графика функции и графика, построенного по выходным значениям нейронной сети, так и аналитически на основе расчета суммарной квадратичной ошибки. Также на данном этапе необходимо провести сравнительный анализ полученных результатов, на основании чего определить нейронную сеть, которая наилучшим образом подходит для решения поставленной задачи. Ниже приведен пример результатов визуальной аппроксимации и прогнозирования функции (1.1):

Параметры обучения:

- Число нейронных элементов в скрытом слое p = 12;

- Обучающая выборка L = 750;

- Желаемая ошибка сети Em = 0,05;

- Шаг обучения α = 0,05.

Результаты аппроксимации и прогнозирования функции:

(значения усреднены для 10-ти экспериментов с разными инициализирующими веса и пороги значениями)

Для архитектуры 3/12/3:

- Суммарная среднеквадратичная ошибка Е = 0.894417

- Время обучения Т = 1 минута 40 секунд

Для архитектуры 6/12/3:

- Суммарная среднеквадратичная ошибка Е = 1.830105

- Время обучения Т = 1 минута 46 секунд

Визуализация результатов:

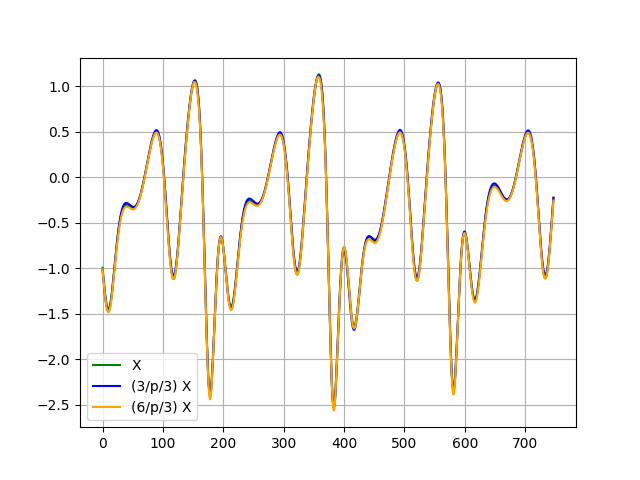


Рис. 5.1. Зависимость X(t)

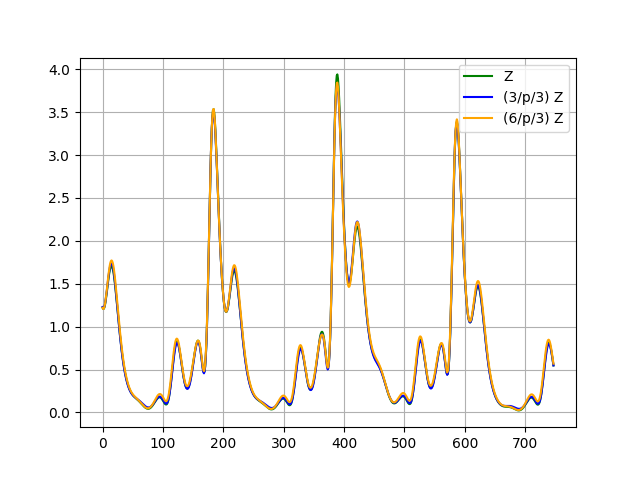


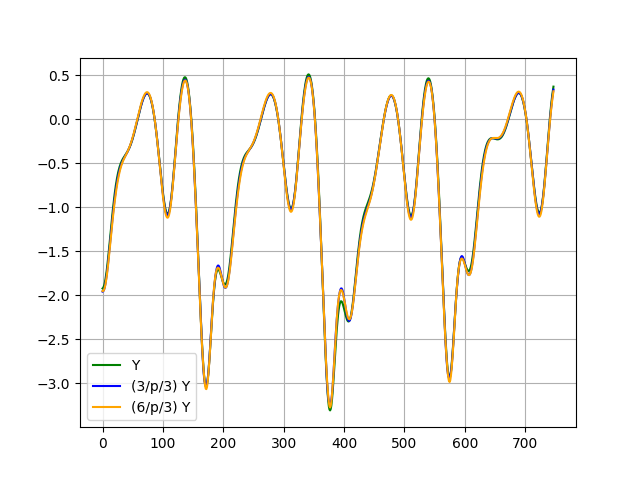
Рис. 5.2. Зависимость Y(t) 

Рис. 5.3. Зависимость Z(t)

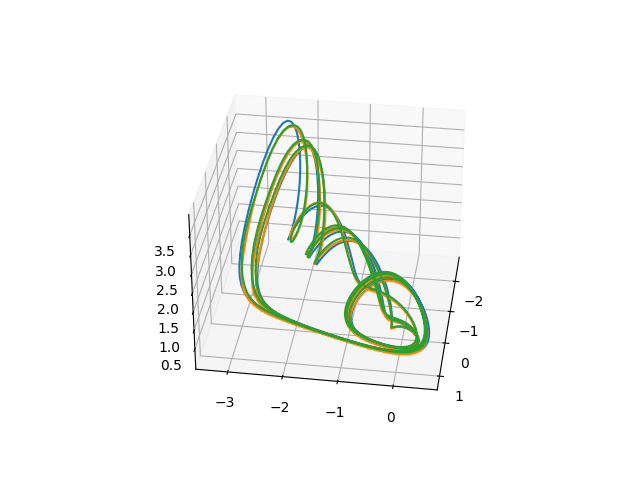


Рис. 5.4. Трёхмерное изображение функций

Обучаясь примерно одинаковое время с аналогичными гиперпараметрами и данными в обучающей выборке, многослойный персептрон с тремя нейронами на входе показывает намного большую точность прогнозов, чем персептрон аналогичной архитектуры с шестью нейронными элементами на входном слое. Следовательно для решения данной задачи с заданной функцией (1.1) предпочтительней будет выбрать архитектуру с меньшим количеством нейронов на входном слое.

Для лучшей из рассмотренных архитектур нейронной сети, прогнозируя следующее значение для первого элемента из тестовой выборки, а затем делая прогноз для получившегося значения и т.д. попытаемся восстановить тестовый набор.

Переобучив сеть на выборке из тысячи элементов с параметрами:

- желаемая минимальная среднеквадратичная ошибка = 0.001

- шаг обучения α = 0.07

и пытаясь восстановить соответствующую тестовую последовательность получили:

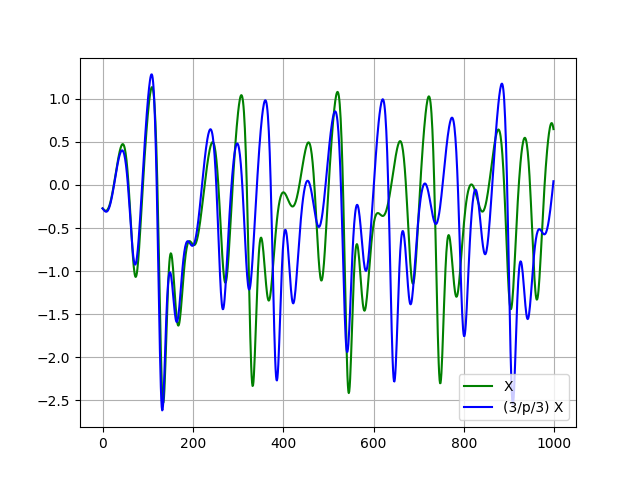


Рис. 5.5. Зависимость X(t)

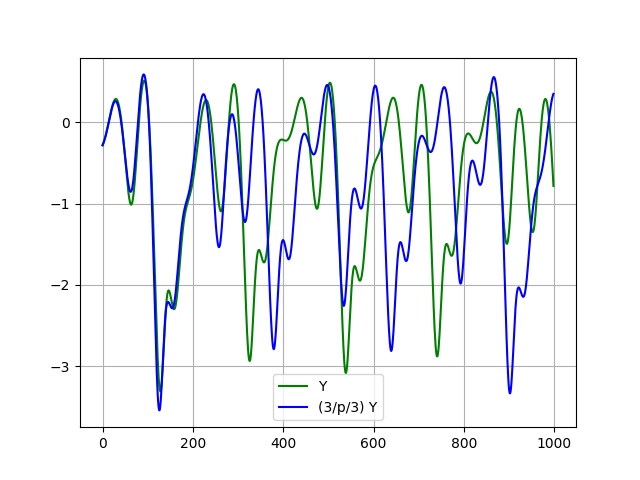


Рис. 5.6. Зависимость Y(t)

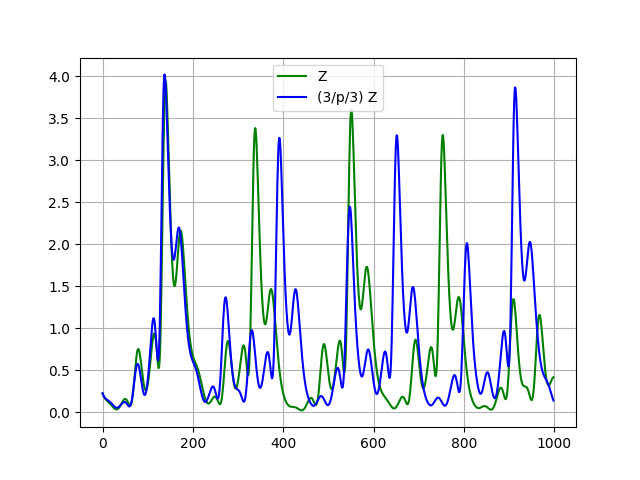


Рис. 5.7. Зависимость Z(t)

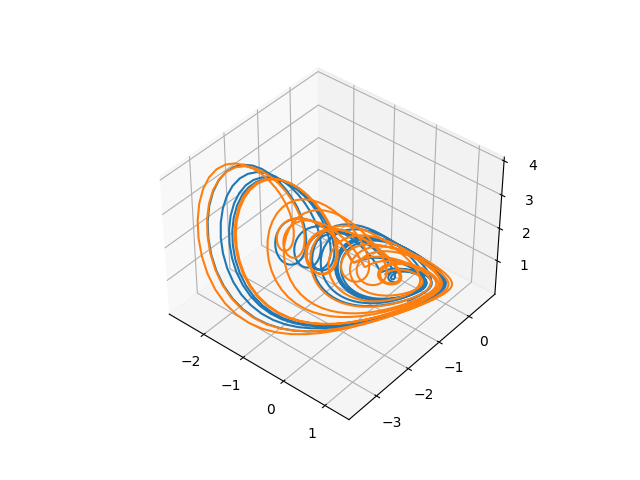


Рис. 5.8. Трёхмерное изображение функций

Как видно из получившейся визуализации, алгоритм сумел с относительно хорошей точностью спрогнозировать первые несколько сотен следующих элементов тестовой выборки, имея лишь первый. Однако постепенно возрастающая погрешность далее даёт сильно отличающийся от ожидаемого результат. Следовательно, такой алгоритм не способен сделать “долгосрочный прогноз” с заданной точностью обучения из-за заметного “лавинного эффекта”