F.L. Lewis, NAI

Moncrief-O'Donnell Chair, UTA Research Institute (UTARI)
The University of Texas at Arlington, USA

Adaptive Control, Robust Control & Neural Networks for Control of Nonlinear Processes and Systems


Bringing Together Control Theory and Computational Intelligence

Importance of Feedback Control

Darwin 1850- FB and natural selection
Vito Volterra 1890- FB and fish population balance
Adam Smith 1760- FB and international economy
James Watt 1780- FB and the steam engine
FB and cell homeostasis

The resources available to most species for their survival are meager and limited

Nature uses Optimal control

Feedback Control Systems

Aircraft autopilots
Car engine controls
Ship controllers
Compute Hard disk drive controllers
Industry process control – chemical, manufacturing
Robot control

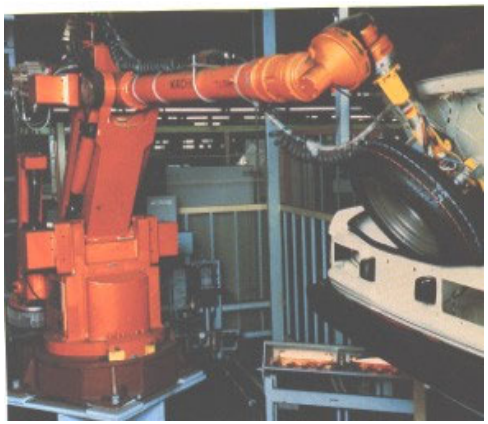
Industrial Revolution –
- Windmill control, British millwrights - 1600s
- Steam engine and prime movers
  - James Watt 1769

Steamship
Steam Locomotive boiler control

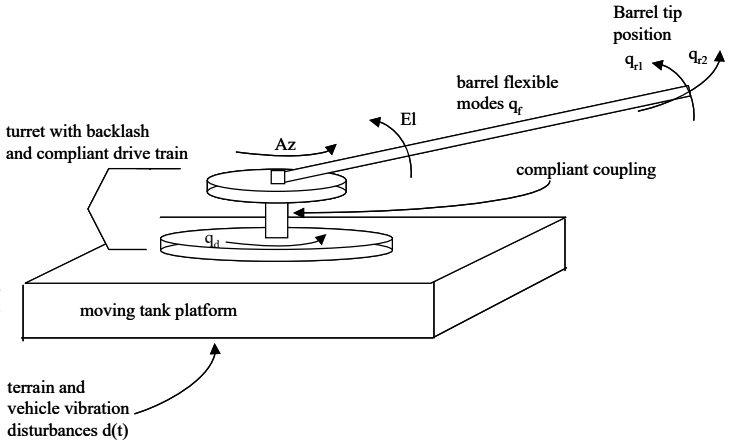
Sputnik 1957
Aerospace systems

Relevance- Machine Feedback Control

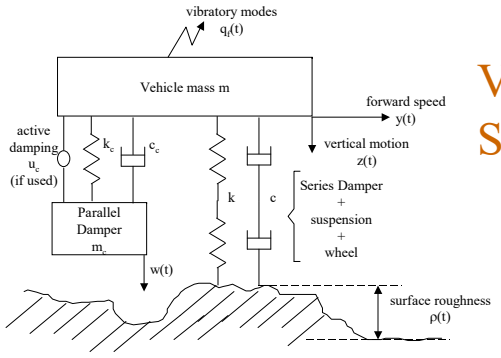
High-Speed Precision Motion Control with unmodeled dynamics, vibration suppression, disturbance rejection, friction compensation, deadzone/backlash control



Industrial Machines



Satellite pointing
Land Systems



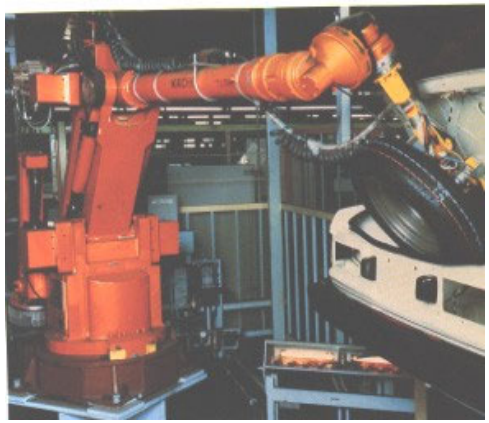
Single-Wheel/Terrain System with Nonlinearities

Vehicle Suspension

Aerospace

Relevance- Industrial Process Control

Precision Process Control with unmodeled dynamics, disturbance rejection, time-varying parameters, deadzone/backlash control



Industrial Machines

XY Table

Chemical Vapor Deposition

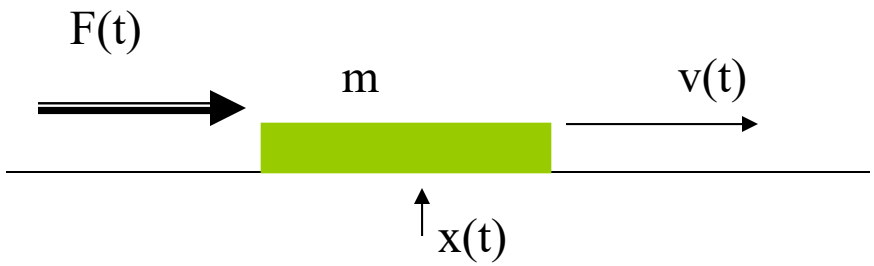
Autoclave

Lagrange Dynamical systems

Newton's Law

$$F = ma = m\ddot{x}$$

$$\ddot{x} = \frac{F(t)}{m} \equiv u(t)$$



Lagrange's Eqs. Of Motion ⟹

Industrial Process and Motion Systems (Vehicles, Robots)

$$M(\dot{q})\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_d = B(q)\tau$$

Control Input

inertia

Coriolis/centripetal force

gravity

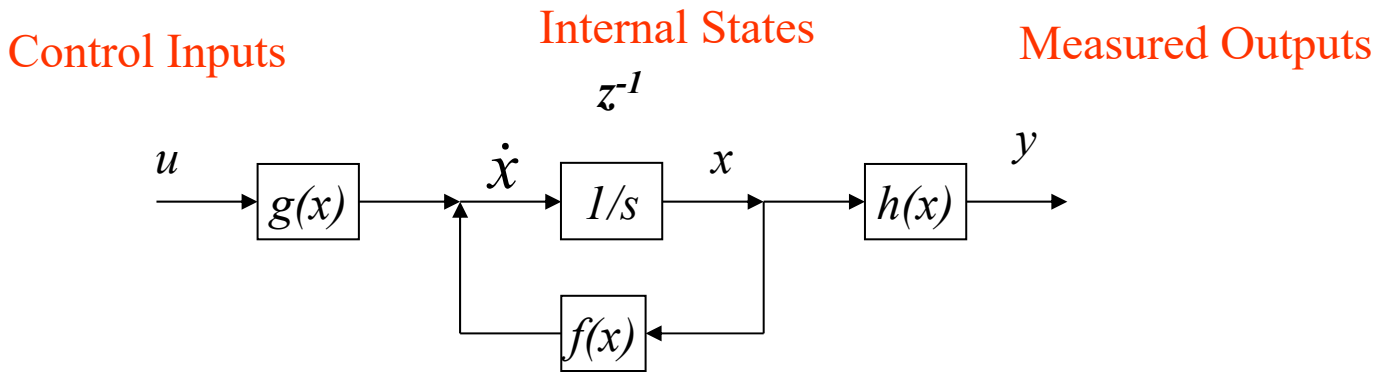
friction

disturbances

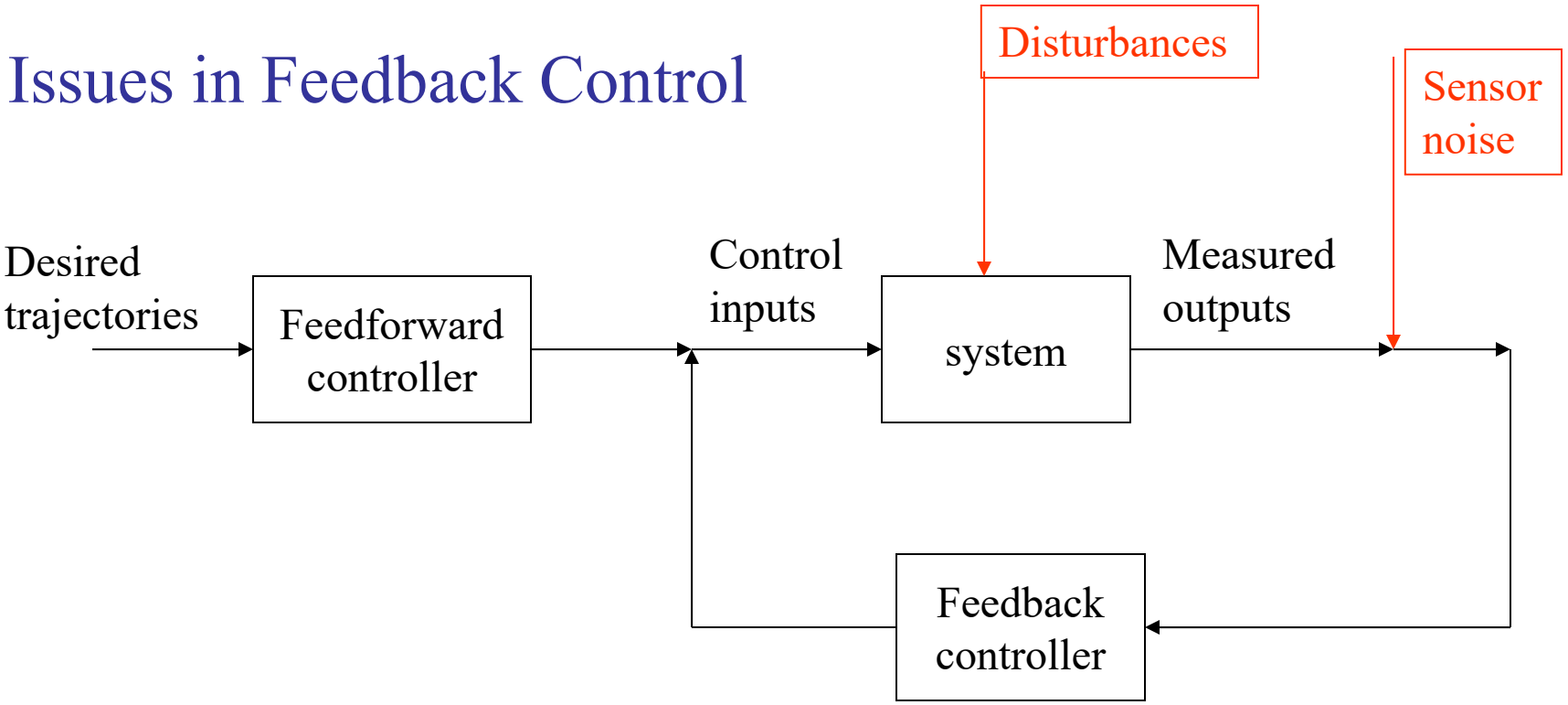
Actuator problems

Dynamical System Models

Continuous-Time Systems	Discrete-Time Systems
Nonlinear system	
$\dot{x} = f(x) + g(x)u$ $y = h(x)$	$x_{k+1} = f(x_k) + g(x_k)u_k$ $y_k = h(x_k)$
Linear system	
$\dot{x} = Ax + Bu$ $y = Cx$	$x_{k+1} = Ax_k + B_k$ $y_k = Cx_k$



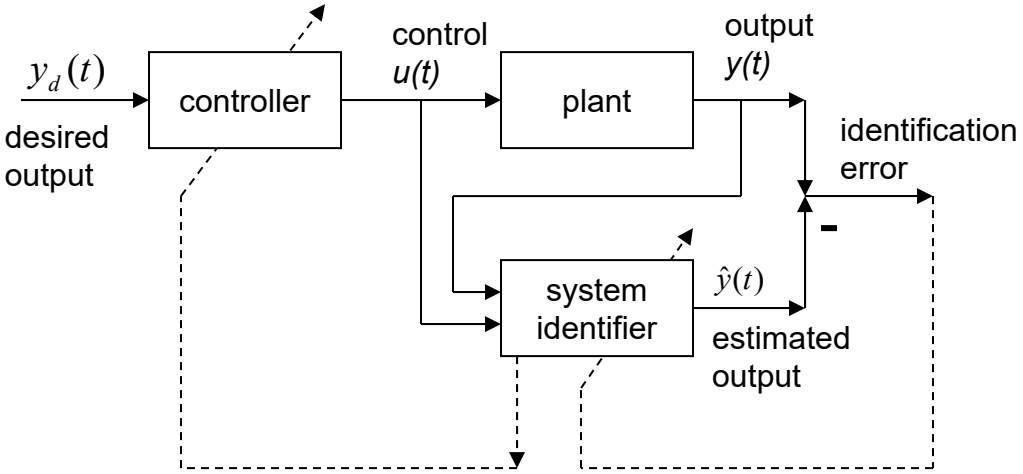
Issues in Feedback Control



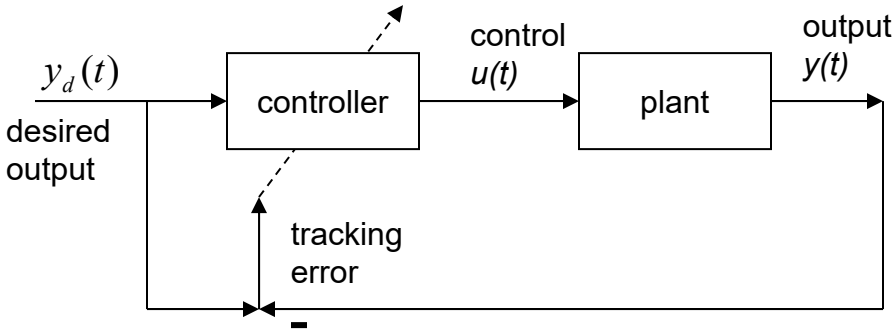
Stability
Tracking
Boundedness
Robustness
- to disturbances
- to unknown dynamics

Unknown Process dynamics
Process Nonlinearities
Unknown Disturbances

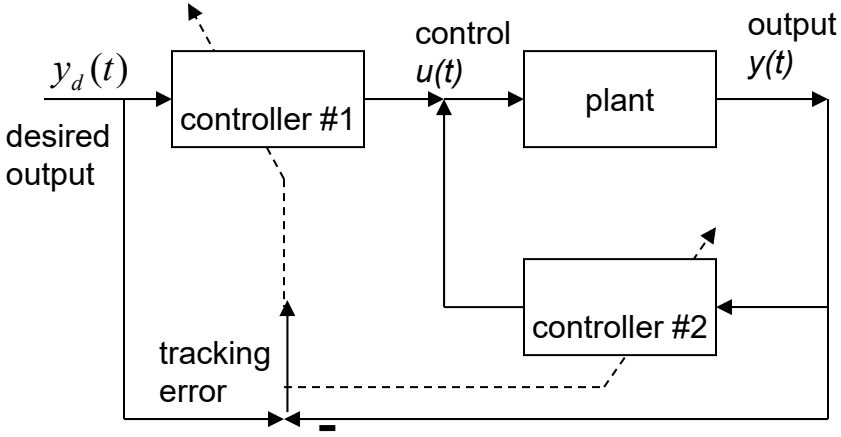
Controller Topologies



Indirect Scheme



Direct Scheme



Feedback/Feedforward Scheme

Feedback Linearization
Adaptive Control
Neural Networks for Control
Neural-adaptive Control

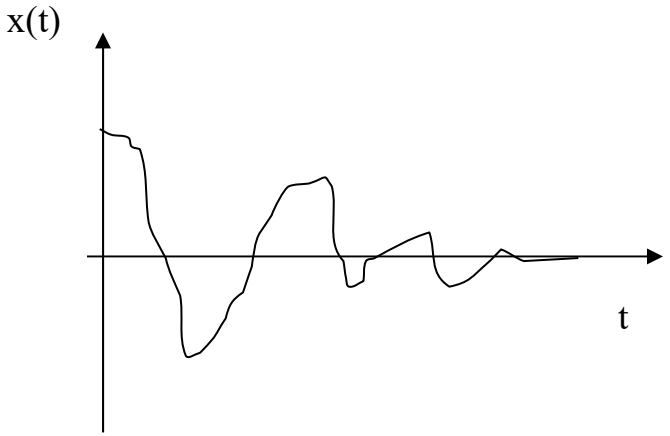
6 US Patents

Sponsored by:
China Qian Ren
China Project 111
US NSF, ARO, ONR, AFOSR

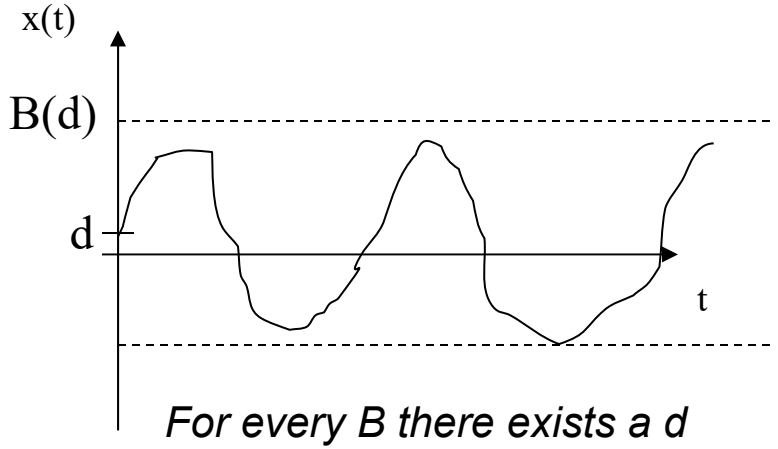
Definitions of System Stability

$$\dot{x} = f(x)$$

$$x_{k+1} = f(x_k)$$

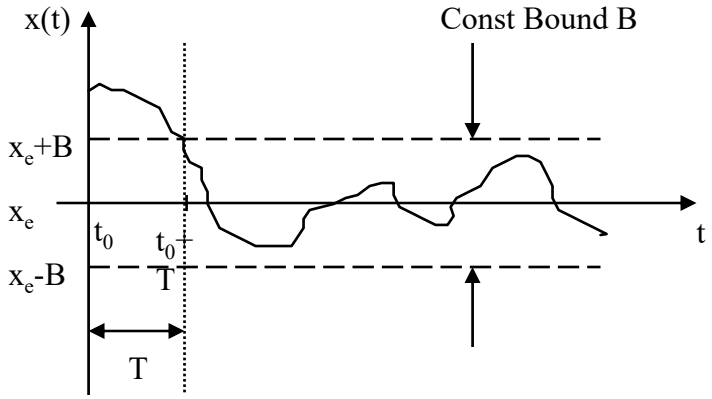


Asymptotic Stability



For every B there exists a d

Marginal or Bounded Stability- SISL



There exists bound B that cannot be prescribed

Uniform Ultimate Boundedness

Feedback Linearization

Example 1. Linear System

$$y = \frac{1}{s^2 + a_1 s + a_2} u \qquad \ddot{y} + a_1 \dot{y} + a_2 y = u$$

desired to track a reference input $y_d(t)$

Tracking error $e = y_d - y$

$$\ddot{e} = \ddot{y}_d + a_1 \dot{y} + a_2 y - u$$

Sliding variable $r = \dot{e} + \Lambda e$

$$\dot{r} = \ddot{e} + \Lambda \dot{e} = \ddot{y}_d + \Lambda \dot{e} + a_1 \dot{y} + a_2 y - u$$

Auxiliary input $u = v + \ddot{y}_d + \Lambda \dot{e}$

Error dynamics $\dot{r} = a_1 \dot{y} + a_2 y - v$

Of the form $\dot{r} = f(x) - v$

Unknown parameters

Known Regression Vector

Unknown function

$$f(x) = a_1 \dot{y} + a_2 y = \begin{bmatrix} a_1 & a_2 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \end{bmatrix} = W^T \phi(x)$$

Example 2. Nonlinear Lagrange System

Feedback Linearization

$$\ddot{y} + d(y, \dot{y}) + k(y) = u$$

unknown nonlinear friction

unknown nonlinear damping term

desired to track a reference input $y_d(t)$

Lagrangian System Appears in:
- Process control
- Mechanical systems
- Robots

Tracking error $e = y_d - y$

$$\ddot{e} = \ddot{y}_d + d(y, \dot{y}) + k(y) - u$$

Sliding variable $r = \dot{e} + \Lambda e$

Auxiliary input $u = v + \ddot{y}_d + \Lambda \dot{e}$

Error dynamics $\dot{r} = f(x) - v$

with $f(x) = d(y, \dot{y}) + k(y)$

Assume Linear in the Parameters (LIP)

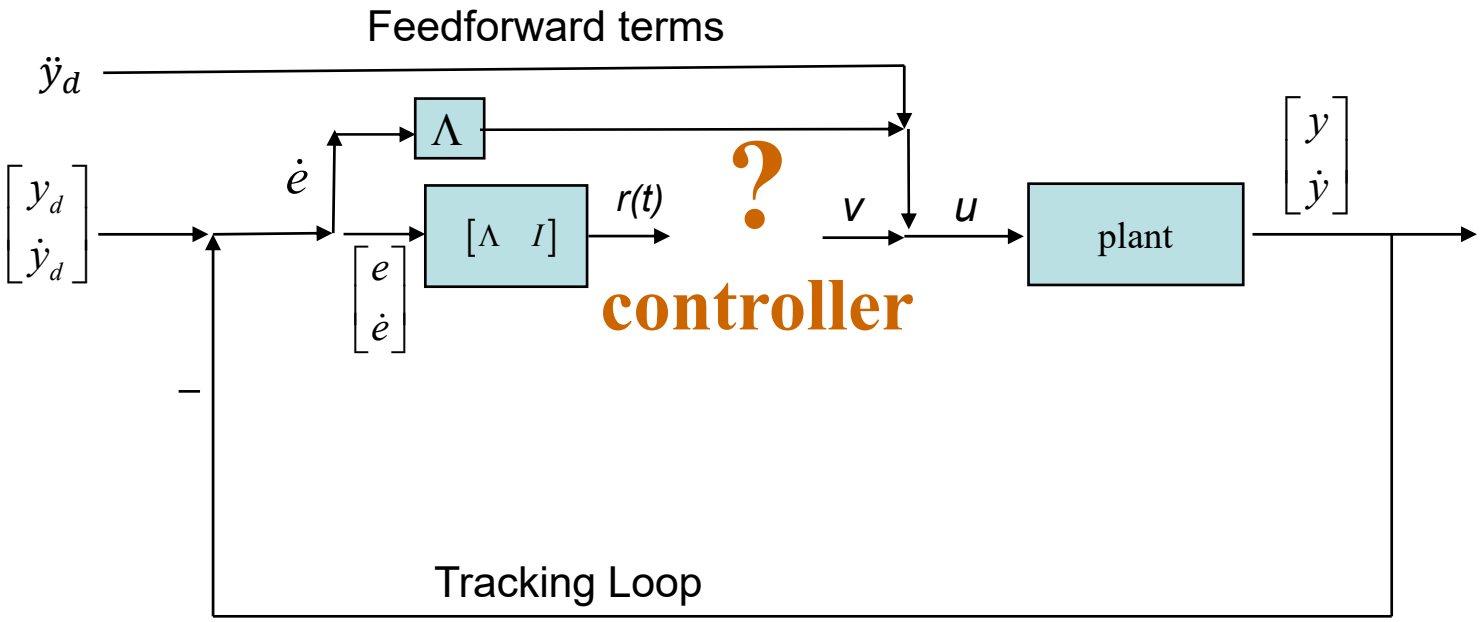
$$f(x) = \begin{bmatrix} D & K \end{bmatrix} \begin{bmatrix} d_1(y, \dot{y}) \\ k_1(y, \dot{y}) \end{bmatrix} = W^T \phi(x)$$

Known possibly nonlinear regression function

Unknown parameters

Feedback Linearization Controller

$$r = \dot{e} + \Lambda e \qquad u = v + \ddot{y}_d + \Lambda \dot{e}$$



The equations give the FB controller structure

Feedback Linearization Controller

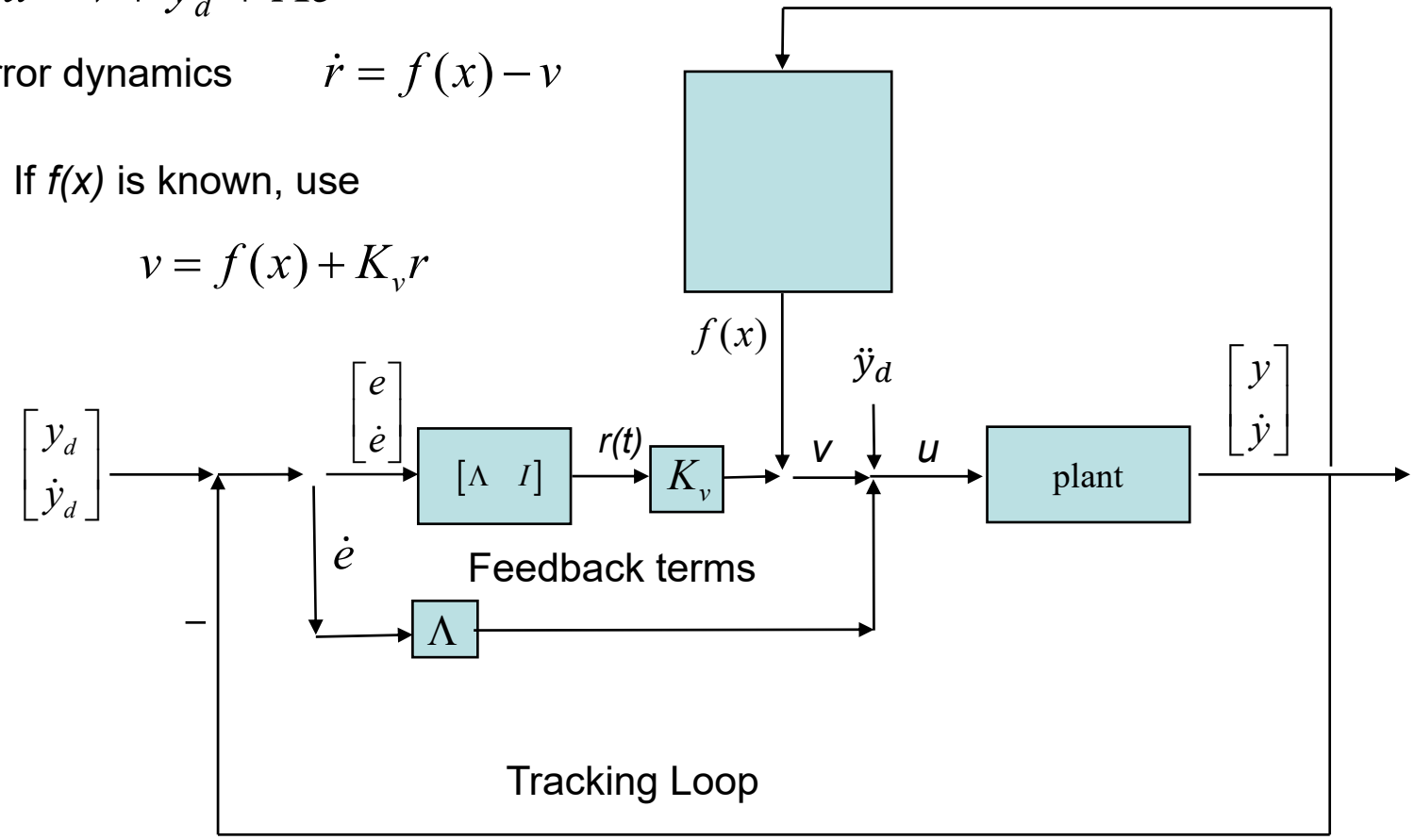
$$r = \dot{e} + \Lambda e$$

$$u = v + \ddot{y}_d + \Lambda \dot{e}$$

Error dynamics $\dot{r} = f(x) - v$

If *f(x)* is known, use

$$v = f(x) + K_v r$$



The equations give the FB controller structure

Adaptive Control

Error Dynamics

$$\dot{r} = f(x) - v$$

r(t)= control error

Control input

Unknown nonlinearities

Assume: *f(x)* is known to be of the structure

$$f(x) = W^T \phi(x)$$

Known basis set= regression vector – DEPENDS ON THE SYSTEM

Unknown parameter vector

LINEAR-IN-THE-PARAMETERS (LIP)

Error Dynamics

$$\dot{r} = W^T \phi(x) - v$$

Adaptive Control

Controller

$$v = \hat{f}(x) + K_v r = \hat{W}^T(t)\phi(x) + K_v r$$

Pos. def. control gain

ESTIMATE of unknown parameters

closed-loop system becomes

$$\dot{r} = W^T\phi(x) - v = W^T\phi(x) - \hat{W}^T\phi(x) - K_v r$$

$$\dot{r} = \tilde{W}^T\phi(x) - K_v r$$

Est. error drives the control error

Parameter estimation error

$$\tilde{W}(t) = W - \hat{W}(t)$$

Parameter estimate is updated (tuned) using the adaptive tuning law

$$\frac{d\hat{W}}{dt} = \dot{\hat{W}} = F\phi(x)r^T$$

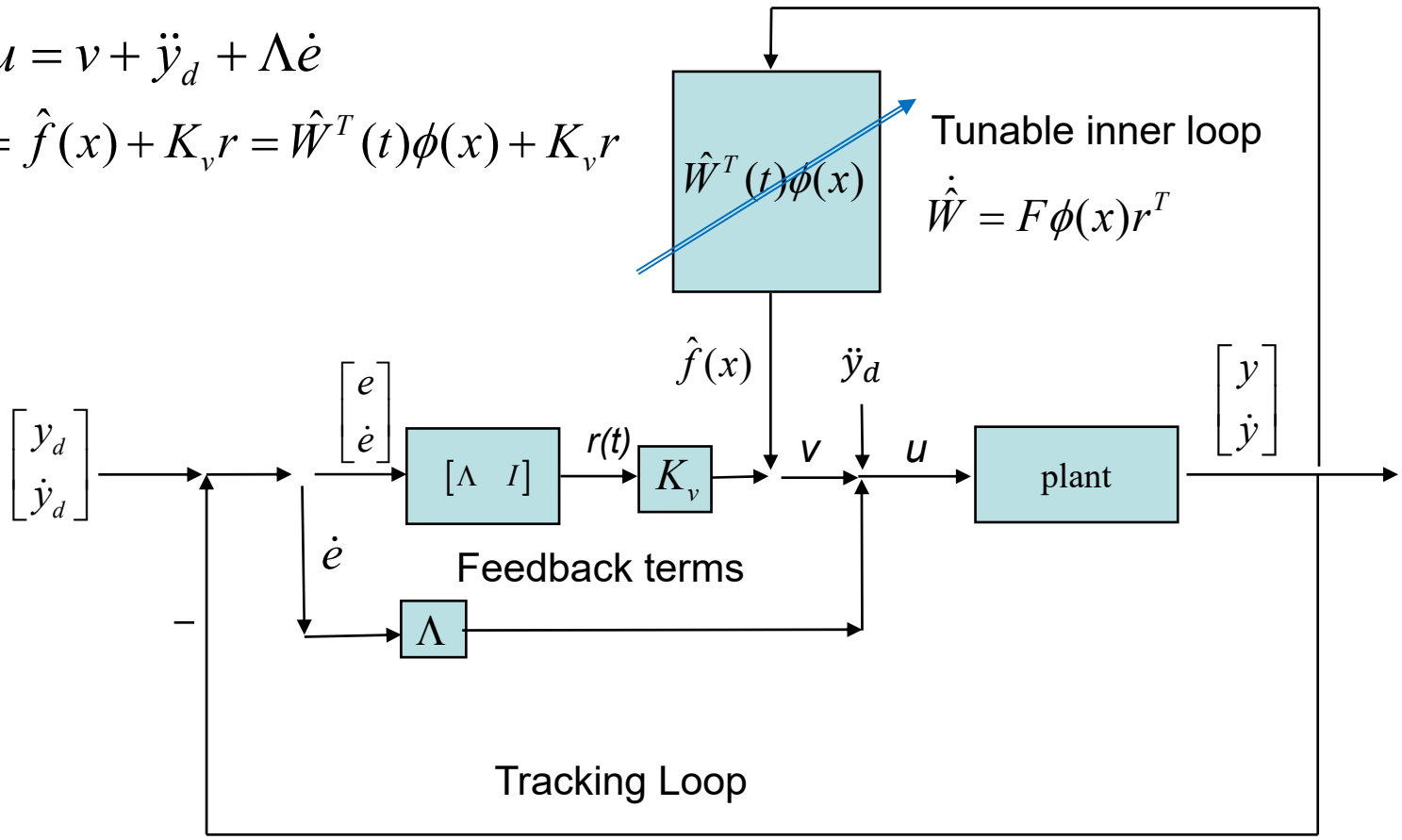
Feedback Linearization Adaptive Controller

A dynamic controller

$$r = \dot{e} + \Lambda e$$

$$u = v + \ddot{y}_d + \Lambda \dot{e}$$

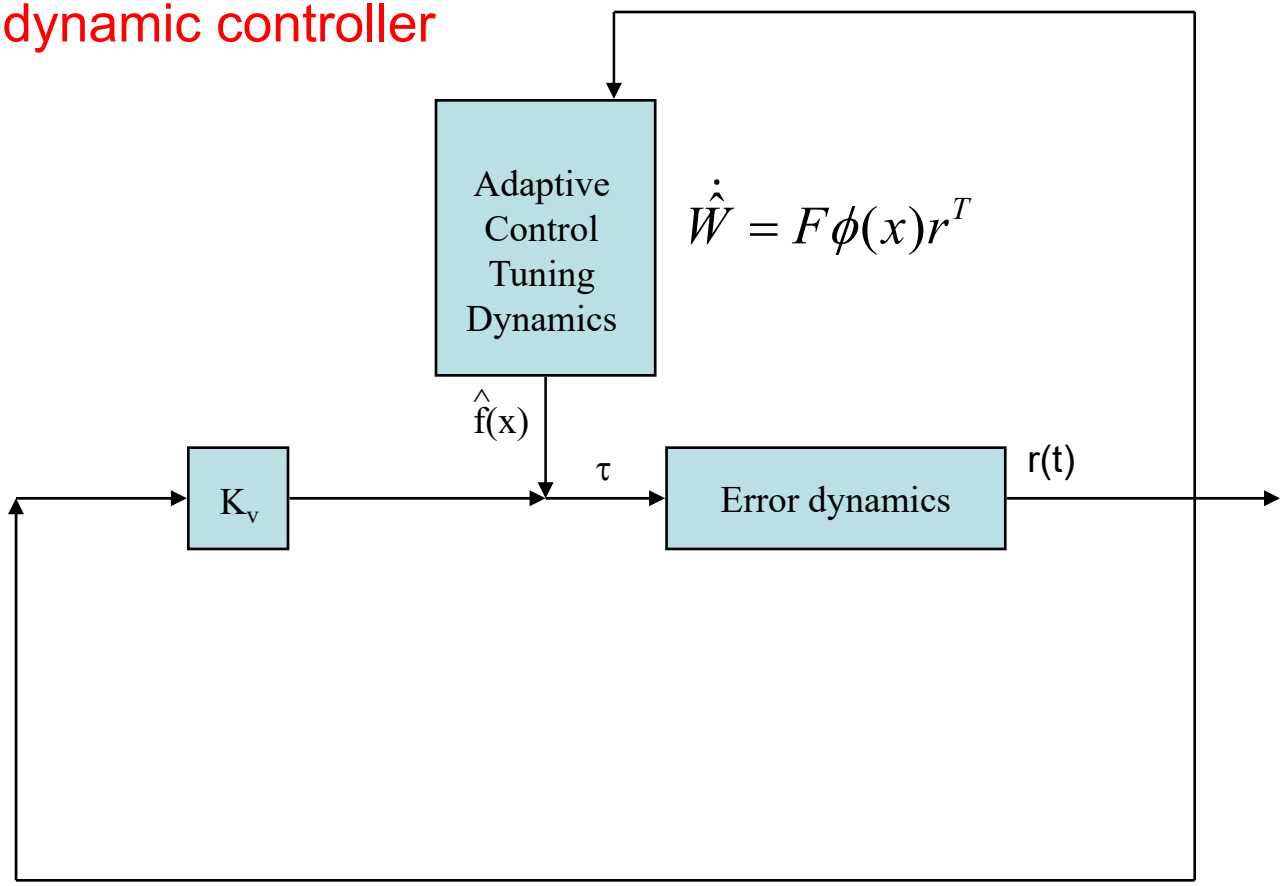
$$v = \hat{f}(x) + K_v r = \hat{W}^T(t)\phi(x) + K_v r$$



The equations give the FB controller structure

Adaptive Controller

A dynamic controller



Adaptive Controller

Adaptive Control

Performance of Adaptive Controller: Using the adaptive controller, the closed-loop system is asymptotically stable, i.e. the control error *r(t)* goes to zero.

If an additional Persistence of Excitation (PE) condition holds, the parameter estimates converge to the actual unknown parameters.

Proof:

$$L = \tfrac{1}{2} r^T r + \tfrac{1}{2} tr\{\tilde{W}^T F^{-1} \tilde{W}\} \qquad > 0$$

$$\dot{L} = r^T \dot{r} + tr\{\tilde{W}^T F^{-1} \dot{\tilde{W}}\}$$

$$\dot{r} = \tilde{W}^T \phi(x) - K_v r \qquad \text{Error dynamics}$$

$$\dot{L} = r^T \left(\tilde{W}^T \phi(x) - K_v r\right) + tr\{\tilde{W}^T F^{-1} \dot{\tilde{W}}\} = -r^T K_v r + tr\{\tilde{W}^T \left(F^{-1} \dot{\tilde{W}} + \phi(x) r^T\right)\}$$

$$\dot{\hat{W}} = F \phi(x) r^T \quad \text{or} \quad \dot{\tilde{W}} = -F \phi(x) r^T \qquad \text{Parameter tuning law}$$

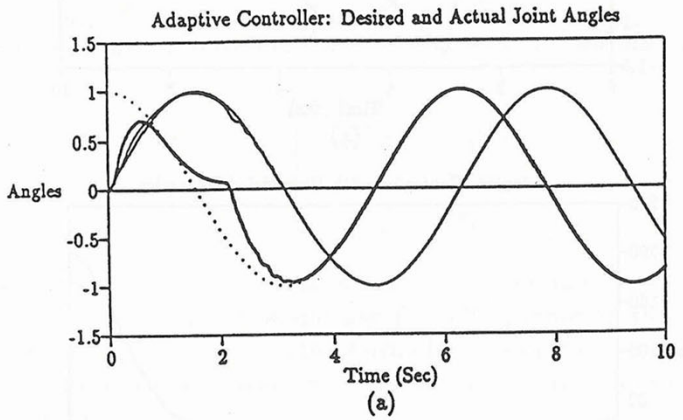
$$\dot{L} = -r^T K_v r \qquad \leq 0$$

Therefore Lyapunov shows that $r(t), \tilde{W}(t)$ are bounded

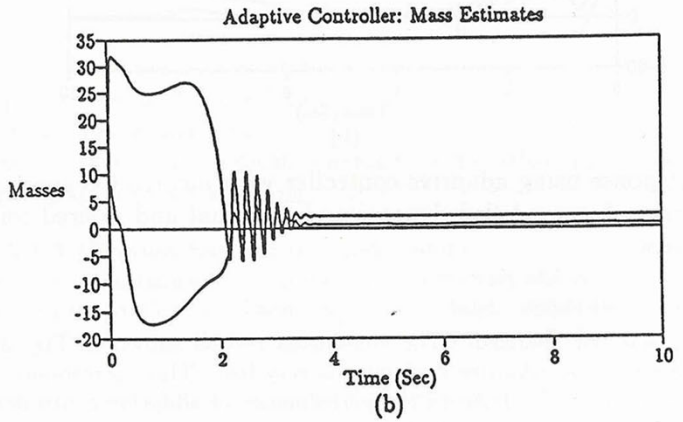
Typical Behavior of Adaptive Controllers

Control errors go to zero and the parameter estimates converge.

This assumes that $f(x) = W^T \phi(x)$ holds exactly, and that there are no disturbances in the system.



Control errors go to zero



Parameter estimates converge

Figure 3.4.3: Response using adaptive controller. (a) Actual and desired joint angles. (b) Mass estimates.

Robust Control

Error dynamics

$$\dot{r} = f(x) - \tau$$

Control input

Unknown nonlinearities

Assume

know a fixed nominal value or estimate $\hat{f}(x)$ for unknown *f(x)*,

estimation error $\tilde{f} = f(x) - \hat{f}(x)$ is bounded like

$$\left\| \tilde{f}(x) \right\| \le F(x)$$

Known bounding function, maybe nonlinear

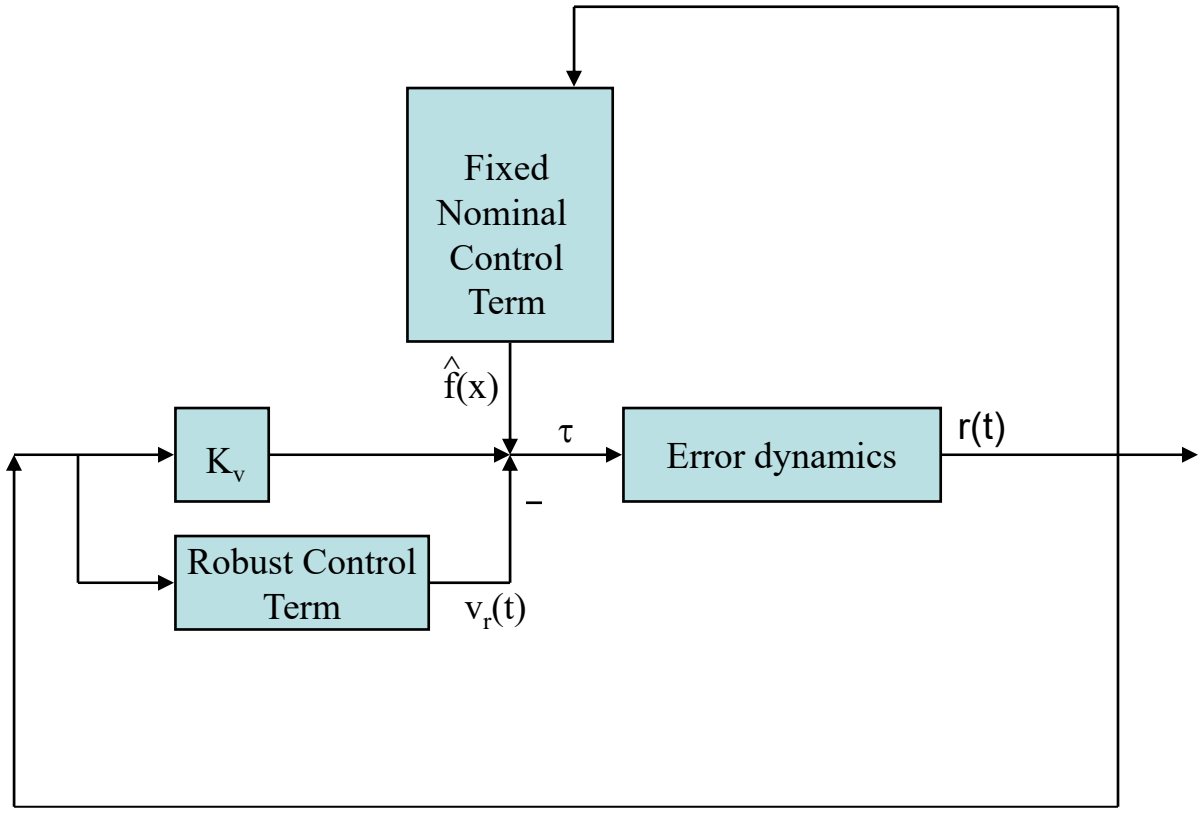
Controller

$$\tau = \hat{f}(x) + K_v r - v_r$$

$$v_r = \begin{cases} -r \dfrac{F(x)}{\|r\|}, & \|r\| \ge \varepsilon \\ -r \dfrac{F(x)}{\varepsilon}, & \|r\| < \varepsilon \end{cases}$$

Robust Controller

A NON Dynamic controller



Robust Controller

Closed-loop Error dynamics

$$\dot{r} = f(x) - \tau = f(x) - \left(\hat{f}(x) + K_v r - v_r \right)$$

$$\dot{r} = \tilde{f}(x) - K_v r + v_r$$

Performance of Robust Controller:

With this control, the closed-loop system is bounded stable with $\|r\|$ bounded with a magnitude near ε

Proof:

$$L = \tfrac{1}{2} r^T r$$

$$\dot{L} = r^T \dot{r}$$

$$\dot{L} = r^T \left(\tilde{f}(x) - K_v r + v_r \right) = -r^T K_v r + r^T \left(\tilde{f}(x) + v_r \right)$$

$$\dot{L} \le -\sigma_{\min}(K_v) \|r\|^2 + \|r\| F(x) + r^T v_r$$

$$v_r = \begin{cases} -r \dfrac{F(x)}{\|r\|}, & \|r\| \ge \varepsilon \\ -r \dfrac{F(x)}{\varepsilon}, & \|r\| < \varepsilon \end{cases}$$

Case 1: $\|r\| \ge \varepsilon$

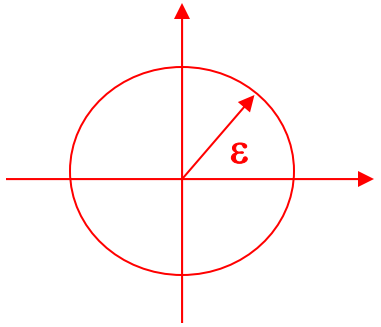
$$\dot{L} \le -\sigma_{\min}(K_v) \|r\|^2 + \|r\| F(x) - \|r\|^2 F(x) / \|r\|$$

$$= -\sigma_{\min}(K_v) \|r\|^2 \qquad \le 0$$

Case 2: $\|r\| < \varepsilon$

$$\dot{L} \le -\sigma_{\min}(K_v) \|r\|^2 + \|r\| F(x) - \|r\|^2 F(x) / \varepsilon$$

$$= -\sigma_{\min}(K_v) \|r\|^2 + \|r\| F(x) \left(1 - \|r\| / \varepsilon\right) \qquad \text{indefinite}$$



Typical Behavior of Robust Controllers

Control error does not go to zero but does indeed stay small.

Robust Control

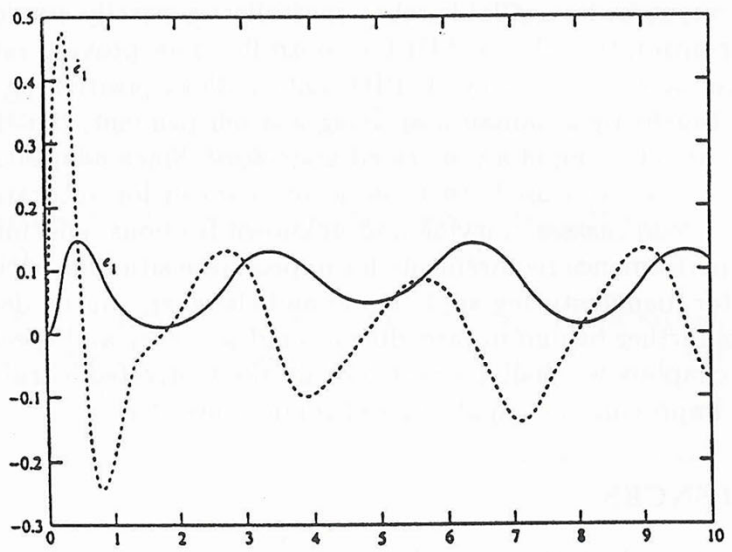
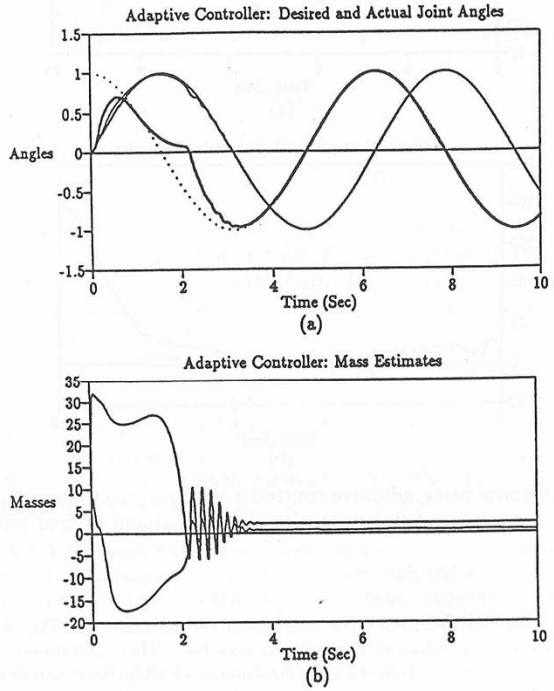


Figure 3.4.6: Typical behavior of robust controller.

Errors are bounded

Adaptive Control

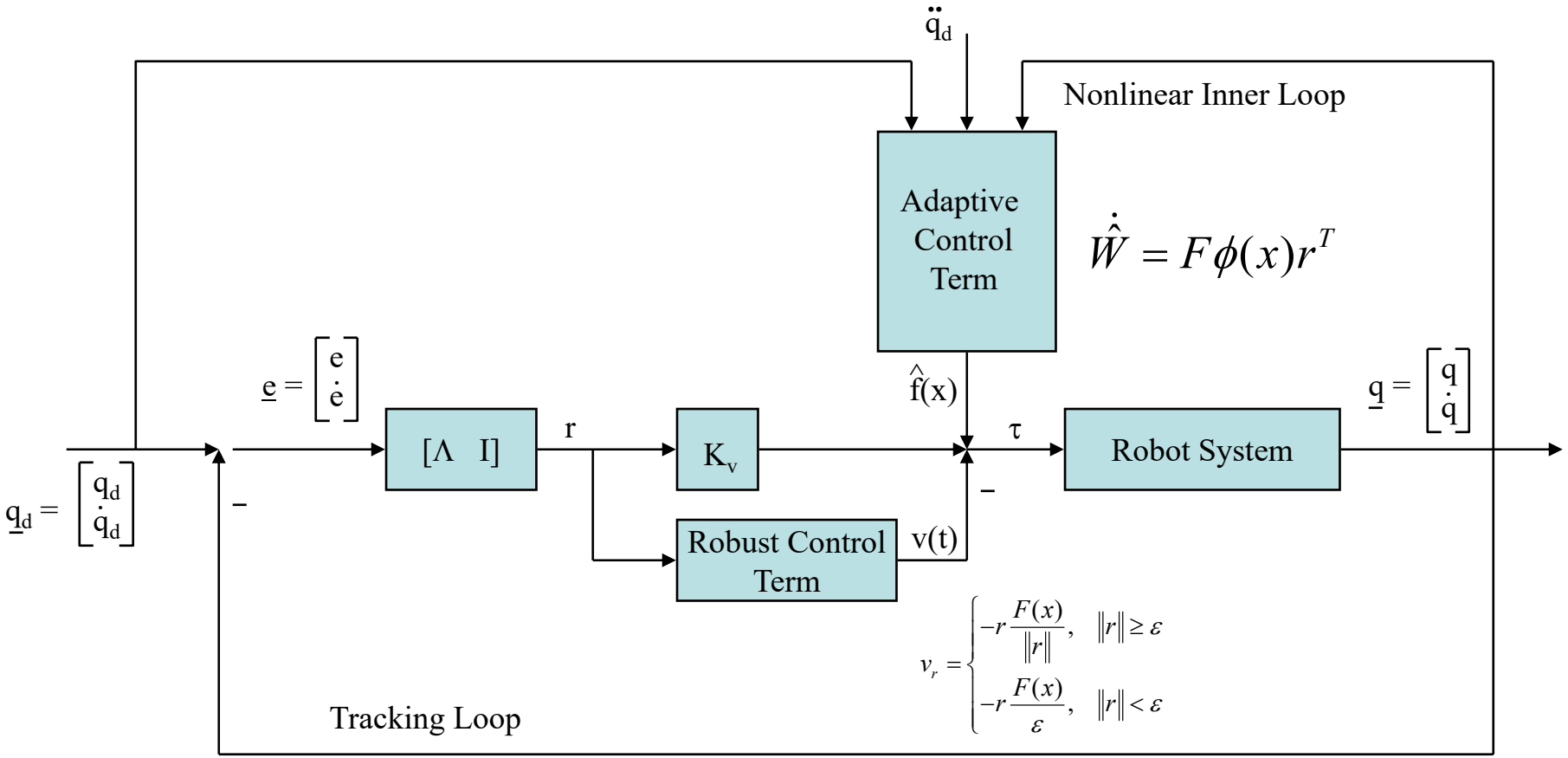


Control errors go to zero

Parameter estimates converge

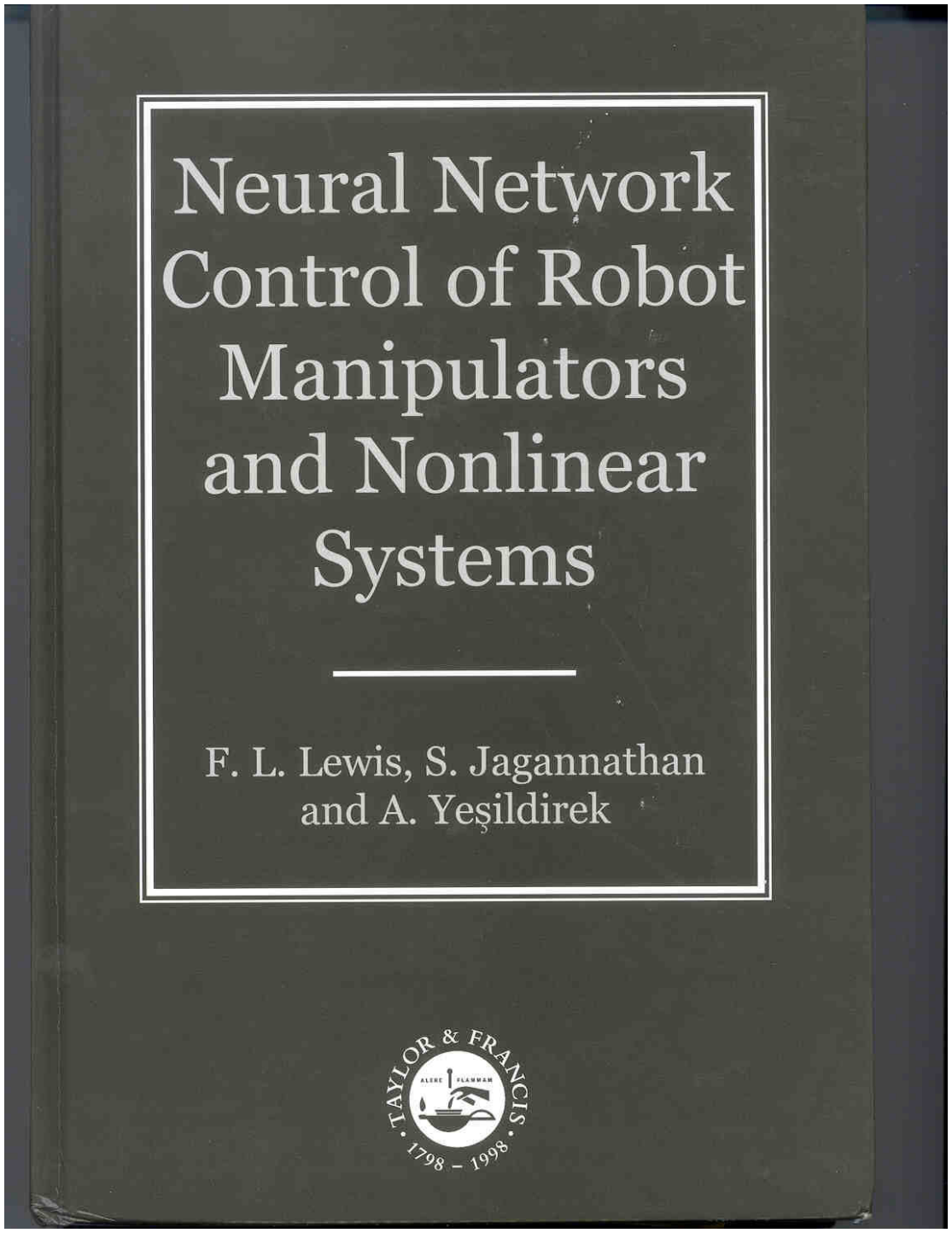
Figure 3.4.3: Response using adaptive controller. (a) Actual and desired joint angles. (b) Mass estimates.

Adaptive plus robust control



Multiloop Nonlinear Controller Structure

Neural Networks for Control



F.L. Lewis, S. Jagannathan, and A. Yesildirek,

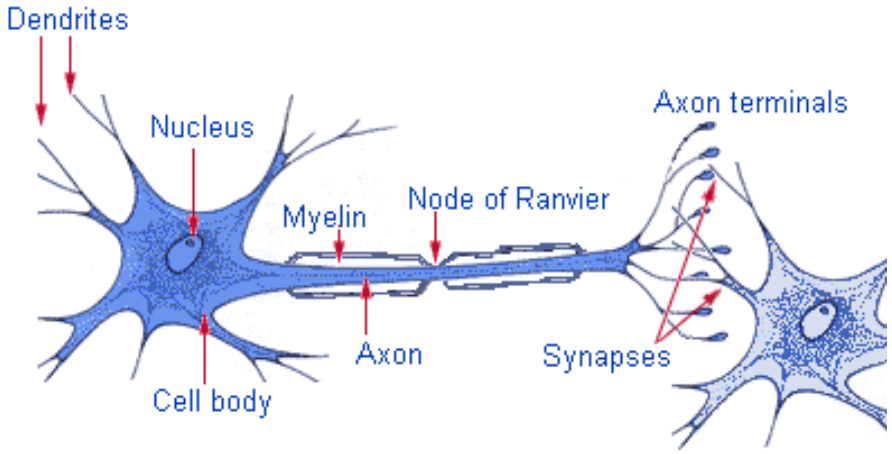
Neural Network Control of Robot Manipulators and Nonlinear Systems,

Taylor and Francis, London, 1999.

NN control in Chapter 4

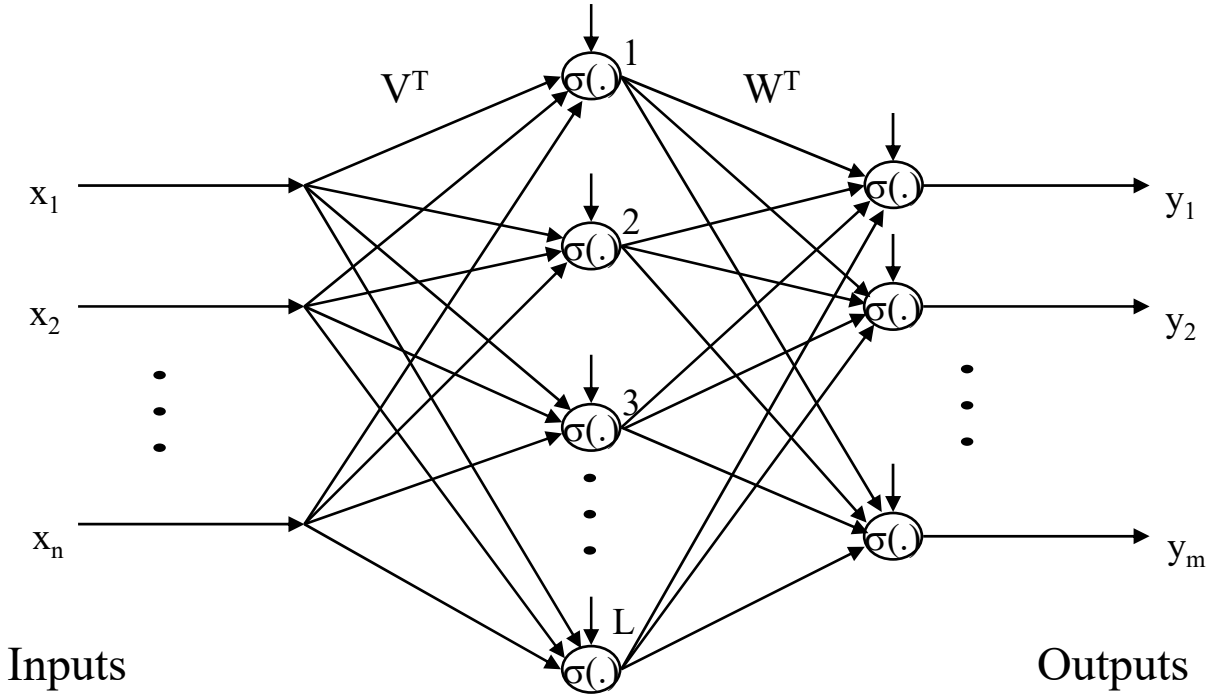
Neural Network Properties

- Learning
- Recall
- Function approximation
- Generalization
- Classification
- Association
- Pattern recognition
- Clustering
- Robustness to single node failure
- Repair and reconfiguration



Nervous system cell.
http://www.sirinet.net/~jgjohnso/index.html

Two-layer feedforward static neural network (NN)



Summation eqs

$$y_i = \sigma\left(\sum_{k=1}^{K} w_{ik} \sigma\left(\sum_{j=1}^{n} v_{kj} x_j + v_{k0} \right) + w_{i0} \right)$$

Matrix eqs

$$y = W^T \sigma(V^T x)$$
$$= W^T \phi(x)$$

Extra freedom to select basis set by tuning first-layer weights V.

Neural Network Universal Approximation Property

Let *f(x)* be any smooth nonlinear function

Then *f(x)* can be approximated by

$$f(x) = \sum_{i=1}^{L} w_i \phi_i(x) \; + \quad \varepsilon(x)$$

For appropriate choice of the basis functions $\phi_i(x), \quad i = 1, L$

Moreover, the approximation error $\varepsilon(x)$ goes uniformly to zero as $L \to \infty$

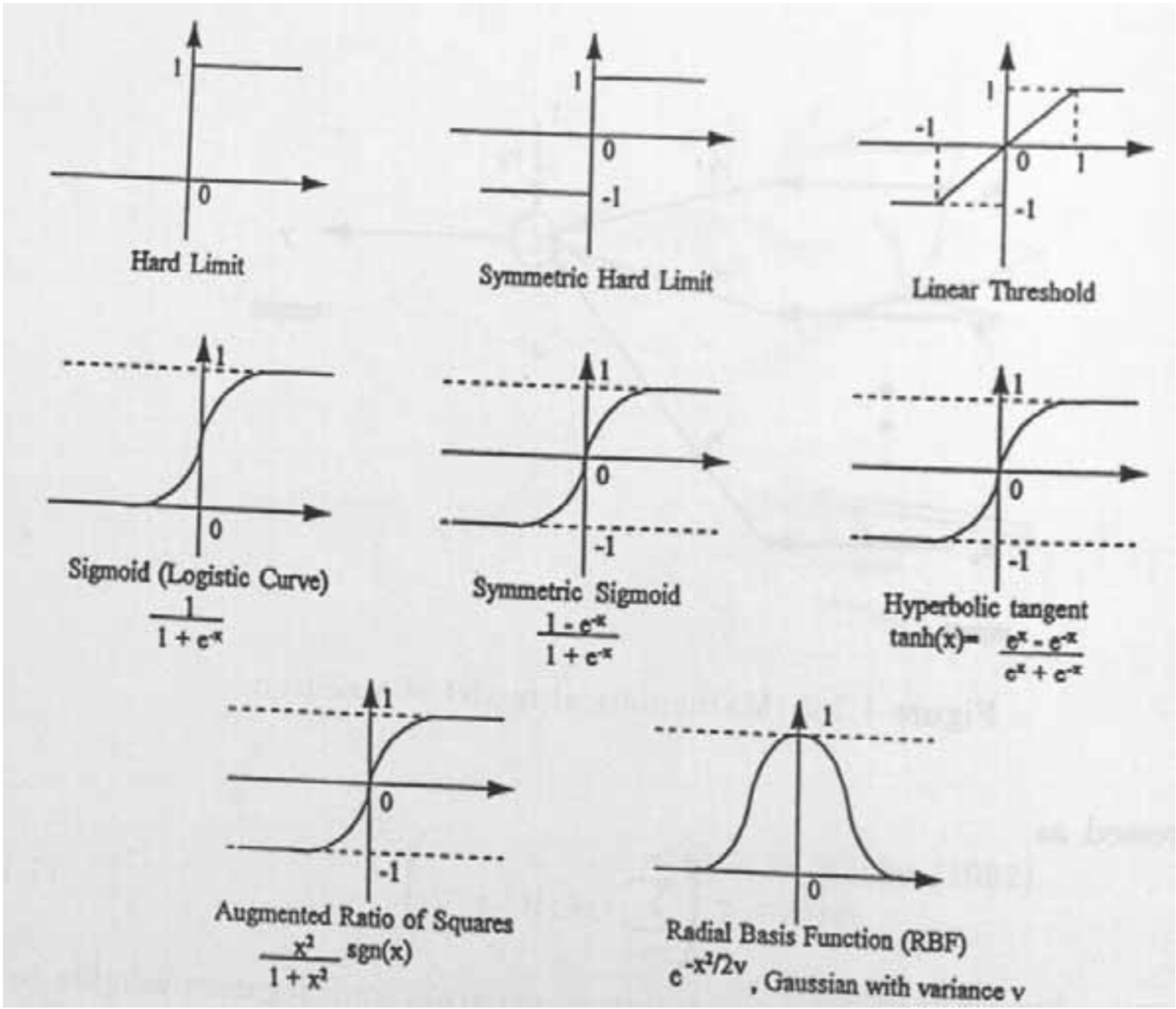
This was shown by Weierstrass for polynomial bases functions (Taylor series)

Hornik and Stinchcomb, Sandberg showed that $\varepsilon(x)$ is bounded on a compact set
For a large class of approximating functions

Need to find the unknown weights w_i

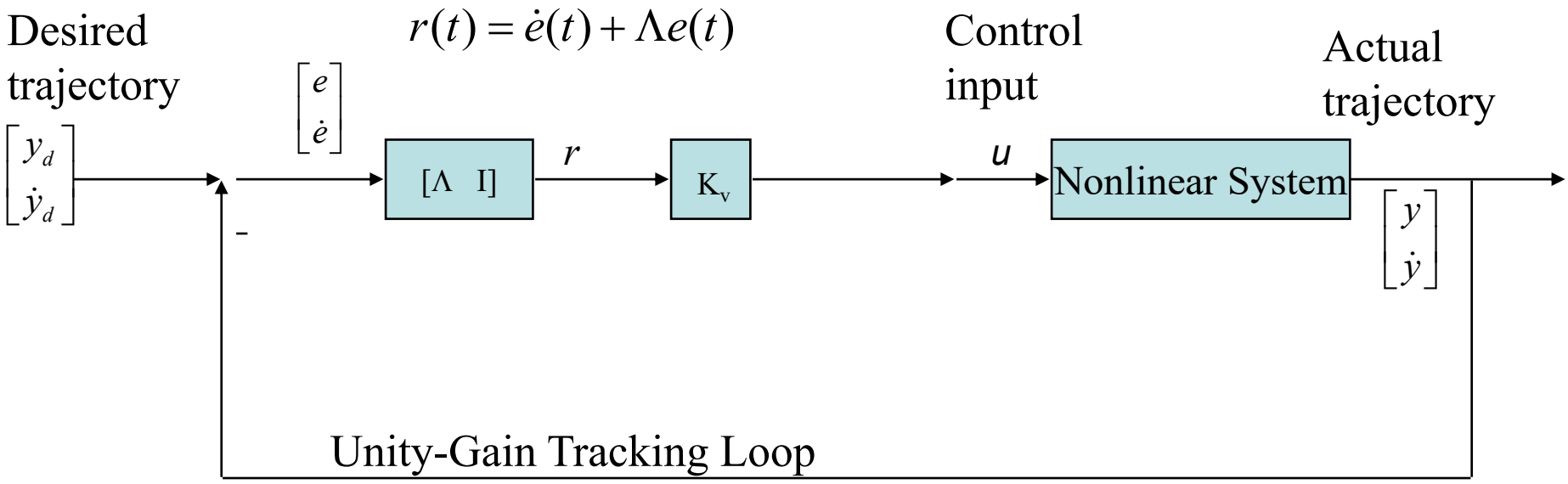
Do this by NN learning – tuning the NN weights

Common activation functions $\sigma(.)$



Industry Standard- PD Controller

Easy to implement with COTS controllers
Fast
Can be implemented with a few lines of code- e.g. MATLAB



But -- Cannot handle-
High-order unmodeled dynamics
Unknown disturbances
High performance specifications for nonlinear systems
Actuator problems such as friction, deadzones, backlash

Feedback Linearization Adaptive Controller

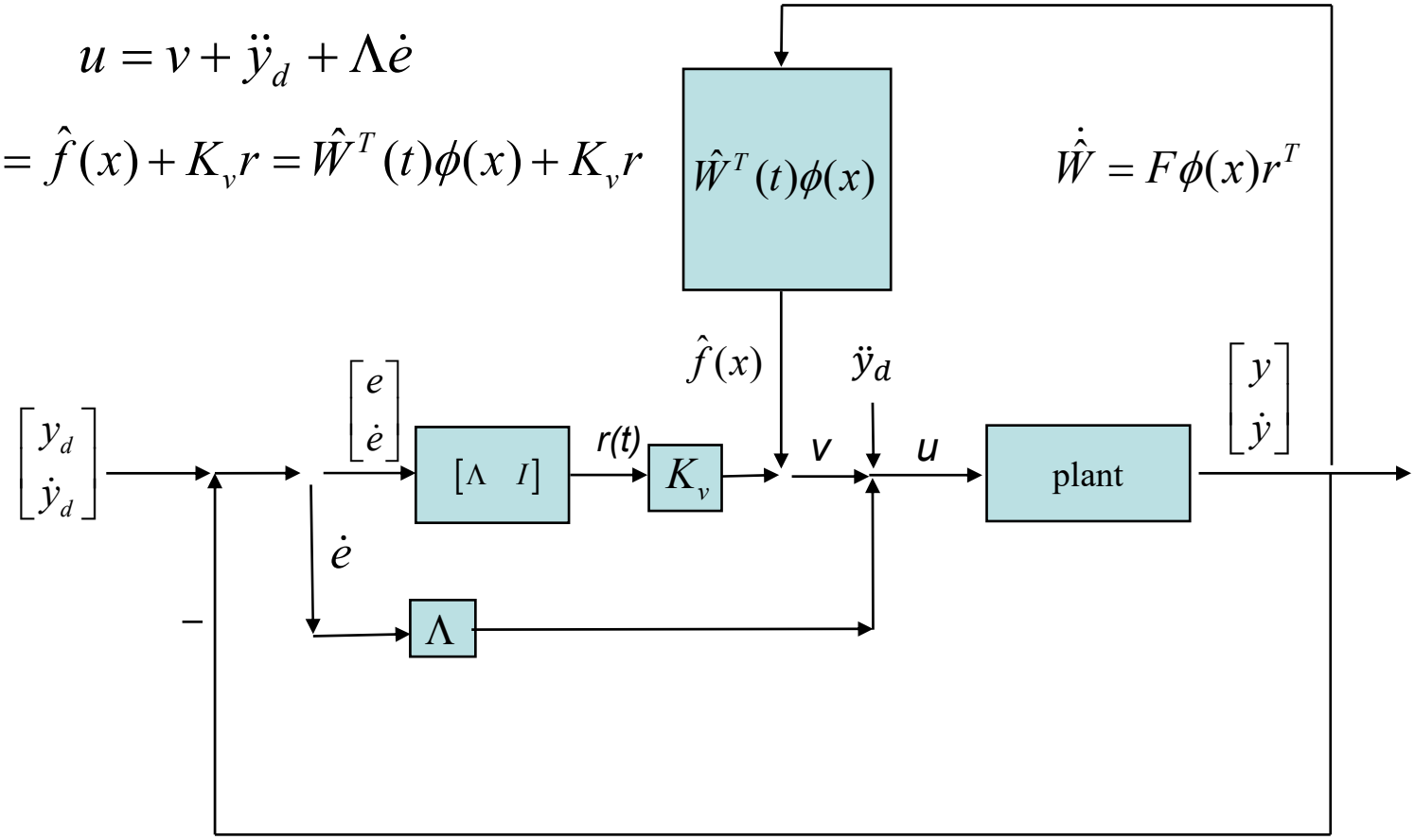
A dynamic controller

$$r = \dot{e} + \Lambda e$$

$$u = v + \ddot{y}_d + \Lambda \dot{e}$$

$$v = \hat{f}(x) + K_v r = \hat{W}^T(t)\phi(x) + K_v r$$

$$\dot{\hat{W}} = F\phi(x) r^T$$



The equations give the FB controller structure

Control System Design Approach

Robot dynamics $$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + + F(\dot{q}) + G(q) + \tau_d = \tau$$

Tracking Error definition $$e(t) = q_d(t) - q(t)$$

Siding variable $$r = \dot{e} + \Lambda e$$

Error dynamics $$M\dot{r} = M(\ddot{e} + \Lambda\dot{e}) = \mathrm{M}(\ddot{q}_d(t) - \ddot{q}(t) + \Lambda\dot{e})$$

$$M\dot{r} = -V_m r + f(x) + \tau_d - \tau$$

Where the unknown function is

$$f(x) = M(q)(\ddot{q}_d + \Lambda\dot{e}) + V_m(q,\dot{q})(\dot{q}_d + \Lambda e) + F(\dot{q}) + G(q)$$

Tracking error

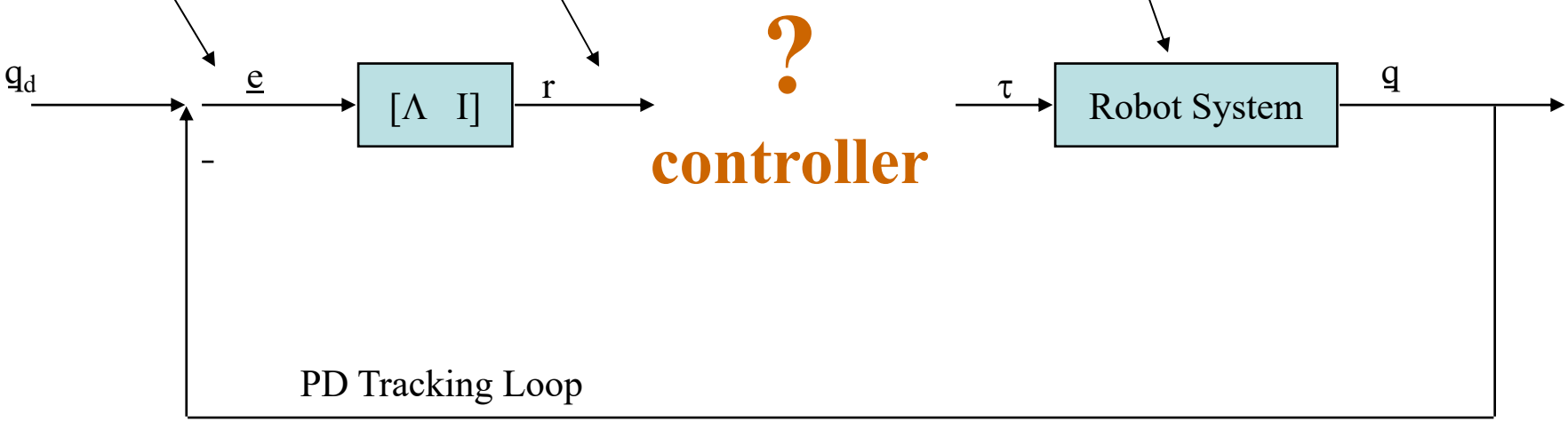
$$e(t) = q_d(t) - q(t)$$

Robot dynamics

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_d = \tau$$

Sliding variable

$$r = \dot{e} + \Lambda e$$



The equations give the FB controller structure

Control System Design Approach

Robot dynamics $$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_d = \tau$$

Tracking Error definition $$e(t) = q_d(t) - q(t) \qquad r = \dot{e} + \Lambda e$$

Error dynamics $$M\dot{r} = -V_m r + f(x) + \tau_d - \tau$$

UNKNOWN FN.

Universal Approximation Property

Approx. unknown function by NN $$f(x) = W^T \sigma(V^T x) + \varepsilon$$

Define control input $$\tau = \hat{W}^T \sigma(\hat{V}^T x) + K_v r - v$$

Closed-loop dynamics

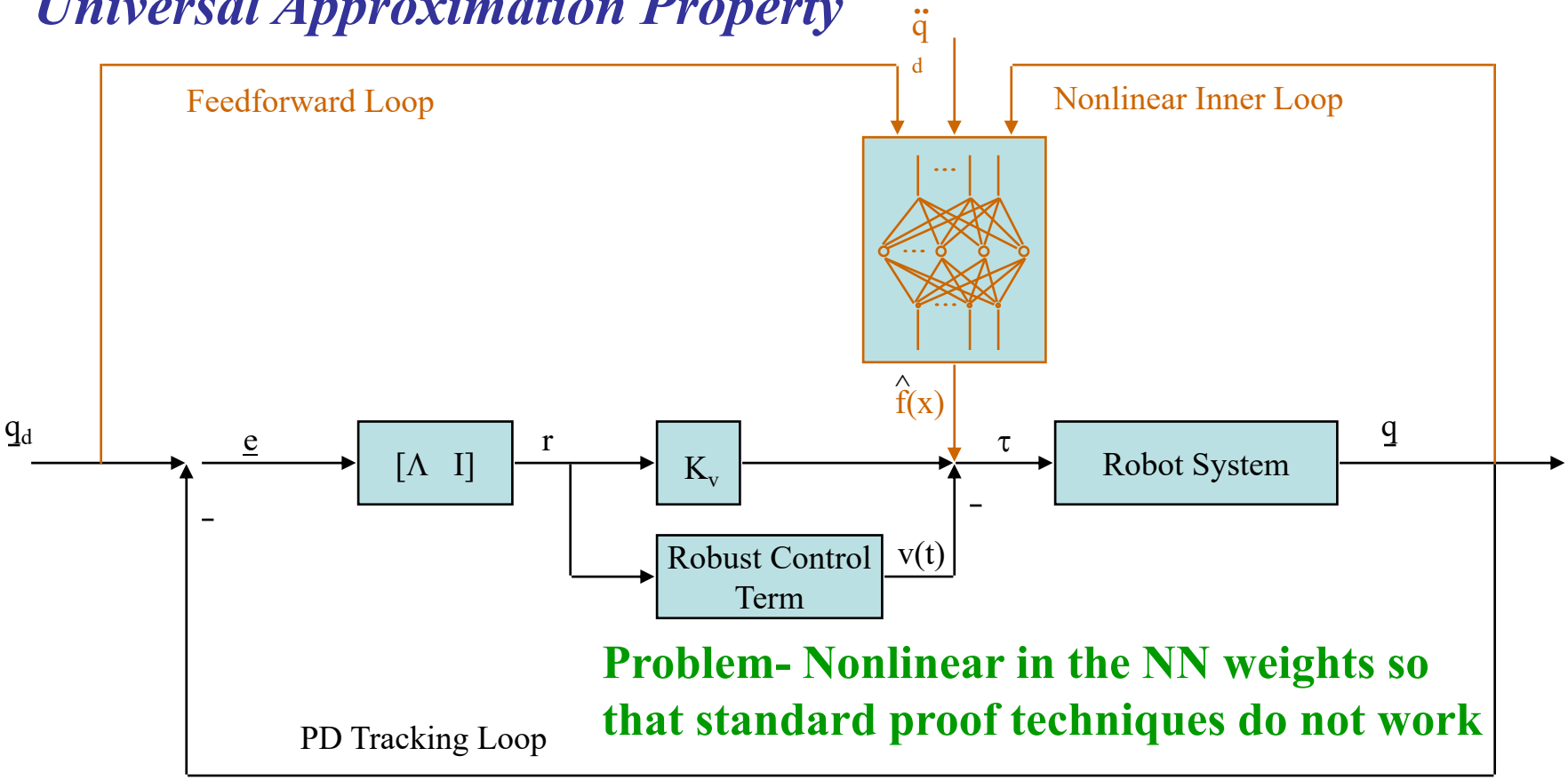
$$M\dot{r} = -V_m r - K_v r + W^T \sigma(V^T x) + \varepsilon - \hat{W}^T \sigma(\hat{V}^T x) + \tau_d + v(t)$$

$$M\dot{r} = -V_m r - K_v r + \tilde{f} + \tau_d + v(t)$$

Neural Network Robot Controller

Universal Approximation Property

Feedback linearization



Problem- Nonlinear in the NN weights so that standard proof techniques do not work

Easy to implement with a few more lines of code

Learning feature allows for on-line updates to NN memory as dynamics change

Handles unmodelled dynamics, disturbances, actuator problems such as friction

NN universal basis property means no regression matrix is needed

Nonlinear controller allows faster & more precise motion

Adaptive part

Robust part

Theorem 1 (NN Weight Tuning for Stability)

Let the desired trajectory $q_d(t)$ and its derivatives be bounded. Let the initial tracking error be within a certain allowable set U. Let Z_M be a known upper bound on the Frobenius norm of the unknown ideal weights Z.

Take the control input as

$$\tau = \hat{W}^T \sigma(\hat{V}^T x) + K_v r - v \qquad \text{with} \qquad v(t) = -K_Z(\|Z\|_F + Z_M)r .$$

Let weight tuning be provided by

$$\dot{\hat{W}} = F\hat{\sigma}r^T - F\hat{\sigma}'\hat{V}^T x r^T - \kappa F\|r\|\hat{W} , \qquad \dot{\hat{V}} = Gx(\hat{\sigma}'^T \hat{W}r)^T - \kappa G\|r\|\hat{V}$$

with any constant matrices $F = F^T > 0, G = G^T > 0$, and scalar tuning parameter $\kappa > 0$. Initialize the weight estimates as $\hat{W} = 0, \hat{V} = random$.

Then the filtered tracking error $r(t)$ and NN weight estimates $\hat{W}, \hat{V}$ are uniformly ultimately bounded. Moreover, arbitrarily small tracking error may be achieved by selecting large control gains K_v.

Backprop terms-
Werbos

Extra robustifying terms-
Narendra's e-mod extended to NLIP systems

Stability Proof based on Lyapunov Extension

Define a Lyapunov Energy Function

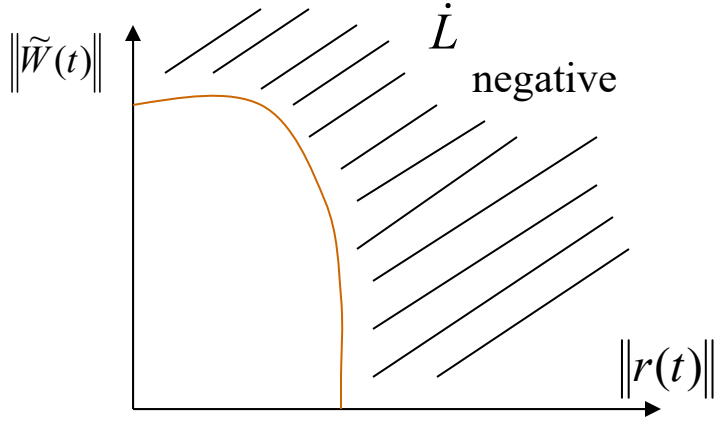
$$L = \tfrac{1}{2} r^T M r + \tfrac{1}{2} tr(\tilde{W}^T \tilde{W}) + \tfrac{1}{2} tr(\tilde{V}^T \tilde{V})$$

Differentiate

$$\dot{L} = -r^T K_v r + \tfrac{1}{2} r^T (\dot{M} - 2V_m) r$$
$$+ tr\, \tilde{W}^T (\dot{\tilde{W}} + \hat{\sigma} r^T - \hat{\sigma}' \hat{V}^T x r^T)$$
$$+ tr\, \tilde{V}^T (\dot{\tilde{V}} + x r^T \hat{W}^T \hat{\sigma}') + r^T (w + v)$$

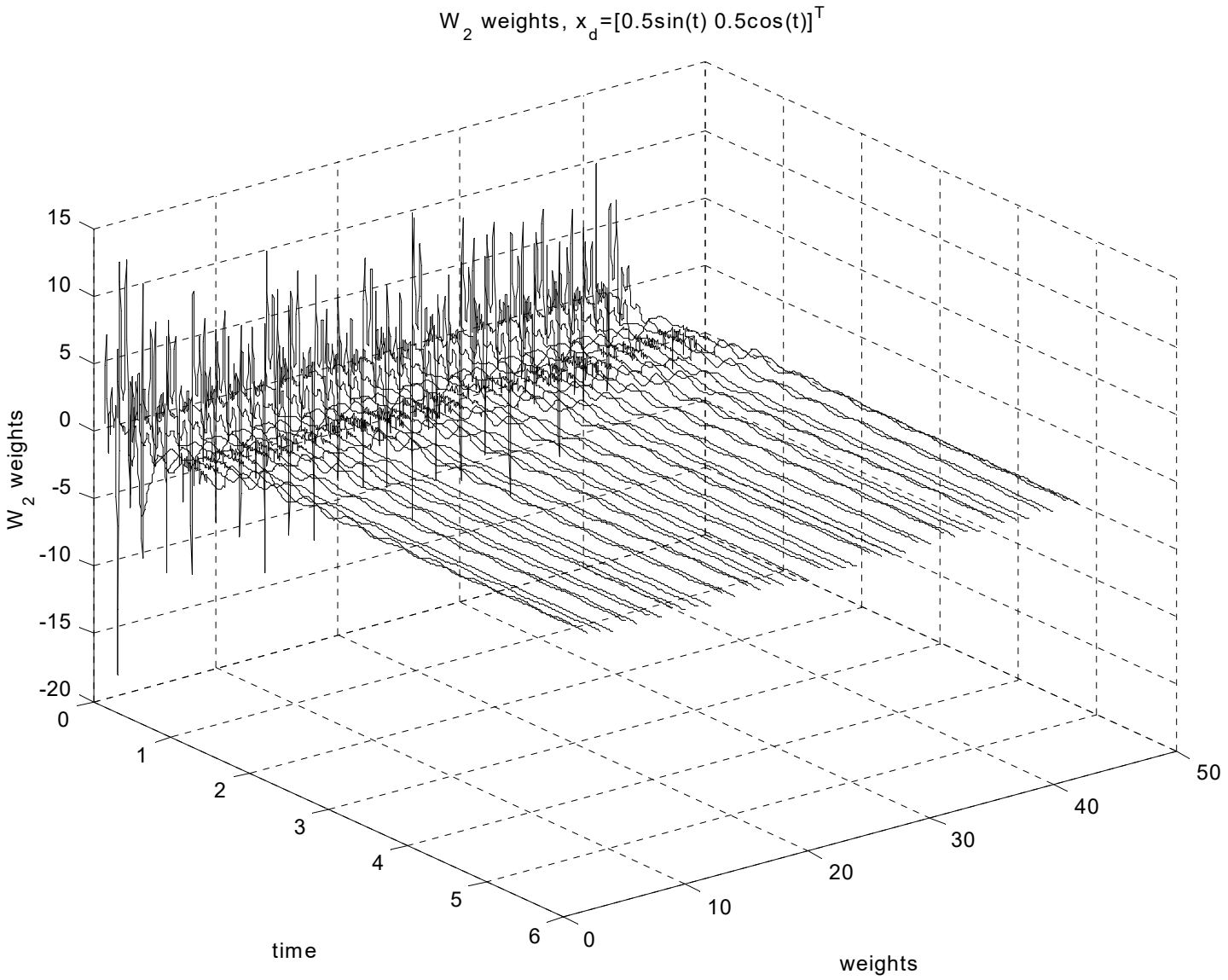
Problems—
1. How to characterize the NN weight errors as 'small'?- use Frobenius Norm
2. Nonlinearity in the parameters requires extra care in the proof

Using certain special tuning rules, one can show that the energy derivative is negative outside a compact set.



This proves that all signals are bounded

UUB- uniform ultimate boundednes



NN weights converge to the best learned values for the given system

Structured Control NN

$$M(\dot{q})\ddot{q}+V_m(q,\dot{q})\dot{q}+G(q)+F(\dot{q})+\tau_d = B(q)\tau$$

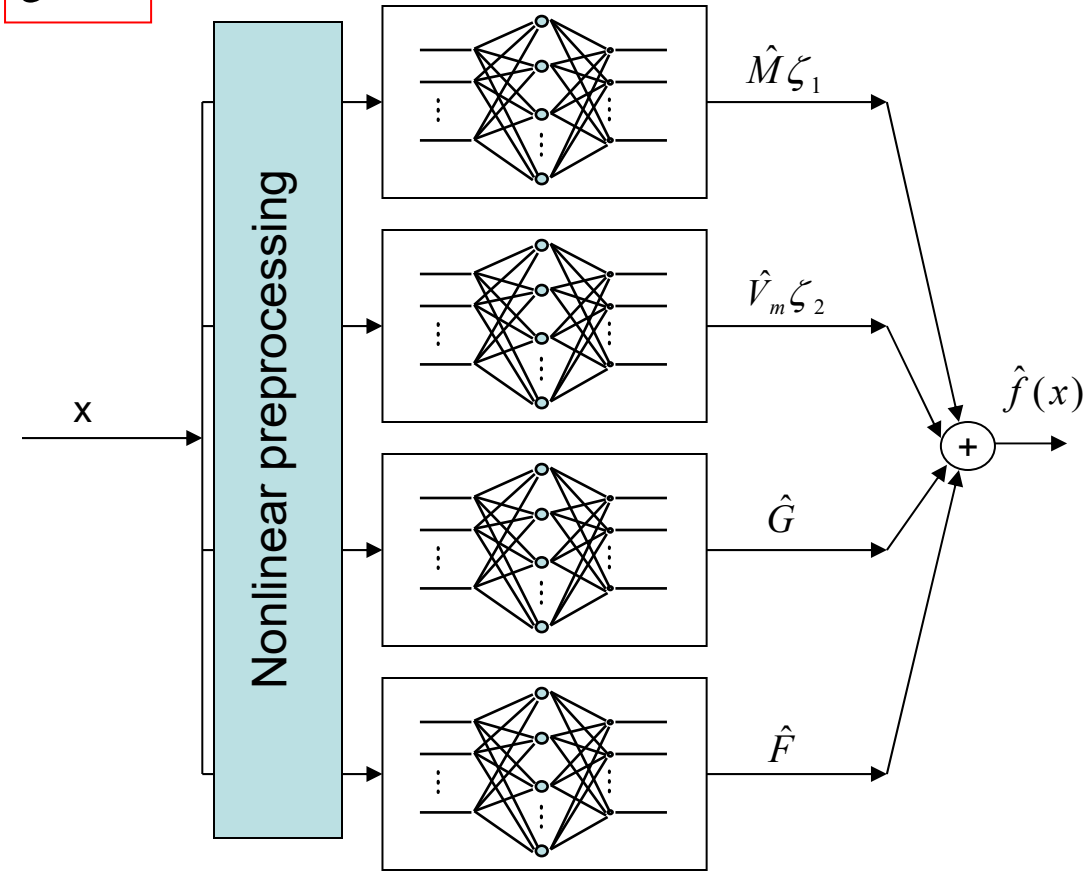
$\sin(q_i), \cos(q_i)$

$\dot{q}_i\dot{q}_j$ coriolis

$\dot{q}_i^2$ centripetal

$\mathrm{sgn}(\dot{q}_i)$

$e^{-|q_i|}$

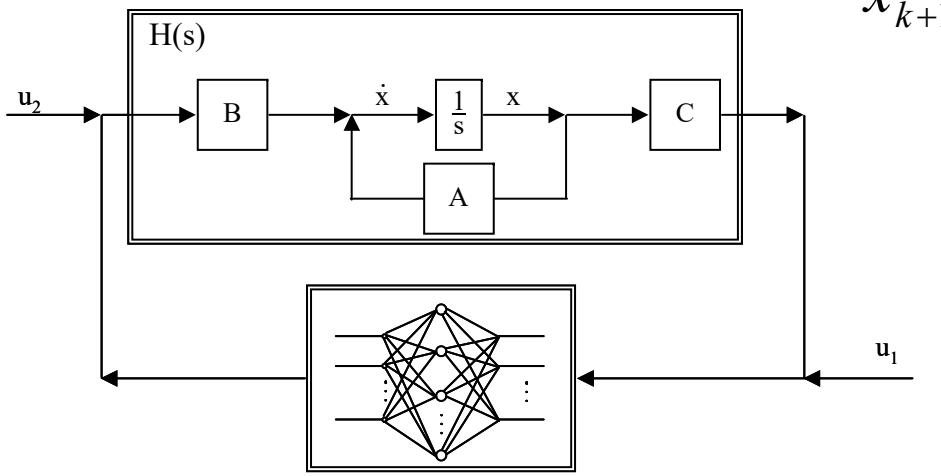


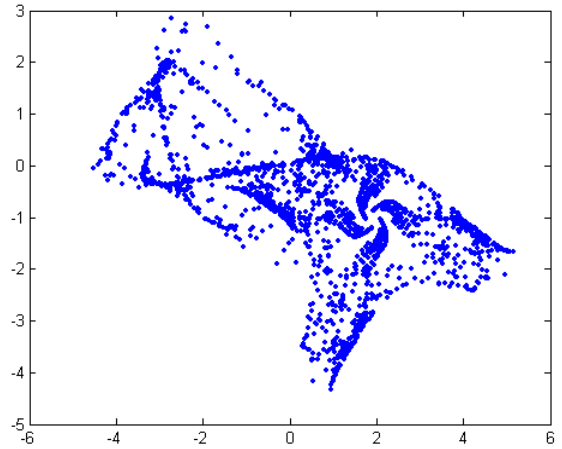
Partitioned neural network

Chaos in Dynamic Neural Networks

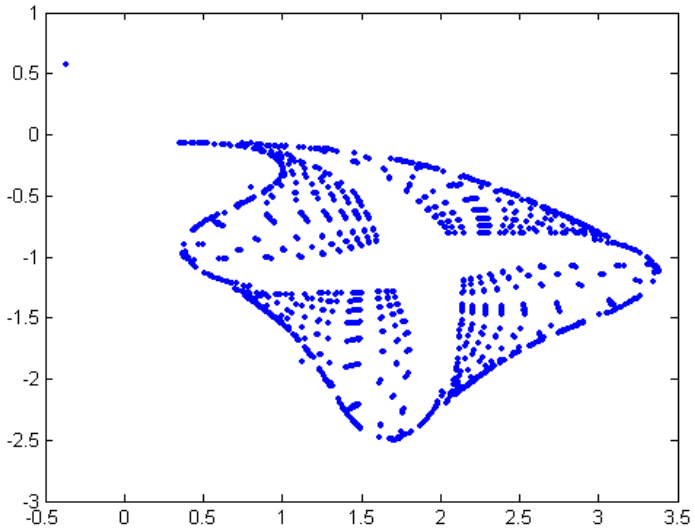
Recurrent or Dynamic NN

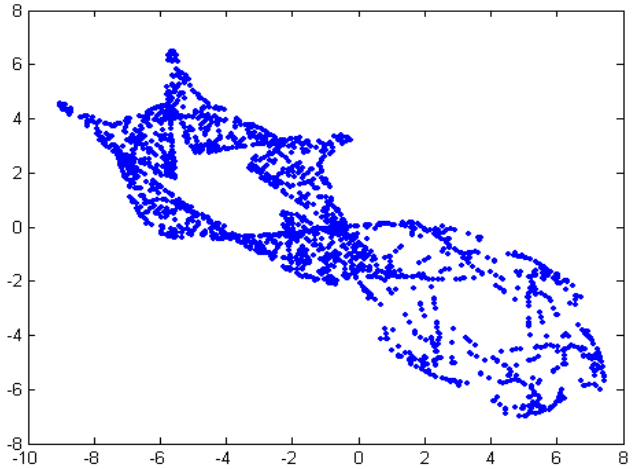
$$x_{k+1} = Ax_k + W^T \sigma(V^T x_k) + u_k$$





c.f. Ron Chen





Jun Wang

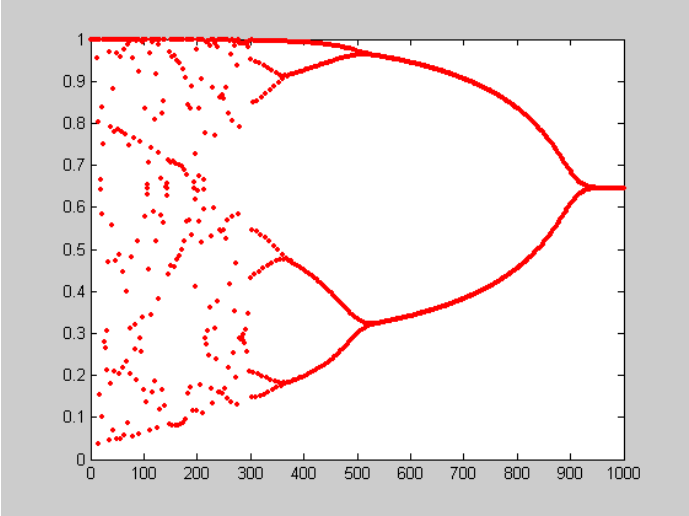
$$z_{k+1} = \beta z_k$$

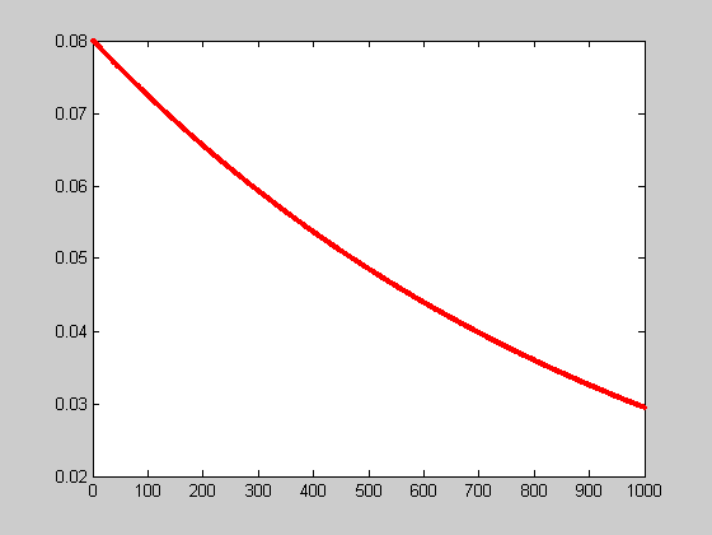
$$y_{k+1} = \alpha y_k + g - z_k\left(\frac{1}{1+e^{-y_k/\rho}} - I\right)$$

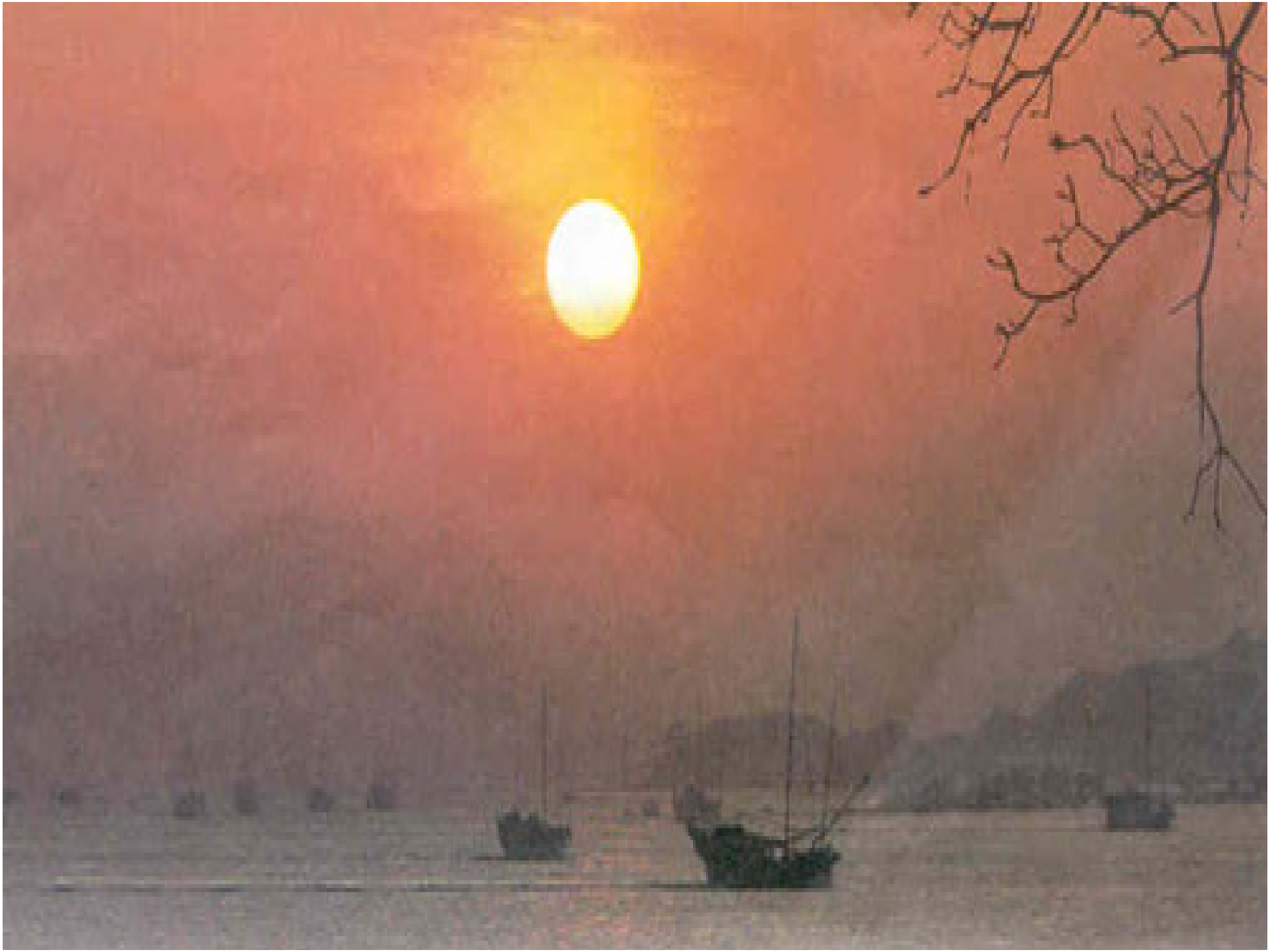
%MATLAB file for chaotic NN
from **Jun Wang's** paper

```
function [ki,x,y,z]=tcnn(N);
y(1)= rand; ki(1)=1; z(1)= 0.08;
a=0.9; e= 1/250; Io=0.65;
g= 0.0001; b=0.001;

for k=1: N-1;
   ki(k+1)= k+1;
   x(k)= 1/(1+exp(-y(k)/e));
   y(k+1)= a*y(k) + g -
z(k)*(x(k) - Io);
   z(k+1)= (1-b)*z(k);
end
x(N)= 1/(1+exp(-y(N)/e));
```

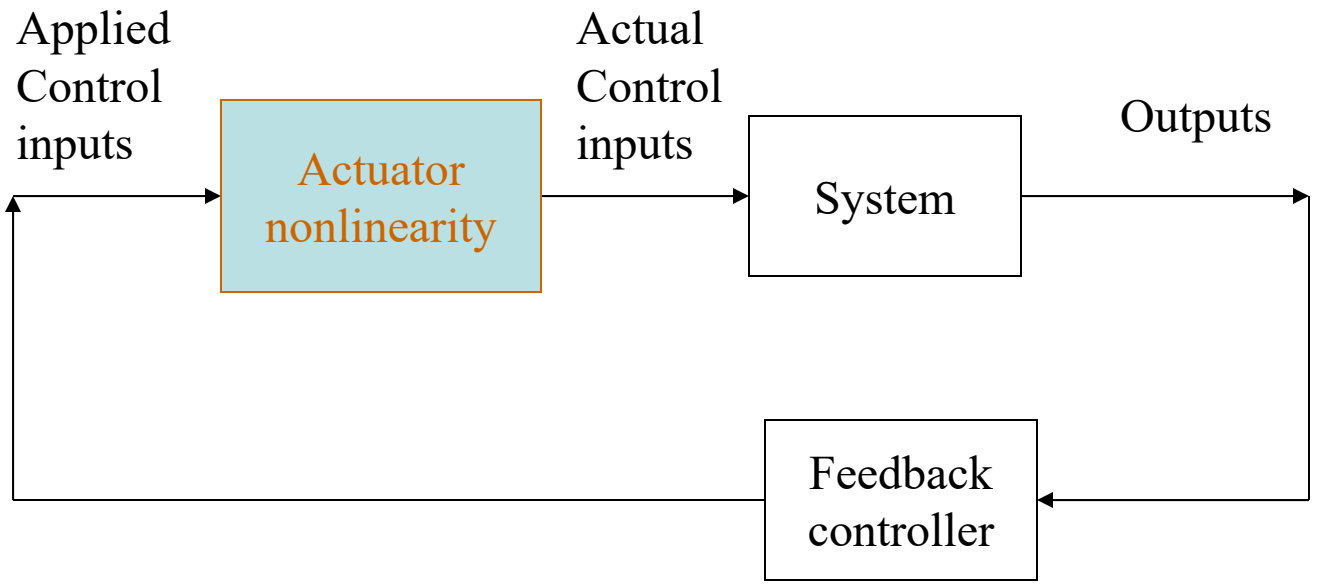




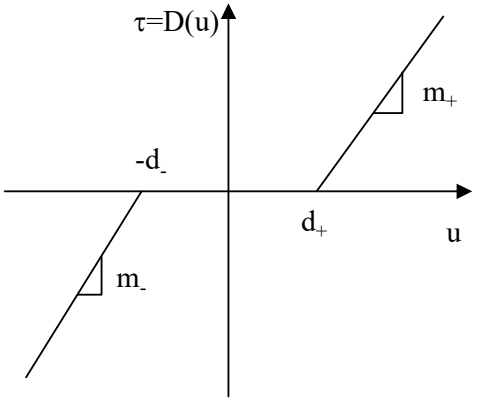


Actuator Dynamics

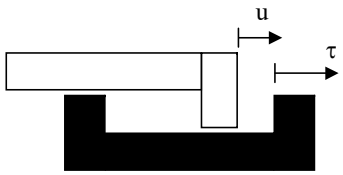
Actuator Nonlinearities

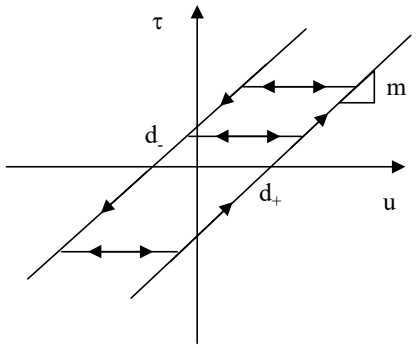


Friction



Deadzone





Backlash

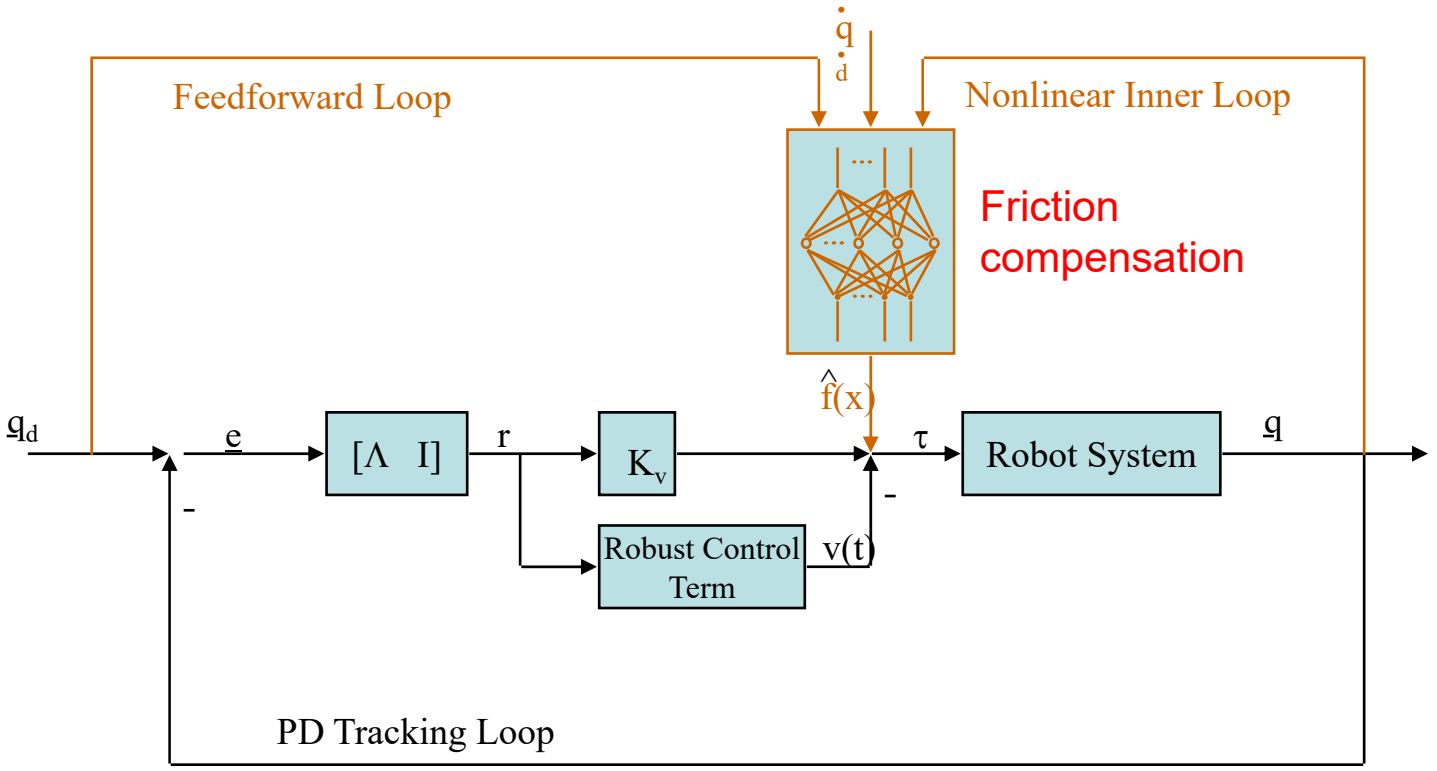
Friction Compensation

Lagrangian System Dynamics – e.g. Robot

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_d = \tau$$

Friction

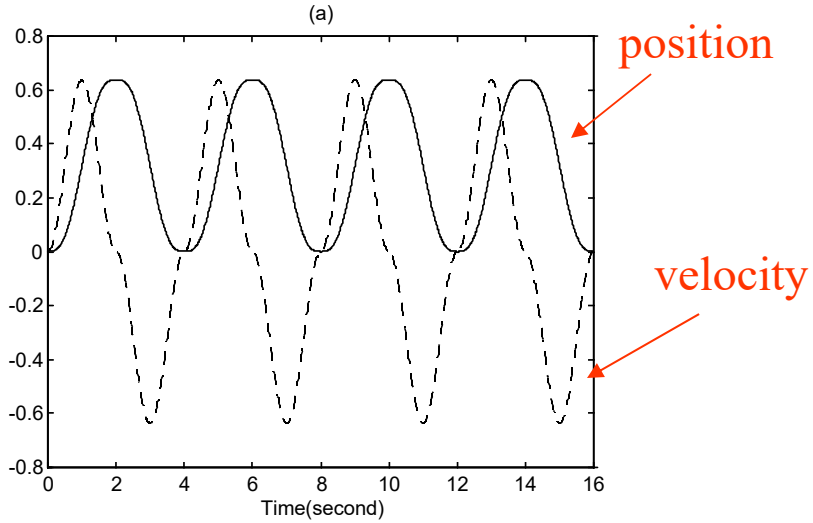
Use the standard NN from Lecture 1

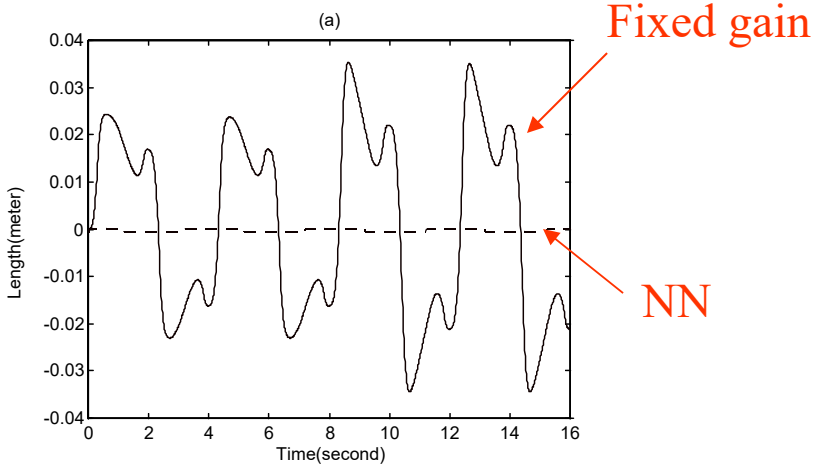


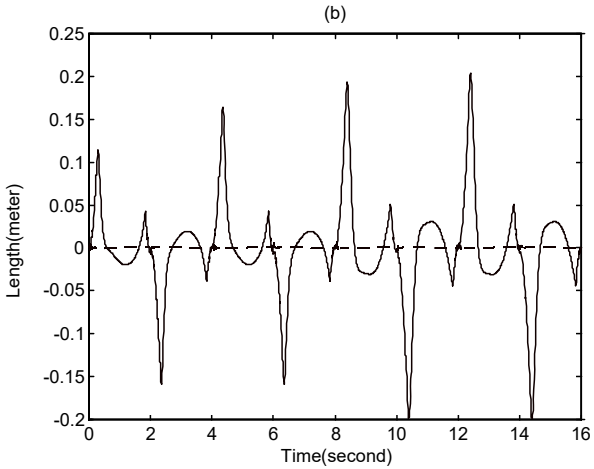
NN Friction Compensator

Trajectory Tracking Controller

Desired trajectory







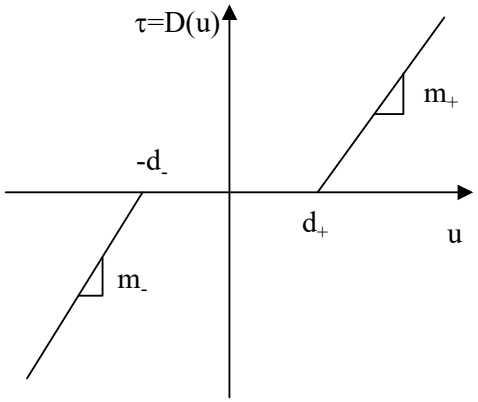
Position

Velocity

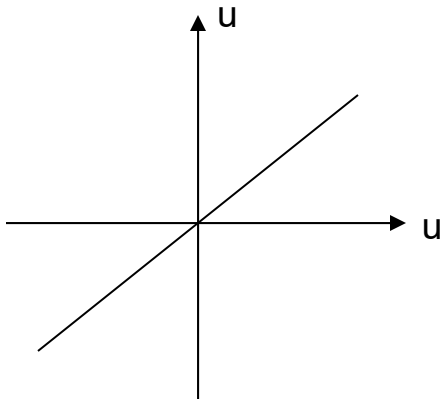
Tracking errors- solid = fixed gain controller, dashed= NN controller

Deadzone

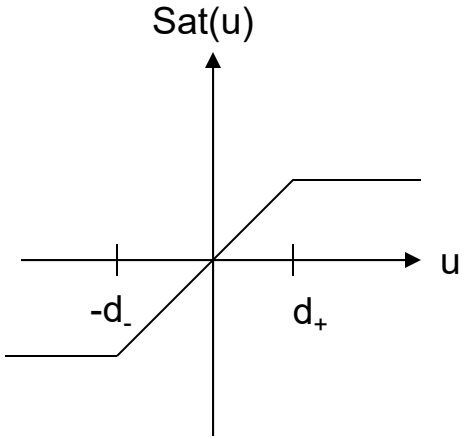
Key fact



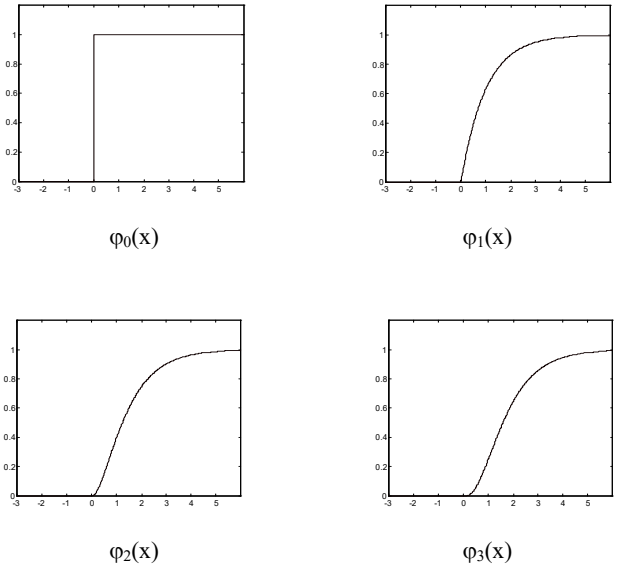
=



−

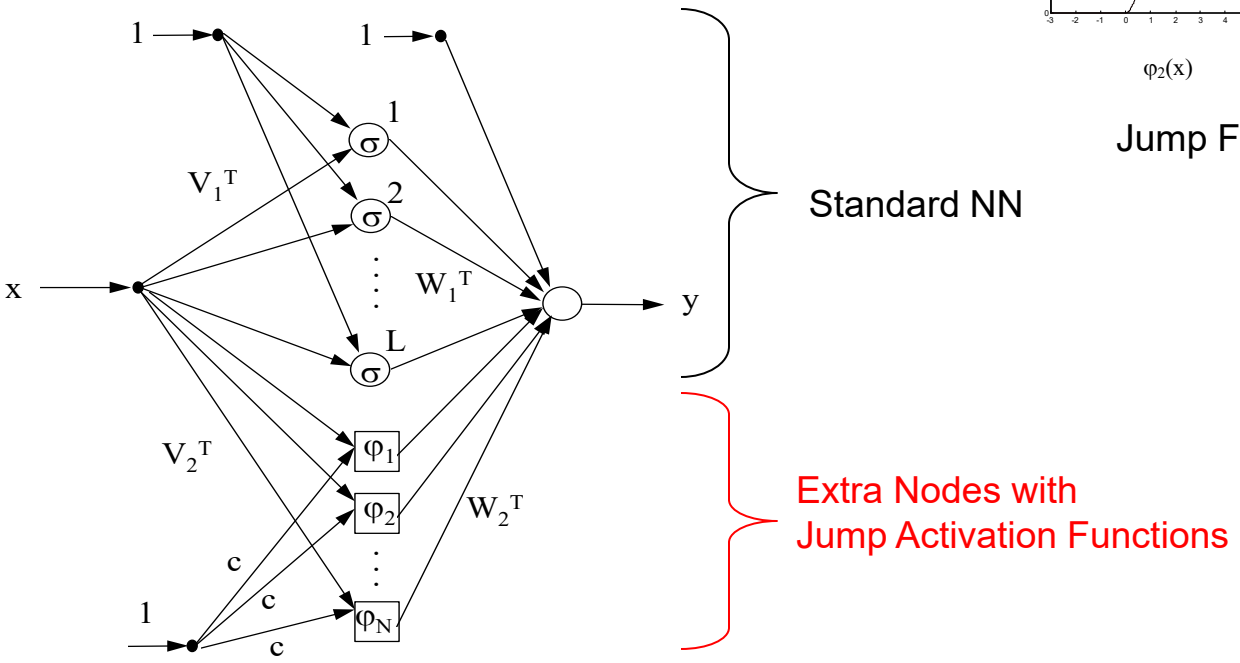


NN Approximation of Functions with Jumps



Jump Function Basis Set

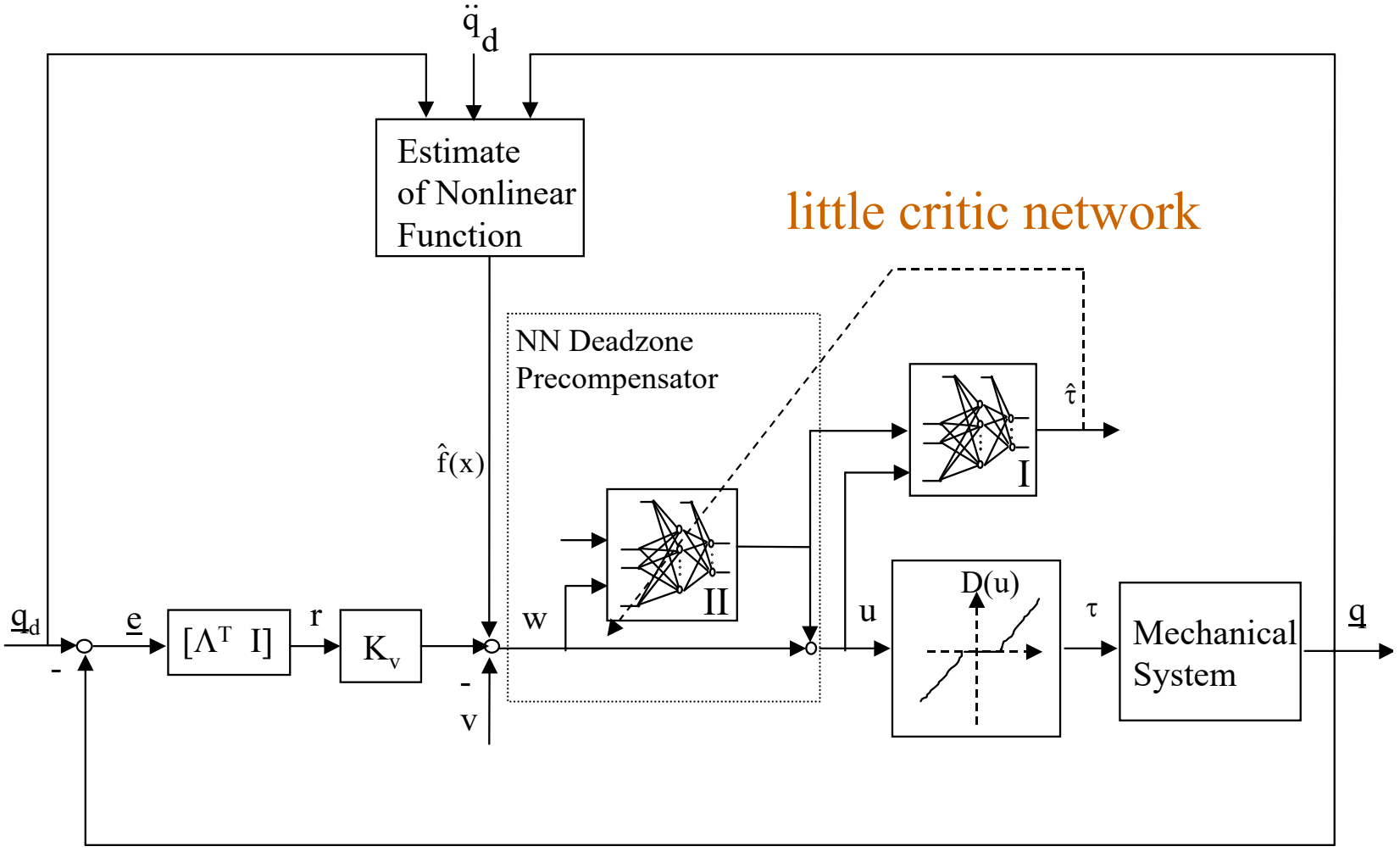
Augmented NN



Standard NN

Extra Nodes with
Jump Activation Functions

NN in Feedforward Loop- Deadzone Compensation

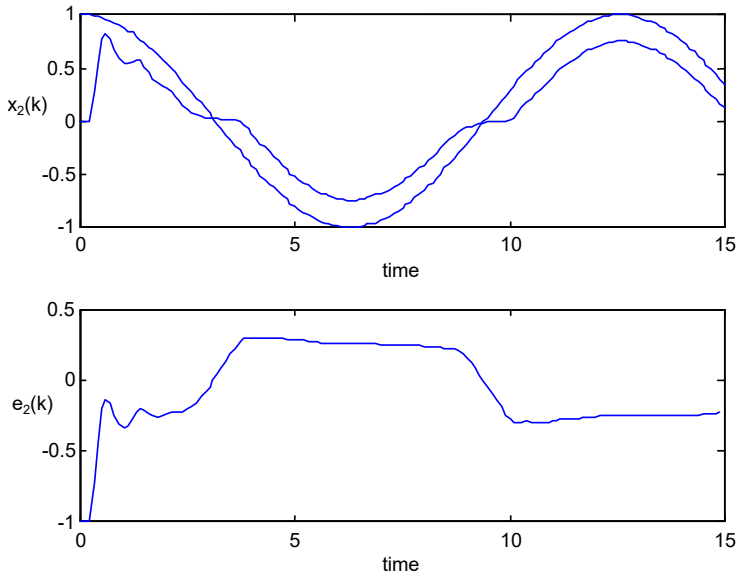


$$\dot{\hat{W}}_i = T\sigma_i(U_i^T w)r^T\hat{W}^T\sigma'(U^T u)U^T - k_1 T\|r\|\hat{W}_i - k_2 T\|r\|\|\hat{W}_i\|\hat{W}_i$$

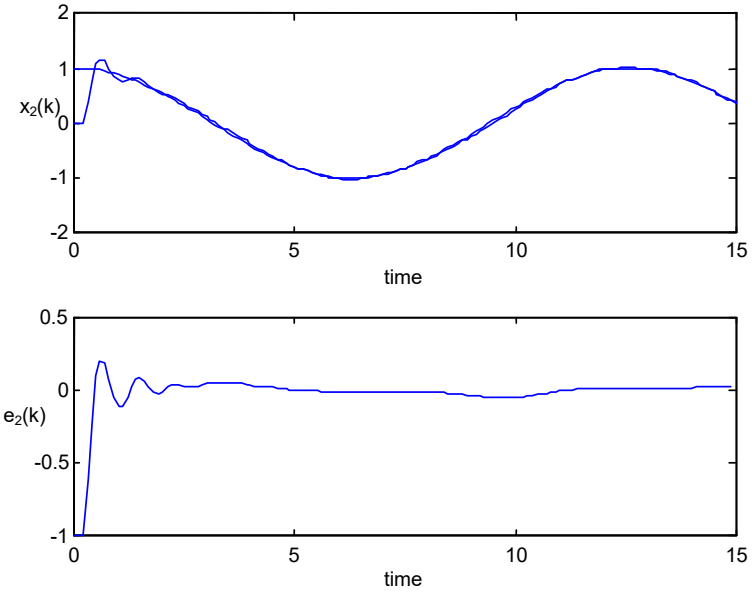
$$\dot{\hat{W}} = -S\sigma'(U^T u)U^T\hat{W}_i\sigma_i(U_i^T w)r^T - k_1 S\|r\|\hat{W}$$

Acts like a 2-layer NN
With enhanced
backprop tuning !

Performance Results



PD control-
deadzone chops out the middle

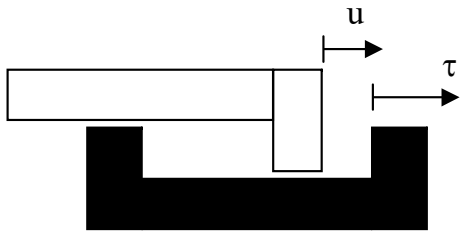


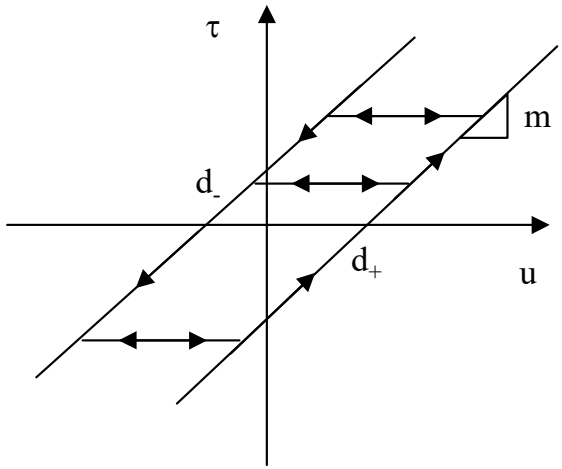
NN control fixes the problem

Backlash

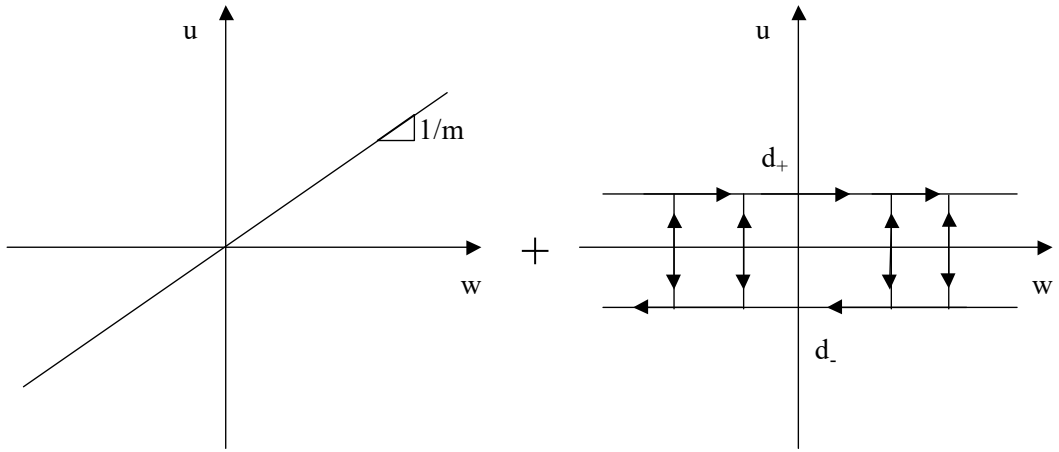
$$\dot{\tau} = B(\tau, u, \dot{u})$$

A dynamic nonlinearity



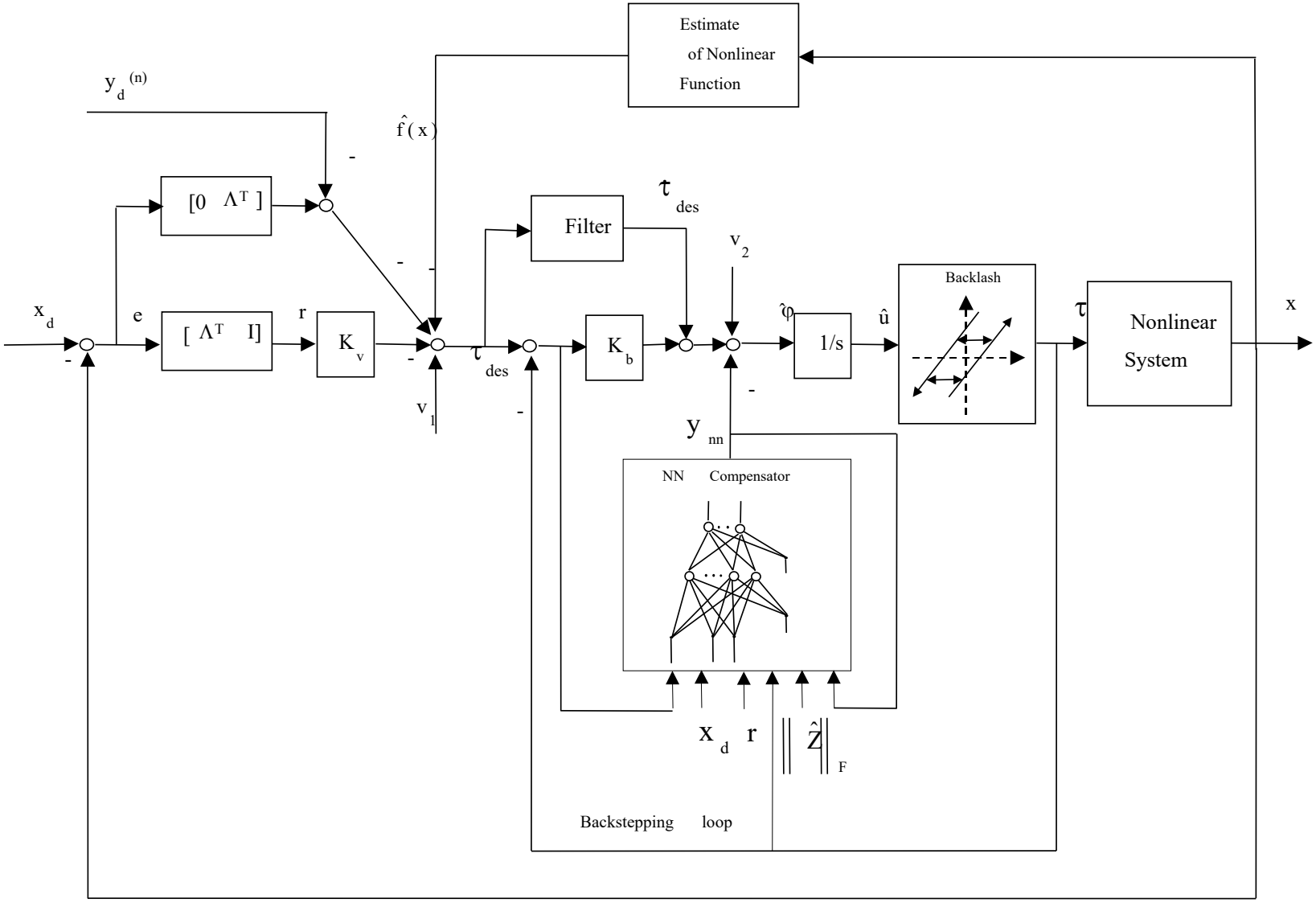


$$\dot{\tau} = B(\tau, u, \dot{u}) =$$



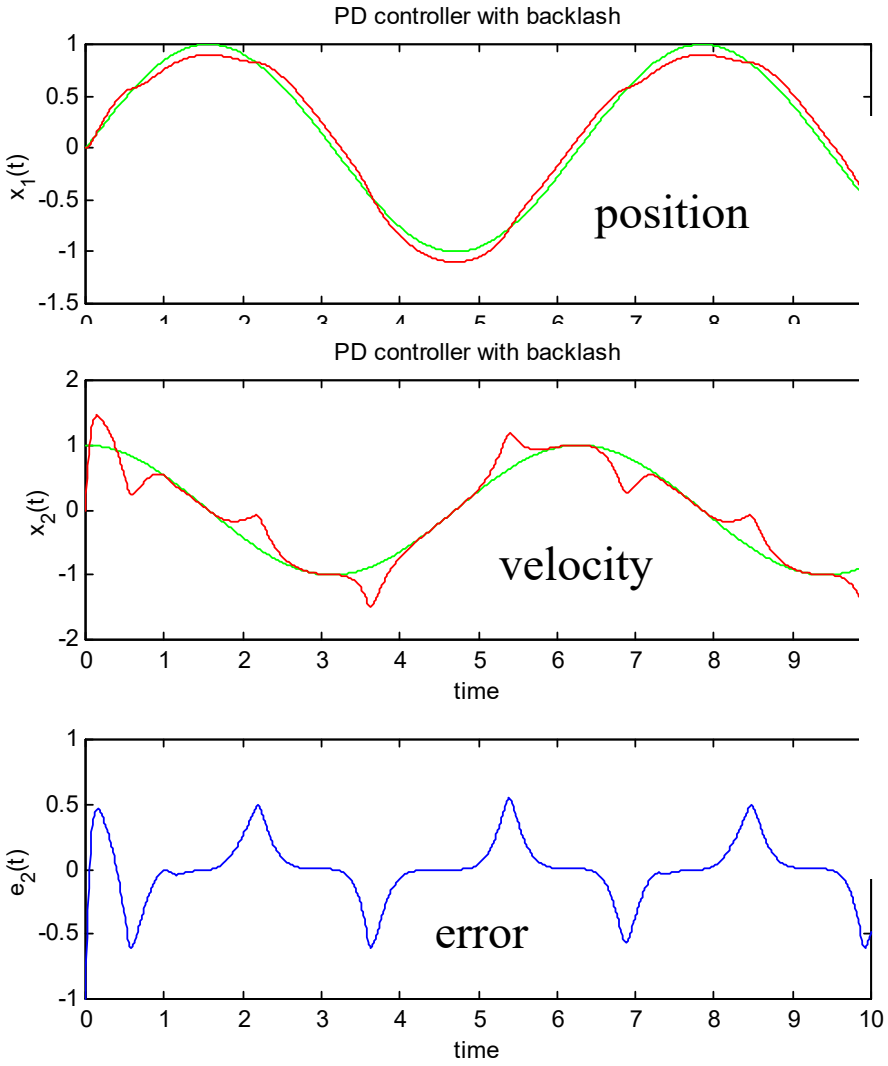
Nominal feedforward path + Unknown dynamic nonlinearity

Dynamic Inversion NN compensator for system with Backlash

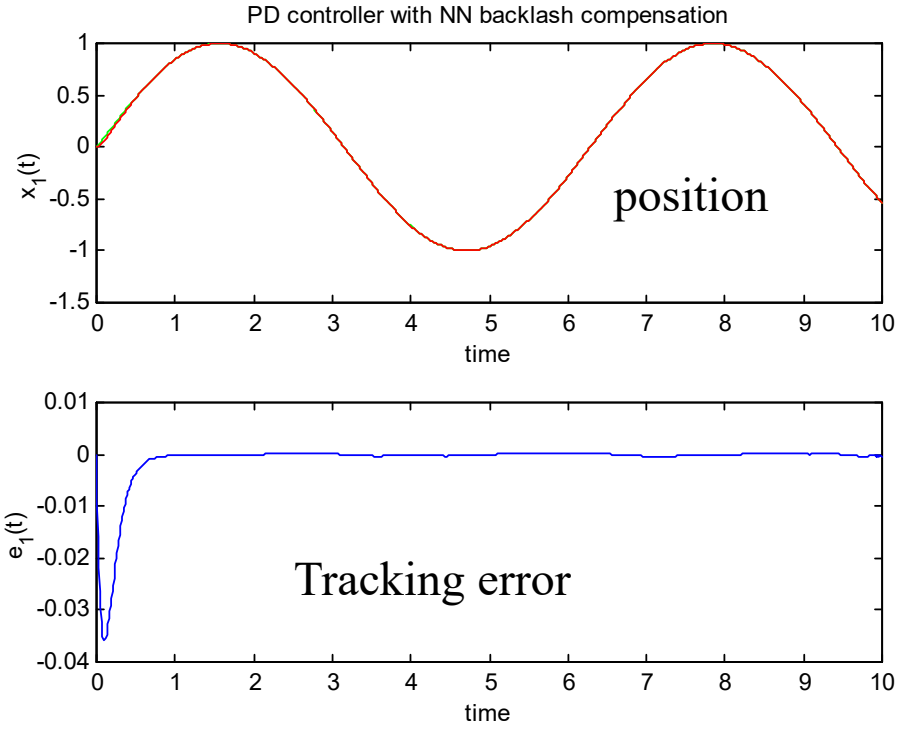


U.S. patent- Selmic, Lewis, Calise, McFarland

Performance Results



PD control-
backlash chops off tops & bottoms



NN control fixes the problem

NN Observers

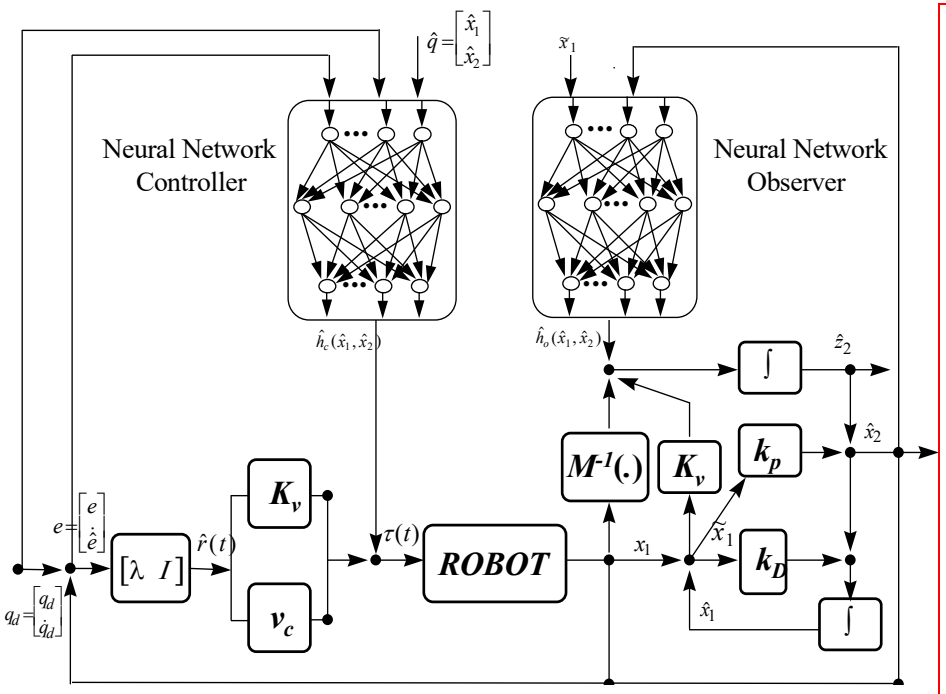
Needed when all states are not measured

i.e. Output feedback

Recurrent NN Observer

$$\dot{\hat{\mathbf{z}}}_1 = \hat{\mathbf{x}}_2 + k_D \tilde{\mathbf{x}}_1$$

$$\dot{\hat{\mathbf{z}}}_2 = \hat{\mathbf{W}}_o^T \sigma_o(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) + \mathbf{M}^{-1}(\mathbf{x}_1)\tau(t) + \mathbf{K}\tilde{\mathbf{x}}_1$$



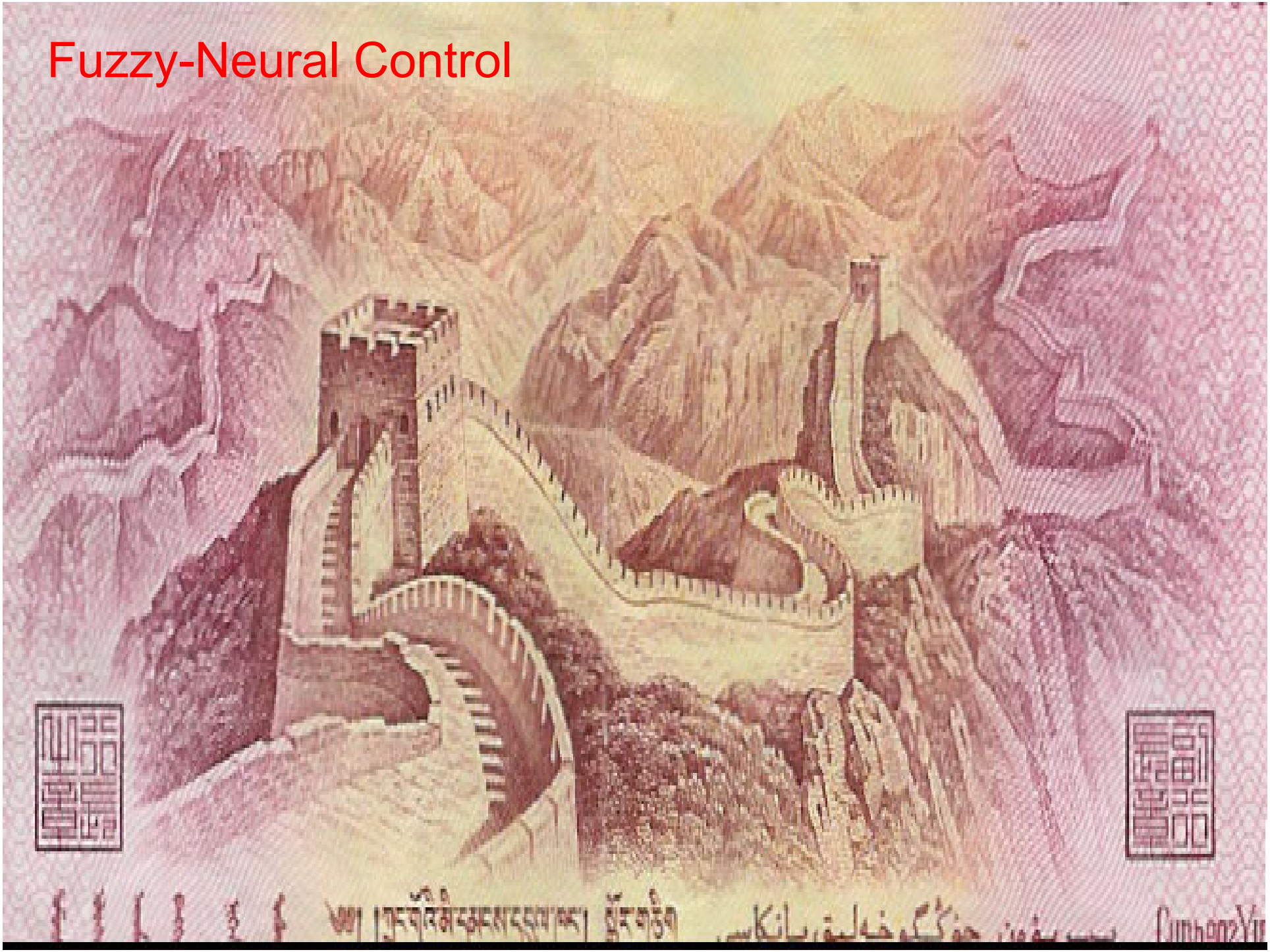
Tune NN observer -

$$\dot{\hat{\boldsymbol{W}}}_o = -k_D \boldsymbol{F}_o \sigma_o(\hat{\boldsymbol{x}}) \tilde{\boldsymbol{x}}_1^T - \kappa_o \boldsymbol{F}_o \left\| \tilde{\boldsymbol{x}}_1 \right\| \hat{\boldsymbol{W}}_o - \kappa_o \boldsymbol{F}_o \hat{\boldsymbol{W}}_o$$

Tune Action NN -

$$\dot{\hat{\boldsymbol{W}}}_c = \boldsymbol{F}_c \sigma_c(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2) \hat{\boldsymbol{r}}^T - \kappa_c \boldsymbol{F}_c \left\| \hat{\boldsymbol{r}} \right\| \hat{\boldsymbol{W}}_c$$

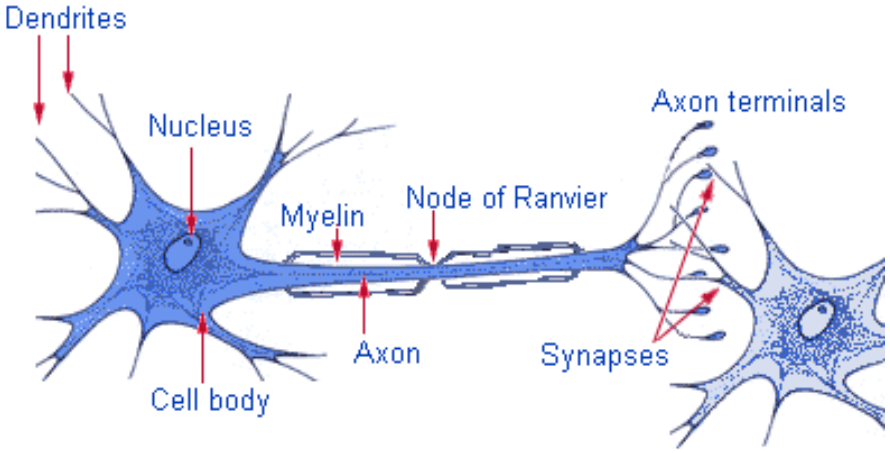
Fuzzy-Neural Control



Neural Network Properties

USED

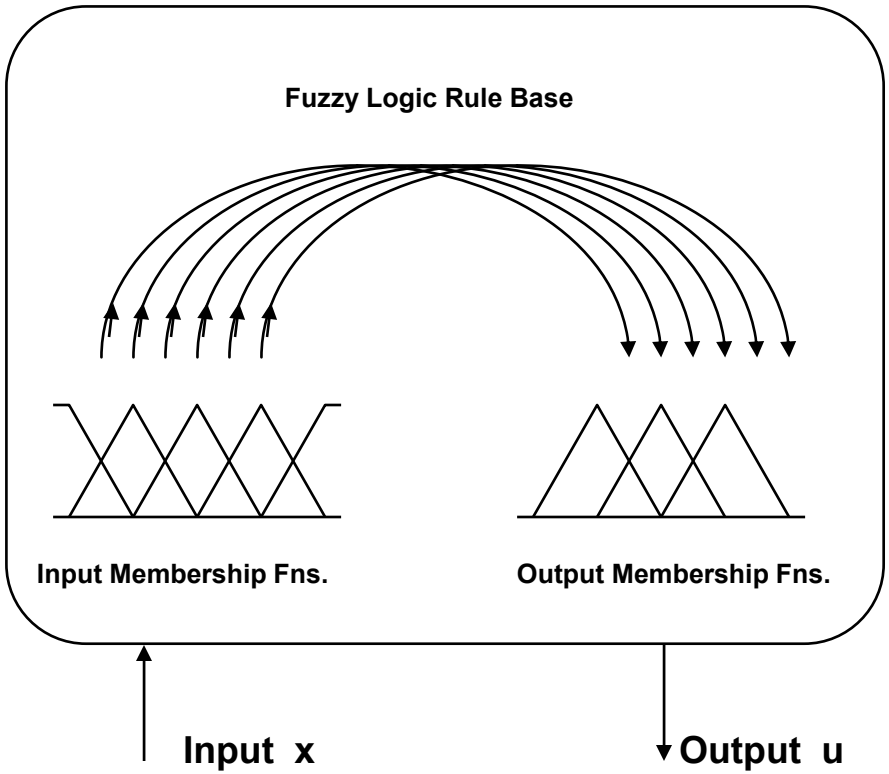
- Learning
- Recall
- Function approximation
- Generalization
- Classification
- Association
- Pattern recognition
- Clustering
- Robustness to single node failure ???
- Repair and reconfiguration



Nervous system cell.
http://www.sirinet.net/~jgjohnso/index.html

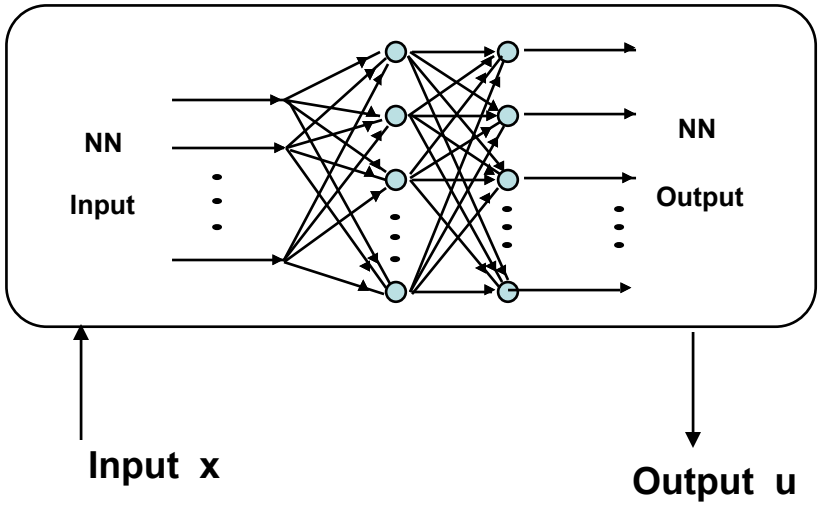
INTELLIGENT CONTROL TOOLS

Fuzzy Associative Memory (FAM)



Neural Network (NN)

(Includes Adaptive Control)

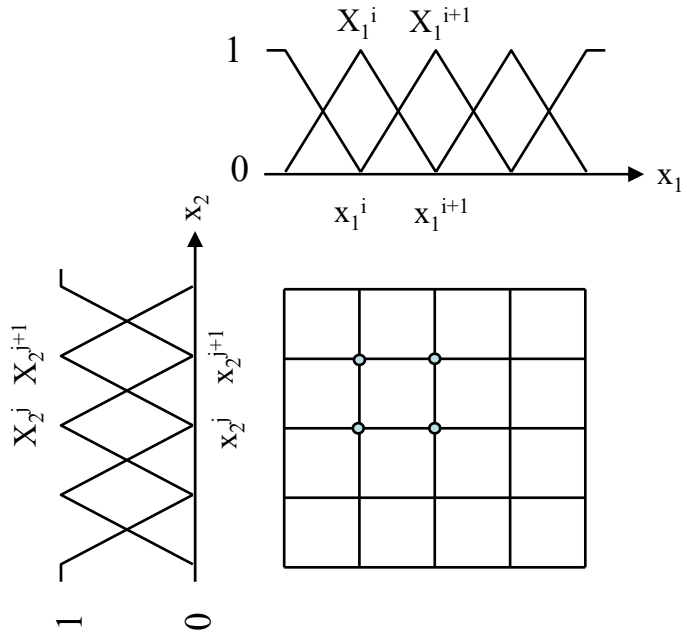


Both FAM and NN define a function u= f(x) from inputs to outputs

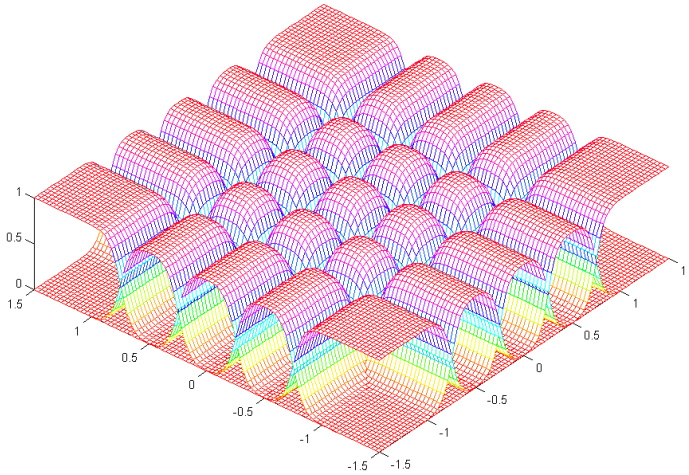
FAM and NN can both be used for: 1. Classification and Decision-Making
2. Control

NN Includes Adaptive Control (Adaptive control is a 1-layer NN)

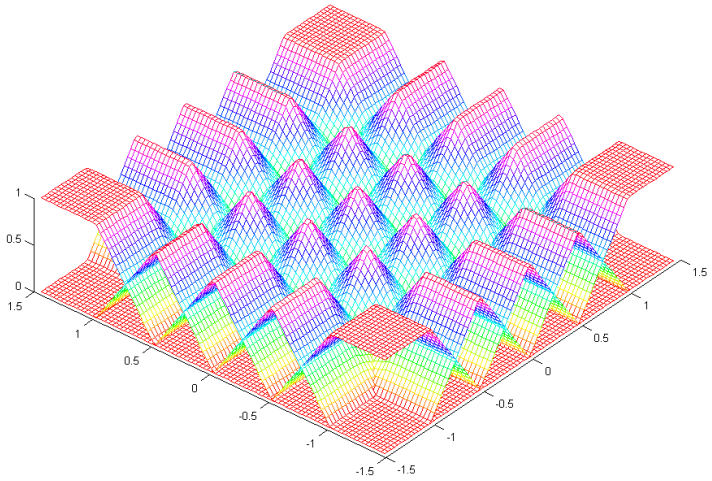
Relation Between Fuzzy Systems and Neural Networks



FL Membership Functions for 2-D Input Vector x

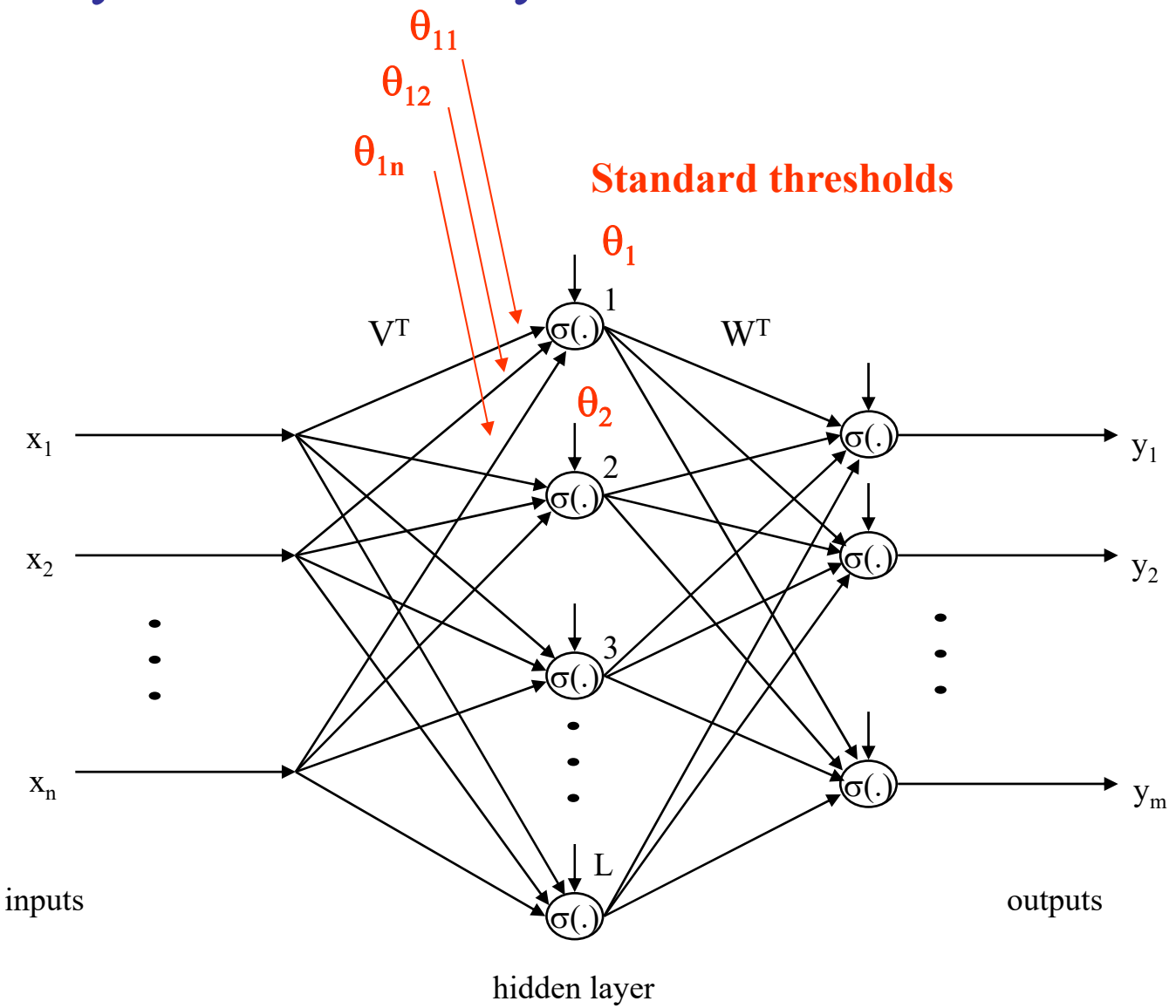


Separable Gaussian activation functions for RBF NN



Separable triangular activation functions for CMAC NN

Two-layer NN as FL System



FL system = NN with VECTOR thresholds

Fuzzy Logic Controllers

Gaussian membership function

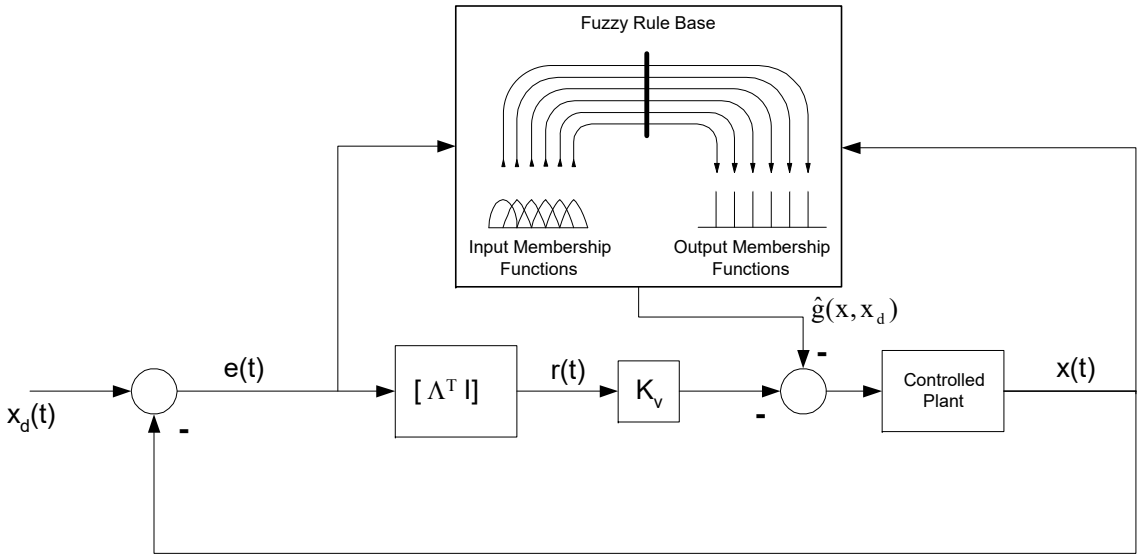
$$\phi_{A_i^l}(z_i, a_i^l, b_i^l) = e^{\left(-a_i^{l\,2}(z_i - b_i^l)^2\right)}.$$

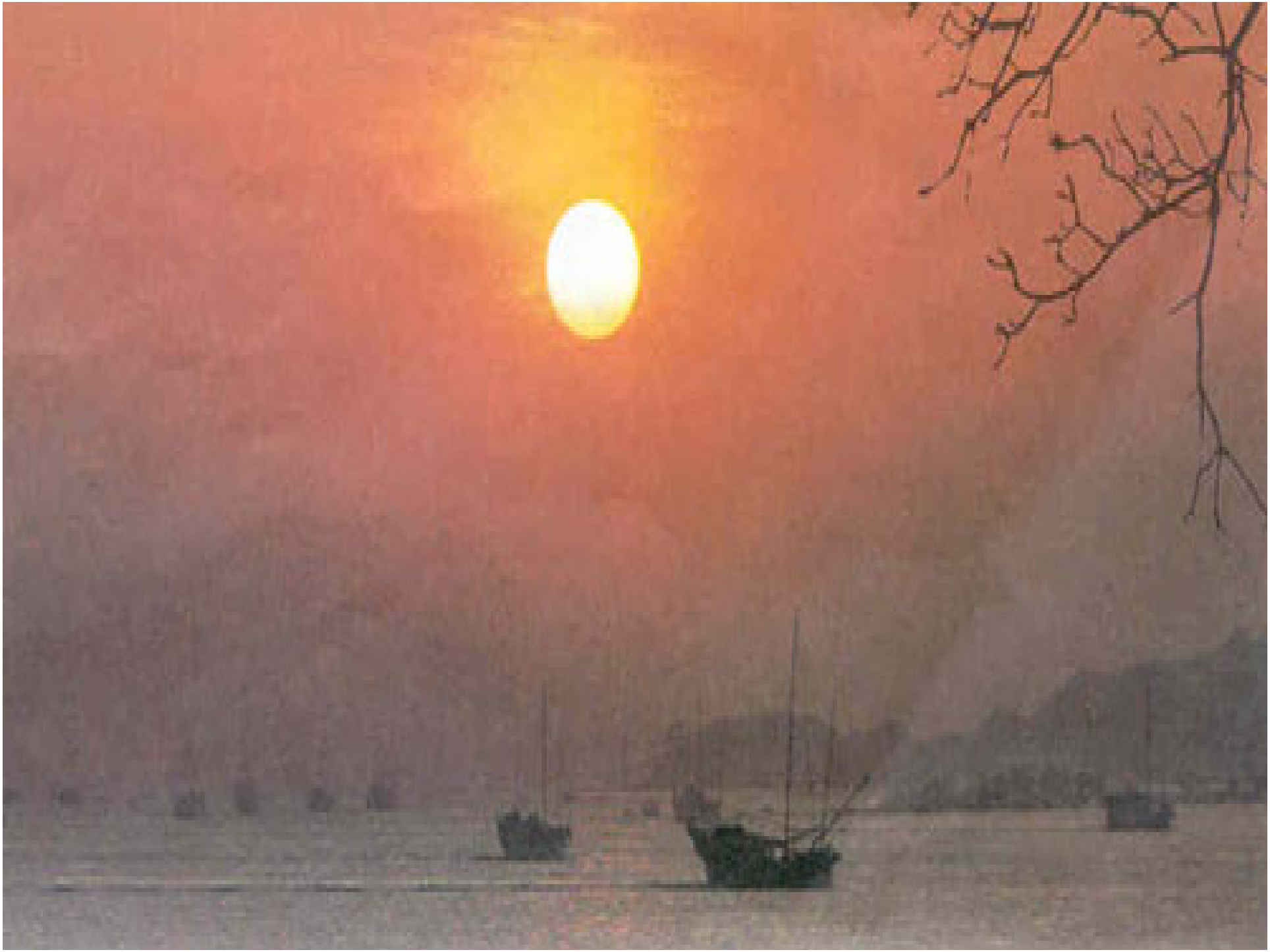
Tuning laws

$$\dot{\hat{a}} = K_a A^T \hat{W} r - k_a K_a \hat{a} \|r\|$$

$$\dot{\hat{b}} = K_b B^T \hat{W} r - k_b K_b \hat{b} \|r\|$$

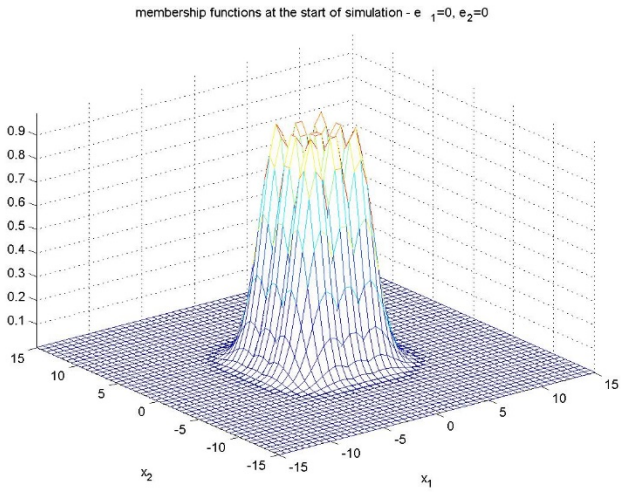
$$\dot{\hat{W}} = K_W (\hat{\Phi} - A\hat{a} - B\hat{b}) r^T - k_W K_W \hat{W} \|r\|$$

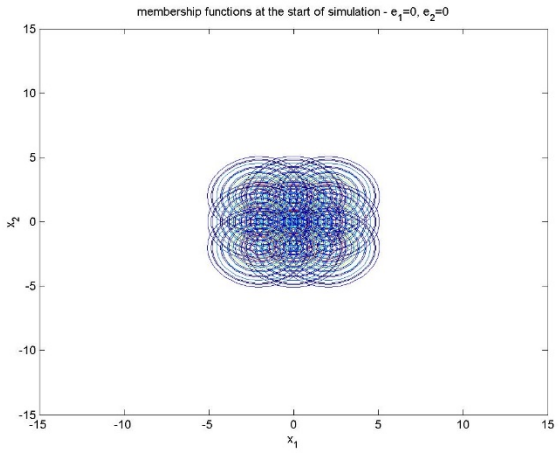




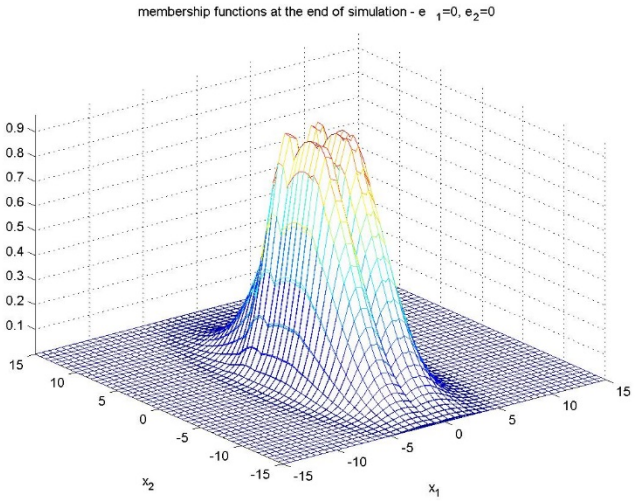
Dynamic Focusing of Awareness

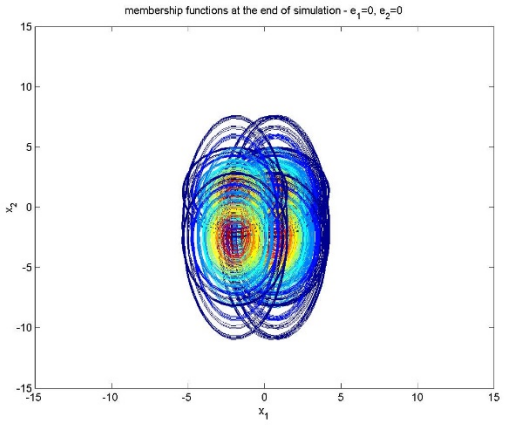
Initial MFs





Final MFs





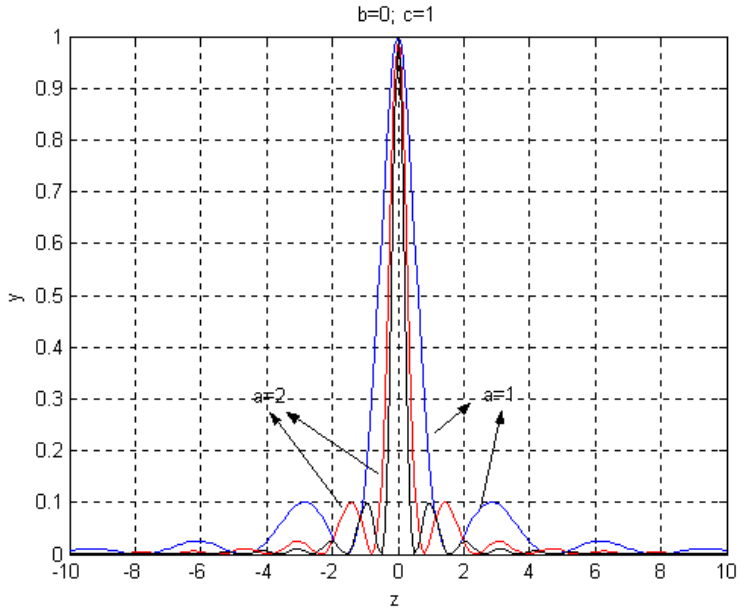
Depend on desired reference trajectory

Elastic Fuzzy Logic- c.f. P. Werbos

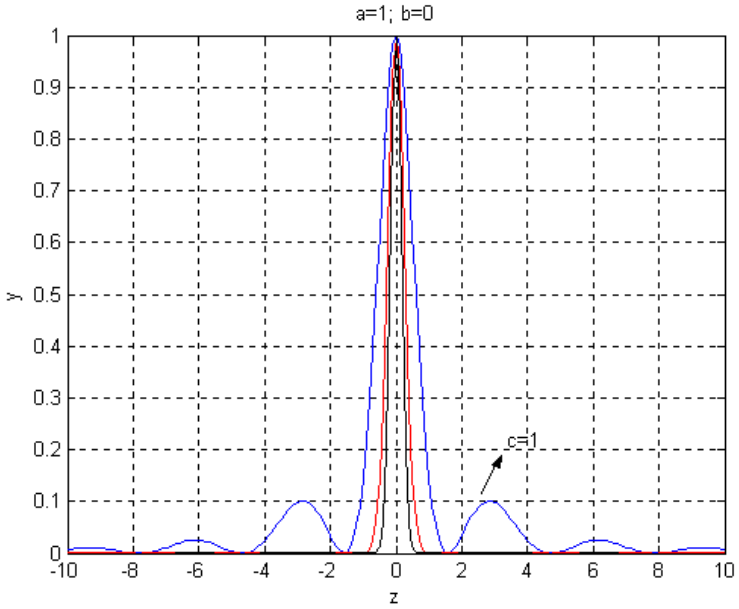
$$\phi(z,a,b,c) = \phi_B(z,a,b)^{c^2}$$

Weights importance of factors in the rules

$$\phi(z,a,b,c) = \left[\frac{\cos^2(a(z-b))}{1+a^2(z-b)^2}\right]^{c^2}$$



Effect of change of membership function spread "a"



Effect of change of membership function elasticities "c"

Elastic Fuzzy Logic Control

Control

$$u(t) = -K_v r - \hat{g}(x, x_d)$$

Tune Membership Functions

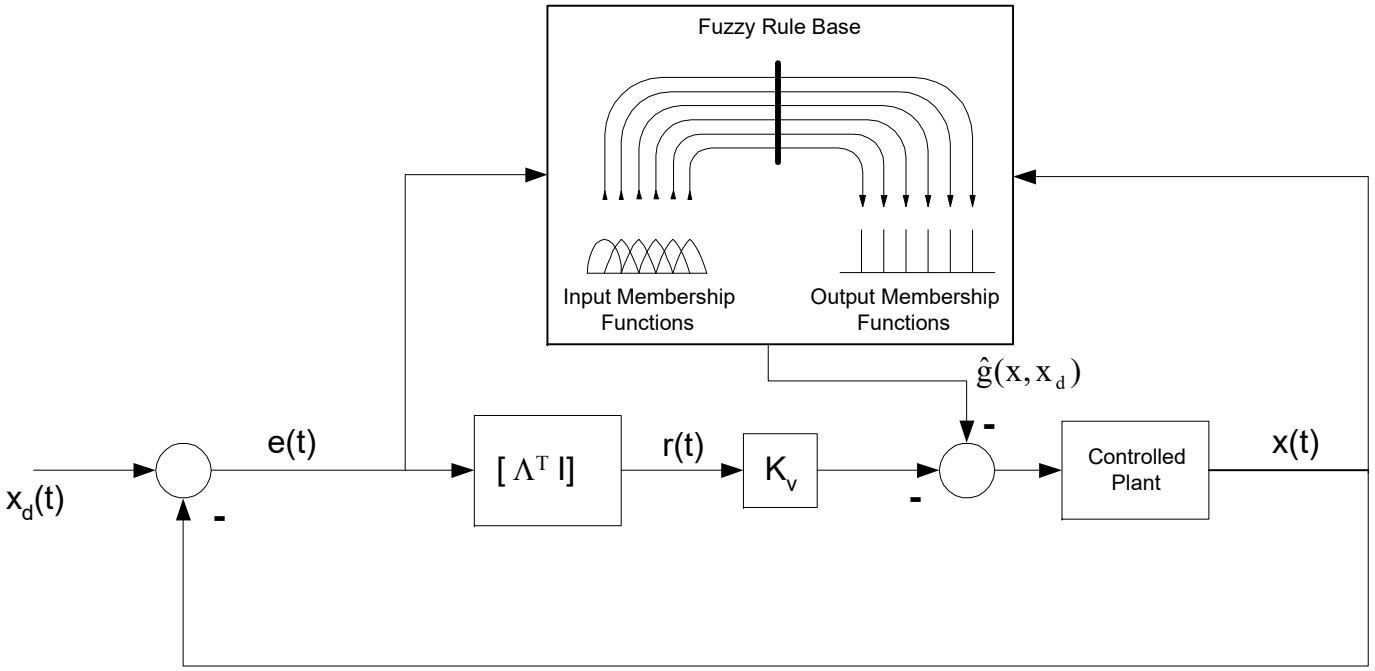
$$\dot{\hat{a}} = K_a A^T \hat{W} r - k_a K_a \hat{a} \|r\|$$

$$\dot{\hat{b}} = K_b B^T \hat{W} r - k_b K_b \hat{b} \|r\|$$

$$\dot{\hat{c}} = K_c C^T \hat{W} r - k_c K_c \hat{c} \|r\|$$

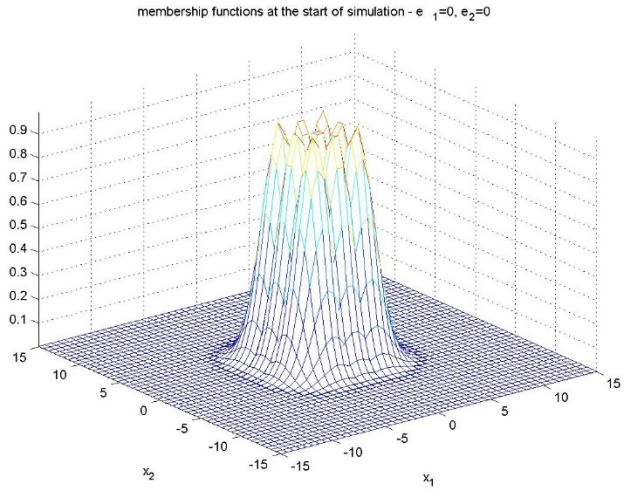
Tune Control Rep. Values

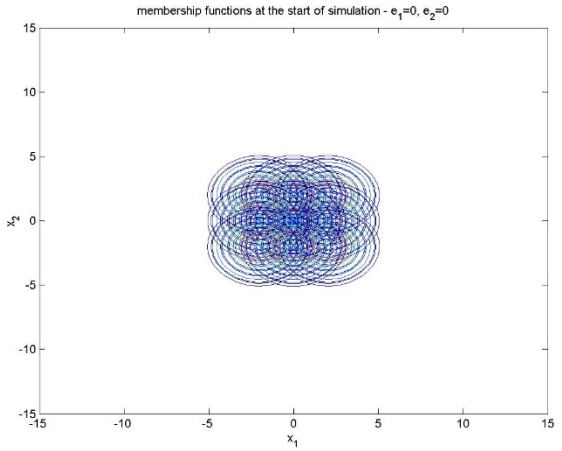
$$\dot{\hat{W}} = K_W (\hat{\Phi} - A\hat{a} - B\hat{b} - C\hat{c}) r^T - k_W K_W \hat{W} \|r\|$$



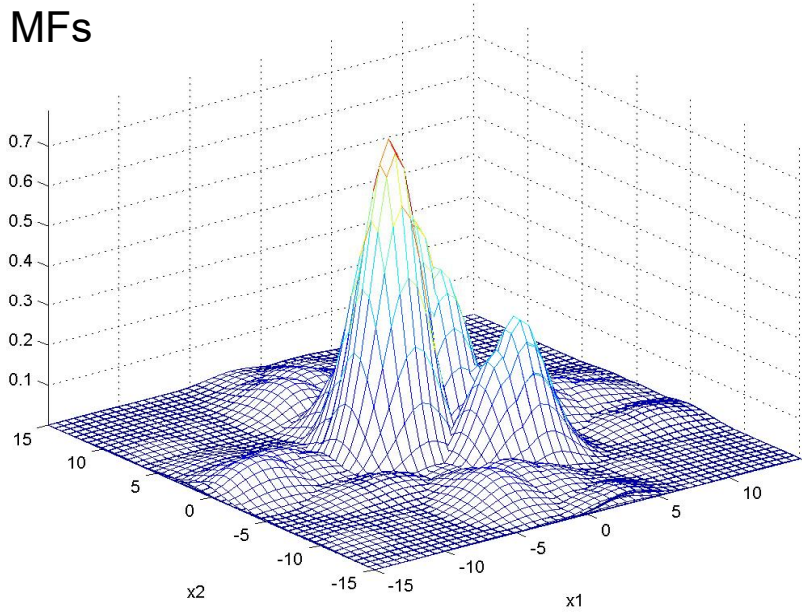
Better Performance

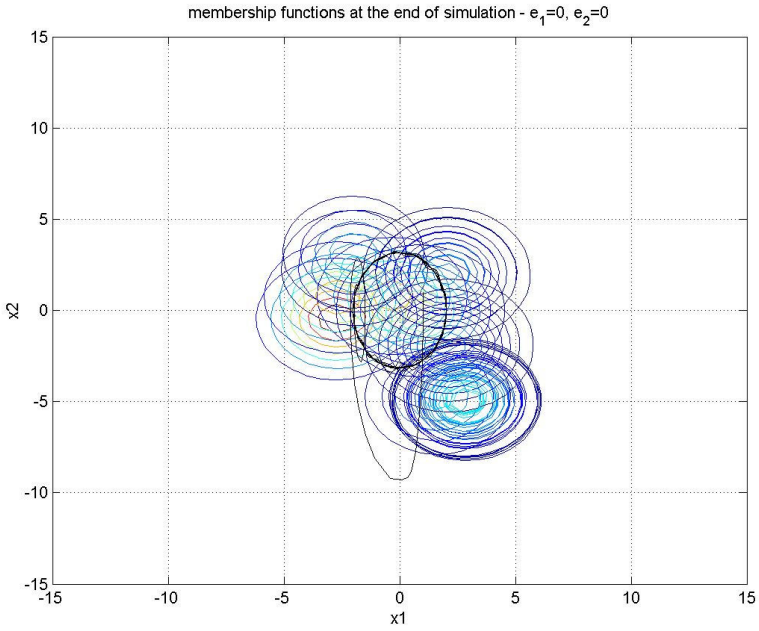
Initial MFs



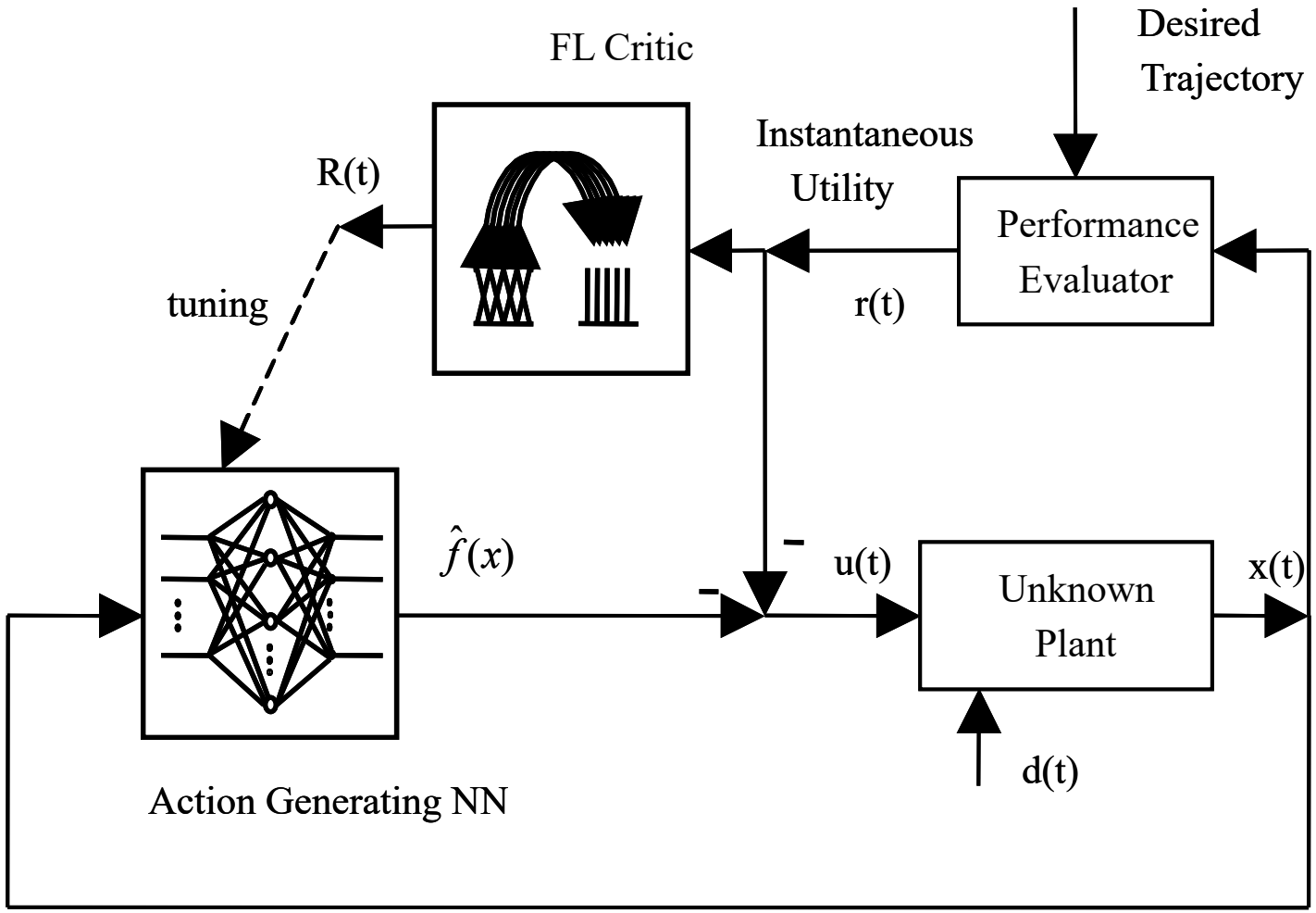


Final MFs





Fuzzy Logic Critic NN controller



Learning FL Critic Controller

Tune Action generating NN (controller)

$$\dot{\hat{W}}_2 = \Gamma_2 \sigma(\chi_2) r^T - \Gamma_2 \sigma(\chi_2) R^T \hat{W}_1^T \mu'(\hat{V}_1^T r) \hat{V}_1^T - \Gamma_2 \hat{W}_2$$

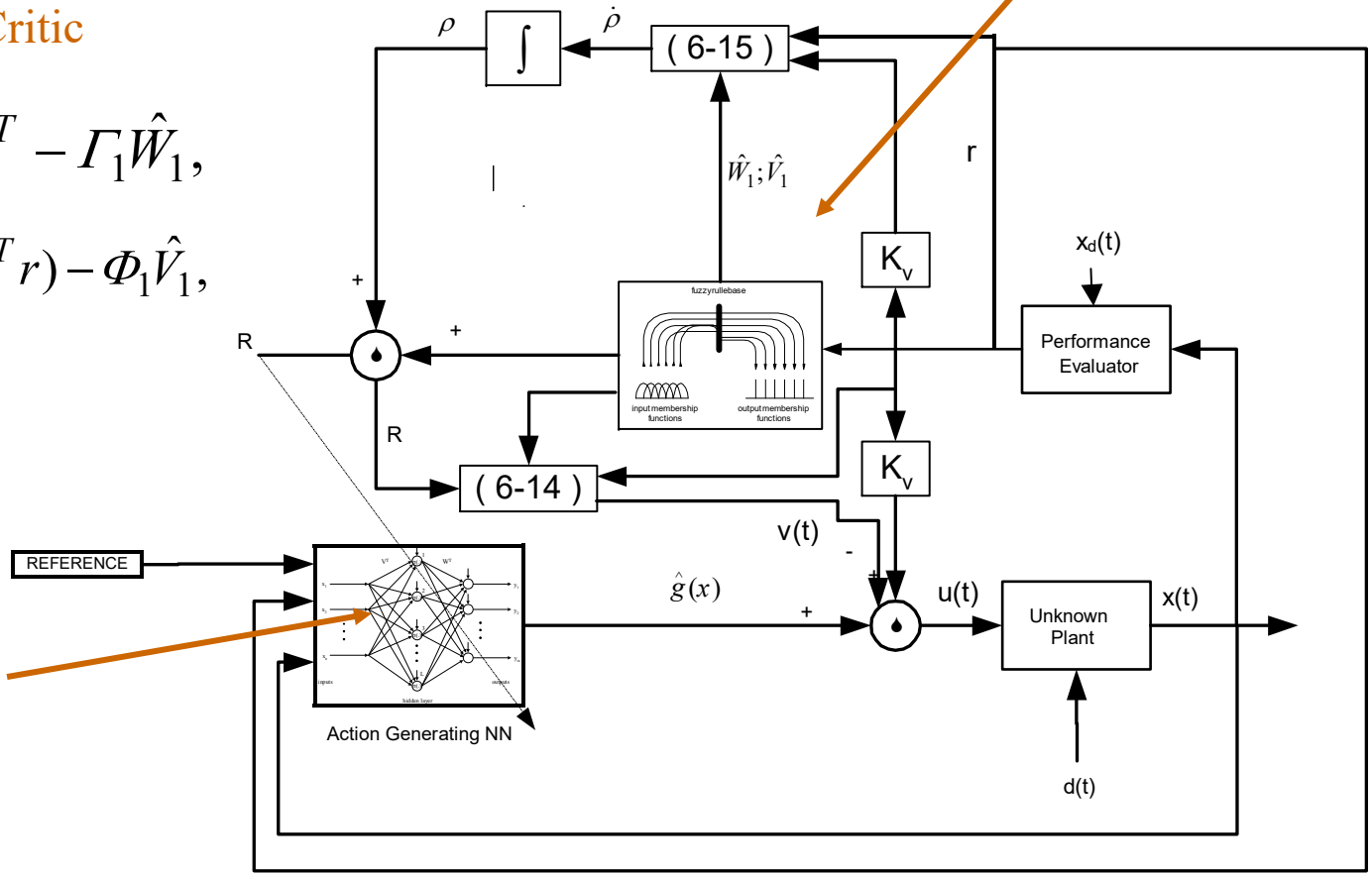
Critic requires MEMORY

FL Critic

Tune Fuzzy Logic Critic

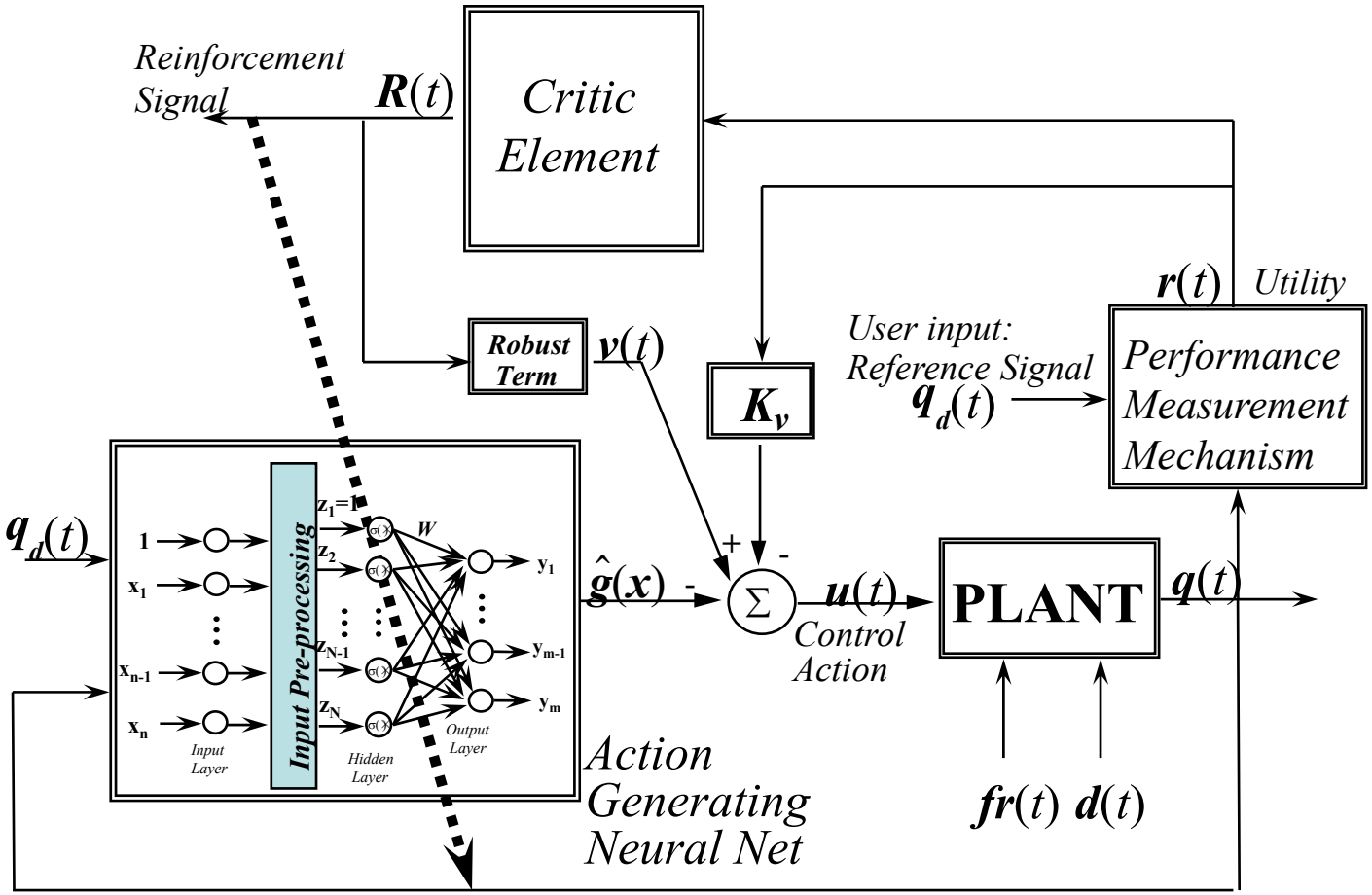
$$\dot{\hat{W}}_1 = -\mu(\hat{V}_1^T r) R^T - \Gamma_1 \hat{W}_1,$$

$$\dot{\hat{V}}_1 = -rH^T \hat{W}^T \mu'(\hat{V}_1^T r) - \Phi_1 \hat{V}_1,$$



Action generating NN

Reinforcement Learning NN Controller



High-Level NN Controllers Need Exotic Lyapunov Fns.

Reinforcement NN control

Simplified critic signal

$$R(t) = \text{sgn}(r(t)) = \pm 1$$

Lyapunov Fn

$$L(t) = \sum_{i=1}^{n} |r_i| \quad + \quad \frac{1}{2} tr(\tilde{W}^T F^{-1} \tilde{W})$$

$$\dot{L} = \boldsymbol{sgn}(\mathbf{r})^T \dot{\mathbf{r}} + tr(\tilde{\mathbf{W}}^T \mathbf{F}^{-1} \dot{\tilde{\mathbf{W}}})$$

Lyap. Deriv. contains *R(t)* !!

Tuning Law only contains *R(t)*

$$\dot{\hat{W}} = F\sigma(x)R^T - \kappa F\hat{W}$$

Adaptive Reinforcement Learning

Critic is output of NN #1

$$R = \hat{W}_1^T \cdot \sigma(\chi_1) + \rho,$$

$$L(t) = \ln(1 + e^{-\alpha r(t)}) + \ln(1 + e^{\alpha r(t)}) \quad + \quad \frac{1}{2} tr(\tilde{W}^T F^{-1} \tilde{W})$$

$$\dot{L} = \left(\frac{\alpha^+}{1 + e^{-\alpha^+ r(t)}} + \frac{-\alpha^-}{1 + e^{\alpha^- r(t)}} \right) \dot{r}(t) + tr(\tilde{\boldsymbol{W}}^T \boldsymbol{F}^{-1} \dot{\tilde{\boldsymbol{W}}})$$

Action is output of second NN

$$\hat{g}(x, x_d) = \hat{W}_2^T \sigma(\chi_2)$$

The tuning algorithm treats this as a SINGLE 2-layer NN

$$\dot{\hat{W}}_1 = -\sigma(\chi_1) R^T - \hat{W}_1,$$

$$\dot{\hat{W}}_2 = \Gamma \sigma(\chi_2) \cdot \left(r + V_1 \sigma'(\chi_1)^T \hat{W}_1 R \right)^T - \Gamma \hat{W}_2,$$