NSL-910503. May, 1991.

Gaussian Networks for Direct Adaptive Control

Robert M. Sanner and Jean-Jacques E. Slotine
Nonlinear Systems Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

A direct adaptive tracking control architecture is proposed and evaluated for a class of continuous-time nonlinear dynamic systems for which an explicit linear parameterization of the uncertainty in the dynamics is either unknown or impossible. The architecture employs a network of gaussian radial basis functions to adaptively compensate for the plant nonlinearities. Under mild assumptions about the degree of smoothness exhibited by the nonlinear functions, the algorithm is proven to be globally stable, with tracking errors converging to a neighborhood of zero.

A constructive procedure is detailed, which directly translates the assumed smoothness properties of the nonlinearities involved into a specification of the network required to represent the plant to a chosen degree of accuracy. A stable weight adjustment mechanism is then determined using Lyapunov theory. The network construction and performance of the resulting controller are illustrated through simulations with example systems.

1 Introduction

The current interest in neural networks, and their ability to construct complex, nonlinear mappings based upon examples, has produced a variety of algorithms for incorporating these models into nonlinear control systems [16, 17, 19, 20, 23, 24, 31]. Most of these designs employ off-line, iterative techniques to adjust the variable parameters of the network so as to develop discrete-time models of the direct or inverse plant dynamics. The resulting trained network is then incorporated as the basis of a nonlinear control law.

These methods of exploiting neural networks for control have several drawbacks. While exact representations of continuous linear systems can be obtained by properly discretizing the plant dynamics, the same is not true for discretizations of continuous, nonlinear physical systems. Further, as a practical matter, neural networks will be most useful for solving real-time control problems if they are implemented in hardware, instead of modeled in software by discrete approximations. These facts suggest examining neural models for control which are amenable to a direct, physical implementation, for example, as the output of the continuous, analog electronic networks suggested by researchers in the field of early vision [12, 28].

Specifically, the goal of the research described in this paper is to develop adaptive architectures capable of exploiting these analog network designs for the control of continuous-time, nonlinear dynamic systems. Our approach is to treat the entire problem in the context of adaptive systems theory, avoiding iterative training procedures in favor of provably stable techniques for tuning the network parameters. While the gradient methods usually employed for adjustment of the parameters have proven to be quite effective in practice, there is currently no systematic method for ensuring when these methods will be successful, especially if the adaptation is carried out on-line in a feedback control system. It was for precisely these reasons that the adaptive control community twenty-five years ago turned away from gradient methods in favor of formal stability theory. For the problem at hand, application of stability theory will allow us not only to guarantee a given level of performance for the system, but will in the process also illuminate the relation between performance and the various free parameters in the design of the network.

There are three key questions which must be answered in attempting to design neural networks for direct adaptive control: do there exist tuned values of the network parameters capable of satisfying the control objectives (this is sometimes called the *matching condition*); how large does the network have to be, in terms of the number of neurons and interconnecting weights required, and how does this scale with the assumed structure of the control objectives; and how can the variable network parameters be adjusted toward their tuned settings in a fashion which guarantees the boundedness of all signals in the closed loop system.

In this paper, we analyze more formally and in greater detail the ideas originally presented in [32]. Specifically, we present a constructive procedure for a class of neural networks which can approximate to a prescribed accuracy the functions required for satisfaction of the control objectives. Since this approximation can be maintained only over a finite subset of the plant state space, to ensure global stability it is necessary to introduce an additional component into the control law, which is capable of stabilizing the dynamics as the neural approximation degrades. To unify these components into a single control law, we propose a novel technique of smoothly blending the two modes to provide a continuous transition from adaptive operation in the region of validity of the network approximation, to a nonadaptive operation in the regions where this approximation is inaccurate. While this continuous blending of adaptive and nonadaptive modes is of interest in its own right in the more general context of adaptive control, it is a requirement for global stability of the neural network control schemes examined below.

Section 2 describes the setting of the control problem to be examined and illustrates the structure of conventional adaptive methods for its solution. Section 3 introduces the use of multivariable Fourier analysis and sampling theory as a method of translating assumed smoothness properties of the plant nonlinearities into a representation capable of uniformly approximating the plant over a compact set. This section then discusses the conditions under which these representations can be mapped onto a neural network with a finite number of components. Section 4 illustrates how these networks may be used as elements of an adaptive tracking control algorithm for a class of nonlinear systems, which will guarantee convergence of the tracking errors to a neighborhood of zero. Section 5 illustrates the method with two examples, and finally, Section 6 closes with some general observations about the proposed controller.

2 Problem Statement

The purpose of this paper is to develop adaptive control algorithms for a class of dynamic systems whose equations of motion can be expressed in the canonical form:

$$x^{(n)}(t) + f(x(t), \dot{x}(t), \ldots, x^{(n-1)}(t)) = bu(t) \tag{2.1}$$

where $u(t)$ is the control input, f is an unknown linear or nonlinear function, and for simplicity in the initial discussion, take $b = 1$ (this condition will be removed in the subsequent development). The control objective is to force the plant state vector, $\mathbf{x} = [x, \dot{x}, \ldots, x^{(n-1)}]^T$, to follow a specified desired trajectory, $\mathbf{x}_d = [x_d, \dot{x}_d, \ldots, x_d^{(n-1)}]^T$. Defining the tracking error vector, $\tilde{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_d(t)$, the problem is thus to design a control law $u(t)$ which ensures that $\tilde{\mathbf{x}}(t) \to \mathbf{0}$ as $t \to \infty$.

One approach to this problem is to take as the control input $u(t) = u_{pd}(t) + u_{ad}(t) + x_d^{(n)}$ where u_{pd} is a "PD" type control component consisting of a linear combination of tracking error states, $u_{ad}(t)$ is an adaptive control law which will attempt to recover, and cancel, the unknown function $f(\mathbf{x})$, and $x_d^{(n)}$ is a feedforward of the nth derivative of the desired trajectory. With use of this control law, (2.1) becomes:

$$\tilde{x}^{(n)}(t) = -\mathbf{k}^T\tilde{\mathbf{x}}(t) + (u_{ad}(t) - f(\mathbf{x}(t)))$$

where $\mathbf{k}$ is a vector of gains chosen to correspond to the coefficients of a Hurwitz polynomial (that is, a polynomial whose roots lie strictly in the open left half of the complex plane). The complete closed loop dynamic equation is thus:

$$\dot{\tilde{\mathbf{x}}}(t) = \Lambda\tilde{\mathbf{x}}(t) + \mathbf{b}(u_{ad}(t) - f(\mathbf{x}(t))) \tag{2.2}$$

where Λ is a matrix in control canonical form, with eigenvalues at the roots of the Hurwitz polynomial which generates $\mathbf{k}$, that is:

$$\Lambda = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -k_0 & -k_1 & -k_2 & \dots & -k_{n-1} \end{bmatrix}$$

and $\mathbf{b} = [0\ 0\ \dots\ 1]^T$.

If it is known *a priori* (as, e.g., in [34]) that $f(\mathbf{x})$ admits an expansion in terms of a set of known (linear or nonlinear) *basis functions*, $Y_i(\mathbf{x})$, i.e.:

$$f(\mathbf{x}) = \sum_{i=1}^{N} a_i Y_i(\mathbf{x}) \tag{2.3}$$

then the adaptive controller can be chosen as:

$$u_{ad}(t) = \sum_{i=1}^{N} \hat{a}_i(t) Y_i(\mathbf{x}(t))$$

where $\hat{a}_i(t)$ is an approximation to the ith coefficient in the expansion of f. Hence,

$$u_{ad}(t) - f(\mathbf{x}(t)) = \sum_{i=1}^{N} \tilde{a}_i(t) Y_i(\mathbf{x}(t))$$

where $\tilde{a}_i(t) = \hat{a}_i(t) - a_i$. The tracking problem can thus be solved if a law for stably adjusting $\hat{a}_i(t)$ can be created which guarantees $\tilde{\mathbf{x}}(t) \to \mathbf{0}$.

The design of such a law is possible if a measure of the tracking error can be found which is sufficiently correlated, in some appropriate sense, with the approximation errors, $\tilde{a}_i(t)$. Given the structure of the error dynamics (2.2), a fundamental result of adaptive control theory is that this correlation condition is satisfied by any linear combination of the error states:

$$s(t) = \boldsymbol{\lambda}^T \tilde{\mathbf{x}}(t)$$

provided that the transfer function $\boldsymbol{\lambda}^T (pI - \Lambda)^{-1} \mathbf{b}$ is *strictly positive real* [22, 35], where p is the Laplace variable.

These facts form the basis for most of the stable adaptive tracking controller designs. The drawback is the relatively large amount of prior information which must be incorporated into these designs in the form of the exact structure of the required basis functions. In the following section, it will be shown that, with much less prior knowledge, these functions can be *approximated to a chosen degree of accuracy* using the output of a class of extremely simple parallel analog networks whose mathematical structure strongly resembles (2.3). These approximations can then be stably tuned, using the methods described above, to produce an effective tracking control architecture.

3 Sampling Theory and Neural Approximation

If the exact basis functions, $Y_i(\mathbf{x})$, are unknown, it may be possible to estimate the required control input using a large collection of simple elementary functions. Memory based methods and their variants [6, 20], partition the state space $\mathcal{R}^n$ into small hypercubes, and attempt to estimate the values assumed by f in each hypercube; the elementary functions in this case are simply the characteristic functions of each hypercube. More generally, when the plant dynamics are not necessarily as in (2.1), these methods can be used to estimate directly the values required for $u_{ad}(\mathbf{x})$ [3, 4, 15]. Potential function techniques [2, 19] similarly partition $\mathcal{R}^n$ but use smoothly decaying elementary functions to provide an explicit means of "spreading" the information learned on a given partition to neighboring points. Neural networks offer yet a third approach, attempting to reconstruct f from compositions and superpositions of simple nonlinearities [21, 30].

These methods are intuitively attractive, but in order to develop stable control laws it is necessary to *guarantee* that the chosen representation can accurately reproduce the required function. For neural networks, the ability to approximate continuous functions to a specified degree of accuracy has been demonstrated by several authors [8, 9, 11], and despite the complexity of some network designs, the same results have shown that these approximation properties hold for networks consisting of a linear superposition of nonlinear nodes, provided a sufficient number of nodes are employed. The approximation implemented by such three layered networks can be represented mathematically as:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N} c_{i2}\, g_i(\mathbf{x}, \mathbf{c}_{i1}) \tag{3.1}$$

where g_i is the elementary function implemented by node i, often taken as a logistic, or "sigmoid", function of the inner product of its arguments. The coefficients $\mathbf{c}_{i1}$ represent the input weights to node i, while c_{i2} represents the output weight for that node.

Theoretically, then, it is possible for a neural network to closely approximate the required control input, $u_{ad}(\mathbf{x})$, provided this function is a continuous function of $\mathbf{x}$. There remains unanswered the important practical question: for a specified tolerance, exactly how many nodes are required to ensure that this theoretical capability is achieved? To answer this question, it is helpful to first briefly review the methods currently employed to train and evaluate networks; the possibility of using these methods for neural controllers can then be evaluated.

Many of the neural approximation algorithms in use employ a "training set" of exemplar values for the function being approximated consisting of the values the network must produce at a finite set of points in its domain. Various heuristics have been suggested relating the size of the training set to the size the network, and iterative gradient methods are usually used to adjust the parameters of the network to minimize the approximation errors at the training points. The ability of a network thus entrained to "generalize" is then expressed as a measure of the approximation errors recorded at domain points not contained in the training set.

For the purposes of control, it is necessary to achieve a high level of accuracy not just at a set of discrete points, but at a continuum of points over as large a subset of the state space as possible; good generalization is not just desirable, it is essential when using networks for control. Further, the points in the "training set" of the neurocontroller will be the continuously evolving values of the plant state vector as it attempts to follow the desired trajectory; what subset of these should be used to design and train the network?

3.1 Smooth Functions and Sampling Theory

One thus encounters immediate problems attempting to apply the existing neural approximation algorithms to neural controllers. In order to address these problems, it is necessary to formally investigate the conditions under which a continuous function can be uniformly approximated using values it assumes at a *discrete* subset of its domain, and how these points should be chosen so as to completely characterize the function.

The ability to accomplish this will certainly depend upon the degree of smoothness exhibited by the function. Smoothness can be quantified in several ways; from an engineering standpoint one of the most familiar and useful of these is through the *spatial Fourier transform*:

$$F(\boldsymbol{\nu}) = \mathcal{F}(f(\mathbf{x})) = \int_{\mathcal{R}^n} f(\mathbf{x})\, e^{-2\pi j \boldsymbol{\nu}^T \mathbf{x}}\, d\mathbf{x}$$

Roughly speaking, large values of F for high values of $\|\boldsymbol{\nu}\|$ indicates that f is much less smooth than a function whose transform nearly vanishes for large $\|\boldsymbol{\nu}\|$; in the limiting case of a function which can assume completely unrelated values for points in $\mathcal{R}^n$ arbitrarily close together, F is a constant for all $\boldsymbol{\nu}$. It is important to note (and this motivates the slight abuse of notation used above) that in this expression f is viewed as a multidimensional function of $\mathbf{x} \in \mathcal{R}^n$ for the purpose of computing (or estimating) its spectral characteristics, and *not* as an implicit function of time, as is more usual in control and filtering applications.

The above transform is certainly not well defined for every function (although the class of applicable functions can be greatly expanded using the theory of distributions [5, 39]); even when it is defined, the inversion formula:

$$f(\mathbf{x}) = \mathcal{F}^{-1}(F(\boldsymbol{\nu})) = \int_{\mathcal{R}^n} F(\boldsymbol{\nu})\, e^{2\pi j \boldsymbol{\nu}^T \mathbf{x}}\, d\boldsymbol{\nu}$$

is not guaranteed to be valid. However, the assumption that the transform exists and that the inversion relation holds for the nonlinearity being approximated forms a good starting point; these conditions can then be weakened in the subsequent development.

Given a spatial frequency domain representation of the function, the question of when a discrete set of points completely determines the behavior of the function is elegantly answered by multidimensional sampling theory [25]. If the support of the Fourier transform of f is compact (i.e., if $F(\boldsymbol{\nu}) = 0$ outside of some bounded set in $\mathcal{R}^n$), sampling theory shows that f can be *exactly* represented in terms of the values it assumes at a countable set of points, with use of appropriately chosen interpolating functions. Further, by a slight re-arrangement of the equations, a much larger class of interpolating functions can be employed which possess the same reconstruction properties as the canonical interpolators. As will be shown below, this extended class of interpolating functions includes several of the neural models currently in widespread use, suggesting that sampling theory can be used to predict the approximating ability of these networks.

Sampling theory reduces the approximation problem to a *countable* number of parameters for certain functions, but this is still too many if the goal is to develop a representation which can be realized with a *finite* number of components. Further, the class of plant nonlinearities which are Fourier transformable and whose transforms have compact support is relatively small. However, the structure of the exact representation produced by the sampling theorem for bandlimited functions suggests methods by which these constraints can be relaxed and the number of parameters reduced. In particular, by limiting the reconstruction to a compact subset of $\mathcal{R}^n$, it is possible to construct a sequence of functions, each uniformly approximating the original function on this set, leading to a final representation for which only a finite number of sampling points are required. The resulting expansion is not exact, but as will be shown, the approximation error can be made vanishingly small on the chosen set.

In the three sections which follow, these observations are made precise and the conclusions used to suggest designs for a class of neural networks. Section 3.2 analyzes canonical reconstruction of bandlimited multivariable functions, and shows how interpolating functions other than the canonical one can be used in the reconstruction. In section 3.3, it is shown that by sacrificing globally exact reconstruction in favor of uniform approximation on a chosen compact set, a much larger class of functions can be approximated, while using only a finite number of interpolating functions. Section 3.4 introduces a particular interpolating function which has found increasing use in neural network designs, and shows how the representation derived from sampling theory maps naturally onto these networks. Finally, Section 3.5 offers some general observations on the ideas introduced in the previous sections. Throughout the development below, it is assumed that the function being approximated is continuous on the subset on which reconstruction is required.

3.2 Bandlimited Functions and Exact Reconstruction

Assume first that f is absolutely integrable (i.e., that $\int |f|$ is finite) and that its spatial Fourier transform has compact support. Under these conditions, the transform pair given above is valid [7, 38], and further, f admits an *exact* expansion in terms of its samples on an appropriately defined lattice in $\mathcal{R}^n$ [25]. To see this, define a sequence of basis vectors for the lattice as $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, and choose the lattice points to be:

$$\boldsymbol{\xi}_I = i_1\Delta_1\mathbf{v}_1 + i_2\Delta_2\mathbf{v}_2 + \cdots + i_n\Delta_n\mathbf{v}_n$$

where $I = \{i_1, \ldots, i_n\}$ is an n-tuple of integers, and the Δ_i are the sample spacings along the lattice basis vectors. For simplicity in this exposition, the sampling is taken as uniform with respect to a square lattice, hence $\{\mathbf{v}_i\}_{i=1}^n = \{\mathbf{e}_i\}_{i=1}^n$, where the $\mathbf{e}_i$ are the standard basis vectors for $\mathcal{R}^n$, and $\Delta_1 = \cdots = \Delta_n = \Delta$.

The sampling operation can be viewed as a modulation of the hypersurface f with a field of Dirac distributions centered at the lattice points, i.e.

$$f_s(\mathbf{x}) = \psi(\mathbf{x}) f(\mathbf{x}), \tag{3.2}$$

where

$$\psi(\mathbf{x}) = \sum_{I \in \mathcal{I}} \delta(\mathbf{x} - \boldsymbol{\xi}_I)$$

and $\mathcal{I} = \mathcal{Z}^n$ (where $\mathcal{Z}$ is the set of all integers). Fourier transforming (3.2), one obtains:

$$F_s(\boldsymbol{\nu}) = F(\boldsymbol{\nu}) * \Psi(\boldsymbol{\nu})$$

where $*$ indicates convolution and,

$$\Psi(\boldsymbol{\nu}) = \frac{1}{\Delta^n} \sum_{I \in \mathcal{I}} \delta(\boldsymbol{\nu} - \boldsymbol{\zeta}_I).$$

Hence

$$F_s(\boldsymbol{\nu}) = \frac{1}{\Delta^n} \sum_{I \in \mathcal{I}} F(\boldsymbol{\nu} - \boldsymbol{\zeta}_I), \tag{3.3}$$

implying that the spectrum of the sampled waveform consists of copies of the original spectrum centered at the frequency lattice points, $\boldsymbol{\zeta}_I = \Delta^{-1}(i_1 \mathbf{v}_1 + i_2 \mathbf{v}_2 + \cdots + i_n \mathbf{v}_n)$.

Let B be the smallest n-cube, centered at the origin, which completely encloses the support of $F(\boldsymbol{\nu})$, and let $\beta = \mathrm{Radius}(B)$. If Δ is chosen to be less than or equal to $1/(2\beta)$, the copies of $F(\boldsymbol{\nu})$ in the sampled spectrum will be non-overlapping, and the canonical reconstruction function, whose spectrum is Δ^n on B and zero elsewhere, can be used to assemble $f(\mathbf{x})$ on the basis of its samples, since, under these conditions:

$$F(\boldsymbol{\nu}) = F_s(\boldsymbol{\nu})\, G_c(\boldsymbol{\nu}) \tag{3.4}$$

where $G_c(\boldsymbol{\nu})$ is the spectrum of the canonical reconstructor. Hence, inverse transforming leads to the canonical result:

$$f(\mathbf{x}) = \sum_{I \in \mathcal{I}} f(\boldsymbol{\xi}_I)\, g_c(\mathbf{x} - \boldsymbol{\xi}_I) \tag{3.5}$$

The structure of equality (3.4), however, suggests that many other functions can potentially be used for the reconstruction. If the hypersurface is oversampled, i.e. if $\Delta < 1/(2\beta)$, then, defining $C_\Delta = [-(2\Delta)^{-1}, (2\Delta)^{-1}]^n \supset B$, the spectrum of the reconstructor is arbitrary on $C_\Delta - B$. In fact, as long as its spectrum is bounded, real valued and strictly positive on B, and vanishes outside of C_Δ, an arbitrary (Fourier transformable) function can be used in place of g_c in expansion (3.5), although the coefficients in this expansion will no longer be samples of f itself.

To see this, assume that $g_1(\mathbf{x})$ is such a function, and using its spectrum, define a new function $c(\mathbf{x}) = \mathcal{F}^{-1}(C(\boldsymbol{\nu}))$, where:

$$C(\boldsymbol{\nu}) = \Delta^n\, F(\boldsymbol{\nu})\, G_1^{-1}(\boldsymbol{\nu}) \quad \text{if} \ \ \boldsymbol{\nu} \in B \tag{3.6}$$

and $C(\boldsymbol{\nu}) = 0$ otherwise. Since, by assumption $G_1(\boldsymbol{\nu})$ is strictly real and positive on B, this new spectrum is well-defined; further, $C(\boldsymbol{\nu})$ is bounded since both $F(\boldsymbol{\nu})$ and $G_1^{-1}(\boldsymbol{\nu})$ are on B. Hence $C(\boldsymbol{\nu})$ is absolutely integrable, and its inverse transform $c(\mathbf{x})$ is a bounded, continuous function.

Consider applying the above sampling algorithm to $c(\mathbf{x})$. It is straightforward to verify from the definitions of $C(\boldsymbol{\nu})$ and $G_1(\boldsymbol{\nu})$ that:

$$F(\boldsymbol{\nu}) = C_s(\boldsymbol{\nu})\, G_1(\boldsymbol{\nu}) \tag{3.7}$$

where $C_s(\boldsymbol{\nu})$ is the sampled spectrum of $c(\mathbf{x})$, defined as in (3.3). Inverse transforming this equation reveals an exact expansion of $f(\mathbf{x})$ in terms of the samples of the bounded function $c(\mathbf{x})$:

$$f(\mathbf{x}) = \mathcal{F}^{-1}\left[C_s(\boldsymbol{\nu}) G_1(\boldsymbol{\nu})\right] = c_s(\mathbf{x}) * g_1(\mathbf{x}) = (c(\mathbf{x})\psi(\mathbf{x})) * g_1(\mathbf{x})$$

thus,

$$f(\mathbf{x}) = \sum_{I\in\mathcal{I}} c(\boldsymbol{\xi}_I) g_1(\mathbf{x} - \boldsymbol{\xi}_I). \tag{3.8}$$

Figure 1 illustrates these ideas for a one-dimensional function with a spectrum $F(\nu) = 1 - |\nu|$ for $|\nu| \leq 1$ and $F(\nu) = 0$ otherwise. The sampling mesh in this figure is $\Delta = 1/2$ and the canonical reconstructing spectrum $G_c(\nu) = 1/2$ for $|\nu| \leq 1$, and $G_c(\nu) = 0$ otherwise is illustrated. Figure 2 illustrates reconstruction of the same function using instead an interpolating function whose spectrum is $G_1(\nu) = \exp(-\nu^2)$. The function $C(\nu)$ is given in this example by

$$C(\nu) = 1/5\ \exp(\nu^2)(1 - |\nu|) \quad \text{for} \quad |\nu| < 1, \quad \text{and} \quad C(\nu) = 0 \quad \text{otherwise.}$$

which is then oversampled with mesh $\Delta = 1/5$.

Notice that the spectrum G_1 does *not* vanish identically outside of C_Δ in this case, however at the beginning of the first copy in the repeating spectrum of $c(\mathbf{x})$ one has $G_1 = \exp(-16) \approx 10^{-7}$, so that the spurious high frequency components are almost completely negligible in the reconstruction. This suggests that it is possible to extend the class of interpolating functions still further, if vanishingly small approximation errors are allowed. The next section makes this observation precise, and derives general conditions when such approximations are valid.

3.3 Approximate Reconstruction on Compact Sets

In this section it is shown that by sacrificing the globally exact representation afforded by expansion (3.8), it is possible to *approximate* a much larger class of functions than considered in the previous section, using a wider range of interpolating functions. The principal result demonstrated is that if the function can be smoothly truncated outside a compact set in such a way that the spatial Fourier transform of this truncation is absolutely integrable, then the function itself can be uniformly approximated to a chosen degree of accuracy on that set with a finite linear superposition of interpolating functions. Any interpolating function is admissible, provided that its spectrum is real-valued, strictly positive, and integrable on $\mathcal{R}^n$.

As noted above, the actual nonlinearity f driving the plant dynamics may not be Fourier transformable, or if transformable, the inversion equation might not hold. However, if f is absolutely integrable, with an absolutely integrable spectrum, then the Fourier transform pair above is valid [38]. Now, absolute integrability of the spectrum reflects local properties of the function capturing its degree of smoothness. Absolute integrability of the function itself, however, is a global condition which is easily relaxed: if the original function f does not satisfy this conditions, create an absolutely integrable function, f_F, which is exactly equal to f on a chosen compact set, A; for example:

$$f_F(\mathbf{x}) = m_1(\mathbf{x}) f(\mathbf{x}),$$

where $m_1(\mathbf{x})$ is an infinitely smooth function which is unity on A, vanishes outside a slightly larger set $A_1 \supset A$, and is such that $0 \leq m_1(\mathbf{x}) \leq 1$ on $A_1 - A$ (such classes of so-called "Schwartz functions" can easily be shown to exist; see, e.g. [1, 36]). One thus has

$$|f(\mathbf{x}) - f_F(\mathbf{x})| = |1 - m_1(\mathbf{x})|\,|f(\mathbf{x})| = \alpha_1(\mathbf{x})$$

where $\alpha_1(\mathbf{x}) = 0$ if $\mathbf{x} \in A$, and otherwise $\alpha_1(\mathbf{x}) \leq |f(\mathbf{x})|$.

While the transform of this f_F exists, it may approach zero only asymptotically. In order to extend the class of reconstructors which can be used, allow the same to be true of the transforms of the functions g_1. Under these conditions, the inversion formula used to create (3.8) might not be valid. As is now demonstrated, however, if the rates of decrease of $|F_F(\boldsymbol{\nu})|$ and $|G_1(\boldsymbol{\nu})|$ are sufficiently fast as $\|\boldsymbol{\nu}\| \to \infty$, then not only is the inversion formula valid, but also choice of a sufficiently fine sampling mesh on $\mathcal{R}^n$ results in an expansion, $f_A(\mathbf{x})$, in terms of the chosen interpolating functions g_1, which can uniformly approximate the actual function to a specified accuracy on a given compact set, i.e. $|f_F - f_A| \leq \epsilon_f$.

To prove this assertion, define $f_{BL}(\mathbf{x})$ (read "f bandlimited") to be the function which would obtain by truncating the spectrum of f_F at a particular spectral radius, i.e.

$$f_{BL}(\mathbf{x}) = \mathcal{F}^{-1}(F_F(\boldsymbol{\nu})\chi_B(\boldsymbol{\nu}))$$

where $\chi_B(\boldsymbol{\nu})$ is the characteristic function of the hypercube, B, of radius β in frequency, that is, $\chi_B(\boldsymbol{\nu}) = 1$ if $\boldsymbol{\nu} \in B$, and $\chi_B(\boldsymbol{\nu}) = 0$ otherwise. It is then easy to show

$$|f_F(\mathbf{x}) - f_{BL}(\mathbf{x})| \leq \|F_F(\boldsymbol{\nu})\,\chi_{B^c}(\boldsymbol{\nu})\|_1$$

where B^c is the complement of the set B in $\mathcal{R}^n$, i.e. $B^c = \mathcal{R}^n - B$. Hence, if the spectrum $F_F(\boldsymbol{\nu})$ is absolutely integrable on $\mathcal{R}^n$, it is possible to choose the truncation radius β so that $|f_F(\mathbf{x}) - f_{BL}(\mathbf{x})|$ is as small as desired, uniformly on $\mathcal{R}^n$, say $|f_F(\mathbf{x}) - f_{BL}(\mathbf{x})| \leq \epsilon_f/3$.

Given this value of β, choose a $g_1(\mathbf{x})$ which satisfies the above requirements for reconstructing $f_{BL}(\mathbf{x})$, that is, $G_1(\boldsymbol{\nu})$ is real and positive for $\boldsymbol{\nu} \in B$, but which perhaps approaches zero only asymptotically on B^c, and define $C(\boldsymbol{\nu})$ by applying (3.6) to $f_{BL}(\mathbf{x})$. Sampling the resulting function $c(\mathbf{x})$ with a mesh size $\Delta \leq 1/(2\beta)$, produces an expansion, $f_R(\mathbf{x})$ ("f reconstructed"), which can be expressed (using (3.8)) as:

$$
\begin{aligned}
f_R(\mathbf{x}) &= \sum_{I\in\mathcal{I}} c(\boldsymbol{\xi}_I) g_1(\mathbf{x}-\boldsymbol{\xi}_I) = \int_{\mathcal{R}^n} G_1(\boldsymbol{\nu}) C_s(\boldsymbol{\nu}) e^{2\pi j \boldsymbol{\nu}^T \mathbf{x}} d\boldsymbol{\nu} \qquad (3.9)\\
&= \int_{C_\Delta} G_1(\boldsymbol{\nu}) G_1^{-1}(\boldsymbol{\nu}) F_{BL}(\boldsymbol{\nu}) e^{2\pi j \boldsymbol{\nu}^T \mathbf{x}} d\boldsymbol{\nu} + \int_{C_\Delta} G_1(\boldsymbol{\nu}) \left[\sum_{I\neq\mathbf{0}} C(\boldsymbol{\nu}-\boldsymbol{\zeta}_I)\right] e^{2\pi j \boldsymbol{\nu}^T \mathbf{x}} d\boldsymbol{\nu}\\
&+ \int_{C_\Delta^c} G_1(\boldsymbol{\nu}) \left[\sum_{I\in\mathcal{I}} C(\boldsymbol{\nu}-\boldsymbol{\zeta}_I)\right] e^{2\pi j \boldsymbol{\nu}^T \mathbf{x}} d\boldsymbol{\nu},
\end{aligned}
$$

where the definition of $C(\boldsymbol{\nu})$ has been used.

Now, the first integral is exactly $f_{BL}(\mathbf{x})$, since $F_{BL}(\boldsymbol{\nu})$ vanishes outside of B by construction, and the choice of the sampling rate ensures that the second integral is identically zero. The last integral represents the error in approximating the bandlimited function $f_{BL}(\mathbf{x})$ with a reconstructing function which is not itself bandlimited. Since $C(\boldsymbol{\nu})$ is bounded by construction by some positive constant C_{max}, and since, by the choice of mesh size, the repeating spectra in the summation are non-overlapping, one has

$$|\sum_{I\in\mathcal{I}} C(\boldsymbol{\nu}-\boldsymbol{\zeta}_I)| \leq C_{max}$$

for each frequency point $\boldsymbol{\nu}$. Thus, if $G_1(\boldsymbol{\nu})$ is also absolutely integrable, it is possible to chose Δ (which defines the domain of integration C_Δ^c) such that this error is as small as desired. Hence, by appropriate choice of the mesh size, $|f_R(\mathbf{x}) - f_{BL}(\mathbf{x})|$ can be made arbitrarily small, again uniformly on $\mathcal{R}^n$, for example $|f_R(\mathbf{x}) - f_{BL}(\mathbf{x})| \leq \epsilon_f/3$.

Finally, truncation error of series (3.9) with absolute magnitude less than or equal to $\epsilon_f/3$ can be achieved at each $\mathbf{x}$, by omitting the terms corresponding to samples $\boldsymbol{\xi}_I$ which lie outside an n-ball of radius ρ surrounding $\mathbf{x}$. To achieve this level of reconstruction everywhere on a particular compact set A, take the union of all such n-balls centered at each point $\mathbf{x} \in A$. This union defines a new set Γ which contains only a finite number of mesh points.

Let $f_A(\mathbf{x})$ be the truncated series obtained by restricting to Γ the summation in the expansion of $f_R(\mathbf{x})$, i.e.:

$$f_A(\mathbf{x}) = \sum_{I\in\mathcal{I}_o} c(\boldsymbol{\xi}_I) g_1(\mathbf{x}-\boldsymbol{\xi}_I), \qquad (3.10)$$

where the index set $\mathcal{I}_o = \{I \mid \boldsymbol{\xi}_I \in \Gamma\}$. By construction

$$|f_R(\mathbf{x}) - f_A(\mathbf{x})| \leq \epsilon_f/3 + \alpha_2(\mathbf{x}),$$

where $\alpha_2(\mathbf{x}) = 0$ if $\mathbf{x} \in A$, and when $\mathbf{x} \in A^c$, $\alpha_2(\mathbf{x})$ provides an upper bound on the approximation error resulting from the terms omitted in the series expansion. It follows that, for all $\mathbf{x} \in A$:

$$
\begin{aligned}
|f(\mathbf{x}) - f_A(\mathbf{x})| &\leq |f(\mathbf{x}) - f_F(\mathbf{x})| + |f_F(\mathbf{x}) - f_{BL}(\mathbf{x})| + |f_{BL}(\mathbf{x}) - f_R(\mathbf{x})| + |f_R(\mathbf{x}) - f_A(\mathbf{x})|\\
&\leq \alpha_1(\mathbf{x}) + \epsilon_f/3 + \epsilon_f/3 + \epsilon_f/3 + \alpha_2(\mathbf{x})\\
&= \epsilon_f + \alpha(\mathbf{x}) \qquad (3.11)
\end{aligned}
$$

where $\alpha = \alpha_1 + \alpha_2$ is such that $\alpha(\mathbf{x}) = 0$ if $\mathbf{x} \in A$.

Sampling theory thus predicts that a continuous function observing fairly mild smoothness constraints can be approximated to a prespecified accuracy everywhere on a compact subset of $\mathcal{R}^n$ using only a finite number of components. It should be noted that these sampling ideas apply even to what appear to be pathologically degenerate examples, for example

$$f(x_1, x_2) = \frac{\sin(4\pi x_1)}{\pi x_1}$$

since this expression can be seen as a product of a bandlimited function of x_1, and a function of x_2 which is constant (in this case unity), and can hence be approximated on a compact subset of $\mathcal{R}^2$ by the methods detailed above.

3.4 Reconstruction with Gaussian Radial Basis Functions

In the preceding two sections, use of sampling theory and Fourier analysis has established the existence of a large class of representations capable of uniformly approximating smooth functions on compact sets. In this section, a precise correspondence is established between the elements of these representations and the components of a class of three layered neural networks.

If a function has a Fourier transform with compact support, then it is infinitely differentiable [5]. Poggio's research into neural approximation using regularization theory [28] has demonstrated that, under this latter assumption, a linear superposition of *gaussian radial basis functions* results in an optimal mean square approximation to an unknown function whose values are specified at a finite set of points in $\mathcal{R}^n$. Further, Girosi and Poggio [10] have proven that any continuous function, not necessarily infinitely smooth, can be uniformly approximated by linear combinations of gaussians.

Seen in the context of sampling theory, gaussian radial basis functions are particularly attractive: they are bounded, strictly positive and absolutely integrable on $\mathcal{R}^n$, and further, they are their own Fourier transforms (modulo a scale factor). The radial gaussian satisfies each of the criteria discussed in the previous section, and hence can be used as the reconstructing function in expansion (3.10). Viewed in this light, Girosi and Poggio's proof of the approximating abilities of gaussians agrees with the above observation regarding approximations of functions whose spectra vanish only asymptotically.

Hence, if the truncation of f outside a compact set produces a function with a bounded, absolutely integrable spatial Fourier transform, f can be uniformly approximated on that set using a finite number of evaluations of the radial gaussian:

$$f_A(\mathbf{x}) = \sum_{I \in \mathcal{I}_o} c_I \, g_\sigma(\mathbf{x}, \boldsymbol{\xi}_I) \tag{3.12}$$

where $c_I = c(\boldsymbol{\xi}_I)$. Unlike other reconstructing functions which satisfy the constraints discussed in Section 3.2, for example, n-fold products of sinc or Bessel functions [7], whose structure changes depending upon the dimension of the underlying space, the radial gaussian and its spectrum have exactly the same representation regardless of dimension:

$$g_\sigma(\mathbf{x}, \boldsymbol{\xi}) = \exp(\frac{-\|\mathbf{x}-\boldsymbol{\xi}\|^2}{\sigma^2}) = \exp(-\frac{(\mathbf{x}-\boldsymbol{\xi})^T(\mathbf{x}-\boldsymbol{\xi})}{\sigma^2}) \tag{3.13}$$

(in analog vision terminology, $\boldsymbol{\xi}$ is the "center" of the gaussian "receptive field", and σ is its "width"). The only change which occurs as the dimension of $\mathbf{x}$ increases is in the computation of the inner product. Further, the gaussian is a *separable* nonlinearity, in that

$$g_\sigma(\mathbf{x}, \boldsymbol{\xi}) = g_\sigma(x_1, \xi_1)\, g_\sigma(x_2, \xi_2) \cdots g_\sigma(x_n, \xi_n),$$

or any intermediate combination of such terms, so that instead of computing a single nonlinear transform of the entire state vector, individual subspaces of $\mathcal{R}^n$ may be transformed separately, then multiplied together to form the final approximation [28].

These properties give the gaussian a particularly "neural" character, especially amenable to implementation in parallel analog computing. As Poggio points out [28, 29], expansion (3.12) maps directly onto a class of three layer networks employing a single layer of "neurons" (or nodes) with radial gaussian input-output characteristics. The current point, $\mathbf{x}$, at which the approximation $f_A(\mathbf{x})$ is required, forms the input to the network. There are a number of nodes in the network equal to the number of elements in the index set $\mathcal{I}_o$, and the weights connecting the network inputs to each gaussian node encode the points $\boldsymbol{\xi}_I$ in the sampling grid. Each node computes a quadratic "activation energy" from its inputs and feed-in weights, given by:

$$r_I^2 = \|\mathbf{x}-\boldsymbol{\xi}_I\|^2 = (\mathbf{x}-\boldsymbol{\xi}_I)^T(\mathbf{x}-\boldsymbol{\xi}_I)$$

and outputs the gaussian of this activation, $\exp(-r_I^2/\sigma^2)$. The network output is formed by weighting each nodal output by the corresponding c_I, and summing together all the weighted outputs. The result represents the approximation $f_A(\mathbf{x})$.

While gradient methods are usually employed for determining the centers ($\boldsymbol{\xi}_I$) and variances (σ^2) of the gaussians, the sampling theory interpretation exactly specifies the required values for these parameters, given the assumed spectral support of f. The $\boldsymbol{\xi}_I$ encode a rectangular sampling mesh on $\mathcal{R}^n$, and the number of nodes is determined by the size of the region on which f must be approximated. The required variance of the gaussians can be computed starting with their frequency domain representation, $G_\sigma(\boldsymbol{\nu})$:

$$G_\sigma(\boldsymbol{\nu}) = \exp(-\frac{\pi\boldsymbol{\nu}^T\boldsymbol{\nu}}{\sigma_\nu^2})$$

which upon inverse transforming gives[7]:

$$g_\sigma(\mathbf{x}, \boldsymbol{\xi}_I) = k\ \exp(-\pi\sigma_\nu^2 r_I^2),$$

which is (3.13) with $\sigma^2 = (\pi\sigma_\nu^2)^{-1}$. The constant k is a scale factor which, without loss of generality, can be absorbed into the coefficients c_I in (3.12).

The sampling density and variance should be chosen such that the gaussian's spectrum is strictly positive on the support of $F_{BL}(\boldsymbol{\nu})$, yet approaches zero quickly enough that the high frequency components in $C_s(\boldsymbol{\nu})$ contribute negligibly in the inverse transform. As a practical matter, the following combination has been found to work well:

$$\sigma_v^2 = \beta^2\pi \quad \text{and} \quad \Delta^2 = \frac{1}{8\sigma_v^2}. \tag{3.14}$$

3.5 Remarks

The Fourier transform is a particular instance of a multidimensional integral transform, attractive because of its physically intuitive basis and familiarity to control engineers. Messner and Horowitz [18, 13, 14] have done extensive research into adaptive control using a much more general class of integral transforms, and have derived stable adaptation algorithms for these controllers. In applying their theories to dynamic systems following repetitive trajectories, they employed a gaussian function of time as the approximating element, although without formal justification for the use of this particular function, nor for the discretization chosen in the actual implementation of their algorithm.

Any additional information known about f can be used in sharpening the above bounds or reducing the size of the required representation. If information about the shape and orientation of the spectrum is known, the sampling lattice and mesh size can be adjusted accordingly. For example, if f is known to have a support which is much smaller in one direction in $\mathcal{R}^n$ the mesh size for the corresponding direction in the sampling grid on $\mathcal{R}^n$ can be taken proportionally larger, reducing the number of nodes required.

For an arbitrary function, it may be difficult to establish precisely the tightest possible bounds ϵ_f on the approximation errors. However, if the assumed value of β accurately reflects the frequency content of f_F, it can be expected that the bound ϵ_f will be dominated by the error introduced in truncating the infinite series (3.9). By estimating the magnitude of the contributions of the omitted nodes to points in the set A, a conservative upper bound on this truncation error can be established. From the standpoint of the the adaptive control algorithms developed below, it is only necessary that the crude bound thus developed exceed the true uniform bound on the approximation errors.

By contrast, the approximation error outside of the set A represented by $\alpha(\mathbf{x})$ is of great importance. This function reflects the fact that the accuracy of the estimate implemented by the representations considered above may degrade rapidly outside A. It is easy to see that the worst case bound on this degradation, $\alpha(\mathbf{x}) = |f(\mathbf{x})| + |f_A(\mathbf{x})|$, can be achieved, especially in the case of gaussian interpolating functions. Since the approximation implemented by a gaussian network decays smoothly to zero as $\mathbf{x}$ moves radially away from the set A, a function, f, which changes sign quickly just outside of A would result in the above equality for $\alpha(\mathbf{x})$. The output of the gaussian network approximation, f_A, in this situation would be incorrect both in magnitude *and* in sign. This rapid degradation of the neural approximation outside of the subset for which the network was designed will significantly influence the adaptive control algorithms which the following section now presents in detail.

4 Adaptive Control with Gaussian Networks

In this section, the above discussion of the approximating power of neural networks is developed into a complete specification of an adaptive tracking control architecture based upon these models, which is then proven effective using Lyapunov stability theory. Following some initial observations regarding the unique aspects of neural adaptive control, Section 4.1 presents an overview of the elements of the proposed controller. Section 4.2 discusses the design specifics for the unity gain plant dynamics of (2.1), and formally proves the stability of the design and the convergence of the tracking errors to a neighborhood of zero. Section 4.3 then extends the design and stability proof developed in Section 4.2 to include plants with non-unity control gain. Finally, Section 4.4 concludes with some general observations about the performance of the controller and the details of its implementation.

To begin exploiting the results of Section 3, notice that if the desired trajectories are contained in a compact subset, A_d, of the state space, in principle the tracking problem posed by system (2.1) could be solved by an adaptive component in the control law capable of reconstructing the unknown function f everywhere on A_d. Since expansion (3.12) provides a network structure which can approximate (sufficiently smooth) unknown plant nonlinearities with a prespecified accuracy, uniformly on a particular compact subset, the adaptive component can be taken as the output of this network, instead of the exact basis functions $Y_i(\mathbf{x})$ used in Section 2. The small approximation errors thus introduced could then be viewed as a uniformly bounded disturbance driving the plant dynamics, and standard results of deadzone adaptation [22, 26] used to derive stable adaptive laws.

However, it is possible that, using such a scheme, the tracking errors would become sufficiently large during the initial stages of adaptation that the plant state would leave the set A_d. Further, impulsive, unmodeled disturbances might also take the plant state outside this set. From inequality (3.11), the ability of a neural adaptive element to approximate f degrades rapidly as the state moves outside of the region for which the network was designed. In fact, the discussion of the previous section indicates that it is possible for the approximation implemented by such an adaptive system, as $\mathbf{x}$ moves beyond A_d, to be incorrect not only in relative magnitude, but also in sign, indicating that the contribution of the adaptive system in this case would force the dynamics in precisely the wrong direction. Simple deadzone adaptation will not suffice to ensure stability of the system under these conditions.

The local nature of the approximation implemented by the adaptive component, and the possibly rapid degradation of this approximation outside a given subset of the state space, thus presents an additional complexity in the design of a globally stabilizing controller. This problem can easily be overcome, however, by including in the control law an additional component which takes over from the adaptive component as its approximation ability begins to degrade, and forces the plant state back into A_d. This component takes the form of a sliding controller with a boundary layer [33], and its design is straightforward given upper bounds on the magnitudes of the functions being approximated. The action of this sliding term will be shown to be sufficient to reduce the tracking error during the times when the plant state lies outside of A_d, while the boundary layer prevents control chattering in these regions.

To prevent unnecessary contention between the sliding component and the (typically erroneous) output of the adaptive subsystem when the plant state lies outside of the set A_d, the entire adaptive subsystem, both its output and its adjustment mechanism, should be turned off when the sliding controller is active. Similarly, it is beneficial to turn off the sliding controller when the approximation implemented by the adaptive controller is accurate, since the sliding controller relies on crude upper bounds of the plant nonlinearities to reduce tracking errors and is hence likely to require large amounts of control authority when active.

The complete control law thus has a dual character, acting as a either a sliding or as an adaptive controller depending upon the instantaneous location of the plant state vector. However, discontinuously switching between adaptive and sliding components creates the possibility that the controller might chatter along the boundary between the two types of operation. To prevent this, pure sliding operation is restricted to the exterior of a slightly larger set A, containing A_d, while pure adaptive operation is restricted to the interior of the set A_d; in between, in the region $A - A_d$, the two modes are effectively *blended* using a continuous *modulation function*, which controls the degree to which each component contributes to the complete control law. The resulting controller smoothly transitions between adaptive and nonadaptive control strategies and, as will be shown below, is capable of globally stabilizing the systems under consideration.

4.1 Controller Architecture

By designing the adaptive neurocontroller to approximate plant nonlinearities on the slightly larger region A, the smooth integration of adaptive and sliding components into a single control law, together with negative feedback of the tracking error states, will be sufficient to globally stabilize the controlled system, and asymptotically reduce the tracking errors to a neighborhood of the origin, as will be proven in the sections which follow. This section presents the general structure of the control law and its constituent components, and introduces some concepts and signals which will be useful in the specific designs which follow.

The structure of the conventional adaptive solutions to the tracking problem examined in Section 2, combined with the above observations, suggests a control law with the general structure

$$u(t) = u_{pd}(t) + (1 - m(t))\, u_{ad}(t) + m(t)\, u_{sl}(t).$$

As in Section 2, $u_{pd}(t)$ is a negative feedback term consisting of a weighted combination of both the measured tracking error states and a tracking metric $s(t)$, to be defined below. The term $u_{sl}(t)$, represents the sliding component of the control law, and similarly the adaptive component is represented by $u_{ad}(t)$. The function $m(t) = m(\mathbf{x}(t))$ is a continuous, state dependent modulation which allows the controller to smoothly transition between sliding and adaptive modes of operation, chosen so that $m(\mathbf{x}) = 0$ on A_d, $m(\mathbf{x}) = 1$ on A^c, and $0 < m(\mathbf{x}) < 1$ on $A - A_d$.

To aid in the construction of this modulation, A_d should be chosen so as to correspond to a unit ball with respect to an appropriate weighted norm function. This certainly poses no problem since, as long as A_d contains all the trajectories the plant will be required follow, and A is chosen to completely contain A_d, the actual dimensions of these sets are arbitrary. Thus, A_d and A can be defined by

$$A_d = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}_0\|_{p,w} \leq 1\} \quad \text{and} \quad A = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}_0\|_{p,w} \leq 1 + \Psi\}.$$

Here Ψ is a positive constant representing the width of the transition region, $\mathbf{x}_0$ fixes the absolute location of the sets in the state space of the plant, and $\|\mathbf{x}\|_{p,w}$ is a weighted p-norm of the form

$$\|\mathbf{x}\|_{p,w} = \left\{ \sum_{i=1}^{n} \left(\frac{|x_i|}{w_i} \right)^p \right\}^{\frac{1}{p}},$$

or, in the limiting case $p = \infty$,

$$\|\mathbf{x}\|_{\infty,w} = \max\left(\frac{|x_1|}{w_1}, \frac{|x_2|}{w_2}, \ldots, \frac{|x_n|}{w_n}\right),$$

for a set of strictly positive weights $\{w_i\}_{i=1}^n$. In $\mathcal{R}^2$, for example, with $p = 2$ the sets A_d and A are ellipses, and with $p = \infty$ these sets are rectangles. With these definitions, the modulation function can be taken as,

$$m(\mathbf{x}(t)) = \max\left(0,\ \mathrm{sat}\left(\frac{(r(t) - 1)}{\Psi}\right)\right), \tag{4.1}$$

where $r(t) = \|\mathbf{x}(t) - \mathbf{x}_0\|_{p,w}$, and sat is the saturation function ($\mathrm{sat}(y) = y$ if $|y| < 1$, and $\mathrm{sat}(y) = \mathrm{sign}\,(y)$ otherwise). When $r(t) \leq 1$, meaning that $\mathbf{x} \in A_d$, the output of the saturation function is negative, hence the maximum which defines $m(t)$ is zero, as desired. When $r(t) \geq 1+\Psi$, corresponding to $\mathbf{x} \in A^c$, the saturation function is unity, hence $m(t) = 1$, again as desired. In between, for $\mathbf{x} \in A - A_d$, it is easy to check that $0 < m(\mathbf{x}) < 1$.

To illustrate the structure of the adaptive controller, consider for example the unity control gain system represented in (2.1). The adaptive control component, $u_{ad}(t)$, for this system would implement an approximation, $\hat{f}_A(\mathbf{x})$, to the plant nonlinearity $f(\mathbf{x})$, which is realized as the output of the radial gaussian network described in Section 3.4, whose inputs are the measured values of the plant states. The variance and mesh size of this network are designed considering assumed upper bounds on the spectral properties of the plant nonlinearity $f(\mathbf{x})$, with use of equation (3.14). Hence:

$$u_{ad}(t) = \hat{f}_A(\mathbf{x}(t)) = \sum_{I \in \mathcal{I}_o} \hat{c}_I(t)\, g_\sigma(\mathbf{x}(t), \boldsymbol{\xi}_I). \tag{4.2}$$

The input weights, $\boldsymbol{\xi}_I$, which encode the sampling mesh, are *fixed* in this architecture, while the output weights, $\hat{c}_I(t)$, are to be adjusted to attempt to match their tuned values, c_I. The set Γ (defined in Section 3.3), chosen so that the bound on the uniform approximation error, ϵ_f, committed by the tuned network on the set A is small enough to allow the desired tracking accuracy, specifies the subset $\mathcal{I}_o$ of the uniform, square sampling mesh in (4.2), and hence determines the number of nodes required and the precise values of the corresponding

$\boldsymbol{\xi}_I$. For reference, Figure 3 illustrates the relations among the three sets A_d, A, and Γ, and the parameters controlling their design.

A useful tracking error metric for both the sliding and adaptive control subsystems is defined by

$$s(t) = (\frac{d}{dt} + \lambda)^{n-1}\tilde{x}(t) \text{ with } \lambda > 0,$$

which can be rewritten as $s(t) = \boldsymbol{\lambda}^T\tilde{\mathbf{x}}(t)$ with $\boldsymbol{\lambda}^T = [\lambda^{n-1}, (n-1)\lambda^{n-2}, \ldots, 1]$. The equation $s(t) = 0$ defines a time-varying hyperplane in $\mathcal{R}^n$ on which the tracking error vector decays exponentially to zero, so that perfect tracking can be asymptotically obtained by maintaining this condition [37, 33]. Further, if the magnitude of s can be shown to be bounded by a constant Θ, the actual tracking errors can be shown [33] to be asymptotically bounded by:

$$|\tilde{x}^{(i)}(t)| \leq 2^i\lambda^{i-n+1}\Theta, \qquad i = 0, \ldots, n-1. \tag{4.3}$$

Using the metric $s(t)$ and the saturation function defined above, the sliding control component can be represented as

$$u_{sl}(t) = -k_{sl}(\mathbf{x}(t)) \;\, \mathrm{sat}(s(t)/\Phi) \tag{4.4}$$

where $k_{sl}(\mathbf{x})$ is the gain of the controller, and Φ is the boundary layer width; the exact values required for each of these parameters will be specified in the specific designs which follow.

The deadzones required in the algorithm for updating the output weights of the gaussian network reflect the fact that no useful information can be gained about the quality of the current approximations when the tracking errors are less than a threshold determined by the bounds on the approximation errors, e.g. ϵ_f in (3.11). However, as originally proposed in [26], deadzone adaptation requires discontinuously starting and stopping the parametric adjustment mechanism according to the magnitude of the error signal. These discontinuities can be eliminated using the metric $s(t)$, as shown in [34], by introducing the continuous function s_Δ, defined as

$$s_\Delta(t) = s(t) - \Phi \;\, \mathrm{sat}(s(t)/\Phi). \tag{4.5}$$

The variable s_Δ has several properties useful in the design of adaptive laws: if $|s| < \Phi$ then $\dot{s}_\Delta = s_\Delta = 0$, while if $|s| > \Phi$, then $\dot{s}_\Delta = \dot{s}$, and $|s_\Delta| = |s| - \Phi$.

4.2 Controller Design

4.2.1 Unity Control Gain

In this section, the above ideas are applied to design a control law for a system in the canonical form:

$$x^{(n)}(t) + f(x(t), \dot{x}(t), \ldots, x^{(n-1)}(t)) = u(t). \tag{4.6}$$

where it is assumed that a prior upper bound $M_0(\mathbf{x})$ is known on the magnitude of f for points outside of the set A_d, i.e.

$$|f(\mathbf{x})| \leq M_0(\mathbf{x}) \quad \text{when} \quad \mathbf{x} \in A_d^c.$$

Use of a given control law $u(t)$ results in the error metric having a time derivative:

$$\dot{s}(t) = a_r(t) - f(\mathbf{x}) + u(t). \tag{4.7}$$

where $a_r(t) = \boldsymbol{\lambda}_v^T \tilde{\mathbf{x}}(t) - x_d^{(n)}$ with $\boldsymbol{\lambda}_v^T = [0,\ \lambda^{n-1},\ (n-1)\lambda^{n-2},\ \ldots,\ \lambda]$, and $x_d^{(n)}$ is the nth derivative of the desired trajectory. Having chosen a set A containing A_d as outlined above, let $f_A(\mathbf{x})$ be a radial gaussian approximation to $f(\mathbf{x})$ designed such that $|f - f_A| \leq \epsilon_f$ uniformly on the set A. In terms of this approximation, (4.7) can be rewritten as:

$$\dot{s}(t) = a_r(t) - f_A(\mathbf{x}) + d(\mathbf{x}) + u(t),$$

where $d(\mathbf{x}) = f_A(\mathbf{x}) - f(\mathbf{x})$ satisfies $|d| \leq \epsilon_f$. This expression, and the considerations of the previous section, suggests use of the control law:

$$u(t) = -k_d s(t) - a_r(t) + (1 - m(t))\,\hat{f}_A(\mathbf{x}(t)) + m(t)\,u_{sl}(t) \tag{4.8}$$

where k_d is a constant feedback gain. The sliding component of this control law is given by (4.4) with state dependent gains $k_{sl}(\mathbf{x}(t)) = M_0(\mathbf{x}(t)) + \epsilon_f$, and the adaptive component $\hat{f}_A(\mathbf{x}(t))$ is realized as the gaussian network described by equation (4.2) above, whose tuned output implements the approximation f_A. The modulation function which blends the two controller modes is specified by equation (4.1).

If the parameters in $\hat{f}_A(\mathbf{x}(t))$ are updated as:

$$\dot{\hat{c}}_I(t) = -\,k_a\,(1 - m(t))\,s_\Delta\,g_\sigma(\mathbf{x}(t), \boldsymbol{\xi}_I), \tag{4.9}$$

where k_a is a positive constant determining the adaptation rate, and if Φ is chosen so that $\Phi \geq \epsilon_f / k_d$ in both the sliding controller and the calculation (4.5) of s_Δ, then all states in the adaptive system will remain bounded, and moreover the tracking errors will asymptotically converge to a neighborhood of zero given by:

$$|\tilde{x}^{(i)}(t)| \leq 2^i\,\lambda^{i-n+1}\,\Phi, \qquad i = 0,\ \ldots,\ n-1. \tag{4.10}$$

To prove this assertion consider, similarly to [34], the Lyapunov function candidate:

$$V = \frac{1}{2}({s_\Delta}^2 + \frac{1}{k_a}\sum_{I \in \mathcal{I}_o} \tilde{c}_I^2) \tag{4.11}$$

where $\tilde{c}_I(t) = \hat{c}_I(t) - c_I$. By definition $\dot{s}_\Delta = 0$ when $|s| \leq \Phi$; this fact and equation (4.9) show that $\dot{V}(t) = 0$ when $|s| \leq \Phi$. Computing the time derivative during times when $|s| > \Phi$, using the fact that $s_\Delta\ \mathrm{sat}(s/\Phi) = |s_\Delta|$ in these regions, produces

$$\begin{aligned}
\dot{V} &= -k_d\,s\,s_\Delta - s_\Delta f + s_\Delta(1-m)\hat{f}_A - |s_\Delta|\,m\,k_{sl} + k_a^{-1}\sum_{I \in \mathcal{I}_o} \tilde{c}_I \dot{\hat{c}}_I \\
&= -(k_d\,{s_\Delta}^2 + |s_\Delta|\,k_d\,\Phi) - s_\Delta f + s_\Delta(1-m)\,\hat{f}_A - |s_\Delta|\,m\,k_{sl} + k_a^{-1}\sum_{I \in \mathcal{I}_o} \tilde{c}_I \dot{\hat{c}}_I
\end{aligned}$$

There are now two possibilities which must be considered separately.

(i.) $\mathbf{x} \in A^c$:

In this region $m = 1$, so that the output of the adaptive controller $(1-m)\hat{f}_A$ is identically zero, and from (4.9) the time derivatives of the parameter estimates similarly vanish, hence $\dot{V}$ becomes:

$$\dot{V} = -(k_d {s_\Delta}^2 + |s_\Delta|\, k_d\, \Phi) - s_\Delta f - |s_\Delta|\, k_{sl}$$

so that

$$\dot{V} \le -k_d {s_\Delta}^2 + |s_\Delta|\,(|f| - k_{sl}).$$

Since the sliding controller gains have been chosen so that $k_{sl} \ge |f|$ for all $\mathbf{x} \in A^c$, one obtains

$$\dot{V} \le -k_d {s_\Delta}^2 \quad \text{when} \quad \mathbf{x} \in A^c.$$

(ii.) $\mathbf{x} \in A$:

In this region $0 \le m \le 1$, and by rewriting f as

$$f = (1-m)\, f_A + m\, f_A - d$$

the equation for $\dot{V}$ can be rearranged to produce

$$\dot{V} = -k_d\, {s_\Delta}^2 - |s_\Delta|\, k_d\, \Phi - s_\Delta m f_A + s_\Delta d + s_\Delta (1-m) \sum_{I \in \mathcal{I}_o} \tilde{c}_I\, g_\sigma(\mathbf{x}, \boldsymbol{\xi}_I) + k_a^{-1} \sum_{I \in \mathcal{I}_o} \tilde{c}_I \dot{\hat{c}}_I$$

so that using (4.9) and the fact that $|f_A| \le |f| + |d|$,

$$\dot{V} \le -k_d {s_\Delta}^2 + |s_\Delta|(|d| - k_d \Phi) + |s_\Delta|\, m\, (|f| + |d| - k_{sl}). \tag{4.12}$$

Since $|d| \le \epsilon_f$ by construction when $\mathbf{x} \in A$, and Φ has been chosen so that $\Phi \ge \epsilon_f / k_d$, the first term in parentheses is less than or equal to zero. Similarly, the choice of sliding controller gains shows that, with this bound on $|d|$, $k_{sl} \ge |f| + |d|$, so the second term in parentheses is also nonpositive. Thus,

$$\dot{V} \le -k_d {s_\Delta}^2 \quad \text{when} \quad \mathbf{x} \in A.$$

In summary, the above considerations prove that $\dot{V} \le -k_d\, {s_\Delta}^2$ for all $t \ge 0$, and hence from (4.11) if s_Δ and all the $\tilde{c}_I$ are bounded at time $t = 0$, they remain bounded for all $t \ge 0$. To complete the proof and establish asymptotic convergence of the tracking error, it is necessary to show that $s_\Delta \to 0$ as $t \to \infty$. While the inequalities developed above do not permit immediate application of Barbalat's lemma to V (the usual method of establishing this limit [35]) by noting that the above considerations show that the inequality $\dot{V} + k_d\, {s_\Delta}^2 \le 0$ holds for all $t \ge 0$, one can create a nonnegative function, V_1, to which Barbalat's lemma can be applied:

$$V_1(t) = V(t) - \int_0^t (\dot{V}(\tau) + k_d\, {s_\Delta}^2(\tau))d\tau \quad \text{with} \quad \dot{V}_1(t) = -k_d\, {s_\Delta}^2(t). \tag{4.13}$$

Since use of (4.11) has already established that all signals in the system are bounded, the function V_1 is used only to conclude asymptotic convergence of the tracking error, as is now demonstrated.

By definition, $\dot{s}_\Delta$ is either 0 or $\dot{s}$, where $\dot{s}$ is given by (4.7) and (4.8). Since $s(t)$ is uniformly bounded in t, inequality (4.3) implies that, if $\tilde{\mathbf{x}}(0)$ is bounded, then $\tilde{\mathbf{x}}(t)$ is also bounded for all t, thus since $\mathbf{x}_d(t)$ is bounded by design, $\mathbf{x}(t)$ is as well. By assumption, $f(\mathbf{x})$ can be bounded as a function of $\mathbf{x}$, and thus $f(\mathbf{x}(t))$ is bounded since $\mathbf{x}(t)$ is; similarly the modulation $m(\mathbf{x})$ and sliding controller gains are both certainly bounded. Since the parameter error is bounded, and the tuned parameter values c_I are finite, the parameter estimates $\hat{c}(t)$ are bounded for all $t \geq 0$; this fact, and the fact that the gaussian basis functions are bounded for all values of $\mathbf{x}$ shows that the output of the adaptive controller is bounded. Finally, the reference input, $x_d^{(n)}$, is also bounded by design, so the term $a_r(t)$ is also bounded.

Every term in (4.7) is thus bounded, hence $\dot{s}$ is bounded, which proves $\dot{V}_1$ to be a uniformly continuous function of time. Since V_1 is bounded below by 0, and $\dot{V}_1 \leq 0$ for all t, use of Barbalat's lemma proves that $\dot{V}_1(t) \to 0$ and hence from(4.13) that $s_\Delta(t) \to 0$ as $t \to \infty$. This means that the inequality $|s(t)| \leq \Phi$ holds asymptotically, and the asymptotic bounds (4.10) follow using (4.3).

4.3 Non-Unity Control Gains

The previous design procedure can be extended to plants with non-unity control gains, of the form:

$$x^{(n)}(t) + f(\mathbf{x}(t)) = b(\mathbf{x}_T(t))\, u(t). \tag{4.14}$$

where $\mathbf{x}_T = [x, \dot{x}, \ldots, x^{(n-2)}]^T$, and it is assumed that the control gain is bounded away from zero and of known sign for all $\mathbf{x}$. Without loss of generality, this sign can be taken as positive, so that

$$0 < M_1(\mathbf{x}_T) \leq b^{-1}(\mathbf{x}_T) \leq M_2(\mathbf{x}_T). \tag{4.15}$$

It is further assumed that a prior bound is known for the norm of the gradient of this inverse, so that

$$|\dot{b}^{-1}(\mathbf{x}_T)| = |\nabla b^{-1}(\mathbf{x})\, \dot{\mathbf{x}}_T| \leq M_3(\mathbf{x})\|\mathbf{x}\|. \tag{4.16}$$

As above, it is also assumed that an upper bound is known for f; to facilitate the development below, it is most convenient to combine this knowledge with the bound on b^{-1} to produce

$$|b^{-1}(\mathbf{x})f(\mathbf{x})| \leq M_0(\mathbf{x}).$$

As before, define $s(t) = \boldsymbol{\lambda}^T\tilde{\mathbf{x}}(t)$, so that

$$b^{-1}\dot{s} = b^{-1}a_r - b^{-1}f + u, \tag{4.17}$$

with a_r as in Section 4.2.1. Define the unknown nonlinear function $h = b^{-1}f$, and let h_A and b_A^{-1} be a radial gaussian network approximation to the functions h and b^{-1} respectively with the properties, valid on the chosen set A:

$$|h(\mathbf{x}) - h_A(\mathbf{x})| \leq \epsilon_h \quad \text{and} \quad |b^{-1}(\mathbf{x}_T) - b_A^{-1}(\mathbf{x})| \leq \epsilon_b, \tag{4.18}$$

where ϵ_h and ϵ_b are as small as desired. As mentioned at the end of Section 3.3, there is no conceptual problem approximating $b^{-1}(\mathbf{x}_T)$ by a function of the entire state vector instead of just $\mathbf{x}_T$; this fact will also allow a considerable simplification in the design of the adaptive subsystem, as will be discussed below.

With these definitions, (4.17) can be rewritten as

$$b^{-1}\dot{s} = b_A^{-1} a_r - h_A + d + u, \tag{4.19}$$

where the disturbance term d is here given by

$$d(\mathbf{x}, \mathbf{x}_d) = (b^{-1}(\mathbf{x}_T) - b_A^{-1}(\mathbf{x}))\, a_r - (h(\mathbf{x}) - h_A(\mathbf{x})),$$

and thus satisfies the inequality

$$|d(\mathbf{x}, \mathbf{x}_d)| \leq \epsilon_h + \epsilon_b\, |a_r|$$

when $\mathbf{x} \in A$. Equation (4.19) suggests a control law of the form:

$$\begin{aligned} u(t) \;\; = \;\; & -k_d\, s(t) - \frac{1}{2} M_3(\mathbf{x}(t))\, \|\mathbf{x}(t)\|\, s_\Delta(t) + m(t)\, u_{sl}(t) \\ & - (1 - m(t)) \left[\widehat{b_A^{-1}}(\mathbf{x}(t))\, a_r(t) - \hat{h}_A(\mathbf{x}(t)) \right], \end{aligned} \tag{4.20}$$

for some positive constant k_d. The sliding term, $u_{sl}(t)$, is again given by (4.4) with state-dependent gain

$$k_{sl}(\mathbf{x}, \mathbf{x}_d) = (M_0(\mathbf{x}) + \epsilon_h) + (M_2(\mathbf{x}) + \epsilon_b)\, |a_r|.$$

Note particularly that the adaptive component of this control law estimates b^{-1} itself, as opposed to estimating b and inverting the estimate.

The adaptive components of this control law, $\hat{h}_A$ and $\widehat{b_A^{-1}}$, can each be realized as the output of a network of radial gaussian nodes similar to the one described for the unity gain system of Section 4.2.1. Each network is designed considering assumed upper bounds on the spectral supports of h and b^{-1} respectively, as well as the set of desired trajectories the system is required to track, and approximation accuracy desired. It is possible to simplify this design somewhat by employing a single network, with the n state variables as inputs, and the two adaptive approximations as the outputs. The number of gaussian nodes and the corresponding input weights for this network are chosen to correspond to a sampling grid with the smaller of the two mesh sizes which would be used in the separate networks. This network then has two sets of output weights for each node, $\hat{c}_I(t)$ and $\hat{d}_I(t)$, creating two outputs which are the approximations $\hat{h}_A(\mathbf{x}(t))$ and $\widehat{b_A^{-1}}(\mathbf{x}(t))$respectively:

$$\begin{aligned} \hat{h}_A(\mathbf{x}(t)) \;\; &= \;\; \sum_{I \in \mathcal{I}_o} \hat{c}_I(t)\, g_\sigma(\mathbf{x}(t), \boldsymbol{\xi}_I) \\ \widehat{b_A^{-1}}(\mathbf{x}(t)) \;\; &= \;\; \sum_{I \in \mathcal{I}_o} \hat{d}_I(t)\, g_\sigma(\mathbf{x}(t), \boldsymbol{\xi}_I) \end{aligned} \tag{4.21}$$

As before, the input weights encoding the sampling mesh are fixed in this architecture, and the output weights $\hat{c}_I(t)$ and $\hat{d}_I(t)$ are adjusted to match their tuned values, c_I and d_I, in the expansions h_A and b_A^{-1} respectively.

Choose the adaptation law for these parameters to be:

$$\begin{aligned}
\dot{\hat{c}}_I(t) &= -k_{a1}\,(1-m(t))\,s_\Delta(t)\,g_\sigma(\mathbf{x}(t),\boldsymbol{\xi}_I) \\
\dot{\hat{d}}_I(t) &= k_{a2}\,(1-m(t))\,s_\Delta(t)\,a_r(t)\,g_\sigma(\mathbf{x}(t),\boldsymbol{\xi}_I)
\end{aligned} \tag{4.22}$$

where k_{a1} and k_{a2} are strictly positive constants determining the adaptation rate. Define

$$\epsilon_r = \epsilon_h + \epsilon_b(2\|\boldsymbol{\lambda}\|r_a + |x_d^{(n)}|_{max}),$$

with r_a being the radius of the largest n-ball containing the set A, and $|x_d^{(n)}|_{max}$ an upper bound on the magnitude of the nth derivative of the reference trajectory. If Φ is chosen such that $\Phi \geq \epsilon_r/k_d$ in both the sliding controller and the calculation (4.5) of s_Δ, then all states in the adaptive system will remain bounded, and moreover the tracking errors will asymptotically converge to a neighborhood of zero given by:

$$|\tilde{x}^{(i)}(t)| \leq 2^i\ \lambda^{i-n+1}\ \Phi \qquad i = 0,\ \ldots,\ n-1. \tag{4.23}$$

Note that this inequality, together with the definitions of Φ and ϵ_r, implies that a good approximation to b^{-1} is much more critical to reduction of the asymptotic tracking errors than is approximation to h; this consideration should be reflected in the network design.

In order to prove convergence, consider the positive function:

$$V = \frac{1}{2}(b^{-1}(\mathbf{x}_T){s_\Delta}^2 + \frac{1}{k_{a1}}\sum_{I\in\mathcal{I}_o}\tilde{c}_I^2 + \frac{1}{k_{a2}}\sum_{I\in\mathcal{I}_o}\tilde{d}_I^2), \tag{4.24}$$

where $\tilde{c}_I(t) = \hat{c}_I(t) - c_I$ and $\tilde{d}_I(t) = \hat{d}_I(t) - d_I$. By definition $\dot{s}_\Delta = 0$ when $|s| \leq \Phi$; this fact and equation (4.22) show that $\dot{V} = 0$ during these times. Computing the time derivative along state trajectories of the adaptive system, during times when $|s| > \Phi$ with use of (4.17), and (4.20)

$$\begin{aligned}
\dot{V} &= \frac{1}{2}(\dot{b}^{-1} - M_3\|\mathbf{x}\| - 2k_d){s_\Delta}^2 - |s_\Delta|\,k_d\,\Phi - |s_\Delta|\,m\,k_{sl} + s_\Delta(b^{-1}a_r - h) \\
&\quad - s_\Delta(1-m)(\widehat{b_A^{-1}}a_r - \hat{h}_A) + k_{a1}^{-1}\sum_{I\in\mathcal{I}_o}\tilde{c}_I\dot{\hat{c}}_I + k_{a2}^{-1}\sum_{I\in\mathcal{I}_o}\tilde{d}_I\dot{\hat{d}}_I,
\end{aligned} \tag{4.25}$$

where $\dot{b}^{-1}$ is the time derivative of b^{-1}, bounded as in (4.16).

As before, there are two separate cases to consider:

(i.) $\mathbf{x} \in A^c$:

In this region $m = 1$ and, as in the previous section, the output of the adaptive controller and the time derivatives of the parameter estimates identically vanish. Under these conditions, (4.25) can be rewritten to obtain the bound:

$$\dot{V} \leq -k_d{s_\Delta}^2 + |s_\Delta|(|b^{-1}|\,|a_r| + |h| - k_{sl}).$$

The above choice of k_{sl} ensures that the second term on the right is nonpositive, hence

$$\dot{V} \leq -k_d {s_\Delta}^2 \quad \text{when} \quad \mathbf{x} \in A^c$$

(ii.) $\mathbf{x} \in A$:

In this region $0 \leq m \leq 1$ and, rewriting b^{-1} and h in terms of b_A^{-1}, h_A, and d, similarly to the technique used in the previous section, (4.25) can be rearranged to produce

$$\begin{aligned}\dot{V} &= \frac{1}{2}(\dot{b}^{-1} - M_3 - 2k_d){s_\Delta}^2 - |s_\Delta| k_d \Phi + s_\Delta d + s_\Delta m(b_A^{-1} a_r - h_A) - m|s_\Delta| k_{sl} \\ &\quad + s_\Delta(1-m)\left(\sum_{I\in\mathcal{I}_o} \tilde{c}_I\, g_\sigma(\mathbf{x},\boldsymbol{\xi}_I) - a_r \sum_{I\in\mathcal{I}_o} \tilde{d}_I\, g_\sigma(\mathbf{x},\boldsymbol{\xi}_I)\right) + k_{a1}^{-1}\sum_{I\in\mathcal{I}_o} \tilde{c}_I \dot{\hat{c}}_I + k_{a1}^{-1}\sum_{I\in\mathcal{I}_o} \tilde{d}_I \dot{\hat{d}}_I\end{aligned}$$

so that, using (4.22),

$$\dot{V} \leq -k_d {s_\Delta}^2 + |s_\Delta|(|d| - k_d\Phi) + m|s_\Delta|(|b_A^{-1}||a_r| + |h_A| - k_{sl}).$$

Now, when $\mathbf{x} \in A$, then $|a_r| \leq \|\boldsymbol{\lambda}\|(2r_a) + |x_d^{(n)}|_{max}$, and the uniform bounds (4.18) hold in this region, so that $|d| \leq \epsilon_r$. Since Φ has been chosen such that $\Phi \geq \epsilon_r/k_d$, the first term in parentheses is less than or equal to zero. To evaluate the second parenthesized term, note from the definition of d that

$$|b_A^{-1}||a_r| + |h_A| \leq |b^{-1}||a_r| + |h| + |d|,$$

hence, substituting explicitly for the sliding controller gains and using the bounds on $|d|$ produces

$$\dot{V} \leq -k_d {s_\Delta}^2 + |s_\Delta|\, m \left[(|b^{-1}| + \epsilon_b)|a_r| + (|h| + \epsilon_h) - (M_2 + \epsilon_b)|a_r| - (M_0 + \epsilon_h)\right].$$

From the definitions of the terms M_i, it is clear that the bracketed term is indeed nonpositive, so that

$$\dot{V} \leq -k_d {s_\Delta}^2 \quad \text{when} \quad \mathbf{x} \in A.$$

Thus it has been established that the specified control and adaptation laws produce $\dot{V}(t) \leq -k_d\, {s_\Delta}^2$ for all $t \geq 0$. Hence, if the initial values of the states of the plant and of the parameter estimates are bounded, they remain bounded for all positive time. Finally, given the assumed constraints on b^{-1}, an argument exactly parallel to that of the unity gain case of Section 4.2.1 can be used to establish that $s_\Delta(t) \to 0$ as $t \to \infty$; use of (4.3) then provides the asymptotic bounds (4.23) on the individual tracking error states.

4.4 Remarks

Since the desired trajectories are contained completely within A_d, and since the tracking errors converge to a neighborhood of zero, the above proofs also show that eventually all the plant trajectories will converge to a set which is either within a small neighborhood of the set A_d, or else is completely contained inside it. In particular, the sliding subsystem will asymptotically be used less and less, and, since the negative feedback terms decay to zero as the tracking error does, the control input will eventually be dominated by the outputs of the neural network, regardless of the initial conditions on the plant or network parameters.

While the use of the proposed neural controllers allows the design of adaptive algorithms which can stably control plants with limited knowledge of the functions driving their dynamics, one still cannot ignore robust stability issues. The adaptive controller can compensate for dynamic effects attributable only to the states it "sees", i.e. those which are measured and which form the inputs to the gaussian networks. There is no way for the adaptive architecture described to compensate for dynamic effects due to (temporal) high frequency modes neglected in the above models. It is thus necessary to ensure that the control signal issued by the network is free from any (temporal) high frequency components which might excite these modes. Inspection of the adaptation laws suggests that the magnitude of the adaptation signals will be much larger for the $\hat{d}_I$ than for the $\hat{c}_I$. To prevent excessively rapid adaptation from adding high frequency components to the control signal, it is hence desirable to choose the adaptation gain k_{a2} to be much smaller than that of k_{a1}.

Choosing the same parameter to control both the adaptation deadzone and the sliding controller boundary layer allows one to guarantee convergence of the tracking errors to the same neighborhood of zero regardless of location of the state vector. However, if extreme tracking accuracy is required, and the network is designed to produce the required reconstruction accuracy, the resulting choice of Φ might produce a sliding controller with a boundary layer so thin it risks exciting the high frequency dynamic modes when this component is active [33]. This suggests that the value of Φ used, and hence the resulting tracking accuracy, must arise as a tradeoff between the frequency range of the unmodeled dynamics and trajectory following requirements.

The parameter Ψ, which defines the size of the transition region $A - A_d$ between adaptive and sliding operation, in a sense also represents the width of a boundary layer, serving to smooth the transition between the two controller modes. Since, with a zero width transition region, there is a possibility the controller might chatter along the boundary of the set A_d, the same comments which apply to Φ above also apply to Ψ: the transition region should be "thick" enough that there is no possibility that the mode switching might excite unmodeled dynamics.

The above control and adaptation laws are unique not only for the fact that the adaptive component is implemented by a neural network, but also for the manner in which a seamless transition between adaptive and sliding operation has been effected. It is important to note that this transition is *necessary* given the limitations on the approximating abilities of the adaptive controller; such a scheme may also prove useful in conventional adaptive applications as well.

Finally, note that the techniques displayed above can be used to extend the above ideas to much more general classes of systems, including multiple input and output plants. If perfect tracking can be accomplished through reconstruction of the values assumed by unknown nonlinear functions on a compact set, and a measure of the tracking error can be found which is related to the function reconstruction errors by a passive operator, then gaussian networks of the type described above can be used to adaptively compensate for the nonlinearities, and reduce tracking errors to a neighborhood of the zero.

5 Simulations

In this section, the above theoretical development is applied to two example systems. In Section 5.1, a controller is designed for the unity gain plant described by (4.6) where the unknown nonlinear function is spatially transformable with a strictly bandlimited spectrum. In Section 5.2, a more realistic example is considered using the nonunity gain plant described by (4.14) where the unknown control gain function $b(\mathbf{x}_T)$ is not (classically) transformable, and the truncation, b_F, of this function outside the set A is not a bandlimited function.

To gauge the effectiveness of the adaptive component of the control law, it is useful to compare the tracking performance of the closed loop system both with and without the output of the adaptive gaussian network. Since use of the sliding controller is sufficient to ensure reduction of the tracking errors outside of a specified subset of the state space, even without the adaptive contribution, these simulations can be conducted without fear of instability.

In the simulations which follow, a two dimensional plant is required to track a set of desired trajectories which are specified as the output of a third order, critically damped system with a bandwidth of 10 rad/sec driven by a unity amplitude, 0.4 Hz. square wave. Figure 4 shows a phase space portrait of the desired position and velocity. Based upon this portrait, the set A_d was chosen to be the unit ball with respect to the weighted infinity norm

$$\|\mathbf{x}\|_{\infty,w} = \max\left(|x|, \frac{|\dot{x}|}{3.0}\right),$$

centered at $\mathbf{x}_0 = [0.5, 0]^T$. This corresponds to the rectangular subset of the state space, $A_d = [-0.5, 1.5]$ x $[-3.0, 3.0]$. A thin transition region between adaptive and sliding operation was chosen for these simulations, corresponding to a value of $\Psi = 0.025$ in (4.1), so that

$$A = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}_0\|_{p,w} \leq 1.025\}.$$

5.1 Unity Control Gain

In this section the ideas of Section 4 are illustrated on a two dimensional plant with unity control gain; i.e. (4.6) with $n = 2$.

The plant nonlinearity f is assumed to have a spectral support of $\beta = 2$, and use of (3.14) suggests gaussian nodes with variance $\sigma^2 = 4\pi$ whose input weights encode a sampling grid with mesh size $\Delta = (32\pi)^{-\frac{1}{2}} \approx 0.1$. A value of $\rho = 0.5$ is used in the truncation of expansion (3.9); for simplicity, Γ is taken as the smallest rectangle which contains all the points in this truncated sampling grid, $\Gamma = [-1.0, 2.0]$ x $[-3.5, 3.5]$. This set actually encloses a few mesh points of distance further than 0.5 to points in A, but the value 0.5 will be used in developing conservative bounds on the truncation error. (Figure 3 may again be helpful). This definition of Γ results in a network with 2201 gaussian nodes whose output weights are adjusted according to (4.9) with $k_a = 500$.

Using the truncation radius $\rho = 0.5$, the omitted nodes closest to the set A contribute terms only of order 10^{-6} to the reconstruction of the values assumed by f on this set. A conservative estimate, taking into account the number of closest omitted nodes and an assumed uniform upper bound on f of 20, suggests that $\epsilon_f \leq 10^{-3}$ on A if this value of ρ is used. Using this bound and a value $k_d = 5$ for the feedback gain, results in a deadzone $\Phi = 2$ x 10^{-4} for use in the adaptation law and sliding controller.

The complete control law is given by equations (4.2), (4.4), and (4.8). Since the nonlinearity f is assumed to be uniformly upper bounded on A^c by 20, the gain $k_{sl} = 20$ is used in the sliding controller. The bandwidth of the sliding surface is taken as $\lambda = 10$ in the definition of $s(t)$.

The true plant is modeled by (2.1) with $n = 2$ and:

$$f(x, \dot{x}) = -4 \left(\frac{\sin(4\pi x)}{\pi x} \right) \left(\frac{\sin(\pi \dot{x})}{\pi \dot{x}} \right)^2$$

which is indeed spatially bandlimited, its spectrum resembling a "roof" of length 2 and width 1. The plant and controller dynamics were simulated in C on DECStation 3100 RISC workstations, using a fourth order, fixed stepsize Runge-Kutta algorithm with $\Delta t = 10^{-4}$ sec. All initial conditions were taken as 0, reflecting the fact that *no* prior knowledge is assumed about the plant save for its order and upper bounds on both the absolute magnitude of the function f and on the spectral radius of its spatial Fourier transform.

Two simulations were conducted, one in which the complete control law (4.8), including the adaptive gaussian network, was employed, and one in which only the PD and sliding components were used. Figure 5 compares the position tracking error, $\tilde{x}(t)$, and the total control signal issued to the plant during the first 5 seconds for both of these simulations. Use of the adaptive controller clearly results in superior tracking performance; worst case tracking error is reduced by a factor of ten compared with that obtained using PD and sliding components alone during this time period, and this ratio continually improves as the adaptive controller learns more about the plant dynamics. Figure 6 shows the tracking error using the adaptive controller during a fifty second period which does indeed demonstrate the predicted asymptotic convergence to the deadzone.

5.2 Nonunity Control Gain

Again, a two dimensional plant is used, with the same f nonlinearity used in the previous section, but with b in equation (4.14) given by:

$$b(x) = 2 + \sin(3\pi(x - 0.5))$$

Note that in this case, the constant function 2 is not Fourier transformable in the classical sense (although it is in the distributional sense), but it can easily be approximated by a function which is 2 on A and decays smoothly to 0 on A^c. Furthermore, b^{-1} is uniformly lower bounded by $M_1 = 1/3$ and upper bounded by $M_2 = 1$, and $\dot{b}^{-1}$ can be uniformly bounded using $M_3 = 3\pi$, so that the control gains satisfy the constraints discussed in Section 4.3.

The desired trajectories described in Section 5.1 are used, and the same network designed in that section is used to generate both $\hat{h}_A$ and $\widehat{b_A^{-1}}$ by adding a new set of output weights, $\hat{d}_I$, which are adjusted as specified in (4.22) with $k_{a2} = 1.0$. The maximum desired acceleration is $|x_d^{(n)}|_{max} = 25$ for these trajectories, so that $|a_r| \leq 100$ when $\mathbf{x} \in A$. Under the assumption that truncation errors dominate the uniform bounds on the approximation errors, ϵ_b is considered to be of the same order as the ϵ_f computed in the previous section, and hence $\epsilon_r \leq 0.1$ which suggests a choice of $\Phi = 0.02$ as the adaptation deadzone and sliding controller boundary layer.

The complete control law is given by equations (4.4), (4.20), and (4.21). Using the above bounds on b and f, and again the value $k_d = 5$, the sliding controller gains are taken as $k_{sl} = 20 + |a_r|$.

The same simulation parameters and initial conditions as above are chosen. Once again, two simulations were conducted, one with and the other without the contribution of the adaptive network. Figure 7 shows both the position tracking error and control signals input to the plant during the first 5 seconds for both these simulations. Once again, it is evident that use of the adaptive controller provides superior performance; initial tracking errors are about three times better than those of the PD and sliding controller alone, with this ratio improving dramatically after the first few seconds. Notice the larger control inputs in places for the simulation without the adaptive component; these occur as the plant state drifts outside the set A prompting full use of the sliding component of the controller.

Figure 8 shows the tracking error for the adaptive controller over the first 50 seconds of operation; again, it is evident that the errors have converged into the deadzone.

6 Concluding Remarks

Under the assumption that the nonlinear functions driving the plant dynamics are sufficiently smooth, we have shown that a network of neurons possessing radial gaussian input-output characteristics can be designed which is capable of uniformly approximating these functions on a compact set. By placing the elements of a gaussian network in one-to-one correspondence with elements of the representations provided by sampling theory, we have been able to assign a precise interpretation to each of the network parameters, and to demonstrate how these parameters are related to the properties of the function the network is required to approximate.

By considering the feed-in weights to each node of this network as representing a fixed sampling mesh on $\mathcal{R}^n$, and adjusting only the feed-out weights during operation, we have demonstrated that a suitable choice of adaptation law, coupled with a constant gain and a sliding controller, results in a globally stable closed loop system, and tracking errors which converge to a neighborhood of zero. A unique feature of this control law is its ability to smoothly transition from adaptive to nonadaptive modes of operation, becoming a sliding controller in the regions of the state space where the network has poor approximating capability, a purely adaptive controller where the network approximating power is good, or a stabilizing blend of the two modes in an intermediate transition region.

The method we have described can be extended to multi-input plants. In fact, the structure of the arguments developed above suggest that the large body of existing adaptive control results for parameterized linear and nonlinear dynamics can be exploited to design stable adaptation laws when the parameterization is replaced with an appropriately designed gaussian network.

The above developments by no means suggest that more conventional adaptive architectures should be discarded. Every piece of information available to the designer should be employed in constructing a control system, and in most cases, good estimates of basis functions driving the plant dynamics may be available. A neural controller can then be employed in parallel with these basis functions, to account for uncertainties which cannot be so easily parameterized. A good example of this idea can be found in robotic applications, for which the basis functions governing the motion of the joints as a function of the applied torques are known with great accuracy, and for which stable adaptive schemes exploiting this knowledge are well known. What is much more difficult to accurately characterize in these applications is the impact of Coulomb friction forces, which are most noticeable only at very low velocities. A neural network of the type we have proposed could be designed to estimate these friction forces in a very small neighborhood of the zero joint velocity region and operate alongside a conventional adaptive controller.

The large number of nonlinear function computations and adjustable parameters in the method we have described suggests that this algorithm will be practical *only* if implemented in inexpensive, plentiful hardware. The simplest of our examples required 4402 fixed and 2201 adjustable weights, and 2201 computations of the radial gaussian; all this for a second order dynamic system! The number of computations required would clearly overwhelm a serial microprocessor attempting to implement the equations in real time. Given hardware designed to exploit the parallel, analog nature of the required computations, however, this problem would not exist. The algorithms we have described above are directly transferable to analog electronics, creating the very real possibility of neural motion control chips. Noting that mass production has already begun for neural chips whose nodes exhibit logistic input-output characteristics, it should not be difficult to produce similar devices using instead the radial gaussian. Actually, implemented in parallel hardware, a neural adaptive controller may prove the superior choice even when the basis functions are known.

Although our intention in this paper has been to derive stable adaptive controllers, intuitively the structure of the above control law is compatible with the observed movements of biological organisms. Large magnitude, gross control movements are employed when outside of accustomed regions of operation, for example when a person momentarily loses balance or when swinging an arm rapidly toward a pad of paper prior to writing a letter, while the finer, tuned movements occur in a much smaller, restricted region of the degrees of freedom which describe the motion. It is thus possible that living organisms also exhibit a kind of blended motion control. In fact, there may be several different regions in which high precision, learned motion patterns are exhibited; that is, the regions we have described above as A need not be topologically connected in the state space describing the motion.

Finally, we note that to a close approximation, the sigmoid function used in back propagation networks is the integral of a gaussian. This suggests that the spatial frequency domain ideas suggested above could also be used in the design of three-layered back propagation networks.

References

[1] Adams, M. and Guillemin, V., *Measure Theory and Probability*, Wadsworth and Brooks/Cole, Monterey, CA, 1986.

[2] Aiserman, M. A., Braverman, E. M., and Rozonoer, L. I., "Potential functions technique and extrapolation in learning systems theory", *Proc. 3rd IFAC Congress*, 14G.1-14G.12, London, 1966.

[3] Arimoto, S., Kawamura, S., and Miyazaki, F., "Bettering operation of robots by learning", *Jour. Rob. Sys.*, 1(2), 123-140, 1984.

[4] Arimoto, S., Kawamura, S., Miyazaki, F., and Tamaki, S., "Learning control theory for dynamical systems", *Proc. 24th CDC*, 1375-1380, Ft. Lauderdale, FL, 1985.

[5] Arsac, J., *Fourier Transforms and the Theory of Distributions*, Prentice-Hall, Englewood Cliffs, N.J., 1966.

[6] Atkeson, C. G., "Memory-based approaches to approximating continuous functions", *Proc. 6th Yale Workshop on Adaptive and Learning Systems*, 212-217, Yale, New Haven, CT, 1990.

[7] Bracewell, R., *The Fourier Transform and Its Applications*, McGraw-Hill, New York, NY, 1965.

[8] Broomhead, D. S., and Lowe, D., "Multivariable functional interpolation and adaptive networks", *Complex Systems*, 2, 321-355, 1988.

[9] Funahashi, K., "On the approximate realization of continuous mappings by neural networks", *Neural Networks*, 2, 183-192, 1989.

[10] Girosi, F., and Poggio, T., "Networks and the best approximation property", *Artificial Intelligence Lab. Memo*, No. 1164, MIT, Cambridge, MA, October 1989.

[11] Hornik, K., Stinchcombe, M., and White, H., "Multilayer feedforward networks are universal approximators", *Neural Networks*, 2, 359-366, 1989.

[12] Horn, B.K.P., "Parallel networks for machine vision," *Artificial Intelligence Lab. Memo*, No. 1071, MIT, Cambridge, MA, December 1988.

[13] Horowitz, R., Messner, W., Moore, J., Kao, W., "Learning motion controllers for robot manipulators", *Proc. 6th Yale Workshop on Adaptive and Learning Systems*, 218-224, Yale, New Haven, CT, 1990.

[14] Horowitz, R., Messner, W., Moore, J., Kao, W., "Convergence properties of learning controllers for robot manipulators", *1990 Japan-USA Symp. on Flex. Autom.*, 635-642, Kyoto, Japan, 1990.

[15] Kawamura, S., Miyazaki, F., and Arimoto, S., "Realization of robot motion based on a learning method", *IEEE Trans. Sys., Man, and Cyber.*, 18, 126-133, 1988.

[16] Kawato, M., Furukawa, K., and Suzuki, R., "A hierarchical neural-network model for control and learning of voluntary movement", *Biological Cybernetics*, 57, 169-185, 1987.

[17] Li, W., and Slotine, J. J. E., "Neural network control of unknown nonlinear systems," *Proc. 1989 ACC*, 1136-1141, 1989.

[18] Messner, W., Horowitz, R., Kao, W.-W., and Boals, M., "A New Adaptive Learning Rule", *IEEE Trans. Autom. Cont.*, 36,2, 188-197, 1991.

[19] Miller, W. T., An, E., and Glanz, F., "The design of CMAC neural networks for control", *Proc. 6th Yale Workshop on Adaptive and Learning Systems*, 218-224, Yale, New Haven, CT, 1990.

[20] Miller, W. T., Glanz, F. H., and Kraft, L.G., "Application of a general learning algorithm to the control of robotic manipulators", *Intl. Jour. Rob. Res.*, 6, 84-98, 1987.

[21] Minsky, M., and Papert, S., *Perceptrons*, MIT Press, Cambridge, MA, 1969.

[22] Narendra, K. S., and Annaswamy, A., *Stable Adaptive Systems*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[23] Narendra, K. S., and Parthasarathy, K., "Identification and control of dynamical systems using neural networks, Part II", *Proc. 6th Yale Workshop on Adaptive and Learning Systems*, 218-224, Yale, New Haven, CT, 1990.

[24] Narendra, K. S., and Parthasarathy, K., "Identification and control of dynamical systems using neural networks", *IEEE Trans. on Neural Networks*, 1, 1, 4-27, March 1990.

[25] Petersen, D. P., and Middleton, D., "Sampling and reconstruction of wave number limited functions in n-dimensional Euclidean spaces", *Information and Control*, 5, 279-323, 1962.

[26] Peterson, B. B., and Narendra, K. S., "Bounded Error Adaptive Control", *IEEE Trans. Autom. Cont.*, 27,6, 1161-1168, 1982.

[27] Poggio, T. and Torre, V., "Ill-posed problems and regularization analysis in early vision", *Artificial Intelligence Lab. Memo*, No. 773, MIT, Cambridge, MA, April 1984.

[28] Poggio, T., and Girosi, F., "A theory of networks for approximation and learning", *Artificial Intelligence Lab. Memo*, No. 1140, MIT, Cambridge, MA, July 1989.

[29] Poggio, T., and Girosi, F., "Extensions of a theory of networks for approximation and learning: dimensionality reduction and clustering", *Artificial Intelligence Lab. Memo*, No. 1167, MIT, Cambridge, MA, April 1990.

[30] Rumelhart, D. E., and McClelland, J. L., *Parallel Distributed Processing, Explorations in the Microstructure of Cognition*, vol. 1, MIT Press, Cambridge, MA, 1986.

[31] Sanner, R. M., and Akin, D. L., "Neuromorphic regulation of an underwater telerobot," *I.E.E.E Control Sys. Mag.*, April 1990.

[32] Sanner, R. M., and Slotine, J.-J. E., "Direct Adaptive Control with Gaussian Networks," *NSL Report*, No. 910303, March 1991; also appears in *Proc. 1991 Automatic Control Conference*, vol. 3, pp. 2153-2159, Boston MA, June 1991.

[33] Slotine, J.-J. E., "Sliding controller design for nonlinear systems", *Intl. Journal of Control*, 40, 421, 1984.

[34] Slotine, J.-J. E. and Coetsee, J.A., "Adaptive sliding controller synthesis for nonlinear systems", *Intl. Journal of Control*, 1986.

[35] Slotine, J.-J. E. and Li, W., *Applied Nonlinear Control*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[36] Spivak, Michel, *Calculus on Manifolds*, Addison-Wesley, Reading, Mass., 1965.

[37] Utkin, V. I., "Variable structure systems with sliding mode: a survey", *IEEE Trans. Auto. Contr.*, 22, 212, 1977.

[38] Zaanen, A. C., *Continuity, Integration, and Fourier Theory*, Springer-Verlag, Berlin, 1989.

[39] Zemanian, A. H., *Distribution Theory and Transform Analysis*, McGraw-Hill, New York, 1965.

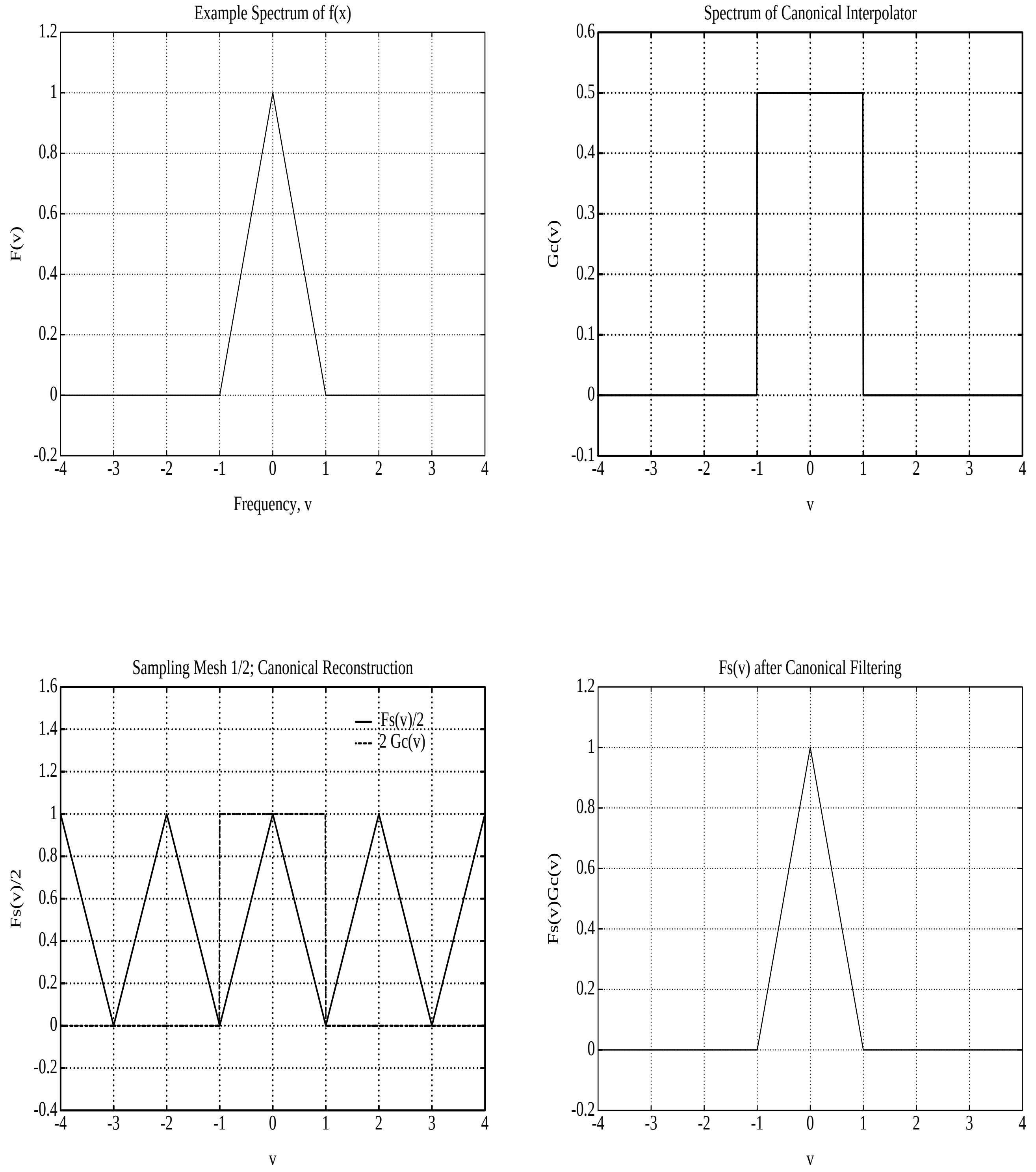


Figure 1: Illustration of canonical reconstruction in the frequency domain. Here, f is a function of one argument whose spectrum looks like the triangle pictured.

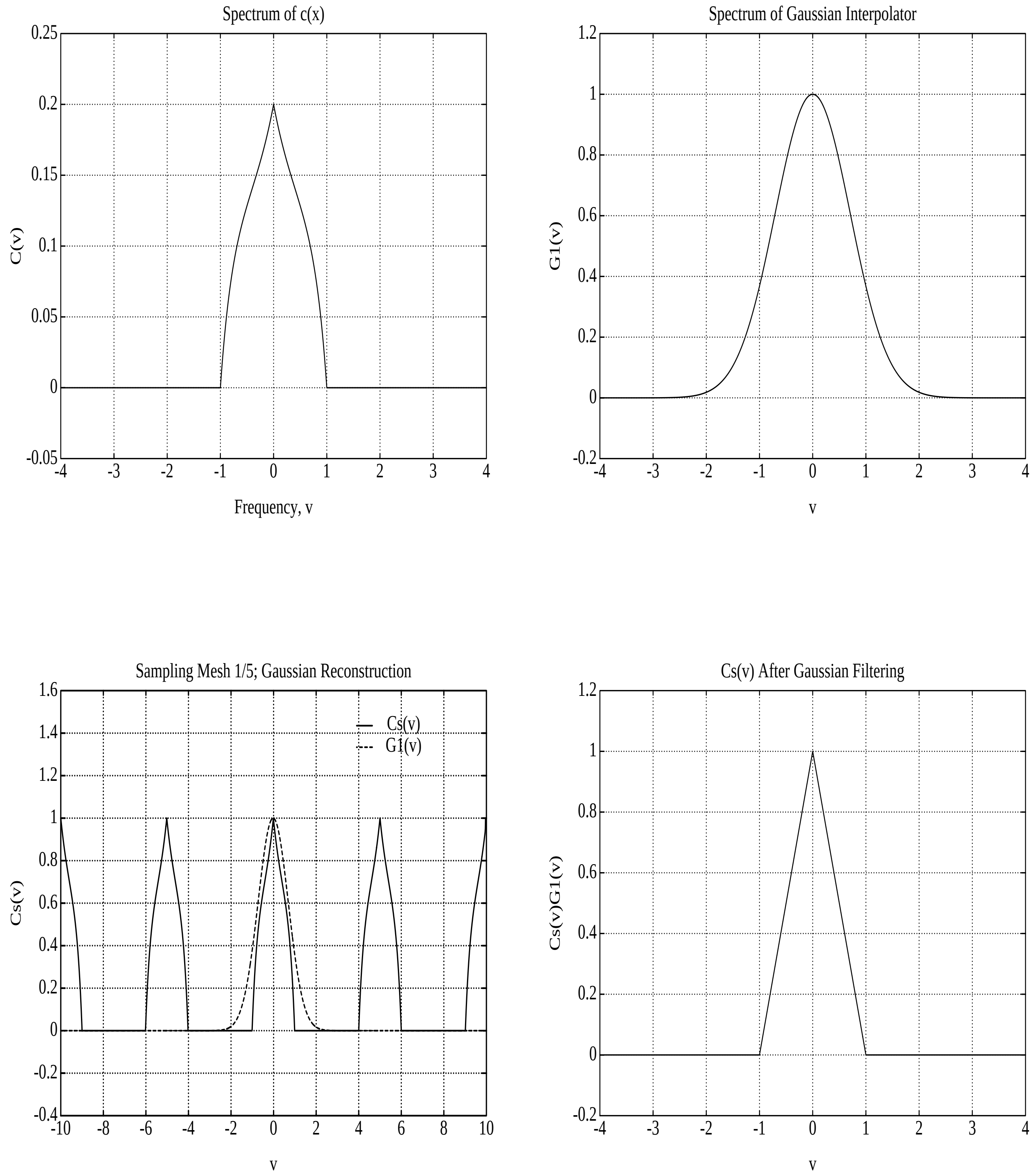


Figure 2: Illustration of gaussian reconstruction in the frequency domain. By sampling a slightly different function c, whose spectrum is at top left, then using a gaussian low pass filter, the triangular spectrum of the original function f is recovered. There are residual high frequency components since the gaussian does not vanish identically for high frequencies, but these are invisible at the scale pictured.

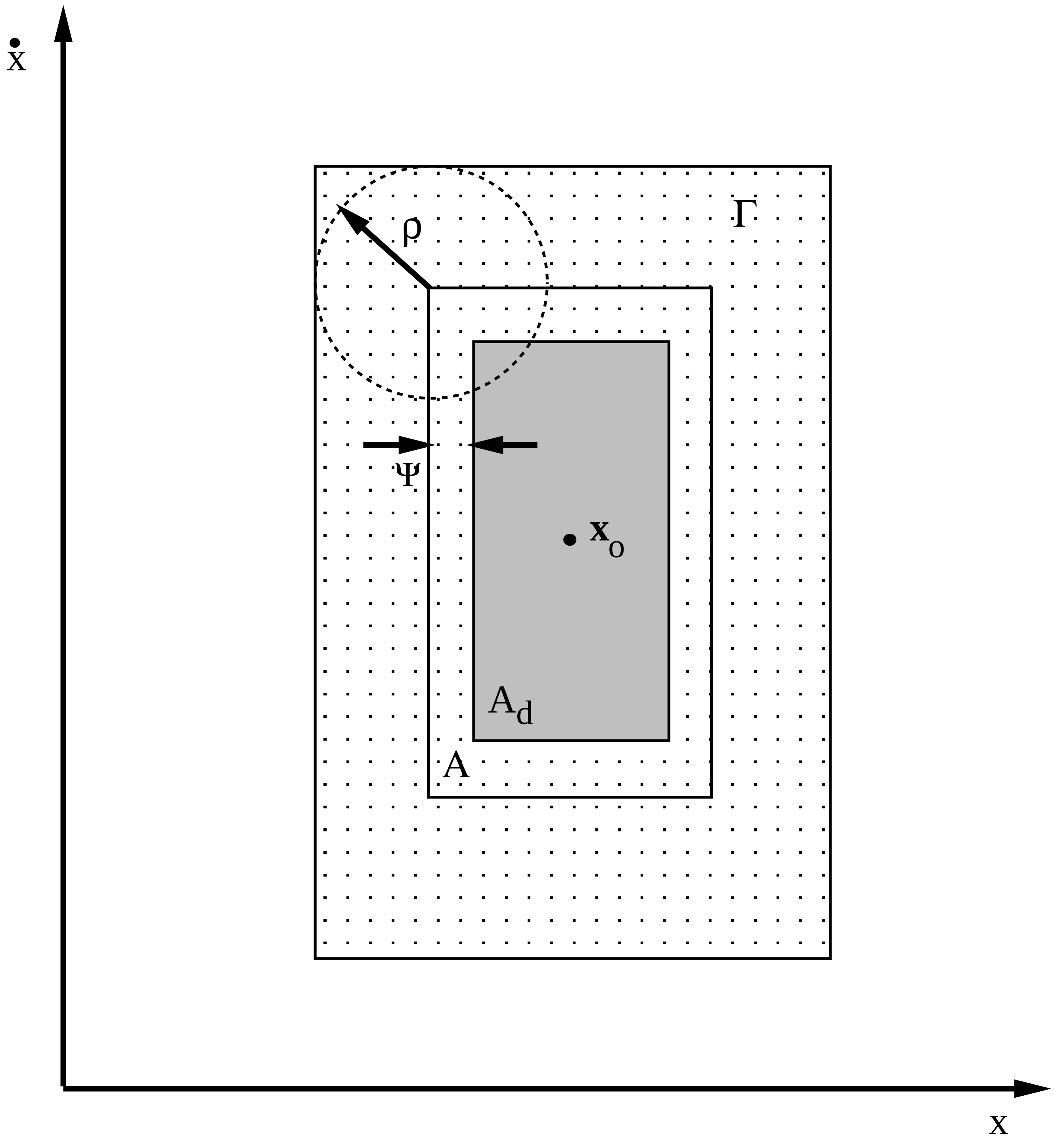


Figure 3: Illustration of the different subsets of the plant state space involved in the design of the proposed adaptive neurocontroller.

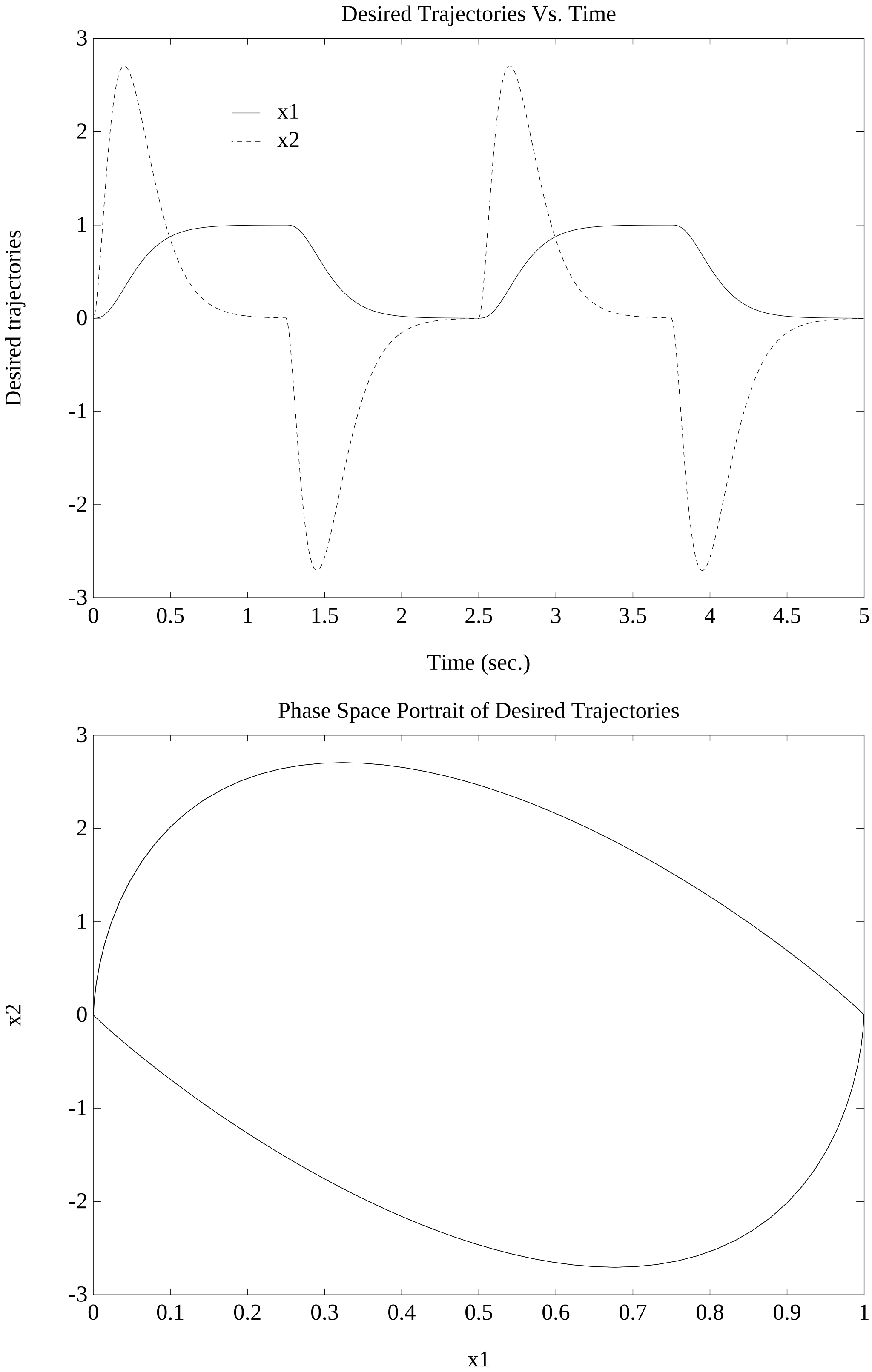


Figure 4: Desired trajectories vs. time for simulations; note that $x_2 = \frac{d}{dt}x_1$. Top: desired trajectories vs. time. Bottom: phase space portrait of desired trajectories.

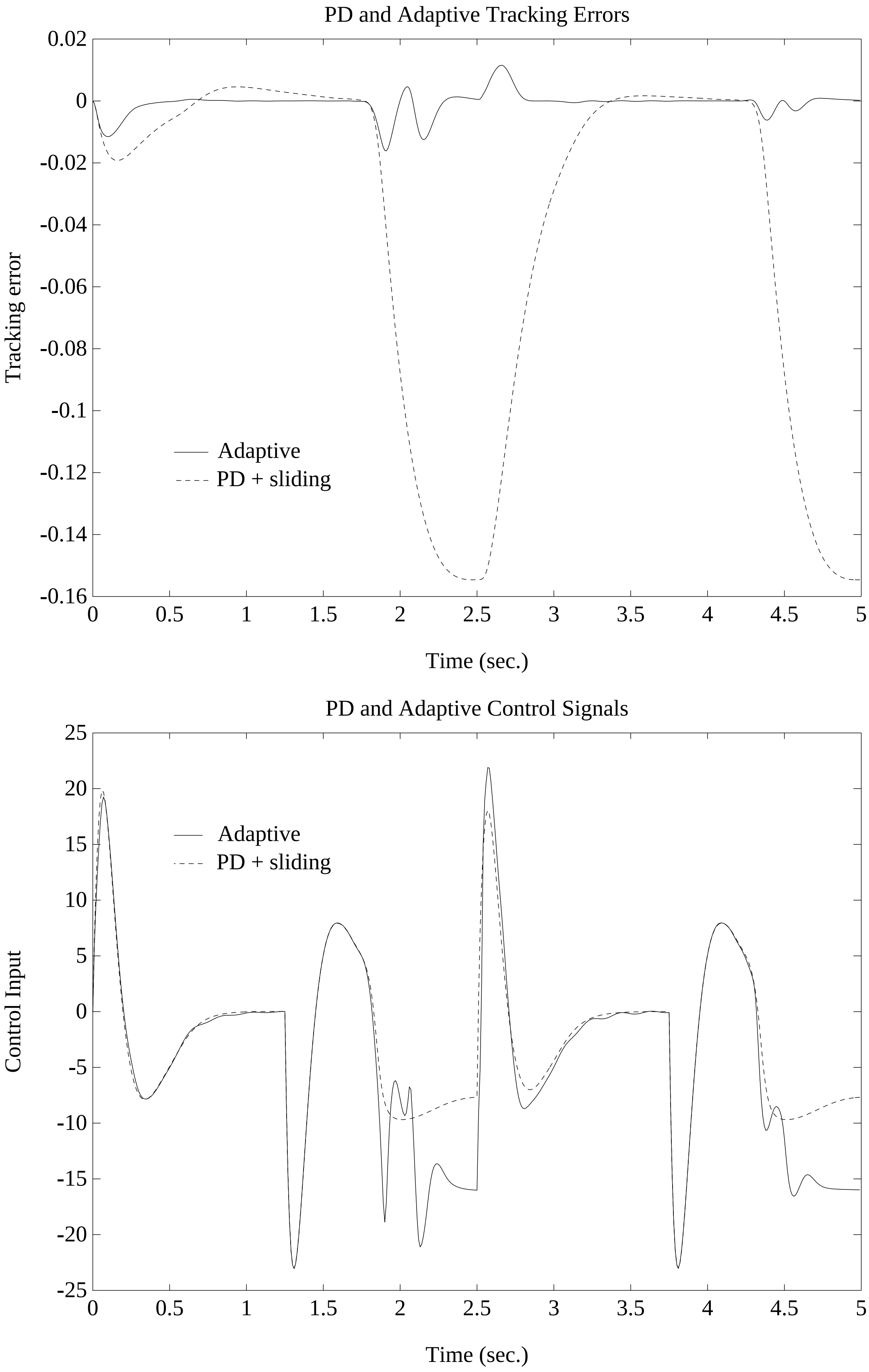


Figure 5: PD and adaptive controller performance during the first 5.0 seconds, unity gain example. Top: tracking errors. Bottom: control signals.

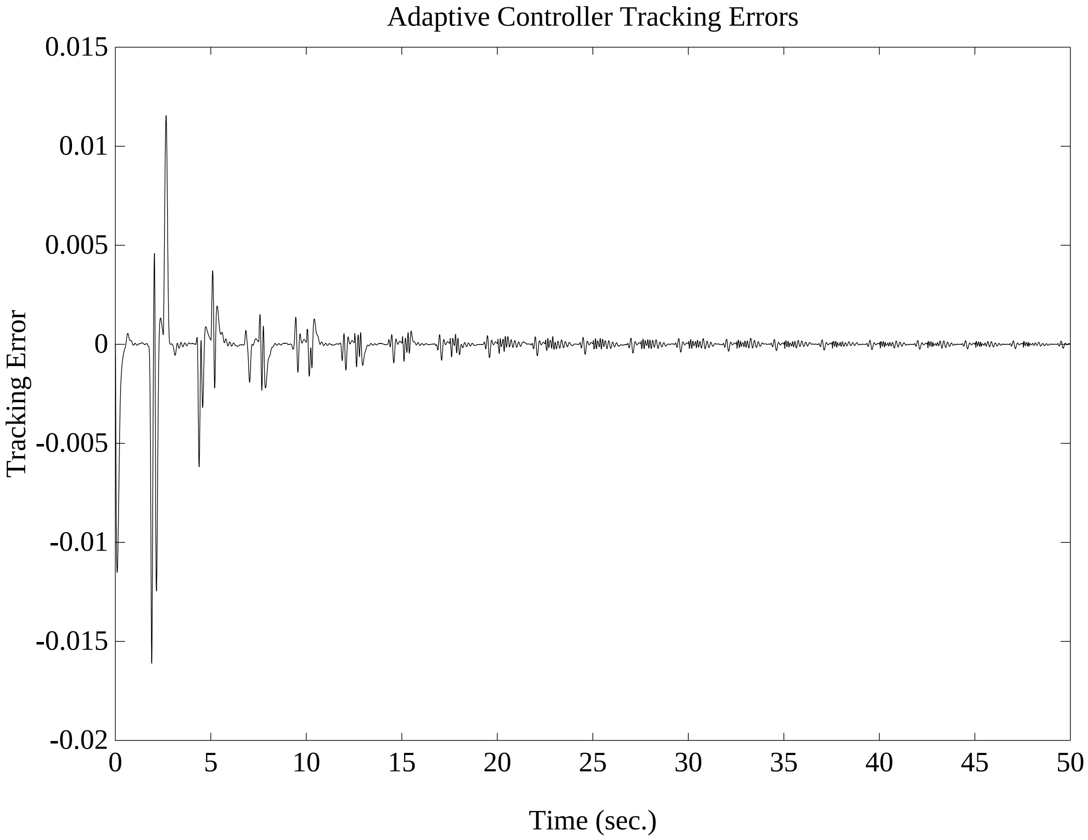
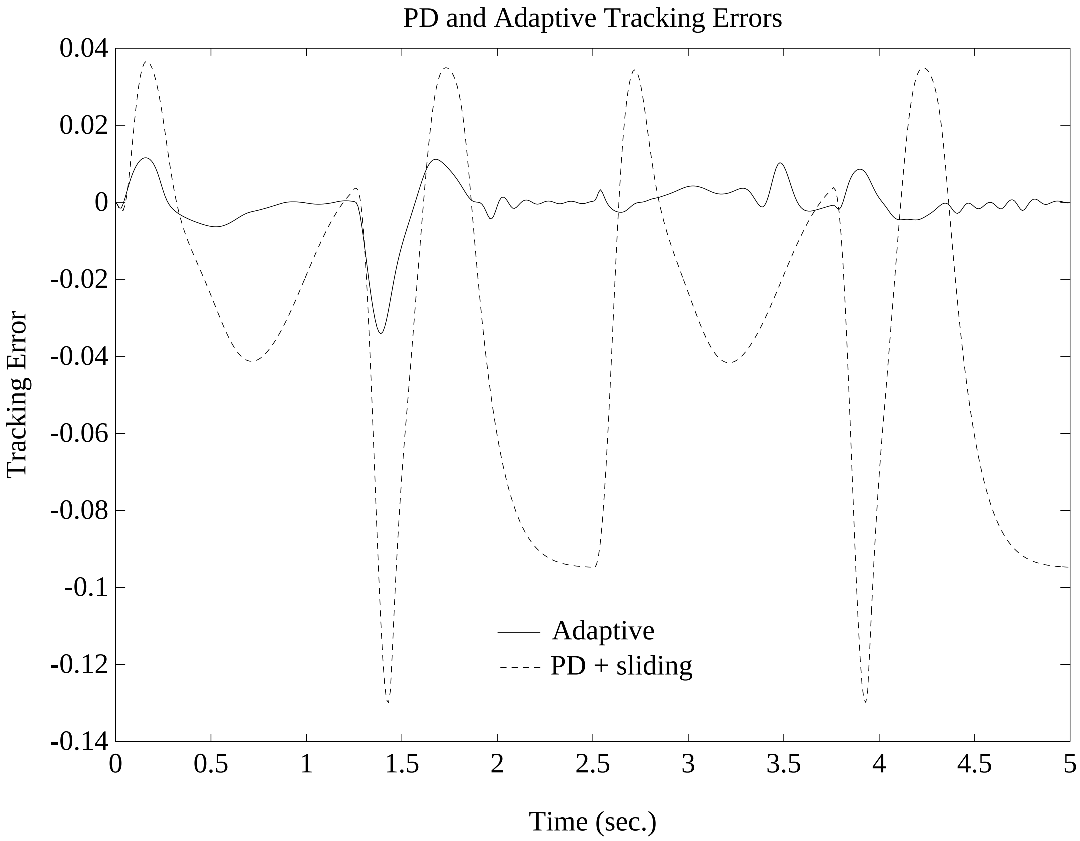


Figure 6: Adaptive controller tracking errors during the first 50 seconds of operation, unity gain example.



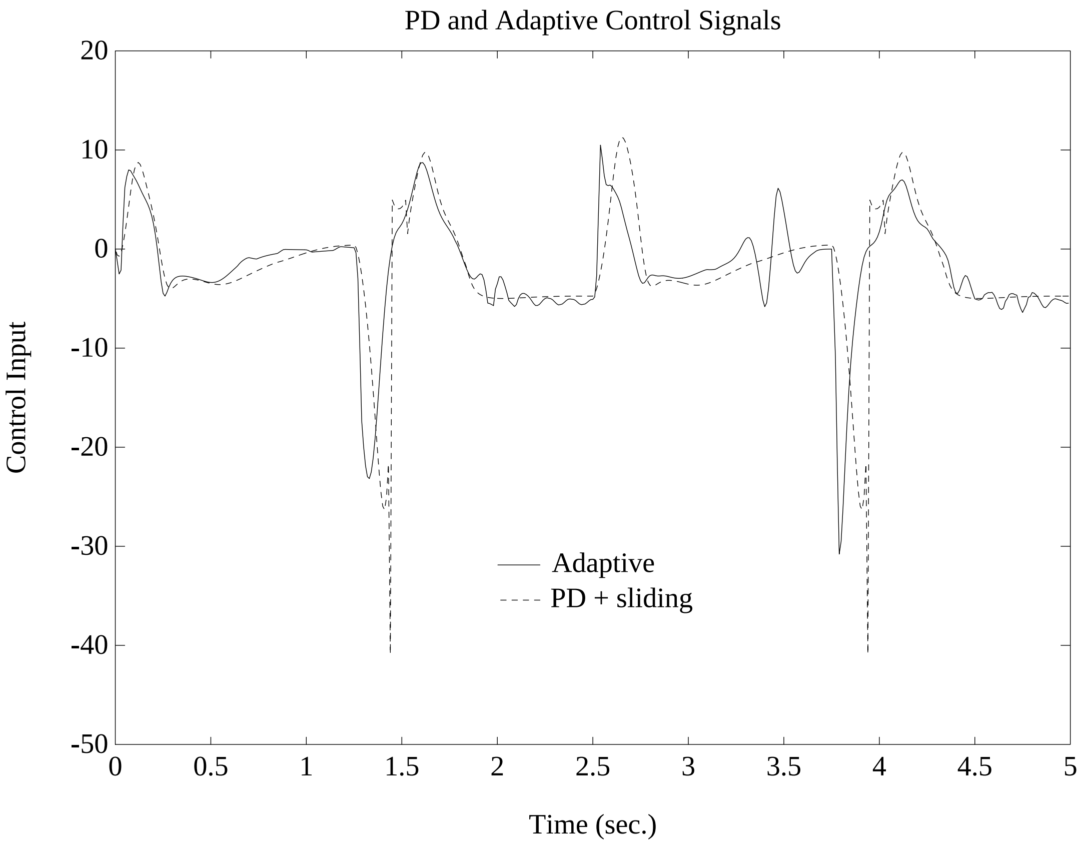


Figure 7: PD and adaptive controller performance during the first 5.0 seconds, unity gain example. Top: tracking errors. Bottom: control signals.

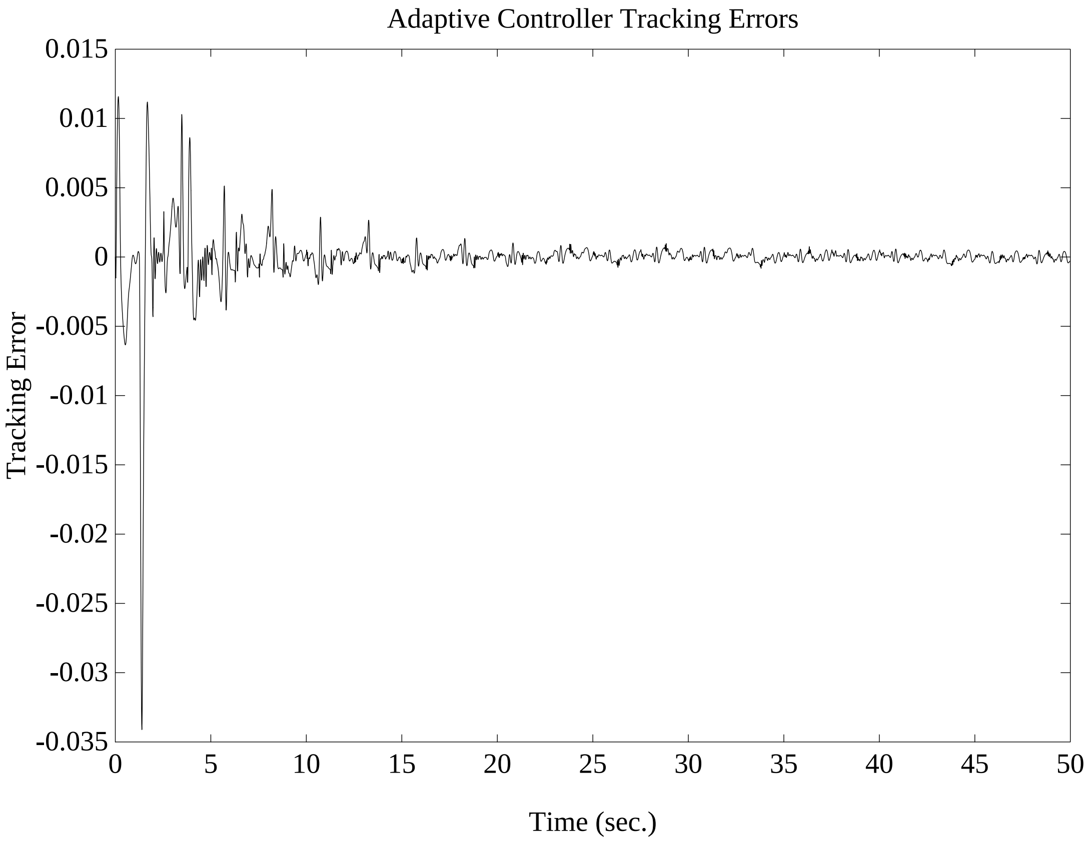


Figure 8: Adaptive controller tracking errors during the first 50 seconds of operation, unity gain example.