Intelligent Control Systems
(0640734)

Lecture (5)

Neural Networks in Control Systems

Prof. Kasim M. Al-Aubidy
Philadelphia University-Jordan

Introduction:

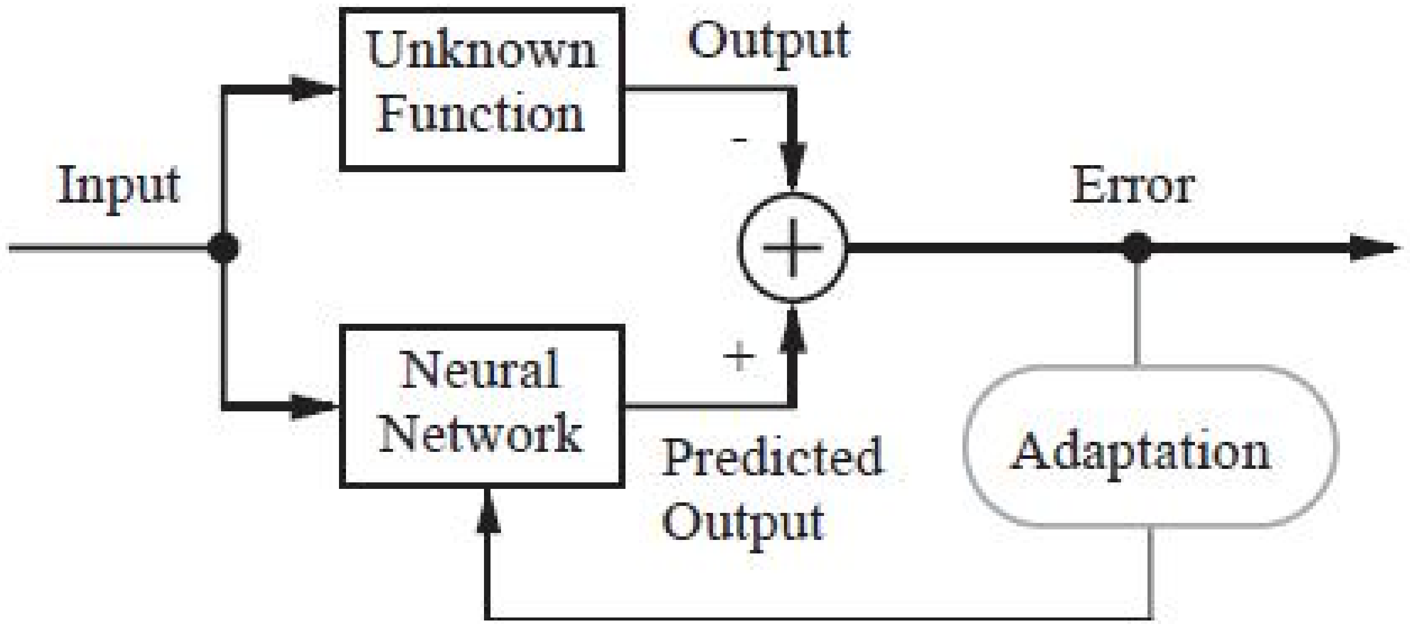
- The use of NN in control systems was first proposed by Werbos [1989] and Narendra [1990].
- NN control has had two major objectives:
  - Approximate Dynamic Programming, which uses NN to approximately solve the optimal control problem, and
  - NN in closed-loop feedback control.
- The challenge in using NN for feedback control purposes is to select a suitable control system structure, and then to demonstrate using mathematically acceptable techniques how the NN weights can be tuned so that closed-loop stability and performance are guaranteed.

Why Neural Control?

- When mathematical models of the plant dynamics are not available, NNs can provide a useful method for designing controllers, provided we have numerical information about the system behavior in the form of input-output data.
- NN can be used as a "black box" model for a plant.
- Controllers based on NNs will benefit from NNs. learning capability that is suitable for adaptive control where controllers need to adapt to changing environment.
- NN controllers have proved to be most useful for time-invariant systems.
- To build a NN based controller that can force a plant to behave in some desirable way, we need to adjust its parameters from the observed errors.
- Adjustment of the controller's parameters will be done by propagating back these errors across the NN structure. This is possible if the mathematical model of the plant is known.
- NN identified models are used in indirect neural control designs.
- NNs are massive parallel computation structures. This allows calculations to be performed at a high speed, making real-time implementations feasible.
- NNs can also provide significant fault tolerance, since damage to a few weights need not significantly impair the overall performance.

Function Approximator:

- Neural Network is used as a Function Approximator.
- We have some unknown function that we wish to approximate.
- We want to adjust the parameters of the network so that it will produce the same response as the unknown function, if the same input is applied to both systems.



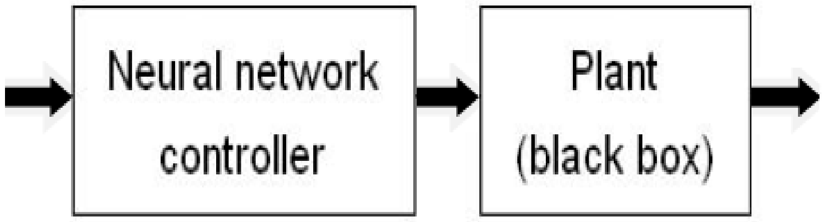
Neural Network Control Topologies:

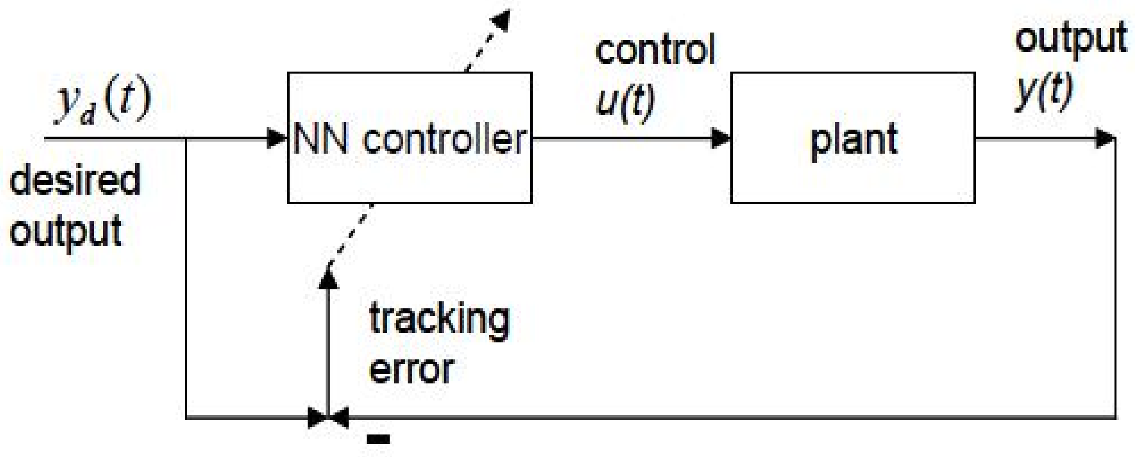
There are two design methods for neural control, these are;

- **Direct Method:** in which the controller itself is a neural network.
- **Indirect Method:** in which controllers are designed based on a neural network model of the plant.

1. Direct Method:

Direct design means that a neural network directly implements the controller (Neural network).
The network must be trained according to some criteria, using either numerical I/O data or a mathematical model of the system.



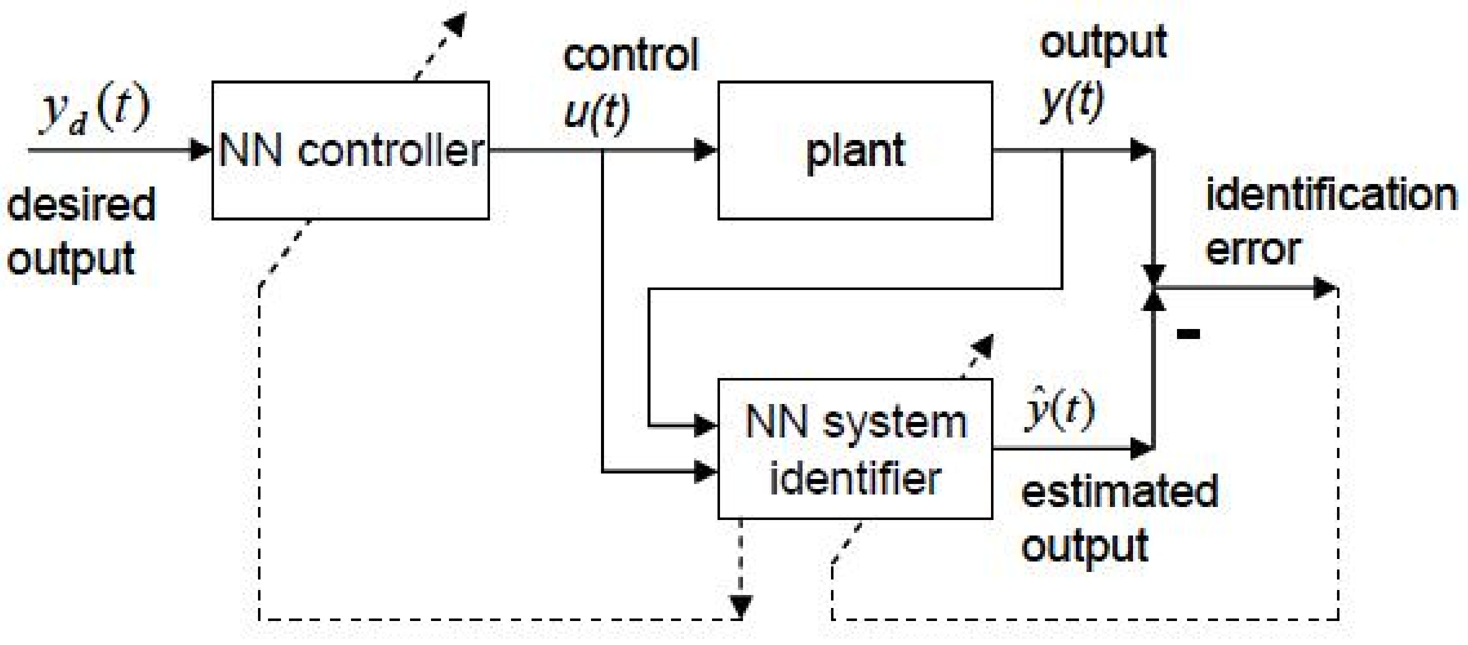


Indirect NN Control:

Indirect neural control design is based on a neural network model of the system to be controlled. In this case, the controller itself may not be a neural network, but it is derived from a plant that is modeled by a neural network.

There are two phases;

- NN System Identifier Unit: the NN is tuned to learn the dynamics of the unknown plant, and
- NN Controller Unit: the NN uses this information to control the plant.



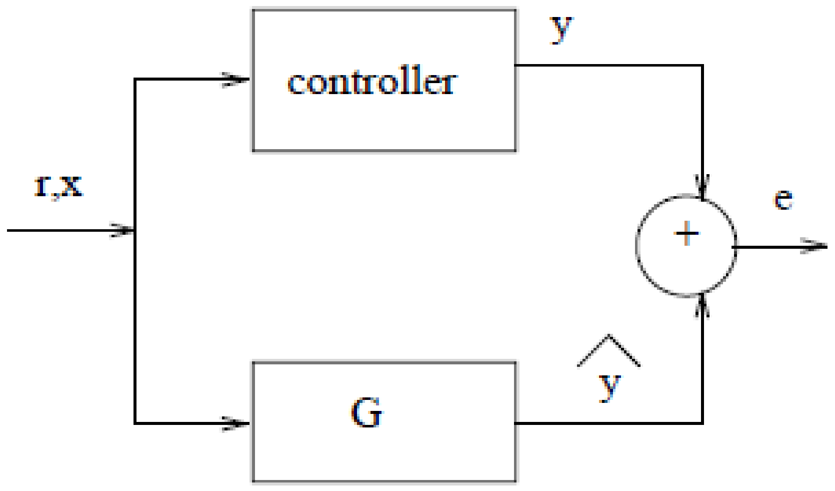
Neural Control:

Control methods utilize NNs in conjunction with classical controllers;

- Control by emulation.
- Inverse and open-loop control.
- Feedback control.

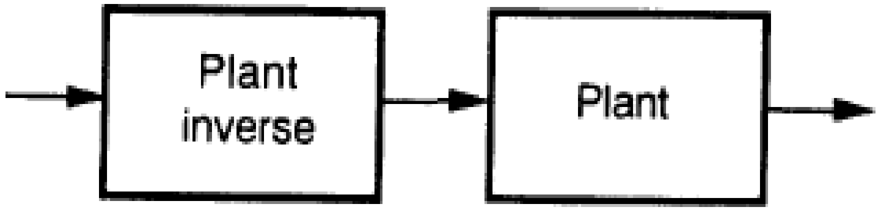
1. Expert Emulator:

If a controller exists, we can emulate it.
There is no feedback, and we can train NN
With standard PB.
This is useful if the controller is a human and
We want to automate the process.



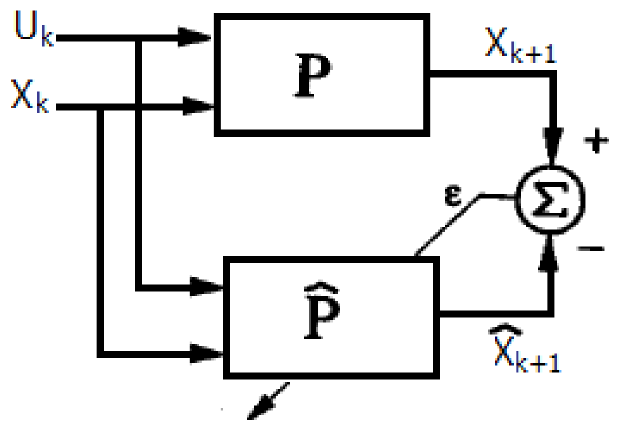
2. Open Loop/ Inverse Control:

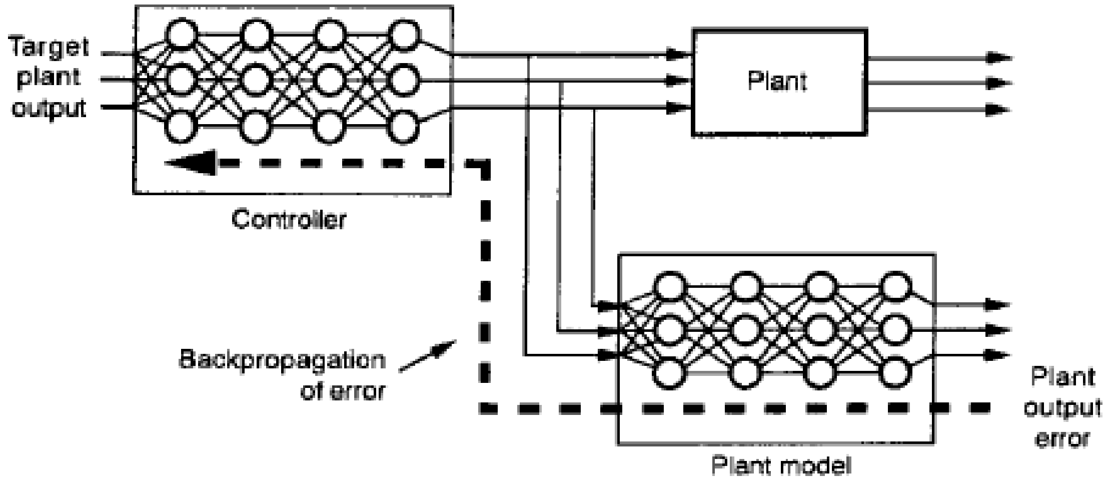
Early systems were designed open loop making the neural controller approximately the inverse of the plant to be controlled.



Note:

It is useful to build a plant emulator, then train the Neural Controller parameters as shown bellow.

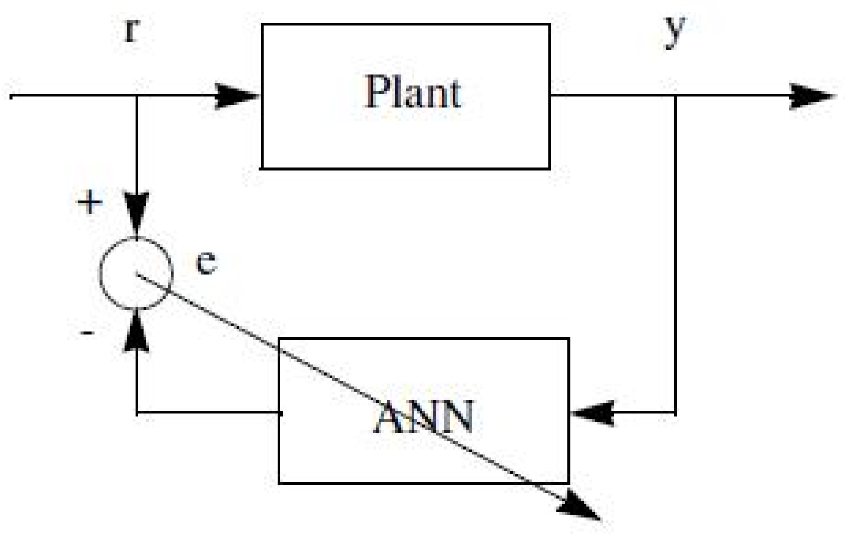




NN Training:

There are several possibilities for training the NN, depending on the architecture used for learning;

1. **Generalized learning architecture:** The error (e) is used to adjust the neural network parameters. NN training involves supplying different inputs to the plant and teaching the network to map the corresponding outputs back to the plant inputs.

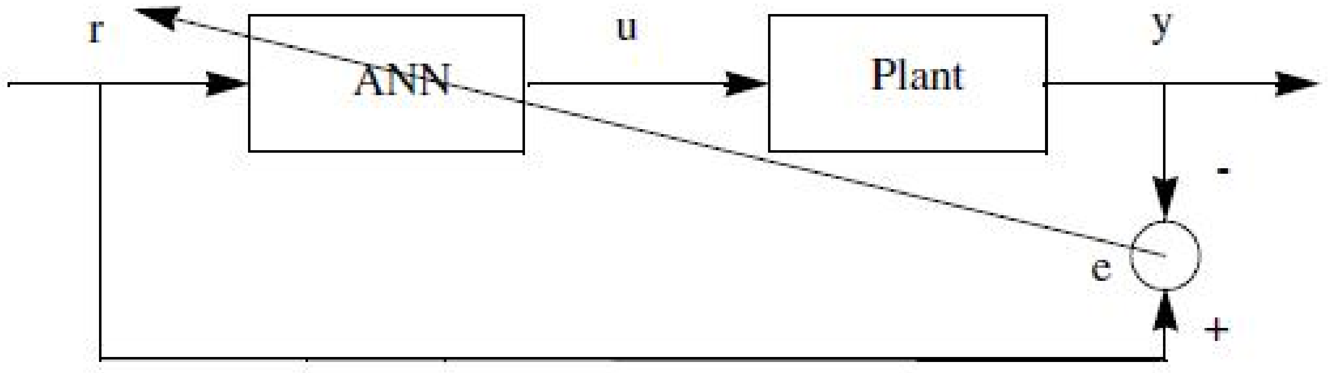


Drawbacks:

- It works well but has the drawback of not training the network over the range where it will operate, after having been trained.
- The NN may have to be trained over an extended operational range.
- It is impossible to train the network on-line.

2. Specialized learning architecture:

It is used to overcome the disadvantages of the generalized learning approach. The NN learns how to find the inputs (u) that drive the system outputs (y) towards the reference signal (r). The NN is trained in the region where it will operate, and can also be trained on-line.



Drawbacks:

The problem with this architecture lies in the fact that although the error (e= r-y) is available, there is no explicit target for the control input (u) to be applied in the training of the NN.

To overcome these difficulties, research paper [] proposed either replacing the Jacobian elements by their sign, or performing a linear identification of the plant by an adaptive least-squares algorithm.

Inverse dynamics:

An ideal control law describes the inverse dynamics of a plant.

Consider, the time-invariant linear system described in discrete time by the equations

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k), \; k = 0, 1, 2, \ldots$$

where A is an $n \times n$ matrix and B is $n \times 1$. Then we have

$$\begin{aligned}
\mathbf{x}(k+2) &= A\mathbf{x}(k+1) + B\mathbf{u}(k+1) \\
&= A(A\mathbf{x}(k) + B\mathbf{u}(k)) + B\mathbf{u}(k+1) \\
&= A^2\mathbf{x}(k) + AB\mathbf{u}(k) + B\mathbf{u}(k+1) \\
\mathbf{x}(k+3) &= A\mathbf{x}(k+2) + B\mathbf{u}(k+2) \\
&= A\left(A^2\mathbf{x}(k) + AB\mathbf{u}(k) + B\mathbf{u}(k+1)\right) + B\mathbf{u}(k+2) \\
&= A^3\mathbf{x}(k) + A^2B\mathbf{u}(k) + AB\mathbf{u}(k+1) + B\mathbf{u}(k+2)
\end{aligned}$$

$$\begin{aligned}\mathbf{x}(k+4) &= A\mathbf{x}(k+3) + B\mathbf{u}(k+3) \\ &= A\left(A^3\mathbf{x}(k) + A^2B\mathbf{u}(k) + AB\mathbf{u}(k+1) + B\mathbf{u}(k+2)\right) + B\mathbf{u}(k+3) \\ &= A^4\mathbf{x}(k) + A^3B\mathbf{u}(k) + A^2B\mathbf{u}(k+1) + AB\mathbf{u}(k+2) + B\mathbf{u}(k+3)\end{aligned}$$

$$\boxed{\begin{aligned}\mathbf{x}(k+n) &= A^n\mathbf{x}(k) + A^{n-1}Bu(k) + \cdots + ABu(k+n-2) + Bu(k+n-1) \\ &= A^n\mathbf{x}(k) + WU\end{aligned}}$$

where $W = \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & A^2B & AB & B \end{bmatrix}$

$U = \begin{bmatrix} u(k) & u(k+1) & \cdots & u(k+n-2) & u(k+n-1) \end{bmatrix}^T$

If the matrix W is nonsingular, we can compute U directly from W and $\mathbf{x}$:

$$\boxed{\begin{aligned}U &= W^{-1}\left(\mathbf{x}(k+n) - A^n\mathbf{x}(k)\right) \\ &= \varphi\left(\mathbf{x}(k), \mathbf{x}(k+n)\right)\end{aligned}}$$

This is a control law for the system that is derived directly from the plant dynamics by computing the inverse dynamics.

Example *Take* $A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -5 & -2 & -3 \end{bmatrix}$ *and* $B = \begin{bmatrix} 0 \\ 0 \\ 9 \end{bmatrix}$. *Then*

$$W = \begin{bmatrix} A^2B & AB & B \end{bmatrix} = \begin{bmatrix} 9 & 0 & 0 \\ -27 & 9 & 0 \\ 63 & -27 & 9 \end{bmatrix}$$

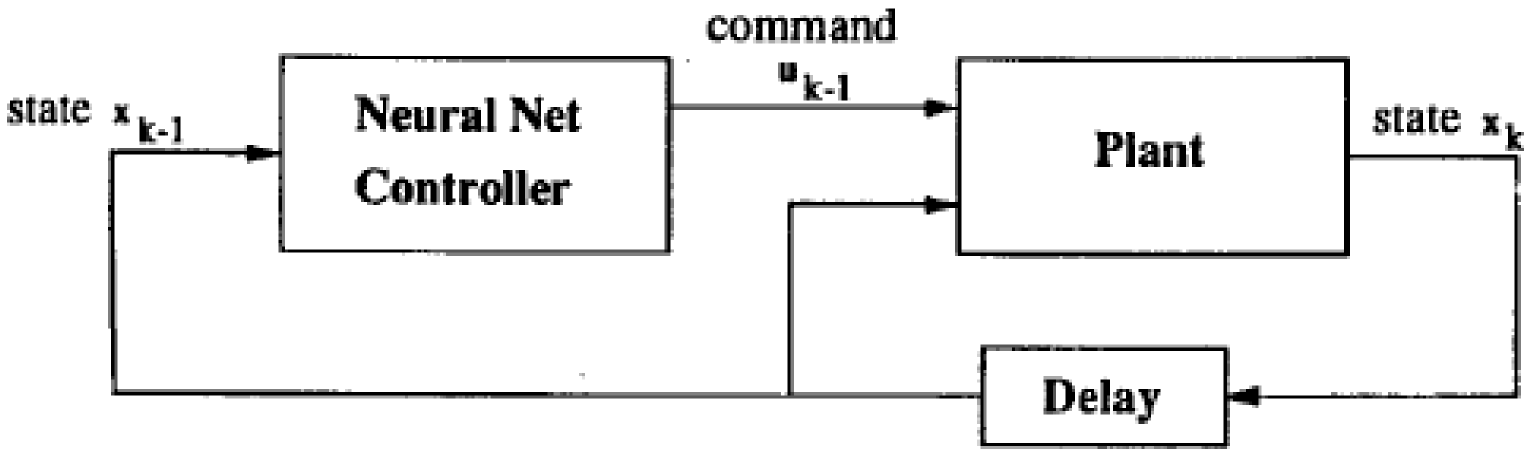
and

$$\begin{aligned} U &= \begin{bmatrix} \frac{1}{9} & 0 & 0 \\ \frac{1}{3} & \frac{1}{9} & 0 \\ \frac{2}{9} & \frac{1}{3} & \frac{1}{9} \end{bmatrix} \left(\mathbf{x}(k+3) - \begin{bmatrix} -5 & -2 & -3 \\ 15 & 1 & 7 \\ -35 & 1 & -20 \end{bmatrix} \mathbf{x}(k) \right) \\ &= \begin{bmatrix} \frac{1}{9} & 0 & 0 \\ \frac{1}{3} & \frac{1}{9} & 0 \\ \frac{2}{9} & \frac{1}{3} & \frac{1}{9} \end{bmatrix} \mathbf{x}(k+3) - \begin{bmatrix} -\frac{5}{9} & -\frac{2}{9} & -\frac{1}{3} \\ 0 & -\frac{5}{9} & -\frac{2}{9} \\ 0 & 0 & -\frac{5}{9} \end{bmatrix} \mathbf{x}(k) \\ &= \begin{bmatrix} \frac{1}{9}x_1(k+3) + \frac{5}{9}x_1(k) + \frac{2}{9}x_2(k) + \frac{1}{3}x_3(k) \\ \frac{1}{3}x_1(k+3) + \frac{1}{9}x_2(k+3) + \frac{5}{9}x_2(k) + \frac{2}{9}x_3(k) \\ \frac{2}{9}x_1(k+3) + \frac{1}{3}x_2(k+3n) + \frac{1}{9}x_3(k+3) + \frac{5}{9}x_3(k) \end{bmatrix} \\ &= \varphi(\mathbf{x}(k), \mathbf{x}(k+n)) \end{aligned}$$

When the inverse dynamics of a plant exist, you can control the plant by modeling its inverse dynamics.

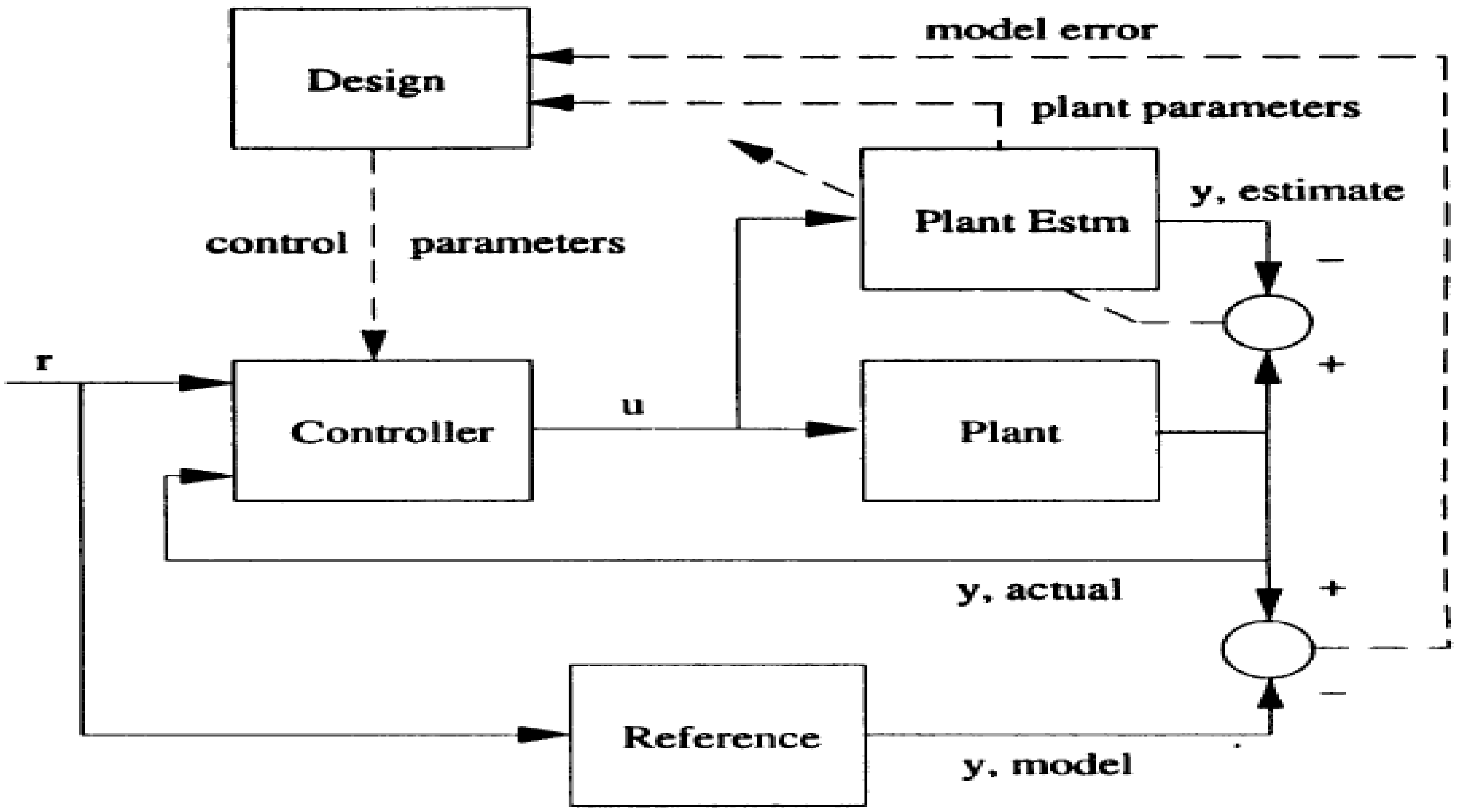
3. Classical Feedback Control:

These methods use neural nets to improve dynamics in conjunction with classical methods. The feedback is ignored during designing the controller.



4. Neural Feedback Control:

Incorporating feedback, neural system can be trained with similar objectives and cost functions, as classical control but solution are iterative and approximate. Optimal control techniques can be applied which are constrained to be approximate by nature of training and NN architectural constraints.



CONTROL SYSTEM APPLICATIONS:

Ref: M.T. HAGAN, H.B. DEMUTH & O. DE JESÚS, "AN INTRODUCTION TO THE USE OF NEURAL NETWORKS IN CONTROL SYSTEMS", Available online on net.

This paper outlines a brief introduction to the use of neural networks in control systems.

MLP NNs have been applied successfully in the identification and control of dynamic systems.

There are three well-known typical neural network controllers:

- Model Predictive Control,
- Feedback Linearization Control, and
- Model Reference Control.

There are two steps involved when using NNs for control:

System Identification: develop a neural network model of the plant that we want to control.

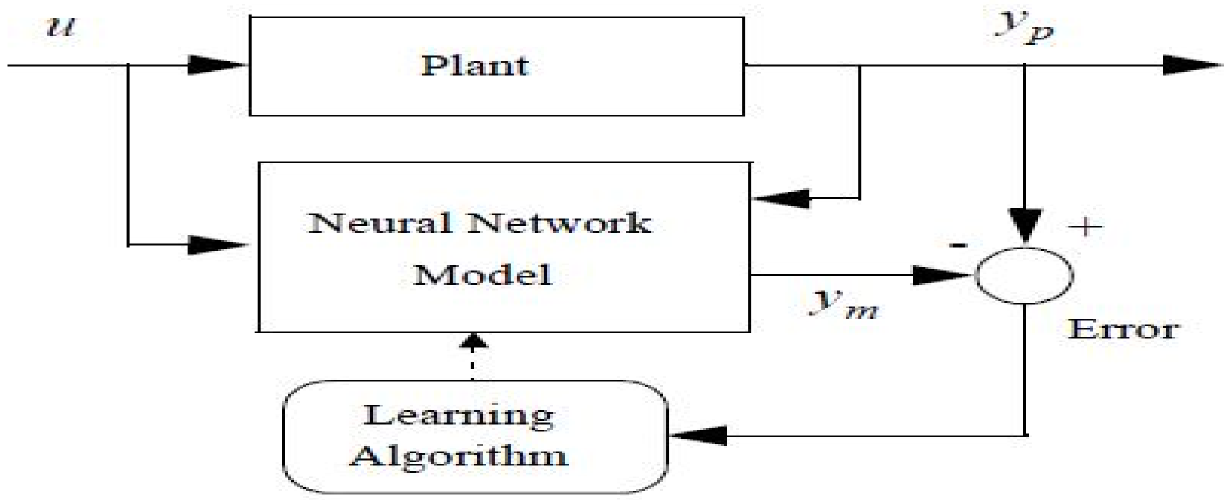
Control Design: use the neural network plant model to design (or train) the controller.

1. Model Predictive Control:

- ➢ The first step in model predictive control is to determine the NN plant model (system identification). Next, the plant model is used by the controller to predict future performance.

System Identification:

- ➢ The first stage of model predictive control is to train a NN to represent the forward dynamics of the plant.
- ➢ The prediction error between the plant output and the NN output is used as the NN training signal.
- ➢ The neural network plant model uses previous inputs and previous plant outputs to predict future values of the plant output.



Predictive Control:

- The NN model predicts the plant response over a specified time horizon.
- The predictions are used by a numerical optimization program to determine the control signal that minimizes the following performance criterion;

$$J = \sum_{j=N_1}^{N_2} (y_r(k+j) - y_m(k+j))^2 + \rho \sum_{j=1}^{N_u} (u'(k+j-1) - u'(k+j-2))^2$$

where;

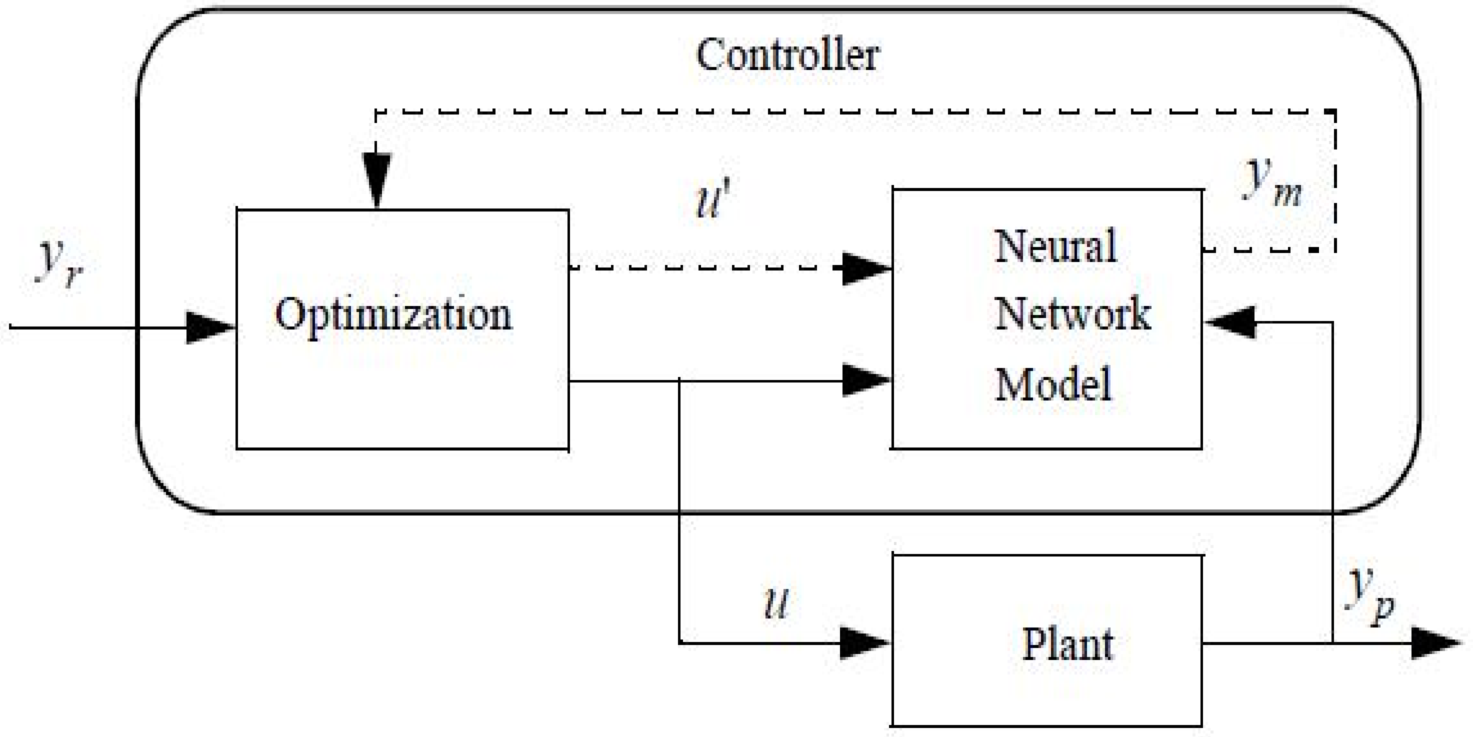
N1, N2, and Nu: define the horizons over which the tracking error and the control increments are evaluated.

ŭ: is the control signal,

Yr: is the desired response and ym is the network model response.

ρ: determines the contribution that the sum of the squares of the control increments has on the performance index.

Neural Network Predictive Control:



The controller consists of the NN plant model and the optimization block. The optimization block determines the values of ŭ that minimize J , and then the optimal u is input to the plant.

2. Feedback Linearization Control:

- The idea of this controller is to transform nonlinear system dynamics into linear dynamics by canceling the nonlinearities.
- The next control input is computed to force the plant output to follow a reference signal.
- The NN plant model is trained with static backpropagation. The controller is a rearrangement of the plant model, and requires minimal online computation.

Identification Stage:

As with model predictive control, the first step is to identify the system to be controlled. The approximate model is given by:

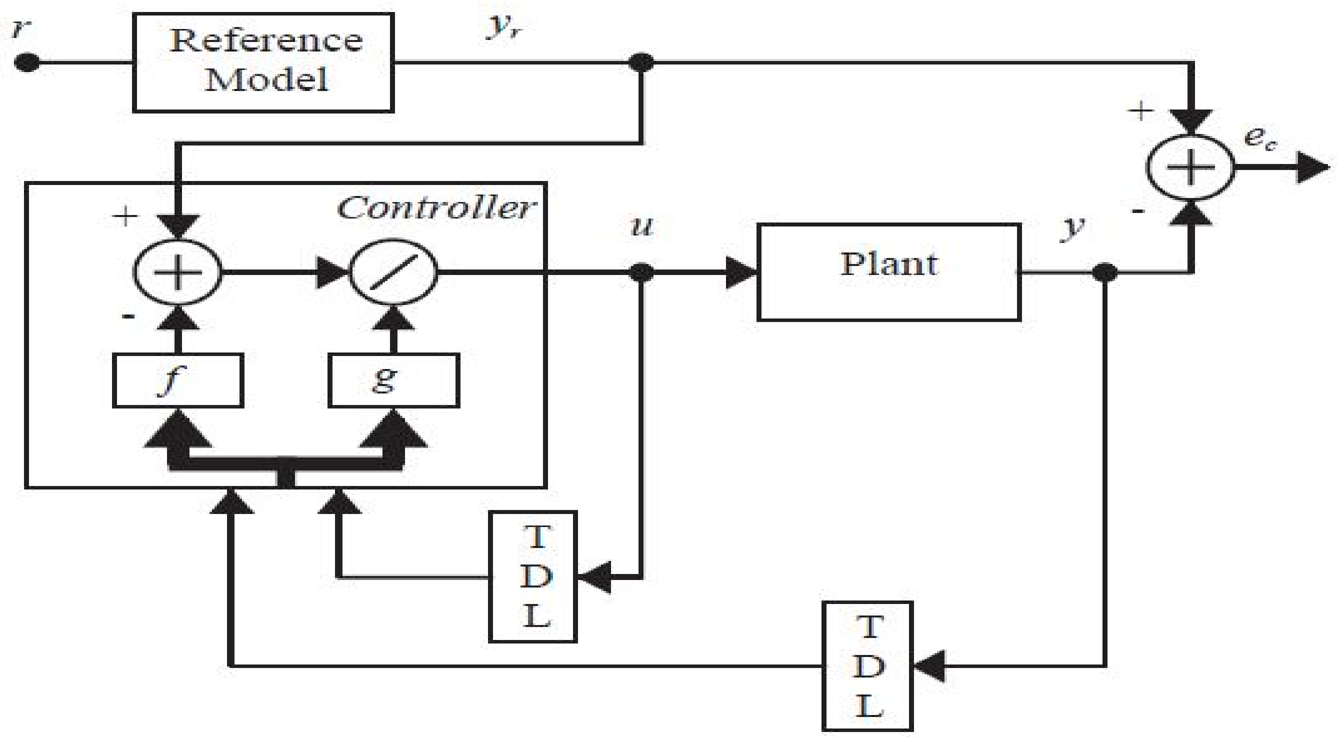
$$\begin{aligned}\hat{y}(k+d) = {} & f[y(k), y(k-1), \ldots, y(k-n+1), u(k-1), \ldots, u(k-m+1)] \\ & + g[y(k), y(k-1), \ldots, y(k-n+1), u(k-1), \ldots, u(k-m+1)] \cdot u(k)\end{aligned}$$

Control Stage:

The advantage of the NN model of the plant is that you can solve for the control input that causes the system output to follow a reference signal: $y(k+d) = y_r(k+d)$

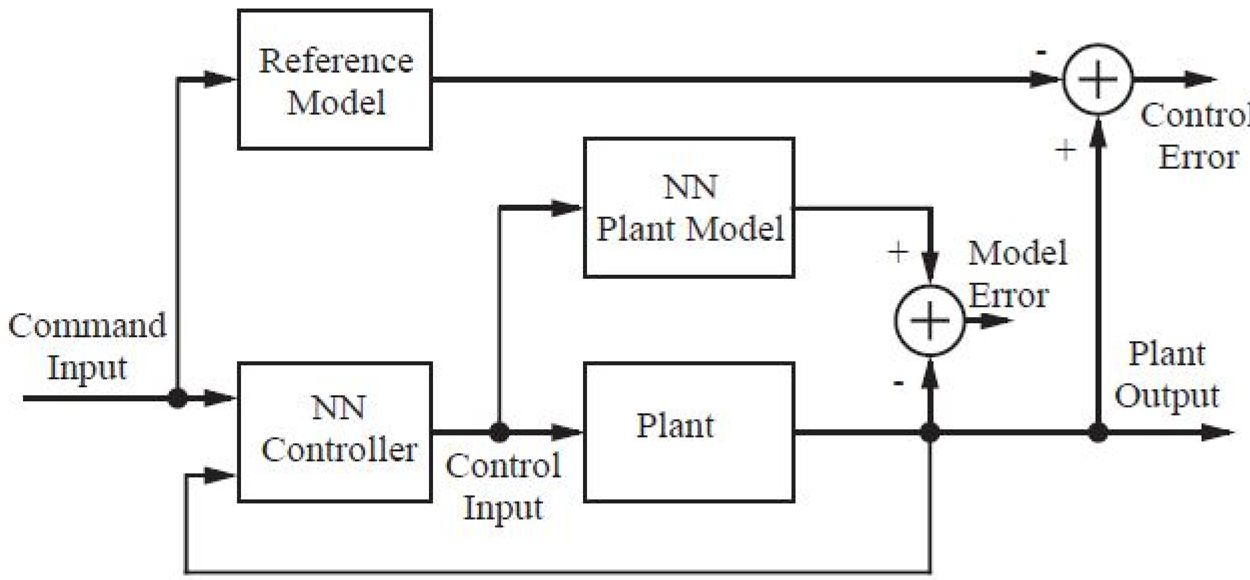
The resulting controller would have the form:

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



3. Model Reference Control:

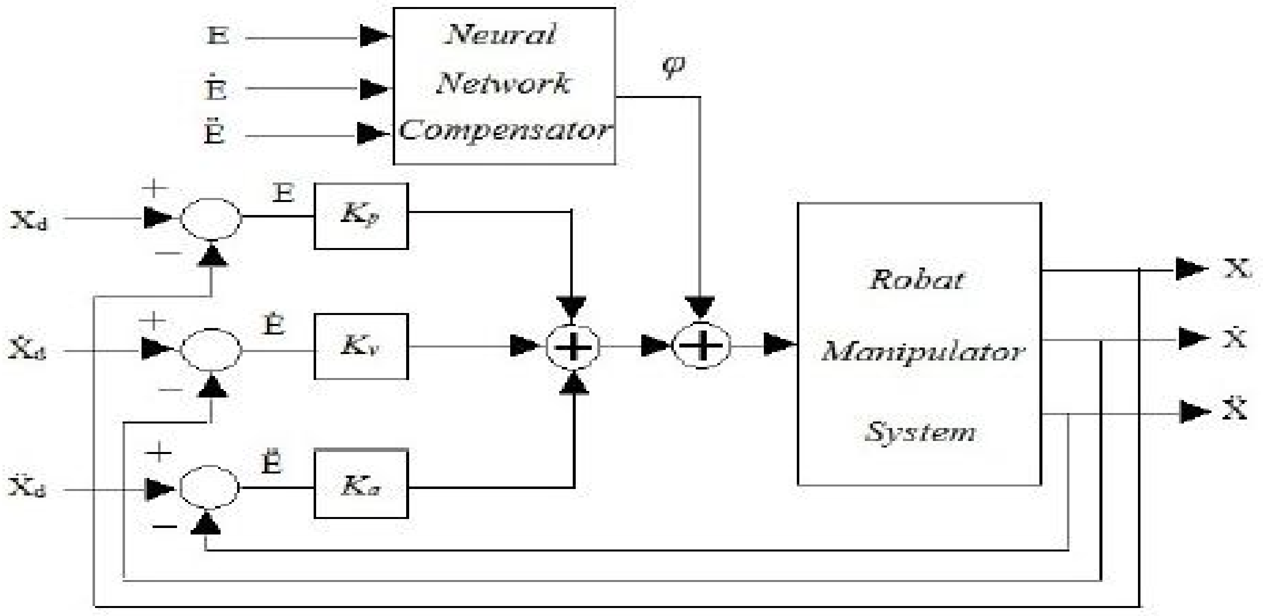
- This architecture uses two NNs: a controller network and a plant model network.
- A neural network plant model is first developed. The plant model is then used to train a neural network controller to force the plant output to follow the output of a reference model.
- This control architecture requires the use of dynamic backpropagation for training the controller. This generally takes more time than training static networks with the standard backpropagation algorithm.

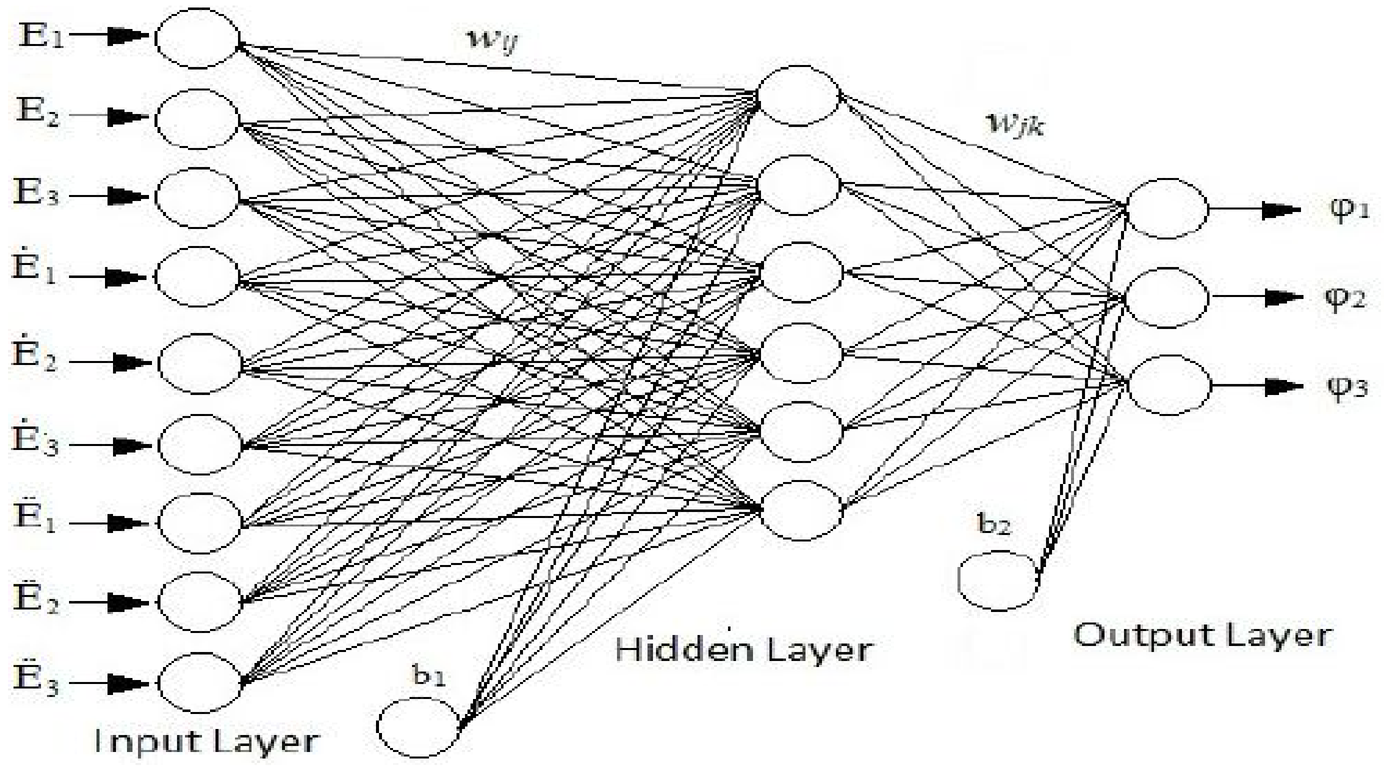


Applications:
NN PID Controller of a Robot Manipulator:

Ref: S. Pezeshki1, S. Badalkhani and A. Javadi, "Performance Analysis of a Neuro-PID Controller Applied to a Robot Manipulator", Int J Adv Robotic Sy, 2012, Vol. 9.

- A NN PID controller is used to improve the performance of a robot manipulator.
- The NN is applied to compensate the effect of the uncertainties of the robot model.
- The NN output φ cancels out the uncertainties caused by an accurate robot model. The output signal φ of the NN is added to the control input U whose dimension equals the number of degrees of freedom of the robot manipulator. Therefore, the vector φ is three dimensional.





- The three layer feedforward neural network structure is used as the compensator.
- It is composed of linear input layer, output layer and a nonlinear intermediate hidden layer which uses a sigmoid function.

Applications: NN-Based PID Auto-tuning:

The majority of the controllers used in industry are of the PID type. These controllers are tuned using simple empirical rules such as the Ziegler-Nichols tuning rule. The controllers are often poorly tuned due to neglect or lack of time.

The continuous PID controller transfer function is;

$$U(s) = k_c\left(1 + T_i s + \frac{T_d s}{1 + T_f s}\right)E(s)$$

The digital PID controller can be written as:

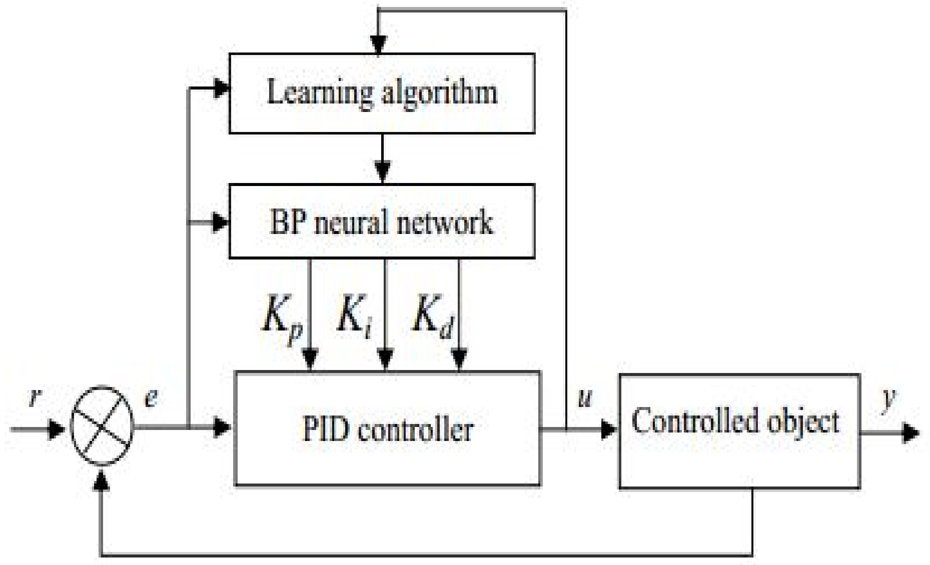
$$u[k] = k_D\left(1 + \frac{T_s}{T_{DI}}\frac{1}{q-1} + \frac{T_{DD}}{T_s}\frac{q-1}{q+\gamma}\right)e[k]$$

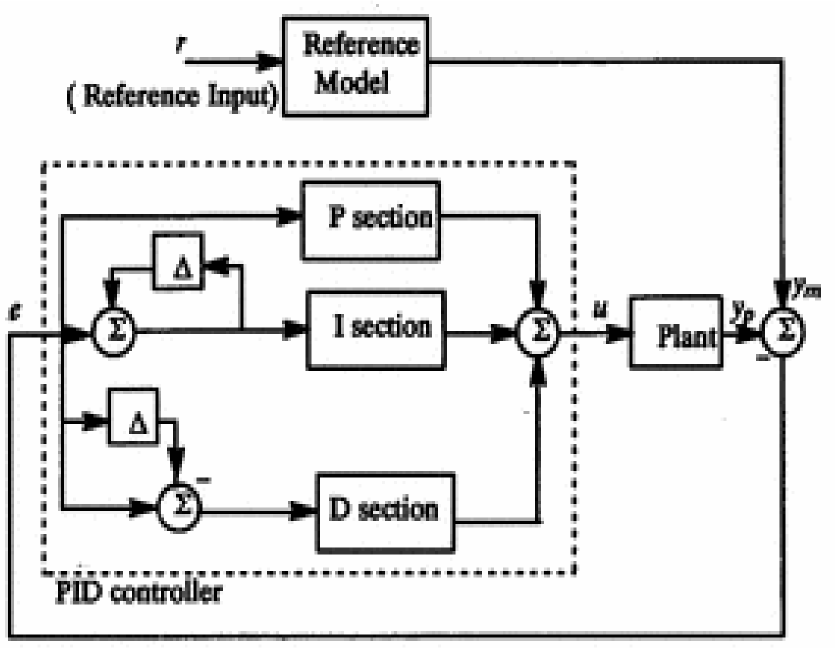
and γ is given by: $\gamma = -e^{-\frac{T_s}{T_f}}$

$$\Delta u[k] = u[k] - u[k-1] = k_D\left(1 - q^{-1} + \frac{q^{-1}T_s}{T_{DI}} + \frac{T_{DD}(1 - 2q^{-1} + q^{-2})}{T_s(1 + \gamma q^{-1})}\right)e[k]$$

Methods of PID auto-tuning:

- Ziegler and Nichols methods.
- On-line parameter estimation.
- Exert and fuzzy logic tuning methods.
- NN-based PID auto-tuner.



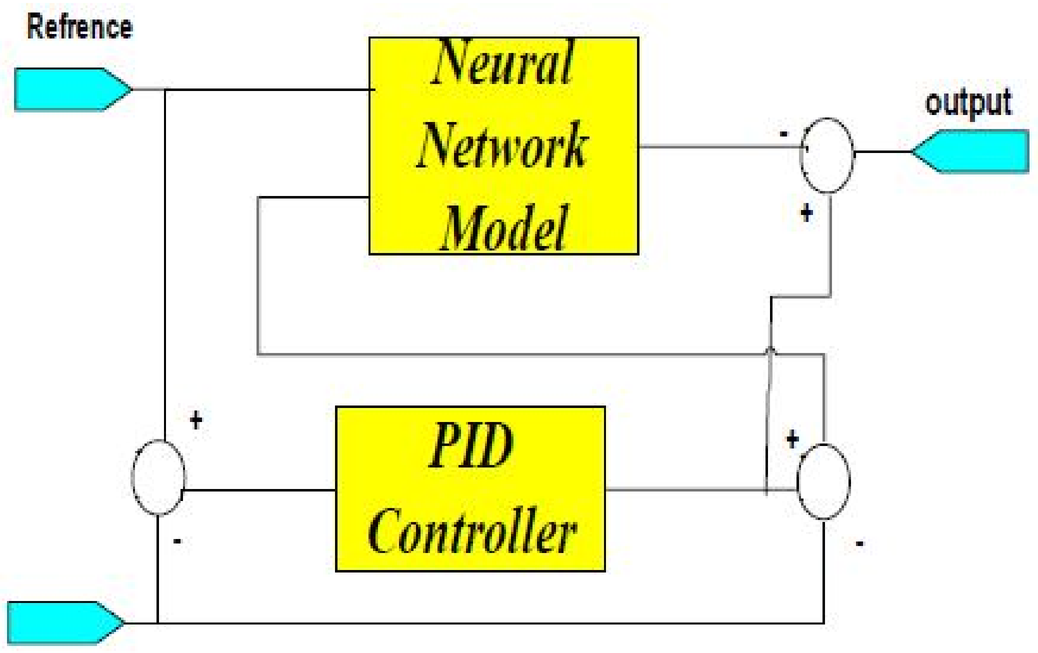


Ref:

1. *L. Luoren, L. Jinling, "Research of PID Control Algorithm Based on Neural Network", Energy Procedia 13 (2011) 6988 – 6993*
2. *M. Bahrami & K.E. Tait, "A neural network-based proportional integral derivative controller", Neural Computing & Applications, 1994, Vol.2, No.3, pp 134-141*

Neural Network controllers based on PID controllers:

This paper presents the control of two link robotic manipulator systems using Neural Network. It covers PD controller, PID Controller, NN Controller based on PD Controller and NN Controller based on PID Controller.



Ref: Leila Fallah Araghi, M. Habibnejad Korayem, Amin Nikoobin, Farbod Setoudeh, "Neural Network Controller Based on PID Controller for Two links- Robotic Manipulator Control", Proceedings of the World Congress on Engineering and Computer Science 2008 WCECS 2008, October 22 - 24, 2008, USA.

Neural PID Auto-tuner:

In this approach the open loop step response of a plant is discretized, its samples being used as inputs to a MLP. The role of the NN is, based on these inputs, to determine the corresponding PID parameters.

The MLP has therefore three outputs, corresponding to the three PID parameters.

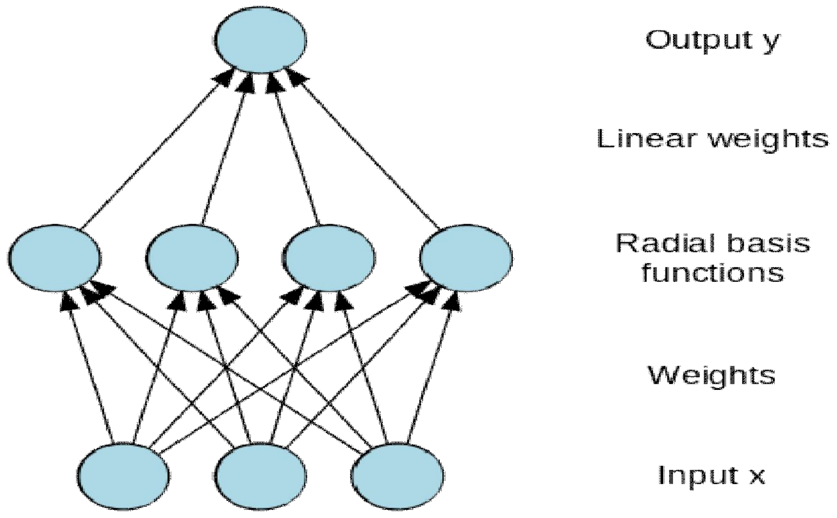
Some observations can be made concerning this method of PID auto-tuning:

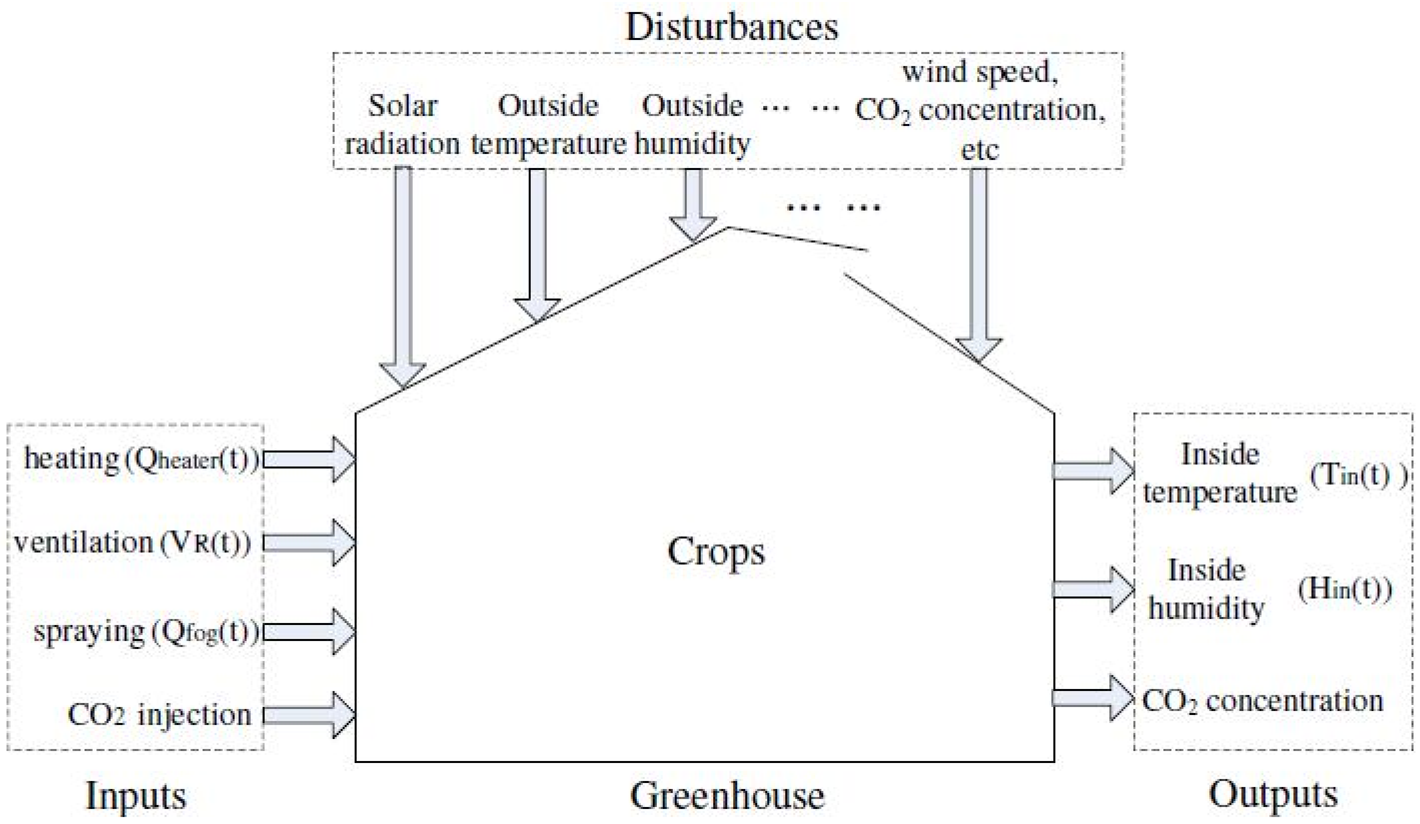
- It is an open loop technique, it can not be applied in closed loop.
- Because samples of the open loop step response are used as inputs to the MLPs,
- obtaining good results might require a high number of samples. This leads to a large number of parameters in the MLP, resulting in large training times.
- This technique is dependent on the sampling time chosen.
- The responses obtained with Ziegler-Nichols tuning rule are not well damped. It seems sensible to obtain the target PID values using a better criterion.
- If the Ziegler-Nichols technique is used to derive the target PID values then there is no need to consider three outputs for the MLP, since the integral and derivative time constants are linearly interdependent.

Applications: Nonlinear Adaptive PID Control

S. Zeng, H. Hu, L, Xu & G. Li , " Nonlinear Adaptive PID Control for Greenhouse Environment Based on RBF Network", Sensors 2012, 12, pp.5328-5348.

- A hybrid controller combining Radial Basis Function (RBF) neural network with PID controllers for the greenhouse climate control.
- A model of nonlinear plant (Greenhouse climate) is formulated.
- RBF NN is used to tune and identify all PID gain parameters online and adaptively.
- a RBF NN is an ANN that uses **radial basis function** as activation functions.
- A **radial basis function** is a real-valued function whose value depends only on the distance from the origin, so that ; or alternatively on the distance from some other point *c*, called a *center*, so that .
- RBF NN Architecture: An input vector (X) is used as input to all radial basis functions, each with different parameters. The output of the network is a linear combination of the outputs from radial basis functions.



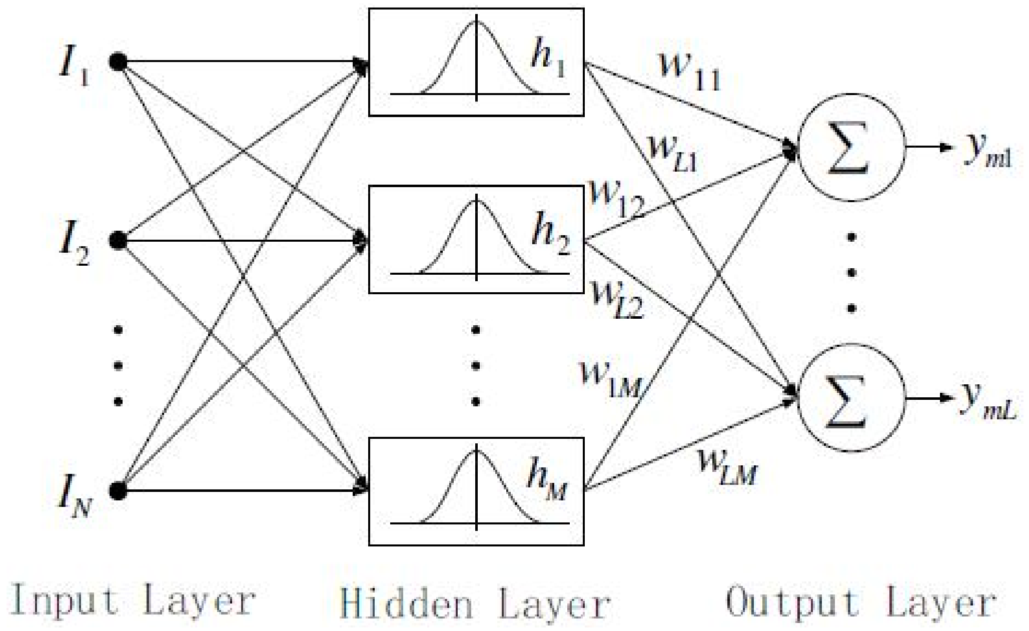


$$\dot{x}_1(t) = \frac{UA}{\rho C_p V_T} x_1(t) - \frac{1}{V_T} x_1(t) u_1(t) - \frac{\lambda}{\rho C_p V_T} u_2(t) + \frac{1}{\rho C_p V_T} v_1(t) + \frac{UA}{\rho C_p V_T} v_2(t) + \frac{1}{V_T} u_1(t) v_2(t)$$

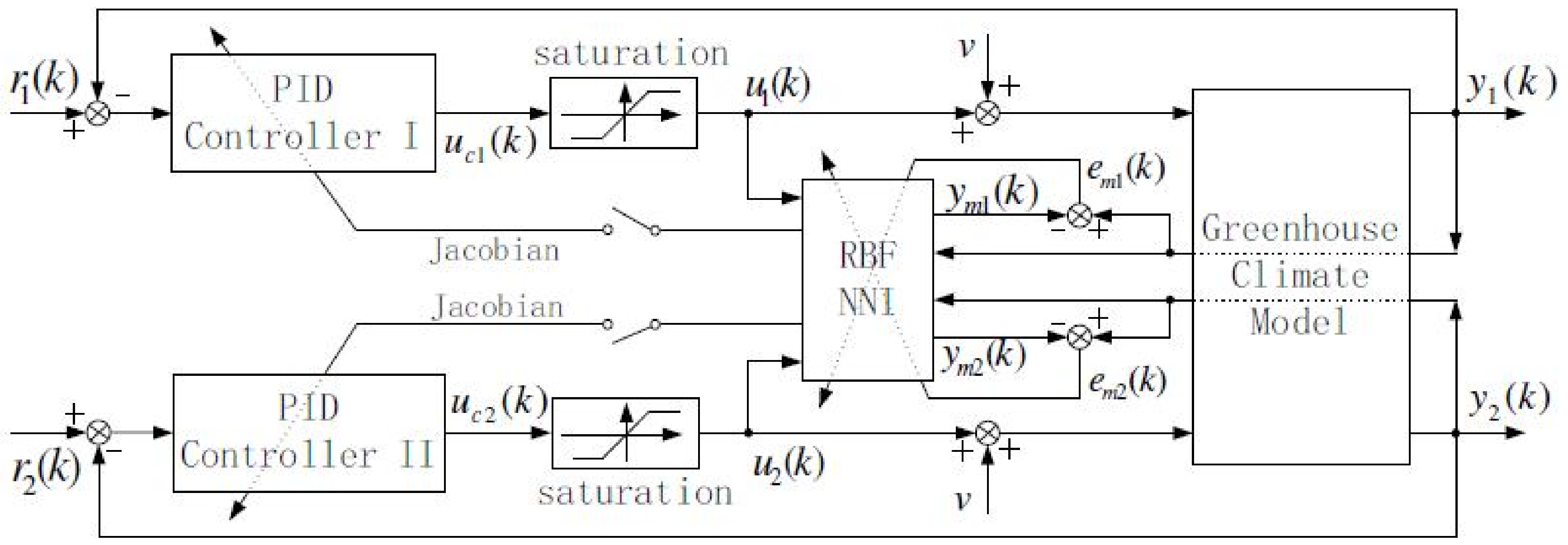
$$\dot{x}_2(t) = \frac{\beta_T}{\rho V_H} x_2(t) + \frac{1}{\rho V_H} u_2(t) + \frac{\alpha}{\lambda \rho V_H} v_1(t) - \frac{1}{\rho V_H} x_2(t) u_1(t) + \frac{1}{\rho V_H} u_1(t) v_3(t)$$

Design of Adaptive Neuro-PID Controller for Greenhouse Climate: RBF Network Structure:

- RBF neural networks have an input layer, a hidden layer and an output layer.
- The neurons in the hidden layer contain Gaussian transfer functions whose outputs are inversely proportional to the distance from the center of the neuron.



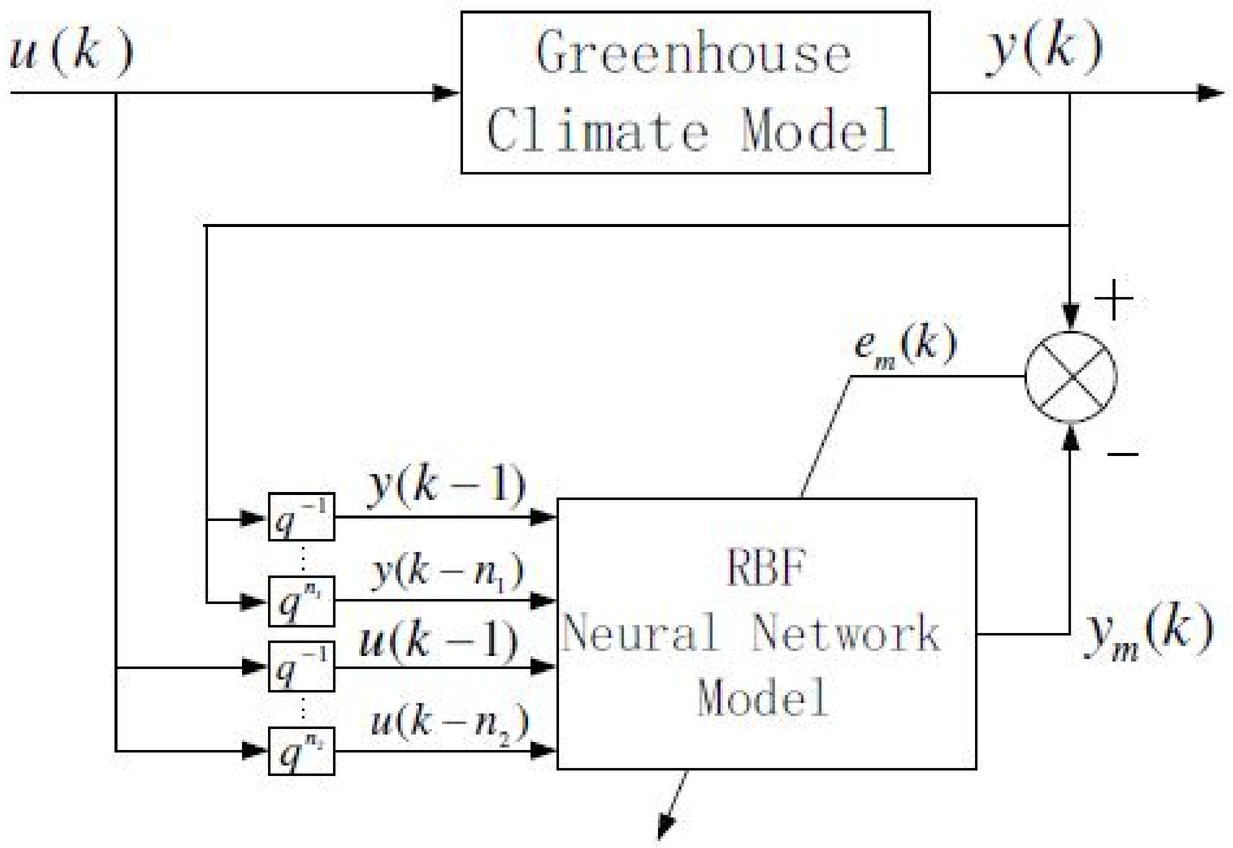
Hybrid Control Strategy:



- ➢ The greenhouse dynamic system is a two-input and two-output continuous time nonlinear system. We adopt a hybrid control strategy by combining RBF network with the conventional PID controller, in which there is one neural network model with two outputs (namely, inside temperature and humidity).
- ➢ RBF network is used to tune the parameters of the conventional PID controller through Jacobian information.

Greenhouse Climate Model:

The RBF neural network has been used to obtain a model for the plant because it has excellent adaptability, learning ability and powerful ability to approximate nonlinear functions.



Remarks:

The results given in the paper outline that the proposed adaptive hybrid control scheme by combining RBF network with a conventional PID controller is a promising method with the following features;

- It has a satisfactory control performance characterized by adaptability, robustness and good set-point tracking.
- It can achieve the real-time online control of various MIMO systems.
- It can be applied in nonlinear dynamic control systems such as greenhouse climate system.
- The approach is not limited to greenhouse applications and could easily be extended to other applications.

References:

1. Eric A. Wan, “Control Systems: Classical, Neural and Fuzzy”, Oregon Graduate Institute, Lecture Notes, 1998.
2. Saerens, M., Soquet, A., ‘Neural-controller based on back-propagation algorithm’, IEE Proceedings, Part F, Vol. 138, No 1, 1991, pp. 55-62.
3. Moonyong Lee and Sunwon Park, “Process Control Using a Neural Network Combined with the Conventional PID Controllers”, ICASE: The Institute of Control, Automation and Systems Engineers, KOREA Vol. 2, No. 3, September, 2000.
4. *S. Pezeshki1, S. Badalkhani and A. Javadi, “Performance Analysis of a Neuro-PID Controller Applied to a Robot Manipulator”, Int J Adv Robotic Sy, 2012, Vol. 9.*
5. FRANCKLIN RIVAS-ECHEVERRÍA, ADDISON RÍOS-BOLÍVAR, JEANETTE CASALES-ECHEVERRÍA, “Neural Network-based Auto-Tuning for PID Controllers”, http://www.wseas.us/e-library/conferences/crete2001/papers/658.pdf
6. Jun Kanga,b, Wenjun Menga, , Ajith Abrahamc,d,e, Hongbo Liuc,e,, “An Adaptive PID Neural Network for Complex Nonlinear System Control”, http://www.softcomputing.net/neucom13_2.pdf
7. *S. Zeng, H. Hu, L, Xu & G. Li , “ Nonlinear Adaptive PID Control for Greenhouse Environment Based on RBF Network”, Sensors 2012, 12, pp.5328-5348.*