A Learning Method for the Approximation of Discontinuous Functions for Stochastic Simulations

by

Alex Arkady Gorodetsky

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2012

Author .....
Department of Aeronautics and Astronautics
May 24, 2012

Certified by .....
Youssef Marzouk
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by .....
Eytan H. Modiano
Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

A Learning Method for the Approximation of Discontinuous Functions for Stochastic Simulations

by

Alex Arkady Gorodetsky

Submitted to the Department of Aeronautics and Astronautics
on May 24, 2012, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Surrogate models for computational simulations are inexpensive input-output approximations that allow expensive analyses, such as the forward propagation of uncertainty and Bayesian statistical inference, to be performed efficiently. When a simulation output does not depend smoothly on its inputs, however, most existing surrogate construction methodologies yield large errors and slow convergence rates. This thesis develops a new methodology for approximating simulation outputs that depend discontinuously on input parameters. Our approach focuses on piecewise smooth outputs and involves two stages: first, efficient detection and localization of discontinuities in high-dimensional parameter spaces using polynomial annihilation, support vector machine classification, and uncertainty sampling; second, approximation of the output on each region using Gaussian process regression. The discontinuity detection methodology is illustrated on examples of up to 11 dimensions, including algebraic models and ODE systems, demonstrating improved scaling and efficiency over other methods found in the literature. Finally, the complete surrogate construction approach is demonstrated on two physical models exhibiting canonical discontinuities: shock formation in Burgers' equation and autoignition in hydrogen-oxygen combustion.

Thesis Supervisor: Youssef Marzouk
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

This thesis is as much the product of my work as it is of the people who have supported me over the past two years. I came into graduate school unaware of all the interesting problems arising in the world and have been dutifully enlightened. I would especially like to thank my advisor, Professor Youssef Marzouk, for his support and dedication throughout my time here. He has always offered his assistance and guidance at all hours of the day and night, and has introduced me to many interesting problems that I hope to pursue in the coming years. Additionally, I would like to thank my mom, grandmother, brother, Dana and friends for their support and acceptance through all the times I have undoubtedly paid less attention to them than I should have. Finally, I would like to thank all of my labmates (too many to name) for all of the wide ranging discussions that have aided me in my time here.

I would also like to thank the BP-MIT Energy Conversion Research Program for funding this work.

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

Uncertainty quantification (UQ) deals with characterizing the effect of uncertainty arising in all aspects of computer simulation. There are two primary types of uncertainty that exist [40]:

- Aleatory uncertainty deals with inherently random quantities that are incoporated into a simulation.

- Epistemic uncertainty deals with quantities that could be measured if proper instrumentation and efforts were expended on measuring them.

These uncertainties present themselves in simulations through various aspects including parameter uncertainty, model inadequacy, residual variability, and observation error, etc. [27]. Parameter uncertainty refers to the uncertainty regarding the parameters a simulation uses to describe some particular physics. Model inadequecy refers to the errors introduced by using a simplified computer simulation to represent a physical process. This uncertainty is present in every computational model because no useful model can incorporate the exact physics used in real world processes. Residual variability refers to incorporating aleatory uncertainties into the simulation. Finally, there are uncertainties introduced into the system when validating/calibrating the simulation to experimental data in the real world. The experimental data are usually noisy measurements of some underyling property of interest.

1.1 Dealing with uncertainty

Uncertainty propagation and inference are two common techniques used to characterize and perhaps improve a simulation before using it to make predictions or to guide decisions. Uncertainty propagation involves propagating the uncertainties in model parameters according to their probability distributions in order to investigate both the distribution of an output of interest as well as sensitivity to the parameters. Contrasting with the forward propagation of uncertainty, another common problem in UQ is using data to infer certain aspects or parameters of a model. This inference is often performed in a Bayesian context in order to generate a distribution over parameter values. Both forward propagation and inference require potentially millions of simulations to be performed and are thus infeasible when dealing with expensive computer simulations. Surrogates for these computationally expensive models must be employed to reduce their computational expense.

1.1.1 Role of surrogates

In order to simplify the analyses described above, surrogates are employed to represent the full models. These surrogates are a computationally efficient approximation of the input to output relationship of a full complex model. They allow millions of simulations to be performed in a reasonable amount of time and are a key enabling technology for uncertainty propagation and inference. Two common types of surrogates employed for this UQ role are polynomial chaos expansions (PCE) [24][59][38] and Gaussian process regression (GPR) [39] [42], also known as kriging [52] in the geostatistics community. PCE represents the model output as a spectral representation using an orthogonal polynomial bases weighted by the distributions of the input. GP represents a Gaussian distribution over functions with a specified covariance kernel. Full model evaluations are used as data to update the GP in to fit the model. The mean function or the entire posterior may then be used for UQ procedures.

When utilizing surrogates for UQ applications there are two considerations to take into account. The first consideration is whether the surrogate actually approximates

the full model in the L_2 sense. When dealing with surrogates one can consider a hierarchy of surrogates, $f_1(x), f_2(x), ..., f_n(x)$ of increasing accuracy or fidelity. The L^2 error for surrogate n is given in Equation 1.1.

$$e_n = \int_{\mathbb{R}^m} |f(x) - f_n(x)|^2 p(x) d(x) \tag{1.1}$$

where $f(x)$ is the true model, $p(x) \geq 0$ is a weight function on $\mathbb{R}^m$. When the surrogate converges in the L^2 sense, we have the error $e_n \to 0$ as $n \to \infty$. In the case of spectral approximations, n corresponds to how many bases are kept in the expansion; whereas in the case of GP, n corresponds to the number of training points. The second consideration is that the probability distribution that is dependent on $f(x)$ is approximated properly by the probability distribution that is dependent on $f_n(x)$. This is a weaker requirement on the surrogate because in many cases multiple functions can result in the same probability distributions. For forward propagation of uncertainty, this requirement is automatically satisfied by the L_2 convergence of the surrogate. Additionally, for PCE surrogates, sensitivity analysis of the system may be performed with no extra expense outside surrogate construction [53] [16].

In the inference context a surrogate replaces the full model in the likelihood function of Bayes rule [34]. The goal is to construct a posterior distribution on some parameters of interest θ, such that the posterior obtained with the surrogate converges to the posterior obtained with the full model as the surrogate is refined. A comparison of this posterior probability distribution can be done using several metrics. A proof stating that the KL divergence of the posterior using the full model (π) from the posterior using the surrogate (π_n) tends to zero as $||f_n - f|| \to 0$ is given in [33]. The KL divergence is given below:

$$D_{KL}(\pi||\pi_n) = \int \pi(\theta) \log \left\{ \frac{\pi(\theta)}{\pi_n(\theta)} \right\} d\theta \tag{1.2}$$

A proof for the convergence in the Hellinger distance can be found in [15] Section 1.2 elaborates on surrogate building methodology.

1.2 Types of surrogate models

Section 1.1.1 introduced incorporating surrogates into uncertainty analyses as a means of increasing computational feasibility. This section introduces some specific surrogates and briefly speaks to issues prohibiting their use.

1.2.1 Spectral representations

Spectral representations are a broad range of surrogates of the form:

$$f(\xi) = \sum_{i=1}^{\infty} f_i \Psi_i(\xi) \tag{1.3}$$

where $f(\xi)$ is a full model evaluation at stochastic parameters ξ. f_i are coefficients corresponding to the i-th mode of the representation and $\Psi(\xi)$ are the modes of the solution and are a function of the stochastic parameters. In practice an approximation is made by truncating the number of bases used for the expansion. A popular approach for choosing bases when dealing with stochastic space is the generalized polynomial chaos (gPC) [59] methodology. This is a Fourier-style representation where $\Psi_i(\xi)$ are orthogonal polynomials with respect to the measure of the random variable ξ:

$$\mathbb{E}_\xi \left[\Psi_i(\xi), \Psi_j(\xi)\right] = \delta_{i,j} \tag{1.4}$$

and chosen based on the Askey-scheme of polynomials corresponding to the distribution of ξ. For example, Hermite polynomials are used when ξ is Gaussian, Legendre polynomials are used when ξ is uniform, etc. [59]. The coefficients can be calculated by projecting the function onto the bases:

$$f_i = \mathbb{E}\left[f(\xi), \Psi_i(\xi)\right] = \int f(\xi)\Psi_i(\xi)\pi(\xi)d\xi, \tag{1.5}$$

where $\pi(\xi)$ is the distribution of the random variable ξ. In practice the infinite series is truncated to an order P, for implementation purposes resulting in an approximation, $\hat{f}_P$, of the true model. This formulation is extended to multiple dimensions using

tensor products of 1D polynomials and total order expansions with $\binom{N}{P}$ terms. N is the number of dimensions of f. The spectral convergence of the truncation satisfies:

$$||f - \hat{f}_P||_{L^2}^2 \leq CP^p||f||_{H_w^p}^2 \tag{1.6}$$

where H_w^p is the weighted Sobolev norm and p is the number of L^2 derivatives of f. Therefore, if f is infinitely smooth, exponential convergence is obtained.

The coefficients f_i can be determined by (intrusive) Galerkin methods, pseudospectral projection methods, regression, or stochastic collocation [57]. The first method involves a reformulation of the governing equations while the latter three methods are non-intrusive and treat the deterministic model as a black box. Because the focus of this work deals with non-intrusive methods, a discussion of the three major methods are presented.

Pseudospectral projection methods evaluate the integral in Equation 1.5 using Monte Carlo methods or quadrature points. In the MC approach coefficient i can be computed with Equation 1.7.

$$f_i \approx \sum_{i=1}^{N_{MC}} f(\xi_i)\psi(\xi_i), \quad \xi_i \sim \pi(\xi) \tag{1.7}$$

Whereas in the quadrature approach the integral becomes:

$$f_i \approx \sum_{j=1}^{N_{quad}} w_j f(\xi_j)\psi(\xi_j) \tag{1.8}$$

Popular quadratures are Clenshaw-Curtis quadrature and Gaussian quadrature. An n-point Gaussian quadrature rule can integrate exactly a polynomial of degree $2n-1$. Therefore, the number of function evaluations depends on the order of the expansion. An extension to higher dimensions can be performed using tensor products of quadrature rules. However, this reduces the scalability of this algorithm and dimension adaptive tensor product quadrature [23] and Smolyak based sparse grid algorithms are often used to determine relevant dimensions for refinement [6] [3]. polynomial re-

gression does not attempt to perform the projection of the full model onto each basis directly using the integral in Equation 1.5; rather the coefficients are determined by solving the linear system:

$$\mathbf{f} = \Psi\hat{\mathbf{u}}, \tag{1.9}$$

where $\mathbf{f}$ is a vector $N \times x1$ of the function values at locations ξ, Ψ is an $N \times xP$ matrix where P is the maximum order of the basis used and N is the number of function evaluations. Finally, $\hat{\mathbf{u}}$ are the P coefficients of the expansion. The locations ξ can be chosen with design of experiments or from previously available model evaluations. Several options exist for solving this linear system depending upon N and P. If $N = P$ standard solution techniques for square systems can be used. If $N > P$ then the overdetermined system can be solved using least squares. For $N < P$ sparsity seeking algorithms have recently been utilized to improve efficiency and scaling of the polynomial chaos approach [20]. These sparse approaches are extensions of the work performed in the compressive sensing community. Recent work has demonstrated that these methods can be very effective if $\mathbf{f}$ is truley sparse in Ψ, meaning that few $\hat{\mathbf{u}}$ are nonzero. Details on solutions of these equations with sparsity seeking algorithms can be found in [9], [8], [10], [19], and [5]. These methods offer the potential benefit of requiring fewer function evaluations to obtain comparable level of accuracies of the other techniques mention hered.

Finally, stochastic collocation [58], or interpolation, is distinguished from the next previous two methods by the fact that the surrogate exactly reproduces the true function at the training data. It is performed using Lagrangian polynomials that are determined by the data:

$$f_n(\xi) = \sum_{i=1}^{N_p} f_i L_i(\xi) \tag{1.10}$$

The Lagrangian polynomial L_i is zero at all collocation points except the i-th collocation point. The u_i are the true function value at the i-th collocation point. When using tensor product quadrature points these polynomials become equivalent to those used in pseudospectral projection. Performance of this technique is maximized when utilizing the same collocation grid as that for the quadrature in the pseudospectral

projection approach [21]. One disadvantage of using the stochastic collocation approach is that the entire interpolation grid must be stored.

1.2.2 Gaussian process regression

Another common surrogate employed for UQ purposes is the GPR model. In this formulation we have a Gaussian distribution over functions:

$$\mathbb{P}(u) = \mathcal{N}(\mu, \Sigma) \tag{1.11}$$

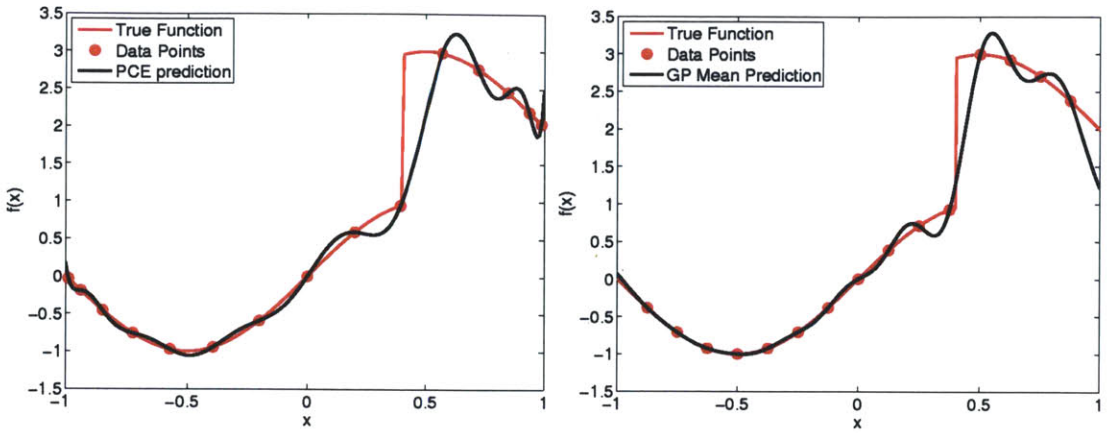
where Σ is the covariance kernel. When training the GP using data from the full model, the covariance matrix is evaluated at the points ξ_i obtained from the data $\{(\xi_i, y_i)\}$ where ξ and $\mathbf{y}$ are the data point vector and the full model evaluation vector respectively. New predictions at a point ξ_* have a distribution give by:

$$\mathbb{P}(\xi^*|\xi) = \mathcal{N}\left(\Sigma_{\xi,\xi_*}\Sigma_{\xi}^{-1}\mathbf{y}, \Sigma_{\xi_*,\xi_*} - \Sigma_{\xi,\xi_*}\Sigma_{\xi}^{-1}\Sigma_{\xi,\xi_*}^{T}\right) \tag{1.12}$$

Σ_{ξ_*,ξ_*} is a scalar. Σ_{ξ,ξ_*} is $N \times 1$ vector where N is the number of data points. Σ_{ξ} is a $N \times N$ matrix. For prediction either the mean of the GP can be used or a fully Bayesian analysis may be performed by incorporating model uncertainty. The evaluation of the mean at a new input requires the inversion of Σ which is an $\mathcal{O}(N^3)$ operation.

1.2.3 Outstanding issues with surrogates

Although surrogates and function approximation have a rich history of algorithms and methods there are still open areas of research dealing with approximating higher dimensional models as well as models exhibiting increasing complexity. The dimensionality problems stem from the fact that exponentially increasing numbers of basis functions and function evaluations are required for increasing dimension. The complexity of the model, specifically discontinuities in the stochastic space, is another issue for the methods mentioned above. These methods' convergence depends on



(a) PCE Gibbs (b) GP Gibbs

Figure 1-1: Gibbs oscillations obtained from using both a pseudospectral projection PCE surrogate and a Gaussian process surrogate using 15 full function evaluations

smoothness and goes from exponential to algebraic convergence in the presence of discontinuities. Approximating discontinuities with smooth global functions yields Gibbs oscillations as shown in Figure 1-1. Figure 1-1 displays the severity of the problem associated with building a surrogate for discontinuous function. Solving this problem is the topic of this thesis.

1.2.4 This work

This thesis aims to deal with surrogates for models exhibiting discontinuities. Chapter 2 provides a literature review of the current algorithms available for accomplishing these tasks. A majority of the state of the art algorithms tackle this problem by decomposing a domain containing a discontinuity into two or more smooth domains. This methodology allows the smooth surrogates and approximation techniques discussed in section 1.2 to be effectively applied in these problems. This methodology is used in this thesis, but executed in an efficient manner that takes advantage of the regularity of discontinuities in the parameter space. Figure 1-2 displays a cartoon of discontinuities exhibiting different levels of regularity in the parameter space. The contributions of this thesis includes the development of an efficient discontinuity detection algorithm to take advantage of fairly regular characteristics of the disconti-

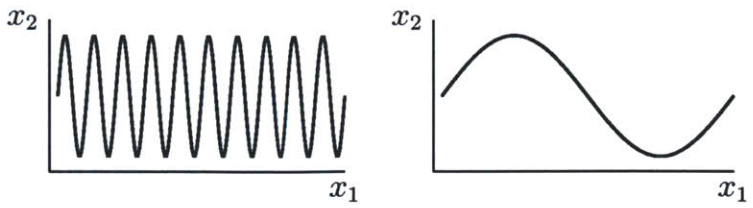


Figure 1-2: A comparison of discontinuities exhibiting various levels of regularity in the parameter space. The cartoon on the left requires function evaluations tracking the discontinuity in order to approximate it accurately. The cartoon on the right indicates a discontinuity that can be well approximated by few function evaluations.

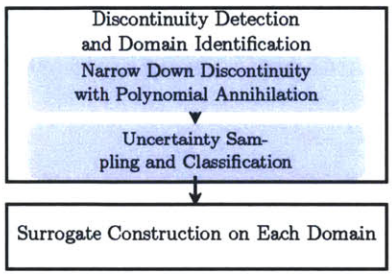


Figure 1-3: Algorithm Workflow

nuity in the parameter space. By utilizing smooth discontinuity approximation tools such as support vector machines, as well as scalable algorithms based on the active learning/uncertainty sampling paradigm, the algorithm developed is shown to have improved scaling within dimension compared to existing techniques. The algorithm performs this task by following three steps: efficient location of discontinuties in multi-dimensional space, the generation of a classifying function, and finally the construction of a surrogate on these two regions with the option to evaluate the full model in the area neighboring the discontinuity. The overall workflow for the algorithm is shown in Figure 1-3. The final result of the algorithm is a Gaussian process representation of the full model:

$$f(x) = \sum_{j=1}^{N_{reg}} I_j \sum_{i}^{n_j} \alpha_{ij} \Psi_{ij}(x), \tag{1.13}$$

where N_{reg} is the number of regions the entire domain is decomposed into, n_j is the number of training samples in region j, and I_j is an indicator function. They are

defined as follows:

$$I_j = \begin{cases} 1 & \text{if } x \text{ is in region } j. \\ 0 & \text{if } x \text{ is not in region } j. \end{cases}$$

In other words, a GP is built on each region that the GP is decomposed into. Additionally, there is an option to evaluate the full model in regions near the discontinuity.

Chapter 3 describes the tools and techniques used as part of the discontinuity detection. Chapter 4 will describe the novel algorithm for discontinuity detection, which is the main focus of this thesis. Next, chapter 5 describes how function approximation is done on the subdomains resulting from discontinuity detection. Chapter 6 summarizes the main findings and provides future work.

Chapter 2

Literature review of surrogates for discontinuous functions

2.1 General methodologies

The algorithms described in this chapter are designed to approximate computationally expensive simulations that exhibit strong nonlinearities and discontinuities in high dimensional ($\mathcal{O}(10)$) spaces. These nonlinearities and discontinuities may be the result of bifurcations, tipping points, etc. of the physical system that a model is simulating. In these situations, the response of the model in the different regions separated by a discontinuity may correspond to different physical regimes or different operating conditions that are strongly dependent on the input parameters. Conventional approximation techniques, described in the previous chapter, using smooth basis functions exhibit Gibbs phenomenon when encountering these discontinuities. Therefore the usual exponential convergence of these techniques is drastically reduced.

The main methods introduced in this literature review utilize spectral methods that are modified to deal with the discontinuities and extreme nonlinearities. The modifications used to approximate these functions generally fall into three categories: decomposing the parameter space into smaller elements on which to construct approximations as in multi-element gPC [56], using a basis function capable of capturing local effects [31] [30], or filtering the Gibbs phenomena [12]. The domain decompo-

sition techniques themselves fall into several categories: those splitting a hypercube domain into smaller hypercubes and those tracking the discontinuity through a variety of edge tracking methods in order to split the domain into two or more irregular domains. Although successful in low dimensions, these techniques have exhibited challenges associated with their application to higher dimensional problems.

2.1.1 Adaptive refinement

The adaptive refinement techniques discussed in this section will approach the approximation problem by breaking up a domain containing a discontinuity into subdomains containing smooth functions. This type of approach allows the approximation of the entire discontinuous function to be broken into the approximation of many smooth functions. The advantage here is that many methods exist for smooth approximation, for example as a linear combination of orthonormal functions or by Gaussian processes. Examples of this approach are given in [56] and [1]. These adaptive refinement algorithms are defined by three aspects: refinement criteria, point selection, and type of approximation. These three aspects are interconnected because the refinement criteria and the point selection are often guided by the type of approximation performed in each subdomain.

Two main methodologies for refinement guided by spectral approximations exist in the literature. The first, by Wan and Karniadakis (2005), exists in the context of intrusive gPC expansions. The criterion involves computing the decay rate of the error of the approximation using the variance in a subdomain. If the decay rate is large, then that subdomain is split into two subdomains. For example, assume we have N subdomains and subdomain k has the expansion:

$$\hat{u}_k(\xi^k) = \sum_{i=1}^{N_P} \hat{u}_{k,i}\Psi_i(\xi^k), \tag{2.1}$$

with a local variance:

$$\sigma_k^2 = \sum_{i=1}^{N_P} \hat{u}_{k,i}^2 \mathbb{E}[\Psi_i^2] \tag{2.2}$$

The global mean and variance of the true function u are then approximately:

$$\begin{aligned}\bar{u} &= \sum_{k=1}^{N} \hat{u}_{k,0} J_k, \\ \bar{\sigma} &= \sum_{k=1}^{N} [\sigma_k^2 + (\hat{u}_{k,0} - \bar{u})^2] J_k,\end{aligned}$$

where J_k is a factor that is dictated by the element size. Now the local decay rate of the error of the gPC approximation in each element is defined as:

$$\eta_k = \frac{\sum_{i=N_{P-1}+1}^{N_P} \hat{u}_{k,i}^2 \mathbb{E}[\Psi_i^2]}{\sigma_k^2} \tag{2.3}$$

A subdomain is furthur split into two if the following condition is met:

$$\eta_k^{\alpha} J_k \geq \theta_1, \quad 0 < \alpha < 1, \tag{2.4}$$

where α and θ_1 is a limit on the size of the subdomains. This method has advantages if the the function is actually extremely nonlinear rather than discontinuous, as the regions required to approximate a discontinuity become increasingly small. However, it still requires progressively smaller scale grids with equal amounts of function evaluations on which to construct the PCE expansions. For these reasons this methodology has not be shown to scale well with stochastic dimensions. For furthur information consult [56].

The second methodology based on spectral expansions is developed by Archibald et al. (2009). This methodology utilizes a different refinement criteria which is based on the ideas of polynomial annihilation (PA) developed in [2]. Polynomial annihilation is thoroughly described in Chapter 3, but the methodology estimates the error in progressively higher order polynomial approximations at a test point. If the error does not decay, then one obtains the estimate of the jump, or size of the discontinuity at the test point. The main innovation of [1] involves applying the PA method axially on a gPC surrogate of the solution. Figure 2-1 obtained from [1] displays a one dimensional example. Figure 2-1(a) shows the Gibbs oscillation as a result of

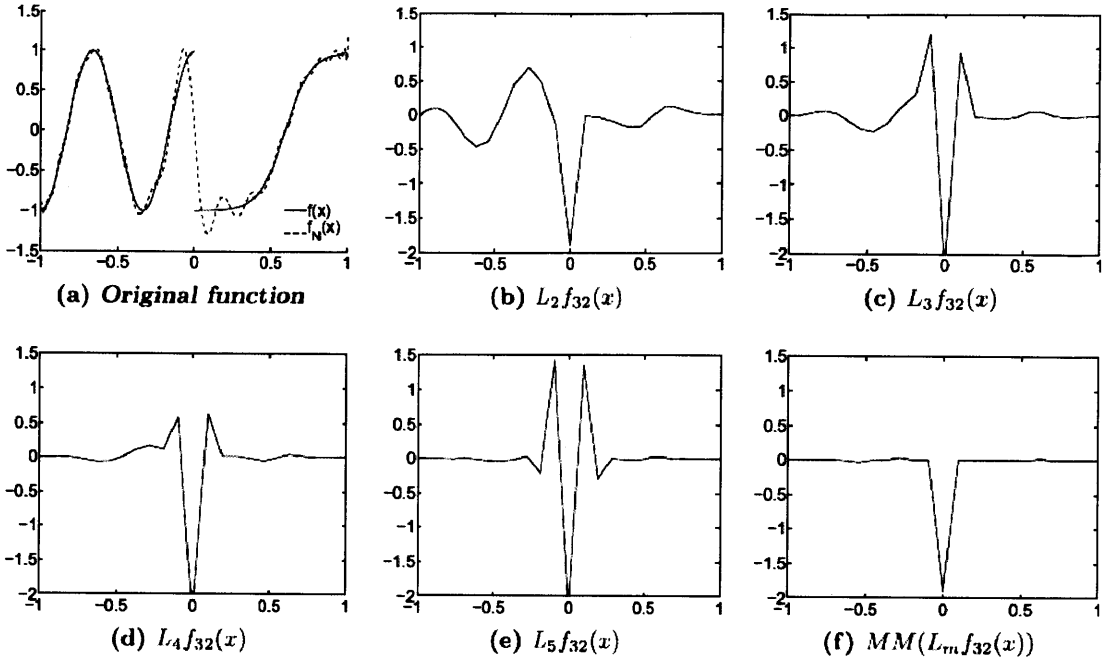


Fig. 1. Example 3.1. (a) underlying function on $N = 32$ Legendre Gauss points; (b)-(e) jump function approximation using various orders m; (f) Minmod results (19), for $\mathcal{M} = \{1, 2, 3, 4, 5\}$.

Figure 2-1: Polynomial Annihilation running on a gPC expansion from Archibald et al. (2009)

approximating the solid discontinuous function using smooth polnyomials. Archibald then takes advantage of these oscillations by performing PA to annihilate 2nd, 3rd, 4th and 5th order polynomials. Subfigures (b)-(e) show the approximations of the size of the discontinuity as a function of the x-axis. Finally, using a MinMod limiter combines these approximations to correctly show a single jump at x=0 of size, $[f](0) = -2$.

This scheme has several clear advantages such as the ability to approximate the full model using standard gPC techniques, thus allowing an efficient evaluation of the jump function throughout the whole grid. However, one disadvantage is that the initial PCE is required to be fairly accurate in the areas far from the discontinuity. This need not be the case as Gibbs oscillations often propagate through the domain. Once a discontinuity is detected, the domain is then split and piecewise continuous gPC bases are used in each subdomain. The result of this adaptation on a linear discontinuity is shown below in Figure 2-2.

A common characteristic of these refinement techniques, evident in Figure 2-2, is that the density of the grid increases as the discontinuity is refined. This characteristic

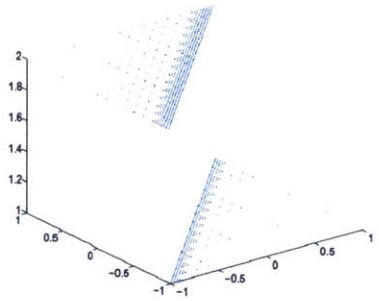


Figure 2-2: Adaptive Refinement Grid by Archibald et al. (2009)

appears regardless of the discontinuity shape in the parameter space. This method can lead to inefficiencies because the discontinuity can potentially be described by far fewer full model evaluations. However, the PA technique provides an attractive and efficient way of locating and evaluating the size of a discontinuity and will be an important tool in the algorithm described in Chapter 4.

2.1.2 Edge tracking

An efficient algorithm for discontinuity detection has been developed by Jakeman et al. (2011). The algorithm focuses specifically on the search for a discontinuity rather than the approximation of the true model. Therefore, effort is spent on efficiently locating the discontinuity and then progressively adding points along the discontinuity using PA as an indicator. This procedure is adaptive, utilizing a divide an conquer approach, and performs PA axially. For a detailed description of the implementation refer to [25]. The points along the discontinuity are then labeled using the approximation to the discontinuity size obtained by PA. Finally, new model evaluations are classified using a nearest neighbor approach. An example of this tracking is given in Figure 2-3. In this example, the discontinuity is a circle with radius 0.55 centered at the origin. The increasing density characteristic of the adaptive refinement is replaced by a clustering of points along the discontinuity, with almost no points in the middle of either region. This technique exhibits improved scaling compared to the adaptive refinement methods discussed in the previous section; however, it also scales poorly with dimension because the points required to track the discontinuity increase

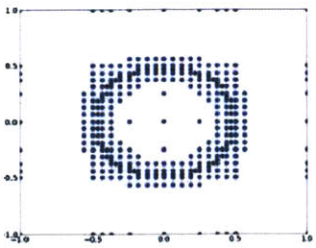


Figure 2-3: Edge Tracking Grid by Jakeman (2011)

exponentially. Finally, the method does not solve the approximation issue, and must be coupled with another approximation method.

2.1.3 Local basis functions

In contrast to domain decomposition methods, methods relying on local basis functions deal with discontinuities by using specially tailored bases. These types of techniques, for example utilizing wavelets [37] [17], have been extensively used in the image processing community [28]. Images often have sharp edges that are accurately represented with a wavelet basis. However, high dimensional discontinuities require specially tailored wavelets to the stochastic dimensions. The description of one dimensional multi-resolution analysis below follows the explanation given in [30]. First the order of polynomials is defined using $N_o = 0, 1, ...$ and the resolution levels are defined using $k = 0, 1, 2, ...,$. The space of piecewise-continuous polynomials, $\mathbf{V}_k^{N_o}$, is defined according to

$$\mathbf{V}_k^{N_o} \equiv \left\{ f : \text{the restriction of f to the interval } \left(2^{-k}l, 2^{-k}(l+1)\right) \right. \tag{2.5}$$

$$\left. \text{is a polynomial of degree} \leq N_i, \text{ for } l = 0, ..., 2^k - 1 \right\}, \tag{2.6}$$

and

$$\mathbf{V}_k^{N_o} \equiv \{ f \text{ vanishes outside the interval } [0, 1] \} \tag{2.7}$$

Now, the multi-wavelet subspace $\mathbf{W}_k^{N_o}, k = 0, 1, 2, ...$ is defined as the orthogonal complement of $\mathbf{V}_k^{N_o}$ in $\mathbf{V}_{k+1}^{N_o}$.

In order to setup the multi-wavelet subspace, an the orthonormal basis of piecewise

polynomials , $\{\psi_0, \psi_1, ..., \psi_{N_0}\}$., is constructed for $\mathbf{W}_0^{N_0}$. Now the space $\mathbf{W}_k^{N_0}$, whose dimension is $(N_0 + 1)2^k$, is spanned by multiwavelets, ψ_{jl}^k which are truncated and dilated version of ψ_i. They are given by:

$$\psi_{jl}^k(x) = 2^{k/2}\psi_j(2^k x - l), \quad j = 0, ..., N_0 \text{and} l = 0, ..., 2^k - 1, \tag{2.8}$$

with support $\text{Supp}(\psi_{jl}^k) = [2^{-}kl, 2^{-k}(l + 1)]$ A basis for V_0^k is created using rescaled orthonormal Legendre polynomials of degree i, defined over [-1,1]:

$$\phi_i(x) = \mathcal{L}_{l_i}(2x - 1), \quad i = 0, 1, ..., N_0 \tag{2.9}$$

Now the space $\mathbf{V}_k^{N_o}$ is defined analogously to Equation 2.8. Using this machinery an approximation f^{N_0,N_r} of a function $f(x) \in L^2([0, 1])$ is constructed by projecting $f(x)$ onto $\mathbf{V}_{N_r}^{N_o}$

$$f^{N_0,N_r} = \mathbb{P}[f] = \sum_{l=0}^{2^{N_r}-1} \sum_{i=0}^{N_0} \left\langle \phi_{il}^{N_r}, f(x) \right\rangle \phi_{il}^{N_r}(x) = \sum_{l=0}^{2^{N_r}-1} \sum_{i=0}^{N_0} \bar{f}_{il}^{N_r} \phi_{il}^{N_r}(x) \tag{2.10}$$

In terms of multi-wavelets the expansion can be written as:

$$f^{N_0,N_r}(x) = \mathbb{P}_0^{N_0}[f(x)] = \sum_{k=0}^{2^{N_r}-1} \sum_{l=0}^{2^k-1} \left(\sum_{i=0}^{N_0} df_{il}^k \psi_{il}^k(x) \right), \tag{2.11}$$

and the coefficients df_{il}^k are given by:

$$df_{il}^k = \left\langle \left\{ \mathbb{P}_{k+1}^{N_0}[f] - \mathbb{P}_k^{N_o}[f] \right\}, \psi_{il}^k \right\rangle \tag{2.12}$$

The ability to expand the function f in terms of these locally supported bases allows for local phenomena to be adequately captured and errors to be localized. This has two advantages over expressing the function as a spectral representation of global bases. The first advantage is that MW may be utilized to adaptively increase resolution levels near localized phenomena. The second advantage is that resulting errors are localized as compared to using global bases where local errors cause global

changes in the approximation. Additionally, LeMeitre et al. (2004) demonstrated that coupling these localized expansions with a domain decomposition algorithm based on the multi-wavelets yields an effective adaptation technique which reduces CPU time and allows improved scaling. This technique becomes similar to the domain decomposition techniques outlined in 2.1.1 with the exception that a multi-wavelet expansion is implemented on each regime.

2.1.4 Filtering

The final prominent methodology for dealing with discontinuities in the literature is based on the observation that discontinuities cause a slow decay of the coefficients of a spectral expansions. Chantrasmi et. al (2009) [12] effectively uses the smeared and oscillating results of using global expansions to filter the coefficients and create a single smooth approximation of the discontinuity. This technique builds global Padé-Legendre polynomials based on quadrature points and employs a filter to remove oscillations. The advantage of this method is that existing global polynomials which capture the smoothly varying portions of the function may be utilized. While these methods have been shown to be fairly effective, scaling to more than two or three dimensions has not been demonstrated. Moreover, filtering a global approximation of a truly discontinuous function may yield an undesirably large degree of smoothing.

2.2 Summary

Domain decomposition techniques based on adaptive grids show promise for high dimensional applications because they take advantage of the exponential convergence for smooth functions for surrogates on the various subdomains. These techniques were utilized not only with global PCEs, but proven effective when utilizing multi-wavelet expansions in the subdomains. However, these techniques have only been showcased on fairly low dimension problems because traditional methods of determining the expansions need a hybercube domain in order to accurately calculate their coefficients using projection. Discontinuities that break up the domain into irregular shapes

involve many high resolution hypercubes in order to accurately capture the area along the discontinuity, dramatically increasing the computational expense.

Domain decomposition techniques based on edge tracking detect discontinuities more efficiently and separate the problems of discontinuity detection and surrogate construction. However, current methods for edge tracking generally scale exponentially with dimension because they involve putting a uniformly spaced grid of points around the discontinuity. These edge tracking methods provide the inspiration for the algorithm described in chapter 4. In chapter 4, a method that takes advantage of smooth discontinuities in the parameter space is described. In fact, this method defaults to edge-tracking if the discontinuity itself contains localized features.

Finally, the Padé-Legendre techniques offer a way to utilize global functions for the approximation of discontinuous functions through a filtering process. These methods have shown to be fairly effective; however, not thoroughly tested in higher dimensional problems. Higher dimensional problems may cause too much smoothing and a greating computational expense.

Overall, the domain decomposition strategy is an opportunity for an extension of efficient approximations of discontinuities to higher dimensions. This strategy forms a backbone of the major contributions in this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

Discontinuity detection tools

The novel discontinuity detection algorithm described in Chapter 4 is built upon the foundations of several different tools common in the machine learning community. These tools will be used to address two problems related to approximating high dimensional discontinuous functions. The first problem is determining an efficient way of parameterizing the discontinuity and the second problem is determining where to evaluate the expensive model in order to best inform the location of the discontinuity. Three methods are used to answer these questions.

The first tool, polynomial annihilation, is used to to label the region that a function evaluation belongs. This is the same polynomial annihilation method used in the domain decomposition schemes described in the previous section, and is described in section 3.1. The second tool is used to create a classifier in order to determine to which region new function evaluations belong. Support vector machines (SVM)s are used to create the classifier and a vital aspect of the functional approximation of discontinuities in the approximation algorithm developed in Chapter 4. For this reason, their foundations are described in section 3.2. The SVM optimization statement, performance measures, and algorithms are also described.

The final tool, uncertainty sampling (US) is one that is able to guide an adaptive sampling scheme in order to efficiently use full model evaluations to locate the discontinuity. US is a subset of active learning. Section 3.3 describes a technique that allows new function evaluations to be placed where the current guess for the

location of the discontinuity is most uncertain. This methodology allows an effective refinement of the discontinuity to be performed in high dimensions.

3.1 Polynomial annihilation

Polynomial annihilation is used in order to measure the size of a discontinuity or region of rapid change in a function. This measurement is vital in order to determine to which region new model evaluations belong. Following [2], a description of one-dimensional polynomial annihilation is given here. The motivation is to construct an approximation to the jump function by removing sucessive orders of polynomials. The jump function is defined below:

$$[f](x) = f(x+) - f(x-). \tag{3.1}$$

Where

$$f(x-) = \lim_{\Delta \to 0} f(x - \Delta), \tag{3.2}$$

$$f(x+) = \lim_{\Delta \to 0} f(x + \Delta). \tag{3.3}$$

Therefore, when the function is dicontinuous at x then $[f](x)$ is non-zero and it is zero in all other instances. The main result of polynomial annihilation is the approximation $L_m f$ of the jump function. This approximation has the following form:

$$L_m f(x) = \frac{1}{q_m(x)} \sum_{x_j \in \mathcal{S}_x} c_j(x) f(x_j). \tag{3.4}$$

The coefficients (c_j) are calculated by solving the system of equations:

$$\sum_{x_j \in \mathcal{S}_x} c_j(x) p_i(x_j) = p_i^{(m)}(x), \quad i = 0, ..., m. \tag{3.5}$$

m is the order of desired annihilation, p_i are a polynomial or monomial basis, and S_x is a stencil of the m+1 nearest points at which a function evaluation has taken place.

An analytic solution for c_j is given in Equation 3.6

$$c_j(x) = \frac{m!}{\prod_{\substack{i=0 \\ i \neq j}}^{m} (x_j - x_i)}, \quad j = 0, ..., m \tag{3.6}$$

The normalization factor is:

$$q_m(x) = \sum_{x_j \in \mathcal{S}_x^+} c_j(x). \tag{3.7}$$

$\mathcal{S}_x^+$ is the set of $\{x_j\}$ where $x_j > x$. Finally the accuracy of this approximation is given below:

$$L_m f(x) = \begin{cases} [f](\xi) + \mathcal{O}(h(x)) & if\ x_{j-1} \leq \xi, x, \leq x_j, \\ \mathcal{O}(h^{min(m,k)}(x)) & if\ f \in C^k(I_x)\ for\ k > 0 \end{cases}$$

I_x is the smallest interval of points $\{x_j\}$ that contains the set S_x. $h(x)$ is defined as the largest difference between neighboring points in the stencil $\mathcal{S}_x$:

$$h(x) = \max \{|x_i - x_{i-1}| : x_{i-1}, x_i \in \mathcal{S}_x\} \tag{3.8}$$

The proof is given in [2] and is based on the residual of the Taylor series expansion around the point at which the jump function is being evaluated. Using the fact that the coefficients for the PA scheme satisfy Equation 3.5 the jump function approximation can be expressed as,

$$|L_m f(x)| = \left| \frac{1}{q_m(x)} \sum_{x_j \in \mathcal{S}_x} c_j(x) R_{m-1} f(x_j). \right|, \tag{3.9}$$

where $R_{m-1}f(x_j)$ is the difference between $f(x_j)$ and the Taylor series expansion of x_j around point x. If a jump (or extreme nonlinearity) exists at a particular point, a low order expansion around the point will not approximate the function well; the residual will be large and indicate the jump.

An analysis of the function $y(x) = \tanh(ax)$ allows the development of some insight into polynomial annihilation. The first scenario depicted in Figure 3-1 deals with evaluating the jump function in a relatively constant part of the curve at x = -0.5 with various points.

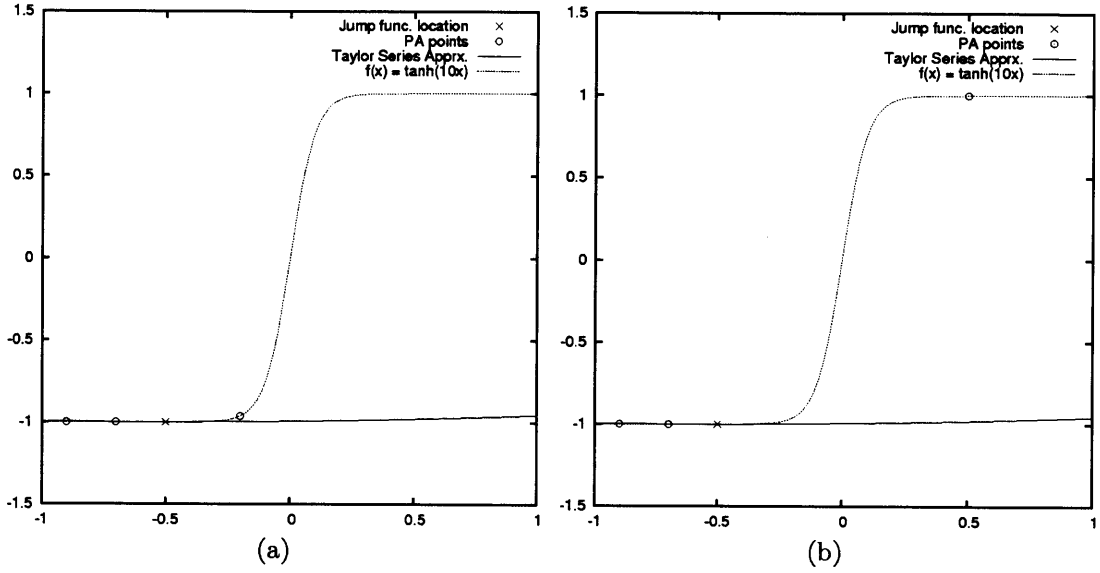


Figure 3-1: Taylor Series of hyperbolic tangent at various points.

Plot 1 Results		Plot 2 Results	
Points	Residual	Points	Residual
-0.2	3.58e-2	0.5	2.00e0
-0.7	-6.00e-5	-0.7	-6.00e-5
-0.9	-4.71 e-5	-0.9	-4.72e-5
$L_2 f(x)$	0.0360	$L_2 f(x)$	2.00

Table 3.1: Annihilating the zeroth and first order polynomials adequately captures an extreme nonlinearity surrounded by relatively constant regions

The second scenario depicted in Figure 3-1 indicates evaluating the jump function in between two neighboring function evaluations that are located on either side of a large functional change. In this scenario the jump function approximation accurately captures the size of this change as seen in Table 3.1.

In order to avoid misdiagnosing a steep slope as a discontinuity (due to low m) or oscillations as a discontinuity (due to high m) the minmod limiter approach similar to that in computational fluid dynamics is used. The jump function is evaluated using a set, M, of several annihilation orders:

$$MM\left(L_m f(x)\right) = \begin{cases} \min\limits_{m \in M} L_m f(x) & \text{if } L_m f(x) > 0 \text{ for all } m \in M, \\ \max\limits_{m \in M} L_m f(x) & \text{if } L_m f(x) < 0 \text{ for all } m \in M, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, all the various orders of polynomial annihilation approximations must agree in order for a jump function to be indicated at a test point.

3.2 Support vector machines

3.2.1 Formulation

After determining a mechanism for labeling points with polynomial annihilation, SVMs [7] [48] [55] are used to build a representation of the discontinuity existing in the full model. The support vector machine formulation stems from solving a Tikhonov regularization problem with hinge-loss. Suppose that we obtain the data $\mathcal{D}\{(x_i, y_i)\}$, the goal is to find a function f_s^λ such that:

$$f_s^\lambda = \text{argmin}_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^{n} V(f(x_i), y_i) + \lambda ||f||_{\mathcal{K}}^2 \right\} \tag{3.10}$$

where $V(f(x_i), y_i) = \max(0, 1 - y_i f(x_i))$ and $\mathcal{K}$ is the reproducing kernel Hilbert space on which our classifier f is defined [47] [18] . Setting a new variable $C = \frac{1}{2\lambda n}$ and reformulating the optimization statement results in:

$$f_s^\lambda = \text{argmin}_{f \in \mathcal{H}} \left\{ C \sum_{i=1}^{n} V(f(x_i), y_i) + \frac{1}{2} ||f||_{\mathcal{K}}^2 \right\} \tag{3.11}$$

This formulation makes it clear that C acts as a penalty on errors. In the sense that if C is large, the loss function will be weighted and the classifier will try to fit all of the data. If C is small then the regularization will be larger and errors will be penalized less. This formulation is non-differentiable because V is non-differentiable when $y_i f_i = 1$. Therefore, the problem is reformulated in terms of slack variables:

$$\begin{aligned} \underset{c\in\mathbb{R}^n,\xi\in\mathbb{R}^n}{\operatorname{argmin}} \quad & C\sum_{i=1}^{n}\xi_i + \frac{1}{2}c^T\mathbf{K}c \\ \text{subject to} \quad & \xi_i \geq 1 - y_i\sum_{j=1}^{n}K(x_i,x_j) \quad i = 1,...,n \\ & \xi_i \geq 0 \quad i = 1,...,n \end{aligned}$$

In order to solve this constrained problem, the Lagrangian and then dual formulations are formed. The Lagrangian is

$$L_p(c,\xi,\alpha,\mu) = C\sum_{i=1}^{N}\xi_i + \frac{1}{2}c^T\mathbf{K}c - \sum_{i=1}\alpha_i\left\{y_i\left(\sum_{j=1}^{n}c_jK(x_i,x_j)\right) - 1 + \xi_i\right\} - \sum_i \mu_i\xi_i \tag{3.12}$$

with KKT conditions,

$$\begin{aligned} \frac{\partial L_p}{\partial c_i} &= c_i - \alpha_i y_i = 0 \\ \frac{\partial L_p}{\partial \xi_i} &= C - \alpha_i - \mu_i = 0, \\ \xi_i \geq 0, \quad \alpha_i \geq 0, \quad \mu_i &\geq 0 \\ \alpha_i\left\{y_i(\sum_{j=1}^{n}c_jK(x_i,x_j)) - 1 + \xi_i\right\} &= 0 \\ \mu_i\xi_i &= 0 \end{aligned}$$

Transforming this into the dual problem results in a quadratic programming problem with linear equality and inequality constraints:

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \text{subject to} \quad & 0 < \alpha_i < C \\ & \sum_i \alpha_i y_i = 0 \end{aligned}$$

This dual formulation can be solved for α_i with standard quadratic programming tools or with specialized algorithms such as the Sequential Minimal Optimization (SMO) algorithm [41] [26]. These specialized algorithms take advantage of the fact that only a few α_i are non-zero. Specifically those α_i which correspond to support vectors x_i that are most informative in determing the discontinuity. After solving for the α_i, the classification of a new point can be evaluated as:

$$f_s^{\lambda}(x) = \sum_{i=1}^{N} \alpha_i K(x_i, x). \tag{3.13}$$

The classifier has the property that:

$$f_s^{\lambda}(x) \begin{cases} > 0 & \text{if } x \in R_1 \\ < 0 & \text{if } x \in R_2 \end{cases}, \tag{3.14}$$

where R_1 and R_2 are two regions separated by the zero level set of the classifier.

3.2.2 Stability and convergence

The goal of a useful solution is its generalization to inputs where there is no data. Following [55], the generalization properties of learning algorithms to new data is based upon expected risk of an algorithm defined as :

$$I[f_S] = \mathbb{E}_{(x,y)}\left[V(f_s(x), y)\right] = \int V(f_s(x), y)\pi(x, y)dxdy \tag{3.15}$$

This quantity is unobtainable because the distribution $\pi(x, y)$ is unknown. Therefore an empirical risk is defined as:

$$I_S[f_s] = \frac{1}{n}\sum_{i=1}^{n} V(f_S(x_i, y_i)) \tag{3.16}$$

The goal of our algorithm is then to obtain a tight generalization bound:

$$\mathbb{P}[I[f_S] - I_S[f_S] > \epsilon] \leq \delta \tag{3.17}$$

If we can show that the emperical risk is small, then we can show that the probability that the expected risk is small as well. In order to determine this generalization bound a procedure based on stability in [4] is followed. The concept of stability maintains that an algorithm is β-stable if the solution does not change much with varying data sets.

Theorem 3.2.1. *Let S denote a training set, $S^{i,z}$ denote a training set obtained by replacing the i-th example in S with a new point $z = (x, y)$. An algorithm $\mathcal{A}$ is β-stable if*

$$\forall (S, z) \in \mathcal{Z}^{n+1}, \forall i, sup_{z' \in Z} |V(f_s, z') - V(f_{s^{i,z}}| \leq \beta \tag{3.18}$$

□

If an algorithm is β stable then one can derive the generalization bounds using McDiarmid's Inequality:

Theorem 3.2.2. *Let $V_1, ..., V_n$ be random variables. If a function F mapping $V_1, ..., V_n$ to $\mathbb{R}$ satisfies*

$$sup_{v_1,...,v_n,v_i'} |F(v_1, ..., v_n) - F(v_1, ..., v_{i-1}, v_i', v_{i+1}, ..., v_n)| \leq c_i \tag{3.19}$$

then McDiarmid's inequality [35] is:

$$\mathbb{P}\left(|F(v_1, ..., v_n) - \mathbb{E}[F(v_1, ..., v_n)]| > \epsilon\right) \leq 2\exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^{n} c_i^2}\right). \tag{3.20}$$

□

The generalization bounds when $\beta = \frac{k}{n}$ for some k become

$$\mathbb{P}[|I[f_S] - I_S[f_S]| > \beta + \epsilon] \leq 2\exp\left(-\frac{n_\epsilon^2}{2(k+M)^2}\right), \tag{3.21}$$

or that with probability $1 - \delta$

$$I[f_S] \leq I_S[f_S] + \frac{k}{n} + 2(k+M)\sqrt{\frac{2\log(2/\delta)}{n}}, \tag{3.22}$$

Here we see the convergence with β is $\mathcal{O}(\beta) + \mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$. In these equations M is the upper bound on the loss. The analysis in [44] indicates that β and M for SVMs are

$$\begin{aligned} \beta &= \frac{L^2\kappa^2}{\lambda n} \\ M &= \kappa L\sqrt{\frac{C_0}{\lambda}} + C_0, \end{aligned}$$

where $L = 1$ is the Lipschitz constant for the Tikhonov problem with hinge loss (SVM) and κ bounds the kernel:

$$\sup_{x\in\Omega_X} K(x,x) \leq \kappa^2 < \infty \tag{3.23}$$

This analysis shows several theoretical features corresponding to SVMs. The first feature is that the bound on the loss scales like $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$ with fixed λ. An extraordinarily large λ will require a large amount of data in order to improve the bound. This is synonomous with the fact that the λ is a regularization parameter which weighs against the data in order to prevent overfitting. Secondly, if λ is too small, no generalization bounds can be stated because f_S will simply overfit and be perfect on all the data no matter how much data is obtained.

3.2.3 Implementation

Implementation of the SVM involves selecting a kernel. The most common kernels are:

- Polynomial: $K(x, y) = (x \cdot y + 1)^p$
- Gaussian: $K(x, y) = \exp\{-||x - y||^2/2\sigma^2\}$
- Sigmoid: $K(x, y) = \tanh(\kappa x \cdot y - \delta)$

In addition to the above kernels, any function that satisfies Mercer's condition is a valid kernel [36]. Each of the above kernels has parameters that must be tuned by the data or set a-priori. For example, in the polynomial kernel the order, p, and the width of the radial basis function σ in the Gaussian kernel. In addition to the kernel dependent parameters, the regularization parameter C, or λ, must be tuned as well. These are often tuned using K-fold cross-validation or leave-one-out cross validation. The computational cost of one optimization using the SMO algorithm is problem dependent but can be from $\mathcal{O}(N)$ to $\mathcal{O}(N^2)$ [41]. The entire procedure is performed various times for cross validation and thus is dependent on the methods for cross validation as well. In this thesis, the implementaion of SVMs used is found in [11].

Cross validation

K-fold cross validation and leave-one-out cross validation are two of the most popular techniques to select the regularization parameter for the Tikhonov regularization problem [29] [51]. K-fold cross validation works by randomly partitioning the available data into k data sets and training the SVM on $k - 1$ data sets followed by testing on the remining data set. This procedure is repeated k times, eventually leaving out every subset. The error may then be averaged among the k classifiers. This complete procedure is performed using various regularization parameters. An optimizer may be wrapped around the whole system in order to select a good regularizer.

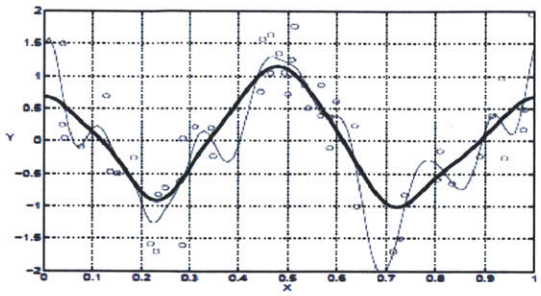


Figure 3-2: Regression example with empty circles indicating data points, thin black line indicating a fit with a weak regularization constant, and thick black line indicating the solution to the optimization problem with the same regularization constant but weak convergence tolerance.

A special case of K-fold cross validation is leave-one-out cross validation (LOOCV). In this procedure, cross-validation is performed on every training datum. This method for determining tuning parameters is computationally expensive and often unaffordable in practical situations. For this reason, the most common cross-validation procedures are 10- or 5-fold validation.

One method to avoid selecting a regularization term C in the Tikhonov problem is to stop the optimization solving the dual problem before it converges [60]. An example for the case of regression is shown in Figure 3-2 from [43]. Here, it is seen that a weak convergence tolerance on the optimizer plays the same role as a stronger regularization parameter.

3.3 Uncertainty sampling and active learning

Active learning [13] [46][50] and specifically uncertainty sampling [32] are commonly used methods in the learning community when labeling data points according to their region is an expensive process. They have been used in text classification where one has a library of documents, but classifying the documents involves the expensive task of a human reading the document. In order to efficiently perform the labeling of the documents, the human is required to only read those documents which a

classifier is uncertain about [54] [49]. Following the labeling of this new data point, a new classifier is built. The situation with building surrogates for complex models is different as there is no library of model evaluations. We are free to evaluate the model wherever we want; however, each evaluation is still expensive.

In these situations US is used to adaptively add data points to a data set and retrain a classifier after each addition. The data points are added where the previous classifier is most uncertain of the class of the point. In the context of SVMs this region corresponds to the zero level-set of the classifier. Furthur details regarding the implementation of an US algorithm are given in Chapter 4.

Chapter 4

Discontinuity detection algorithm

4.1 Overall methodology

This chapter describes a discontinuity detection algorithm that takes advantage of any regularity exhibited by a discontinuity in model outputs, avoids the creation of structured grids and rectangular subdomains, and incorporates guided random sampling to improve scaling. The regularity of the discontinuity is exploited by creating a smooth approximation of the boundary between domains using support vector machines (SVM). This approximation allows for a more efficient description of the discontinuity than a nearest neighbor approach used in the edge tracking and adaptive refinement schemes available in the literature. Additionally, SVMs are robust and in practice do not over-fit the data.

In order to build support vector machines, one must have a means of labeling data points on either side of the discontinuity based on the region they belong to. For this purpose a polynomial annihilation scheme based on [25] is used. This scheme allows the determination of the discontinuity size and subsequently allows the labeling of new model evaluations based on their function value. Finally, refinement of the discontinuity is performed using uncertainty sampling by optimizing randomly chosen points in the full domain to the subspace spanned by the zero level-set of the classifier. These three techniques form an algorithm that is shown to be more efficient than those found in the literature. Additionally, the algorithm is shown to default to an edge

tracking scheme in the cases where the discontinuity is irregular.

Several innovations have been made for the construction of this algorithm. The first innovation is the addition of an off-axial tolerance parameter when utilizing an axial based PA grid refinement technique such as in [2]. By allowing a wider range of points to be considered axial, the discontinuity may be refined in a more efficient manner. The second innovation is the use of optimization methods to drive randomly sampled points onto the zero level-set of the classifier. This optimization allows uncertainty sampling to proceed by providing a mechanism by which to sample the approximate boundary between regions.

4.2 Initializing with polynomial annihilation

The purpose of an initialization phase of the discontinuity detection algorithm is to provide a mechanism for labeling future data points near either side of a discontinuity based upon their function value. The initialization procedure is built on the basis of a divide and conquer approach guided by repeated application of one dimensional PA. The details of point selection and refinement are given in subsequent sections. The inputs and outputs of the initialization phase are provided in Table 4.1, and they allow the reader to understand what kind of information can be obtained from the discontinuity detection as well as the information required to perform it.

Inputs	Outputs
Initial Grid	Sparse set of points
Off-Axial Tolerance, *tol*	Function values at the sparse set
Edge Point Tolerance, δ	Locations of edge points
Maximum PA order	Jump function values at edge points
Desired number of edge points	

Table 4.1: Input/Outputs of the Discontinuity Detection Algorithm.

In the description of the algorithm that follows, the following five sets of points

are used repeatedly:

1. Set $S = \{\mathbf{x_i}$ s.t $f(\mathbf{x_i})$ has been evaluated.$\}$
2. Set $M = \{\mathbf{x_i}$ added to S in the last iteration.$\}$
3. Set $E = \{$Edge Points$\}$
4. Set $A = \{$Semi-axial points in direction j w.r.t point $y_i\}$. This set is a subset of S.

Now that the syntax is established, the selection of grid points for running PA at a given test point is described.

4.2.1 Stencil point selection

The core of the divide and conquer approach for PA will require creating a jump approximation at various test points based on a grid of already evaluated data points. Assuming that a grid of function evaluations exists, in order to select a stencil to use polynomial annihilation, the set of available points, set S, is first narrowed down to a set of semi-axial points, A. Semi-axial points are those that lie within a pre-defined off-axial tolerance. These points are considered axial for the purposes of applying the polynomial annihilation algorithm. This approximation is used to reduce the number of function evaluations necessary to evaluate the jump function. Intuitively, the off-axial tolerance indicates an accepted minimum resolution level of the discontinuity.

Figure 4-1 displays a typical scenario from which the field of axial points is chosen. The pink circle, referred to as POI, is the point to be tested for discontinuity, and the arrows denoted "+" and "-" refer to the relative directions of surrounding points. For the purposes of polynomial annihilation, at least one point in each direction is necessary. The boxes refer to all available points in the sparse grid. The horizontal direction is the axial direction in which PA is applied. The vertical direction refers to all non-axial directions. Two lightly shaded grey lines bound the region inside the accepted tolerance for points to be considered semi-axial. Only those boxes within the tolerance lines may be considered for the axial point selections, and those selected

Algorithm 1 Refinement Initialization

Input: Add initial points $\mathbf{x}$ to T and S and evaluate $f(x)$.
Add and evaluate all boundary points corresponding to points in T to S.
Copy points in T to M. Clear T.
for all M_i **do**
  for each dimension j **do**
    Refine1D(S,M_i,j,$edge_{lim}$)
    if $|E| \geq edge_{lim}$ **then**
      break
    end if
  end for
end for

Algorithm 2 Refine1D

Input: S, T, direction, j, $edge_{lim}$
if $|E| \geq edge_{lim}$ **then**
  break
end if
Spawn two points from T in direction j at which to evaluate the jump function, Mid_1 and Mid_2.
Evaluate Jump Function at each Mid_i.
if Jump function exists for point Mid_i **then**
  if Mid_i is an edge point **then**
    Add Mid_i to E.
    if $|E| \geq edge_{lim}$ **then**
      break
    end if
  else
    Add Mid_i to S.
    Add boundary parents of Mid_i to S.
    Add Mid_i to T_{new}.
    for each dimension j **do**
      Refine1D(S,T_{new},j,$edge_{lim}$)
    end for
  end if
end if

Algorithm 1 Refinement Initialization

1: **Input:** Add initial points **x** to T and S and evaluate $f(x)$.
2: Add and evaluate all boundary points corresponding to points in T to S.
3: Copy points in T to M. Clear T.
4: **for all** M_i **do**
5: **for** each dimension j **do**
6: Refine1D(S,M_i,j,$edge_{lim}$)
7: **if** $|E| \geq edge_{lim}$ **then**
8: break
9: **end if**
10: **end for**
11: **end for**

Algorithm 2 Refine1D

1: **Input:** S, T, direction, j, $edge_{lim}$
2: **if** $|E| \geq edge_{lim}$ **then**
3: break
4: **end if**
5: Spawn two points from T in direction j at which to evaluate the jump function, Mid_1 and Mid_2.
6: Evaluate Jump Function at each Mid_i.
7: **if** Jump function exists for point Mid_i **then**
8: **if** Mid_i is an edge point **then**
9: Add Mid_i to E.
10: **if** $|E| \geq edge_{lim}$ **then**
11: break
12: **end if**
13: **else**
14: Add Mid_i to S.
15: Add boundary parents of Mid_i to S.
16: Add Mid_i to T_{new}.
17: **for** each dimension j **do**
18: Refine1D(S,T_{new},j,$edge_{lim}$)
19: **end for**
20: **end if**
21: **end if**

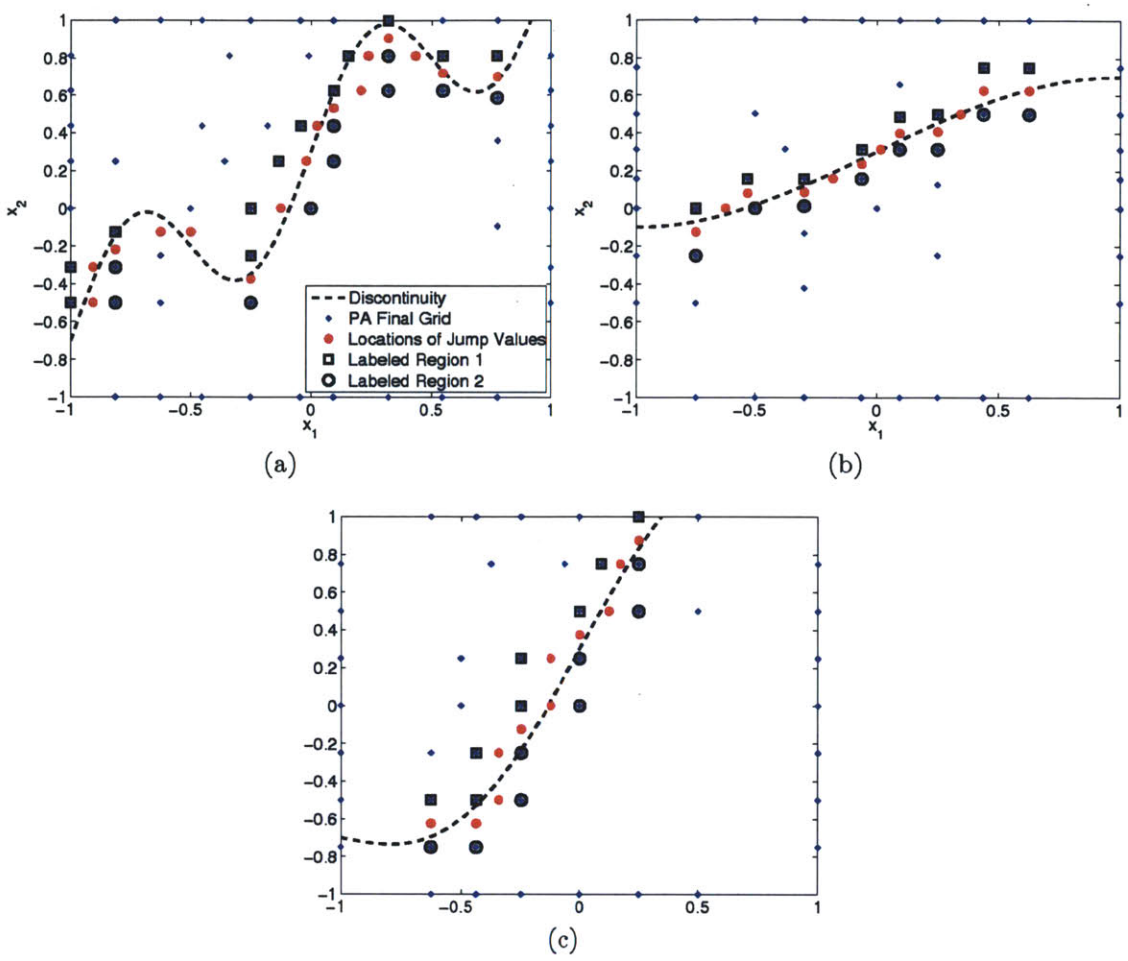


Figure 4-2: PA refinement algorithm applied to several discontinuities. The labels are created using the labeling procedure described in section 4.2.3.

points indicate locations at which the jump function is an estimate for the size of the discontinuity. The red points are then used in a labeling algorithm to label their surrounding function evaluations as residing in either region 1 or region 2.

4.2.3 Labeling

Now that a procedure for estimating the size of the discontinuity using PA has been described, these estimates must be used for labeling function evaluations resulting from discontinuity detection. The labeling methodology employed for determining which regions function evaluations from the PA procedure lie is dependent on function evaluations neighboring the edge points. Grid points within *tol* of the edge points are

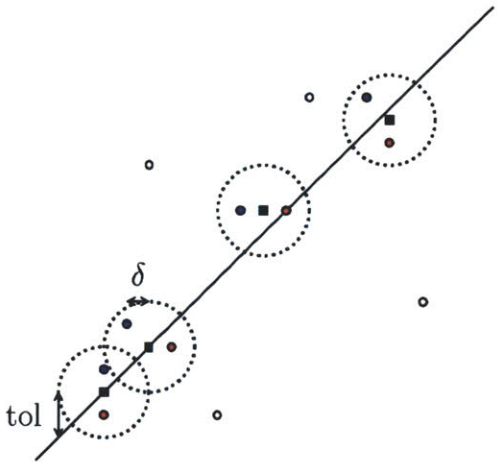


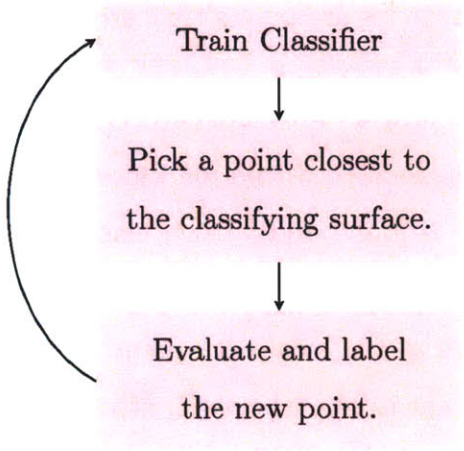
Figure 4-3: Labeling Algorithm: Circles are locations where the function have been evaluated. Squares are the location of the edge point. Blue circles are points that are labeled as class 1 and red circles are points that are labeled as class 2.

the only ones labeled and sent to the classifier. For each edge point, the algorithm finds a subset of grid points located within *tol*. Out of this subset, the grid point with the largest function value is found and labeled class 1. Then the function values at the other grid points in the subset are compared to the large function value using the value of the jump function at the local edge point. If the difference is less than the jump function then the point is labeled class 1; otherwise, it is labeled class 2. There is an implicit assumption in this algorithm that local large values are always of the same class and there is no cross over along the discontinuity. Figure 4-3 illustrates an example of this process.

4.3 Uncertainty sampling implementation

In order to improve the scalability and resolution of the discontinuity detection algorithm, an active learning technique based on evaluating points which the classifier is least sure of is used. This active learning technique offers a scalability improvement over higher resolution PA because of the ability to evaluate the model at points which are most informative for modifying the classifier. However, these points are sampled randomly along the classifier instead of optimizing such that the classified changes most because such a technique is computationally infeasible. These additional points

are chosen based on their proximity to the classifying surface. Each iteration of active learning proceeds by selecting a random point that is closest to the classifier. This point is then labeled by comparing its function value to the function values of the near edge points obtained from the PA scheme. Once the new point is labeled, a new classifier is built and the procedure is repeated. In practice no cross validation is performed for two reasons: the computational expense of performing cross validation at every iteration of US is too great, especially when the data set contains thousands of data points and because the algorithm is found to be fairly robust to the regularization parameter. A schematic of the procedure is shown below:



4.3.1 Generating new data points

A new function evaluation is obtained by sampling a point in the full domain and then driving it to the classifier by minimizing the squared classifier function:

$$g(x) = \left(\sum_{i=1}^{N_{sv}} \alpha_i K_i(x) \right)^2 \tag{4.1}$$

Gradient and non-gradient based algorithms have been used and result in similar performance. This optimization problem is not a quadratic program and may have local minimum. In practice, a clustering of points in local minima has not prevented the eventual refinement of the discontinuity. Additionally, low discrepancy points along the discontinuity are obtained by forcing new function evaluations to occur

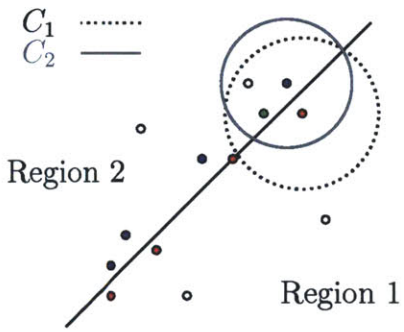


Figure 4-4: Uncertainty Sampling Labeling algorithm: A new test point (green) is classified based on a comparison of its function value to its nearest neighbors in region 1 and region 2.

farther than a minimum distance away from its nearest neighbors. Finally, because of the Lipschitz and nearest neighbor tolerances described in section 4.3.2, this algorithm defaults to edge tracking if performed for a sufficiently large number of iterations.

4.3.2 Labeling new data points

The procedure for labeling points near the discontinuity obtained from active learning are not labeled using the procedure described above. The reason being that there may not be jump approximations near to the new point. The procedure to label new points is similar with the exception that the nearest points to the the unclassified point in each region are located. In order to perform this procedure a Lipschitz constant must be specified for the model. The Lipschitz constant L is defined as:

$$|f(x_1) - f(x_2)| \leq L|x_1 - x_2|. \tag{4.2}$$

In other words, L is the maximum that a function can change given a change in location. In practice, this involves specifying a trust region from the nearest neighbors of an unclassified point to the unclassified point. If the unclassified point is not within the trust region of both nearest neighbors in each classes, then the unclassified point cannot be reliably classified and is ignored. The labeling procedure is depicted in Figure 4-4. Skipping the evaluation of new data points achieved through US means that new function evaluations will appear within a certain distance from previously

labeled points. However, as uncertainty sampling progresses, the entire discontinuity will be explored. This procedure provides a method for reliably labeling new function evaluations while no accurate approximation for the discontinuity exists. In practice if one believes that the size of discontinuity is far greater than the variability within each region, L, the viable labeling region may span the entire domain. Whereas if one believes there may be a lot of variability in the size of the discontinuity, the trust region around the labeled points will be smaller. The accurate approximation for the discontinuity is obtained only after the final iteration of uncertainty sampling.

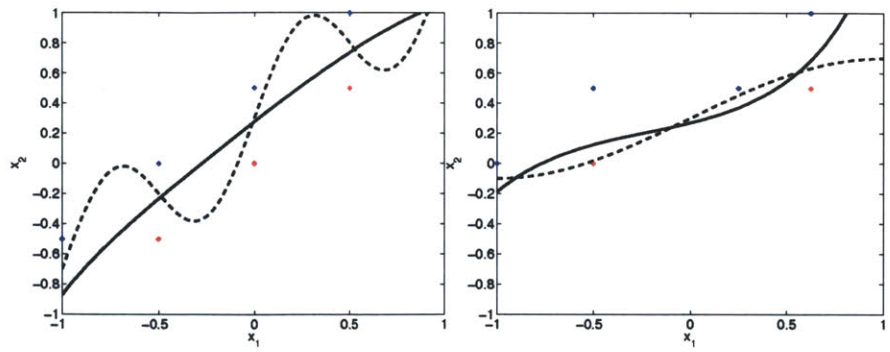
4.3.3 Examples of US

Figures 4-5 and 4-6 show the application of US to several different discontinuities. The equations for the discontinuities are:

$$\begin{aligned} x_2 &= 0.3 + 0.4\sin(2\pi x_1) + x_1 \\ x_2 &= 0.3 + 0.4\sin(\pi x_1) \\ x_2 &= 0.3 + 0.4\sin(\pi x_1) + x_1 \end{aligned}$$

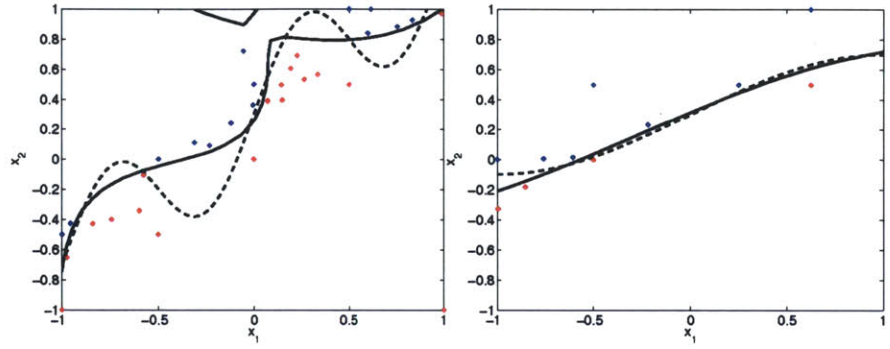
Gaussian kernels are used for the SVM, with $1/2\sigma^2 = 0.5$ and with $C = 10^{-4}$. Plots of the distribution of positively labeled points, negatively labeled points, and the current guess for the discontinuity are shown at various iterations of US. The first column contains a discontinuity that contains oscillations and requires the largest number of function evaluations in order to accurately create a classifier. An interesting characteristic of Gaussian kernels is visible in subfigures 4-5 (c) and (e) in that the SVM actually separates the space into several regions. However, this problem is resolved in subfigure (g) as the discontinuity is properly refined.

The second column contains a discontinuity that is almost linear, and it requires the fewest number of function evaluations to accurately capture. The third column contains a discontinuity that is fairly linear in a large region but contains a tail near the lower left hand corner. The US algorithm effectively locates this tail and accurately creates an approximation for the discontinuity.



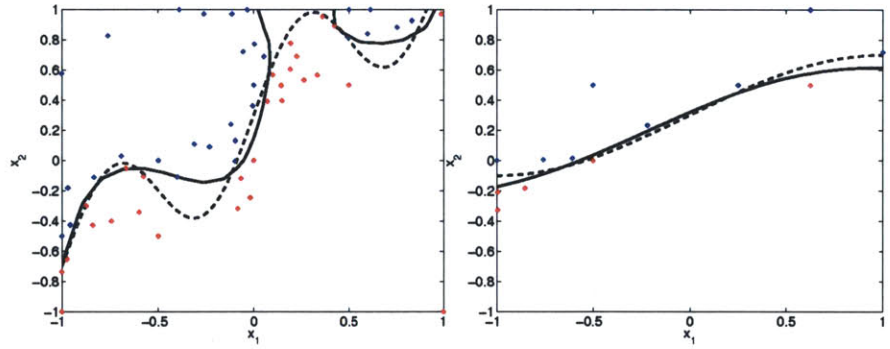
(a) Initial classifier and labeled points obtained from PA for the first test function.

(b) Initial classifier and labeled points obtained from PA for the second test function



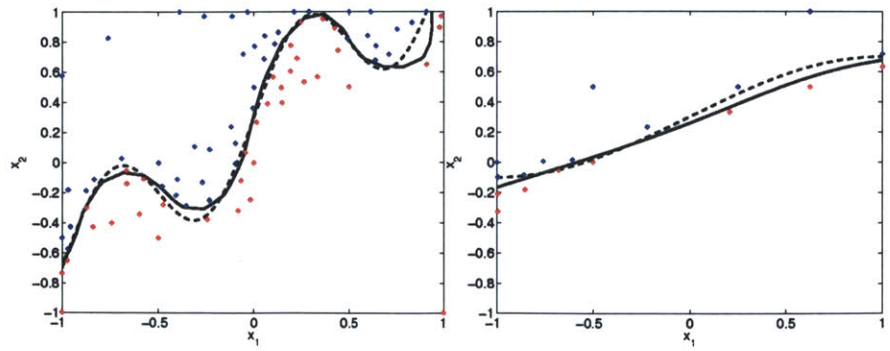
(c) Test function 1 after 25 iters.

(d) Test function 2 after 5 iters.



(e) Test function 1 after 50 iters.

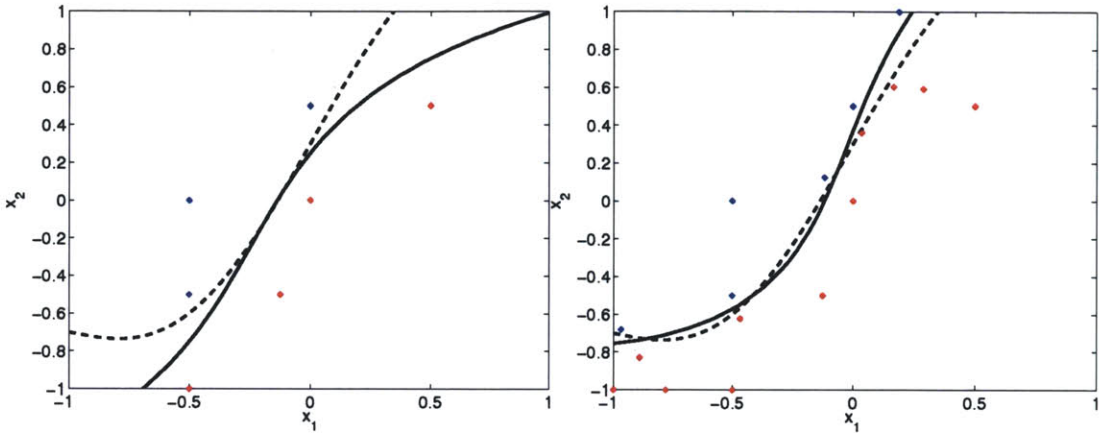
(f) Test function 2 after 10 iters.



(g) Test function 1 after 75 iters.

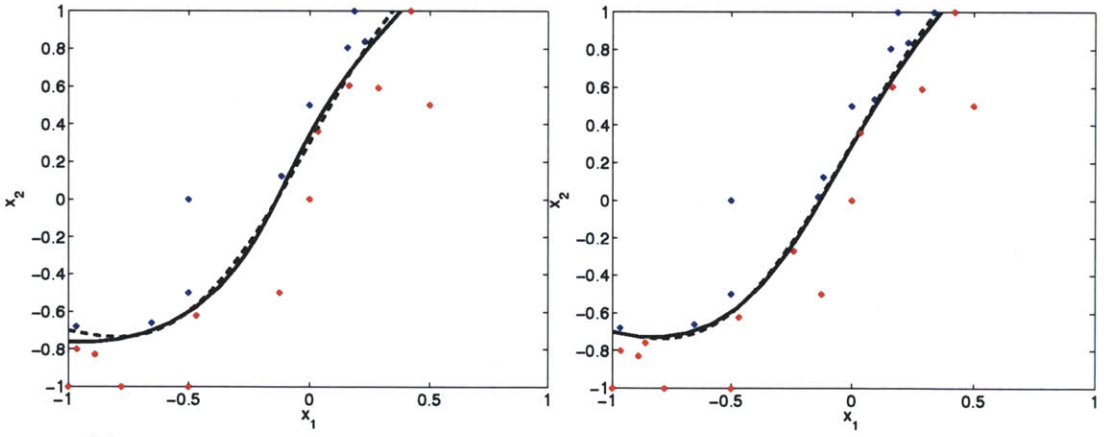
(h) Test function 2 after 15 iters.

Figure 4-5: Uncertainty sampling is performed on two different discontinuities



(a) Initial classifier and labeled points obtained from PA for the third test function

(b) Test function 3 after 10 iters.



(c) Test function 3 after 15 iters.

(d) Test function 3 after 20 iters.

Figure 4-6: Uncertainty sampling is performed on a third discontinuities

4.4 Results

Now that the algorithm has been fully described, results on several test problems are presented. The results presented illustrate the advantages of this method in terms of scaling to higher dimensions than those demonstrated in the literature. Section 4.4.1 demonstrates the scaling of the algorithm on a variable dimension test problem and section 4.4.2 compares this algorithm to several test problems found in the literature.

4.4.1 Dimension scaling

Consider a function $f(x)$, where $x = (x_1, ..., x_D)$ and $x \in [-1, 1]^D$:

$$f(x) = \begin{cases} x^2 + 10 & \text{if } x_D > \sum_{i=1}^{D-1} x_i^3 \\ x^2 - 10 & \text{otherwise} \end{cases} \tag{4.3}$$

This equation is a quadratic with a cubic discontinuity. In Figure 4-7 uncertainty sampling is performed by adding 10 function evaluations each iteration until 99% of 100000 randomly sampled points in the entire domain are classified correctly. A Gaussian kernel is used with $1/sigma^2 = 1/D$, and $C = 1e-4$. Uncertainty sampling is initially fed by a grid obtained from running PA until no new function evaluations can be added (due to tolerance levels). The trust region for labeling points obtained in US is taken to be an arbitrarily high number (1e6) because the variability in each subdomain is not as great as the size of the discontinuity. Figure 4-7 clearly indicates that the bottle-neck in this algorithm is polynomial annihilation. The performance of PA until a grid tolerance is reached is an expensive exponentially scaling algorithm. In order to reduce this computational expense, PA is only performed until a 15 labeled points are obtained. The results are displayed in Figure 4-8. Once the fixed number of edge points is obtained, US is performed as before. Since this algorithm is random, it is performed 100 times and the mean results are shown. The variance of the results becomes indistinguishable from the mean at higher dimensions, thus the variability in total number of function evaluations is not great. In this scenario the scaling of the algorithm appears to be several orders of magnitude improved. This suggests the

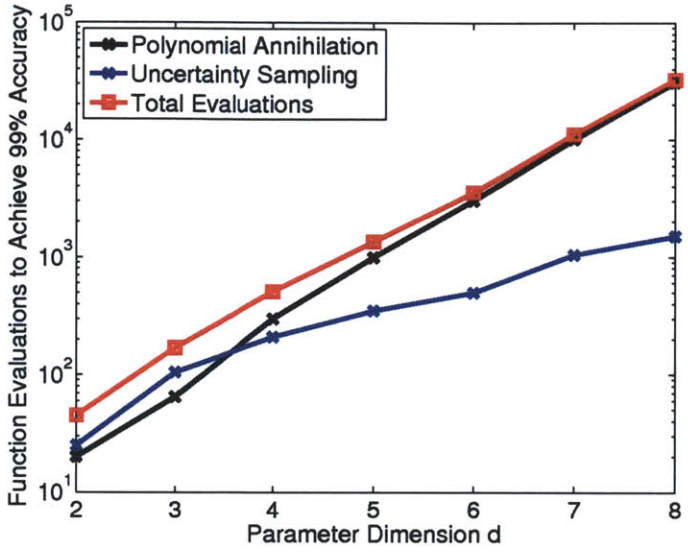


Figure 4-7: Scaling results when running PA to completion.

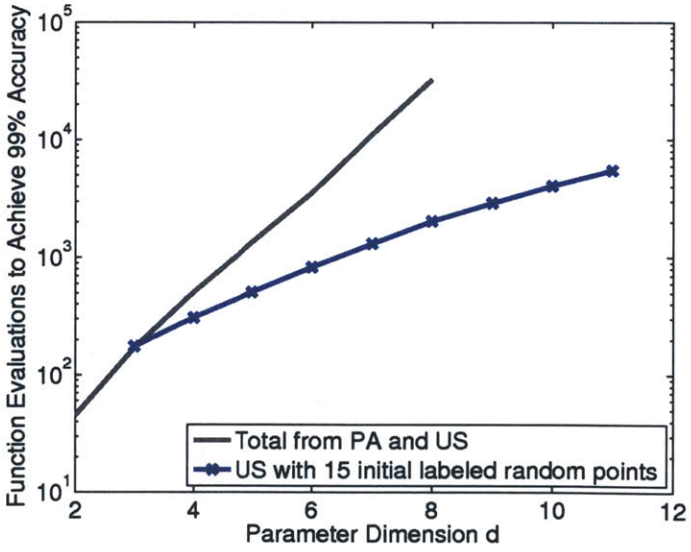


Figure 4-8: Scaling results when running PA to obtain a fixed number of edge points.

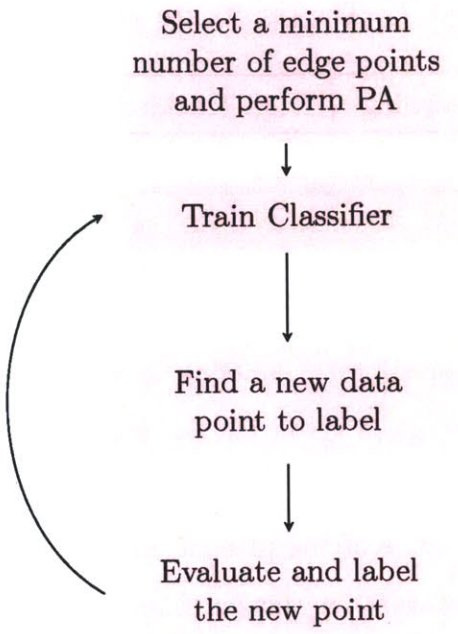


Figure 4-9: Best discontinuity detection algorithm configuration.

best way to perform discontinuity detection is given in Figure 4-9.

4.4.2 Examples and comparisons with other algorithms

Genetic Toggle Switch

A popular example of a bifurcating ode is the genetic toggle switch [22]. This ode is described below:

$$\frac{du}{dt} = \frac{\alpha_1}{1+\nu^\beta} - u \tag{4.4}$$

$$\frac{dv}{dt} = \frac{\alpha_2}{1+w^\gamma} - v \tag{4.5}$$

$$w = \frac{u}{(1+[IPTG]/K)^\eta} \tag{4.6}$$

For this example, $< p >= (\alpha_1, \alpha_2, \eta, K)$ are random variables that are uniformly distributed around their nominal values $\langle p_{nom} \rangle = \langle 156.25, 15.6, 2.0015, 2.9618 \times 10^{-5} \rangle$. The variable of interest y is distributed in the range $[-1, 1]^4$ and $\langle p \rangle = \langle p_{nom} - \sigma y \rangle$ where $\sigma = 0.1$. γ is set to 1, $[IPTG]$ is set to 4.0×10^{-5}, and β is set to 2.5, in the

same manner as in [25] and [1]. The results are shown in the table below:

	Learning	Edge Tracking [25]	Adaptive Refin. [1]
Model Evals	1500 (avg)	31,379	91,250

For this simulation a Gaussian kernel is used, $C = 10^{-4}$, and 15 edge points are sought. The trust region for US labeling is set to an arbitrarily high number. The model evaluation average is given for the learning algorithm described in this chapter because as it is a random algorithm, it is performed 100 times to describe performance. The performance of the present algorithm is much improved over the techniques found in the literature. In this particular example, this improvement is due to the fact that for a 1% classification error, the discontinuity may be described using a hyperplane as in [1]. Hyperplanes need very few points to be well approximated.

Discontinuity in subspace of full domain

Another extension of this algorithm can be demonstrated in the case where a discontinuity only exists in a subspace of the total dimension of the problem. Here we consider a 20 dimensional problem containing 3D sphere centered at the origin that has a radius of 0.125. The equation for this function is provided below:

$$f(x) = \begin{cases} 1 & \text{if } \sum_{i=1}^{3} x_i^2 < r^2 \\ -1 & \text{else} \end{cases}$$

For this example, the classification error will be evaluated at 1000 uniformly random points located within a distance of 0.125 of the discontinuity. This region is difficult to approximate well because if 1000 uniformly random points were generated in the full 20 dimensional domain, the classification error would be excessively low. In order to make a comparison of this algorithm to [25], US is performed until a 93% classification error is achieved. The same algorithm settings as the previous example are used in this example, with the exception that only one edge point in each dimension is sought,

starting with a grid of one point at the origin. The edge tracking results indicated $\mathcal{O}(10^4)$ required function evaluations. The learning algorithm presented here required 6 labeled points from PA and $\approx$ 300 US sampling iterations. Again, this improvement is due to the fact that a circle is a very regular discontinuity shape, and a Gaussian kernel SVM can approximate it quite easily.

4.5 Outstanding issues

The most prevalent outstanding issue regarding the algorithm presented is a stopping criteria. The first issue is a stopping criteria for PA in the form of a specification of the number of edge points. Some knowledge of the variability of the function on either side of the discontinuity may aid in setting this parameter. For example if it is known that the variability of the functions in the region separated by the discontinuity is low, one needs only to search for an edge point in each direction. If no edge point in a direction is found, then no refinement must occur in that direction. If an edge point is found, then a valid jump approximation is available for the entire domain. If however, there is a large variability in the function in each region, more edge points are necessary. No rigorous method is available for determining the precise number necessary.

For US, a stopping criteria based upon the SVM classifier contains several drawbacks. For example, if one uses a stopping criteria where US stops if the SVM classifier is not changing with iteration, one cannot reliably know that this occurs just because US sampling points are not sampling the correct regime of error. In practice, this issue becomes less important as often one simply wants to obtain the best surrogate possible in a given amount of time. Chapter 5 will discuss this approach, but it is based on feeding PA an initial grid with function evaluations spread throughout the domain, and running PA and US until more function evaluations are unaffordable.

In addition to stopping criteria, a remaining issue is a rigorous evaluation of the performance of this algorithm as a function of geometric complexity. Some initial results shown in this section indicated that complex geometries require large numbers

of function evaluations. Finally, an extension to more than one discontinuity must be pursued. One major issue in applying the presented algorithm to several discontinuities separating several domains is the development of a robust labeling scheme so that US may take place.

Chapter 5

Function Approximation

5.1 Methodology

Chapter 4 described an efficient method for locating discontinuities. The second half of the approximation problem involves building a surrogate for the computational model on each region of smooth behavior, separated by the discontinuity. The approximatoin method should be able to do the following:

- handle complex domain geometry
- employ model evaluations on random sets of points.

The first requirement stems from the fact that the subdomains separated by discontinuity need not be rectangular. Thus spectral methods are not the best fit for this problem. The reason spectral methods have issues is that they assume the input parameters are independent. Under the independence assumption, one can easily construct orthogonal polynomials based on the probability measure of the underlying variables. In the dependent case these polynomials would no longer be orthogonal and thus any methodology used to calculate the coefficients of the expansion will be ill conditioned. The authors in [45] have employed Rosenblatt mappings in these situations in order to transform the dependent variables into independent variables. These mappings have not been shown to scale well with dimension. Additionally, care must be taken to make sure these mappings themselves are not excessively nonlinear (high

order) and that they do not introduce new discontinuities. The method described in this chapter avoids these issues by using the GP regression mean function as the surrogate. This method does not suffer from ill conditioning because the surrogate is represented as a sum of radial basis functions. The radial basis functions do not require special treatment on irregular domains.

The second requirement stems from the fact that the resulting function evaluations resulting from discontinuity detection do notlie on a structured grid, but rather are concentrated along the discontinuity itself. In order to reduce the computational cost of building a surrogate, one would like to utilize these function evaluations. This requirement reduces the applicable approximation technology to regression or interpolation based methods. Again, GP regression fits this requirement and is thus used for approximation.

5.1.1 Background

The surrogate for the full model is a GP regression mean estimate in each subdomain. For the implementation of GPs, we use a squared exponential covariance kernel with a different correlation length (l_i) for each dimension. Additionally, we perform a maximum likelihood estimation for the parameters of the kernel including the correlation lengths and signal variance, σ_f^2. A nugget with a value of $\sigma_n = 10^{-5}$ is used to enhance the conditioning of the covariance matrix. The squared exponential kernel is given as:

$$k(x, x') = \sigma_f^2 \exp\left[\sum_{i=1}^{D} \frac{-(x_i - x_i')^2}{2l_i^2}\right] + \sigma_n^2 \delta(x, x'). \tag{5.1}$$

The prior mean function (see equation 1.11) is chosen to be a polynomial series:

$$\mu(x) = \sum_{i=1}^{P} a_i, \Phi_i(x) \tag{5.2}$$

a_i are determined the pseudospectral projection algorithm based on [14]. The order of the spectral approximation is low in order to obtain a smoothed out approximation of the true function instead of one that exhibits high frequency oscillations. Now

that the GP approximation method is described, Section 5.1.2 describes how GPs are utilized within the discontinuity detection framework.

5.1.2 Implementation

The implementation of GPs for approximation follows from the decomposition approach taken for discontinuity detection. Surrogates are built for each region separated from the discontinuity. The SVM classifier provides a method to gauge the confidence that a certain point lies in a certain region based on the classifier function. This characteristic is utilized by only constructing surrogates a certain distance away from the discontinuity. In the regions immediately bordering the discontinuity, the full model is evaluated. This technique follows from the thought that creating an accurate surrogate next to the discontinuity is extraordinarily expensive as often the approximation of the discontinuity will not be fully accurate. If surrogates are built in the regions immediately neighboring the discontinuity and their training points are wrongly classified, the surrogate will be inaccurate. Rather than expending a large computational effort on refining this area of the surrogate, the full model should simply be evaluated when necessary. As the dimension increases, the volume around the discontinuity progressively becomes a smaller fraction of the total volume and thus this methodology actually improves in efficiency. The final surrogate is of the form:

$$f(x) = \sum_{j=1}^{N_{reg}} I_j(x) \sum_{i}^{n_j} \alpha_{ij} \Psi_{ij}(x), \tag{5.3}$$

where N_{reg} is the number of regions the entire domain is decomposed into, n_j is the number of training samples in region j, and I_j is an indicator function. α_{ij} is the coefficient of the ith basis in the jth domain and Ψ_{ij} is the ith basis in the jth domain. However, when region j has a border which is the zero level set of the classifier, the full model is used for the evaluation of points.

5.1.3 Training point selection

When constructing the GP approximation, the function evaluations used for discontinuity detection will be reused for approximation construction. In order to retain accuracy in regions far away from this discontinuity, a Latin Hypercube (LHS) or Quasi-Monte Carlo (QMC) sampling scheme is used as the input to discontinuity detection. Not only does this provide a good coverage of the entire volume, but it offers many training points with which to start polynomial annihilation. In addition to the Latin Hypercube samples, a quadrature grid is initially laid out in order to generate a prior mean function using pseudospectral projection. Pseudospectral projection onto a low order basis (2-3) allows the mean function to be a smoothed out version of the discontinuous function. Specifically, the regions near the discontinuity will be smoothed out. However, these regions are exactly those in which function evaluations are concentrated during the US sampling process. Therefore, the GP surrogate retains the accuracy of the smooth approximation in regions away from the discontinuity, and increases the accuracy in regions near the discontinuity because of the distribution of the training data obtained from discontinuity detection.

5.2 Results

Two applications of this methodology are presented. The first application is on Burgers equation with uncertain initial conditions. This application contains a discontinuity that is of variable size ranging from 2 to 0.4. The second application is on a chemical kinetics problem with uncertain initial state variables.

5.2.1 Burgers' equation

The Burgers' equation under consideration is given below:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) = \frac{\partial}{\partial x}\left(\frac{\sin^2 x}{2}\right), \quad 0 \leq x \leq \pi, t > 0,$$

$$\text{Uncertain IC: } u(x,0) = \beta \sin x,\ \beta \sim \mathcal{U}(0,1). \quad \text{BC: } u(0,t) = u(\pi,t) = 0.$$

For the approximation a quadratic prior mean function is used and a total of 114 evaluations are necessary to construct the approximation. The results of the approximation are shown in Figure 5-1 and the GP variance and point-wise errors are shown in Figure 5-2. The surrogate is evaluated in 93% of the total area and this area is shown in Figure 5-3. The two solid black lines surround the discontinuity and provide boundaries for each surrogate. The function evaluations used for discontinuity detection and approximation are also shown. The blue dots indicate all the function evaluations that occurred and those surrounded by squares are used to construct the approximation. For reference, the blue dots surrounded by a circle are those that were labeled through the US process. These points are all clustered around the discontinuity.

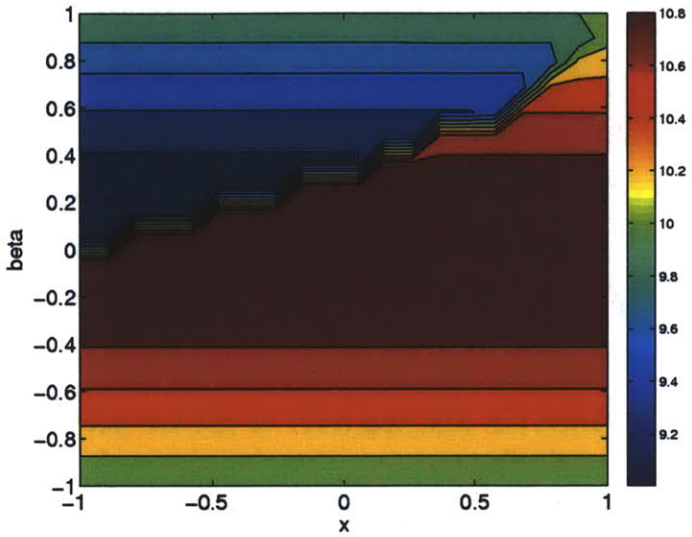
From the variance of the GP and the error of the GP plots we can see that the high error regions are not always represented by a high variance region. In fact, the region with highest variance does not correspond to the region with highest error. Overall, only 114 evaluations are necessary to achieve a 1.9% error in 93% of the area and the resulting surrogate does not contain spurious oscillations .

5.2.2 Combustion problem

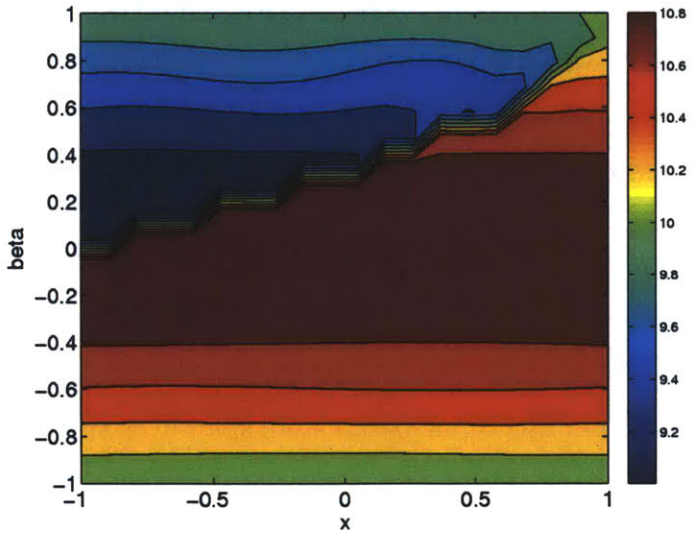
Finally, a 3-dimensional combustion problem is presented here. A homogeneous 0-D $H_2 - O_2$ reaction model [61] with nine species and nineteen reactions is used. A surrogate is created with with the initial temperature, initial pressure, stoichiometric ratio as input parameters. The input parameters span over the following range:

- $T_{init} = [1000, 1400]K$
- $P_{init} = [0.5, 1.5]$bar
- $\Phi_{stoich} = [0.5, 1.5]$

A surrogate is created for the temperature at 8×10^{-5} seconds into the reaction. The surrogate uses a second order prior mean function (7 function evaluations), 105 initial QMC function evaluations, 10 edge point tolerance for PA, and 100 US iterations.

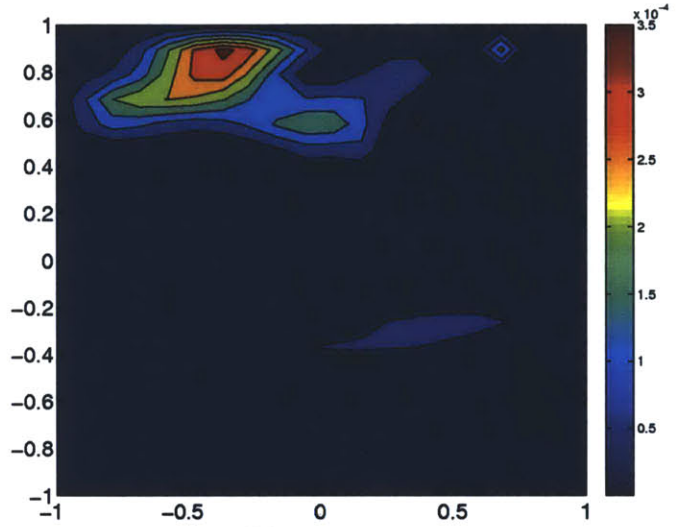


(a) True Function

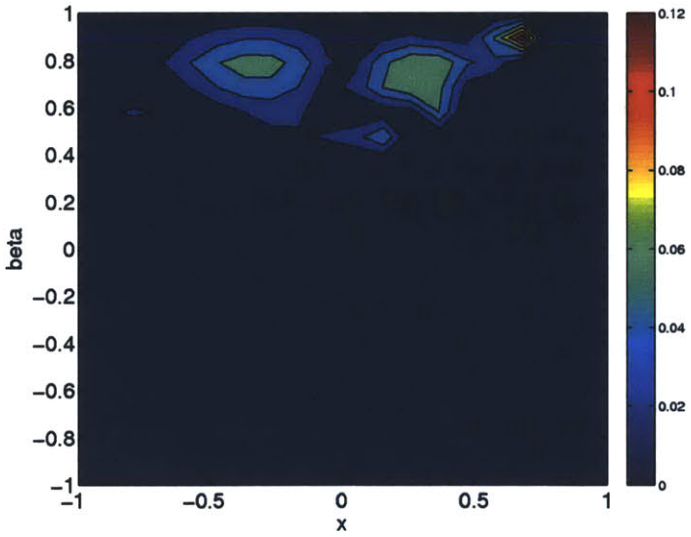


(b) GP mean

Figure 5-1: Surrogate for Burgers equation

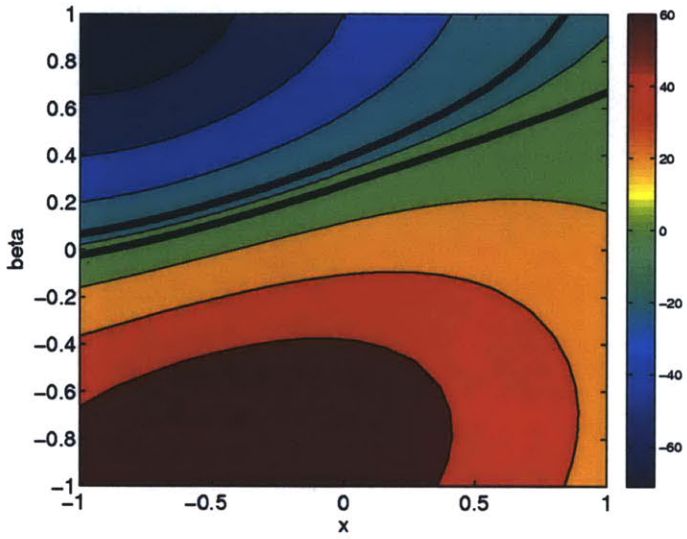


(a) GP variance

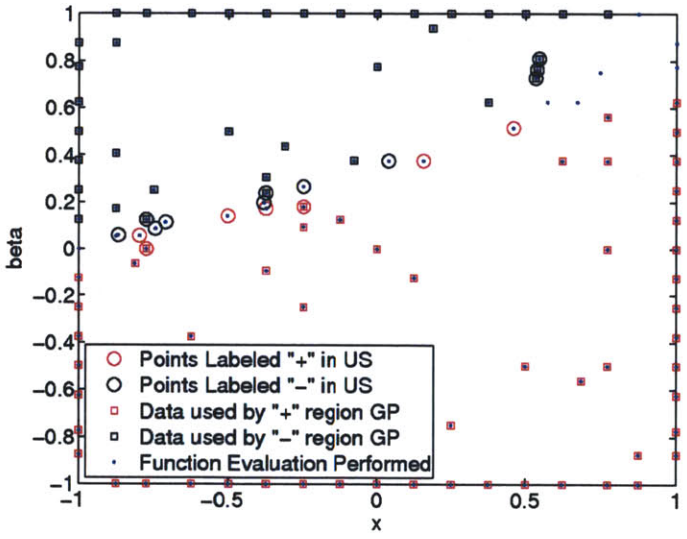


(b) Absolute Error, 1.9% L2 error

Figure 5-2: GP variance and absolute error of Burgers surrogate



(a) Classifier, middle region is 7% of the total area



(b) 114 total Function Evaluations

Figure 5-3: Burgers equation classifier and full model evaluations

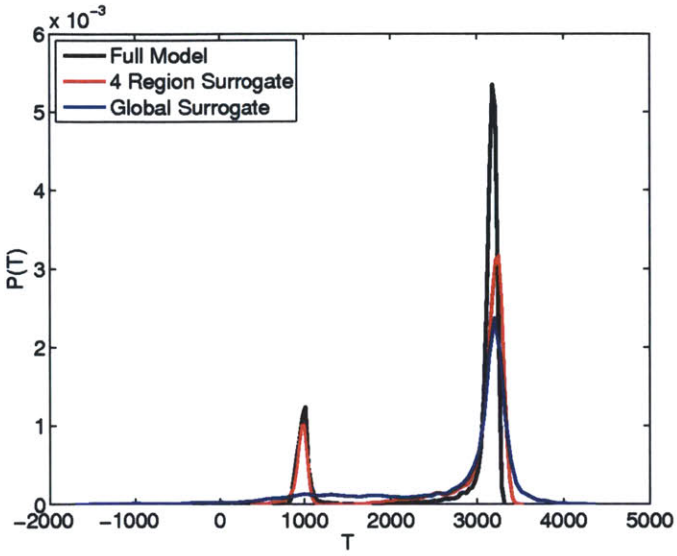


Figure 5-4: Output temperature pdf using a discontinuous surrogate and a global surrogate

For labeling in edge detection, a trust region of size 0.125 is used. 307 total function evaluations are necessary for surrogate construction. Once the surrogate is obtained, the region near the discontinuity where the full model is evaluated is set to be 2.5% of the total volume. For comparison with a global surrogate, a pseudospectral PCE is built using [14], with 339 function evaluations. The resulting error for the discontinuous surrogate is 6% and for the global surrogate is 14% measured on 10^4 full model evaluations.

In addition to creating the surrogate, uncertainty is propagated through the model to obtain distributions on the output temperature, assuming the input distributions are uniform. The output temperature is bimodal indicating whether or not ignition has taken place. The comparison in output probability distributions may be found in Figure 5-4.

This example has shown the utility of the discontinuous surrogate in practical applications involving complex model response. The global surrogate approach in the kinetics problem is wholly inappropriate and the discontinuous approach provides an efficient and accurate way to propagate uncertainty. The global surrogate smooths out the entire function and the second mode (due to non-ignition) is not visible in

the resulting approximation. The domain decomposition and interpolation approach locates the discontinuity and accurately builds an approximation for each region. As a result, both modes are evident in the output.

5.3 Outstanding issues

While the techniques discussed in this chapter have been successfully applied to the examples presented, several issues remain. The primary issue is how to best split the domain using the available function evaluations and classifier as guides. Several options exist including partitioning each region into subregions bounded by the contours of the surrogate. This type of partitioning would increase the efficiency of surrogate evaluation. However, the classifier contours may not be the best boundaries for subregions as they really do not take into account the underlying model.

Additionally, and perhaps more pressing, is the question of the distribution of the function evaluations among the various sub-algorithms that make up the entire method. In the experience of the author, US has been shown to be more effective than PA at refining the discontinuity and thus more function evaluations should be saved for US. Alternatively, if there is significant variability in the regions separated by the surrogate, more function evaluations may be necessary in the initial QMC point set.

Chapter 6

Conclusion

6.1 Summary

This thesis has described a framework aimed at computing surrogates for models exhibiting discontinuities. The overall guiding methodology for constructing this algorithm is two-fold: take advantage of discontinuity smoothness in the parameter space and reduce the excessive expense of accurately approximating the region near the discontinuity by building surrogates a minimum distance away from classifier boundary.

After the tools used for the methodology were described in chapter 3, the scaling potential of the discontinuity detection algorithm was displayed in chapter 4. In particular, the scaling results indicated that an edge tracking grid along the entire discontinuity is not necessary and SVM approximations are effective at representing boundaries between regions. The superior performance of discontinuity detection was shown against the state of the art domain decomposition and edge tracking schemes.

In regards to surrogate construction, GPs were shown to be effective at approximating the true model in regions away from the discontinuity. These methods are geometry independent and require low-discrepancy points to be effective. The methodology was applied to Burgers' equation and a chemical kinetics equation. Burgers' equation contains a shock and the methodology approximated 93% of the region to 2% error. Additionally, the approach has been shown to be effective for chemical

kinetics models where a discontinuity occurs due to ignition of the reaction. The approach described in this thesis was shown to accurately capture the bimodality exhibited due to this discontinuity.

6.2 Contributions

The contribution of this thesis has been the creation of a flexible algorithm for creating surrogates for discontinuous models. Though the motivation for surrogate creation has been presented as uncertainty quantification, these surrogate are generally applicable whenever computational expense must be reduced. The algorithm presented is more efficient than those in the literature because of its ability to take advantage of the regularity of a discontinuity. Additionally, the algorithm becomes an edge tracking scheme in the scenario that discontinuity regularity does not exist.

6.3 Outstanding issues

There are remaining issues pertaining to the general use of the algorithm. The primary issue remaining is the difficulty in selecting the required number of labeled function evaluations desired after polynomial annihilation. This choice is particularly difficult to make if the variability of the function in each region is substantial with respect to the size of the discontinuity. If this is not the case then obtaining one edge point in each dimension of the discontinuity suffices.

The second major issue with discontinuity detection is a lack of stopping criteria or error estimate for uncertainty sampling. The author does not have a solution to this issue, but its concern diminishes when one can only afford to run a certain number of iterations. However, even when one decides how many function evaluations are affordable, the decision as to how to spread them between US points and initial PA points is still unclear.

6.4 Future directions

Besides working on the issues described above, one extension of the algorithm presented here is the generalization of the methodology to multiple discontinuities separating multiple regions. Support vector machines for multi-class problems exist and would transfer well to this new scenario. The difficulty involved in dealing with multiple regions is the labeling of function evaluations for PA and US. This becomes difficult because it is no longer possible to characterize a function as belonging to a region with high values or low values. Another extension to the above framework can involve an adaptive approach where one alternates between discontinuity detection and surrogate construction in order to work towards some sort of convergence.

THIS PAGE INTENTIONALLY LEFT BLANK

Bibliography

[1] R. Archibald, A. Gelb, R. Saxena, and D. Xiu. Discontinuity detection in multivariate space for stochastic simulations. Journal of Computational Physics, 228(7):2676 – 2689, 2009. 26, 27, 62

[2] R. Archibald, A. Gelb, and J. Yoon. Polynomial fitting for edge detection in irregularly sampled signals and images. SIAM Journal on Numerical Analysis, 43(1):259 – 279, 2006. 27, 36, 37, 48

[3] Volker Barthelmann, Erich Novak, and Klaus Ritter. High dimensional polynomial interpolation on sparse grids. Advances in Computational Mathematics, 12:273–288, 2000. 19

[4] O. Bousquet and A. Elisseeff. Stability and generalization. Journal of Machine Learning Research, pages 499–526, 2002. 42

[5] Alfred M. Bruckstein, David L. Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Rev., 51(1):34–81, February 2009. 20

[6] H. Bungartz and M. Griebel. Sparse grids. Acta Numerica, 13:147–269, 2004. 19

[7] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2:121–167, 1998. 39

[8] E.J Candes. The restricted isometry property and its implications for compressed sensing. Comptes Rendus Mathematique, 346:589–592, 2008. 20

[9] E.J. Candes, J.K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. Communications on Pure and Applied Mathematics, 59:1207–1223, 2006. 20

[10] E.J. Candes and M.B. Wakin. An introduction to compressive sampling. Signal Processing Magazine, IEEE, 25(2):21 –30, march 2008. 20

[11] C. Chang and C. Lin. Libsvm: A library for support vector machines. ACM Trans. Intell. Syst. Technol., 2(3):27:1–27:27, May 2011. 44

[12] T. Chantrasmi, A. Doostan, and G. Iaccarino. Padé-legendre approximants for uncertainty analysis with discontinuous response surfaces. Journal of Computational Physics, 228(19):7159 – 7180, 2009. 25, 32

[13] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. Machine Learning, 15:201–221, 1994. 45

[14] P. G. Constantine, M. S. Eldred, and E. T. Phipps. Sparse Pseudospectral Approximation Method. ArXiv e-prints, September 2011. 66, 73

[15] S. L. Cotter, M. Dashtim, and A. M. Stuart. Approximation of bayesian inverse problems. SIAM Journal of Numerical Analysis, 48(1):322–356, 2010. 17

[16] T. Crestaux, O. P. Le Maitre, and J Martinez. Polynomial chaos expansion for sensitivity analysis. Reliability Engineering & System Safety, 94(7):1161 – 1172, 2009. 17

[17] I. Daubechies. Orthonormal bases of compactly supported wavelets. Communications on Pure and Applied Mathematics, 41:909–996, 1988. 30

[18] E. De Vito, L. Rosasco, A. Caponnetto, M. Piana, and A. Verri. Some properties of regularized kernel methods. Journal of Machine Learning Research, 5:1363–1390, 2004. 39

[19] D.L. Donoho. Compressed sensing. IEEE Transactions on Information Theory, 52:1289–1306, 2006. 20

[20] A. Doostan and H. Owhadi. A non-adapted sparse approximation of pdes with stochastic inputs. Journal of Computational Physics, 230(8):3015 – 3034, 2011. 20

[21] M.S. Eldred and J. Burkardt. Comparison of non-intrusive polynomial chaos and stochastic collocation methods for uncertainty quantification. In Proceedings of the 47th AIAA Aerospace Sciences Meeting, 2009. 21

[22] T. S. Gardner, C. R. Cantor, and J.J. Collins. Construction of a genetic toggle switch in escherichia coli. Nature, 403:339–342, 2000. 61

[23] T. Gerstner and M. Griebel. Dimension-adaptive tensor-product quadrature. Computing, 71:65–87, 2003. 19

[24] R. Ghanem and P. Spanos. Stochastic Finite Elements: A spectral Approach. Springer-Verlag New York, Inc., 1991. 16

[25] J. D. Jakeman, R. Archibald, and D. Xiu. Characterization of discontinuities in high-dimensional stochastic problems on adaptive sparse grids. Journal of Computational Physics, 230(10):3977 – 3997, 2011. 29, 47, 62

[26] T Joachims. Making large-scale svm learning practical. Advances in Kernel Methods Support Vector Learning, pages 169–184, 1999. 41

[27] M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 63(3):425–464, 2001. 15

[28] N Kingsbury. Image processing with complex wavelets. Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, 357(1760):2543–2560, 1999. 30

[29] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the 14th international joint conference on Artificial Intelligence, pages 1137–1143, 1995. 44

[30] O.P. Le Maitre, H.N. Najm, R.G. Ghanem, and O.M. Knio. Multi-resolution analysis of wiener-type uncertainty propagation schemes. Journal of Computational Physics, 197(2):502 - 531, 2004. 25, 30

[31] O.P Le Matre, O.M Knio, H.N Najm, and R.G. Ghanem. Uncertainty propagation using wienerhaar expansions. Journal of Computational Physics, 197(1):28 - 57, 2004. 25

[32] D. D. Lewis and J Catlett. Heterogeneous uncertainty sampling for superviced learning, pages 148–156. Morgan Kaufmann, 1994. 45

[33] Y. Marzouk and D. Xiu. A stochastic collocation approach to bayesian inference in inverse problems. Communications in Computational Physics, 6(4):826–847, 2009. 17

[34] Y. M. Marzouk, H. N. Najm, and L. A. Rahn. Stochastic spectral methods for efficient bayesian solution of inverse problems. Journal of Computational Physics, 224(2):560 - 586, 2007. 17

[35] C. McDiarmid. On the method of bounded differences. Surveys in combinatorics, 1989. 42

[36] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. Philosphical Transactions of the Royal Society of London. Series A., 209:415–446, 1909. 44

[37] Y. Meyer. Wavelets - Algorithms and applications. Society for Industrial and Applied Mathematics, 1993. 30

[38] H. Najm. Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. Annual Review of Fluid Mechanics, 41(1):35–52, 2009. 16

[39] A. O'Hagan and J.F.C. Kingman. Curve fitting and optimal design for prediction. Journal of the Royal Statistical Society. Series B, 40(1):1–42, 1978. 16

[40] T O'Hagan. Dicing with the unknown. Significance, 1(3):132–133, 2004. 15

[41] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C.J.C Burges, and A. J. Smola, editors, Advances in Kernel Methods – Support Vector Learning. MIT Press, 1998. 41, 44

[42] C. Rasmussen. Gaussian process in machine learning. In O. Bousquet, U. von Luxburg, and G. Ratsch, editors, Advanced Lectures on Machine Learning, volume 3176 of Lecture Notes in Computer Science, pages 43–71. Springer Berlin /Heidelberg, 2004. 16

[43] L. Rosasco and T. Poggio. Regularization via spectral filtering, 2012. 45

[44] L. Rosasco and T. Poggio. Stability of tikhonov regularization, 2012. 43

[45] K. Sargsyan, C. Safta, B. Debusschere, and H. Najm. Uncertainty quantification in the presence of limited climate model data with discontinuities. In IEEE International Conference on Data Mining Workshops, 2009., pages 241 –247, dec. 2009. 65

[46] G. Schohn and D. Cohn. Less is More: Active Learning with Support Vector Machines, volume 282, pages 839–846. Morgan Kaufmann, San Francisco, CA, 2000. 45

[47] B. Schökopf, R. Herbrich, and A. Smola. A generalized representer theorem. In D. Helmbold and B Williamson, editors, Computational Learning Theory, Lecture Notes in Computer Science, pages 416–426. Springer Berlin / Heidelberg, 2001. 39

[48] B. Schölkopf and A.J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, 2002. 39

[49] Fabrizio Sebastiani. Machine learning in automated text categorization. ACM Comput. Surv., 34:1–47, 2002. 46

[50] Burr Settles. Active learning literature survey. SciencesNew York, 15(2):201221, 2010. 45

[51] J Shao. Linear model selection by cross-validation. Journal of the American Statistical Association, 88(422):486–494, 1993. 44

[52] M.L. Stein. Interpolation of Spatial Data: Some Theory for Kriging. Springer Series in Statistics. Springer, 1999. 16

[53] B. Sudret. Global sensitivity analysis using polynomial chaos expansions. Reliability Engineering & System Safety, 93(7):964 - 979, 2008. 17

[54] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. Journal of Machine Learning Research, 2:45–66, 2002. 46

[55] V. N. Vapnik. The Nature of Statistical Learning Theory. Springer, 1995. 39, 41

[56] X. Wan and G. Karniadakis. An adaptive multi-element generalized polynomial chaos method for stochastic differential equations. Journal of Computational Physics, 209(2):617 – 642, 2005. 25, 26, 27

[57] D. Xiu. Fast numerical methods for stochastic computations: A review. Communications in Computational Physics, 2009. 19

[58] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. SIAM J. Sci. Comput., 27(3):1118–1139, October 2005. 20

[59] D. Xiu and G. Karniadakis. The wiener-askey polynomial chaos for stochastic differential equations. SIAM Journal on Scientific Computing, 24(2):619–644, 2002. 16, 18

[60] Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. Constructive Approximation, 2007. 45

[61] R. A. Yetter, F. L. Dryer, and H. A. Rabitz. A comprehensive reaction mechanism for carbon monoxide/hydrogen/oxygen kinetics. Combustion Science and Technology, 79:97–128, 1991. 69