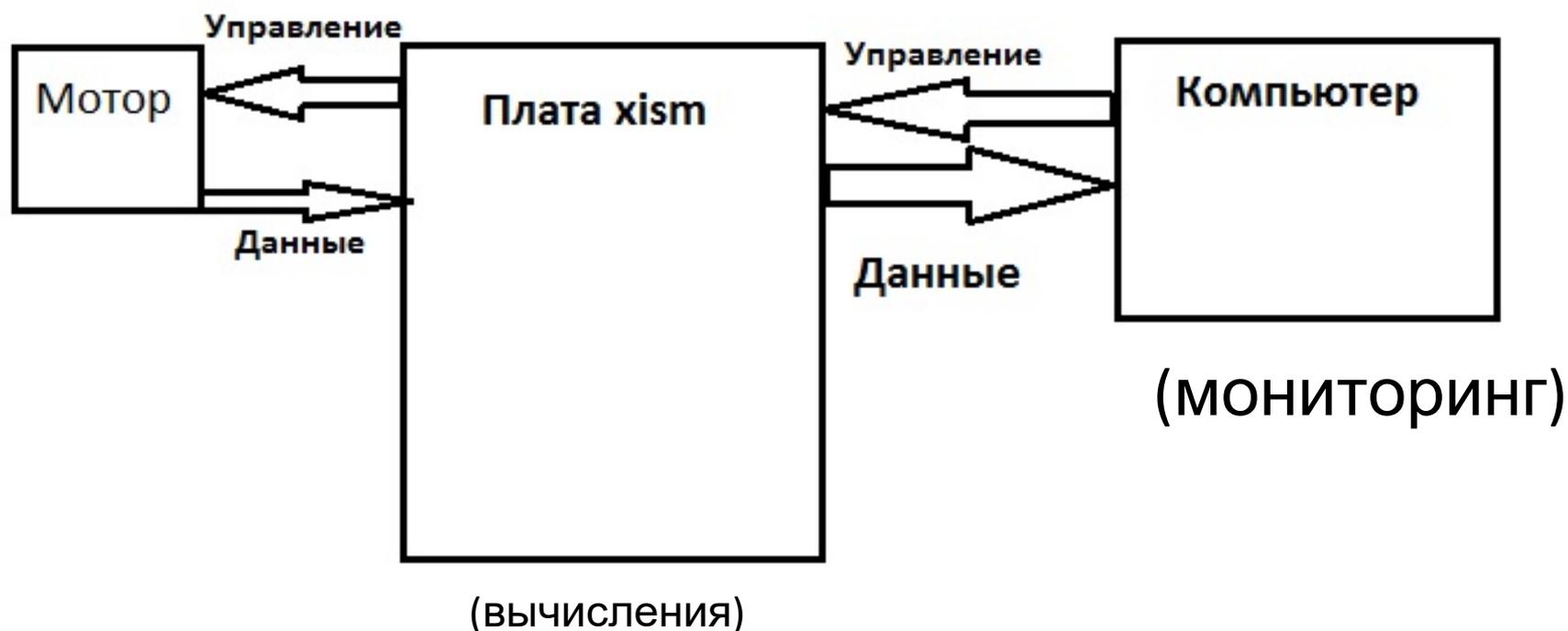


# Использование нейросетевых алгоритмов для регулирования скорости вращения двигателя постоянного тока

# Структура, внешний вид



# Структура проекта

## Микроконтроллер

Высокий уровень: нейропакет  
(в перспективе: генетические алгоритмы)



Прерывания: обработка данных  
с двигателя, отправка данных на  
компьютер

Основная программа: вычисления,  
отправка данных на компьютер



Низкий уровень: инициализация,  
ввод-вывод данных для  
двигателя и компьютера

## Компьютер

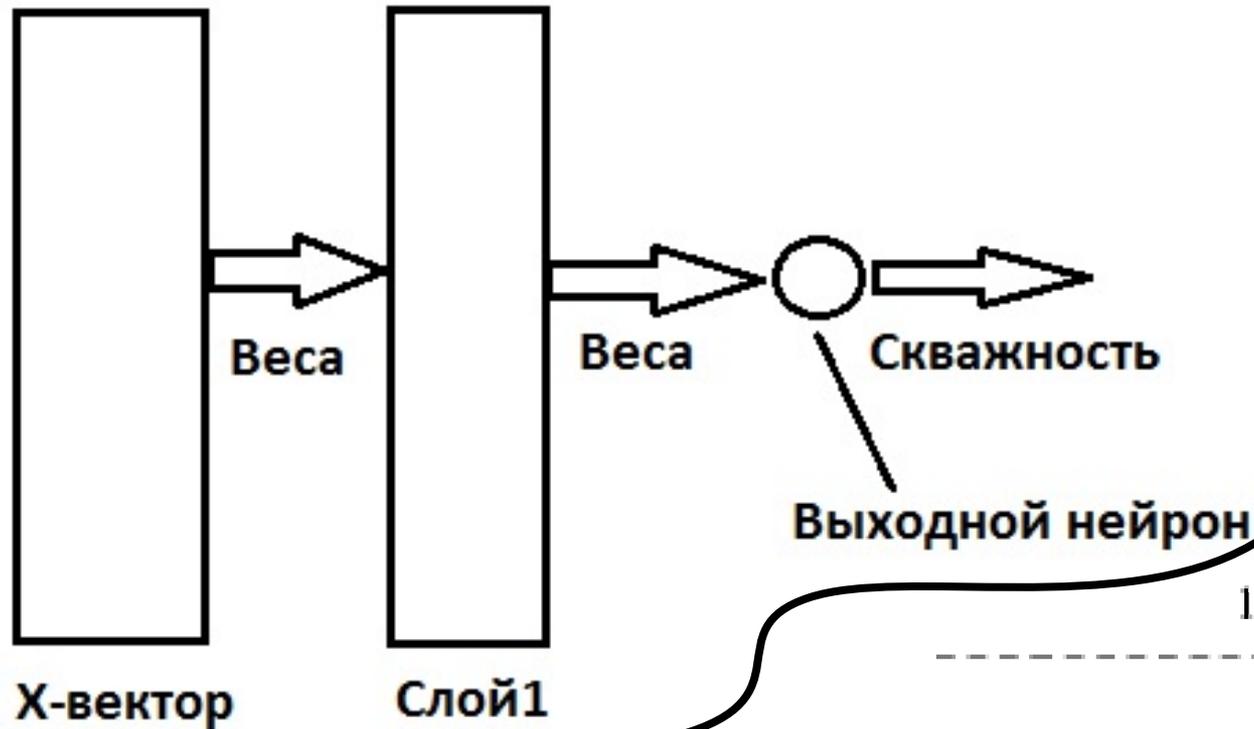
GUI

И\О, интерпретация

COM-порт

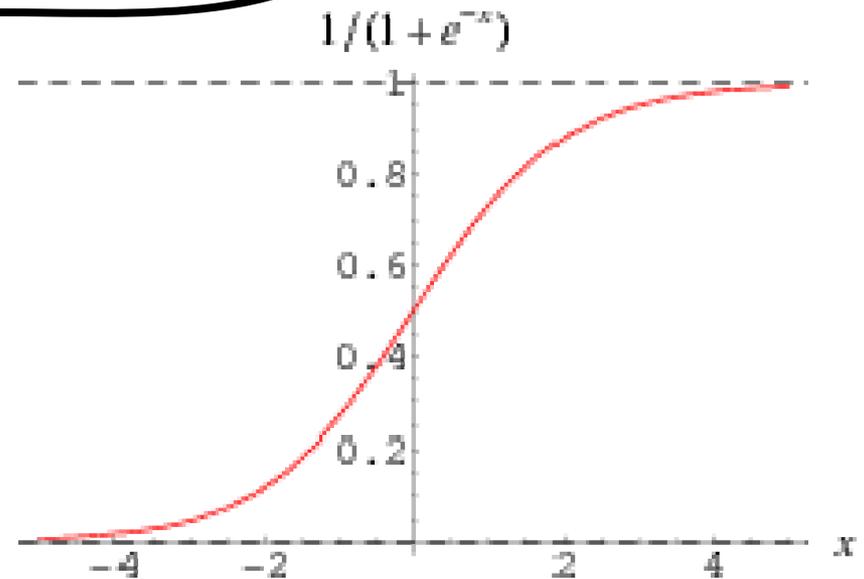
В перспективе:  
больше  
«параллельности»

# Структура нейронной сети (Изначальный вариант)

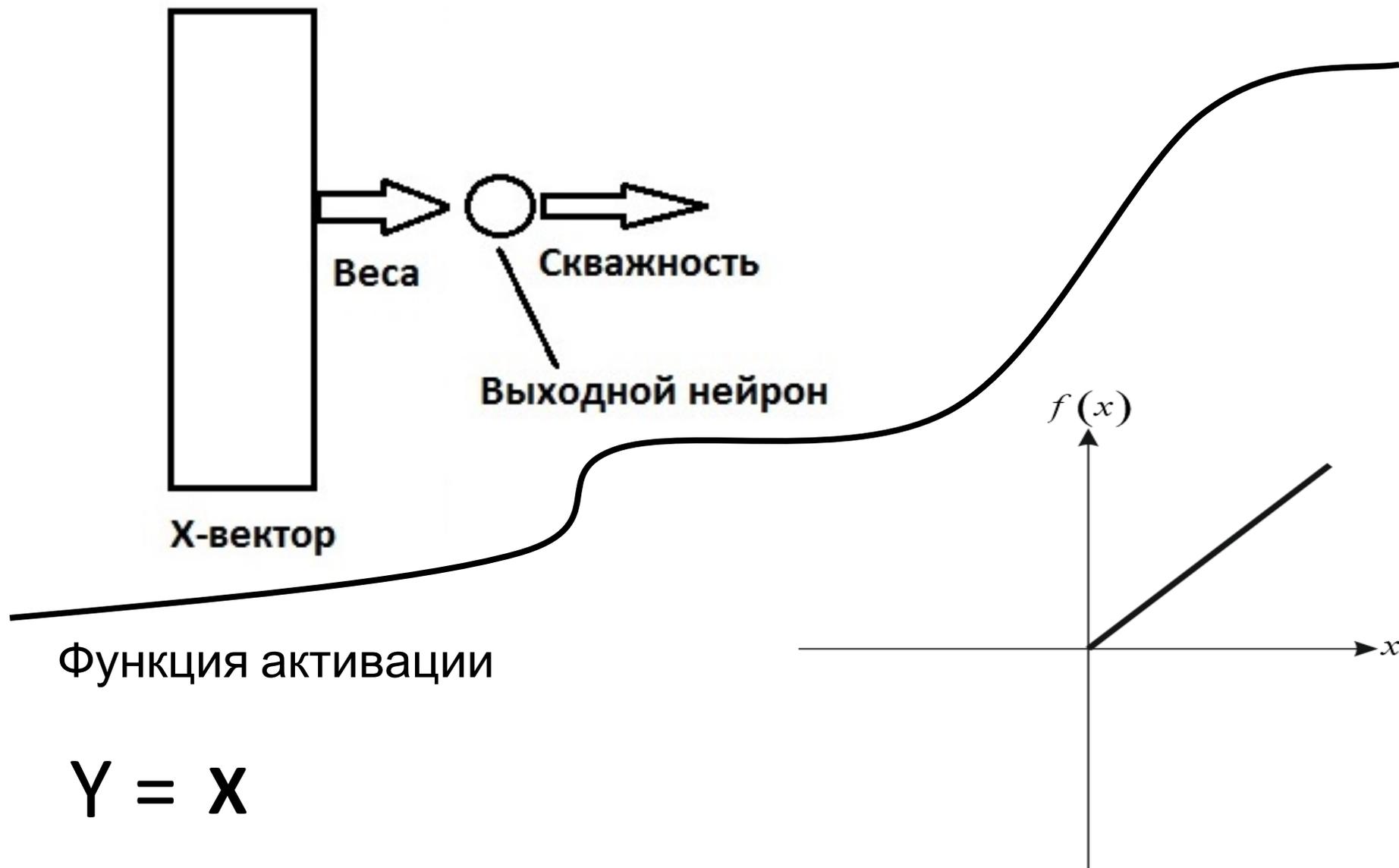


Функция активации

$$Y = \frac{1}{1 + e^{-X}}$$



# Структура нейронной сети (Действующий вариант)



# Алгоритмы обучения

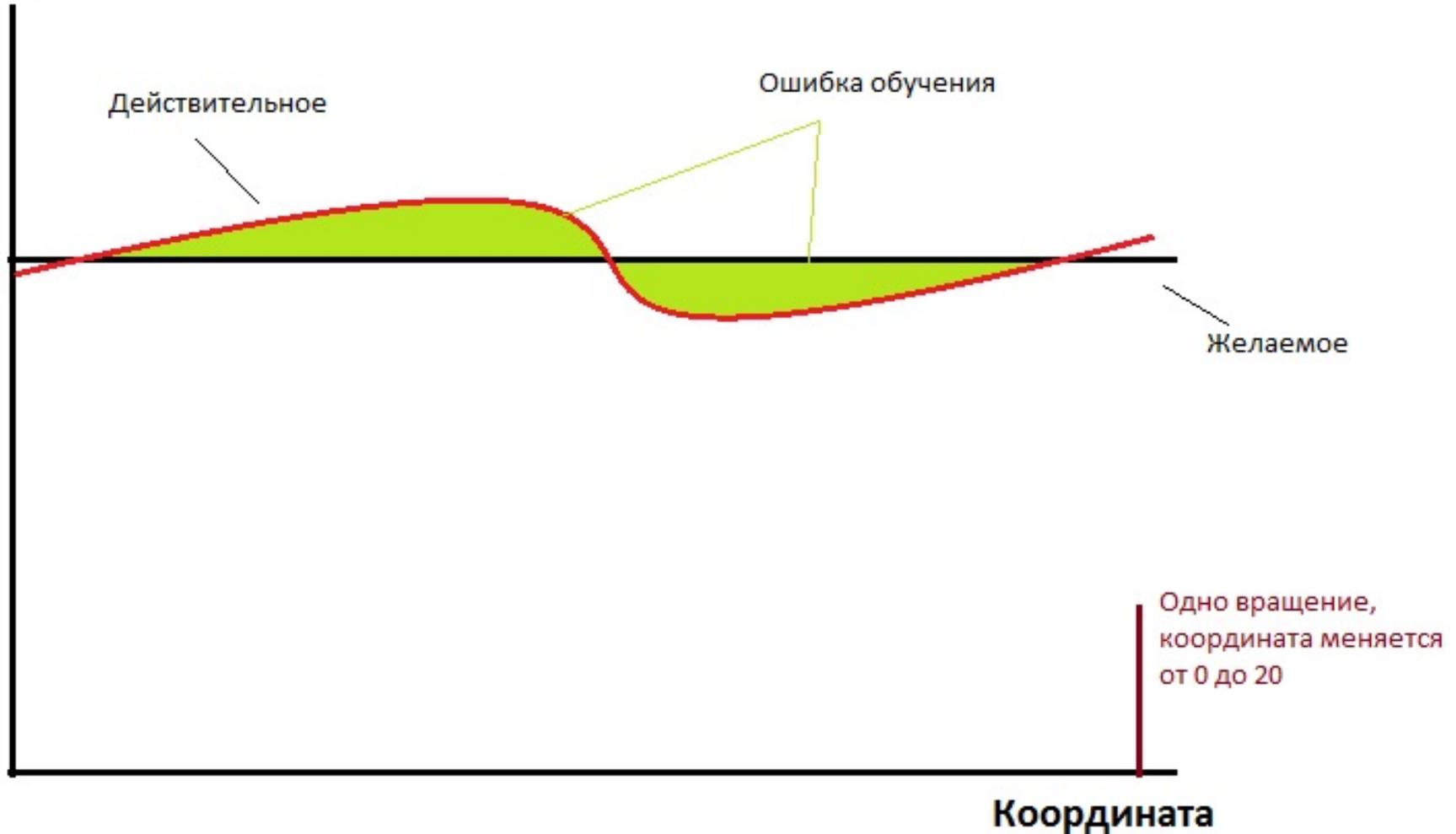
- Обратное распространение ошибки; в качестве градиента ошибки для выходного слоя – разница между ошибками для двух скважностей ШИМ при одном положении двигателя.
- Метод Недлера (ограничены вычислительные возможности)
- Генетические алгоритмы (в перспективе)

# Недостаток градиентного метода

- Размерность пространства ошибки (без использования алгоритма обратного распространения)  $\sim 20 \times 20 = 400$  (!)
- Целевая функция вычисляется экспериментально, при вращении 1Гц – одна секунда
- $400 * 1\text{с} = 400\text{с} \sim 6$  минут

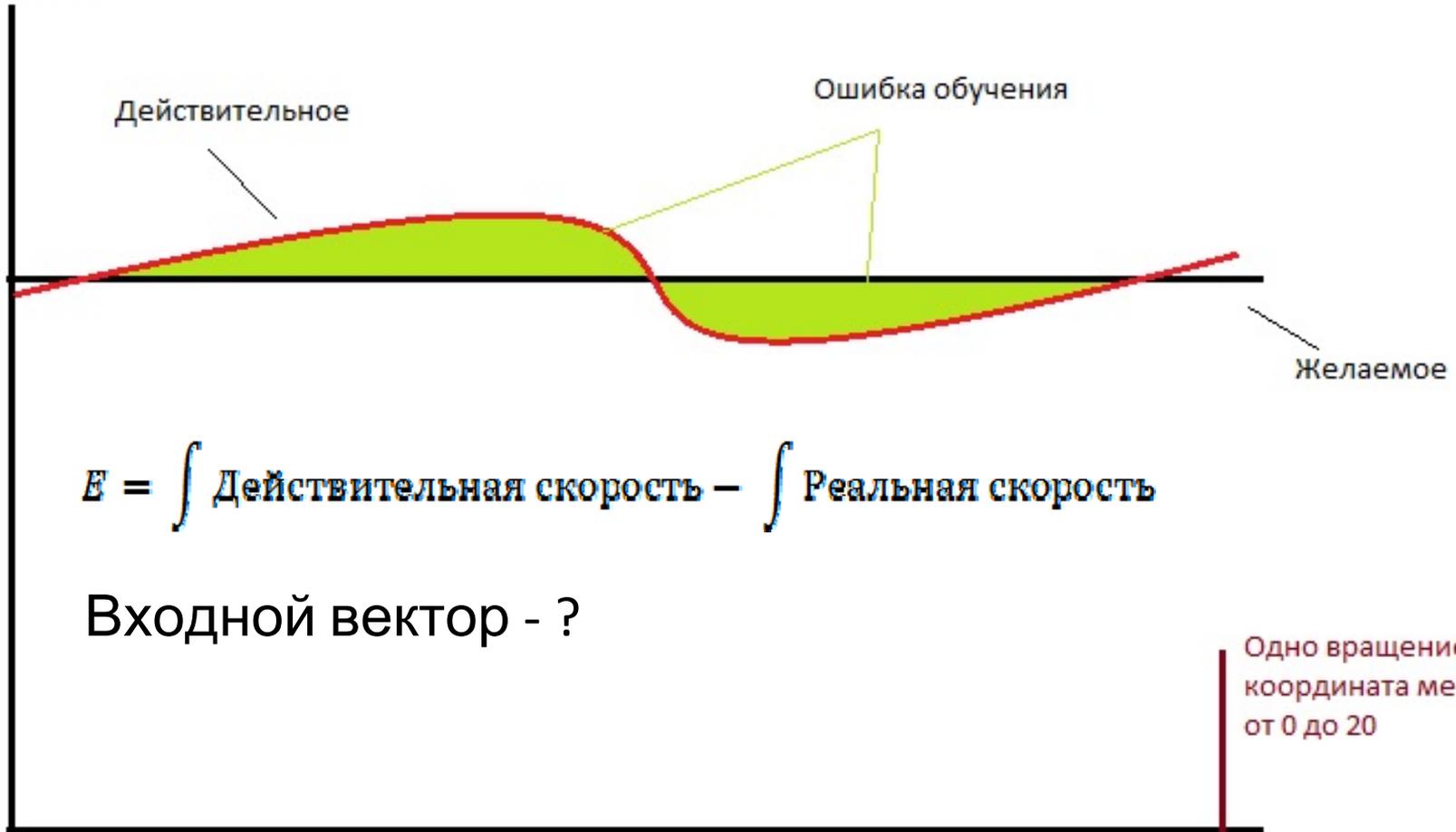
# Обратное распространение ошибки

Скорость



# Обратное распространение ошибки

Скорость



Действительное

Ошибка обучения

Желаемое

$$E = \int \text{Действительная скорость} - \int \text{Реальная скорость}$$

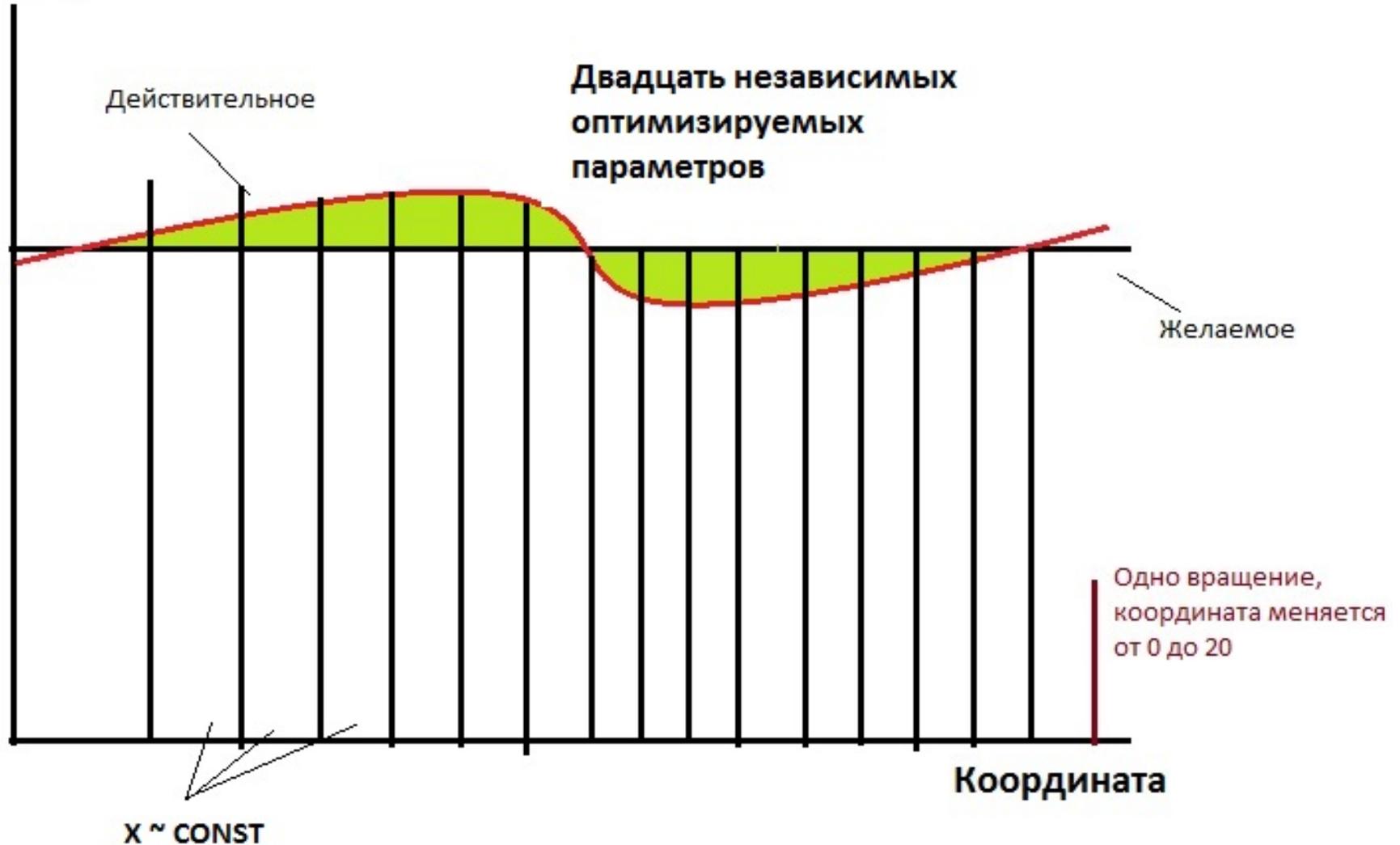
Входной вектор - ?

Одно вращение,  
координата меняется  
от 0 до 20

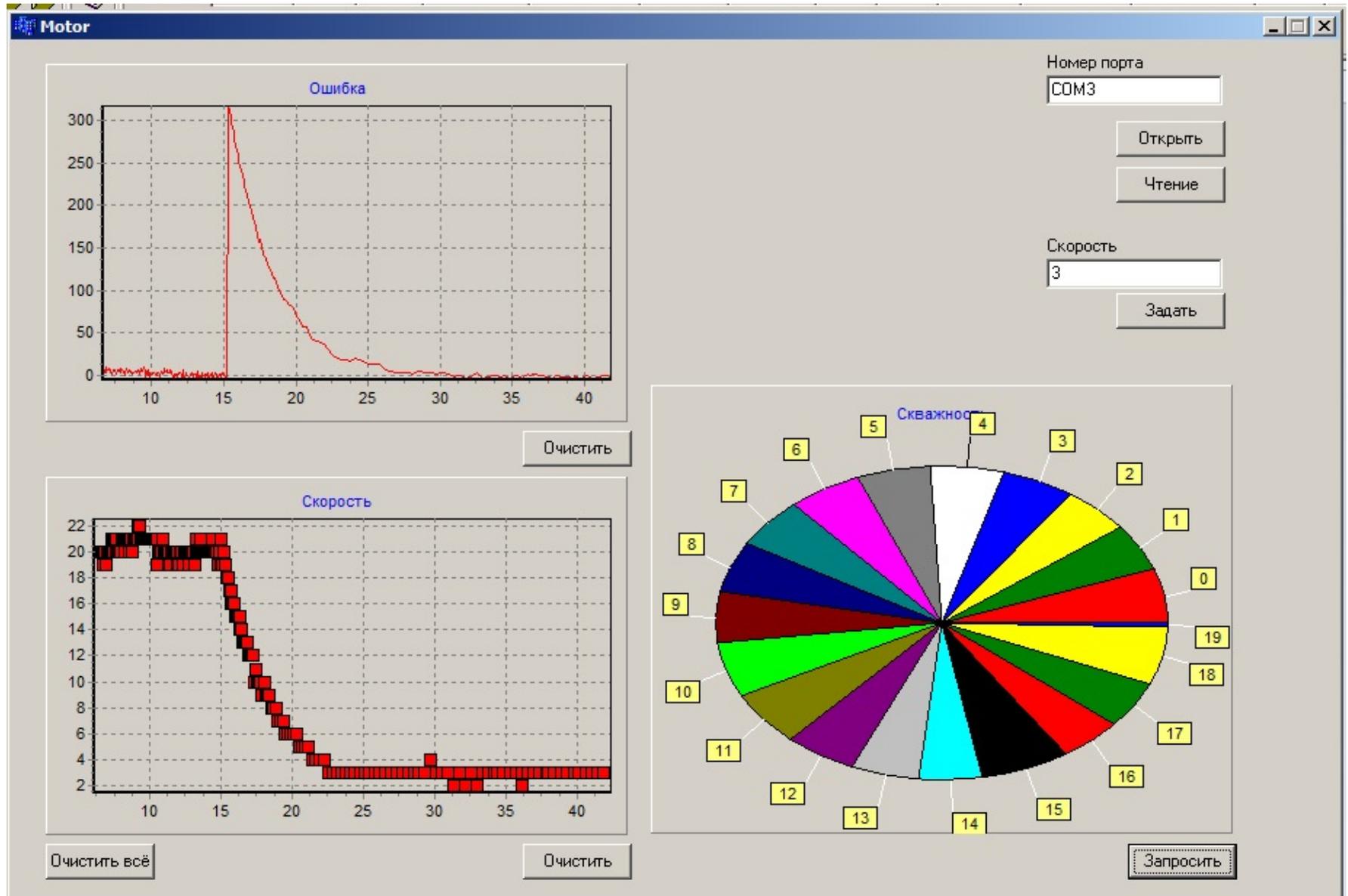
Координата

# Обратное распространение ошибки

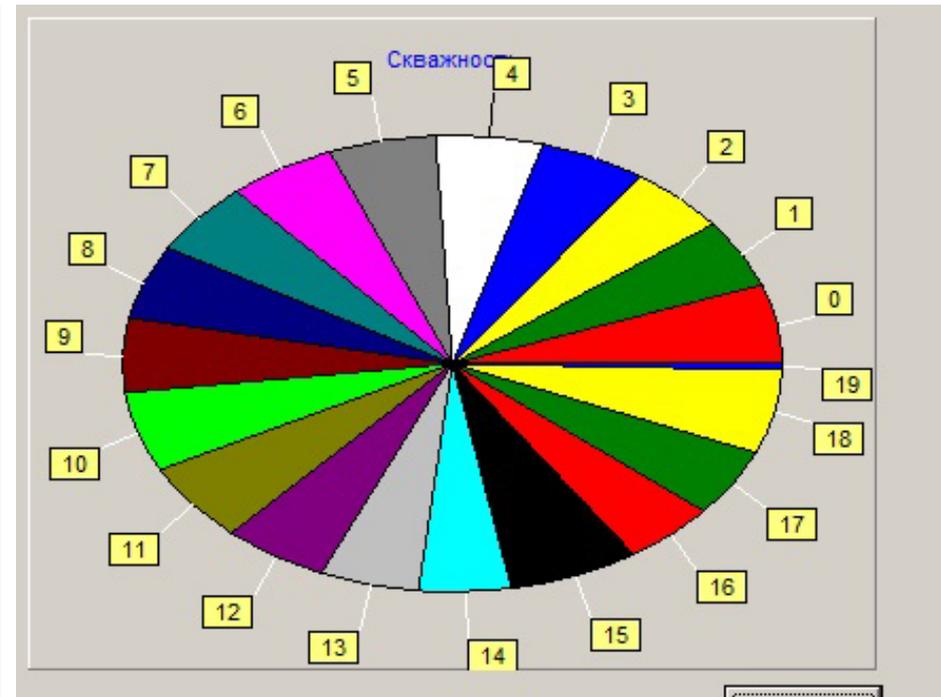
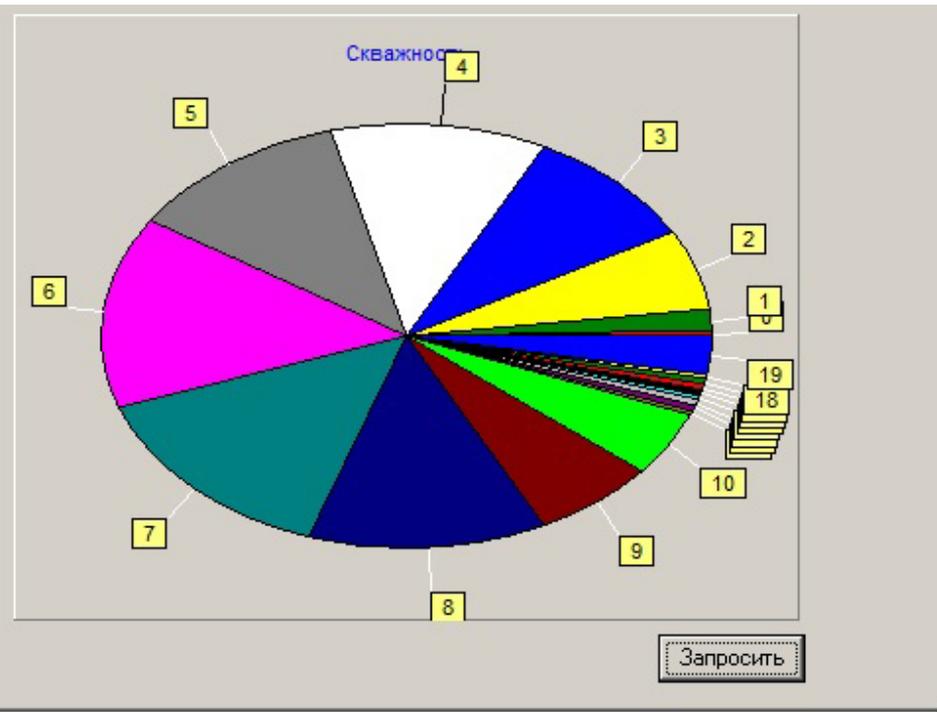
Скорость



# Результаты эксперимента



# Результаты эксперимента

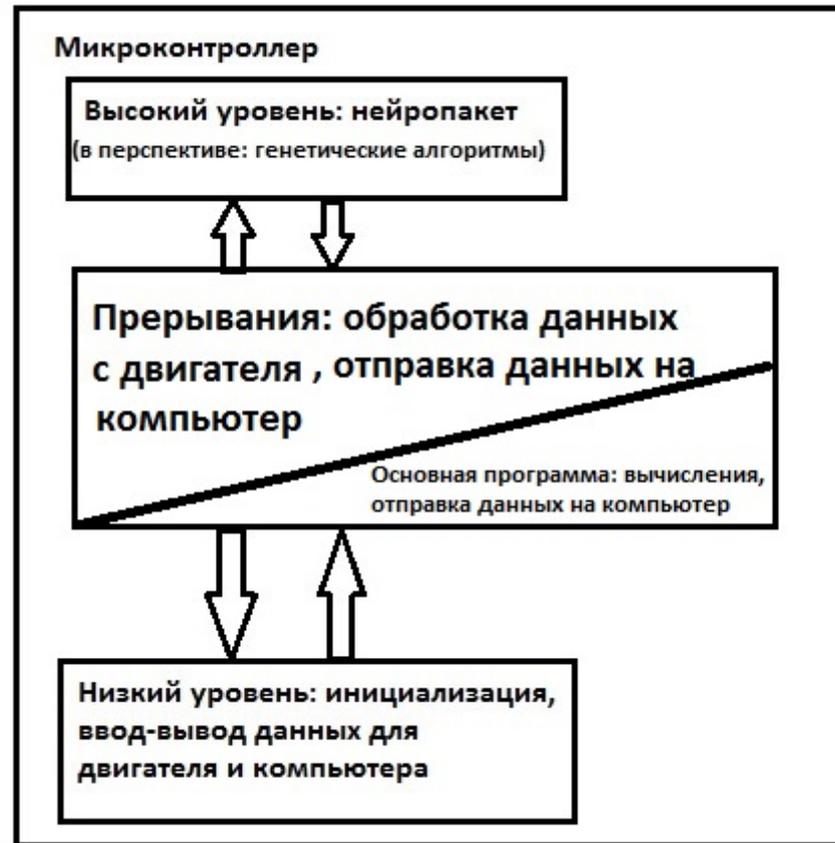


**Слева: на двигатель повешен груз, справа – нормальный ротор**

Простейшая нейронная сеть подбирает скважность в зависимости от положения двигателя

# Результаты работы

- Реализованы эти модули
- Есть оптимизация



# Дальнейшие совершенствования

- Больше алгоритмов
- Вынесение вычислительной работы в отдельный модуль (на ПЛИС?)
- Устранение багов (глюки программы для компьютера, при передаче данных; баги на низком уровне, в прошивке микроконтроллера)
- Произвольная функция вращения

```

TNeuroValue BackPropIter(TNeuroValue *X, TNeuroValue *Y)
{
    int i,j = 0;

    TNeuroValue b = 0; //коэф. "b" (см. учебник)
    TNeuroValue bl[LAY_SIZE] = {0};

    Calc(X);

    b = Grad*fActp(lS);

    for(i=0;i<LAY_SIZE;i++) //и сместит
    {
        bl[i] = b*out_w[i]*fActp(S[i]);

        out_w[i] = TEACH_CONST*bl[i]*X[i]; //и сместит
    }

    for(i=0;i<LAY_SIZE;i++)
    {
        for(j=0;j<X_SIZE;j++)
        {
            inter_w[i][j] += -TEACH_CONST*bl[i]*X[j];
        }
    }
}

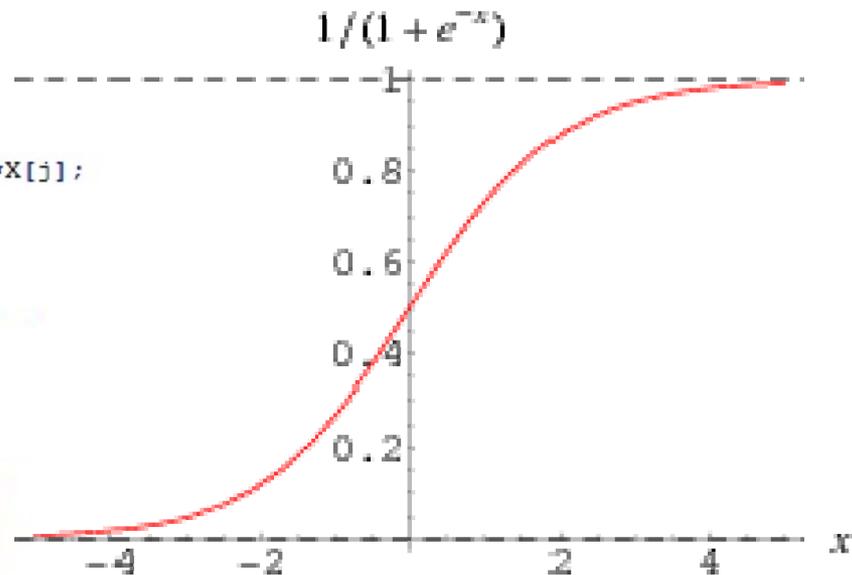
```

```

TNeuroValue out_w[LAY_SIZE]; //массив весов
TNeuroValue inter_w[LAY_SIZE][X_SIZE]; //массив интервальных весов
TNeuroValue S[LAY_SIZE] = {0}; //массив сумм
TNeuroValue Sv[LAY_SIZE] = {0}; //массив сумм
TNeuroValue lS = 0; //сумма потерь

```

**Спасибо за внимание!**



```

ralEnable(SYSCTL_PERIPH_PWM);
ralEnable(SYSCTL_PERIPH_GPIOA);
ralEnable(SYSCTL_PERIPH_GPIOF);

```