Lecture 5

9. MRAC Design for Affine-in-Control MIMO Systems

Reading material:
[1]: Chapter 8, Section 8.3
[1]: Chapter 8, Section 8.5

In this section, we consider MRAC design for a class of multi-input-multi-output (MIMO) nonlinear systems whose plant dynamics is linearly parameterized, the uncertainties satisfy the so-called matching conditions, and if the full state is measurable, (i.e., available on-line as the system output). More specifically, consider the nth order MIMO system in the form:

$$\dot{x} = A\,x + B\,\Lambda\left(u + f\left(x\right)\right) \tag{9.1}$$

where $x \in R^n$ is the system state, $u \in R^m$ is the control input, $B \in R^{n\times m}$ is known matrix, $A \in R^{n\times n}$ and $\Lambda \in R^{m\times m}$ are unknown matrices. In addition, it is assumed that Λ is diagonal, its elements λ_i are non-negative, and the pair $\left(A,\, B\,\Lambda\right)$ is controllable. The uncertainty in Λ is introduced to model a control failure phenomenon.

Moreover, the unknown possibly nonlinear function $f\left(x\right): R^n \to R^m$ represents the so-called <u>*system matched uncertainty*</u>. It is assumed that the function can be written as a linear combination of N known <u>*bounded*</u> basis functions with unknown constant coefficients.

$$f\left(x\right) = \Theta^T\,\Phi\left(x\right) \tag{9.2}$$

In (9.2), $\Theta \in R^{N\times m}$ is the unknown constant matrix, while $\Phi\left(x\right) \in R^N$ represents the known regressor vector.

The <u>*control objective*</u> of the <u>*MIMO tracking problem*</u> is to choose the input vector u such that all signals in the closed-loop system are bounded and the state x follows the state $x_{ref} \in R^n$ of a reference model specified by the LTI system

$$\dot{x}_{ref} = A_{ref}\,x_{ref} + B_{ref}\,r\left(t\right) \tag{9.3}$$

where $A_{ref} \in R^{n\times n}$ is Hurwitz, $B_{ref} \in R^{n\times m}$, and $r\left(t\right) \in R^m$ is a bounded reference input vector. Note that the reference model and its external input $r\left(t\right)$ must be chosen so that $x_{ref}\left(t\right)$ represents a <u>*desired*</u> trajectory that $x\left(t\right)$ has to follow. In other words, the control

input u needs to be chosen such that the tracking error vector asymptotically tends to zero.

$$\lim_{t\to\infty}\left\|x(t)-x_m(t)\right\|=0 \tag{9.4}$$

If the matrices A and Λ were known, one could apply the control law

$$u=K_x^T\,x+K_r^T\,r-\Theta^T\,\Phi(x) \tag{9.5}$$

and obtain the closed-loop system

$$\dot{x}=\left(A+B\,\Lambda\,K_x^T\right)x+B\,\Lambda\,K_r^T\,r \tag{9.6}$$

Comparing (9.6) with the desired dynamics in (9.3), it follows that the ideal (unknown) matrix gains must be chosen to satisfy the so-called <u>*matching conditions*</u>:

$$\left[\begin{array}{l} A+B\,\Lambda\,K_x^T=A_{ref} \\ B\,\Lambda\,K_r^T=B_{ref} \end{array}\right. \tag{9.7}$$

Assuming that the matching conditions take place, it is easy to see that the closed-loop system is the same as that of the reference model, and consequently, asymptotic (exponential) tracking is achieved for any bounded reference input signal $r(t)$.

Remark 9.1
Given the matrices $A, B, \Lambda, A_{ref}, B_{ref}$, no K_x, K_r may exist to satisfy the matching conditions (9.7) indicating that the control law (9.5) may not have enough structural flexibility to meet the control objective. Often in practice, the structure of A is known, and the reference model matrices A_{ref}, B_{ref} are chosen so that (9.7) has a solution for K_x, K_r.

Assuming that K_x, K_r in (9.7) exist, consider the following control law:

$$u=\hat{K}_x^T\,x+\hat{K}_r^T\,r-\hat{\Theta}^T\,\Phi(x) \tag{9.8}$$

where $\hat{K}_x\in R^{n\times m}, \hat{K}_r\in R^{m\times m}, \hat{\Theta}\in R^{N\times n}$ are the estimates of the ideal unknown matrices K_x, K_r, Θ, respectively. The estimated matrices will be generated on-line and by an appropriate adaptive law.

Substituting (9.8) into (9.1), the closed-loop system dynamics can be written.

$$\dot{x} = \left(A + B\Lambda \hat{K}_x^T\right)x + B\Lambda\left(\hat{K}_r^T r - \left(\hat{\Theta} - \Theta\right)^T \Phi(x)\right) \tag{9.9}$$

Subtracting (9.3) from (9.9), closed-loop dynamics of the n – dimensional tracking error vector $e(t) = x(t) - x_{ref}(t)$ can be obtained.

$$\dot{e} = \left(A + B\Lambda \hat{K}_x^T\right)x + B\Lambda\left(\hat{K}_r^T r - \left(\hat{\Theta} - \Theta\right)^T \Phi(x)\right) - A_{ref}\, x_{ref} - B_{ref}\, r \tag{9.10}$$

Using matching conditions (9.7) further yields:

$$\begin{aligned}\dot{e} &= \left(A_{ref} + B\Lambda\left(\hat{K}_x - K_x\right)\right)x - A_{ref}\, x_{ref} + B\Lambda\left(\hat{K}_r - K_r\right)r - B\Lambda\left(\hat{\Theta} - \Theta\right)^T \Phi(x) \\ &= A_{ref}\, e + B\Lambda\left[\left(\hat{K}_x - K_x\right)^T x + \left(\hat{K}_r - K_r\right)^T r - \left(\hat{\Theta} - \Theta\right)^T \Phi(x)\right]\end{aligned} \tag{9.11}$$

Let $\Delta K_x = \hat{K}_x - K_x$, $\Delta K_r = \hat{K}_r - K_r$, and $\Delta\Theta = \hat{\Theta} - \Theta$ represent the <u>*parameter estimation errors*</u>. In terms of the latter, the tracking error dynamics becomes:

$$\dot{e} = A_{ref}\, e + B\Lambda\left[\Delta K_x^T x + \Delta K_r^T r - \Delta\Theta^T \Phi(x)\right] \tag{9.12}$$

Vector and matrix norms

Before proceeding any further, recall that given a matrix $A = \left[a_{ij}\right] \in R^{n\times m}$, the <u>*Frobenius norm*</u> is defined by

$$\|A\|_F = \operatorname{tr}\left(A^T A\right) = \sum_{i,j} a_{ij}^2 \tag{9.13}$$

with $\operatorname{tr}(\ )$ the trace operator. On the other hand, given any vector *p*-norm, the <u>*induced matrix norm*</u> is defined by

$$\|A\|_p = \sup_{x\neq 0} \frac{\|Ax\|_p}{\|x\|_p} \tag{9.14}$$

Collection of Facts about vector and matrix norms, (prove it).

- For vector 1-norm $\|x\|_1 = \sum_{i=1}^{n} |x_i|$, the induced matrix norm is equal to the maximum absolute column sum, that is: $\|A\|_1 = \max_{1\le j\le m}\left|\sum_{i=1}^{n} a_{ij}\right|$.

- For vector 2-norm $\|x\|_2 = \sum_{i=1}^{n} x_i^2$, the induced matrix norm is equal to the maximum singular value of A, that is: $\|A\|_2 = \sigma_{\max}(A)$.
- For vector ∞-norm $\|x\|_\infty = \max_{1 \le i \le n} |x_i|$, the induced matrix norm is equal to the maximum absolute row sum, that is: $\|A\|_\infty = \max_{1 \le i \le n} \left| \sum_{j=1}^{m} a_{ij} \right|$.
- The induced matrix norm satisfies: $\|A\,x\|_p \le \|A\|_p \, \|x\|_p$, and for any two compatibly dimensioned matrices, A and B, one also has: $\|A\,B\|_p \le \|A\|_p \, \|B\|_p$.
- The Frobenius norm <u>*is not*</u> an induced norm of any vector norm, but it is compatible with the 2-norm in the sense that: $\|A\,x\|_2 \le \|A\|_F \, \|x\|_2$.
- For any two compatibly dimensioned matrices A and B, the <u>*Frobenius inner product*</u> is defined as: $\langle A, B \rangle_F = A^T B$.
- According to the Schwartz inequality one has: $\left\| \langle A, B \rangle_F \right\|_F = \left\| A^T B \right\|_F \le \|A\|_F \, \|B\|_F$.
- For any two co–dimensional vectors a and b, the <u>*trace identity*</u> takes place: $a^T b = \mathrm{tr}\left(b\, a^T \right)$.

Let $\Gamma_x = \Gamma_x^T > 0, \quad \Gamma_r = \Gamma_r^T > 0, \quad \Gamma_\Theta = \Gamma_\Theta^T > 0$. Going back to analyzing the tracking error dynamics in (9.12), consider the Lyapunov function candidate:

$$V\left(e, \Delta K_x, \Delta K_r, \Delta\Theta\right) = e^T P\, e + \mathrm{tr}\left(\left[\Delta K_x^T \, \Gamma_x^{-1} \, \Delta K_x + \Delta K_r^T \, \Gamma_r^{-1} \, \Delta K_r + \Delta\Theta^T \, \Gamma_\Theta^{-1} \, \Delta\Theta \right] \Lambda \right) \quad (9.15)$$

where $P = P^T > 0$ satisfies the <u>*algebraic Lyapunov equation*</u>

$$P\, A_{ref} + A_{ref}^T \, P = -Q \quad (9.16)$$

for some $Q = Q^T > 0$. Then the time derivative of V, evaluated along the trajectories of (9.12), can be calculated.

$$
\begin{aligned}
\dot{V} &= \dot{e}^T P e + e^T P \dot{e} + 2\,\mathrm{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x + \Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r + \Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\right]\Lambda\right) \\
&= \left(A_{ref}\,e + B\,\Lambda\left(\Delta K_x^T\,x + \Delta K_r^T\,r - \Delta\Theta^T\,\Phi(x)\right)\right)^T P e \\
&+ e^T P\left(A_{ref}\,e + B\,\Lambda\left(\Delta K_x^T\,x + \Delta K_r^T\,r - \Delta\Theta^T\,\Phi(x)\right)\right) \\
&+ 2\,\mathrm{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x + \Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r + \Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\right]\Lambda\right) \\
&= e^T\left(A_{ref}\,P + P\,A_{ref}\right)e + 2\,e^T P\,B\,\Lambda\left(\Delta K_x^T\,x + \Delta K_r^T\,r - \Delta\Theta^T\,\Phi(x)\right) \\
&+ 2\,\mathrm{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x + \Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r + \Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\right]\Lambda\right)
\end{aligned}
\tag{9.17}
$$

Using (9.16), yields:

$$
\begin{aligned}
\dot{V} &= -e^T\,Q\,e + \left[2\,e^T P\,B\,\Lambda\,\Delta K_x^T\,x + 2\,\mathrm{tr}\left(\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x\,\Lambda\right)\right] \\
&+ \left[2\,e^T P\,B\,\Lambda\,\Delta K_r^T\,r + 2\,\mathrm{tr}\left(\Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r\,\Lambda\right)\right] \\
&+ \left[-2\,e^T P\,B\,\Lambda\,\Delta\Theta^T\,\Phi(x) + 2\,\mathrm{tr}\left(\Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\,\Lambda\right)\right]
\end{aligned}
\tag{9.18}
$$

Using the trace identity, one gets

$$
\left[
\begin{aligned}
&\underbrace{e^T P\,B\,\Lambda}_{a^T}\,\underbrace{\Delta K_x^T\,x}_{b} = \mathrm{tr}\left(\underbrace{\Delta K_x^T\,x}_{b}\,\underbrace{e^T P\,B\,\Lambda}_{a^T}\right) \\
&\underbrace{e^T P\,B\,\Lambda}_{a^T}\,\underbrace{\Delta K_r^T\,r}_{b} = \mathrm{tr}\left(\underbrace{\Delta K_r^T\,r}_{b}\,\underbrace{e^T P\,B\,\Lambda}_{a^T}\right) \\
&\underbrace{e^T P\,B\,\Lambda}_{a^T}\,\underbrace{\Delta\Theta^T\,\Phi(x)}_{b} = \mathrm{tr}\left(\underbrace{\Delta\Theta^T\,\Phi(x)}_{b}\,\underbrace{e^T P\,B\,\Lambda}_{a^T}\right)
\end{aligned}
\right.
\tag{9.19}
$$

Substituting (9.19) into (9.18), further yields

$$
\begin{aligned}
\dot{V} &= -e^T\,Q\,e + 2\,\mathrm{tr}\left(\Delta K_x^T\left[\Gamma_x^{-1}\,\dot{\hat{K}}_x + x\,e^T P\,B\right]\Lambda\right) \\
&+ 2\,\mathrm{tr}\left(\Delta K_r^T\left[\Gamma_r^{-1}\,\dot{\hat{K}}_r + r\,e^T P\,B\right]\Lambda\right) + 2\,\mathrm{tr}\left(\Delta\Theta^T\left[\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}} - \Phi(x)\,e^T P\,B\right]\Lambda\right)
\end{aligned}
\tag{9.20}
$$

<u>*Adaptive laws*</u> are chosen as follows:

$$\left[\begin{array}{l}\dot{\hat{K}}_x = -\Gamma_x\, x\, e^T P\, B \\ \dot{\hat{K}}_r = -\Gamma_r\, r(t) e^T P\, B \\ \dot{\hat{\Theta}} = \Gamma_\Theta\, \Phi(x) e^T P\, B\end{array}\right. \tag{9.21}$$

Then the time-derivative of V becomes negative semi-definite.

$$\dot{V} = -e^T Q e \le 0 \tag{9.22}$$

Therefore the closed-loop error dynamics is stable, that is the tracking error $e(t)$ and the parameter estimation errors $\Delta K_x(t)$, $\Delta K_r(t)$, $\Delta\Theta(t)$ are bounded signals in time. Therefore, the parameter estimates $\hat{K}_x(t)$, $\hat{K}_r(t)$, $\hat{\Theta}(t)$ are also bounded. Since $r(t)$ is bounded then $x_{ref}(t)$ and $\dot{x}_{ref}(t)$ are bounded. Hence, the system state $x(t)$ is bounded and, consequently the control input $u(t)$ in (9.8) is bounded. The latter implies that $\dot{x}(t)$ is bounded and, hence $\dot{e}(t)$ is bounded. Furthermore, the 2nd time derivative of $V(t)$

$$\ddot{V} = -e^T Q e = -2 e^T Q \dot{e} \tag{9.23}$$

is bounded and thus $\dot{V}(t)$ is a uniformly continuous function of time. The latter coupled with the facts that $V(t)$ is lower bounded and $\dot{V}(t) \le 0$ implies (Barbalat's Lemma) that $\lim_{t\to\infty} \dot{V}(t) = 0$. Thus $\lim_{t\to\infty} \|e(t)\| = 0$ and the MIMO tracking problem is solved.

Remark 9.2 (prove it)
If some of the diagonal elements λ_i of the unknown diagonal matrix Λ are negative and the signs of all of them are known, then the adaptive laws

$$\left[\begin{array}{l}\dot{\hat{K}}_x = -\Gamma_x\, x\, e^T P\, B \operatorname{sgn}\Lambda \\ \dot{\hat{K}}_r = -\Gamma_r\, r(t) e^T P\, B \operatorname{sgn}\Lambda \\ \dot{\hat{\Theta}} = \Gamma_\Theta\, \Phi(x) e^T P\, B \operatorname{sgn}\Lambda\end{array}\right. \tag{9.24}$$

solve the MIMO tracking problem, where $\operatorname{sgn}\Lambda = \operatorname{diag}\left[\operatorname{sgn}\lambda_1, \ldots, \operatorname{sgn}\lambda_m\right]$.

10. Artificial Neural Networks for Function Approximation

Motivation.

A typical control design process starts with *modeling* which is basically the process of constructing a mathematical description (such as a set of ODE-s) for the physical system to be controlled. Note that *more accurate models are not always better*. They may require unnecessarily complex control design and analysis and more demanding computation. The key here is to model essential effects in the system dynamics in the operating range of interest. In addition, a good model should also provide some characterization of the *model uncertainties* – the so-called "unknown unknowns", which can be used for robust design, adaptive design, or merely simulation and system testing, (such as Monte Carlo runs). Model uncertainties are the differences between the model and the real physical process. Uncertainties in parameters are called *parametric*, while the others are called *non-parametric* uncertainties.

Example 10.1
For the model of a controlled mass $m\ddot{x} = u$ the uncertainty in m is parametric, while the neglected motor dynamics, measurement noise, sensor dynamics represent the non-parametric uncertainties.

Example 10.2
Consider the scalar model with uncertain dynamics $\dot{x} = f(x) + u$, where $f(x)$ is not known. Suppose that

$$f(x) = \sum_{i=1}^{N} \theta_i \varphi_i(x) + \varepsilon(x) = \underbrace{\theta^T \Phi(x)}_{\text{Parametric Uncertainty}} + \underbrace{\varepsilon(x)}_{\text{Non-Parametric}}$$

that is suppose that the unknown function $f(x)$ can be approximated by a linear combination of *known basis functions* $\varphi_i(x)$ and *unknown constant parameters* θ_i. The approximation error $\varepsilon(x)$ is the non-parametric uncertainty, while the unknown constant parameters θ represent the parametric uncertainty in the system dynamics. In order to characterize the latter, one needs to be able to find a good set of basis functions $\Phi(x)$ such that the approximation error $\varepsilon(x)$ becomes small on a compact x – domain. Polynomials, Fourier series expansions, splines and *feedforward neural networks* can be used to approximate functions on compact domains.

In what follows, we show how to *adapt to parametric uncertainties*, while maintain *robustness in the presence of non-parametric uncertainties*.

Definition 10.1
Artificial Feedforward Neural Networks are multi-input-multi-output systems composed of many inter-connected nonlinear processing elements (neurons) operating in parallel.

Figures 10.1 and 10.2 show sketches of two feedforward NN-s.

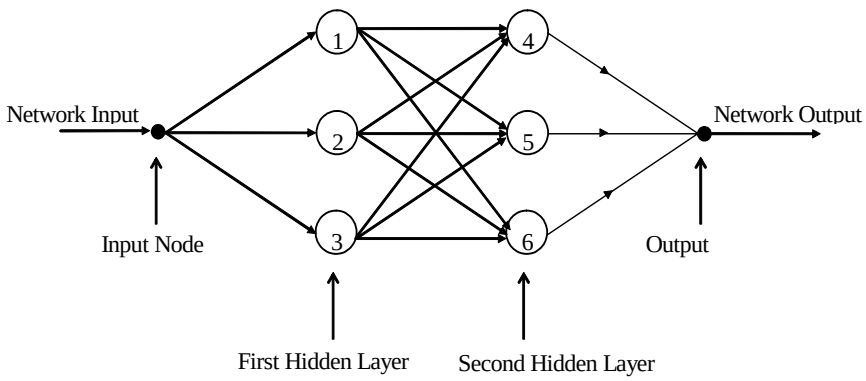

Figure 10.1: Feedforward Neural Network with 2 hidden layers and 6 neurons

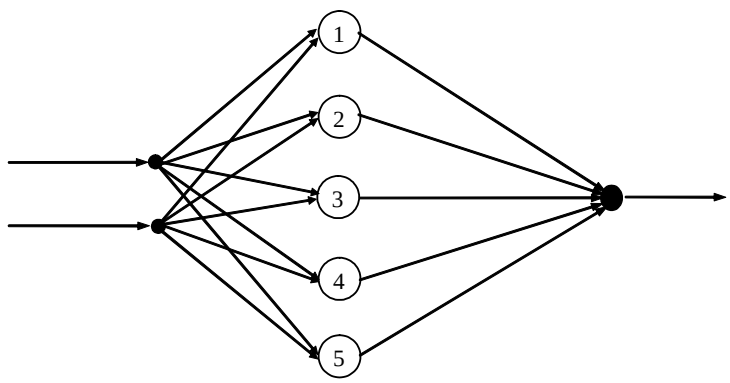

Figure 10.2: Feedforward Neural Network with 1 hidden layer and 5 neurons

As seen from the examples, an artificial feedforward neural network consists of neurons and their connections. A block-diagram of a neuron is shown below.

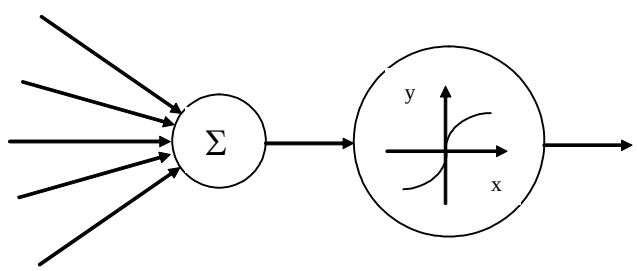

Figure 10.3: Artificial Neuron Block-Diagram

Neurons, the basic processing elements of NN-s, have two main components:

- a weighted summer
- a *nonlinear* activation function

The activation functions of interest are <u>*Radial Basis Functions*</u> or ridge functions, (often called the <u>*sigmoids*</u>).

Definition 10.2
A Radial Basis Function (RBF) is defined as a Gaussian:

$$\varphi\left(x, x_c\right) = e^{-\left(x-x_c\right)^T W \left(x-x_c\right)} = e^{-\left\|x-x_c\right\|_W^2} \tag{10.1}$$

In (10.1), $x \in R^n$ is the input, $x_c \in R^n$ is the center, and $W = W^T > 0$ is a positive-definite symmetric matrix of weights. Most often we will write $\varphi\left(x, x_c\right) = \varphi_i\left(x\right)$ to abbreviate and denote an RBF which is centered at the ith center.

Remark 10.1
Other definitions of an RBF are available. Often in the literature, an RBF is defined as $\phi = \phi\left(\left\|x - x_c\right\|_W\right)$, where $\left\|\bullet\right\|_W$ denotes the usual Euclidean weighted norm. In addition, it is required that $\phi\left(\bullet\right)$ is integrable on R^n and $\int_{R^n} \phi\left(x\right) dx \neq 0$. Basically, this type of RBF depends only on the weighted distance $r = \left\|x - x_c\right\|_W$ between its current input x and the center x_c. The Gaussian RBF in (10.1) is an example of this type of activation function. Others include:

- multiquadrics

$$\varphi\left(r\right) = \sqrt{\left(r^2 + c^2\right)}, \quad c > 0$$

- inverse multiquadrics

$$\varphi\left(r\right) = \frac{1}{\sqrt{\left(r^2 + c^2\right)}}, \quad c > 0$$

Micchelli's Theorem
Let $\varphi = \varphi\left(r\right)$ be the Gaussian, the multiquadrics, or the inverse multiquadrics function. Let $\left\{x_i\right\}_{i=1}^N$ be a set of distinct points in R^n. Then the $\left(N \times N\right)-$ interpolation matrix Φ, whose $\left(i, j\right)^{th}$ element is $\varphi_{ij} = \varphi\left(\left\|x_i - x_j\right\|\right)$, is nonsingular.

Remark 10.2
There is a large class of RBF-s that is covered by Micchelli's theorem. The theorem provides theoretical basis for RBF based function approximation problems. In other words, using an RBF $\varphi = \varphi\left(r\right)$ and a finite set of N points $\left\{x_i\right\}_{i=1}^N$ in R^n, it is always

possible to approximate a large class of functions $f(x)$ with $\hat{f}(x)=\sum_{i}^{N}\theta_i\,\varphi(x-x_i)$ such that $f(x_i)=\hat{f}(x_i)$, for all $\{x_i\}_{i=1}^{N}$.

Definition 10.3
A ridge function / sigmoid is a <u>*nonlinear*</u> function in the form:

$$\sigma=\sigma\left(w^T\,x+b\right) \tag{10.2}$$

where $w\in R^n$ denotes the vector of weights, b is a scalar called the threshold, and $\sigma(\bullet)$ is a nonlinear function (not necessarily continuous) defined in R with the following properties:

$$\begin{cases}\lim\limits_{s\to-\infty}\sigma(s)<\infty\\ \lim\limits_{s\to\infty}\sigma(s)<\infty\end{cases} \tag{10.3}$$

The most common examples of a ridge function are: a) the logistic sigmoid

$$\sigma(s)=\frac{1}{1+e^{-s}} \tag{10.4}$$

and b) the hyperbolic tangent

$$\sigma(s)=\frac{1-e^{-s}}{1+e^{-s}} \tag{10.5}$$

A feedforward NN with N neurons in its hidden layer is shown in Figure 10.4.

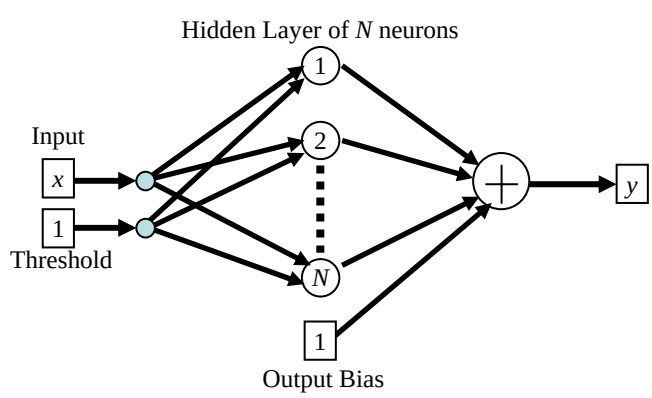

Figure 10.4: Single-hidden-layer feedforward NN with N neurons

Formally, a feedforward NN maps R^n into R^m, that is:

$$y = NN(x), \quad x \in R^n, \quad y \in R^m \tag{10.6}$$

Definition 10.4
A <u>*sigmoidal feedforward NN*</u> is:

$$NN(x) = W^T \vec{\sigma}\left(V^T x + \theta\right) + b \tag{10.7}$$

where $W \in R^{N \times m}$ is the matrix of the <u>*outer-layer weights*</u>,

$$\vec{\sigma}(\bullet) = \left(\sigma\left(V_1^T x + \theta_1\right) \quad \ldots \quad \sigma\left(V_N^T x + \theta_N\right)\right)^T \in R^N$$

is the vector of N sigmoids, $V \in R^{n \times N}$ is the matrix of the <u>*inner-layer synaptic weights*</u> with its ith column denoted by $V_i \in R^n$, $\theta \in R^N$ is the <u>*vector of thresholds*</u>, and $b \in R^m$ denotes the NN <u>*bias*</u> vector.

Definition 10.5
A feedforward <u>*RBF NN*</u> is:

$$NN(x) = \theta^T \begin{pmatrix} \varphi\left(\|x - C_1\|_{W_1}\right) \\ \vdots \\ \varphi\left(\|x - C_N\|_{W_N}\right) \end{pmatrix} + b = \underbrace{\begin{pmatrix} \theta^T & b \end{pmatrix}}_{\Theta^T} \underbrace{\begin{pmatrix} \varphi_1(x) \\ \vdots \\ \varphi_N(x) \\ 1 \end{pmatrix}}_{\Phi(x)} = \Theta^T \Phi(x) \tag{10.8}$$

where $\Theta = \begin{pmatrix} \theta^T & b \end{pmatrix}^T \in R^{(N+1) \times m}$ is the <u>*vector of weights*</u>, $C_i \in R^n$ is the <u>*center*</u> of the ith receptive field, $W_i = W_i^T > 0$ is the norm weighting matrix, $b \in R^m$ is the NN <u>*bias*</u>, and $\Phi(x) = \begin{pmatrix} \varphi_1(x) & \ldots & \varphi_N(x) & 1 \end{pmatrix}^T \in R^{N+1}$ is the so-called <u>*regressor vector*</u>, whose components are the basis activation functions $\varphi_i(x) = \varphi\left(\|x - C_i\|_{W_i}\right)$ and the unity function.

Remark 10.3
Often in practical applications, the symmetric positive-definite matrix W in (10.8) is chosen to be diagonal and in the form:

$$W_i = \frac{1}{2\sigma_i^2}, \quad (i = 1, \ldots, N)$$

where σ_i represents the <u>*width*</u> of the ith Gaussian function, that is:

$$\varphi_i(x) = e^{-\frac{\|x - C_i\|^2}{2\sigma_i^2}}$$

becomes the ith component of the regressor vector $\Phi(x)$ in (10.8). Most often, the components of the regressor will be constructed using an <u>*isotropic Gaussian*</u> function

$$\varphi_i(x) = e^{-\left(\frac{N}{d_{\max}^2}\right)\|x - C_i\|^2}$$

whose standard deviation (i.e., width) σ is fixed according to the spread of the centers C_i, N is the number of centers, and $d_{\max}$ is the maximum distance between the chosen centers. In this case, the standard deviation σ of all the isotropic Gaussian RBF components is fixed at

$$\sigma = \frac{d_{\max}}{\sqrt{2N}}$$

This formula ensures that the individual RBF-s are not too peaked or too flat. Both of these two extreme conditions should be avoided.

Feedforward NN-s have been shown to be capable of approximating generic classes of functions, including continuous and integrable ones, on a compact domain and to within any tolerance. This property of feedforward NN-s is often referred to as the <u>*Universal Approximation*</u> property, while the NN-s themselves are often called the <u>*universal approximators*</u>. Two related theorems are given below.

Universal Approximation Theorem for Sigmoidal NN-s, (G. Cybenko, 1989)
Any continuous function $f(x): R^n \to R$ can be uniformly approximated by a single-hidden-layer NN with a bounded monotone-increasing continuous activation function and on a compact domain $X \subset R^n$, that is:

$$\forall \varepsilon > 0 \quad \exists N, W, b, V, \theta \quad \forall x \in X \subset R^n \quad \left\| \underbrace{W^T \vec{\sigma}\left(V^T x + \theta\right) + b}_{NN(x)} - f(x) \right\|_\infty \le \varepsilon \quad (10.9)$$

Remark 10.4

The universal approximation theorem extends to the class of L_1 functions on a compact domain. In that case, it is assumed that the activation function is a bounded measurable sigmoid and the approximation is understood in terms of the L_1 norm.

Rate of Approximation Theorem for Sigmoidal NN-s, (12. Barron, 1993)
Consider a class of functions $f(x)$ on R^n for which there is a Fourier representation of the form

$$f(x) = \int_{R^n} e^{i\omega x}\, \tilde{f}(\omega)\, d\omega$$

for some complex-valued function $\tilde{f}(\omega)$ for which $\omega\, \tilde{f}(\omega)$ is integrable, and define

$$C_f = \int_{R^n} \|\omega\| \left\| \tilde{f}(\omega) \right\| d\omega < \infty$$

Then for every function $f(x)$ with C_f finite, and every $N \geq 1$, there exists a sigmoidal NN of the form (10.7), such that

$$\left\| f(x) - NN(x) \right\|_{L_2}^2 \triangleq \int_{\|x\| \leq r} \left(f(x) - NN(x) \right)^2 dx \leq \frac{\left(2\, r\, C_f\right)^2}{N}$$

Remark 10.5
Functions with C_f finite are continuously differentiable on R^d. Moreover, the NN approximation error is measured by the integrated squared error, (L_2- norm), on the ball of radius r.

Universal Approximation Theorem for RBF NN-s, (Park and Sandberg, 1991)
Let $\varphi(x): R^n \to R$ be an integrable bounded continuous function and assume that

$$\int_{R^n} \varphi(x)\, dx \neq 0$$

Then for any continuous function $f(x)$ and any $\varepsilon > 0$ there is an RBF NN with N neurons, a set of centers $\{C_i\}_{i=1}^N$, and a common width $\sigma > 0$

$$\hat{f}(x) = \sum_{i=1}^{N} \theta_i \underbrace{\varphi\left(\frac{x - C_i}{\sigma}\right)}_{\varphi_i(x)} = \Theta^T\, \Phi(x)$$

such that

$$\left\| f(x) - NN(x) \right\|_{L_2}^2 \triangleq \int_{\|x\| \le r} \left(f(x) - NN(x) \right)^2 dx \le \varepsilon = \mathrm{O}\left(N^{-\frac{1}{n}} \right)$$

Comparison of sigmoidal and RBF NN-s

- RBF and sigmoidal NN-s are universal approximators
- RBF NN depends the Euclidean distances between the input vector x and the centers C_i. Meanwhile, sigmoidal NN-s depend on the inner product of the input vector x with the synaptic weight vectors V_i and biased by θ.
- Sigmoidal NN-s provide $\mathrm{O}\left(N^{-\frac{1}{2}} \right)$ rate of approximation which does not explicitly depend on the dimension of x. On the other hand, the rate of approximation for the RBF NN-s is of order $\mathrm{O}\left(N^{-\frac{1}{2n}} \right)$ and, consequently it decreases exponentially as the dimension of the input vector x increases. This phenomenon is called the Curse of Dimensionality, (due to R. Bellman).
- An RBF has local support while a sigmoid does not. The local support implies learning and adaptation ability of RBF NN-s. Sigmoidal NN-s adapt but don't learn.

With specific reference to NN-s in control, it is their ability to represent nonlinear mappings, and hence to model nonlinear systems, which is the feature to be most readily exploited in the synthesis of nonlinear controllers.

Reading material / References:

1) F. Scarselli, A.C. Tsoi, "Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results", Neural Networks, vol. 11, No. 1, pp. 15-37, 1998.

2) K.J. Hunt, D. Sbarbaro, R. Zbikowski, P.J. Gawthrop, "Neural Networks for Control Systems – A Survey", Automatica, vol. 28, No. 6., pp. 1083-1112, 1992.

3) G. Cybenko, "Approximation by superposition of a sigmoidal function", Math. Control Signals Systems, vol. 2, pp. 303-314, 1989.

4) J. Park, I.W. Sandberg, "Universal approximation using radial-basis-function networks", Neural Computation, vol. 3, no. 2, pp. 246-257.

5) C. A. Micchelli, "Interpolation of scattered data: Distance matrices and conditionally positive definite functions", Constructive Approximation, vol.2, pp. 11-12, 1986.

11. MRAC for MIMO Systems with Unstructured Uncertainties

In section 9, we considered affine-in-control MIMO systems in the form

$$\dot{x} = A\,x + B\,\Lambda\left(u + f\left(x\right)\right) \tag{11.1}$$

where $x \in R^n$ is the system state vector, $u \in R^m$ is the control input, $B \in R^{n\times m}$ is a known matrix, $A \in R^{n\times n}$ and $\Lambda \in R^{m\times m}$, (a diagonal matrix with positive unknown elements), are unknown matrices.

An <u>*MRAC tracking design*</u> was carried out, assuming that the matched possibly nonlinear uncertain function $f\left(x\right): R^n \to R^m$ could be <u>*exactly*</u> represented by an RBF NN in the form $f\left(x\right) = \Theta^T\,\Phi\left(x\right)$ with constant unknown coefficients $\Theta \in R^{N\times m}$ and a fixed known regressor vector $\Phi\left(x\right) \in R^N$.

In this section, we assume that the unknown function $f\left(x\right)$ can be <u>*approximated*</u> using an RBF NN with known basis functions, (such as Gaussians with fixed centers). In effect, using Universal Approximation property of RBF-s, it is assumed that the unknown mapping $f\left(x\right)$ can be approximated by an RBF NN with N <u>*fixed*</u> neurons $\varphi_i\left(x\right)$ and <u>*ideal unknown constant weights*</u> matrix $\Theta \in R^{N\times m}$

$$f\left(x\right) = \Theta^T\,\Phi\left(x\right) + \varepsilon\left(x\right) \tag{11.2}$$

on a compact domain $X \subset R^n$, and to within a given <u>*approximation tolerance*</u> $\varepsilon_{\max} > 0$, that is:

$$\left\|\varepsilon\left(x\right)\right\| \le \varepsilon_{\max}, \quad \forall\, x \in X \tag{11.3}$$

The <u>*control objective*</u> is to design an adaptive state feedback controller which guarantees boundedness of all variables in the corresponding closed-loop system, while tracking the state $x_{ref} \in R^n$ of the <u>*desired (open-loop asymptotically stable) reference model*</u>

$$\dot{x}_{ref} = A_{ref}\,x_{ref} + B_{ref}\,r\left(t\right) \tag{11.4}$$

which is in turn driven by a <u>*bounded reference signal*</u> $r\left(t\right) \in R^m$. Note that the MRAC controller must operate in the presence of the system structured and unstructured uncertainties, where the latter is represented by the function approximation error vector $\varepsilon\left(x\right) \in R^m$, which satisfies (11.3).

The MRAC solution is chosen in the same form as in Section 9:

$$u = \hat{K}_x^T\, x + \hat{K}_r^T\, r - \hat{\Theta}^T\, \Phi(x) \tag{11.5}$$

where $\hat{K}_x \in R^{n\times m}$, $\hat{K}_r \in R^{m\times m}$, $\hat{\Theta} \in R^{N\times n}$ are the adaptive parameters. In order for the solution to exists the following <u>*model matching assumptions*</u> must hold:

$$\left[\begin{array}{l} A + B\,\Lambda\, K_x^T = A_{ref} \\ B\,\Lambda\, K_r^T = B_{ref} \end{array}\right. \tag{11.6}$$

In, (11.6), K_x, K_r are the ideal feedback / feedforward gain matrices. Note that only existence of the ideal gains is assumed, while their knowledge is not required.

Remark 11.1

In (11.5), $\hat{f}(x) = \hat{\Theta}^T\, \Phi(x)$ represents the RBF NN based on-line function approximator. Moreover, the corresponding function approximation error $\Delta f(x)$ linearly depends on the parameter estimation error $\Delta\Theta$.

$$\Delta f(x) = \hat{f}(x) - f(x) = \underbrace{\left(\hat{\Theta} - \Theta\right)^T}_{\Delta\Theta} \Phi(x) - \varepsilon(x) = \Delta\Theta^T\, \Phi(x) - \varepsilon(x) \tag{11.7}$$

We now proceed with the development of the adaptive laws. The tracking error dynamics is obtained by subtracting (11.4) from (11.1) and using the matching conditions (11.6). It yields:

$$\begin{aligned} \dot{e} &= \left(A + B\,\Lambda\,\hat{K}_x^T\right)x + B\,\Lambda\left(\hat{K}_r^T\, r - \left(\hat{\Theta} - \Theta\right)^T \Phi(x) + \varepsilon(x)\right) - A_{ref}\, x_{ref} - B_{ref}\, r \\ &= \left(A_{ref} + B\,\Lambda\left(\hat{K}_x - K_x\right)\right)x - A_{ref}\, x_{ref} + B\,\Lambda\left(\hat{K}_r - K_r\right)r - B\,\Lambda\left(\hat{\Theta} - \Theta\right)^T \Phi(x) + B\,\Lambda\,\varepsilon(x) \\ &= A_{ref}\, e + B\,\Lambda\left[\left(\hat{K}_x - K_x\right)^T x + \left(\hat{K}_r - K_r\right)^T r - \left(\hat{\Theta} - \Theta\right)^T \Phi(x) + \varepsilon(x)\right] \end{aligned} \tag{11.8}$$

or, equivalently

$$\dot{e} = A_{ref}\, e + B\,\Lambda\left[\Delta K_x^T\, x + \Delta K_r^T\, r - \Delta\Theta^T\, \Phi(x) + \varepsilon(x)\right] \tag{11.9}$$

Let $P = P^T > 0$ be the solution of the Lyapunov equation.

$$P\, A_{ref} + A_{ref}^T\, P = -Q, \quad Q = Q^T > 0 \tag{11.10}$$

Consider the Lyapunov function candidate

$$V\left(e,\Delta K_x,\Delta K_r,\Delta\Theta\right)=e^T P\,e+\operatorname{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\Delta K_x+\Delta K_r^T\,\Gamma_r^{-1}\,\Delta K_r+\Delta\Theta^T\,\Gamma_\Theta^{-1}\,\Delta\Theta\right]\Lambda\right) \quad (11.11)$$

where $\Gamma_x=\Gamma_x^T>0,\quad \Gamma_r=\Gamma_r^T>0,\quad \Gamma_\Theta=\Gamma_\Theta^T>0$ are the rates of adaptation, and

$$\begin{cases}\Delta K_x=\hat{K}_x-K_x\\ \Delta K_r=\hat{K}_r-K_r\\ \Delta\Theta=\hat{\Theta}-\Theta\end{cases} \quad (11.12)$$

are the parameter estimation errors. Also in (11.11), $\operatorname{tr}(\bullet)$ denotes the trace of a matrix. The time derivative of V along the trajectories of the error dynamics (11.9) is given by:

$$\begin{aligned}\dot{V}&=\dot{e}^T P\,e+e^T P\,\dot{e}+2\operatorname{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x+\Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r+\Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\right]\Lambda\right)\\
&=\left(A_{ref}\,e+B\,\Lambda\left(\Delta K_x^T\,x+\Delta K_r^T\,r-\Delta\Theta^T\,\Phi(x)+\varepsilon(x)\right)\right)^T P\,e\\
&+e^T P\left(A_{ref}\,e+B\,\Lambda\left(\Delta K_x^T\,x+\Delta K_r^T\,r-\Delta\Theta^T\,\Phi(x)+\varepsilon(x)\right)\right)\\
&+2\operatorname{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x+\Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r+\Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\right]\Lambda\right)\\
&=e^T\left(A_{ref}\,P+P\,A_{ref}\right)e+2\,e^T P\,B\,\Lambda\left(\Delta K_x^T\,x+\Delta K_r^T\,r-\Delta\Theta^T\,\Phi(x)+\varepsilon(x)\right)\\
&+2\operatorname{tr}\left(\left[\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x+\Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r+\Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\right]\Lambda\right)\end{aligned} \quad (11.13)$$

Regrouping the terms and using (11.10) yields:

$$\begin{aligned}\dot{V}&=-e^T\,Q\,e+2\,e^T P\,B\,\Lambda\,\varepsilon(x)\\
&+\left[2\,e^T P\,B\,\Lambda\,\Delta K_x^T\,x+2\operatorname{tr}\left(\Delta K_x^T\,\Gamma_x^{-1}\,\dot{\hat{K}}_x\,\Lambda\right)\right]\\
&+\left[2\,e^T P\,B\,\Lambda\,\Delta K_r^T\,r+2\operatorname{tr}\left(\Delta K_r^T\,\Gamma_r^{-1}\,\dot{\hat{K}}_r\,\Lambda\right)\right]\\
&+\left[-2\,e^T P\,B\,\Lambda\,\Delta\Theta^T\,\Phi(x)+2\operatorname{tr}\left(\Delta\Theta^T\,\Gamma_\Theta^{-1}\,\dot{\hat{\Theta}}\,\Lambda\right)\right]\end{aligned} \quad (11.14)$$

Using the trace identity, one can write:

$$
\left[\begin{array}{l}
\underbrace{e^T P B \Lambda}_{a^T} \underbrace{\Delta K_x^T x}_{b} = \operatorname{tr}\left(\underbrace{\Delta K_x^T x}_{b} \underbrace{e^T P B \Lambda}_{a^T} \right) \\
\underbrace{e^T P B \Lambda}_{a^T} \underbrace{\Delta K_r^T r}_{b} = \operatorname{tr}\left(\underbrace{\Delta K_r^T r}_{b} \underbrace{e^T P B \Lambda}_{a^T} \right) \\
\underbrace{e^T P B \Lambda}_{a^T} \underbrace{\Delta \Theta^T \Phi(x)}_{b} = \operatorname{tr}\left(\underbrace{\Delta \Theta^T \Phi(x)}_{b} \underbrace{e^T P B \Lambda}_{a^T} \right)
\end{array}\right. \tag{11.15}
$$

Substituting (11.15) into (11.14) results in:

$$
\begin{aligned}
\dot{V} &= -e^T Q e + 2 e^T P B \Lambda \varepsilon(x) \\
&+2 \operatorname{tr}\left(\Delta K_x^T \left[\Gamma_x^{-1} \dot{\hat{K}}_x + x e^T P B \right] \Lambda \right) \\
&+2 \operatorname{tr}\left(\Delta K_r^T \left[\Gamma_r^{-1} \dot{\hat{K}}_r + r e^T P B \right] \Lambda \right) + 2 \operatorname{tr}\left(\Delta \Theta^T \left[\Gamma_\Theta^{-1} \dot{\hat{\Theta}} - \Phi(x) e^T P B \right] \Lambda \right)
\end{aligned} \tag{11.16}
$$

Remark 11.2
If adaptive laws are chosen as in Section 9, that is

$$
\left[\begin{array}{l}
\dot{\hat{K}}_x = -\Gamma_x x e^T P B \\
\dot{\hat{K}}_r = -\Gamma_r r(t) e^T P B \\
\dot{\hat{\Theta}} = \Gamma_\Theta \Phi(x) e^T P B
\end{array}\right.
$$

then the time derivative becomes:

$$
\begin{aligned}
\dot{V} &= -e^T Q e + 2 e^T P B \Lambda \varepsilon(x) \le -\lambda_{\min}(Q) \|e\|^2 + 2 \|P B \Lambda\| \varepsilon_{\max} \|e\| \\
&= -\|e\| \left(\lambda_{\min}(Q) \|e\| - 2 \|P B \Lambda\| \varepsilon_{\max} \right)
\end{aligned}
$$

Consequently, $\dot{V} < 0$ outside of the compact set in the $e-$ domain:

$$
E = \left\{ e \in R^n : \|e\| \le \frac{2 \|P B \Lambda\| \varepsilon_{\max}}{\lambda_{\min}(Q)} \right\}
$$

Hence, one may attempt to argue that the tracking error vector e is UUB. Unfortunately, inside E nothing can be said about the adaptive gains $\hat{K}_x$, $\hat{K}_r$, $\hat{\Theta}$. The adaptive laws are modified as follows:

$$\begin{cases} \dot{\hat{K}}_x = \Gamma_x \operatorname{Proj}\left(\hat{K}_x, -x e^T P B\right) \\ \dot{\hat{K}}_r = \Gamma_r \operatorname{Proj}\left(\hat{K}_r, -r(t) e^T P B\right) \\ \dot{\hat{\Theta}} = \Gamma_\Theta \operatorname{Proj}\left(\hat{K}_r, \Phi(x) e^T P B\right) \end{cases} \tag{11.17}$$

where $\operatorname{Proj}(\bullet, \bullet)$ denotes the <u>Projection Operator</u>. The operator maps two $(n \times N)-$ matrices $\Omega = [\theta_1 \quad \ldots \quad \theta_N] \in R^{n \times N}$ and $Y = [y_1 \quad \ldots \quad y_N] \in R^{n \times N}$ into the $(n \times N)$ – matrix $\operatorname{Proj}(\Omega, Y)$, and it is defined column-wise, that is

$$\operatorname{Proj}(\Omega, Y) = \left(\operatorname{Proj}(\theta_1, y_1) \quad \ldots \quad \operatorname{Proj}(\theta_N, y_N)\right) \tag{11.18}$$

In particular, the operator components are:

$$\operatorname{Proj}(\theta_j, y_j) = \begin{cases} y_j - \dfrac{\nabla f(\theta_j)\left(\nabla f(\theta_j)\right)^T}{\left\|\nabla f(\theta_j)\right\|^2} y_j \, f(\theta_j), & \text{if } \left[f(\theta_j) > 0 \wedge y_j^T \nabla f(\theta_j) > 0 \right] \\ y_j, & \text{if not} \end{cases} \tag{11.19}$$

where $f(\theta_j): R^n \to R$ is a convex function. Given $\theta_j^{\max}$ – the maximum allowable magnitude of the vector θ_j, and $\varepsilon_j > 0$, the function definition is given below.

$$f(\theta_j) = \frac{\left\|\theta_j\right\|^2 - \theta_j^{\max}}{\varepsilon_j \, \theta_j^{\max}} \tag{11.20}$$

Key properties and geometric interpretation of the Projection Operator are discussed in the next section.

References

1. J.J. Slotine, W. Li, Applied Nonlinear Control, Prentice Hall, 1995.
2. H.K. Khalil, Nonlinear Systems, 3rd Edition, Prentice Hall, New Jersey, 2000.
3. S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, Prentice Hall, New Jersey, 1991.