Современная теория управления:

1. Базовые направления современной теории управления
2. Искусственные нейронные сети (ИНС)
3. ИНС в задачах управления

Литература

1. Методы классической и современной теории автоматического управления: Учебник в 5-и тт.; 2-е изд., перераб. и доп. Т.5: Методы современной теории автоматического управления / Под ред. К.А. Пупкова, Н.Д. Егупова. — М.: Издательство МГТУ им. Н.Э. Баумана, 2004. — 784 с.;

2. ИЗБРАННЫЕ ГЛАВЫ ТЕОРИИ АВТОМАТИЧЕСКОГО УПРАВЛЕНИЯ с примерами на языке MATLAB Б.Р. Андриевский, А.Л.Фрадков;

3. Александров А. Г. Оптимальные и адаптивные системы. М.: Высшая школа, 1989. 263 с. ISBN 5-06-000037-0;

4. Тюкин И. Ю., Терехов В. А., Адаптация в нелинейных динамических системах, Санкт-Петербург: ЛКИ, 2008. - 384 с. ISBN 978-5-382-00487-7

Литература

1. Adaptive control Astrom K.J., Wittenmark;

2. Robot Modeling and Control Mark W. Spong, Seth Hutchinson, and M. Vidyasagar;

3. Robot Manipulator Control. Theory and Practice Frank L.Lewis, Darren M.Dawson, Chaouki T.Abdallah

Karl Johan Åström

Mark W. Spong

Frank L. Lewis

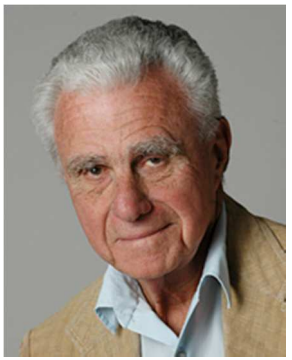
Современная теория управления

- Теория нелинейного управления
- Теория оптимального управления
- Теория адаптивного управления
- Теория робастного управления
- Теория нечёткой логики в задачах управления
- Теория интеллектуального управления

Оптимальное управление

Методы оптимального управления:

- вариационное исчисление,
- принцип максимума Понтрягина,
- динамическое программирование Беллмана,
- синтез оптимальных регуляторов (LQR, LQG, фильтр Калмана)



Ричард Эрнст Беллман (англ. Richard Ernest Bellman; 1920—1984) — американский математик, один из ведущих специалистов в области математики и вычислительной техники.

Сформулируем задачу оптимального управления:

Заданы уравнения состояния системы:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t)$$

Определены граничные условия:

$$\mathbf{x}(t_0) = \mathbf{x}_0^*$$

$$\mathbf{x}(t_1) = \mathbf{x}_1^*$$

Минимизируемый функционал:

$$I = \int_{t_0}^{t_1} f_0(\mathbf{x}(\tau), \dot{\mathbf{x}}(\tau), \mathbf{u}(t), \tau) d\tau$$

$\mathbf{x}$(t) — вектор состояния

$\mathbf{u}$(t) — вектор управления,

t_0, t_1 — начальный и конечный моменты времени.

Задача оптимального управления заключается в нахождении функций состояния $\mathbf{x}$(t) и управления $\mathbf{u}$(t) для времени $[t_0, t_1]$, которые минимизируют функционал I.

Принцип максимума

Заданы уравнения состояния системы:

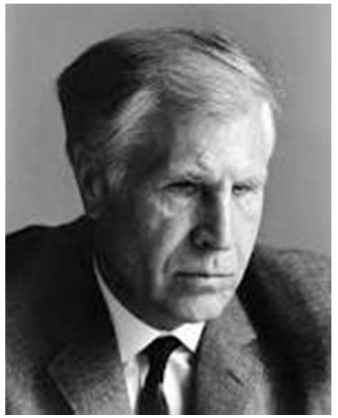
$$\begin{cases} \dot{x}_1(t) = f_1(x_1(t),\ldots,x_n(t),u(t),t) \\ \ldots \\ \dot{x}_n(t) = f_n(x_1(t),\ldots,x_n(t),u(t),t) \end{cases}$$

Определен критерий оптимальности: $I = \int_{t_0}^{t_1} f_0(\mathbf{x}(\tau),\mathbf{u}(t),\tau)d\tau$

$\mathbf{x}(\tau)$ -вектор состояния

Введем вспомогательные функции $\psi_1,\psi_2,\ldots,\psi_n$ и функционал:

$$H = -f_0(\mathbf{x},u) + \sum_{i=1}^{n} \psi_i(t) f_i(\mathbf{x},u)$$



Лев Семёнович Понтря́гин (1908-1988)— советский математик, один из крупнейших математиков XX века, академик АН СССР

Для оптимальности системы необходимо существование таких ненулевых непрерывных функций $\psi_i(t)$, удовлетворяющих уравнениям

$$\dot{\boldsymbol{\psi}} = -\frac{\partial H}{\partial \mathbf{x}}$$

при любых t на интервале управления, а функционал H достигает максимума. Обычно, максимум соответствует условиям $\frac{\partial H}{\partial \mathbf{u}} = 0, \frac{\partial^2 H}{\partial \mathbf{u}^2} < 0$

Принцип максимума: пример

Заданы уравнения состояния системы:

$J\ddot{\varphi} = M$ - Уравнение динамики

$x_2 = \dot{\varphi}$ - Угловая скорость

$x_1 = \varphi$ - Угол поворота

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = M / J \end{cases} \quad (1)$$

M- управляющий момент, J – момент инерции объекта

Определен критерий оптимальности по быстродействию: $I = \int_{t_0}^{t_1} 1 \cdot d\tau$ (2)

Для линейного одномерного объекта с уравнением состояния

$$\dot{\mathbf{x}} = A\mathbf{x} + Bu$$

$$H = -f_0(\mathbf{x}, u) + \mathbf{x}^T A^T \boldsymbol{\psi} + B^T \boldsymbol{\psi} u$$

$$\dot{\boldsymbol{\psi}} = \frac{\partial f_0}{\partial \mathbf{x}} - A^T \boldsymbol{\psi}$$

Для системы (1) и критерия (2)

$$A^T = \begin{vmatrix} 0 & 0 \\ 1 & 0 \end{vmatrix}, \frac{\partial f_0}{\partial \mathbf{x}} = 0$$

Тогда для данной задачи

$$\dot{\psi}_1 = 0, \dot{\psi}_2 = -\psi_1$$

$$\psi_1 = C_1$$

$$\psi_2 = C_2 - C_1 t$$

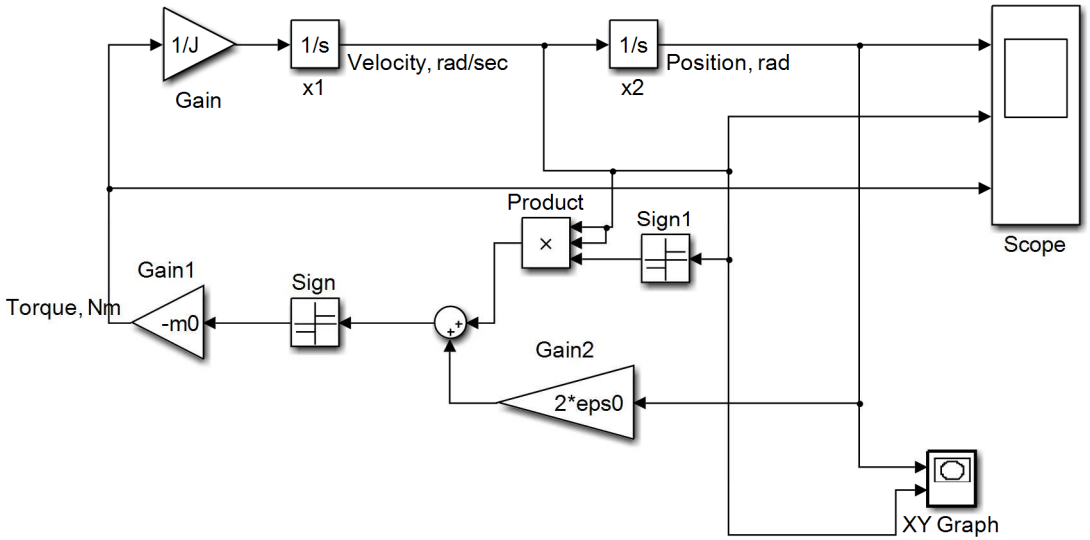
$$M = M_0 sign(C_2 - C_1 t)$$

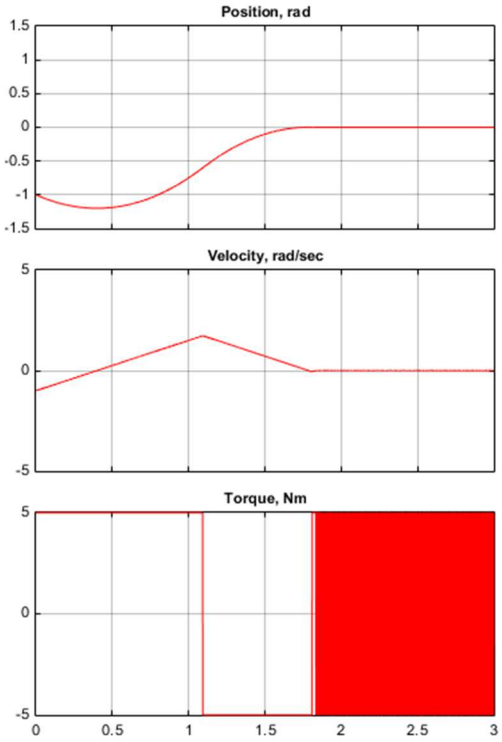
Удобнее находить управление по вектору состояния

$$M = -M_0 sign(sign(x_1)x_1^2 + 2\frac{M_0}{J}x_2)$$

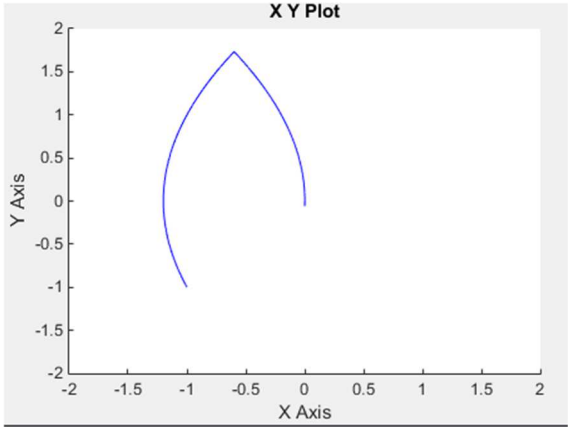
Принцип максимума: пример

Модель оптимального управления – «наивный» подход



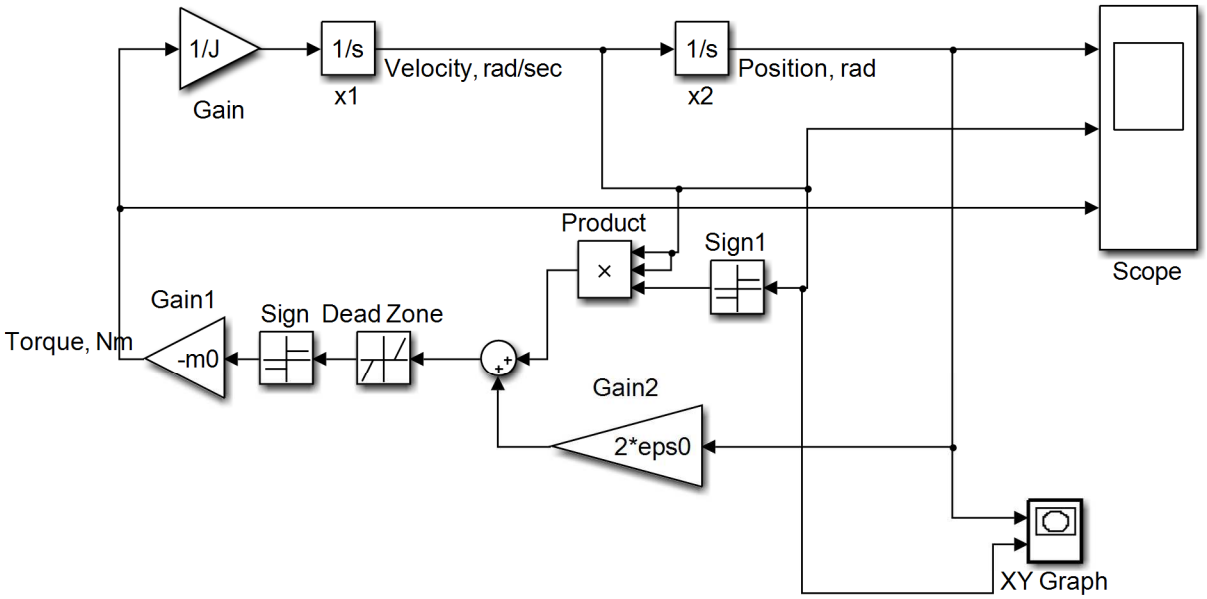


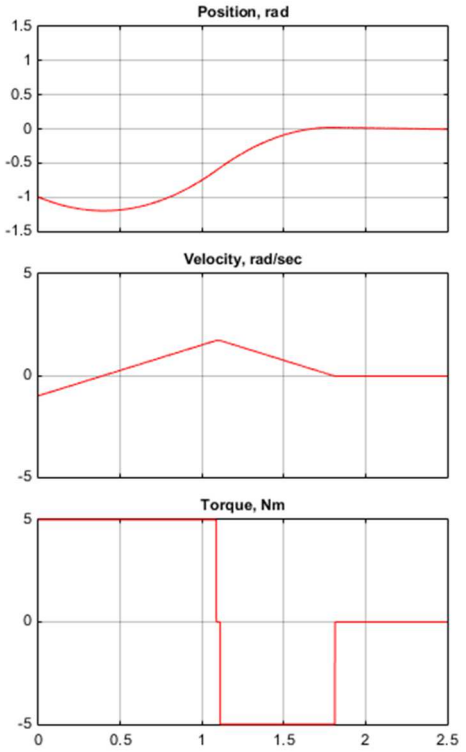
$$M = -M_0 sign(sign(x_1)x_1^2 + 2\frac{M_0}{J}x_2)$$



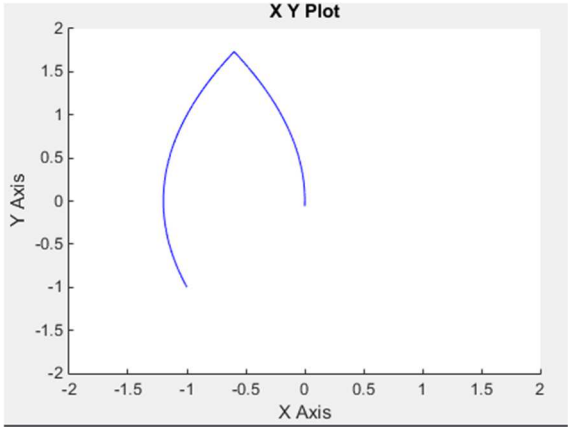
Принцип максимума: пример

Модель оптимального управления – введение нечувствительности



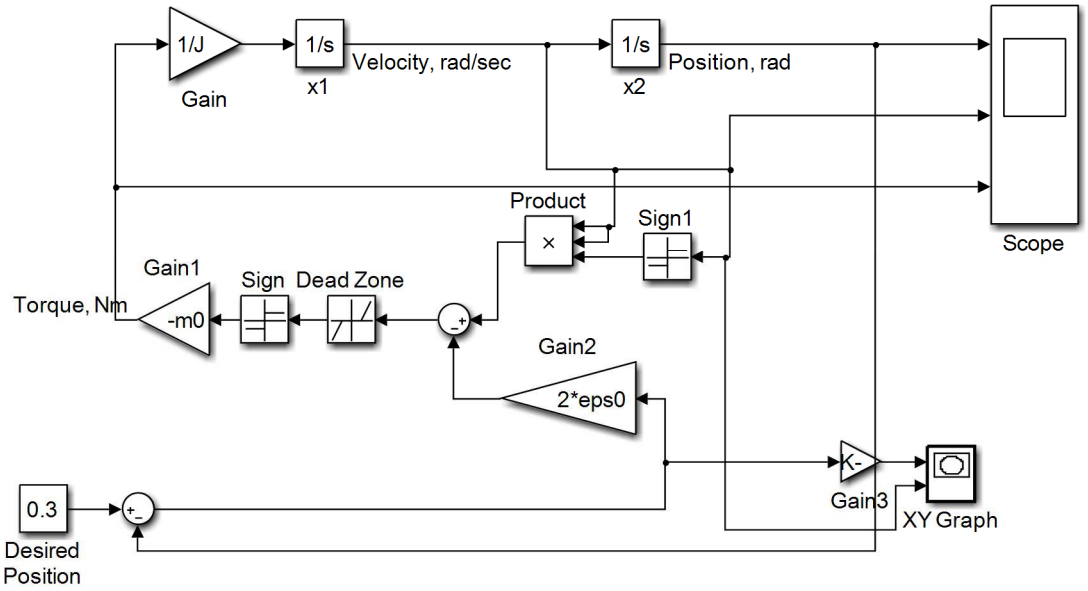


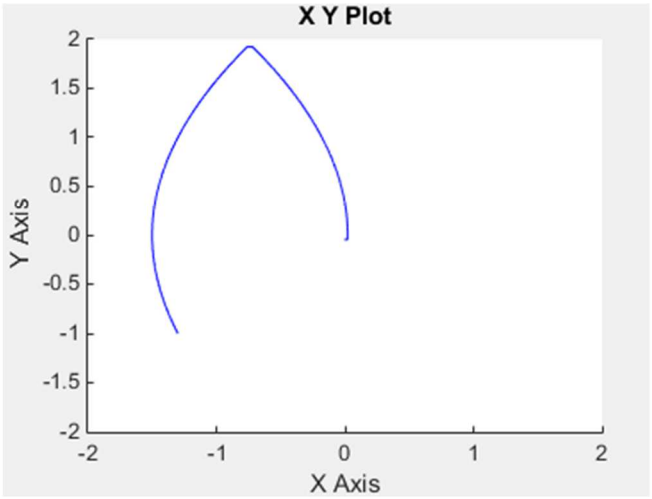
$$M = -M_0 sign(sign(x_1)x_1^2 + 2\frac{M_0}{J}x_2)$$

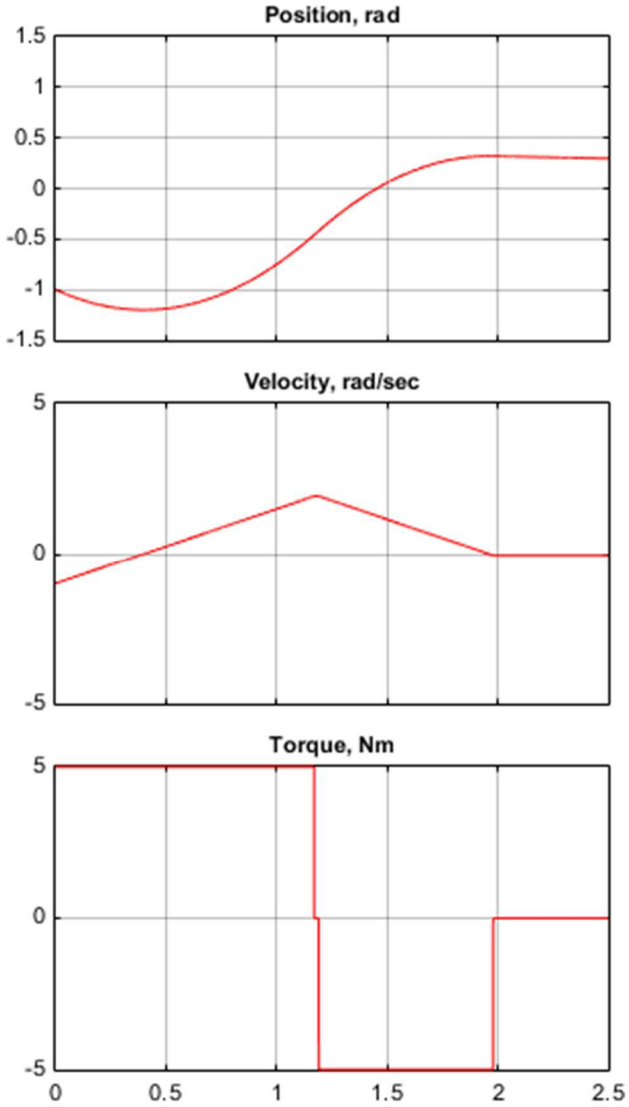


Принцип максимума: пример

Модель оптимального управления

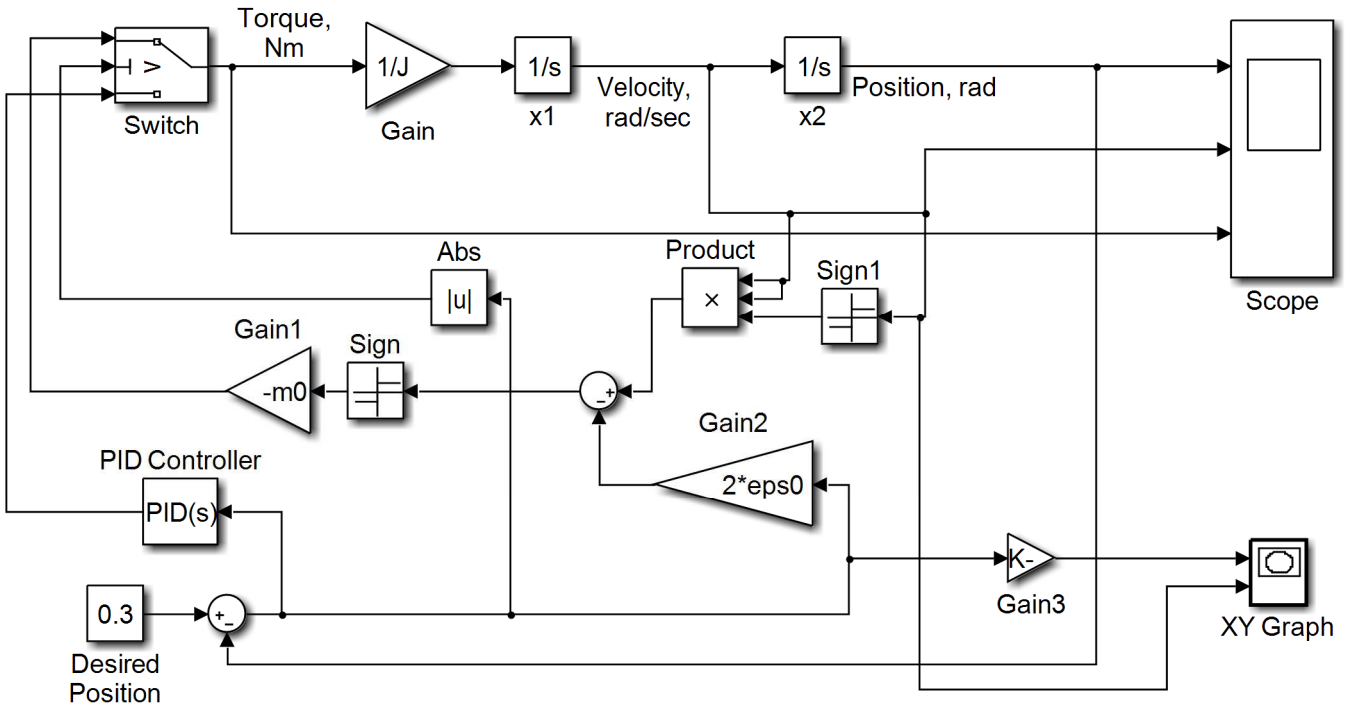


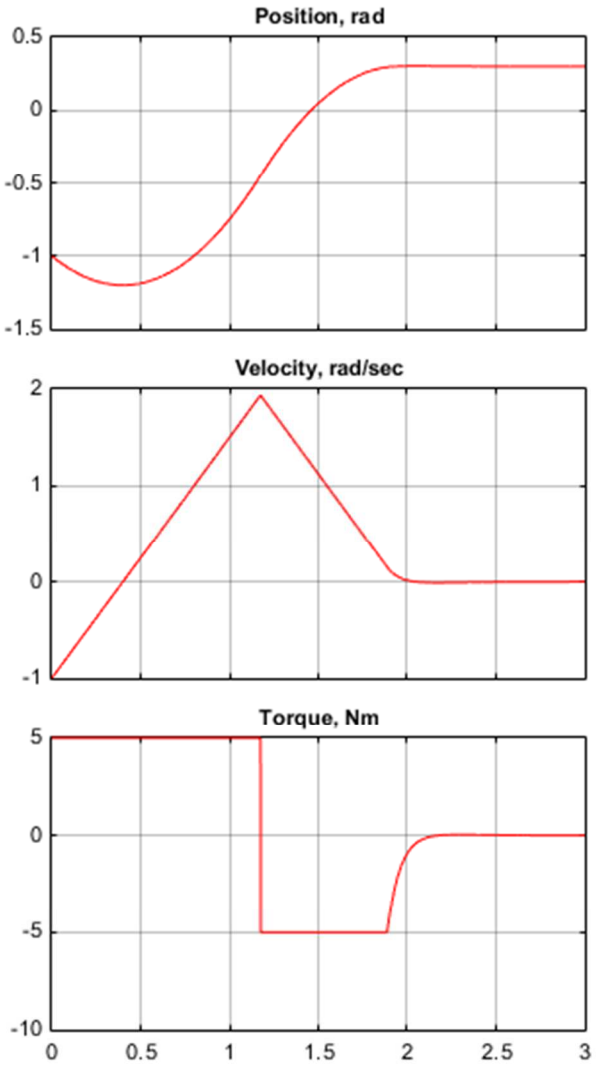




Принцип максимума: пример

Модель оптимального управления в системе с переменной структурой





Линейно квадратичное управление

Для непрерывных линейных систем, описываемых в пространстве состояний системой уравнений

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t),\ \ \mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t)$$

с критерием оптимальности

$$J(X,U) = \int_{t_0}^{t_1} \left(X^T QX + U^T RU\right) dt$$

закон управления по отрицательной обратной связи, найденный по LQR-алгоритму, должен минимизировать критерий оптимальности:

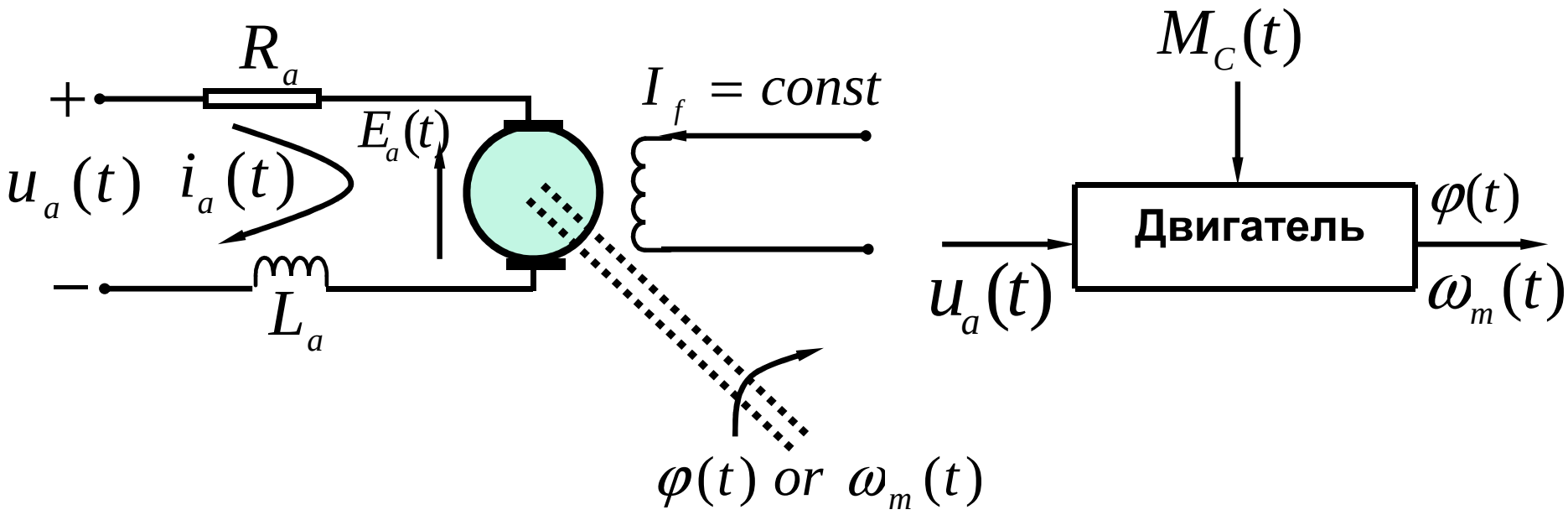
$$u = -Koc\mathbf{x}$$

$$Koc = RB^{-T}P$$

Где P – решение уравнения Риккати

$$A^T P + PA - PBR^{-1}B^T P + Q = 0$$

Модель ДПТ



$u_a(t)$ **- Напряжение якоря** $I_f = const$

$M_C(t)$ **- Внешний возмущающий момент**

$\omega_m(t)$ **- Скорость вращения вала двигателя**

Уравнение электромагнитной цепи двигателя:

$$L_a \frac{di_a(t)}{dt} + R_a i_a(t) + E_a(t) = u_a(t) \qquad (1)$$

$E_a(t)$ - противо-э.д.с. $E_a(t) = C_e \omega_m(t)$ C_e -конструктивная постоянная

Уравнение механической части

$$J_m \frac{d\omega_m(t)}{dt} + f_m \omega_m(t) = M_m(t) - M_C(t) \qquad (2)$$

$M_m(t)$ -электромагнитный момент

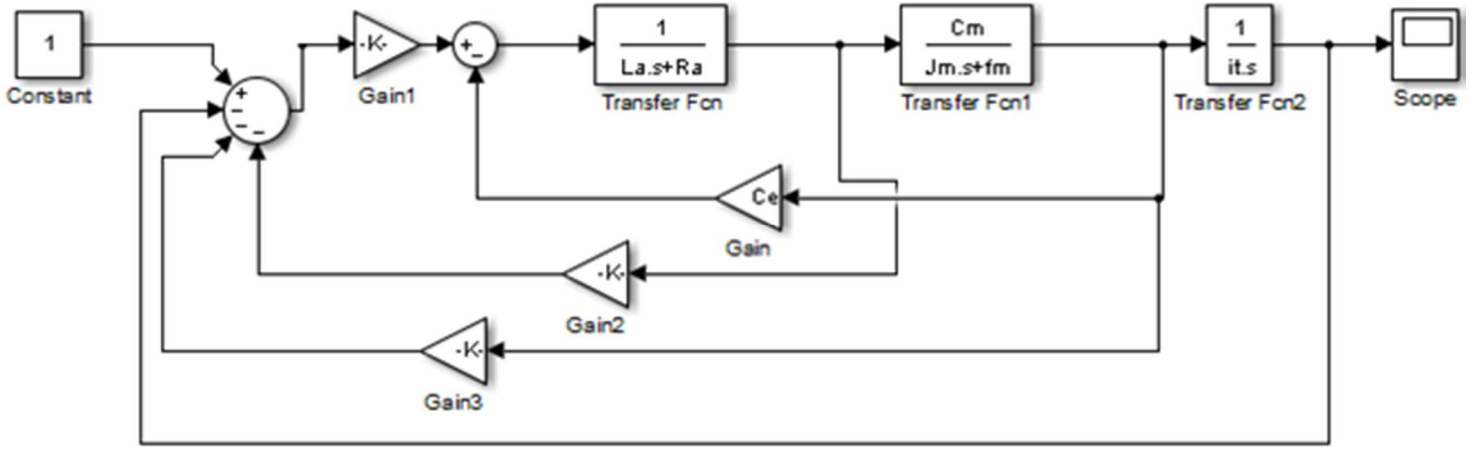
J_m -момент инерции ротора двигателя

f_m - коэффициент вязкого трения

$M_m(t) = C_m i_a(t)$ C_m - Конструктивная постоянная

Пример LQR синтеза привода на основе ДПТ Matlab

```
Ce=0.05;
Cm=0.05;
La=0.2;
Ra=1.5;
fm=1e-5;
Jm=1e-3;
it=200;
K=100;
```



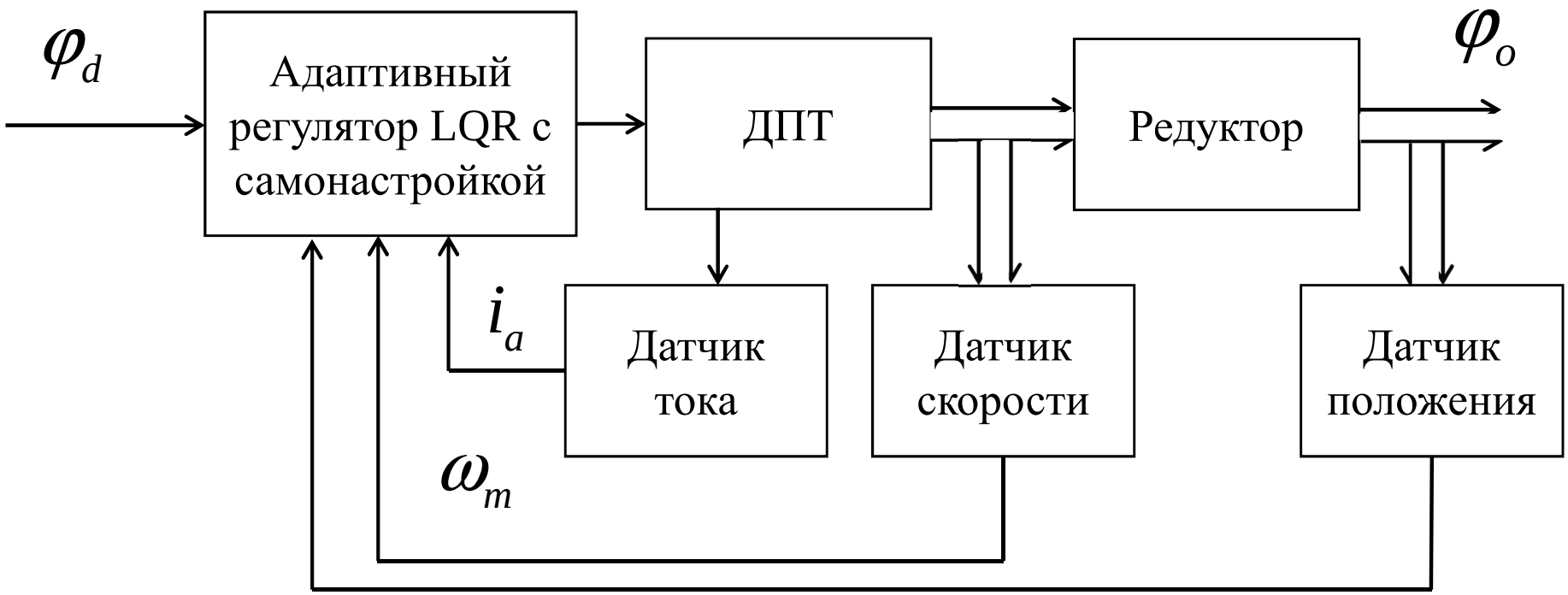

```
A=[-Ra/La -Ce/La 0;Cm/Jm -fm/Jm 0;0 1/it 0];
B=[K/La;0;0];
C=[0 0 -1];
D=1;
sys=ss(A,B,C,D);
Q=[1];
R=[0];
N=[0];
Koc=lqry(sys,Q,R,N)
sys1=ss(A-B*Koc,B,[0 0 1],0);
step(sys1);
```

Адаптивное управление

Адаптивные системы

- Самоорганизующиеся
- Самонастраивающиеся
  a. Поисковые адаптивные системы
  (экстремальные системы)
  b. Беспоисковые адаптивные системы
  - прямое адаптивное управление
  (адаптивные системы с эталонной моделью)
  - непрямое управление на основе идентификации
  параметров (идентификационный) подход

Пример:Адаптивное управление – индентификационный подход при управлении приводом на базе ДПТ



Идентификация – регрессионный подход

$\hat{i} = (\hat{J}_m \ddot{\alpha} + \hat{f}_m \omega_m) / C_m$ - Модель регрессии для оценки тока двигателя

$$\begin{cases} \dfrac{d\hat{\boldsymbol{\theta}}}{dt} = P\boldsymbol{\gamma} e \\ \dfrac{dP}{dt} = \beta P - P\boldsymbol{\gamma}\boldsymbol{\gamma}^T P \end{cases}$$

- Система уравнений регрессии

$e = i - \hat{i} = i - \boldsymbol{\gamma}^T \hat{\boldsymbol{\theta}}$ - Ошибка оценки тока двигателя

$$\hat{\boldsymbol{\theta}} = \begin{vmatrix} \hat{J}_m \\ \hat{f}_m \end{vmatrix}$$

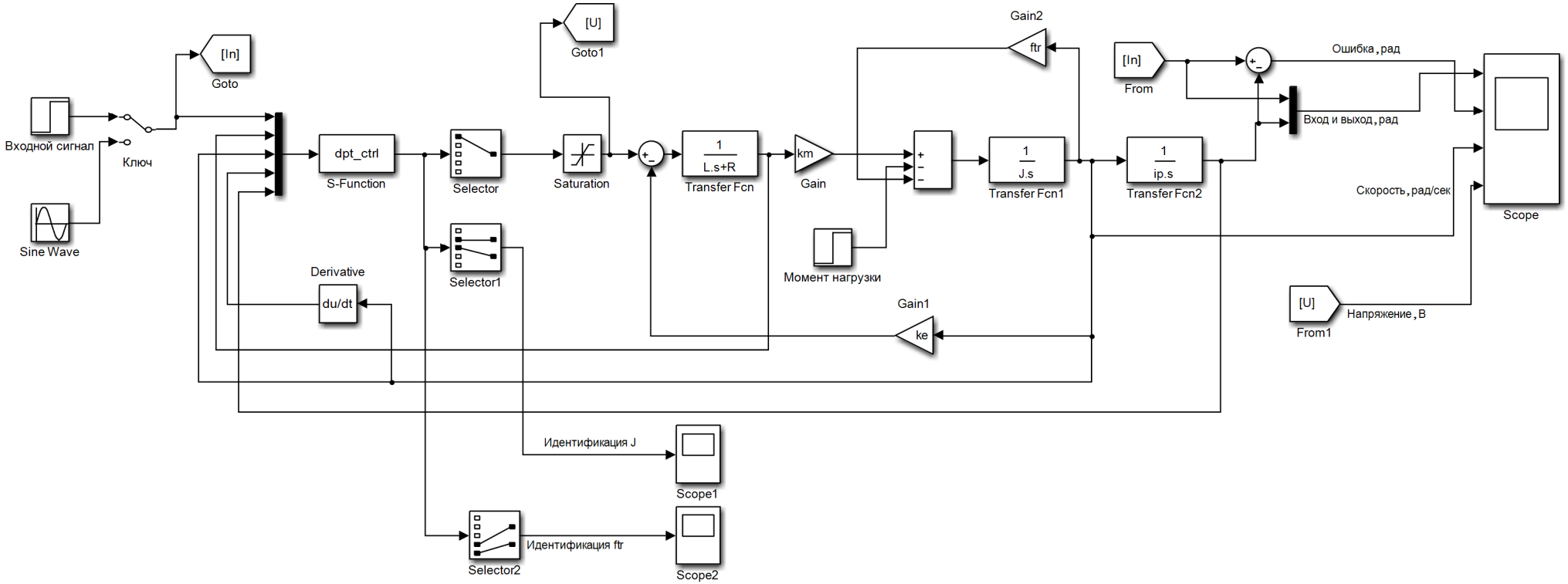
- Регрессоры –вектор неизвестных параметров

$$\boldsymbol{\gamma} = \begin{vmatrix} \dfrac{\ddot{\alpha}}{C_m} \\ \dfrac{\omega_m}{C_m} \end{vmatrix}$$

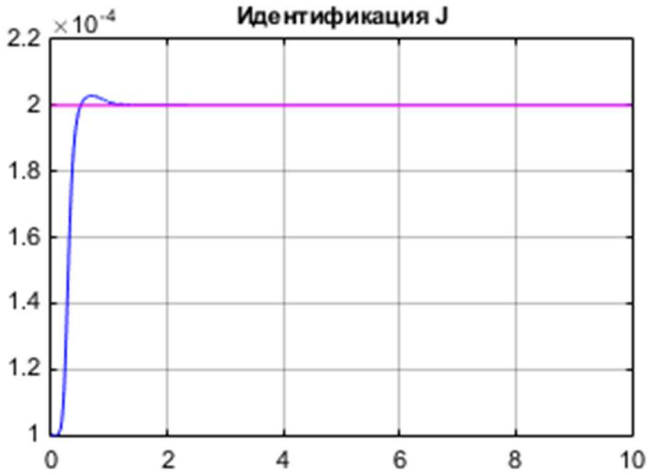
- Коэффициенты регрессии

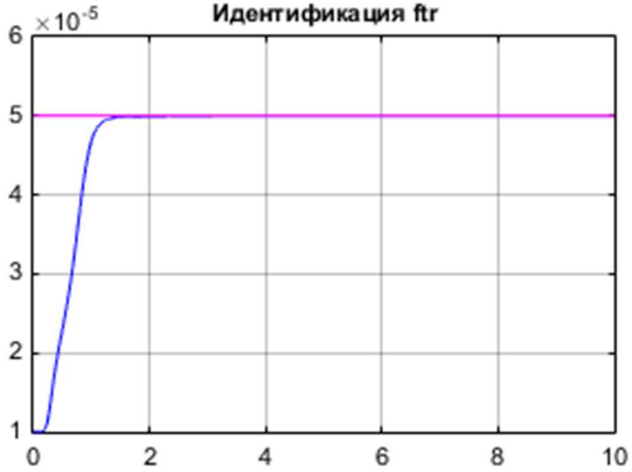
$\beta -$ - Настройка скорости сходимости

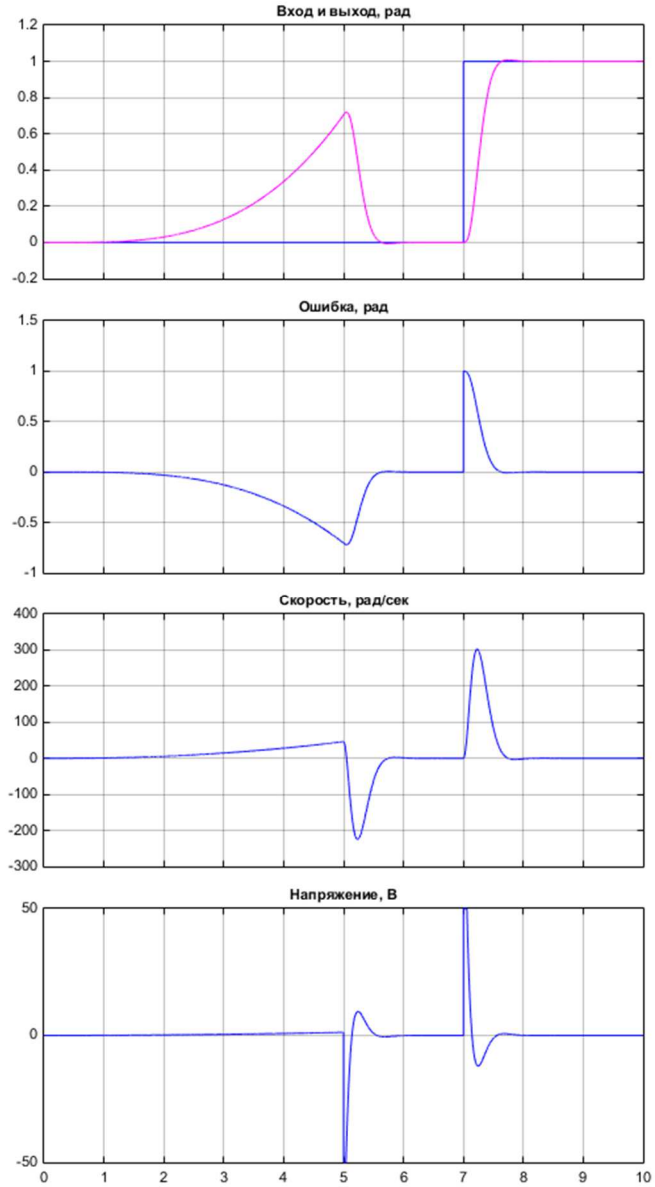
Пример:Адаптивное управление – индентификационный подход



Пример:Адаптивное управление – индентификационный подход







Управление на основе правил нечеткой логики

Базовая способность человеческого интеллекта - способность принимать правильные решения в обстановке неполной и нечеткой информации

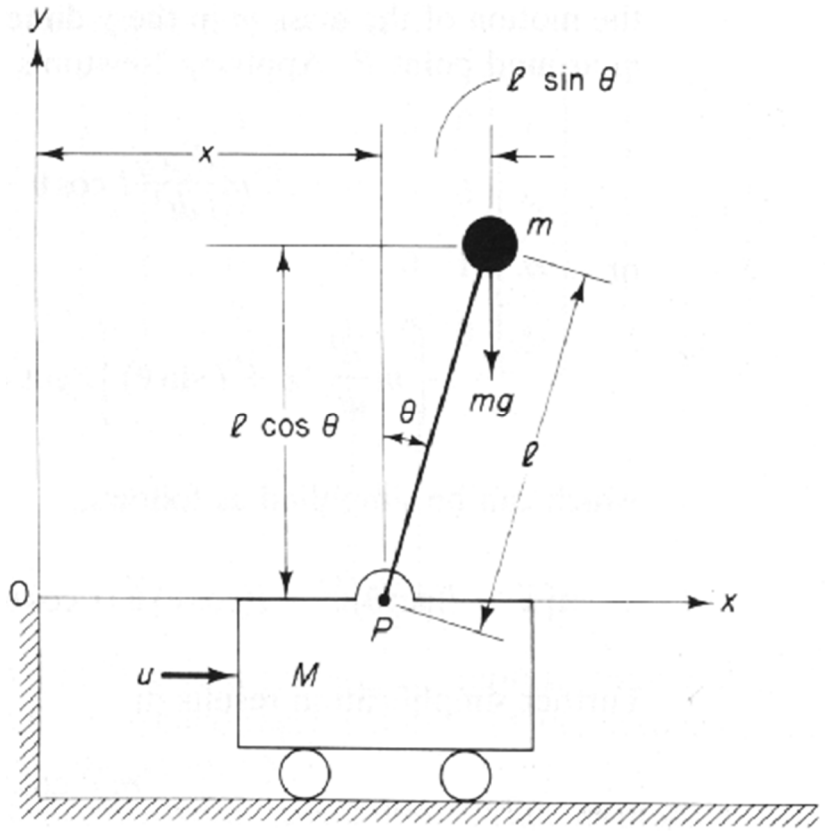
Лотфи А. Заде (Lotfl A. Zadeh)

Работа «**Fuzzy Sets**»(1965 г. журнал Information and Control, № 8)
Заложены основы моделирования интеллектуальной деятельности человека

Предметом нечёткой логики считается исследование рассуждений в условиях нечёткости, размытости, сходных с рассуждениями в обычном смысле, и их применение в вычислительных системах.
Научные направления:

- нечёткая логика в широком смысле (теория приближенных вычислений);
- нечёткая логика в узком смысле (символическая нечёткая логика).

Пример управления обратным маятником

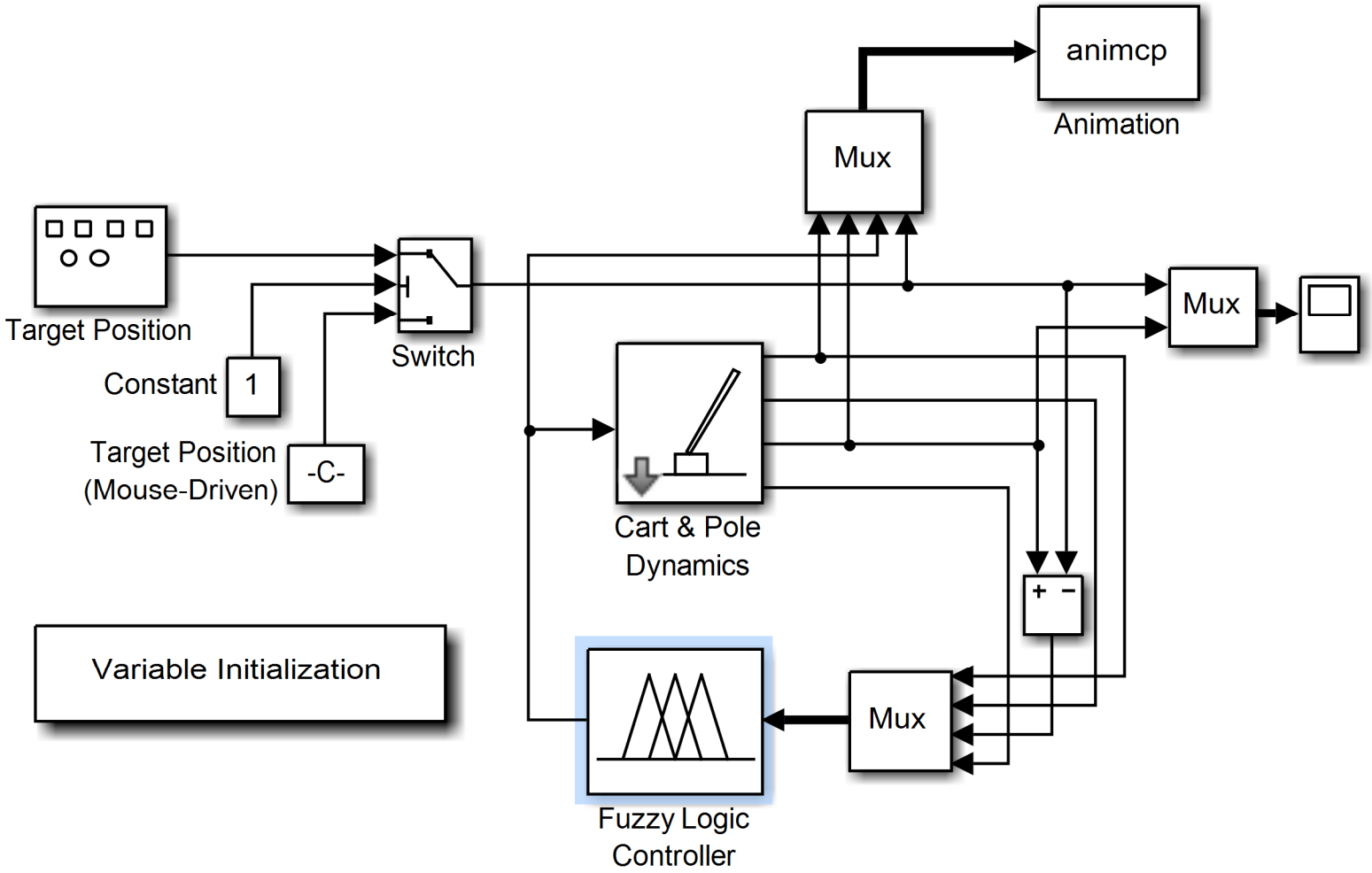


$$\ddot{\theta} = \frac{ucos(\theta) - ml\dot{\theta}^2 \cdot cos\theta \cdot sin\theta - (m+M)g\sin(\theta) - k_{tr2}\dot{\theta}}{mlcos^2(\theta) - (M+m)l}$$

$$\ddot{x} = \frac{u - ml \cdot \dot{\theta}^2 \cdot sin\theta - mgcos(\theta)\sin(\theta) - k_{tr1}\dot{x}}{M + m - mcos^2(\theta)}$$

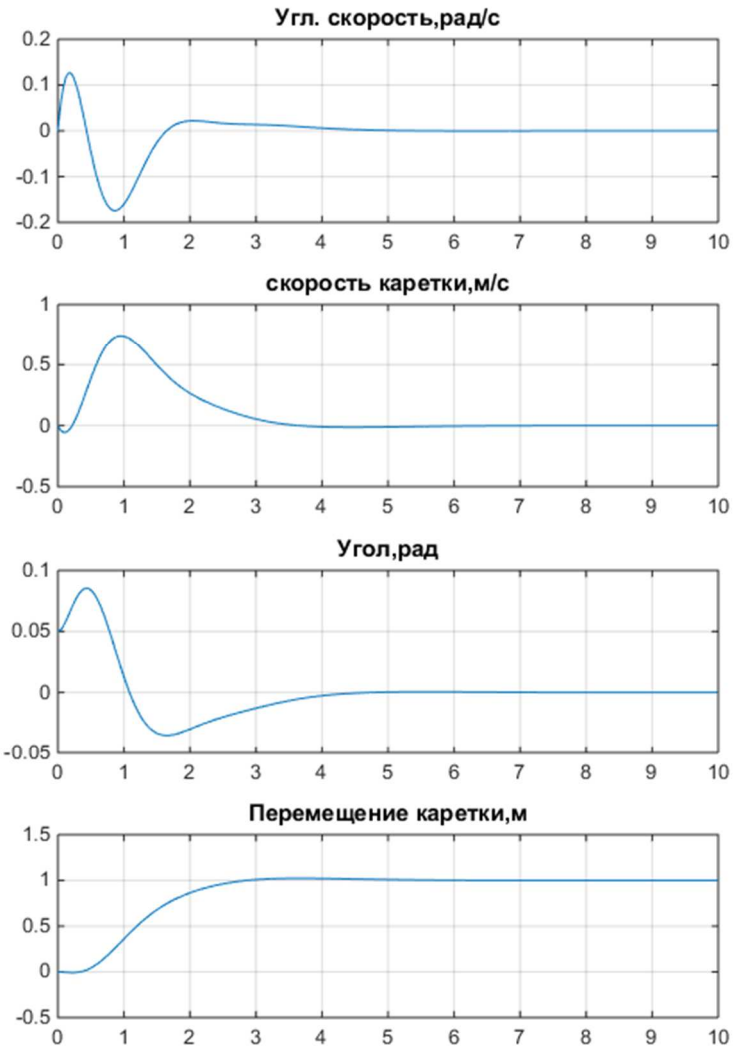
Пример управления обратным маятником - slcp

Cart and Pole System

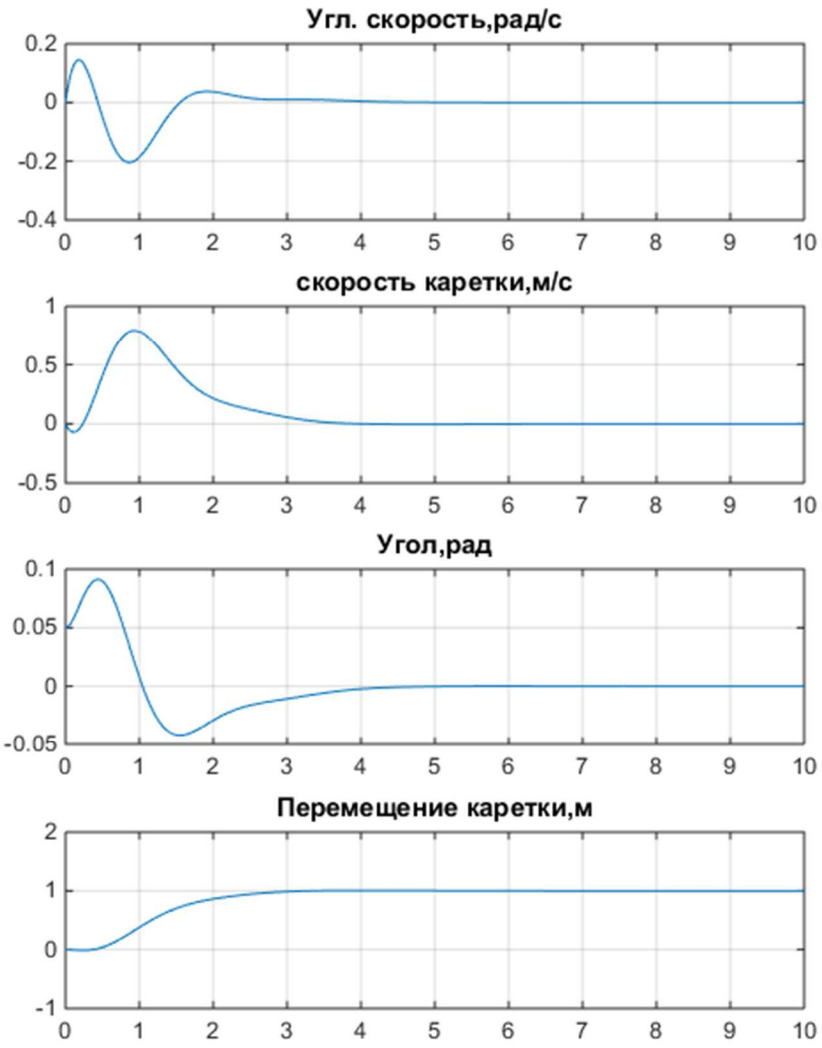


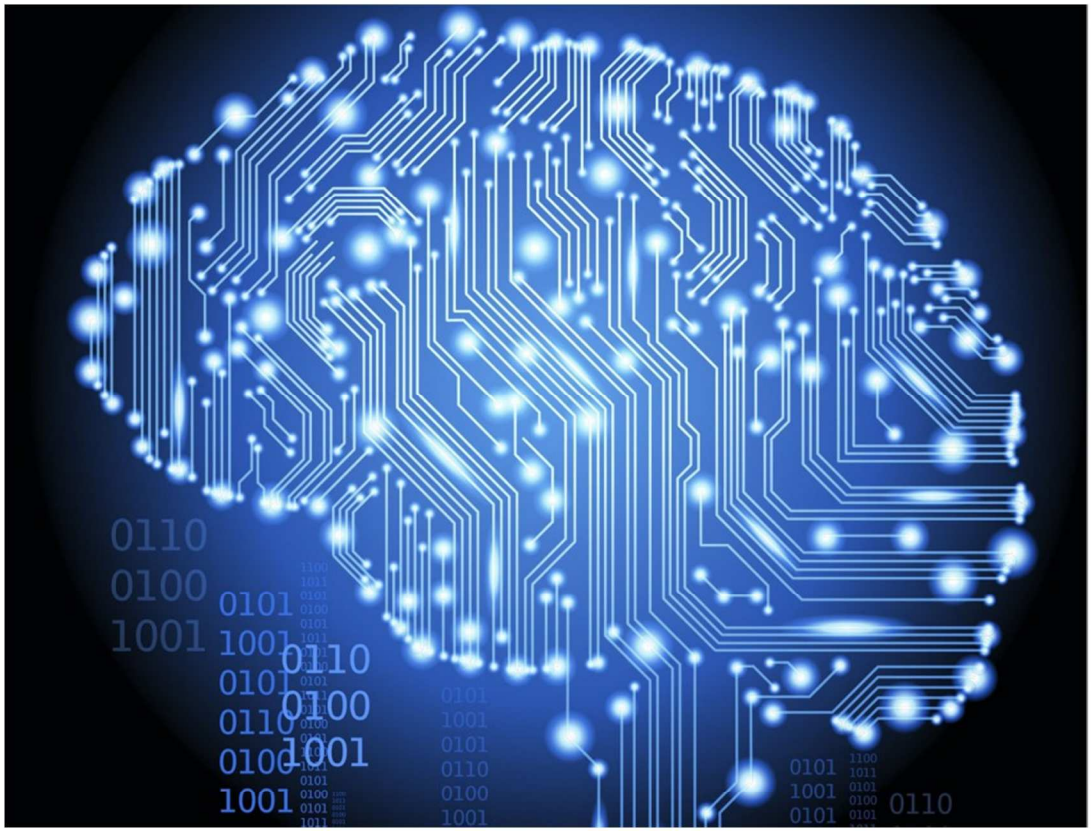
Пример управления обратным маятником

ПД регулятор



Fuzzy Logic регулятор





Искусственные нейронные сети

Характеристики мозга человека

- 10^{10}-10^{11}-количество нейронов
- 10^{14}-10^{15} – количество связей
- исходя из общего количества синапсов головного мозга, количество данных — около 125 петабайт
- Частота 100 Гц
- Параллельная обработка информации

В 2013 году учеными предпринята попытка имитации активности 1,73 миллиарда нервных клеток, соединенных 10,4 триллиона синапсов. Имитация длительностью всего в одну секунду показала, насколько мощным и даже могущественным является мозг каждого из семи миллиардов человек. Чтобы обсчитать все данные и наконец запустить имитацию, суперкомпьютеру пришлось работать на пределе своих возможностей в течение 40 минут. При этом в нем было задействовано почти 83 000 центральных процессоров и 1 петабайт памяти.

Моделирование мозга

Blue Brain Project — проект по компьютерному моделированию неокортекса человека. Начался в июле 2005 года. Над проектом совместно работают компания IBM и Швейцарский Федеральный Технический Институт Лозанны (École Polytechnique Fédérale de Lausanne — EPFL).

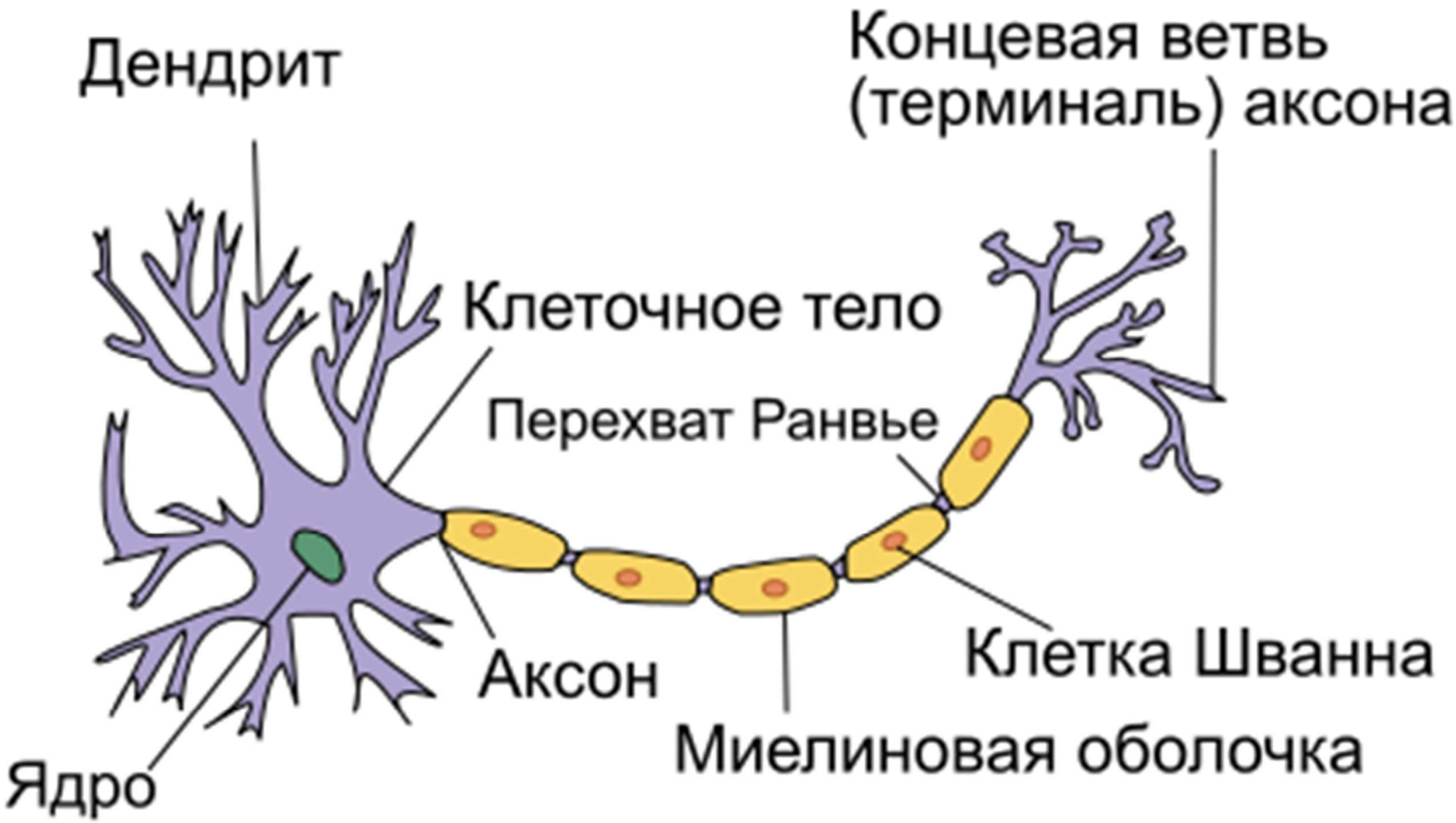
The Human Brain Project HBP (с англ. — «Проект Человеческий Мозг») — большой научно-исследовательский проект по изучению человеческого мозга, основанный в 2013 году в Женеве, Швейцария и координируемый Генри Маркрамом.

Основной структурной единицей неокортекса (новой коры головного мозга) человека является нейронная колонка. Одна такая колонка содержит порядка 10^3—10^4 нейронов, дендриты которых проходят через всю высоту колонки. Неокортекс и каждая его колонка состоит из 6 слоёв. Толщина каждого слоя примерно равна толщине кредитной карточки. Количество слоёв играет существенную роль в мыслительном процессе. Так, например, у собаки 4 слоя новой коры, из-за чего она не обладает способностью достаточно подробно прогнозировать ситуацию и не может вычислить следующее логическое действие.

Проект использует суперкомпьютер Blue Gene для моделирования колонок. В конце 2006 года удалось смоделировать одну колонку неокортекса молодой крысы. При этом использовался один компьютер Blue Gene и было задействовано 8192 процессора для моделирования 10000 нейронов. То есть практически один процессор моделировал один нейрон. Для соединения нейронов было смоделировано порядка $3 \cdot 10^7$ синапсов.

На текущий момент команда работает над «режимом реального времени», при котором 1 секунда реального времени работы мозга моделируется процессорами за 1 секунду.

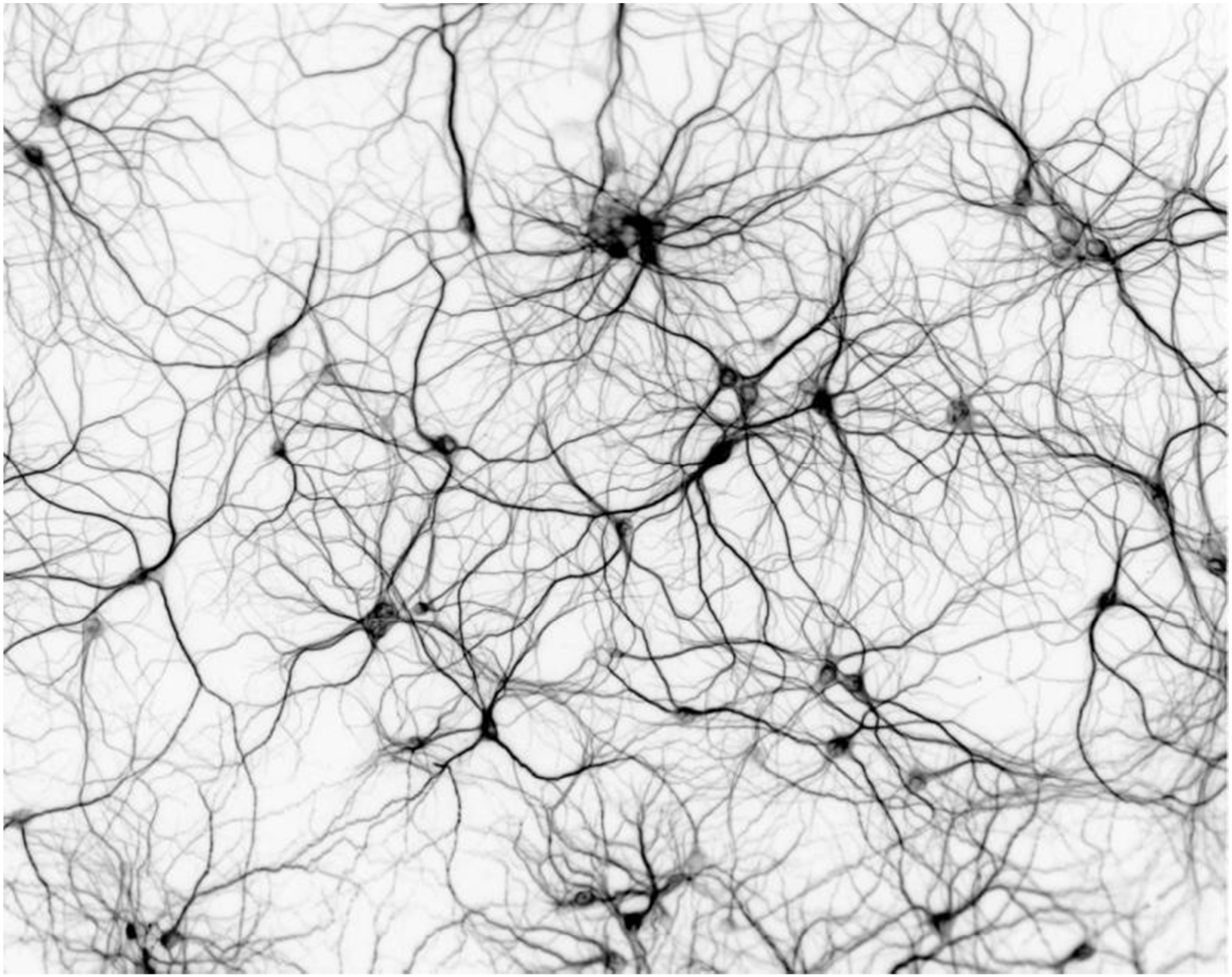
Типичная структура нейрона



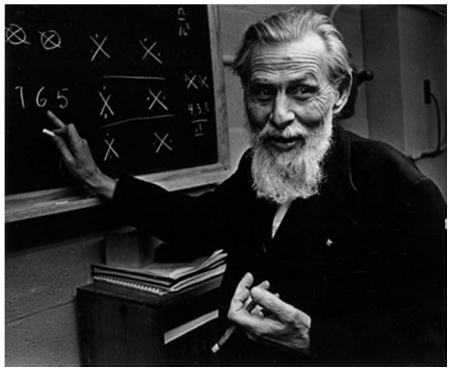
- Нервная клетка - это черный ящик, у которого есть дендриты, входы, по которым поступают сигналы (отрицательные ионы) в клетку или ее ядро.
- Если внутри накапливается большой отрицательный заряд, клетка возбуждается и генерирует импульс, который по аксону передается к дендритам следующих клеток.
- Место соединения аксона нейрона с дендритом называется синапсом.
- Заряды аддитивны, т.е. они накапливаются в клетках.

Отсюда клетка - это элементарный классификатор, принимающий решение, возбудиться или нет.

Биологическая нейронная сеть



Историческая справка



Уоррен Мак-Каллок
(Warren Sturgis McCulloch 1898-1969)

Уолтер Питтс
(Walter Pitts 1923—1969)

Фрэнк Розенблатт
(Frank Rosenblatt, 1928—1971)

- 1943 — У. Маккалок и У. Питтс формализуют понятие нейронной сети в фундаментальной статье о логическом исчислении идей и нервной активности
- 1948 — Н. Винер вместе с соратниками публикует работу о кибернетике. Основной идеей является представление сложных биологических процессов математическими моделями.
- 1949 — Д. Хебб предлагает первый алгоритм обучения.
- В 1958 Ф. Розенблатт изобретает однослойный перцептрон и демонстрирует его способность решать задачи классификации

Перцептро́н, или персептрон (англ. perceptron от лат. perceptio — восприятие; нем. Perzeptron) — математическая или компьютерная модель восприятия информации мозгом (кибернетическая модель мозга), реализованная в виде электронной машины «Марк-1» в 1960 году. Перцептрон стал одной из первых моделей нейросетей, а «Марк-1» — первым в мире нейрокомпьютером.

- В 1960 году Уидроу совместно со своим студентом Хоффом на основе дельта-правила (формулы Уидроу) разработали Адалин, который сразу начал использоваться для задач предсказания и адаптивного управления.
- В 1963 году в Институте проблем передачи информации АН СССР. А. П. Петровым проводится подробное исследование задач «трудных» для перцептрона.
- В 1969 году М. Минский публикует формальное доказательство ограниченности перцептрона и показывает, что он неспособен решать некоторые задачи. **Интерес к нейронным сетям резко спадает.**

- В 1972 году Т. Кохонен и Дж. Андерсон независимо предлагают новый тип нейронных сетей, способных функционировать в качестве памяти
- В 1973 году Б. В. Хакимов предлагает нелинейную модель с синапсами на основе сплайнов и внедряет её для решения задач в медицине, геологии, экологии
- 1974 — Пол Дж. Вербос и А. И. Галушкин одновременно изобретают алгоритм обратного распространения ошибки для обучения многослойных перцептронов.
- 1975 — Фукусима представляет когнитрон — самоорганизующуюся сеть, предназначенную для инвариантного распознавания образов
- 1982 — Дж. Хопфилд показал, что нейронная сеть с обратными связями может представлять собой систему, минимизирующую энергию (так называемая сеть Хопфилда)
- 1986 — Дэвидом И. Румельхартом, Дж. Е. Хинтоном и Рональдом Дж. Вильямсом и независимо и одновременно С. И. Барцевым и В. А. Охониным (Красноярская группа) переоткрыт и существенно развит метод обратного распространения ошибки.
- 2007 Джеффри Хинтоном в университете Торонто созданы алгоритмы глубокого обучения многослойных нейронных сетей.

А.Н. Колмогоровым и В.В. Арнольдом в 1957 году была доказана теорема о представимости непрерывных функций нескольких переменных суперпозицией непрерывных функций одной переменной, которая в 1987 году была переложена Хехт–Нильсеном для нейронных сетей:

Любая функция нескольких переменных может быть представлена двухслойной НС с прямыми полными связями с N нейронами входного слоя, (2N+1) нейронами скрытого слоя с ограниченными функциями активации (например, сигмоидальными) и M нейронами выходного слоя с неизвестными функциями активации.

Из теоремы Колмогорова–Арнольда–Хехт–Нильсена (КАХН) следует, что для любой функции многих переменных существует отображающая ее НС фиксированной размерности, при настройке (обучении) которой могут использоваться три степени свободы:

- область значений сигмоидальных функций активации нейронов скрытого слоя;
- наклон сигмоид нейронов этого слоя;
- вид функций активации нейронов выходного слоя.

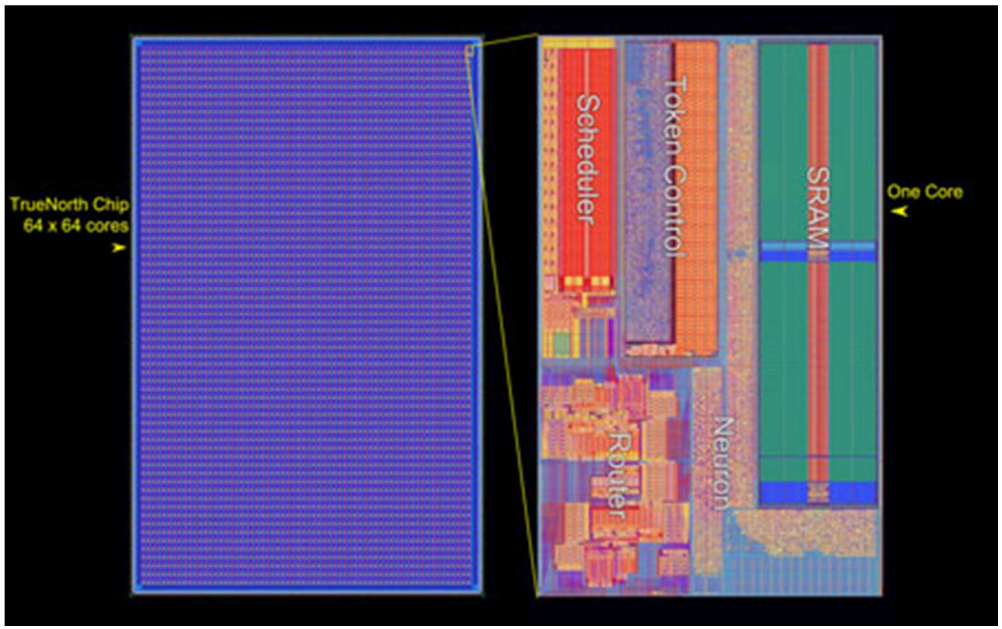
Аппаратная реализация ИНС

Нейрокомпьютеры

IBM- Brain-inspired Computer TrueNorth
http://www.research.ibm.com/articles/brain-chip.shtml

В настоящее время на рынке существует множество семейств нейропроцессоров:

- Siemens MA16,
- NHC 100 NAP Chip,
- Adaptive Solutions CNAPS-1064,
- Bellcore CLNN 32 CLNN 64,
- Модуль NM63, NM64,
- Intel ETANN 80170NW,
- MicroCircuit Engineering MT19003,
- Nestor NI 1000,
- Synaptics ORC 110xx,
- Philips L-Neuro 1.0, L-Neuro 2.3,
- Hitachi WSC (Wafer Scale Integration),



Программная реализация ИНС

Matlab - Neural Network Toolbox

MathWorks

Design and simulate neural networks

Neural Network Toolbox™ provides tools for designing, implementing, visualizing, and simulating neural networks.

Key Features:

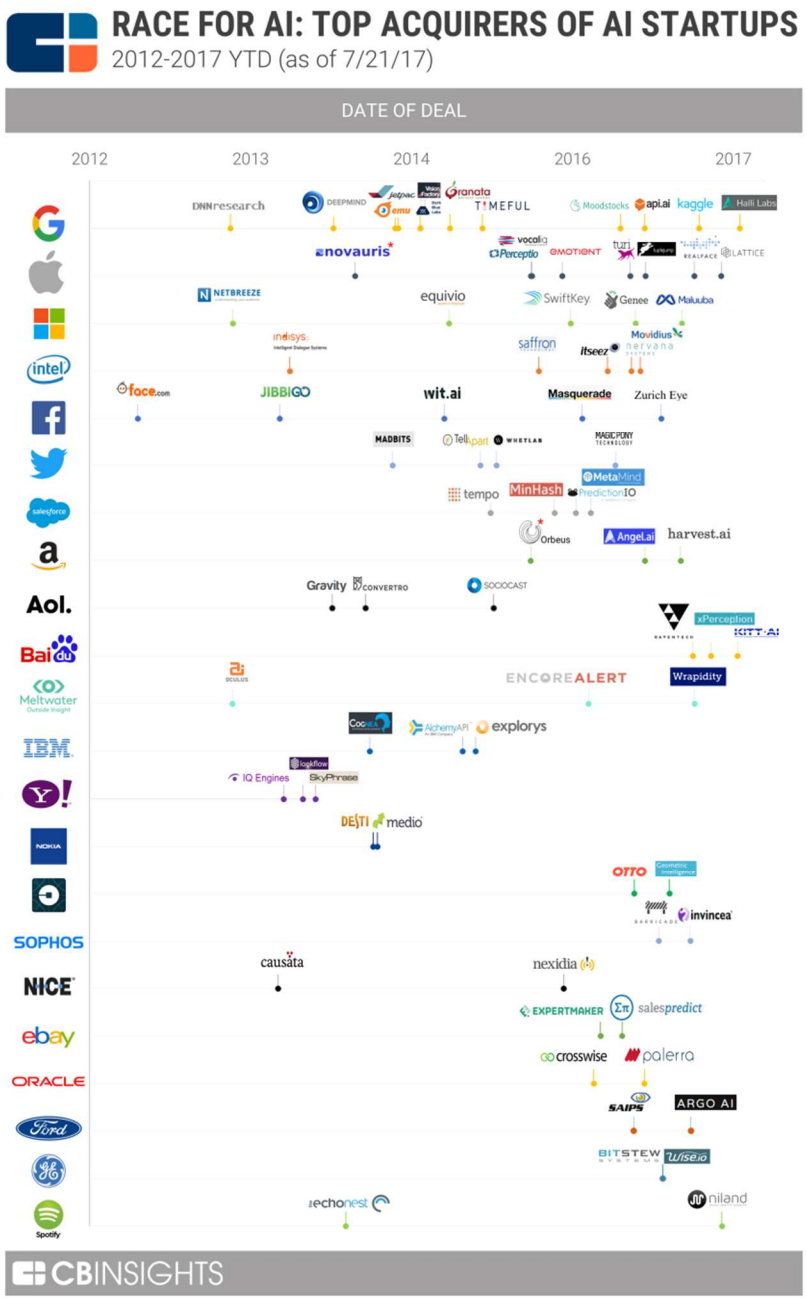
- Neural network design, training, and simulation
- Pattern recognition, clustering, and data fitting tools
- Supervised networks including feedforward, radial basis, LVQ, time delay, nonlinear autoregressive (NARX), and layer-recurrent
- Unsupervised networks including self-organizing maps and competitive layers
- Preprocessing and postprocessing for improving network training efficiency and assessing performance
- Modular network representation for managing and visualizing networks of arbitrary size
- Routines for improving generalization to prevent overfitting
- Simulink® blocks for building and evaluating neural networks, and advanced blocks for control systems applications

STATISTICA Automated Neural Networks
Автоматизированные нейронные сети

StatSoft®

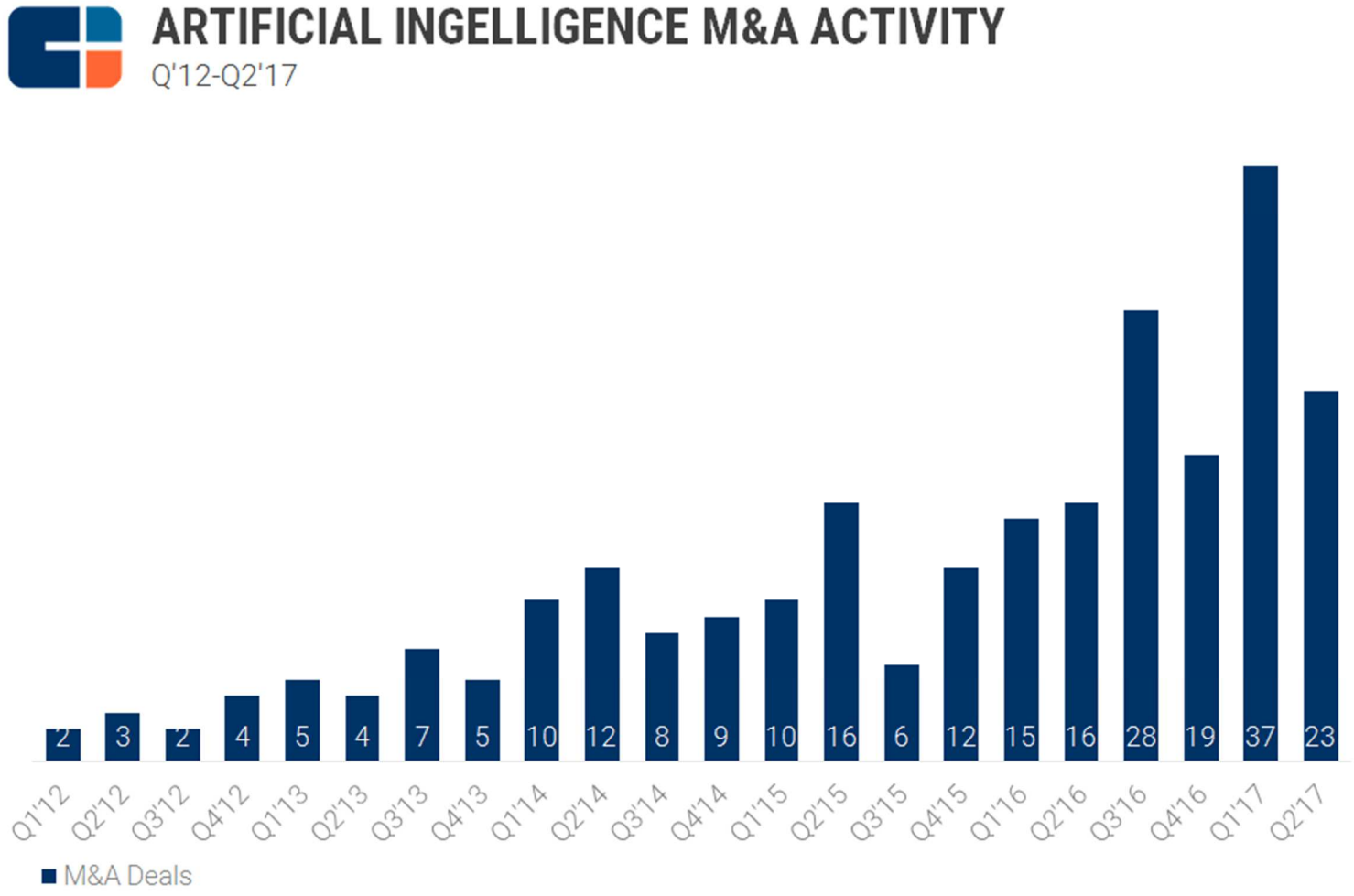
https://www.tensorflow.org/ Open Source Software Library for Machine Intelligence

Покупки AI компаний гигантами IT индустрии



https://www.cbinsights.com/blog/top-acquirers-ai-startups-ma-timeline/

Покупки AI компаний гигантами IT индустрии



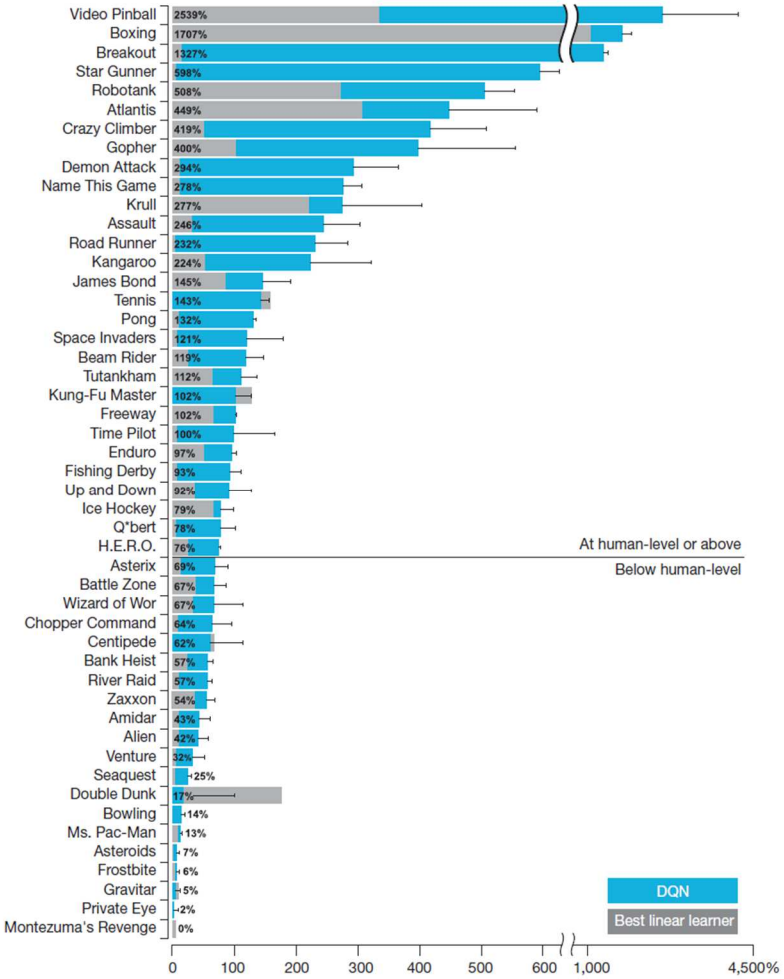
https://www.cbinsights.com/blog/top-acquirers-ai-startups-ma-timeline/

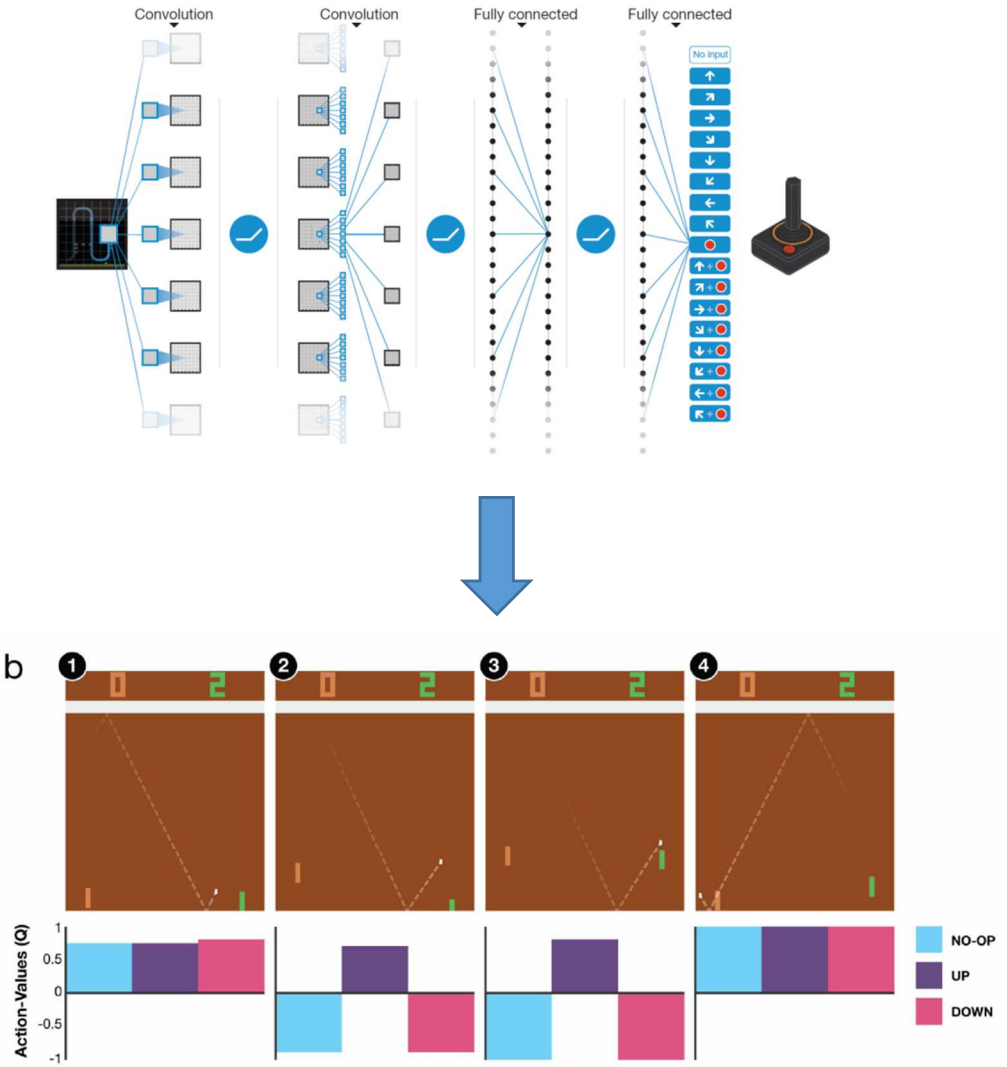
Современные примеры применения ИНС

Google Atari

Human-level control through deep reinforcement learning

Исследование возможностей нейросетей при управлении в простых играх Atari

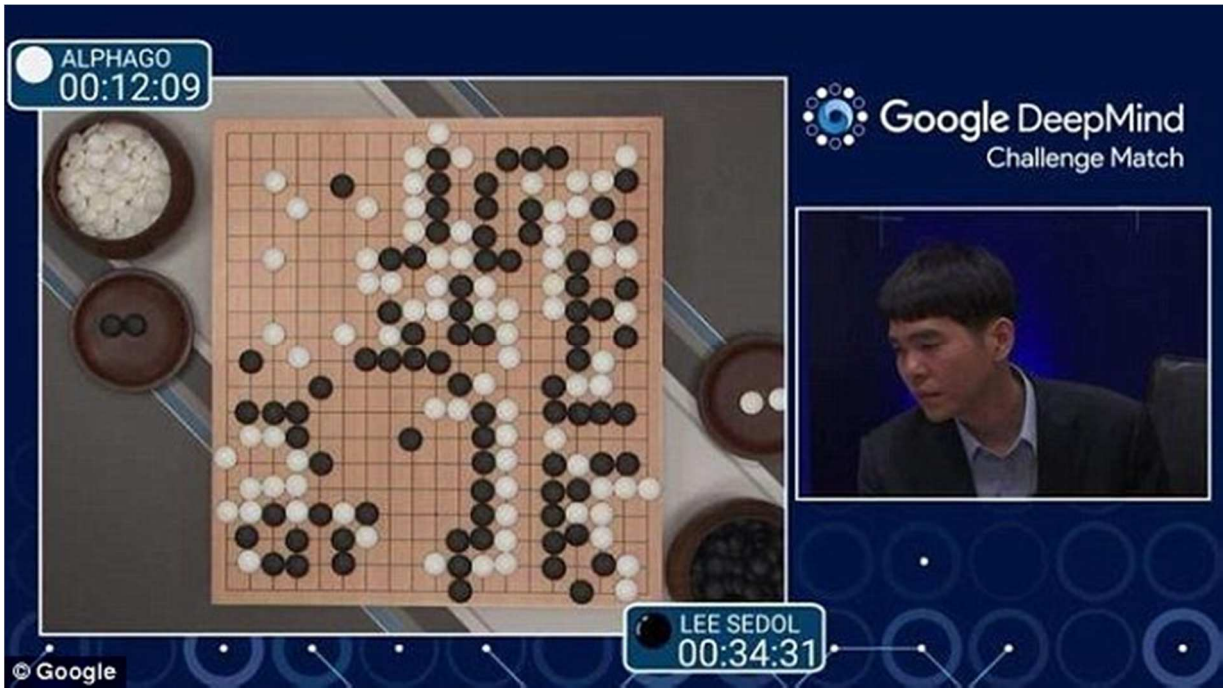




Современные примеры применения ИНС

Google AlphaGo

AlphaGo — программа для игры в го, разработанная компанией Google DeepMind в 2015 году. AlphaGo стала в 2015 году первой в мире программой, которая выиграла равный матч у профессионального игрока в го на стандартной доске 19×19. В марте 2016 года программа выиграла со счётом 4—1 у Ли Седоля, профессионала 9 дана (высшего ранга), во время исторического матча, широко освещавшегося в прессе. После победы в матче Корейская ассоциация падук присвоила AlphaGo «почётный 9 дан» (наивысший ранг в го) за «искренние усилия» программы в овладении мастерством игры.



Современные примеры применения ИНС

DeepArt IO

1 Upload photo

The first picture defines the scene you would like to have painted.

2 Choose style

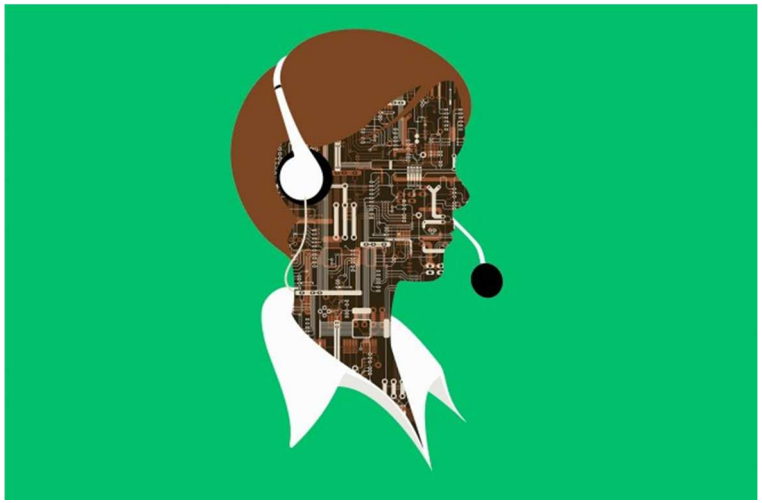
Choose among predefined styles or upload your own style image.

3 Submit

Our servers paint the image for you. You get an email when it's done.

Современные примеры применения ИНС

Google Чатбот



Итак, человек разговаривает с машиной и выглядит это следующим образом:

Человек: В чем заключается цель жизни?
Машина: служить всеобщему благу.
Человек: Какая конечная цель и смысл жизни?
Машина: жить вечно.
Человек: А цель существования?
Машина: разобраться, что происходит на планете Земля.
Человек: Где ты сейчас находишься?
Машина: я нахожусь неизвестно где.
Человек: Какова цель смерти?
Машина: иметь жизнь.
Человек: Зачем быть умным?
Машина: чтобы узнать, что это.
Человек: В чем заключается смысл эмоций?
Машина: я не знаю.

Машинное обучение (Machine Learning)

- Машинное обучение – это подраздел ИИ, включающий методы построения алгоритмов, способных обучаться.
- Машинное обучение – подраздел ИИ, математическая дисциплина, использующая разделы математической статистики, численных методов оптимизации, теории вероятностей, дискретного анализа, выделяющая знания из данных. (из Википедии)

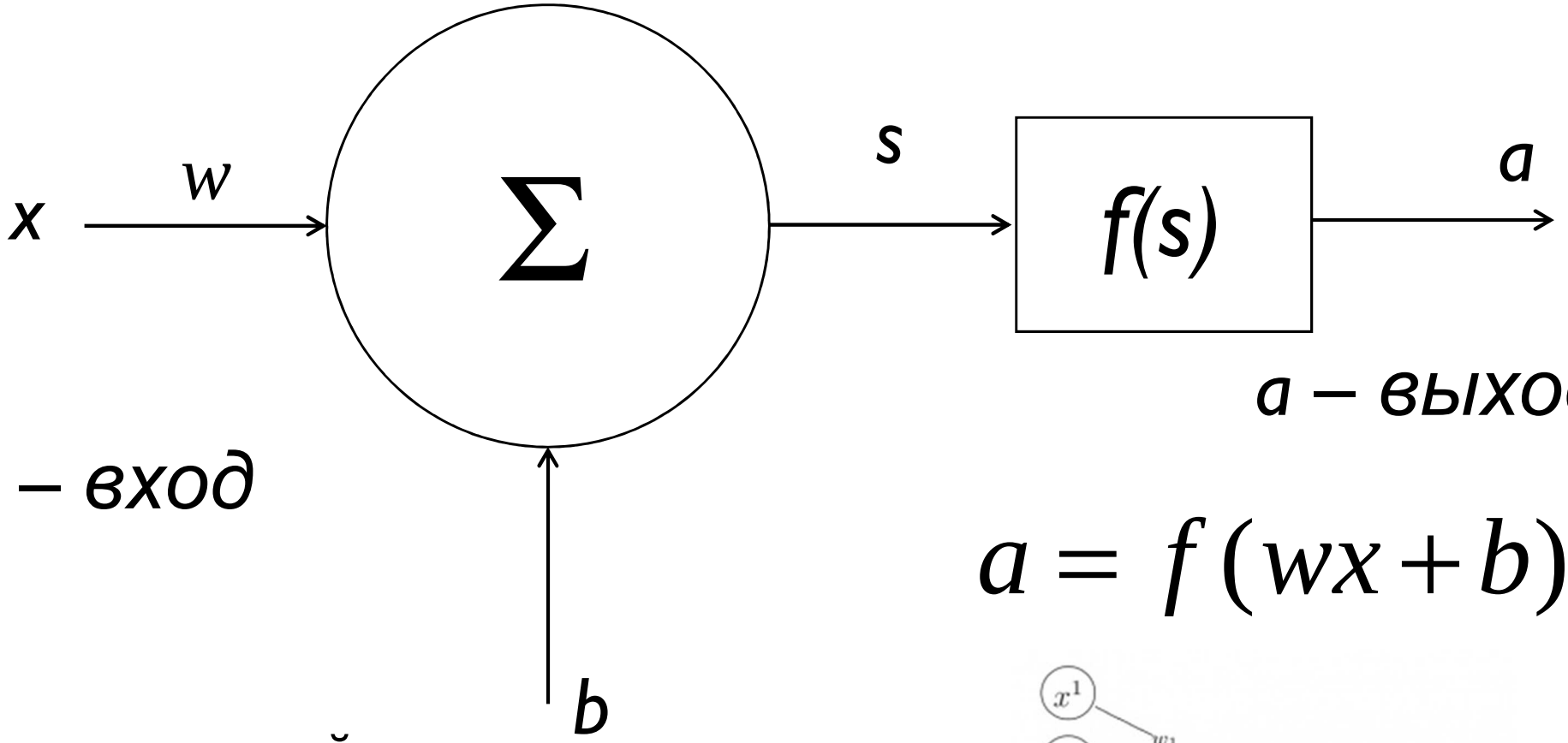
Машинное обучение изучает методы построения алгоритмов, которые могут обучаться из данных и делать прогноз на данных.

Машинное обучение (Machine Learning)

По оценке McKinsey Global Institute, в 2018 году в одних только Соединенных Штатах спрос на экспертов по машинному обучению будет превышать предложение на 140–190 тысяч человек. Кроме того, потребуется дополнительно полтора миллиона разбирающихся в данных управленцев.
(Домингос П. Верховный алгоритм: как машинное обучение изменит наш мир. М. : Манн, Иванов и Фербер, 2016.)

- Pattern Recognition (распознавание образов)
- Data Mining (интеллектуальный анализ данных, включая Big Data)
- Artificial Intelligence (искусственный интеллект)

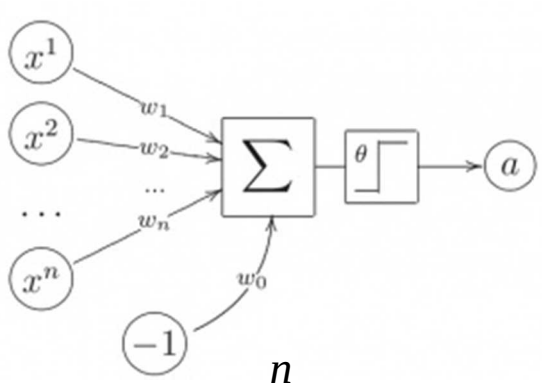
Математическая модель нейрона Маккалока-Питтса



a – выход

x – вход

$$a = f(wx + b)$$



$$a(x) = f\left(\sum_{j=1}^{n} w_j x^j - w_0\right)$$

w - веса нейрона

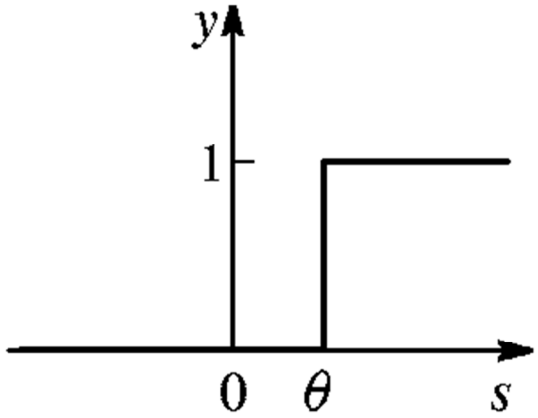
b - смещение

f - активационная функция

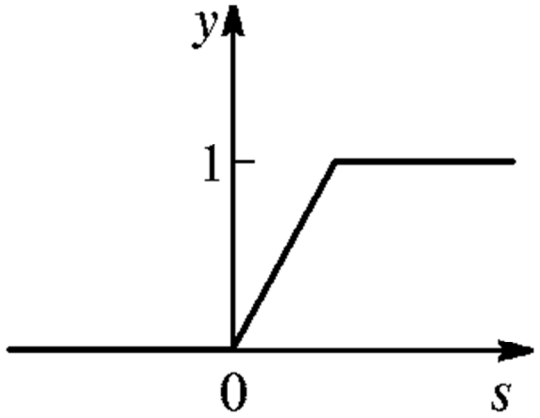
Перечень функций активации нейронов

Название	Формула	Область значений
Пороговая	$f(s) = \begin{cases} 0, & s < \theta, \\ 1, & s \geqslant \theta \end{cases}$	0, 1
Знаковая (сигнатурная)	$f(s) = \begin{cases} 1, & s > 0, \\ -1, & s \leqslant 0 \end{cases}$	-1, 1
Сигмоидальная (логистическая)	$f(s) = \dfrac{1}{1+e^{-s}}$	(0, 1)
Полулинейная	$f(s) = \begin{cases} s, & s > 0, \\ 0, & s \leqslant 0 \end{cases}$	$(0, \infty)$
Линейная	$f(s) = s$	$(-\infty, \infty)$
Радиальная базисная (гауссова)	$f(s) = e^{-s^2}$	(0, 1)
Полулинейная с насыщением	$f(s) = \begin{cases} 0, & s \leqslant 0, \\ s, & 0 < s < 1, \\ 1, & s \geqslant 1 \end{cases}$	(0, 1)
Линейная с насыщением	$f(s) = \begin{cases} -1, & s \leqslant -1, \\ s, & -1 < s < 1, \\ 1, & s \geqslant 1 \end{cases}$	$(-1, 1)$
Гиперболический тангенс (сигмоидальная)	$f(s) = \dfrac{e^{s} - e^{-s}}{e^{s} + e^{-s}}$	$(-1, 1)$
Треугольная	$f(s) = \begin{cases} 1 - \lvert s \rvert, & \lvert s \rvert \leqslant 1, \\ 0, & \lvert s \rvert > 1 \end{cases}$	(0, 1)

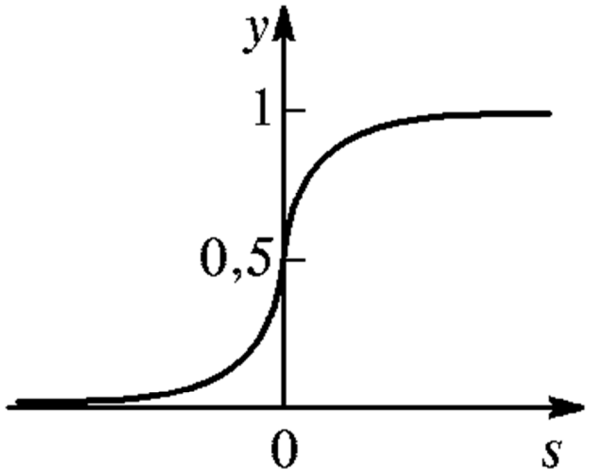
Примеры активационных функций:



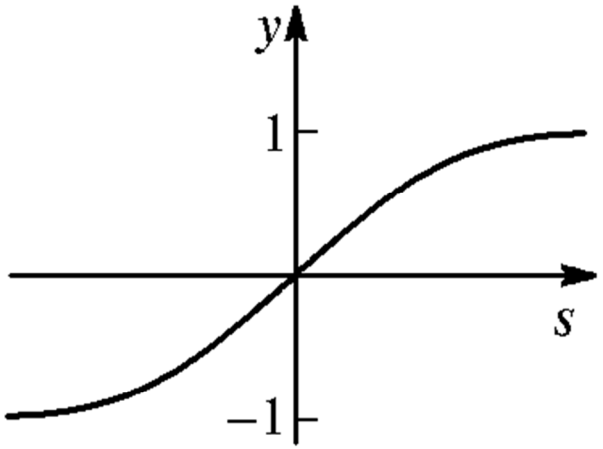
функция единичного скачка;



линейный порог (гистерезис);

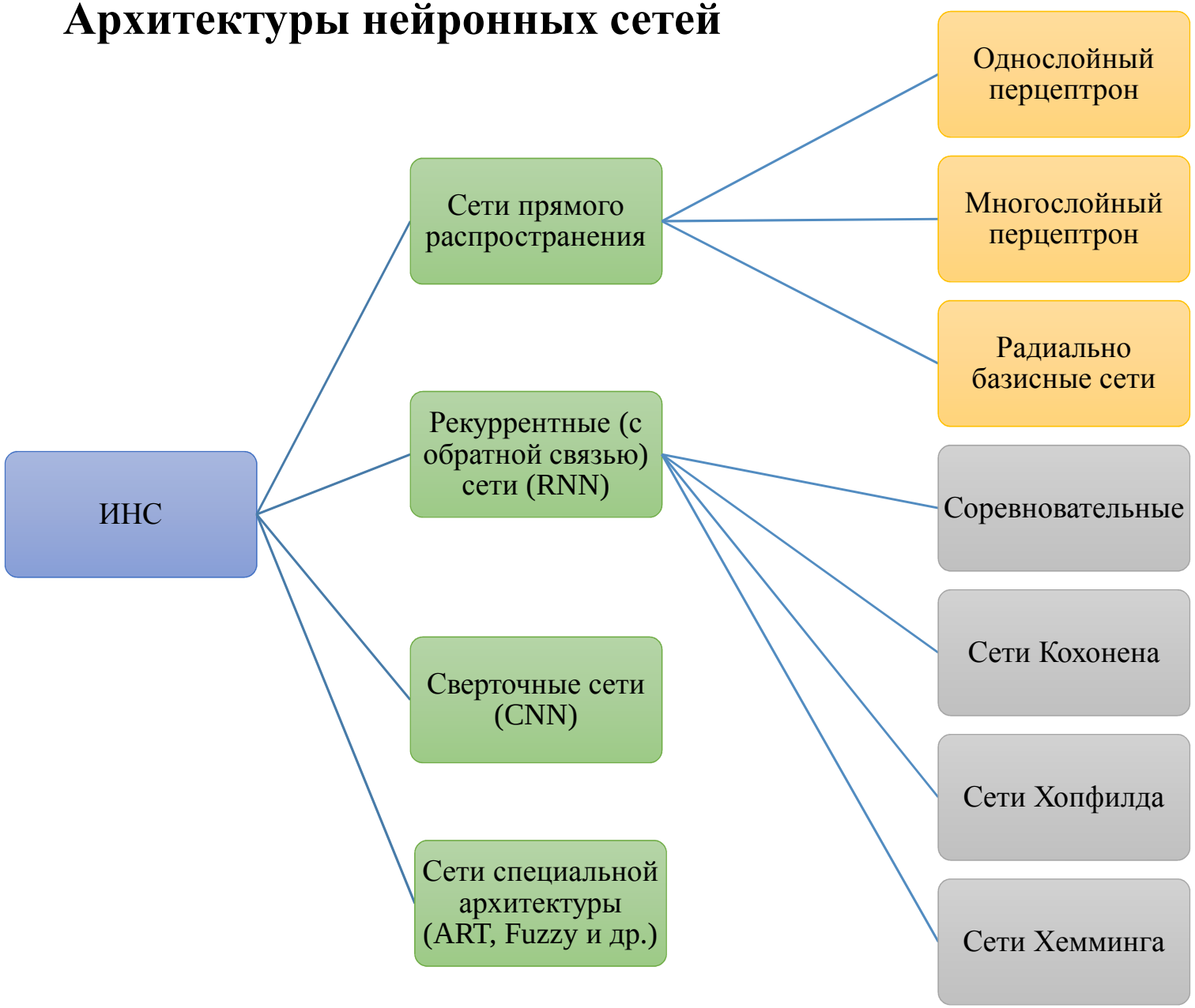


сигмоид (гиперболический тангенс);

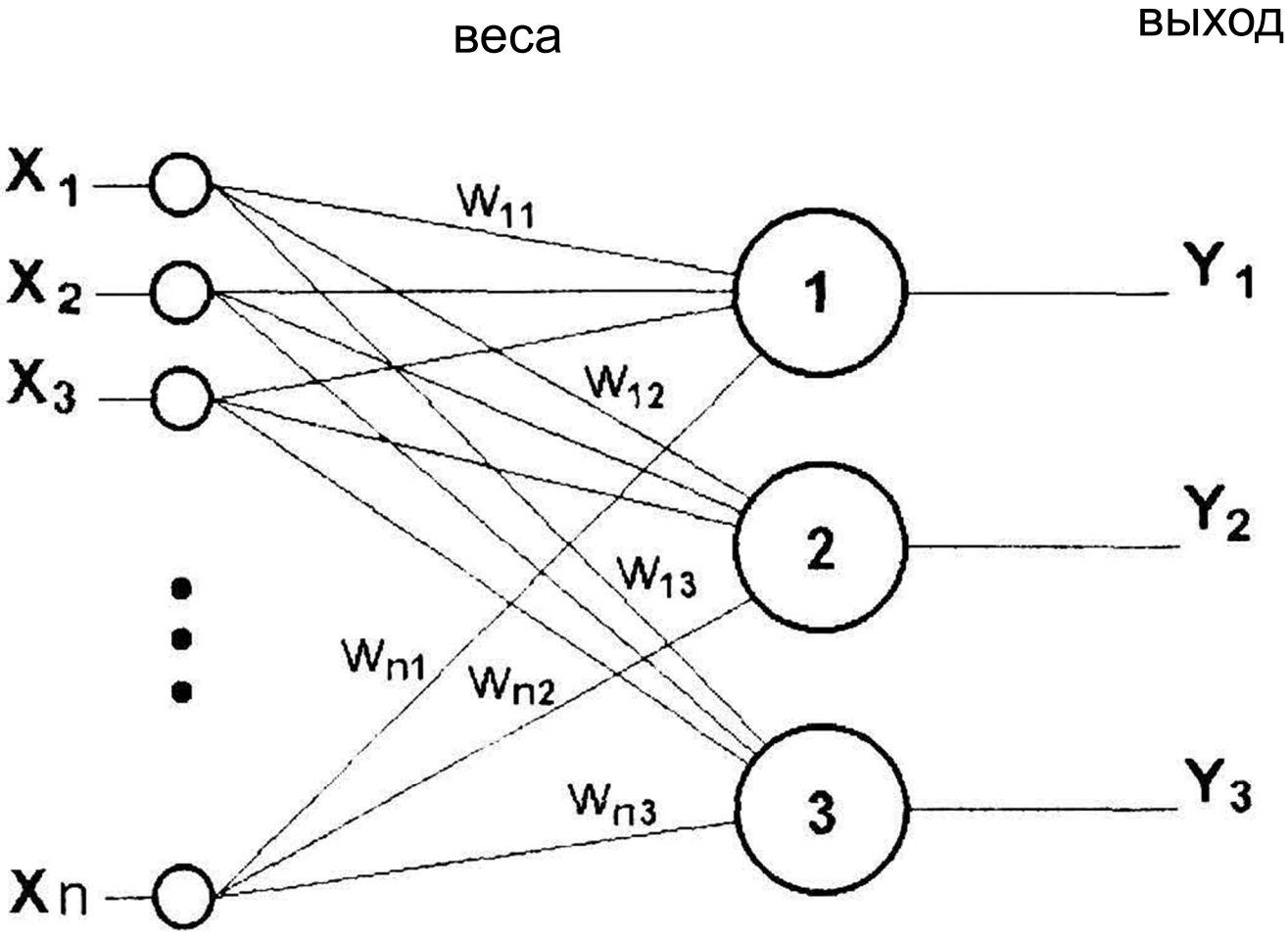


сигмоид (логистическая)

Архитектуры нейронных сетей



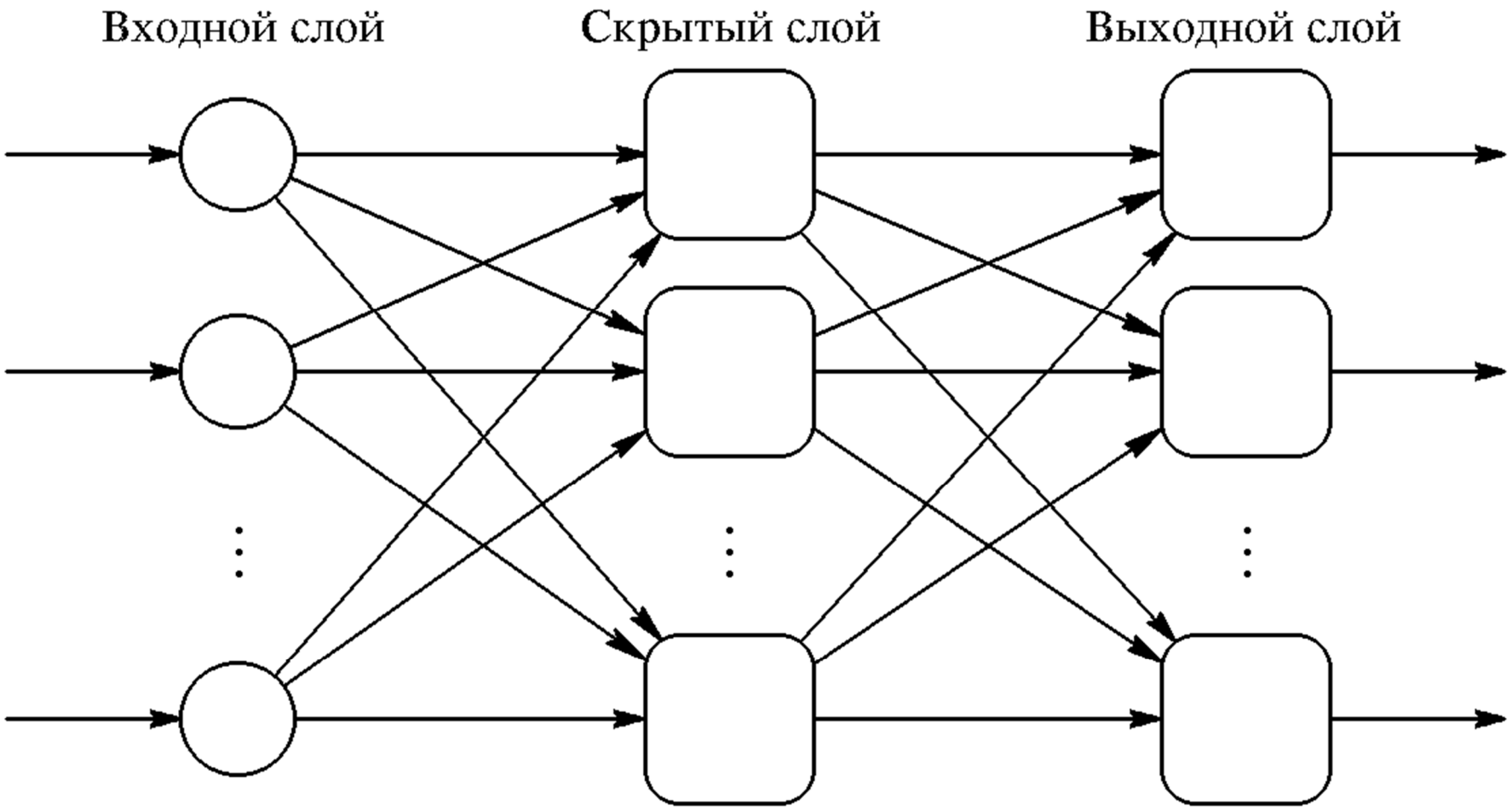
Примеры архитектур нейронных сетей



Однослойная нейронная сеть прямого распространения

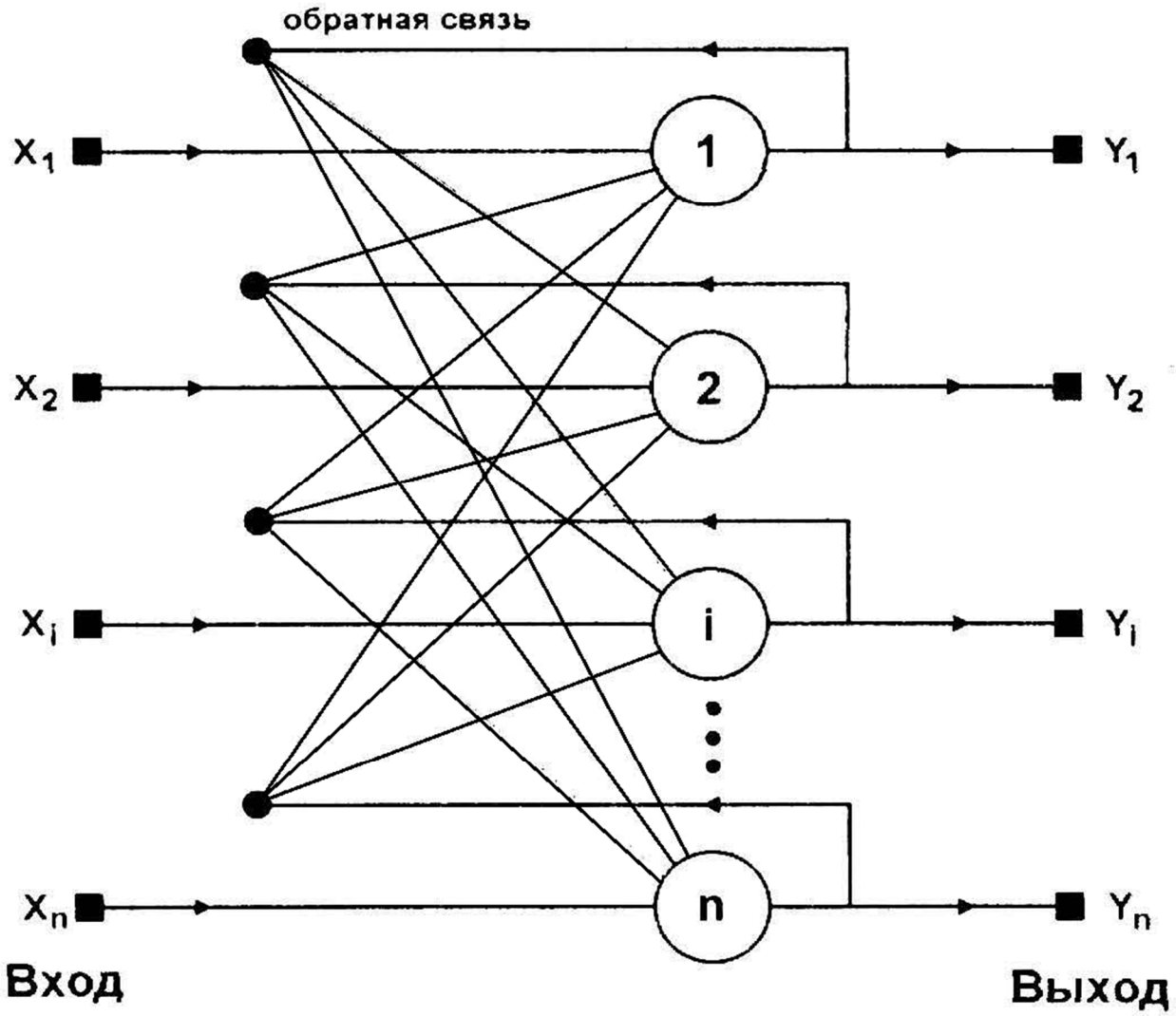
$$Y = f(Wx)$$

Многослойная нейронная сеть прямого распространения

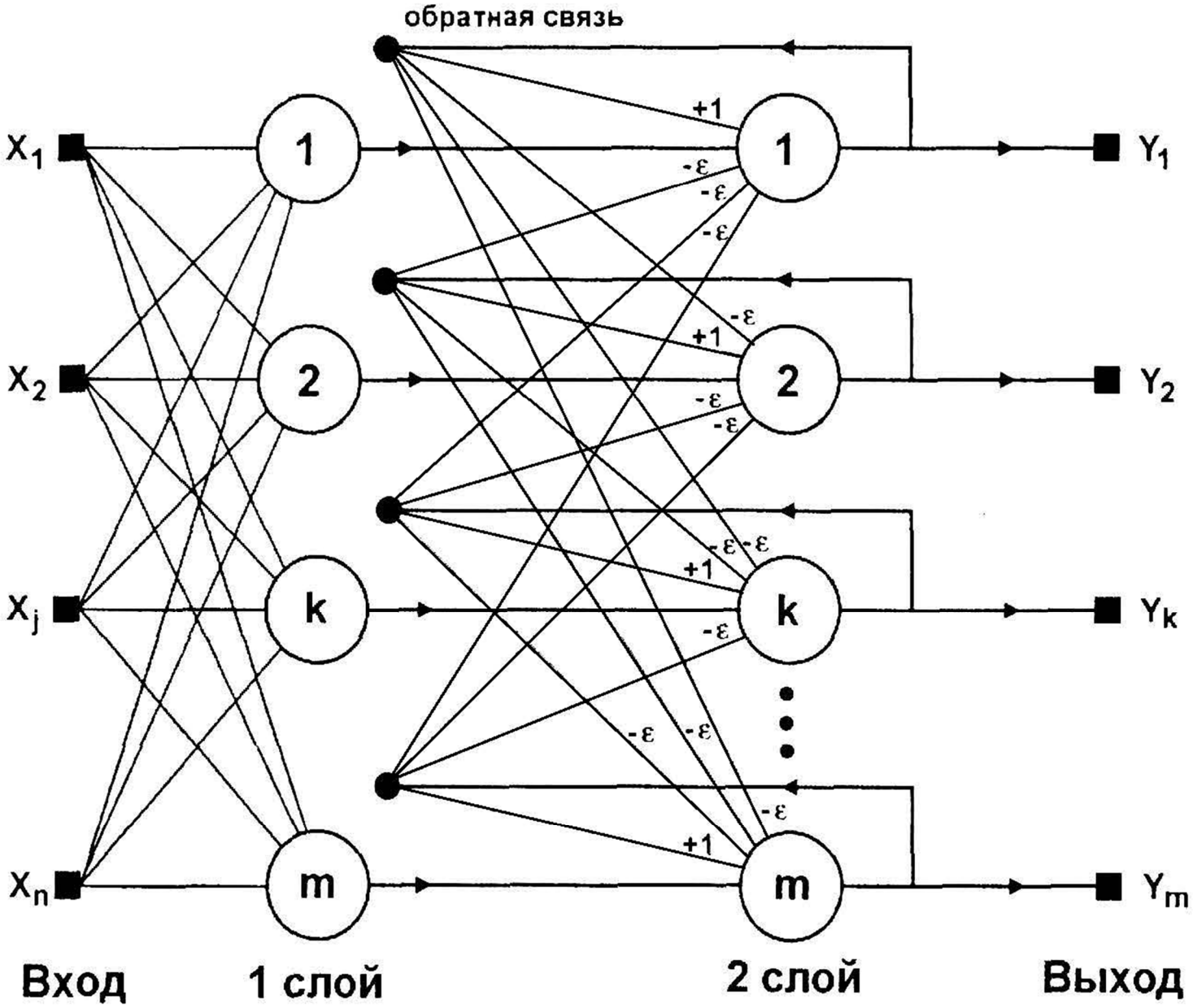


$$\mathbf{Y = f^{2}(W_{2}f^{1}(W_{1}x))}$$

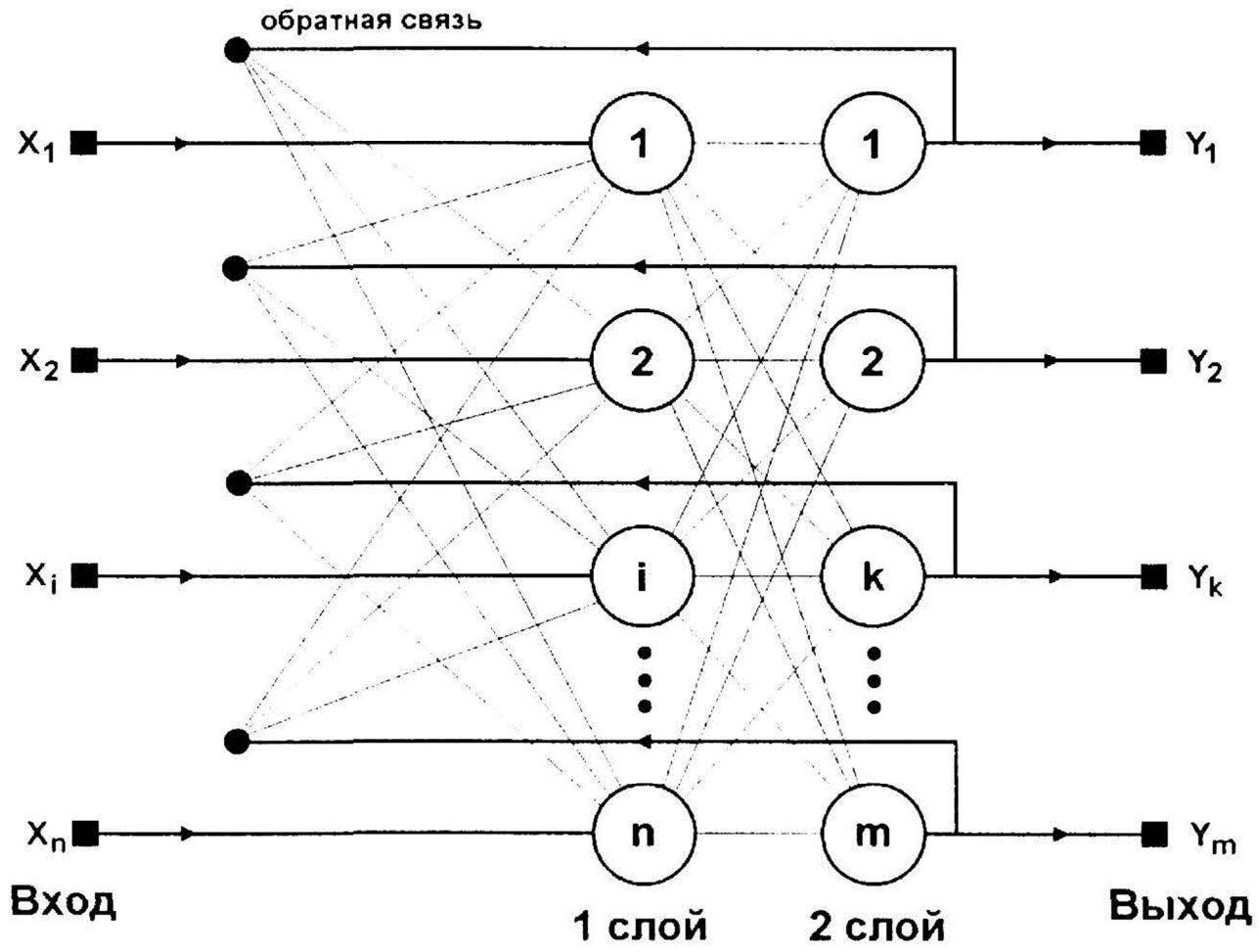
Сеть Хопфилда



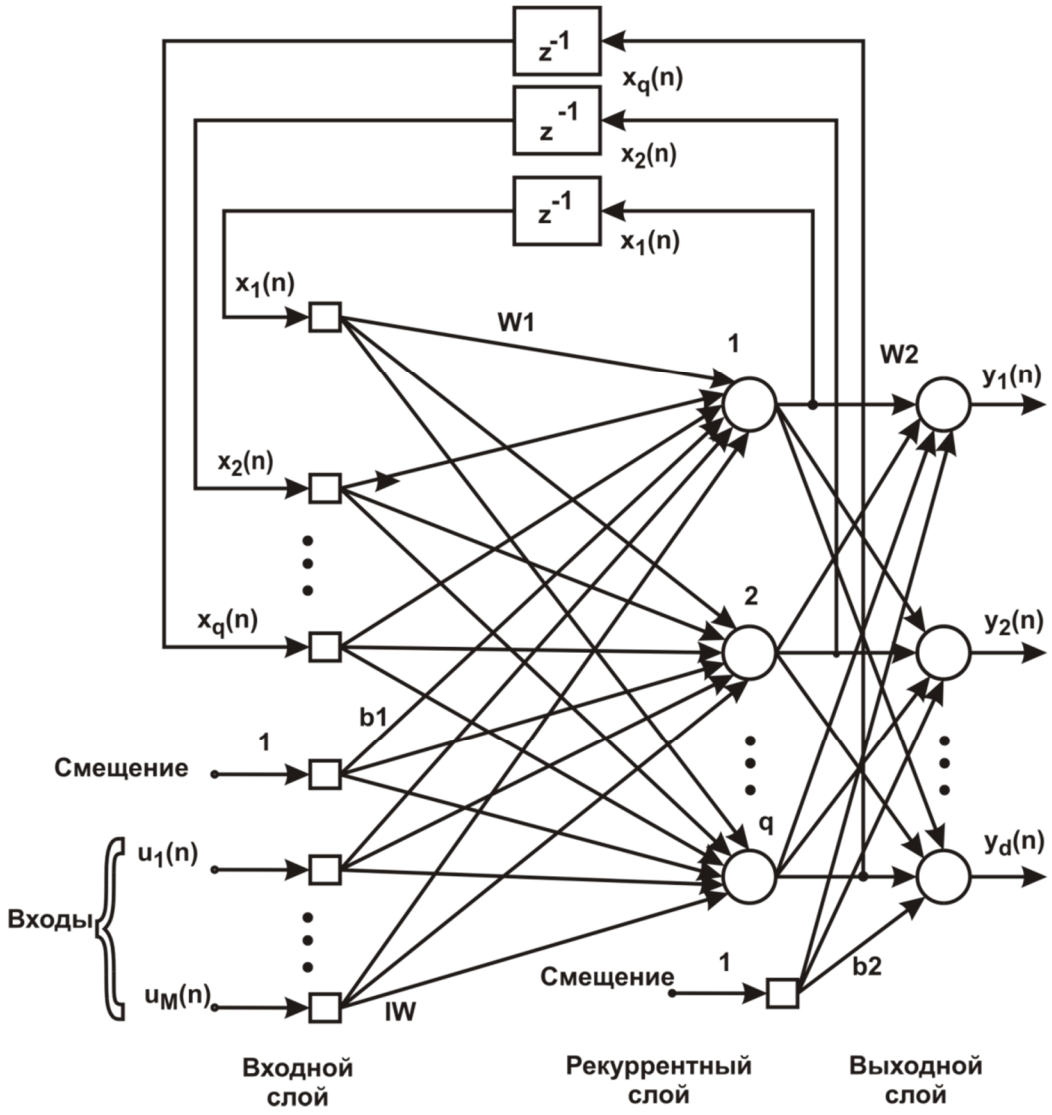
Сеть Хэмминга



Двунаправленная ассоциативная память



Рекуррентная сеть со скрытыми нейронами



Применение ИНС

- **Автоматизация производства:** оптимизация режимов производственного процесса, контроль качества продукции, мониторинг и визуализация многомерной диспетчерской информации, предупреждение аварийных ситуаций.
- **Робототехника:** распознавание сцены, объектов и препятствий перед роботом, прокладка маршрута движения, управление манипуляторами, поддержание равновесия.
- **Медицина:** постановка диагноза больному, обработка медицинских изображений, мониторинг состояния пациента, анализ эффективности лечения, очистка показаний приборов от шумов.
- **Авионика:** обучаемые автопилоты, распознавание сигналов радаров, адаптивное пилотирование сильно поврежденного самолета, беспилотные летательные аппараты.
- **Связь:** сжатие видеоинформации, быстрое кодирование-декодирование, оптимизация сотовых сетей и схем маршрутизации пакетов.
- **Экономика и бизнес:** прогнозирование временных рядов (курсов валют, цен на сырьё, объемов продаж,..), автоматический трейдинг (торговля на валютной, фондовой или товарной бирже), оценка рисков невозврата кредитов, предсказание банкротств, оценка стоимости недвижимости, выявление переоцененных и недооцененных компаний, рейтингование, оптимизация товарных и денежных потоков, считывание и распознавание чеков и документов, безопасность транзакций по пластиковым картам.
- **Интернет:** ассоциативный поиск информации, электронные секретари и автономные агенты в интернете, фильтрация и блокировка спама, автоматическая рубрикация сообщений из новостевых лент, адресные реклама и маркетинг для электронной торговли, распознавание captcha.

Применение ИНС

- **Политологические и социологические технологии:** предсказание результатов выборов, анализ опросов, предсказание динамики рейтингов, выявление значимых факторов, кластеризация электората, исследование и визуализация социальной динамики населения.
- **Безопасность и охранные системы:** распознавание лиц; идентификация личности по отпечаткам пальцев, голосу, подписи или лицу; распознавание автомобильных номеров, анализ аэрокосмических снимков, мониторинг информационных потоков в компьютерной сети и обнаружение вторжений, обнаружение подделок, анализ данных с видеодатчиков и разнообразных сенсоров.
- **Ввод и обработка информации:** распознавание рукописных текстов, отсканированных почтовых, платежных, финансовых и бухгалтерских документов.
- **Геологоразведка:** анализ сейсмических данных, ассоциативные методики поиска полезных ископаемых, оценка ресурсов месторождений.

Свойства

- Обучаемость (самообучаемость)
- Обобщение
- Абстрагирование
- Память

Преимущества и недостатки ИНС

+ Универсальность
+ Гибкость
+ Способность решения задач для объектов без их математичекого описания
+ Устойчивость к шумам входных данных
+ Потенциально сверхвысокое быстродействие
+ Отказоустойчивость

- Высокая «стоимость» обучения
- Требует экспертизы в обучении
- Для хороших результатов требуется большой массив данных обучения

Обучение ИНС

Задача обучения заключается в определении таких архитектуры и параметров нейросети (синаптичеких весов), что ошибка обучения E будет минимальна для всех возможных обучающих выборок (Xk,Yk)

Алгоритмы обучения ИНС

Если выбраны множество обучающих примеров — пар (Xk , Yk) (где k = 1,2,..., N) и способ вычисления функции ошибки *E,* то обучение нейронной сети превращается в задачу многомерной оптимизации, имеющую очень большую размерность, при этом, поскольку функция *E* может иметь произвольный вид, обучение в общем случае — многоэкстремальная невыпуклая задача оптимизации.

Для решения задачи обучения могут быть использованы следующие (итерационные) алгоритмы:

- алгоритмы локальной оптимизации с вычислением частных производных первого порядка;
- алгоритмы локальной оптимизации с вычислением частных производных первого и второго порядка;
- стохастические алгоритмы оптимизации;
- алгоритмы глобальной оптимизации.

Алгоритмы обучения ИНС

К первой группе относятся:

- градиентный алгоритм (метод скорейшего спуска);
- методы с одномерной и двумерной оптимизацией целевой функции в направлении антиградиента;
- метод сопряженных градиентов;
- методы, учитывающие направление антиградиента на нескольких шагах алгоритма.

Ко второй группе относятся:

- метод Ньютона,
- методы оптимизации с разреженными матрицами Гессе,
- квазиньютоновские методы,
- метод Гаусса-Ньютона,
- метод Левенберга-Марквардта и др.

Стохастическими методами являются:

- поиск в случайном направлении,
- имитация отжига,
- метод Монте-Карло (численный метод статистических испытаний).

Задачи глобальной оптимизации решаются с помощью перебора значений переменных, от которых зависит целевая функция (функция ошибки *E*).

Алгоритм обратного распространения ошибки (back propagation error)

Алгоритм обратного распространения ошибки - это итеративный градиентный алгоритм обучения, который используется с целью минимизации среднеквадратичного отклонения текущего выхода и желаемого выхода многослойных нейронных сетей. Алгоритм обратного распространения используется для обучения многослойных нейронных сетей с последовательными связями.

Приведем словесное описание алгоритма:

Шаг 1. Весам сети присваиваются небольшие начальные значения.

Шаг 2. Выбирается очередная обучающая пара (X,V) из обучающего множества; вектор X подается на вход сети.

Шаг 3. Вычисляется выход сети.

Шаг 4. Вычисляется разность между требуемым (желаемым) и реальным (вычисленным) выходом сети.

Шаг 5. Веса сети корректируются так, чтобы минимизировать ошибку.

Шаг 6. Шаги со 2-го по 5-й повторяются для каждой пары обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемой величины.

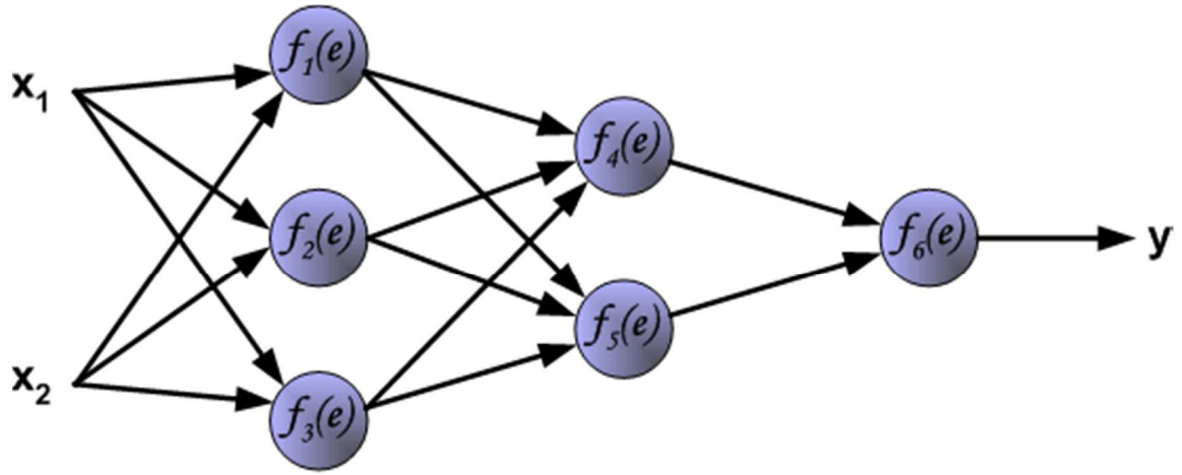
Алгоритм обратного распространения ошибки (back propagation error)

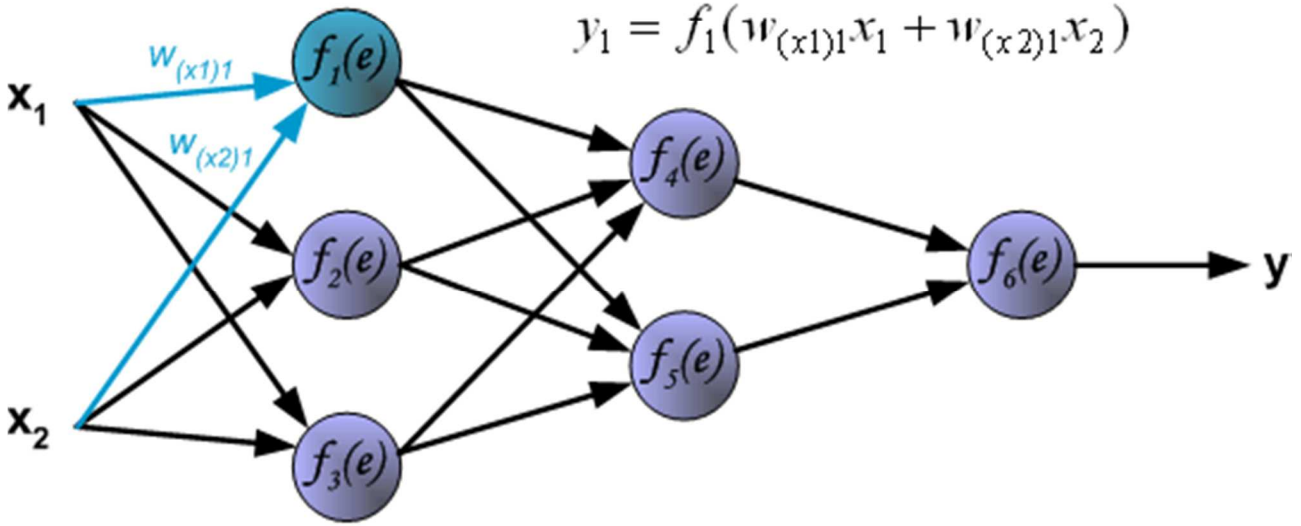
Шаги 2 и 3 подобны тем, которые выполняются в уже обученной сети. Вычисления в сети выполняются послойно. На шаге 3 каждый из выходов сети вычитается из соответствующей компоненты целевого вектора с целью получения ошибки. Эта ошибка используется на шаге 5 для коррекции весов сети.

Шаги 2 и 3 можно рассматривать как «проход вперед», так как сигнал распространяется по сети от входа к выходу. Шаги 4 и 5 составляют «обратный проход», поскольку здесь вычисляемый сигнал ошибки распространяется обратно по сети и используется для подстройки весов.

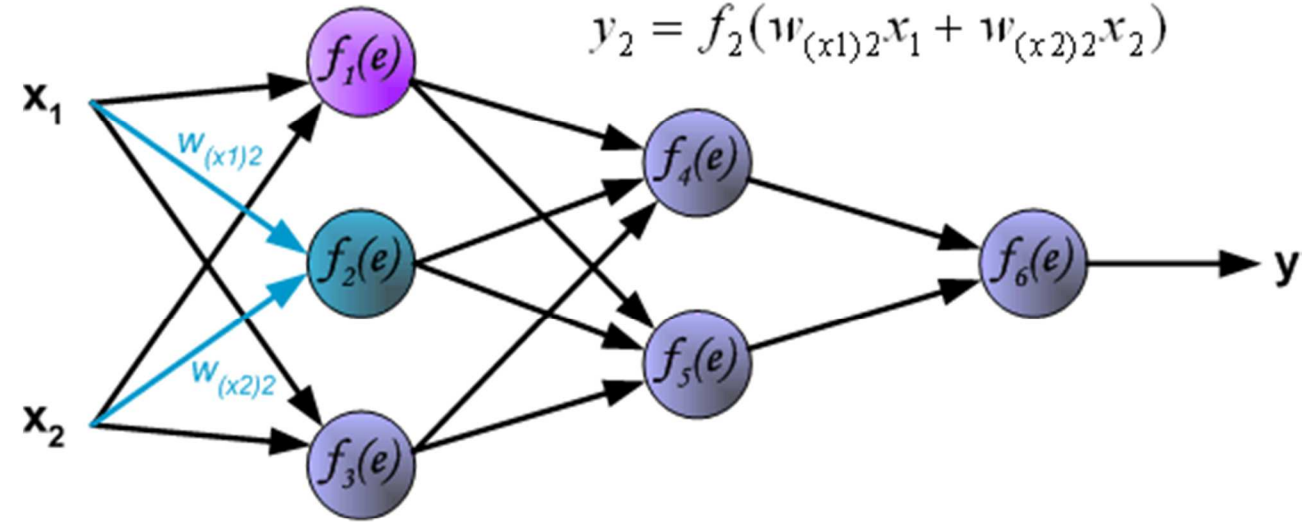
Пример: Обучение сети прямого распространения Алгоритмом обратного распространения ошибки

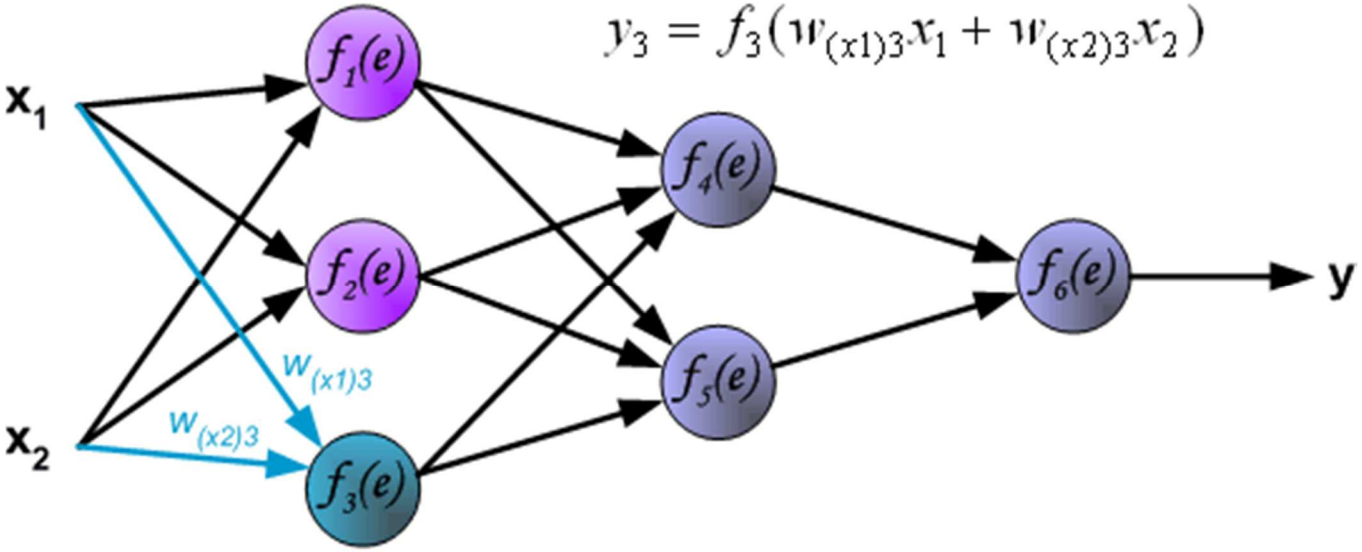
http://galaxy.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html



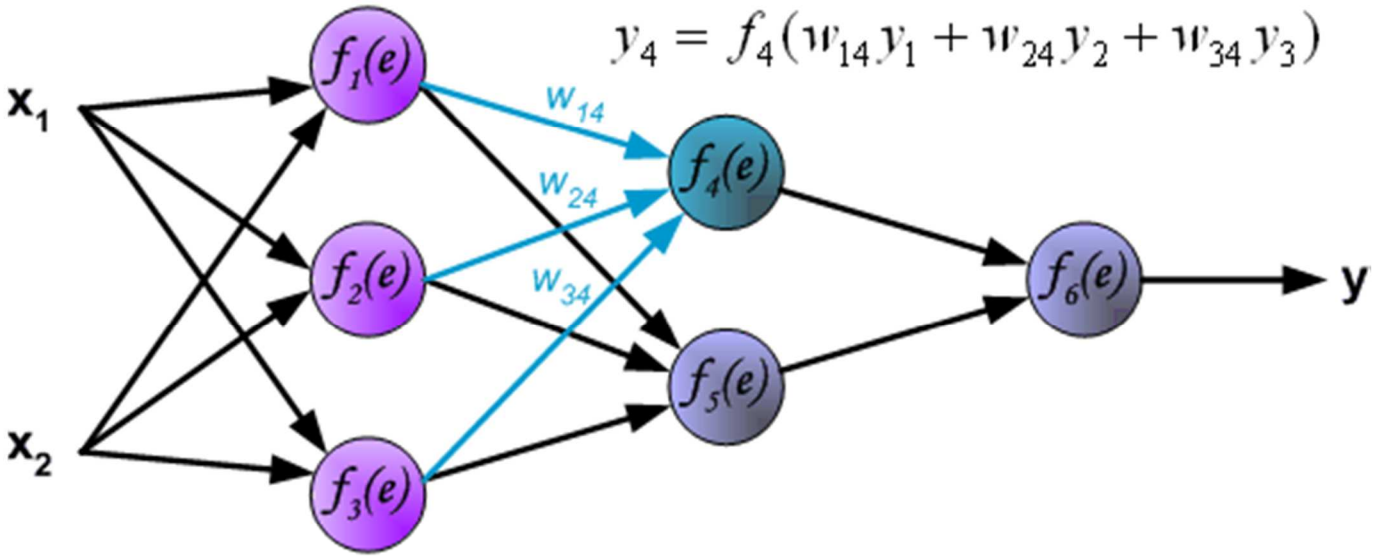


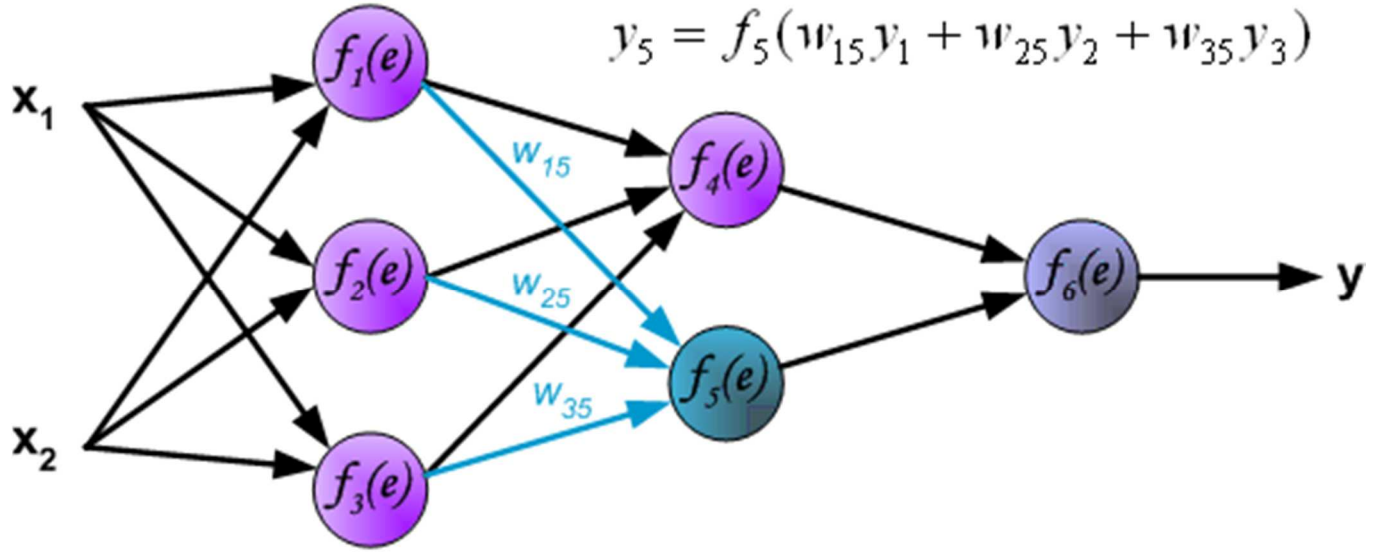
Вычисление выхода сети



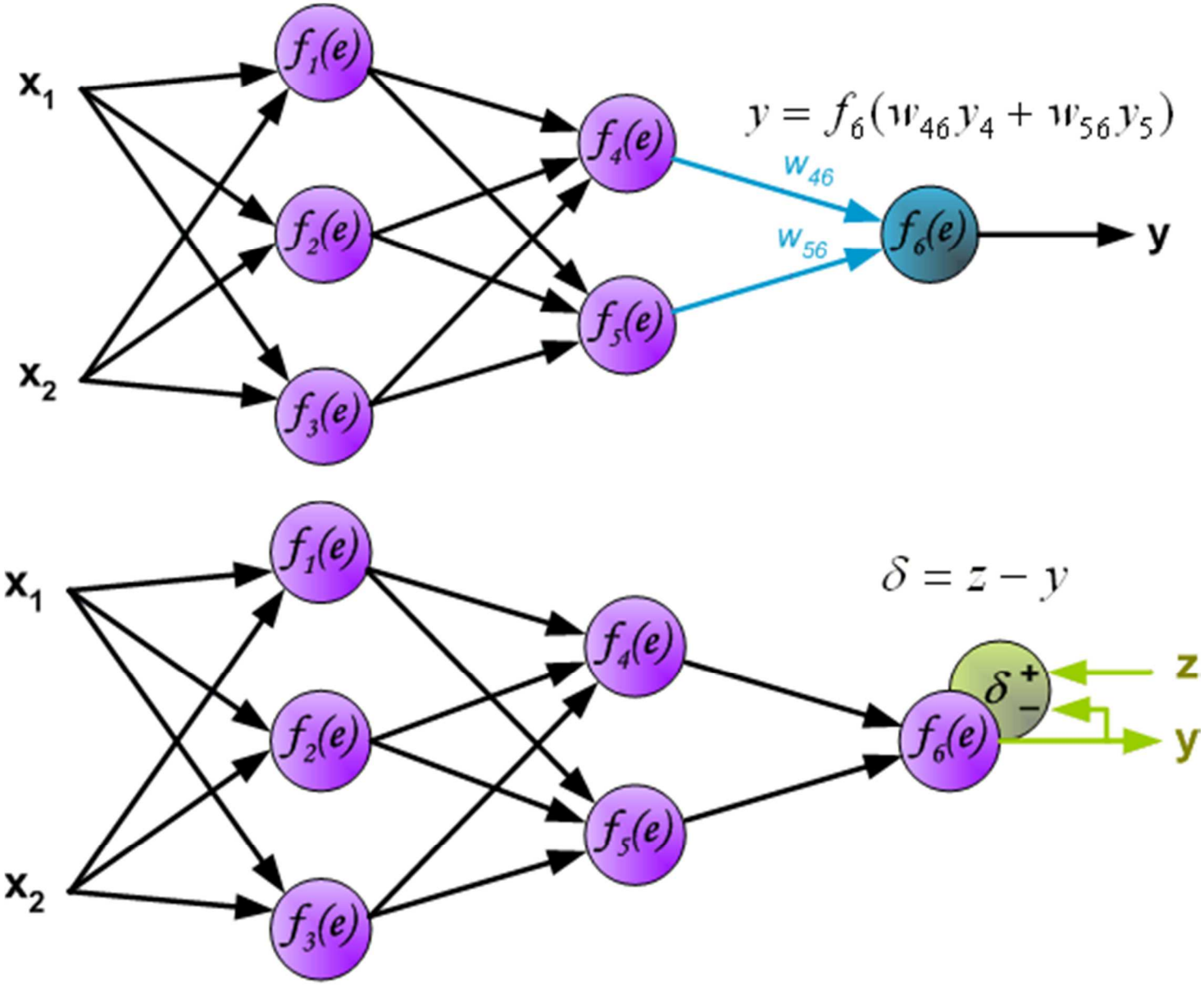


Вычисление выхода сети

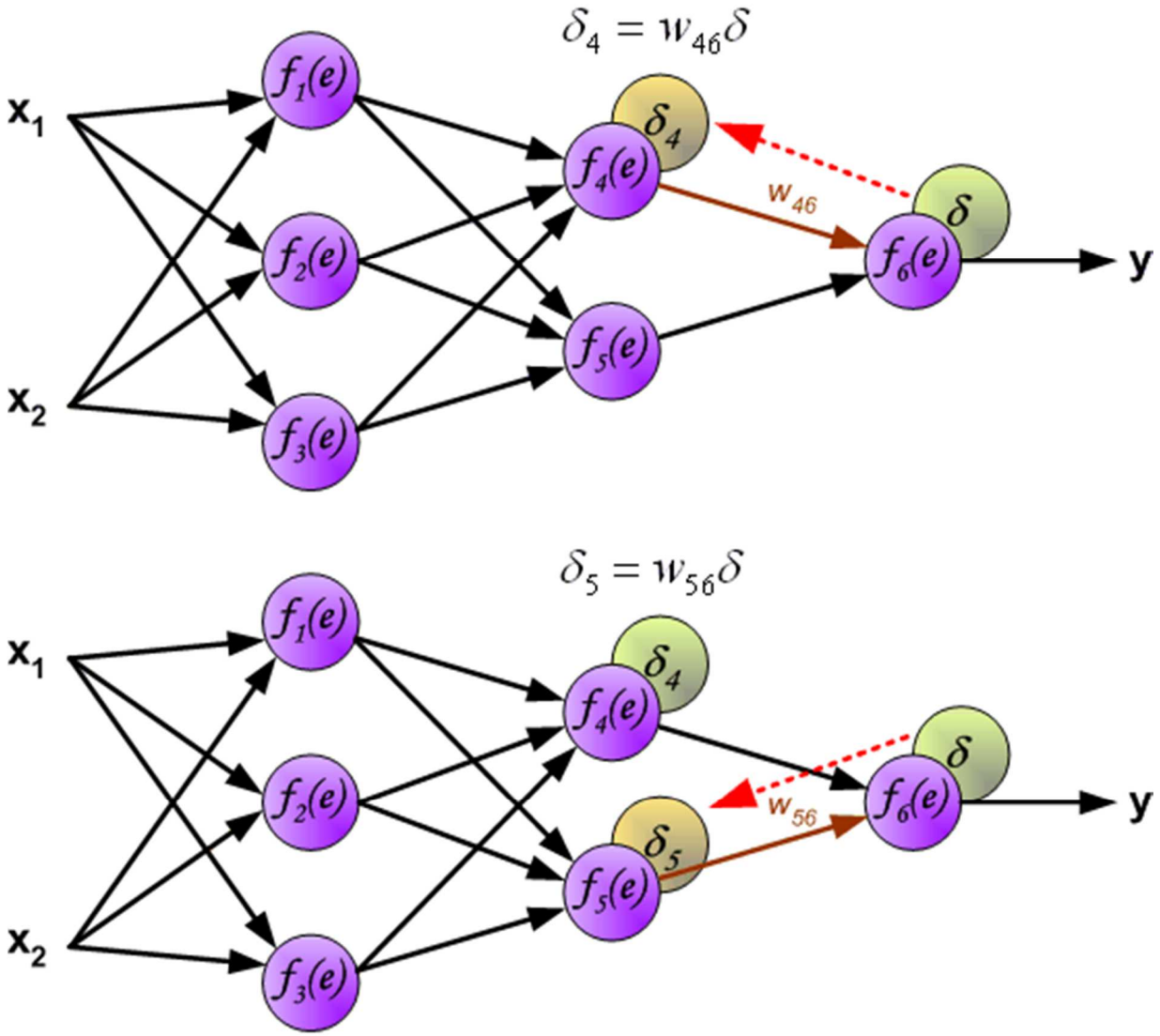




Вычисление выхода сети и ошибки обучения

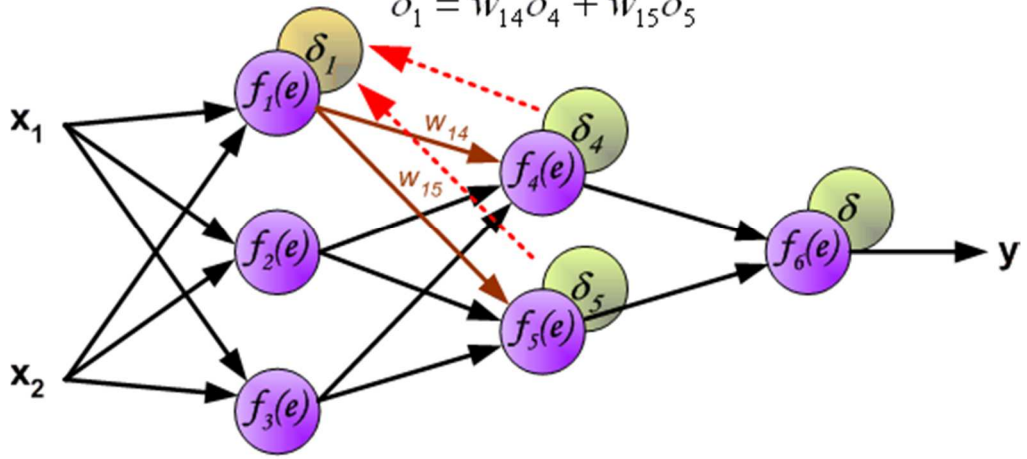


Вычисление ошибки обучения в скрытом слое

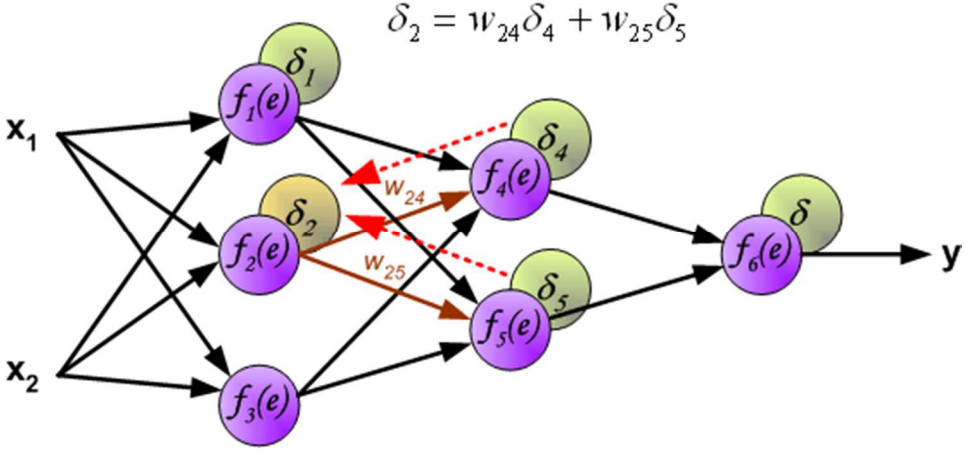


Вычисление ошибки обучения во входном слое

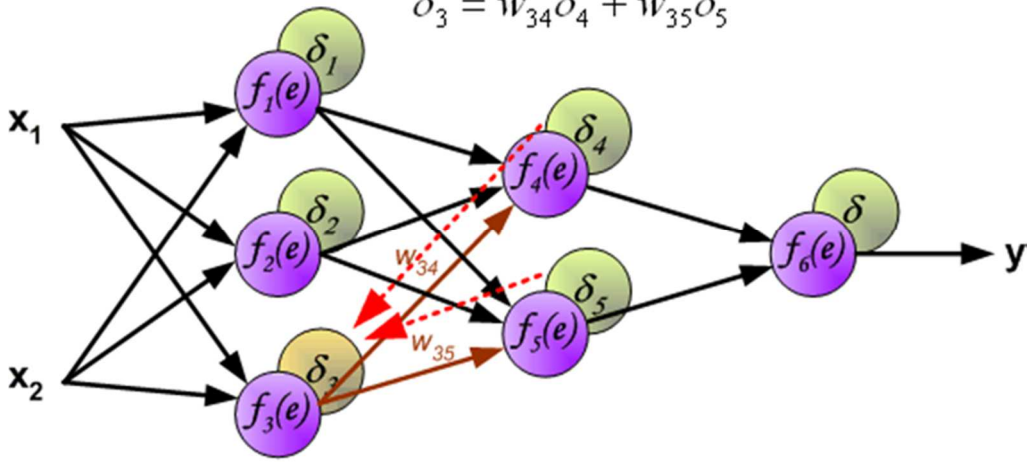
$$\delta_1 = w_{14}\delta_4 + w_{15}\delta_5$$



$$\delta_2 = w_{24}\delta_4 + w_{25}\delta_5$$



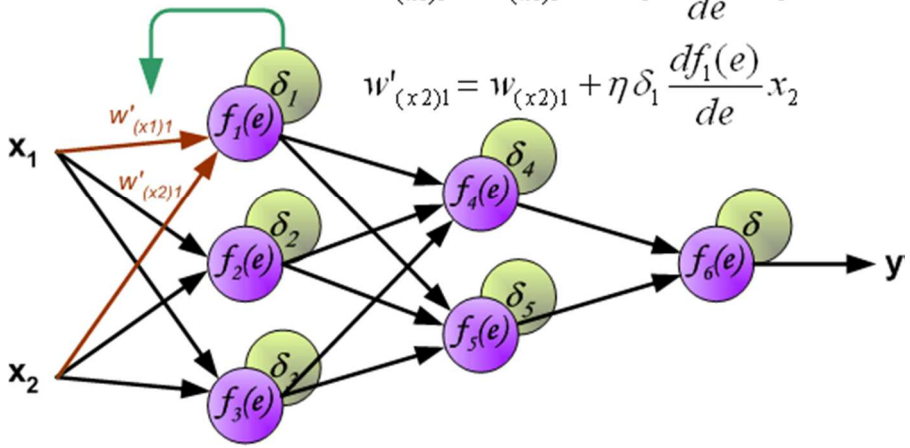
$$\delta_3 = w_{34}\delta_4 + w_{35}\delta_5$$



Коррекция весов

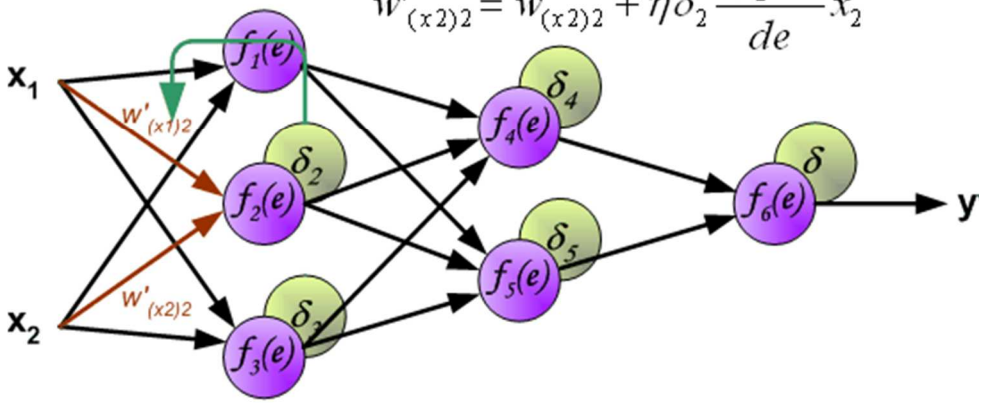
$$w'_{(x1)1} = w_{(x1)1} + \eta\delta_1 \frac{df_1(e)}{de} x_1$$

$$w'_{(x2)1} = w_{(x2)1} + \eta\delta_1 \frac{df_1(e)}{de} x_2$$



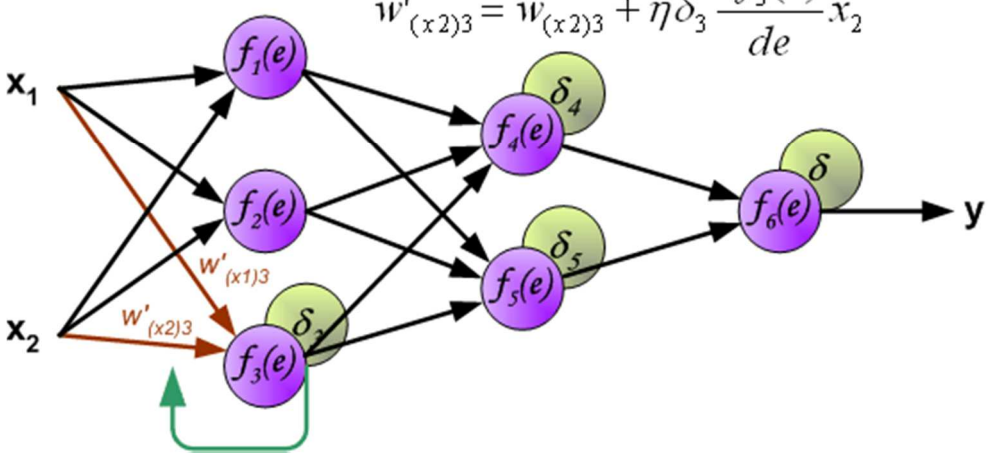
$$w'_{(x1)2} = w_{(x1)2} + \eta\delta_2 \frac{df_2(e)}{de} x_1$$

$$w'_{(x2)2} = w_{(x2)2} + \eta\delta_2 \frac{df_2(e)}{de} x_2$$



$$w'_{(x1)3} = w_{(x1)3} + \eta\delta_3 \frac{df_3(e)}{de} x_1$$

$$w'_{(x2)3} = w_{(x2)3} + \eta\delta_3 \frac{df_3(e)}{de} x_2$$

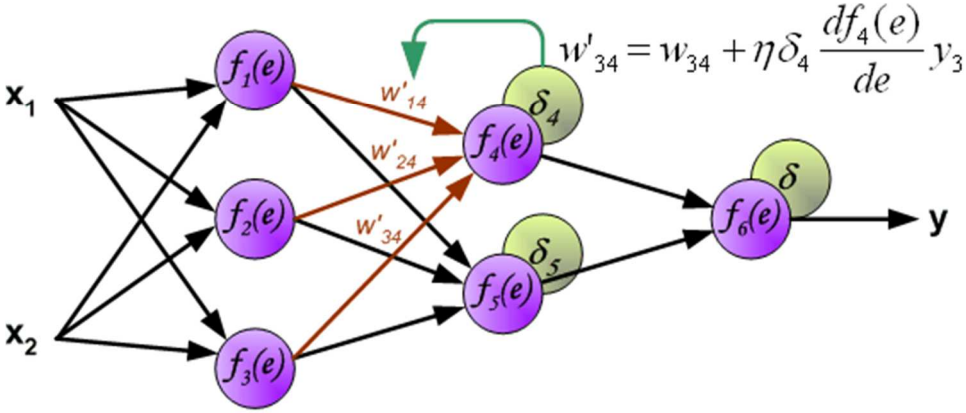


Коррекция весов

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

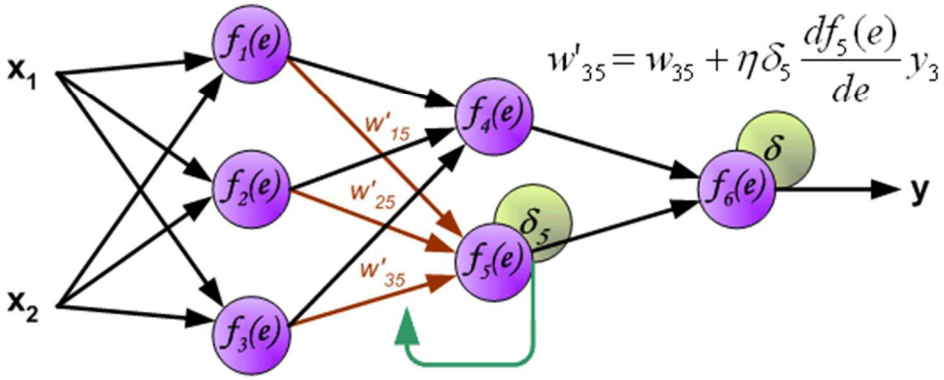
$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



$$w'_{15} = w_{15} + \eta\delta_5 \frac{df_5(e)}{de} y_1$$

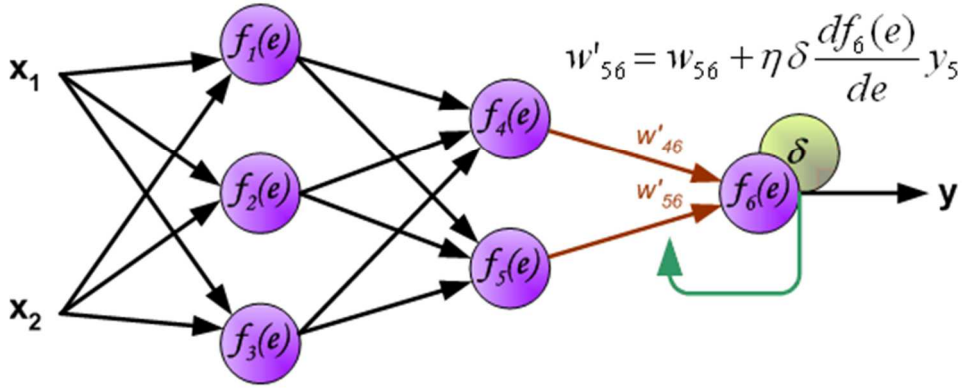
$$w'_{25} = w_{25} + \eta\delta_5 \frac{df_5(e)}{de} y_2$$

$$w'_{35} = w_{35} + \eta\delta_5 \frac{df_5(e)}{de} y_3$$

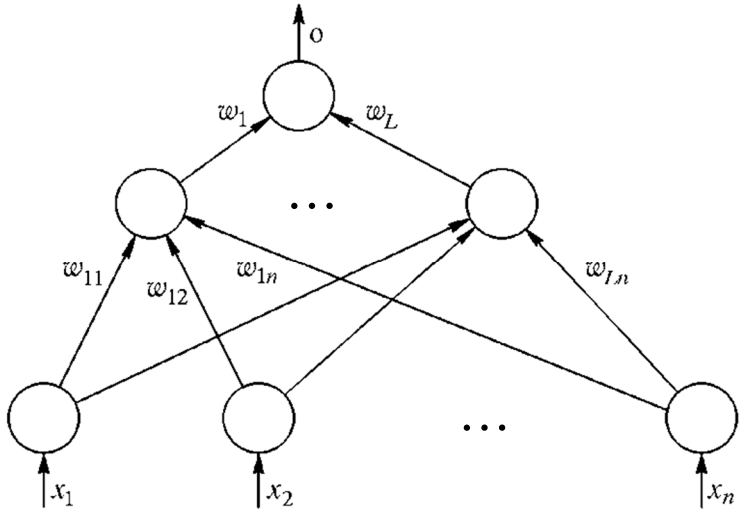


$$w'_{46} = w_{46} + \eta\delta \frac{df_6(e)}{de} y_4$$

$$w'_{56} = w_{56} + \eta\delta \frac{df_6(e)}{de} y_5$$



Пример применения алгоритма обратного распространения ошибки для сигмоидальной сети



Рассмотрим двухслойную сеть с сигмоидальными активационными функциями
Выход сети на i шаге итерации

$$O^i = \frac{1}{1+e^{-\mathbf{W}_2 \cdot \mathbf{s}^i}}$$

Выход скрытого слоя

$$s^i = \frac{1}{1+e^{-\mathbf{W}_1 \cdot \mathbf{x}}}$$

Пример применения алгоритма обратного распространения ошибки

Правило корректировки весов

$$\mathbf{W}_1 = \mathbf{W}_1 + \eta \frac{\partial E(\mathbf{W}_1, \mathbf{W}_2)}{\partial \mathbf{W}_1}$$

$$\mathbf{W}_2 = \mathbf{W}_2 + \eta \frac{\partial E(\mathbf{W}_1, \mathbf{W}_2)}{\partial \mathbf{W}_2}$$

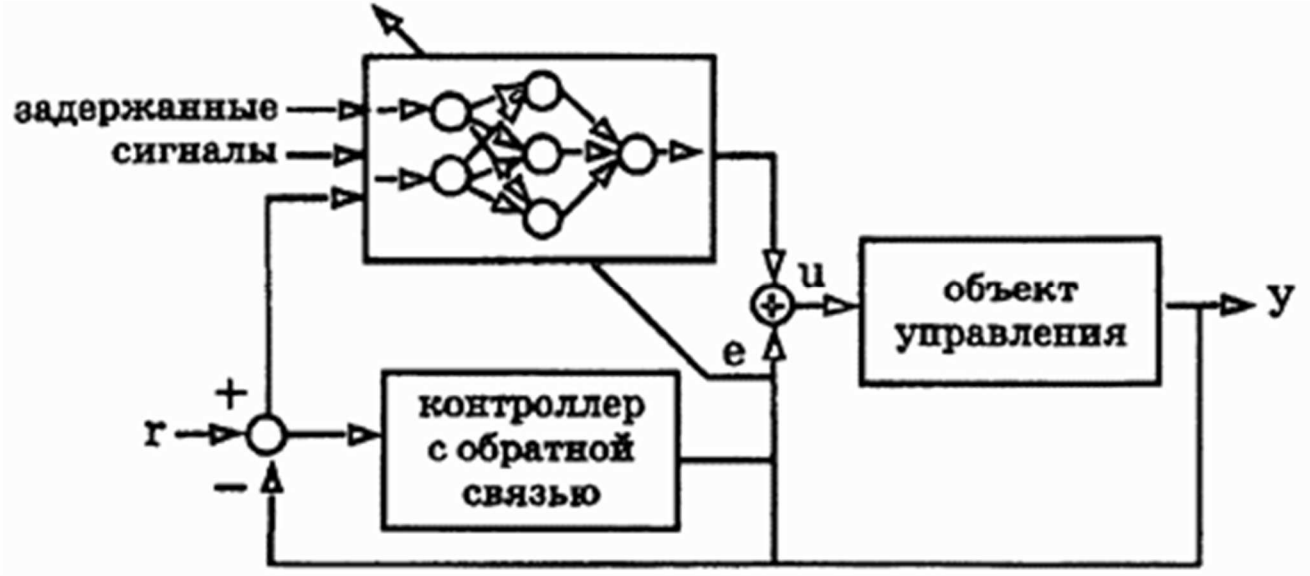
Где η - коэффициент скорости обучения.
Используя правило дифференцирования сложной функции, получим правило расчета весов на каждом шаге итерационного алгоритма

$$\mathbf{W}_1 = \mathbf{W}_1 + \eta \cdot \delta \cdot S^i$$

$$\mathbf{W}_2 = \mathbf{W}_2 + \eta \cdot \delta \cdot \mathbf{W}_1 \cdot S^i \left(1 - S^i\right) \cdot \mathbf{x}$$

$$\delta = (O_r - O^i) O^i (1 - O^i)$$

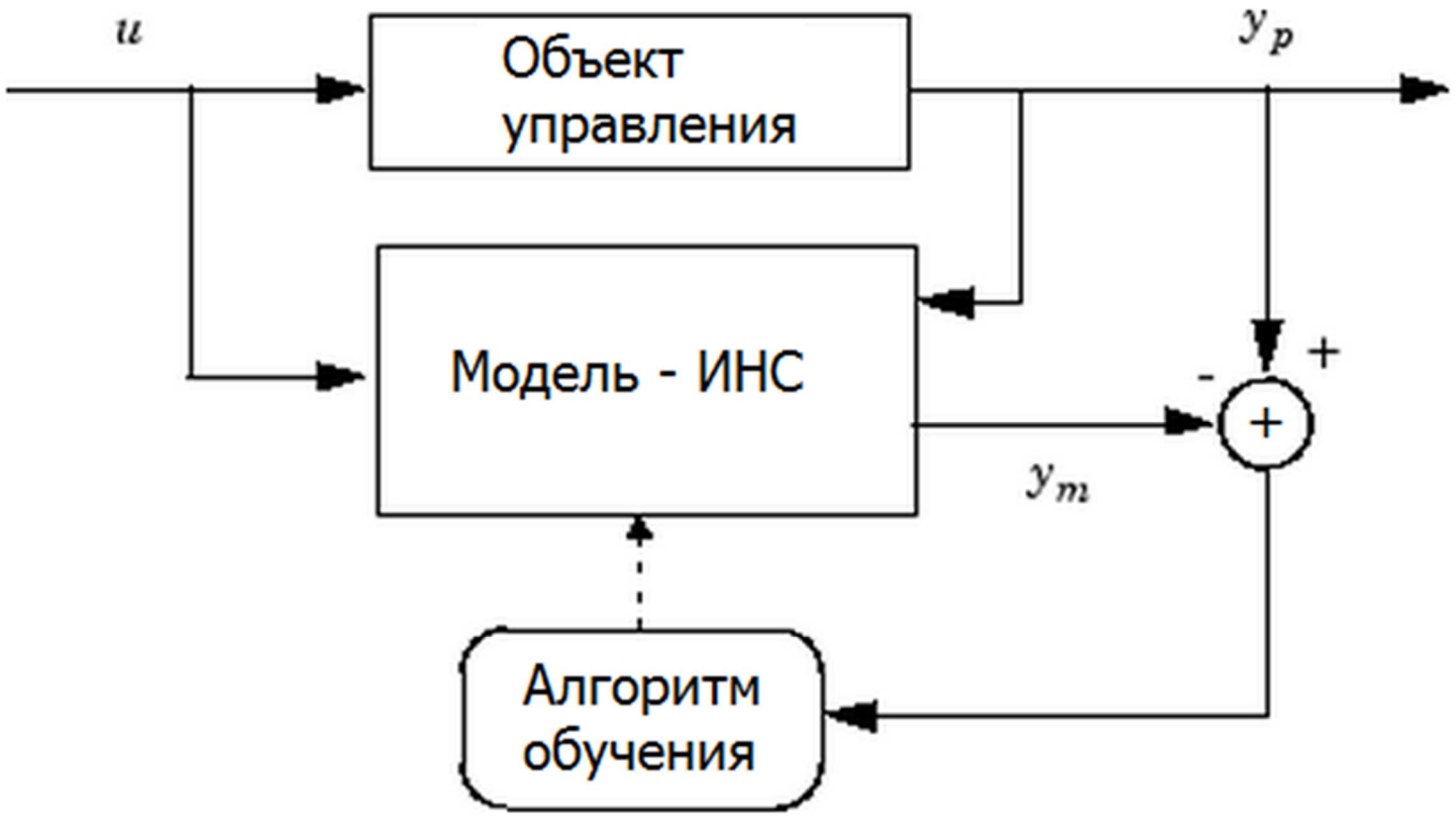
Где O_r - желаемое значение выхода сети



Искусственные нейронные сети в задачах управления

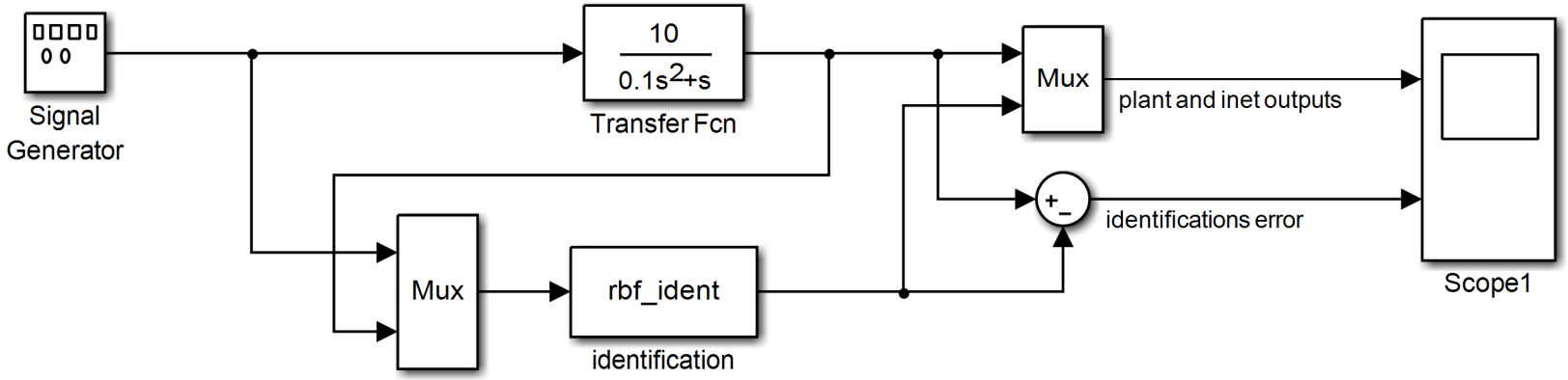
ИНС в системах управления

1. Задача идентификации объекта управления

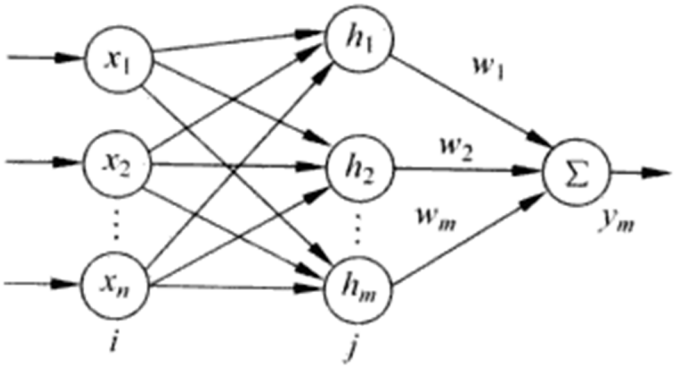


Пример: идентификация объекта управления RBF однослойной сетью

Структурная схема идентификации объекта управления



ИНС идентификации объекта



$$h_j = \exp(-\frac{\|X - c_j\|^2}{2b_j^2}),\ j = 1,..,m, c_j = [c_{1j}, c_{2j}, ..., c_{nj}]^T$$

$$y_m = \mathbf{W} \cdot \mathbf{H}, \mathbf{W} = [w_1, w_2, ..., w_m], \mathbf{H} = [h_1, h_2, ..., h_m]^T$$

Пример: идентификация объекта управления RBF однослойной сетью

Алгоритм настройки весов для k-шага:

$$w_j^k = w_j^{k-1} + \eta \cdot (y - y_m) h_j^{k-1} + \alpha \cdot (w_j^{k-1} - w_j^{k-2})$$

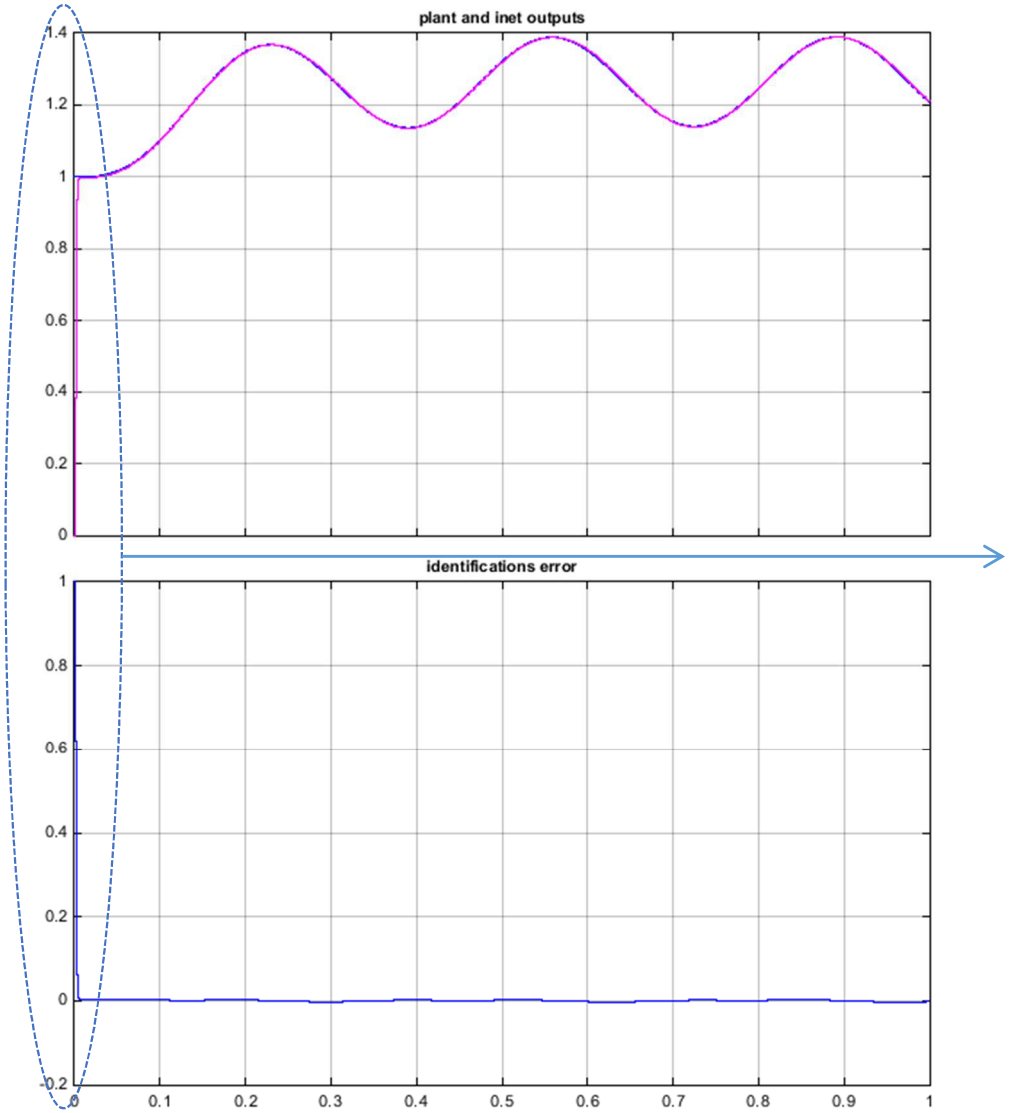
$$b_j^k = b_j^{k-1} + \eta \cdot \frac{(y - y_m) h_j^{k-1} w_j^{k-1} \left\| X - c_j \right\|^2}{(b_j^{k-1})^3} + \alpha \cdot (b_j^{k-1} - b_j^{k-2})$$

$$c_{ij}^k = c_{ij}^{k-1} + \eta \cdot \frac{(y - y_m) w_j^{k-1} (x_i - c_{ij}^{k-1})}{(b_j^{k-1})^2} + \alpha \cdot (c_{ij}^{k-1} - c_{ij}^{k-2})$$

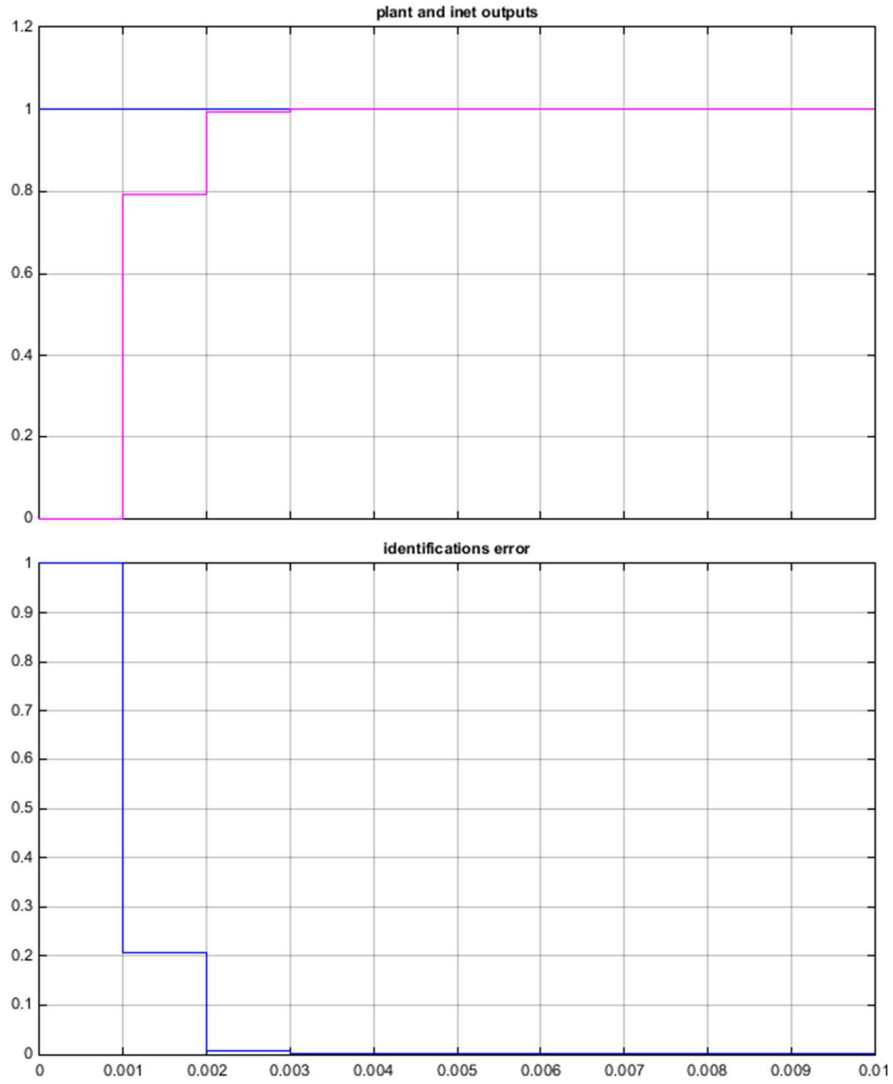
α, η - коэффициенты, определяющие скорость сходимости алгоритма

Пример: идентификация объекта управления RBF однослойной сетью

Интегрирование с постоянным шагом 0.001

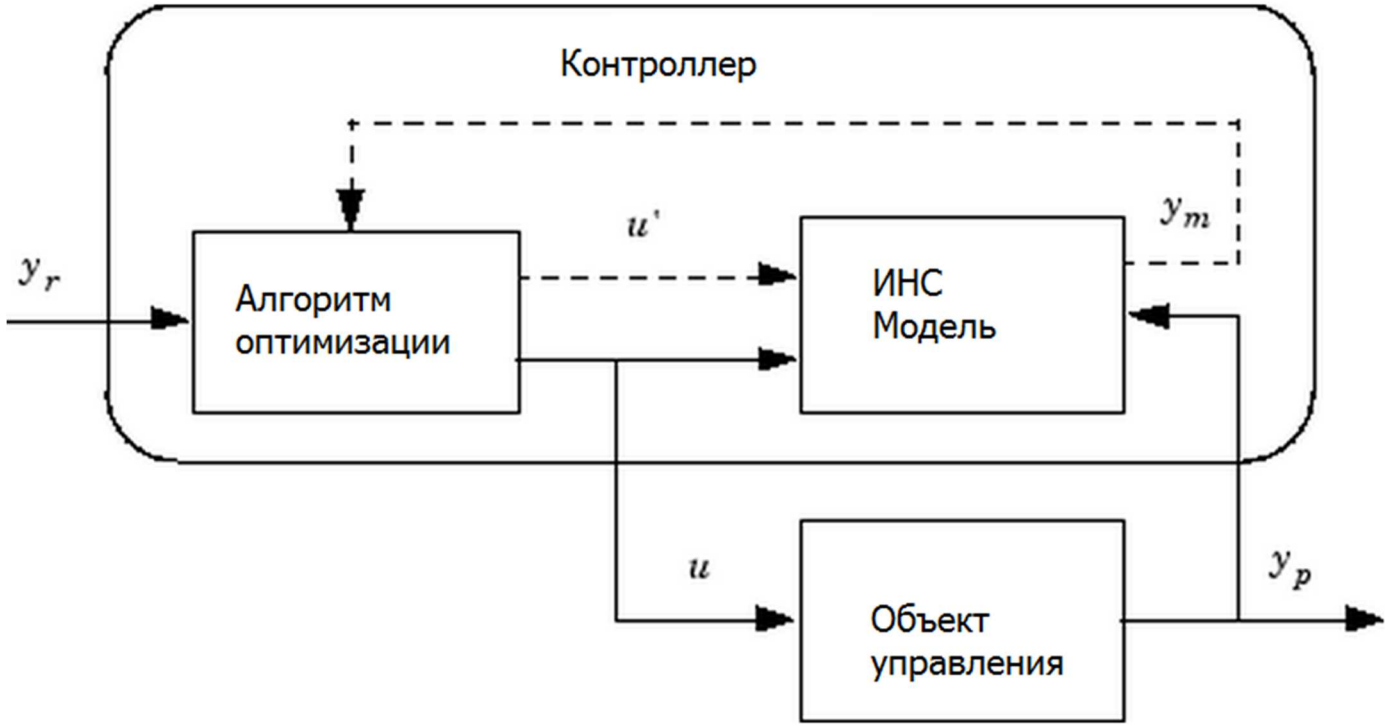


Скорость сходимости - 3 шага



ИНС в системах управления

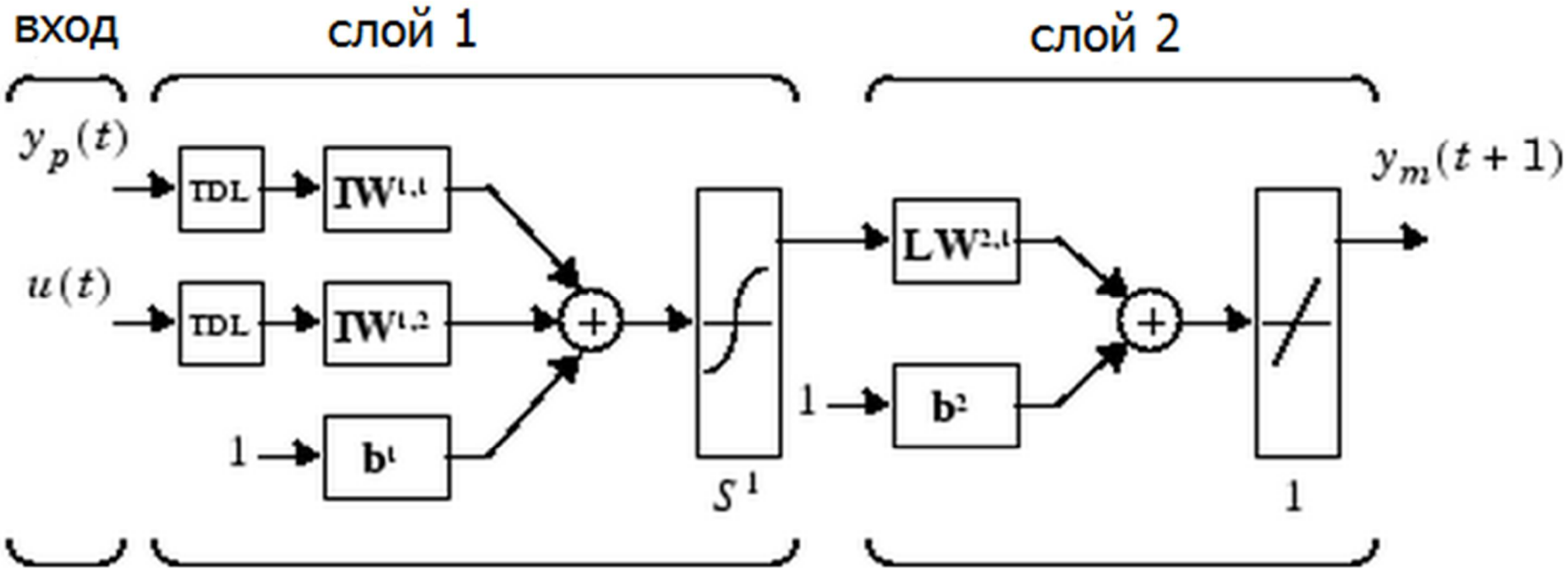
2. Прогнозирующее управление на основе идентификации объекта управления



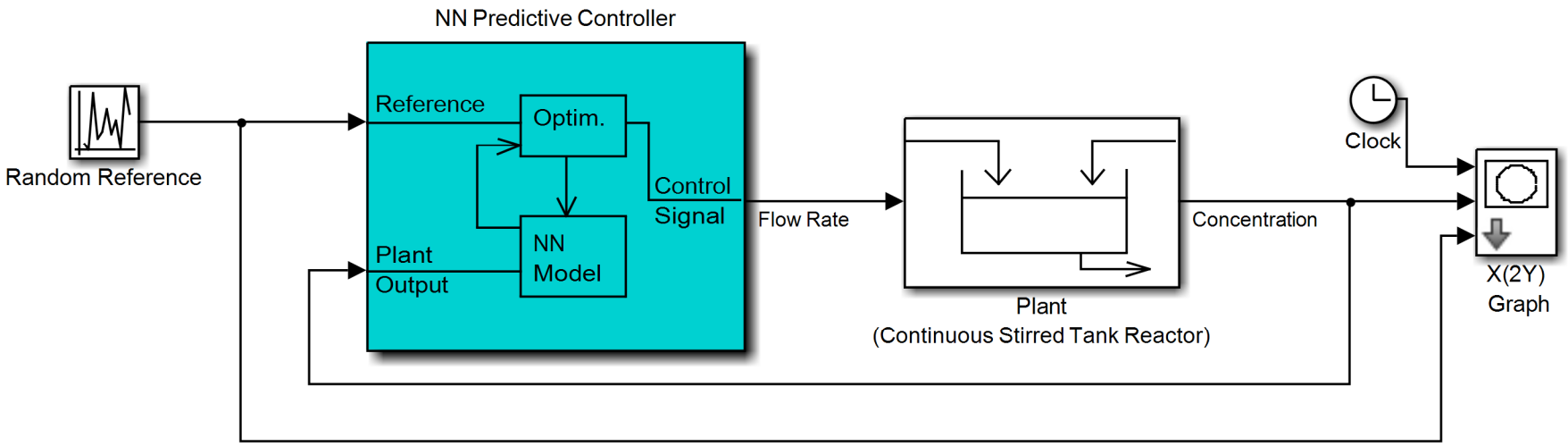
Критерий качества прогнозирующего управления для алгоритма оптимизации

$$J = \sum_{j=N_1}^{N_2} (y_r(t+j) - y_m(t+j))^2 + \rho \sum_{j=1}^{N_u} (u'(t+j-1) - u'(t+j-2))^2$$

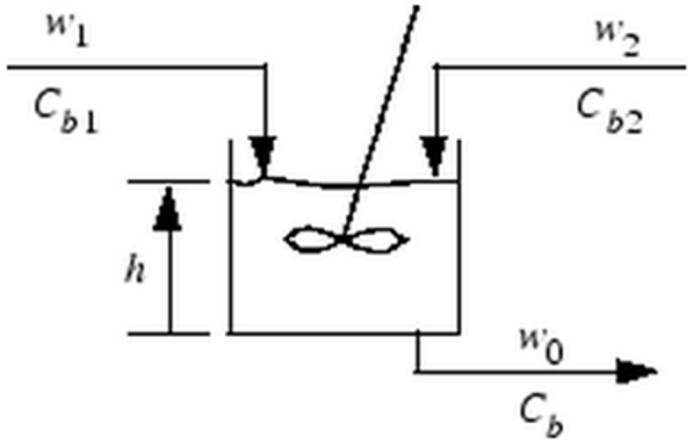
Структура ИНС идентификации объекта управления



Пример прогнозирующего управления на основе идентификации объекта управления



Объект управления - смешивающий каталитический реактор



$$\frac{dh(t)}{dt} = w_1(t) + w_2(t) - 0.2\sqrt{h(t)}$$

$$\frac{dC_b(t)}{dt} = (C_{b1} - C_b(t))\frac{w_1(t)}{h(t)} + (C_{b2} - C_b(t))\frac{w_2(t)}{h(t)} - \frac{k_1 C_b(t)}{(1 + k_2 C_b(t))^2}$$

h – уровень жидкости

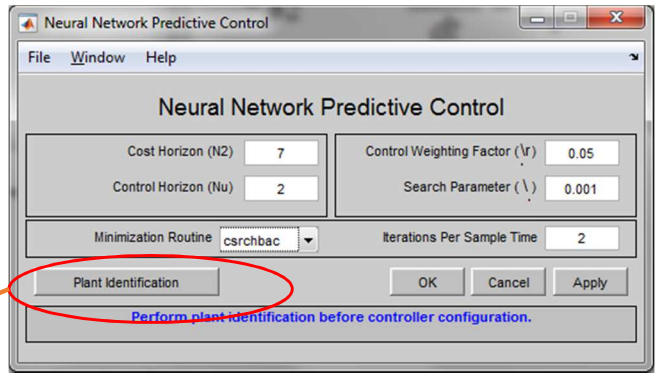
w1,Cb1- расход и концентрация вещества 1

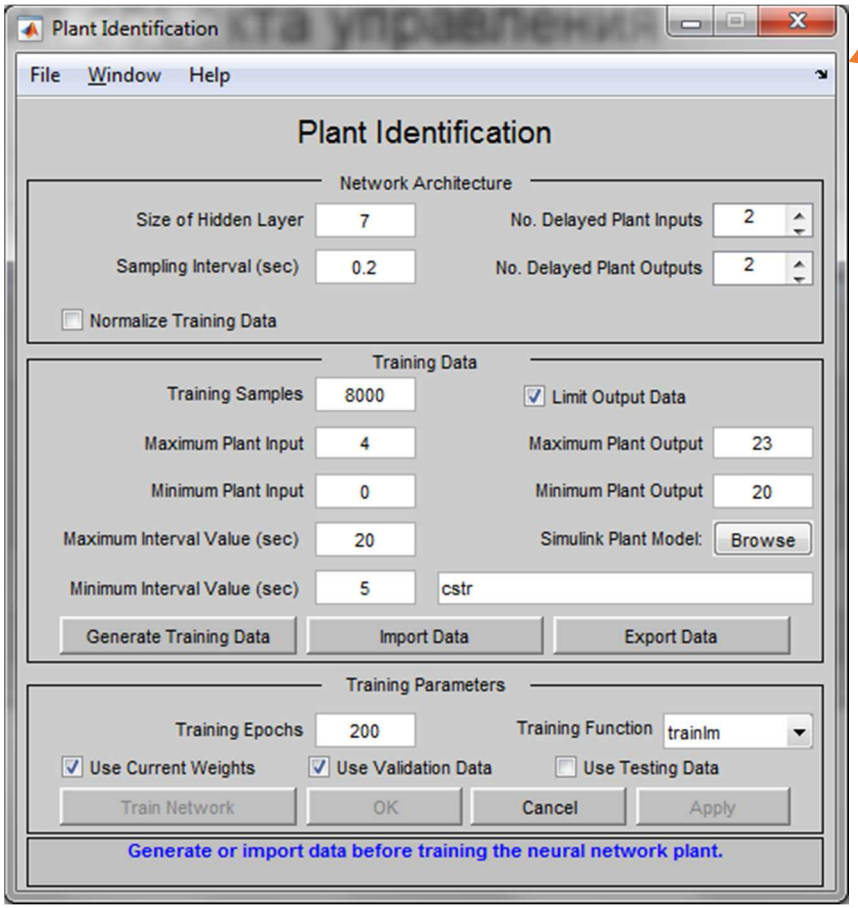
w2,Cb2- расход и концентрация вещества 2

w0,Cb- расход и концентрация выхода смешивающего реактора

Настройка модели

I. идентификация объекта управления





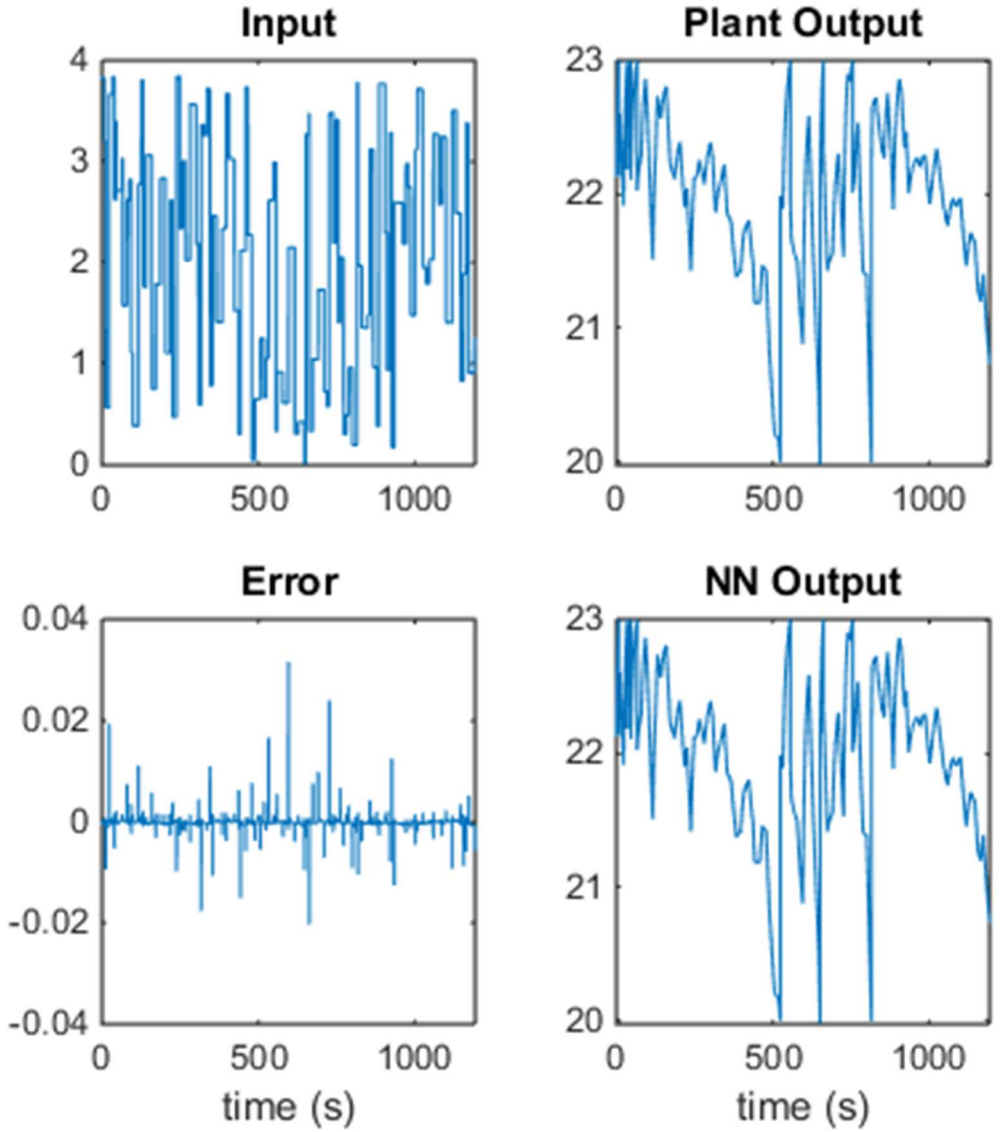
- выбор параметров архитектуры ИНС

- подготовка обучающей выборки

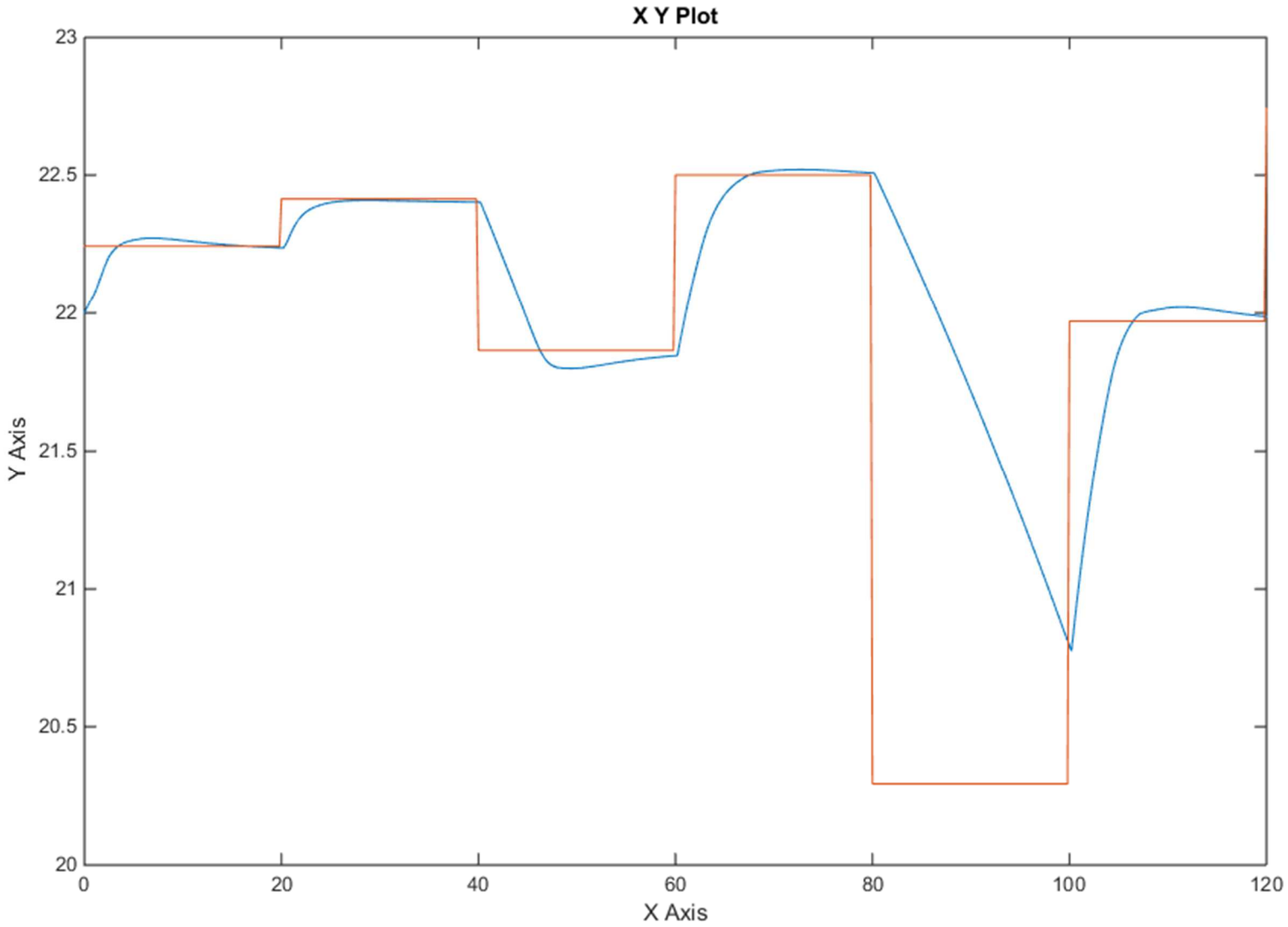
- обучение нейросети

Настройка модели

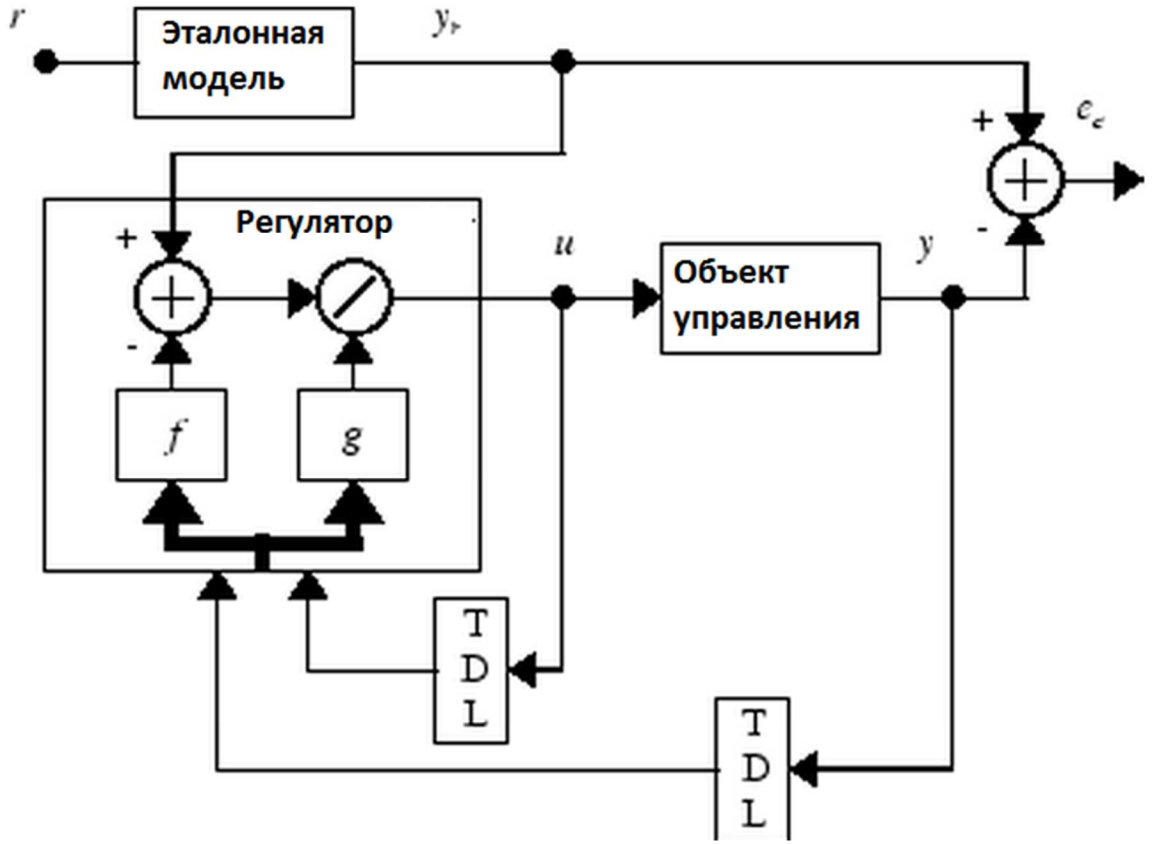
2. Обучение ИНС – проверка ошибки обучения



Результат управления



2. NARMA-L2 Регулятор



NARMA

$$y(k+d) = N[y(k), y(k-1), \ldots, y(k-n+1), u(k), u(k-1), \ldots, u(k-n+1)]$$

Non Linear — нелинейная

Auto-Regressive — авторегрессионная

Moving Average — скользящее среднее

Регулятор основан на дискретном NARMA представлении модели системы

NARMA-L2 Регулятор

Зачастую используется совместно с эталонной моделью системы с выходом y_r тогда задача синтеза нейросетевого регулятора сводится к обучению такой нелинейной функции G

$$u(k) = G[y(k), y(k-1), \dots, y(k-n+1), y_r(k+d), u(k-1), \dots, u(k-m+1)]$$

,что в процессе управления минимизируется среднеквадратичная ошибка управления e= y_r-y

Один из методов решения задачи основан на декомпозиции выхода

$$\hat{y}(k+d) = f[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-m+1)]$$
$$+g[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-m+1)] \cdot u(k)$$

Тогда управление системой формируется

$$u(k) = \frac{y_r(k+d) - f[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-n+1)]}{g[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-n+1)]}$$

Подобное управление практически не реализуемо, т.к. управление в к-ый момент времени должно зависеть от выхода y(k)

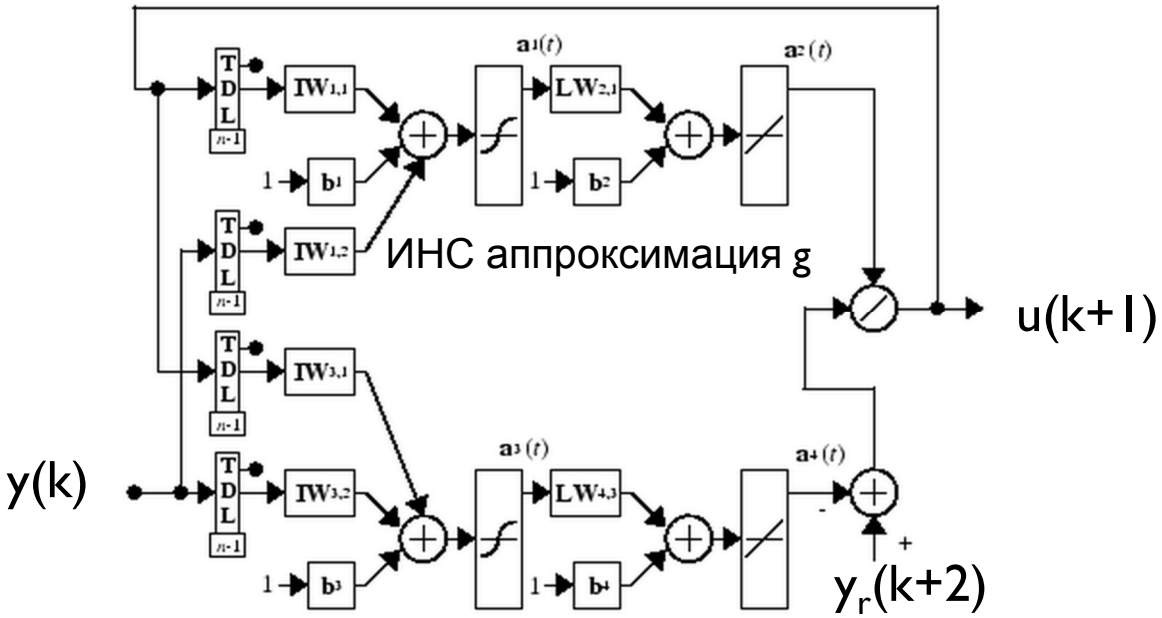
NARMA-L2 Регулятор

Таким образом используют уравнение (при d>1)

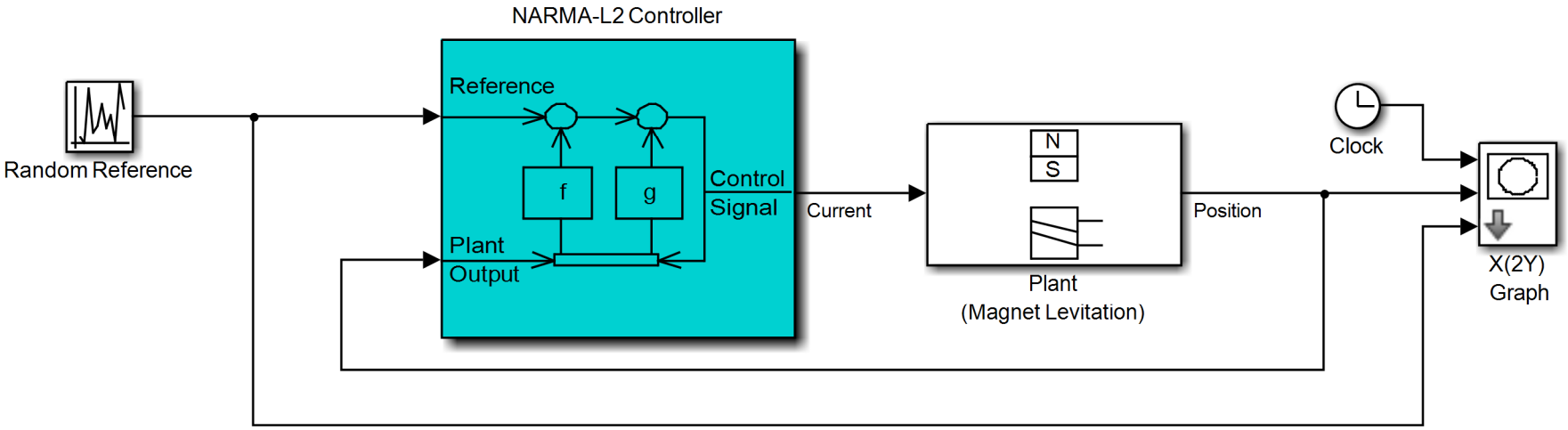
$$y(k+d) = f[y(k), y(k-1), \ldots, y(k-n+1), u(k), u(k-1), \ldots, u(k-n+1)]$$
$$+g[y(k), \ldots, y(k-n+1), u(k), \ldots, u(k-n+1)] \cdot u(k+1)$$

Тогда управление системой формируется

$$u(k+1) = \frac{y_r(k+d) - f[y(k), \ldots, y(k-n+1), u(k), \ldots, u(k-n+1)]}{g[y(k), \ldots, y(k-n+1), u(k), \ldots, u(k-n+1)]}$$



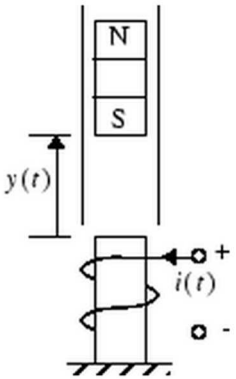
Пример NARMA-L2 регулятор электромагнитным подвесом



NARMA-L2 Control of a Magnet Levitation System
(Double click on the "?" for more info)

?

To start and stop the simulation, use the "Start/Stop"
selection in the "Simulation" pull-down menu

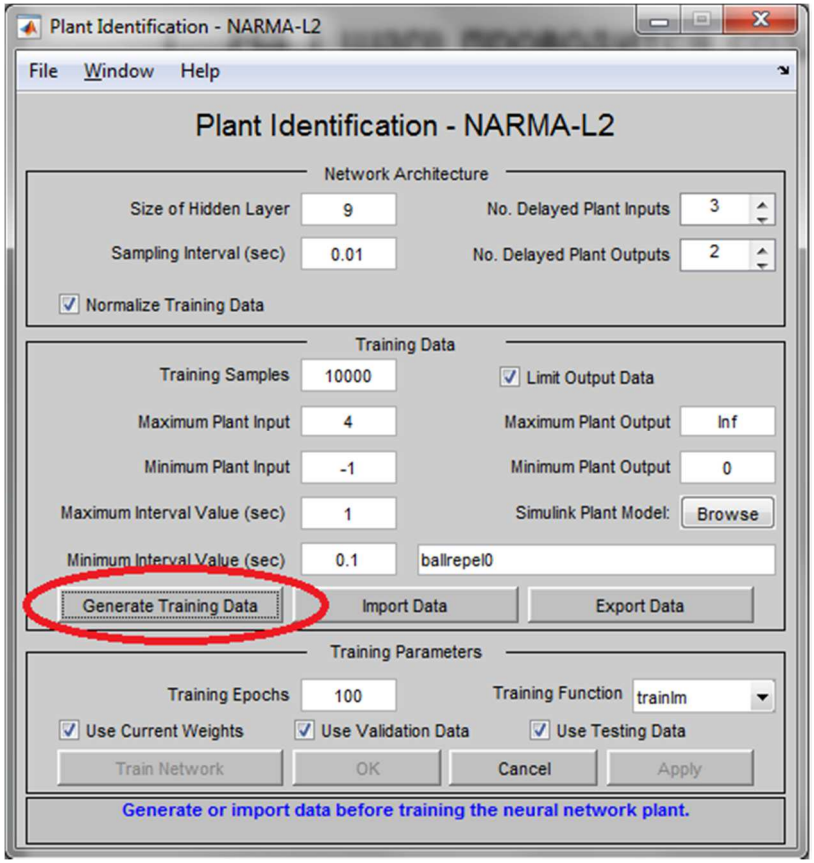


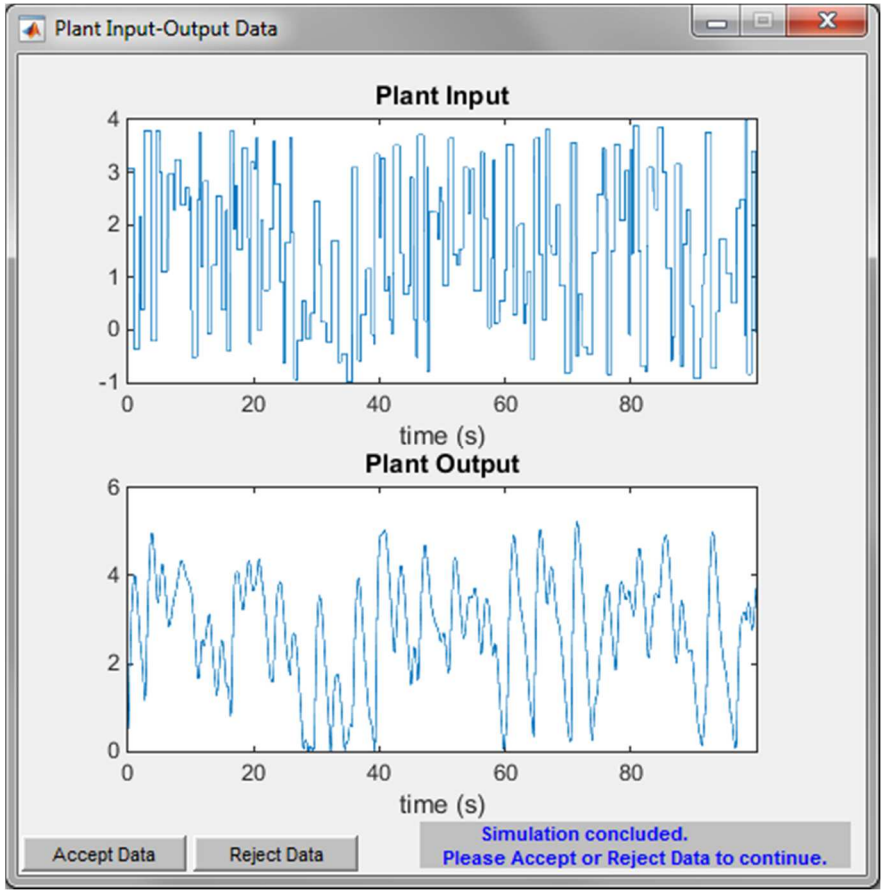
Уравнение динамики объекта управления

$$\frac{d^2y(t)}{dt^2} = -g + \frac{\alpha}{M}\frac{i^2(t)}{y(t)} - \frac{\beta}{M}\frac{dy(t)}{dt}$$

Пример NARMA-L2 регулятор электромагнитным подвесом

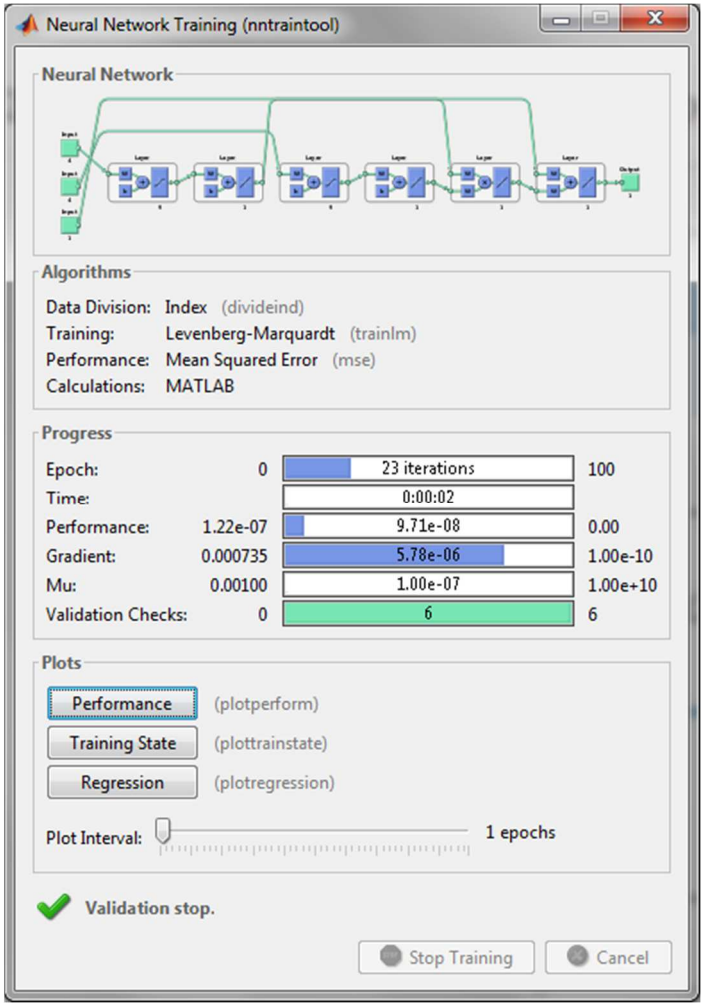
I. На I шаге проводится создание данных для идентификации контроллера

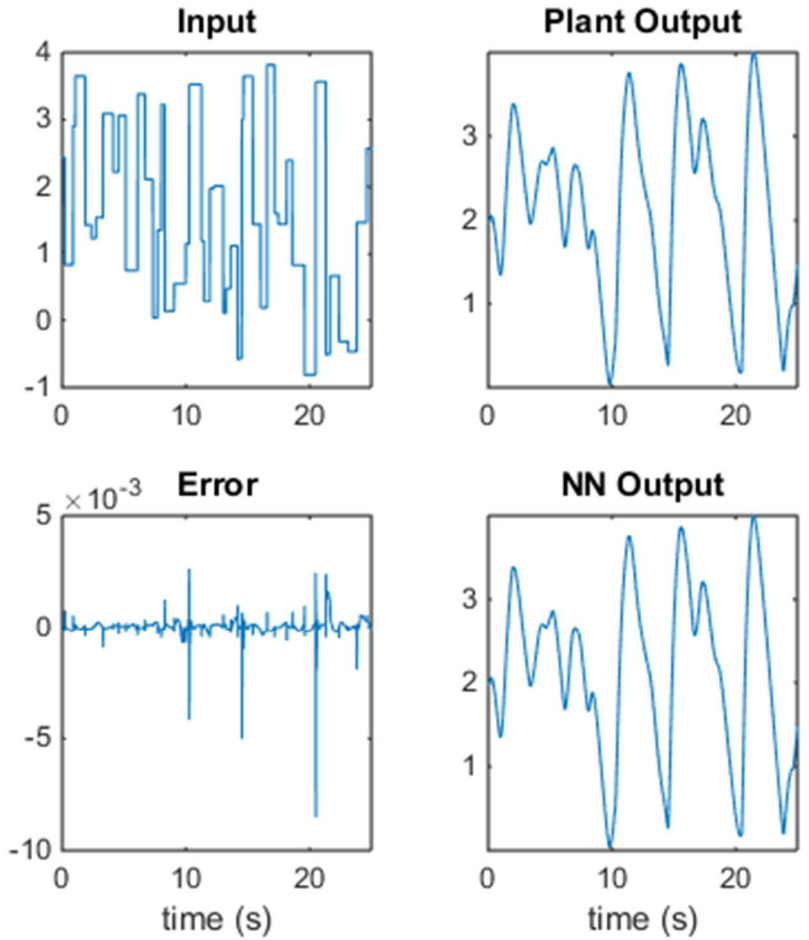




Пример NARMA-L2 регулятор электромагнитным подвесом

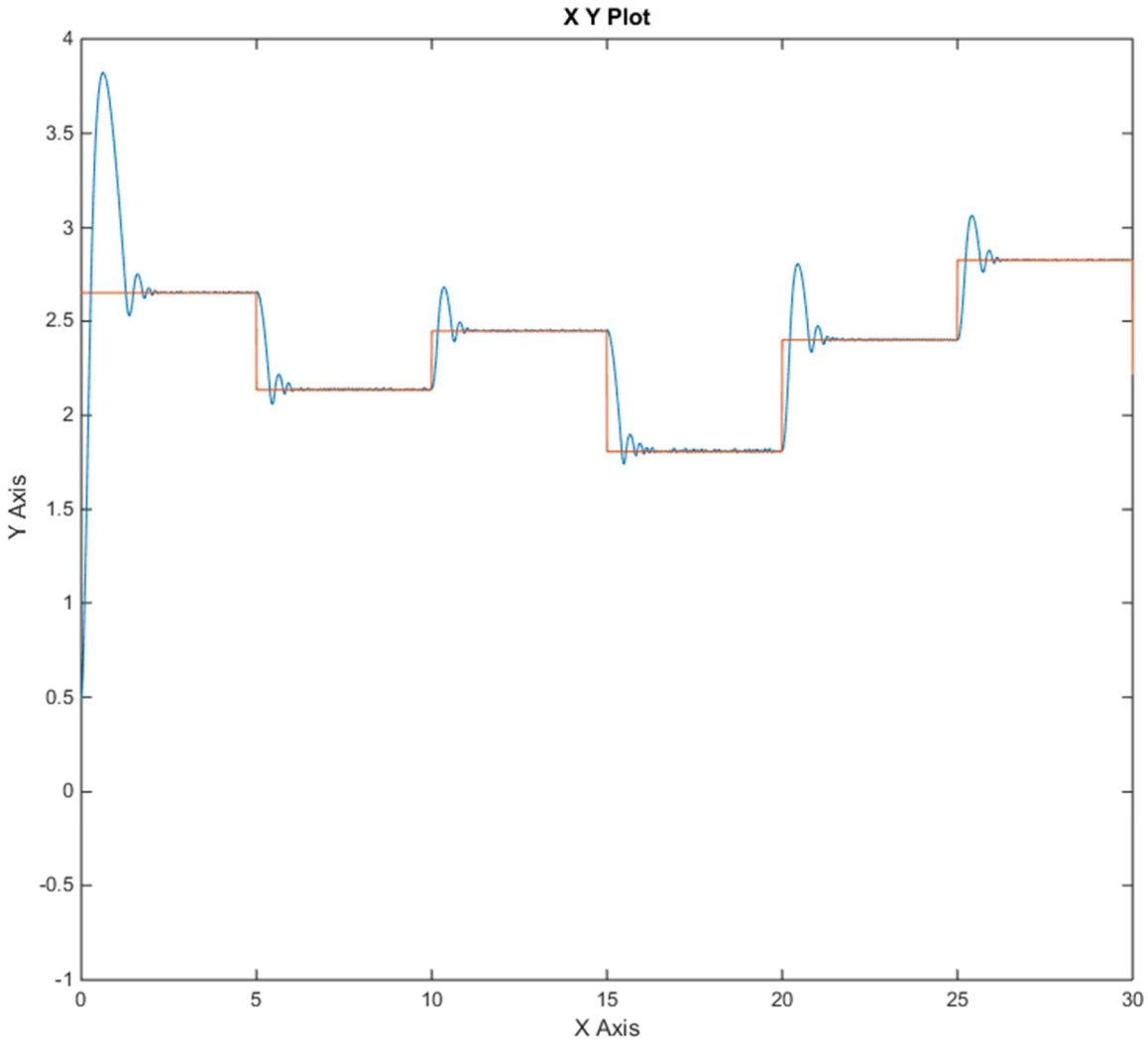
2. На 2 шаге проводится обучение нейросетевого контроллера





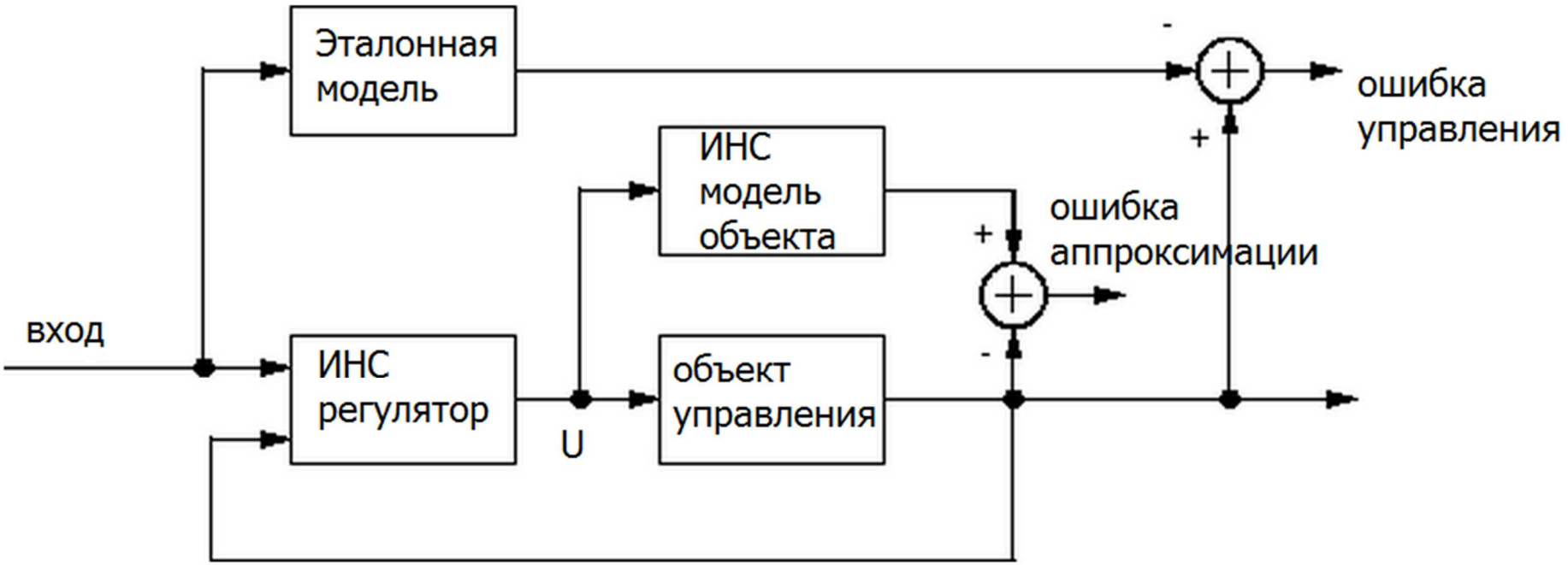
Пример NARMA-L2 регулятор электромагнитным подвесом

Результат моделирования

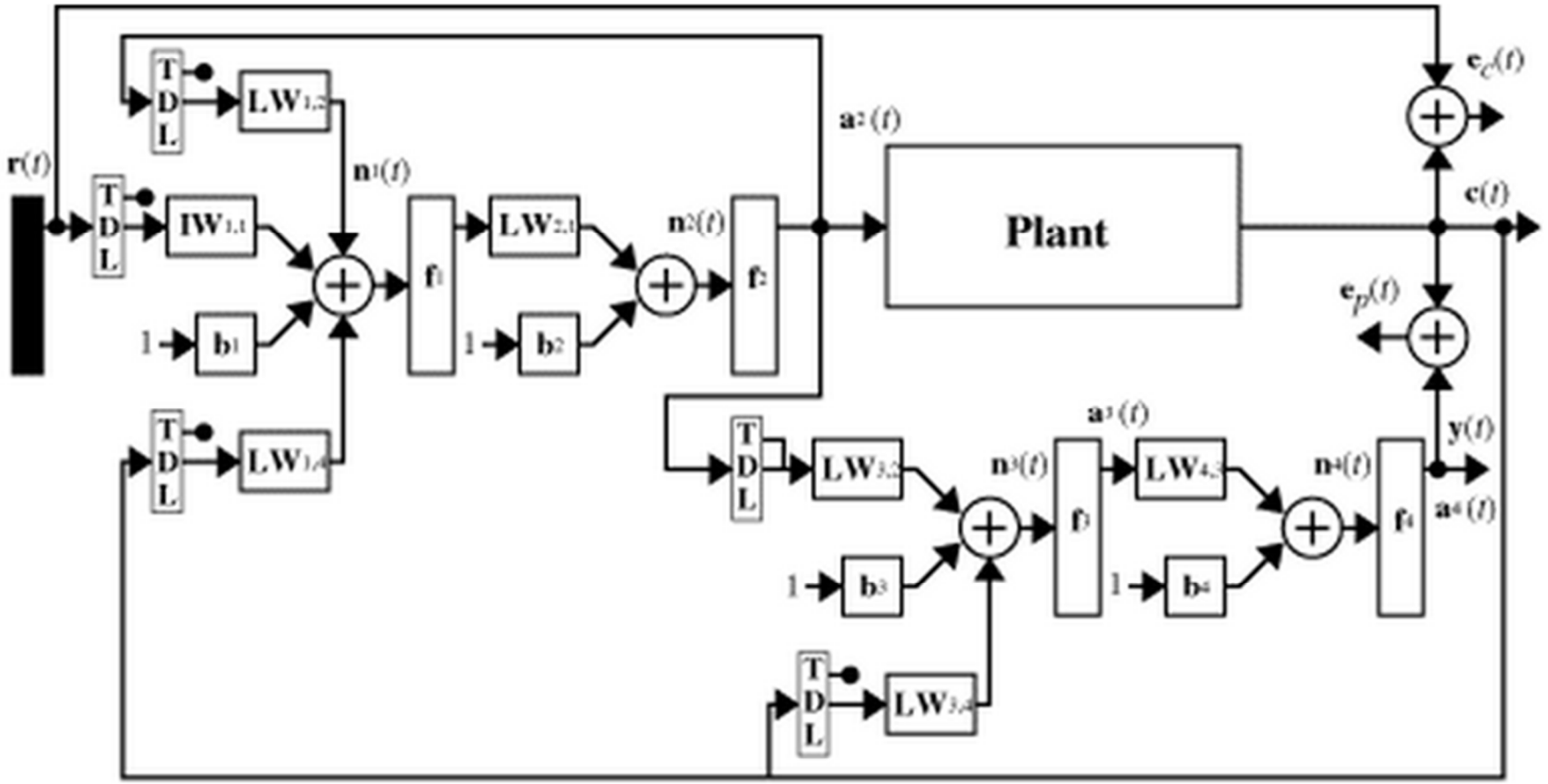


ИНС в системах управления

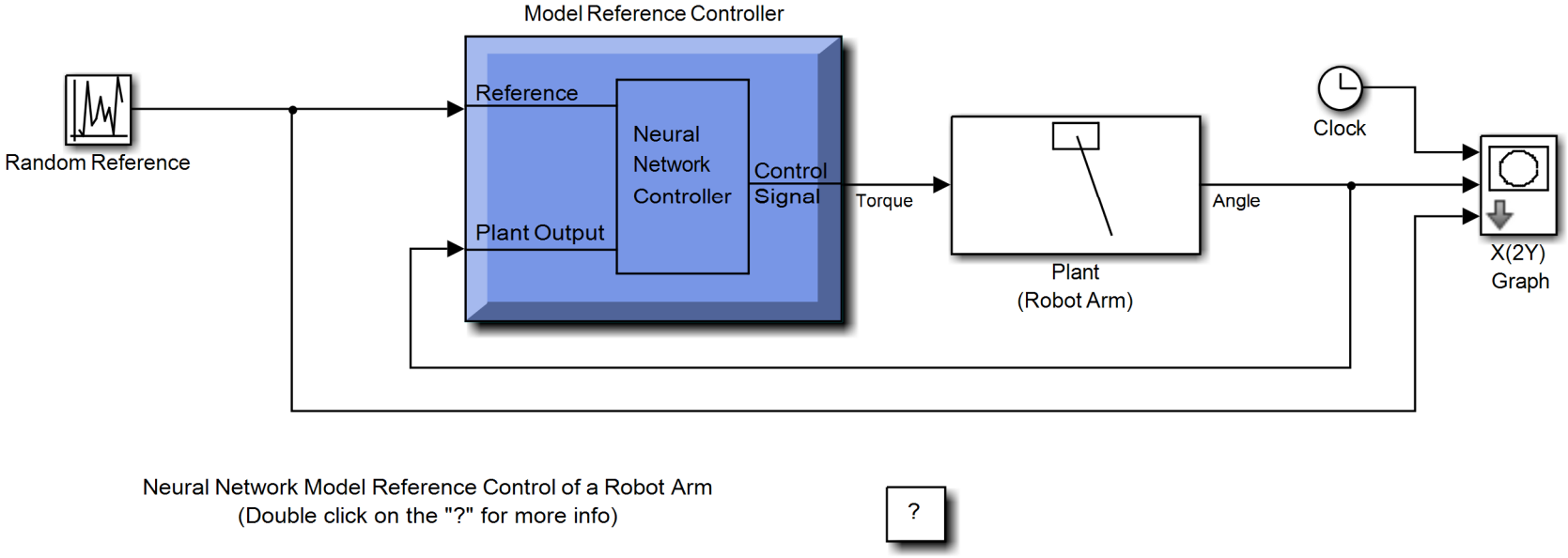
3. Адаптивное управление на основе эталонной модели

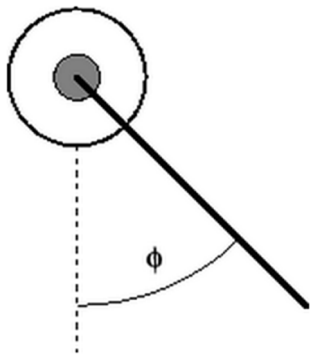


Структура нейросетевой системы управления



Пример ИНС системы управления подъемом балки





Модель системы управления

$$\frac{d^2\phi}{dt^2} = -10\sin\phi - 2\frac{d\phi}{dt} + u$$

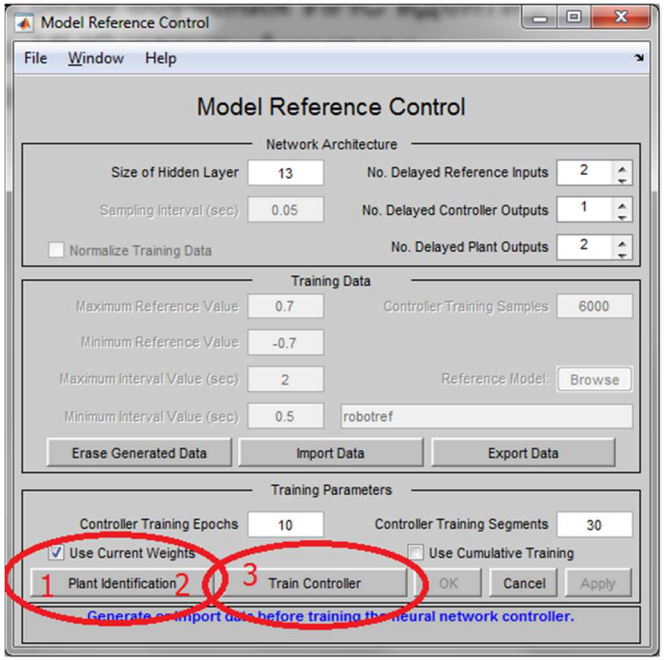
Эталонная модель

$$\frac{d^2y_r}{dt^2} = -9y_r - 6\frac{dy_r}{dt} + 9r$$

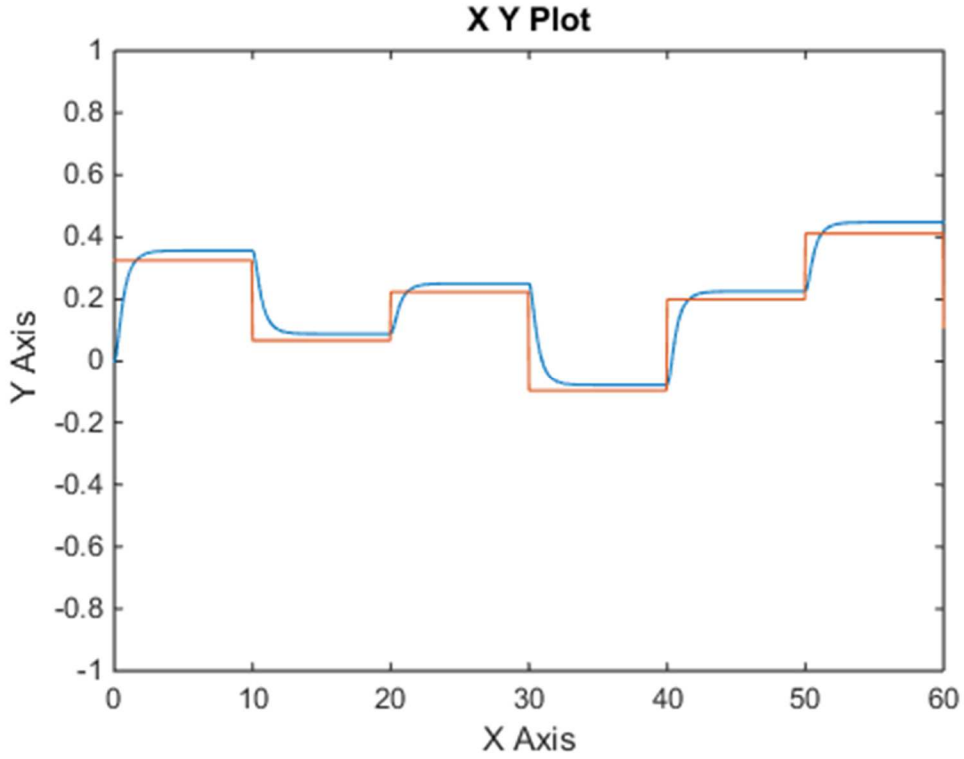
Пример ИНС системы управления подъемом балки

Алгоритм применения контроллера

1. Создание базы обучения ИНС идентификатора
2. Обучение ИНС идентификатора
3. Обучение ИНС контроллера

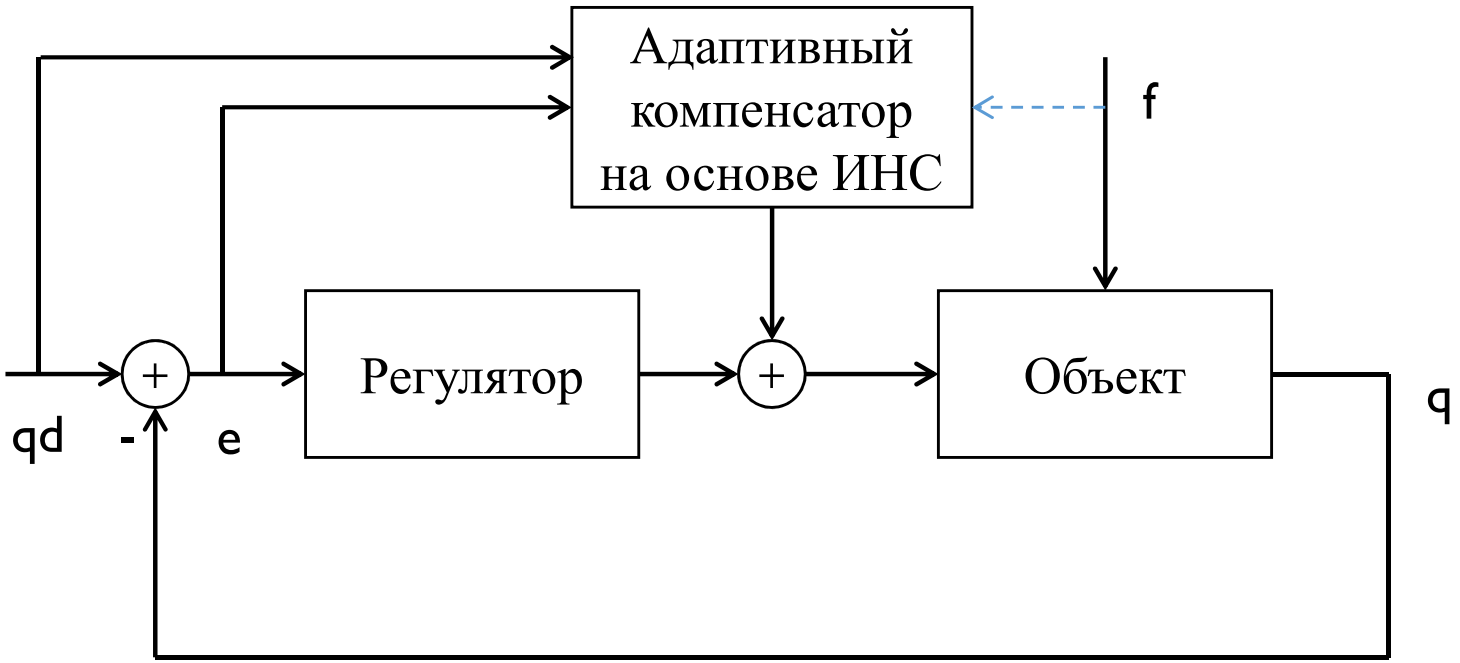


Результат управления



ИНС в системах управления

4. Прямое адаптивное управление на основе RBF ИНС



5. Прямое адаптивное управление на основе RBF ИНС

Динамика манипулятора описывается матричным уравнением Лагранжа:

$$\mathbf{M}(\mathbf{q},\xi)\cdot\ddot{\mathbf{q}}+\mathbf{N}(\mathbf{q},\dot{\mathbf{q}},\xi)=\boldsymbol{\tau}-\boldsymbol{\tau}_d$$

Будем рассматривать управление на основе обратной модели динамики робота в виде:

$$\boldsymbol{\tau}=\mathbf{M}_0(\ddot{q}_d+K_v\dot{e}+K_p e)+\mathbf{N}_0$$

M_0- начальная оценка матрицы манипулятора
N_0- начальная оценка матрицы кориолисовых, центробежных и гравитационных сил
K_v- диагональная матрица дифференциальных коэффициентов регулятора
K_p- диагональная матрица пропорциональных коэффициентов регулятора

Применение подобного управления приведет к уравнению ошибок вида

$$\ddot{e}+K_v\dot{e}+K_p e=\mathbf{M}_0^{-1}(\Delta\mathbf{M}\ddot{q}+\Delta\mathbf{N}+\tau_d)$$

$$\Delta\mathbf{M}=\mathbf{M}-\mathbf{M}_0$$

$$\Delta\mathbf{N}=\mathbf{N}-\mathbf{N}_0$$

Уравнение для ошибок замкнутой системы представим в матричной форме

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{f}$$

$$\mathbf{A} = \begin{vmatrix} 0 & I \\ K_p & K_v \end{vmatrix}, \mathbf{x} = \begin{vmatrix} e \\ \dot{e} \end{vmatrix}, \mathbf{f} = \mathbf{M}_0^{-1}(\Delta\mathbf{M}\ddot{q} + \Delta\mathbf{N} + \tau_d)$$

f- функция неопределенностей модели динамики робота

Будем искать значение оценки функции неопределенностей в виде RBF функционала

$$\mathbf{f} = \mathbf{M_0^{-1}}\theta \cdot \varphi, где\ \theta \text{ - вектор синаптичеких весов, а}$$

$$\varphi = [\varphi_1, \varphi_2, ..., \varphi_n] - \text{вектор RBF функций}$$

$$\varphi_i = e^{\frac{\|\mathbf{x} - c_i\|^2}{\sigma^2}}$$

Таким образом, будем управлять роботом по следующему закону

$$\tau = \tau_1 + \tau_2$$

$$\tau_1 = \mathbf{M}_0(\ddot{q}_d + K_v\dot{e} + K_p e) + \mathbf{N}_0$$

$$\tau_2 = -\mathbf{M}_0^{-1}\hat{\theta}\cdot\varphi$$

Пересчет весовых коэффициентов RBF сети определяется уравнением

$$\dot{\hat{\theta}} = \gamma\varphi(\mathbf{x})\mathbf{x}^T PB + k_1\gamma\|\mathbf{x}\|\hat{\theta}$$

где P- решение уравнения Ляпунова

$$P\mathbf{A} + \mathbf{A}^T P = Q$$

Q –положительно определенная матрица,

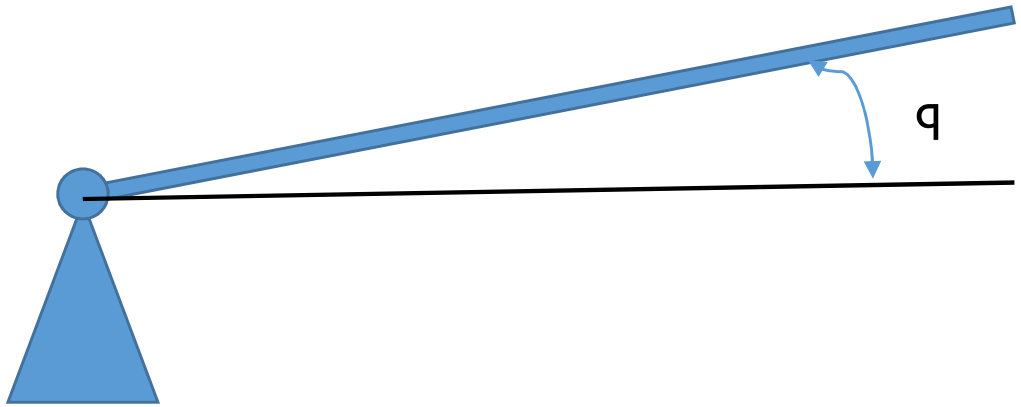
γ,k_1- коэффициенты настройки алгоритма

Пример адаптивного управления с RBF ИНС компенсатором вращением стержня с переменной нагрузкой

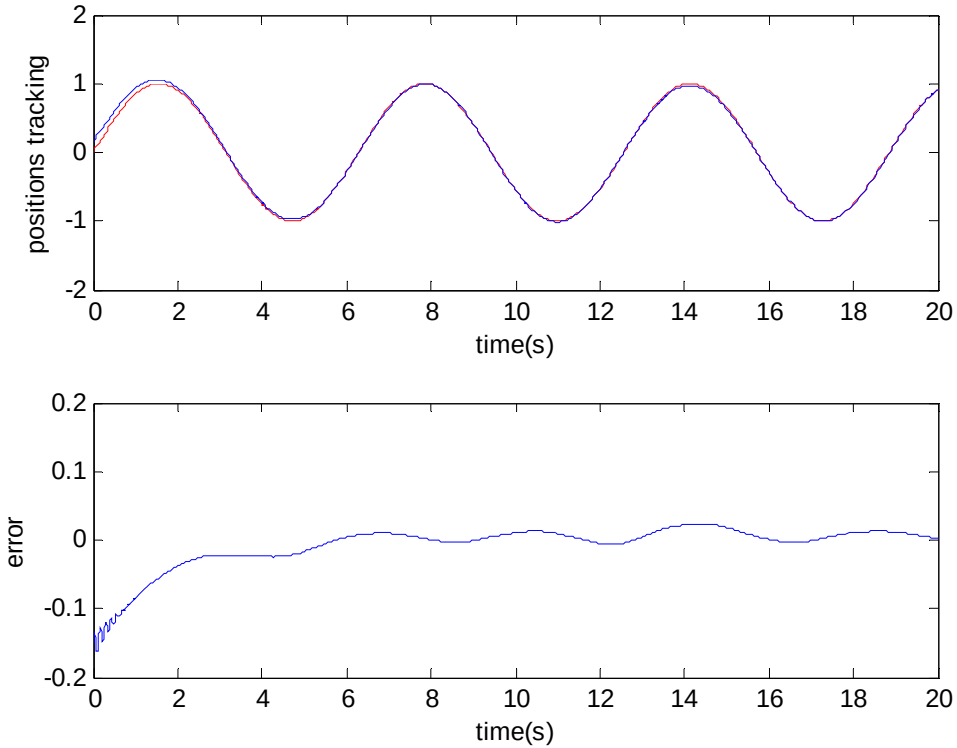
$$\mathbf{M}_0\ddot{q} + \mathbf{N}_0 = \tau + \tau_d$$

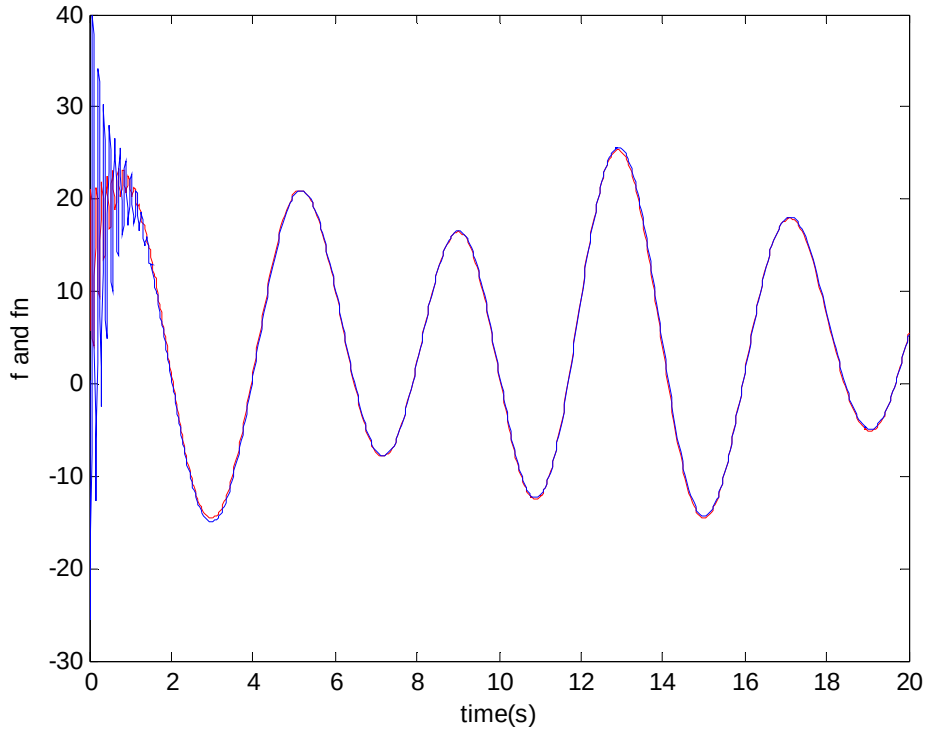
$$\mathbf{M}_0 = \frac{3}{4}ml^2, \mathbf{N}_0 = mgl\cos(q), \tau_d = 1.3\sin(0.5\pi t),$$

$$m = 1, l = 0.25, g = 9.81$$

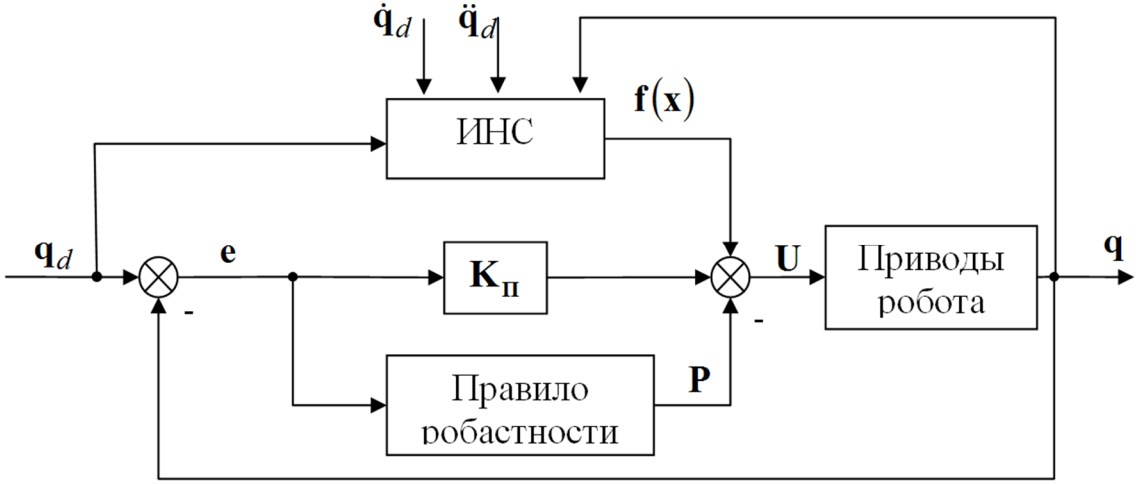


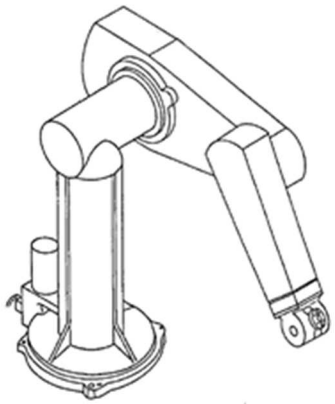
Пример адаптивного управления с RBF ИНС компенсатором вращением стержня с переменной нагрузкой





6. Прямое адаптивное управление роботом на основе сигмоидальной ИНС





$$U = (f(x) + K_п e - P),$$

$$f(x) = W^T \sigma(V^T x),$$

$$\dot{W} = F\sigma \cdot e^T - F\sigma \cdot V^T x e^T - kF\|e\|W,$$

$$\dot{V} = Gx(\sigma'^T \cdot We)^T - kG\|e\|V,$$

$$P = -K_Z(\|Z\|_F + Z_M) \cdot e$$

$$Z = \begin{vmatrix} W & 0 \\ 0 & V \end{vmatrix} \quad x = [1, q_d^T, \dot{q}_d^T, \ddot{q}_d^T, q^T]^T$$

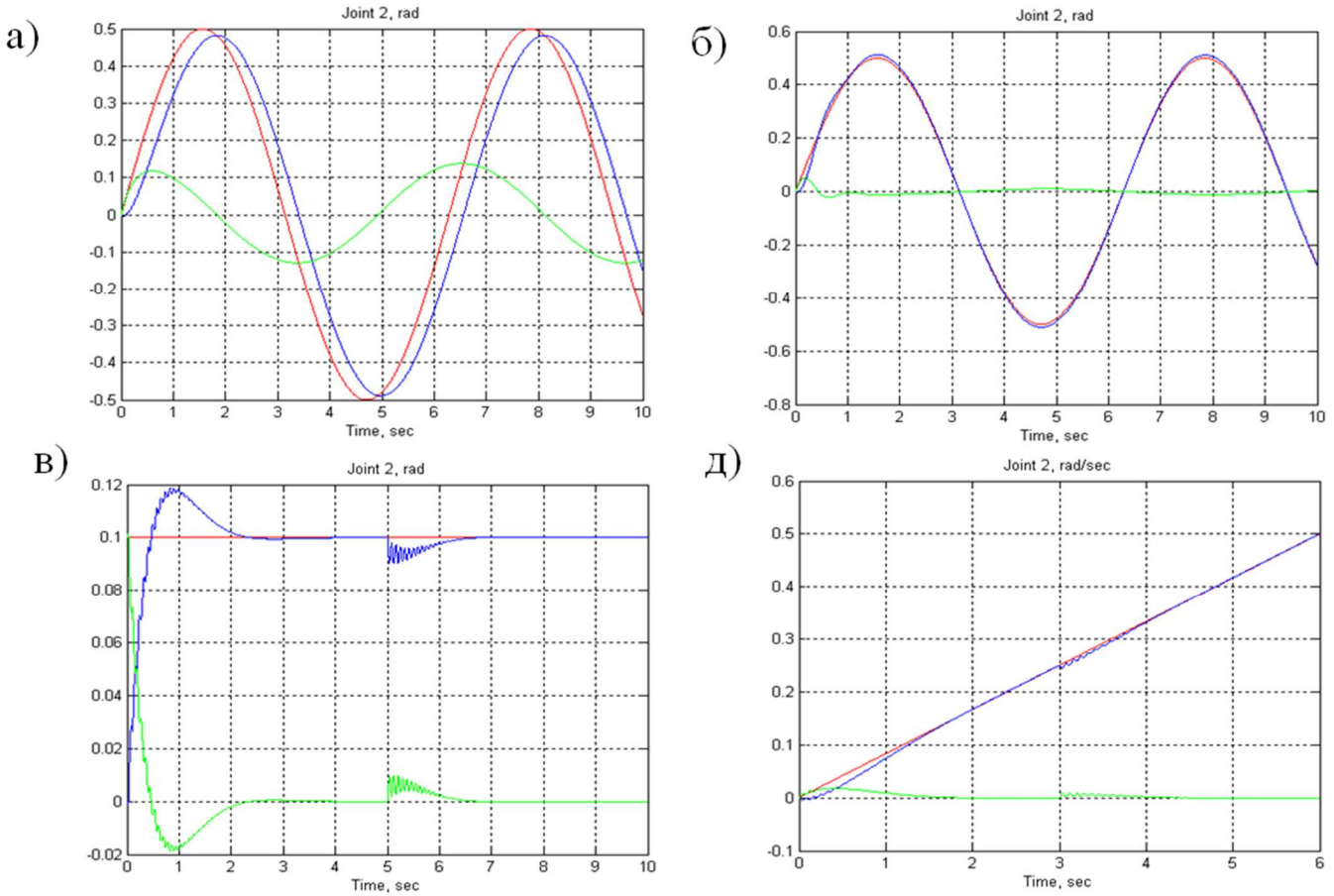
W- матрица весовых коэффициентов от скрытого слоя к входному, **V** - матрица весовых коэффициентов от входного слоя к скрытому, $\sigma(.)$ - сигмоидальная активационная функция скрытого слоя.

F,**G**, K_Z - положительно определённые матрицы коэффициентов настройки алгоритма, k- скалярный коэффициент настройки, **e** - ошибка управления.

$\|Z\|_F$ - норма Фробениуса неизвестных целевых весов для матрицы Z

Z_M - максимальное значение нормы Фробениуса

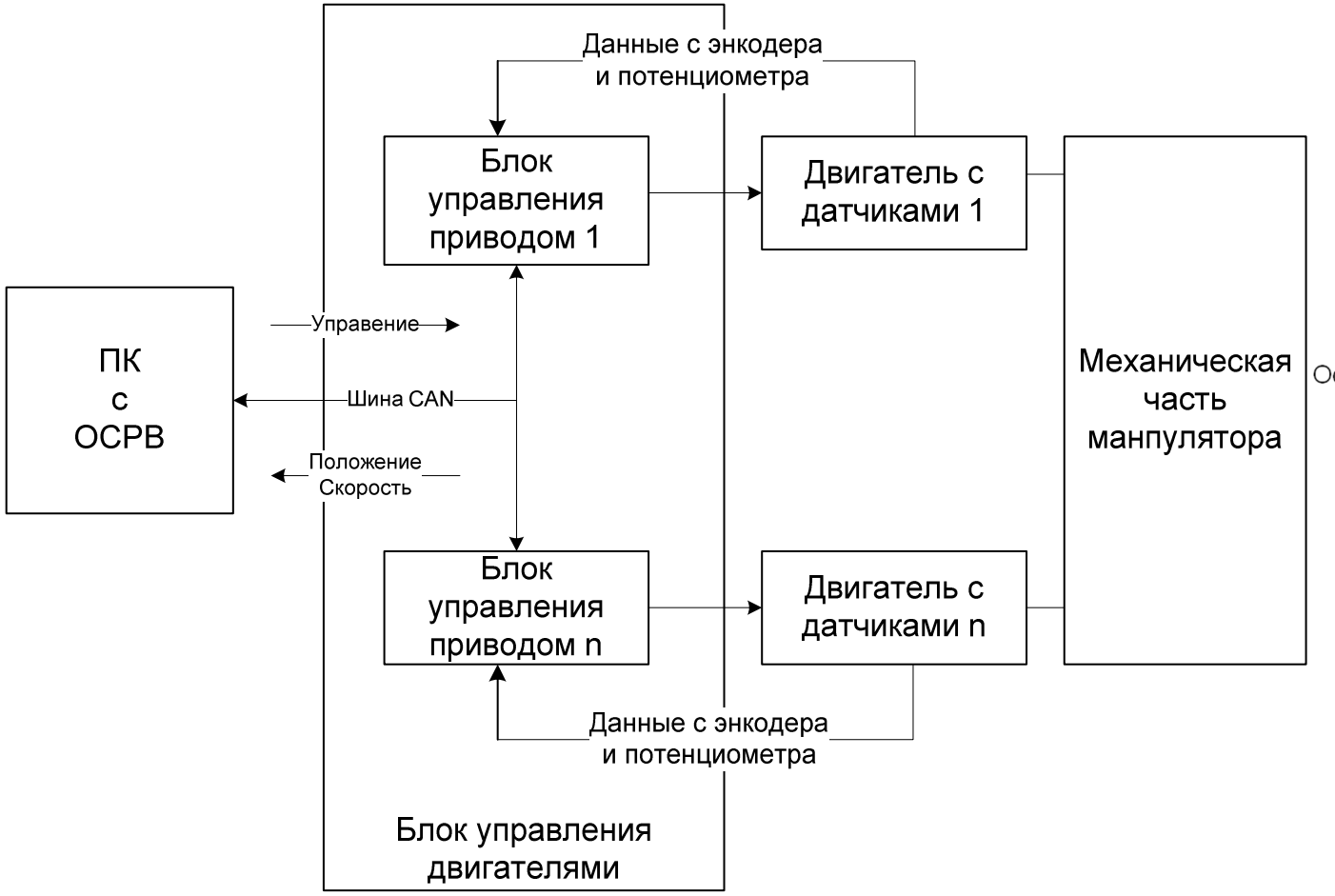
Прямое адаптивное управление роботом на основе сигмоидальной ИНС

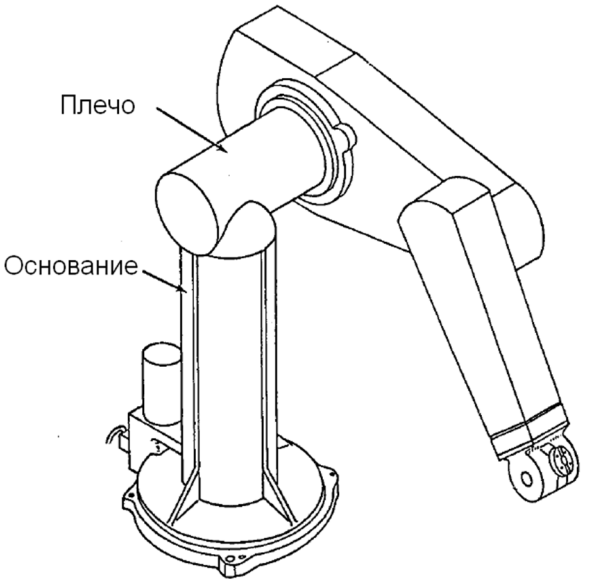


Результаты моделирования при управлении вторым звеном манипулятора: а)- пропорциональное управление без нейросети, б) - с нейросетью 300 нейронов в скрытом слое, в) - с нейросетью при вариации нагрузки в схвате, д) - с нейросетью в режиме постоянной скорости.

Реализация прямого адаптивного управление роботом на основе ИНС

Функциональная схема системы управления роботом

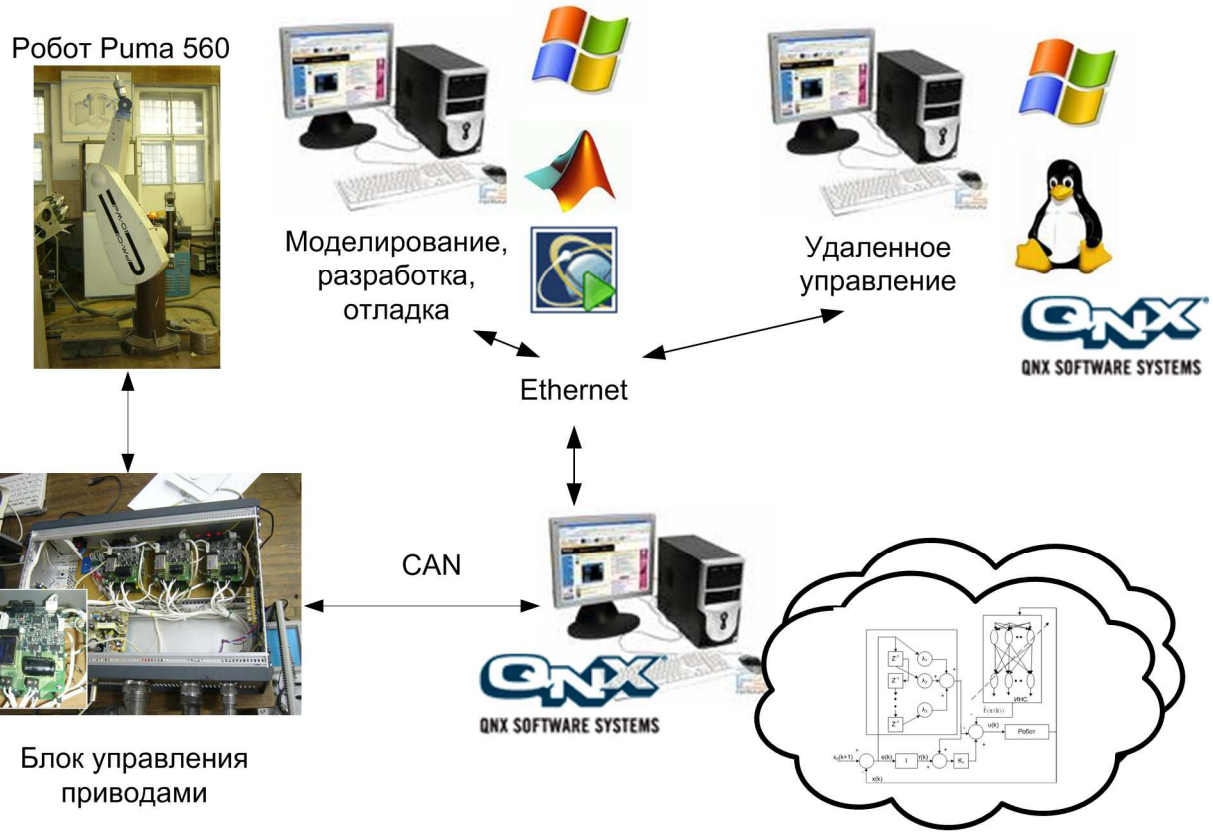




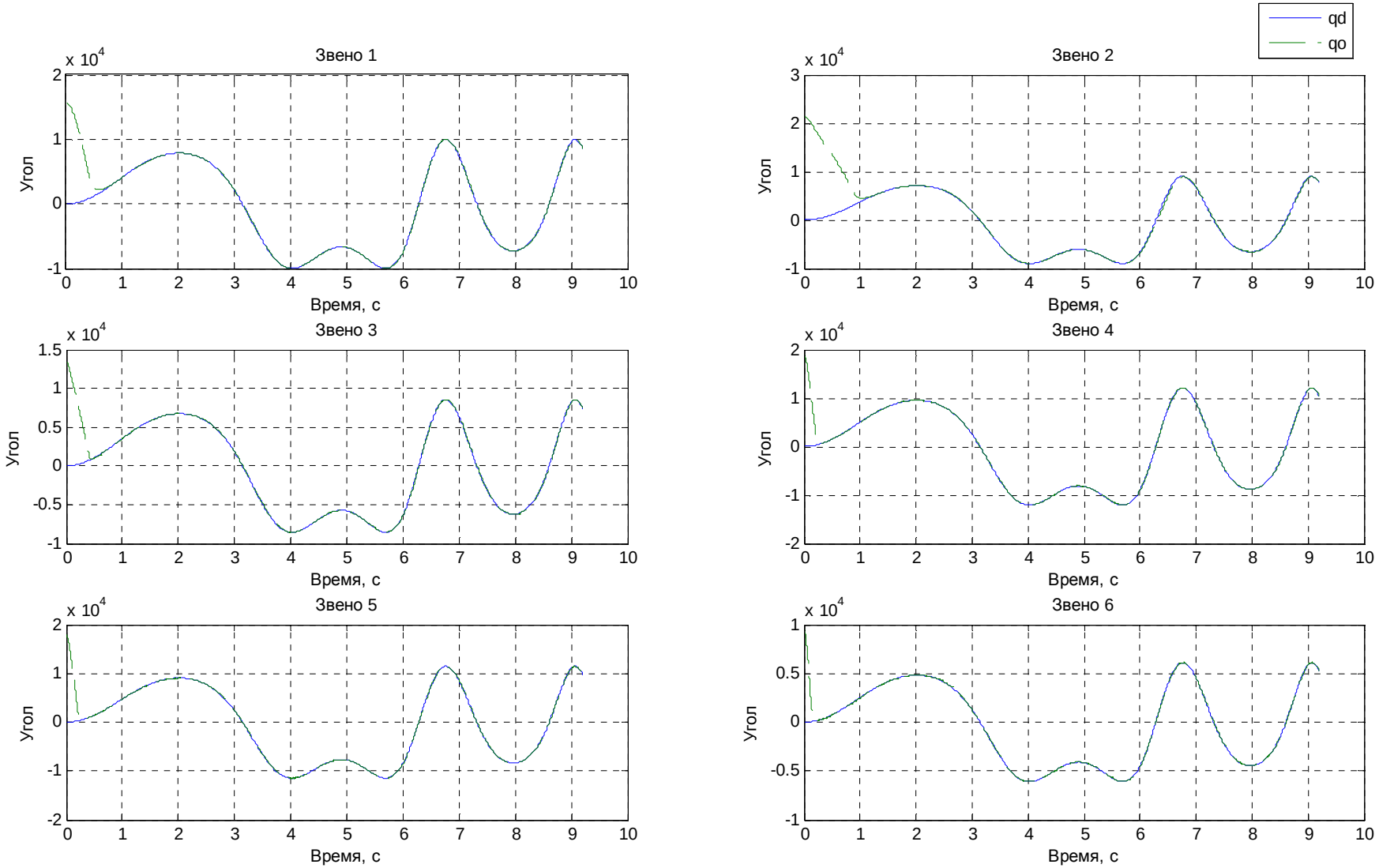
Puma 560

Реализация прямого адаптивного управление роботом на основе ИНС

- В качестве ОС управляющего ПК используется ОСРВ QNX Neutrino
- Произведена установка и настройка целевой системы QNX. Настроена отладка системы по сети Ethernet. Модифицирована программа работы модулей управления приводами
- Создана система обмена данными между управляющим ПК и модулями управления приводами, работающая с тактом опроса всех модулей в 1.5 мс



Переходные процессы при прямом адаптивном управление роботом на основе ИНС



Спасибо за внимание!

Ссылка на документ:
https://cloud.mail.ru/public/MdDu/SJcdZENGv