Actively Learning Specific Function Properties with Applications to Statistical Inference

Brent Bryan

December 2007
CMU-ML-07-122

Carnegie Mellon

Actively Learning Specific Function Properties with Applications to Statistical Inference

Brent Bryan

December 2007
CMU-ML-07-122

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Jeff Schneider, Chair
Christopher R. Genovese
Andrew W. Moore (Google Pittsburgh)
Christopher J. Miller (NOAO / CTIO)
Robert C. Nichol (University of Portsmouth)
Larry Wasserman

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by the National Science Foundation under grant no. ACI-0121671, Department of Energy subcontract via University of California under grant number B563433, Science Applications International Corporation under grant no. 4400134507, and Air Force Research Laboratory under grant no. FA865005C7264.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Active Learning, Statistical Methods, Astronomy, Cosmology

For my parents.

Abstract

Active learning techniques have previously been shown to be extremely effective for learning a target function over an entire parameter space based on a limited set of observations. However, in many cases, only a specific property of the target function needs to be learned. For instance, when discovering the boundary of a region — such as the locations in which the wireless network strength is above some operable level, — we are interested in learning only the level-set of the target function. While techniques that learn the entire target function over the parameter space can be used to detection specific properties of the target function (e.g. level-sets), methods that learn only the required properties can be significantly more efficient, especially as the dimensionality of the parameter space increases.

These active learning algorithms have a natural application in many statistical inference techniques. For example, given a set of data and a physical model of the data, which is a function of several variables, a scientist is often interested in determining the ranges of the variables which are statistically supported by the data. We show that many frequentist statistical inference techniques can be reduced to a level-set detection problem or similar search of a property of the target function , and hence benefit from active learning algorithms which target specific properties. Using these active learning algorithms significantly decreases the number of experiments required to accurately detect the boundaries of the desired $1 - \alpha$ confidence regions. Moreover, since computing the model of the data given the input parameters may be expensive (either computationally, or monetarily), such algorithms can facilitate analyses that were previously infeasible.

We demonstrate the use of several statistical inference techniques combined with active learning algorithms on several cosmological data sets. The data sets vary in the dimensionality of the input parameters from two to eight. We show that naive algorithms, such as random sampling or grid based methods, are computationally infeasible for the higher dimensional data sets. However, our active learning techniques can efficiently detect the desired $1-\alpha$ confidence regions. Moreover, the use of frequentist inference techniques allows us to easily perform additional inquiries, such as hypothetical restrictions on the parameters and joint analyses of all the cosmological data sets, with only a small number of additional experiments.

Acknowledgments

There are are great many people that have inspired and aided me along the Ph.D. quest. I am extremely grateful for the guidance and support of my advisor Jeff Schneider. Besides providing intriguing ideas and insightful criticisms, Jeff patiently allowed me to follow my own hair-brain ideas, even when the prospects were poor. I would also like to thank the rest of my thesis committee: Andrew Moore, Bob Nichol, Chris Miller, Chris Genovese and Larry Wasserman for working with me on our joint research efforts, as well as critiquing our publications. Thanks also to Diane Stidle for organizing and arranging the impossible may times over.

Additionally, I'd like to thank my wife and children for showing me what is important in life and to my friends who provided temporary escapes from the grad school life. Thanks to Kate Bracher, Andrea Dobson, U.J. Sofia, Russ Gordon, and Albert Schueller for guidance through my undergraduate tenure. Thanks to all that increased my happiness and sustained my sanity.

Finally, I would like to thank my parents for providing guidance and inspiration to pursue a Ph.D. Despite times when my goals seemed less than clear and my career opportunities looked dim, my parents remained supportive of my decisions, even going as far as reading my dissertation. For this, and so much more, they deserve my love and gratitude.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In many scientific and engineering problems where one is modeling some function over an experimental space, one is not necessarily interested in the precise value of the function over an entire region or even the point which maximizes the function. Rather, one is curious about determining the set of all points for which the function exceeds some particular value. For instance, for the task of determining the functional range of wireless networks, it is important to be able to predict where the signal strength of the network drops below a given threshold [Ramakrishnan et al., 2005]. However, it is not necessarily important to be able to predict signal strength over the entire region. While being able to make a prediction over the entire space would enable us to determine the extent of the network, it will indubitably require extensive sampling. Instead we wish to focus on predicting only a specified subsets of a target function. This problem naturally arises in a number of tasks including factory optimization analysis, gauging the extent of environmental regions in geostatistics [Stein et al., 1999], and statistical inference. In this thesis, we will focus on latter problem.

Scientific inquest often requires the ability to compare hypothetical models with observed data to assess the models' validity. When the model is a function of several unknown parameters, we are interested in finding combinations of parameter values which fit the observed data with some statistical precision. While techniques for finding point estimates (e.g. maximum likelihood) for the parameters are available, here we consider computing confidence intervals (or regions, when we take the parameters as an ensemble), as they provide a critical view into the range of parameter values that are reasonable fits to the data. $1 - \alpha$ confidence regions are especially useful for statistical inference in many scientific disciplines as complex multi-parameter models are common. In astronomy, confidence regions can be used to determine ranges for the age, composition and eventual fate

of the universe [Tegmark et al., 2001, Spergel et al., 2007, Bryan et al., 2007b]. Similarly, biologists could use this method to compute properties such as the ranges for stroke kinematics parameters, wing muscle force, and Ca^{2+} presence in fruit fly flights [Dickinson et al., 1998, Lehmann and Dickinson, 1998, Gordon and Dickinson, 2006]. Finally, physicists are interested in determining the ranges of parameters which negatively affect explosions which cannot reasonably be tested (due to treaties or possible human health side-effects) [Higdon et al., 2005].

In this thesis, we describe active learning algorithms which efficiently sample the given parameter space in search of the $1 - \alpha$ confidence regions. There are many ways that machine learning techniques can be used to improve sampling efficiency in this domain. The ideas that we propose, revolve around one central theme: learning only that portion of the underlying function that is needed for the task at hand. As we shall see in the subsequent chapters, it is often enough to learn a target function when it is below a specified value, or even just learn a specific level-set of the target function. In either case, learning on a small portion is dramatically faster than trying to learn the entire target function.

1.1 Motivation

The motivation of this work is the computation of a set of core parameter values for cosmological models for our Universe. These parameters are unknown values that come from theoretical models of the Universe and correspond to physical properties, such as the amount of Baryonic (e.g. dirt-, plant-, human-like) matter in the Universe, as well as more exotic quantities, like dark energy (a large-scale repulsive force). In order to test these cosmological models and determine the values of the associated parameters, astronomers have compiled several independent data sets of different astronomical phenomena. One specific aim of this thesis is to use these data sets to accurately and efficiently determine the allowed ranges of each cosmological parameter.

To compute meaningful values of these cosmological parameters, we will use statistical inference techniques. There are many different statistical methods that can be employed. However, in general, the techniques which give us the most information are also those that require the most computational effort. Thus, we will need to develop algorithms and approximations which will allow us to efficiently compute the necessary statistical inferences that yield the tightest estimates of the cosmological parameters in which we are interested.

Therefore, the contributions of this thesis are four-fold. First, we develop general purpose active learning algorithms. These algorithms utilize well-motivated heuristics to

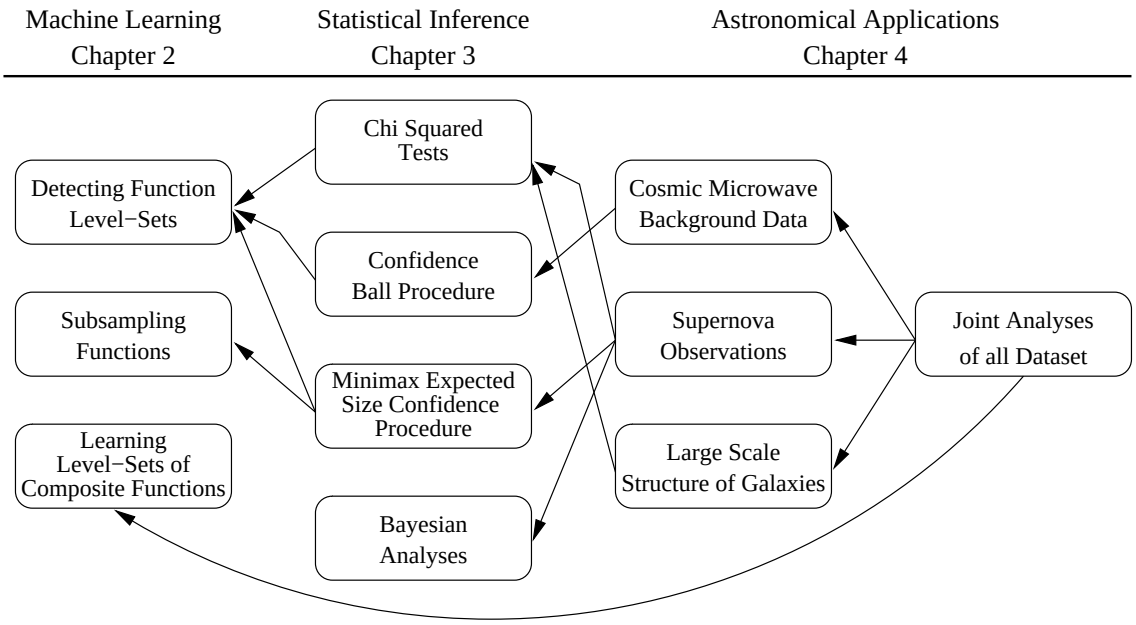


Figure 1.1: Outline of the Chapters 2 - 4 of the thesis, along with their inter-connections. Chapter 2 develops an active learning framework and several techniques to efficiently sample specific portions of a target function. Chapter 3 uses this learning algorithm to design statistical inference techniques to efficiently compute confidence regions for model parameters. Finally, in Chapter 4, we apply the statistical techniques of Chapter 3 to several data sets independently and as an ensemble to accurately compute the values of the core cosmological parameters.

efficiently learn specific properties of a target function. Second, we show how these general active learning algorithms can be used to efficiently compute inferences for several different statistical techniques. We also offer strategies for approximating and computing these statistical methods. Third, we use the combination of machine learning and statistical techniques to efficiently compute the allowed values of the core cosmological parameters in which we are interested. We will see that the combination of efficient approximate algorithms for both the statistical method and the sampling approach result in extremely tight regions much faster than traditional approaches. Finally, we develop an active learning technique to prove that we have not underestimated the ranges of the parameter values due to the finite sampling of the parameter space.

Figure 1.1 depicts the connection between the active learning algorithms, statistical techniques and astronomical data sets which we will examine in this thesis.

1.2 Overview of Thesis

The remainder of this thesis is organized as follows.

We begin, in Chapter 2, by discussing our active learning framework, and several heuristics which can be employed to efficiently learn specific portions of an underlying target function. We shall see that such a targeted approach can be orders of magnitude more efficient that learning the entire function.

In Chapter 3, we will discuss several ways to compute confidence regions and show how the active learning heuristics discussed in Chapter 2 can be used to increase their data and computational efficiency.

In Chapter 4, we briefly describe several astronomical data sets which we then use to demonstrate both the active learning framework and the statistical methods from Chapters 2 and 3. In particular, we compute $1 - \alpha$ confidence regions for several data sets both independently and jointly, and discuss the impact of these statistical inferences on cosmology.

In Chapter 5, we present another active learning technique, this time to prove that the results gathered in Chapter 4 are complete. We show that the use of high dimensional Voronoi diagrams composed of sites with nonzero radii can result in sampling techniques which not only find regions of interest, but also prove that no other regions above a certain size exist.

Finally, in Chapter, 6 we summarize our contributions and conclusions, and describe future work in the area of active learning techniques for specific function properties and their applications to statistics and astronomy.

Chapter 2

Active Learning Strategies

In this chapter, we discuss techniques for selecting experiments. We will assume that we are given a set of possible experiments, Θ, from which we can select some small, specified number of experiments. We will assume that Θ, which we call a "parameter space", is composed of W independent parameters. Each point in Θ corresponds to a vector, θ, which contains the value of the W parameters for that point. In general, each of the W parameters lies within some interval on the real line, and hence Θ can be viewed as the cross-product of these one dimensional ranges.

Let $g : \Theta \mapsto \mathbb{R}$ be the function that we are interested in learning — for instance the variance weighted sum of squares between a model and a set of data used in a χ^2 test (We shall use this mapping explicitly in Section 3.2.1). We will assume that g is not invertible, which is often the case. While we do not know the form of g, we can sample g at any point $\theta \in \Theta$ and compute $g(\theta)$. However, we assume that computation of $g(\theta)$ is costly. Hence, evaluations of g should be used sparingly, as picking optimal, or near-optimal, experiments ($\theta \in \Theta$) may reduce the run time of the algorithm by orders of magnitude (as we shall see in Section 4.2.1). Thus, it is preferable to analyze the current knowledge about the target function, g, and select experiments which quickly refine the estimate of g around our region(s) of interest. Our task, then, is to select experiments from the (most likely infinite) set of possible experiments Θ, which will best aid in the learning of a specific feature or property of the target function g.

There are many approaches that one could use to select experiments. One approach is to select all of our experiments up front. For instance, we could select experiments randomly from Θ. Another possibilities include selecting experiments which form a grid, or lattice in high dimension. Unfortunately, both of these techniques either implicitly or explicitly, require an exponential number of experiments in the number of dimensions

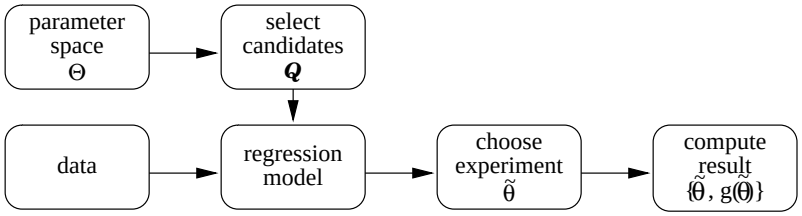


Figure 2.1: Outline of one iteration of our sampling algorithm. Given an initial set of points (possibly empty), we randomly select a set of candidates and score them according to a kriging model. The function g is evaluated at the highest scoring candidate point and the result is added to our data set.

to cover the parameter space. Alternatively, one could disambiguate the effects of each parameter using a Latin squares configuration [Denes and Keedwell, 1974, Tang, 1993]. Latin squares divide the parameter space into a grid and then select $\mathcal{O}(W)$ samples (where W is the dimensionality of Θ), such that each row for each dimension is ensured to contain at least one sample. However, an intuitively better approach is to interactively select the next experiment based upon the results of the previously sampled experiments.

In general, the problem of determining which fixed number of samples should be chosen from a finite set of possibilities in order to best model a target function g is NP-hard. However, actively choosing samples to learn a concept based on a model of the true function is known to significantly reduce the required number of samples, in some cases by exponential factors (e.g. Angluin [1988], Baum [1991]).

Often the input parameters θ_i are real-valued. Thus, the size of Θ is infinite and the optimal selection of samples becomes impossible. Despite the fact that we cannot compute the optimal set of samples, we can compute the approximately best set of samples using a greedy learning algorithm to learn a non-parametric model of the currently observed data.

In this chapter, we describe an active learning framework which can be used to efficiently learn specific properties of a target function. We then look at four of these target function properties. For each property, we show how heuristics which are tailored to learn only the property of interest can be constructed. Finally, we present experiments which indicate that using these heuristics can be several times faster than techniques which try to learn the target function over the entire parameter space.

2.1 Active Learning Framework

Using a non-parametric model of those $\theta \in \Theta$ that we have already observed, we can select the single sample in Θ which best aids in refining our model estimate of g. Clearly, this regression model will be a rough estimate of g, but it will allow us to determine the relative merits of potential sample candidates. Since the size of Θ is assumed to be large or infinite, it is intractable to estimate the value of g for all $\theta \in \Theta$. Instead we select a set of points uniformly at random from Θ and use these points as a sample candidate set, denoted $\mathcal{Q}$. g can be estimated for each $\theta \in \mathcal{Q}$, and we select the θ which best refines our estimate of g: $\tilde{\theta}$. Finally we compute $g(\tilde{\theta})$ and add the result, $\{\tilde{\theta}, g(\tilde{\theta})\}$, to our data set and repeat the process until we have obtained the desired approximation accuracy of g. An outline of the active learning algorithm is illustrated in Figure 2.1.

There are several methods one could use to approximate g. However, we chose to approximate g using Gaussian process regression, a non-parametric generalization of linear regression. Unlike other forms of regression, Gaussian processes can be formulated such that they do not necessarily smooth the data, allowing them to represent subtle features of the function that may become more pronounced with additional data. When modeling functions related to statistical inference for physical models, this property is desirable as these hypothetical models generally return a deterministic response when given a parameter vector $\theta \in \Theta$. Predictions for unobserved points are computed by using a weighted combination of the function values for those points which have already been observed where a distance-based kernel function is used to determine the relative weights. These distance-based kernels generally weight nearby points significantly more than distant points. Thus, assuming the underlying function is continuous, Gaussian processes will perfectly describe the function given an infinite set of unique data points.

Here, we use ordinary kriging, a form of Gaussian processes that assumes the semivariance, $\mathcal{K}(\cdot,\cdot)$, between two points is a linear function of their distance [Cressie, 1991]; for any two points $\theta_i, \theta_j \in \Theta$,

$$\mathcal{K}(\theta_i, \theta_j) = \frac{1}{2}\Big[g(\theta_i) - g(\theta_j)\Big]^2$$

Therefore, the expected semi-variance between two points, $\gamma(\theta_i, \theta_j)$, is given by

$$\gamma(\theta_i, \theta_j) = \mathsf{E}(\mathcal{K}(\theta_i, \theta_j)) = \eta\mathcal{D}(\theta_i, \theta_j) + \zeta$$

where $\mathcal{D}(\cdot,\cdot)$ is a distance function defined on the parameter space Θ, η the maximum magnitude of the semivariance g, and ζ is the observed variance (e.g. experimental noise) when repeatedly sampling the function g at the same location. We have found that using a

simple weighted Euclidean distance function where each dimension is linearly scaled such that the semi-variance along the axis is unity reasonably ensures that parameters are given equal consideration considering their disparate values and derivatives.

Sampled data are assumed to be Normally distributed with means equal to their observed values $g(\theta)$ and variance given by the sampling noise. Note that any subset of the observed points results in a Normal distribution over the value of unobserved points. Thus, we can use the observed set of data, $\mathcal{B} \subset \Theta$, to predict the value of g for any $\theta_q \in \Theta$. This predicted value of $g(\theta_q)$ will be Normally distributed, $(N(\phi_{\theta_q}, \sigma_{\theta_q}))$, with mean and variance given by

$$\begin{aligned} \phi(\theta_q) &= \bar{g}_{\mathcal{B}} + \Sigma_{\mathcal{B}q}^T \Sigma_{\mathcal{B}\mathcal{B}}^{-1} (g_{\mathcal{B}} - \bar{g}_{\mathcal{B}}) && (2.1) \\ \sigma^2(\theta_q) &= \Sigma_{\mathcal{B}q}^T \Sigma_{\mathcal{B}\mathcal{B}}^{-1} \Sigma_{\mathcal{B}q} && (2.2) \end{aligned}$$

where the elements of the matrix $\Sigma_{\mathcal{B}\mathcal{B}}$ and arrays $\Sigma_{\mathcal{B}q}$ and $g_{\mathcal{B}} - \bar{g}_{\mathcal{B}}$ are given by

$$\begin{aligned} \Sigma_{\mathcal{B}\mathcal{B}}[i,j] &= \gamma(\theta_i, \theta_j) \\ \Sigma_{\mathcal{B}q}[i] &= \gamma(\theta_i, \theta_q) \\ (g_{\mathcal{B}} - \bar{g}_{\mathcal{B}})[i] &= g(\theta_i) - \bar{g}_{\mathcal{B}} \end{aligned} \qquad \bar{g}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} g(\theta_i)$$

and the θ_i's and θ_j's are the observed data used to make an inference: $\theta_i, \theta_j \in \mathcal{B}$, $0 \leq i, j \leq |\mathcal{B}|$.

As given, for a set of b observed points ($|\mathcal{B}| = b$), prediction with a Gaussian process requires $\mathcal{O}(b^3)$ time, as a $b \times b$ linear system of equations must be solved. However, for many Gaussian processes — and ordinary kriging in particular — the correlation between two points decreases as a function of distance. Thus, the full Gaussian process model can be approximated well by a local Gaussian process, where only the s nearest neighbors of the query point are used to compute the prediction value. This reduces the computation time to $\mathcal{O}(s^3 + \log(b))$ per prediction, since $\mathcal{O}(\log(b))$ time is required to find the s-nearest neighbors using spatial indexing structures (e.g. kd-trees).

A simplistic Gaussian process model, using kriging with its isotropic and homoskedastic kernel allows for much quicker function estimates than do approaches using adaptive kernels. As we are only interested in an estimate of g and compute this estimate for the reasonably small number of elements contained in our candidate set, $\mathcal{Q}$, the predictions provided by kriging are sufficient to choose the next sample from $\mathcal{Q}$. However, if the cost of computing $g(\theta)$ is sufficiently large, the benefits of increasing the size of $\mathcal{Q}$ and using a heteroskedastic kernel (e.g. Kersting et al. [2007]) may out-weight the computational cost.

Given this active learning framework, we must still decide which element of $\mathcal{Q}$ should be chosen for the next experiment. Strategies for selecting experiments in active learn-

ing frameworks include choosing samples where the model contains no data [Whitehead, 1991], samples where the model predicts poorly [Linden and Weber, 1993], samples where the model has low confidence in its prediction [Thrun and Möller, 1992], samples where we expect the model to have the greatest change [Atlas et al., 1990, Cohn et al., 1994], samples which provided large changes to the model in the past [Schmidhuber and Storck, 1993], samples which reduce the variance of the model (e.g. MacKay [1992], Cohn et al. [1996], McKay et al. [2000], Tong and Koller [2000]), and samples with the maximal entropy [Kulkarni et al., 1993, Sollich, 1994, Sung and Niyogi, 1995]. Recently, Guestrin et al. [2005] showed that choosing samples which maximally decrease the uncertainty about the target function, g, over the entire sample space is $(1 - \frac{1}{e})$ optimal when trying to predict g over the entire domain.

However, in this chapter, we shall see that if we are only interested in a portion of the target function, g, we can hope to do much better. Intuitively, if we design heuristics which are biased toward the region of interest, they will generally out-perform the global heuristics mentioned above by large margins. In the following, we look at three different cases where using specialized heuristics greatly decreases the computational complexity of the search task.

2.2 Learning Function Level Sets

The first task we will examine is that of learning the level-sets of a function.[1] That is, we are interested in where the target function, g, is equal to some specified value, t. This problem naturally arises in situations in which we are interested in partitioning the input space into one class in which the target function is above the threshold, and the other class where it is not. Applications include determining the functional range of wireless networks [Ramakrishnan et al., 2005], factory optimization analysis, gaging the extent of environmental regions in geostatistics [Stein et al., 1999], and computing the extent of statistical confidence regions [Bryan et al., 2005, 2007b].

In one dimension, the level-set discovery problem can be formulated as a root-finding problem where no hints as to the location or number of solutions are given. Several methods exist which can be used to solve this problem (e.g. bisection, Newton-Raphson). However, one dimensional algorithms cannot be easily extended to the multivariate case. In particular, the ideas of root bracketing and function transversal are not well defined

[1]This work was originally published in Bryan et al. [2005] with co-authors Jeff Schneider, Christopher J. Miller, Robert C. Nichol, Christopher R. Genovese and Larry Wasserman. Additional results were taken from Bryan et al. [2007d] with co-authors Jeff Schneider, Chad M. Schafer and H. Brendan McMahan.

[Press et al., 1992]; given a particular bracket of a continuous surface, there will be an infinite number of solutions to the equation $g(\theta) - t = 0$, since the solution in multiple dimensions is a set of surfaces, rather than a set of points.

Numerous active learning papers deal with similar problems in multiple dimensions. For instance, Ramakrishnan et al. [2005] presents a method for picking experiments to determine the localities of local extrema when the input space is discrete. Alternatively, others have used a variety of techniques to reduce the uncertainty over the problem's entire domain to map out the function (as mentioned in Section 2.1) or locate the optimal value (e.g. Moore and Schneider [1996]).

We are interested in locating the subset of the input space wherein the function is above a given threshold. Algorithms that merely find a local optimum and search around it will not work in general, as there may be multiple disjoint regions above the threshold. While techniques that map out the entire surface of the underlying function will correctly identify those regions which are above a given threshold, we assert that methods can be developed that are more efficient at localizing a particular contour of the function. Intuitively, points on the function that are located far from the boundary are less interesting, regardless of their variance. Let us see how we can use this intuition to develop a heuristic which is significantly more efficient than global variance minimization when one is solely interested in localizing a function level-set.

2.2.1 Active Learning Algorithm

Suppose we are given a sample space, $\Theta \subseteq \mathbb{R}^W$, and a black box which allows us to evaluate g. Given a threshold t, we want to find the set of points from the input space such that $g(\theta) = t$: $\mathcal{V} = \{\theta \in \Theta | g(\theta) = t\}$. Note that for some of the applications we mentioned in Section 2.2, we are actually interested in the points in the input space where g is greater than t. However, these points can easily be found as they are either the interior, or exterior of the level-set $\mathcal{V}$. We propose to learn the set $\mathcal{V}$ using the active learning framework discussed in Section 2.1. In order to do so, we must determine how to select the next experiment given the observations we have made so far.

2.2.2 Choosing Experiments

As discussed in Section 2.1, there are many strategies that we could employ to select experiment in order to learn $\mathcal{V}$. In this section, we look at a mix heuristics, some of which try to optimize the global solution, while others focus solely on the task of level set

detection. Here we present heuristics taken from throughout the performance range of the heuristics we tried. The heuristics we discuss here are:

Random This heuristic selects one of the candidate points uniformly at random. This method serves as a baseline for comparison of the other heuristics.

Variance This heuristic selects the candidate point which has the largest predicted variance, $\sigma^2(\theta)$. Using model variance to pick the next experiment is common for active learning methods whose goal is to map out the target function over a parameter space [MacKay, 1992, Guestrin et al., 2005]. Since variance is closely related to distance when using kriging, this heuristic samples points which are distant from their nearest neighbors. However, when searching for level-sets, we are less interested in the function away from the level-set boundary, and instead want to focus our sampling resources near this predicted boundary. Intuitively, this algorithm should perform substantially worse than heuristics that concentrate on the function level-set.

Probability of Incorrect Classification: Since we are trying to map the boundary between points above and below a threshold, we consider choosing the point from our random sample which has the largest probability of being misclassified by our model. Using the distribution defined by Equations 2.1 and 2.2, the probability, p, that the point is above the given threshold can be computed. The point is predicted to be above the threshold if $p > 0.5$ and thus the expected misclassification probability is $\min(p, 1-p)$.

Entropy: Related to misclassification probability, is the class entropy: $-p\log_2(p) - (1-p)\log_2(1-p)$. Note that entropy is a monotonic function of the misclassification rate, so both heuristics will choose the same points, given the same candidate set. However, the relationship is non-linear. The probability of misclassification heuristic places higher relative weight on those points closer to the boundary than does the entropy heuristic. Hence, we expect these two heuristics to have different effects when combined with other heuristics.

Information Gain Information gain is a common myopic metric used in active learning. Information gain at the query point is the same as entropy in our case because all run experiments are assumed to have the same variance, ζ. Computing a full measure of information gain over the whole state space would provide an optimal 1-step experiment

choice. In particular, Guestrin et al. [2005] showed that greedily picking experiments with the maximal mutual information gain results in a $(1 - 1/e)$ optimal approximate solution when learning the target over the entire parameter space. In some discrete or linear problems this can be done, but it is intractable for continuous non-linear spaces. Specifically, calculating the information gain of a proposed sample requires integrating the difference between the current model and expected result of the proposed sample over all space. Since our function approximator has only local support for predictions, we can reduce this integral down to the local region. However, even on this local region, computing the expected value of the model requires multiple matrix inversions to account for differences in the one hundred nearest neighbors over the local region. Even approximating this integral with a (small) finite sum, was found to be prohibitively expensive. As such we do not consider a traditional information gain heuristic, but rely on efficient point estimates which act as proxies for global information gain.

Products of metrics: One way to rectify the problems of point policies that focus solely on points near the boundary or points with large variance regardless of their relevance to refining the predictive model is to combine the two measures. Intuitively, doing this can mimic the idea of information gain; the entropy of a query point measures the classification uncertainty, while the variance is a good estimator of how much impact a new observation would have in this region and thus what fraction the uncertainty would be reduced. Ramakrishnan et al. [2005] proposed scoring points based upon the product of their entropy and variance to identify the presence of local maxima and minima, a problem closely related to boundary detection. We shall also consider scoring points based upon the product of their probability of incorrect classification and variance. Note that while entropy and probability of incorrect classification are monotonically related, the product of entropy and variance and the product of the probability of incorrect classification and variance are not.

Straddle Another strategy for combining entropy and variance estimators is the straddle heuristic:

$$\mathsf{straddle}(\theta_q) = 1.96\sigma(\theta_q) - \big|\phi(\theta_q) - t\big|.$$

This heuristic combines the desire to search the entire input space with that of refining our estimate around known interesting regions, by picking points that the model predicts are both close to the boundary and have large variances.

Note that the straddle heuristic chooses those points with large variances which straddle the boundary. In particular, if a point is near the boundary, then $\phi(\theta_q) \simeq t$ and this

metric is equivalent to a variance-only metric, choosing points that are removed from one another. However, if the point is not on the boundary, then its score drops off proportionally to the distance from the boundary. The straddle score for a point may be negative, which indicates that we predict that the probability that the point is on a boundary is less that five percent. The straddle algorithm scores points highest that are both unknown and near the boundary, and thus gives scores that intuitively are similar to that of information gain. However, note that the straddle heuristic relies on the variance estimate, so is also subject to oversampling edge positions.

2.2.3 Experiments

Let us now evaluate how these heuristics perform on the task of level-set identification. We are interested in heuristics which not only sample points on the boundary often, but also sample around the entirety of the boundary. That is, we are not particularly interested in knowing if the specified level-set exists (as we assume it does), but rather want to know where in the parameter space the level-set is located. As noted in Section 2.2.1, discovering the level-sets of g and the regions of g which are above t are duals of each other. Thus, we use classification accuracy of points from Θ as a metric for assessing our ability to learn those regions of g which are above t, and hence our ability to learn $\mathcal{V}$. For the five following functions, we recorded the number of experiments sampled by each heuristics until they reached the desired classification accuracy (95 or 99% depending on the problem). The sampling and testing process was repeated 20 times to account for variations due to the random nature of the candidate generation process. The considered functions are:

2D Peak This task involves finding the level-set of the sinusoidal function

$$g(x, y) = \sin(x) * \sin(y)$$

where $g(x, y) = -0.8$, where $x, y \in [-3.4 : 3.4]$. The level-set, $\mathcal{V}$, consists of two separated rings, shown as the red lines in Figure 2.2. This function was fairly easy for most of the heuristics to learn, but the discontinuous surface (composed of the separate rings), revealed those heuristics which correctly explored the space, from those that merely exploited the level-set boundary once it became apparent.

2D Deboor The second target function we looked at was the 2D DeBoor function:

$$g(x, y) = \cos\left(6 + \frac{2x}{|x|} + \frac{4y}{|y|}\right)\left[1 - 3\delta^2 + 2\delta^3\right] + 1$$

where

$$\delta = \left(|x| - \frac{1}{2}\right)^2 + \left(|y| - \frac{1}{2}\right)^2.$$

For this task, we were interested in where the function was equal to -1.3 when x and y were both restricted to the interval $[-1 : 1]$. Like the 2D Peak function, this function has two clearly defined level-sets which again form rings. However, unlike the 2D Peak function, this function has regions where the semi-variance is extremely high. Moreover, when x or y equals zero, the function is undefined. While these properties violate the assumptions of our Gaussian process (namely that the surface is smooth and continuous), we will see that even in these adverse conditions, our framework seems to hold up. We chose to use this function, as it was the function used by Ramakrishnan et al. [2005] to test their ent-var heuristic.

2D Sine The third problem we look at is the 2D Sine function:

$$g(x, y) = \cos(3 * x * y) - \sin(10 * x) - \cos(4 * y).$$

Here we restrict x and y to the ranges $x \in [0, 1]$ and $y \in [0, 2]$, and set the threshold, t, to zero. The level-set of the target function is denoted by the solid red lines in Figure 2.3. This target function has several interesting properties. First, the target level-set winds through the parameter space giving ample length to test the accuracy of the learned approximating model. Second, the boundary is discontinuous with several small pieces. Third, there is an ambiguous region around (0.9, 1.0) where the target function is both close to the boundary $t = 0$, and has a small derivative. Finally, there are areas in the parameter space where the function is far from the threshold and hence should be quickly pruned by the search heuristics. Thus we hope to see regions which are far from the target level-set, such as (0.4, 0.8), sparsely sampled.

4D Sine The previous three target function have all been two dimension, in order to provide visual evidence that the heuristics are performing as we desire. However, our methods are not limited to small dimensional tasks; indeed in Section 4.2.1 we will apply this algorithm to a seven dimensional target function. However, here we will restrict ourselves to the four dimensional problem:

$$g(\vec{x}) = \sin(10x_1) + \cos(4x_2) - \cos(3x_1x_2) + \cos(2x_3) + \cos(3x_4) - \sin(5x_3x_4).$$

Again, we let the threshold be $t = 0$, and we restrict the parameter space to $x_1 \in [0, 1]$, $x_2, x_4 \in [0, 2]$ and $x_3 \in [1, 2]$. This function was chosen due to its similarity to the 2D

	2D Peak	2D DeBoor	2D Sine	4D Sine	SNLS
random	153±42	7727±987	617±158	6254±364	1012±188
entropy	DNF	DNF	DNF	6121±1740	DNF
variance	108±8	4306±573	207±7	2320±57	664±227
ent-var	48±2	1621±201	117±5	1210±43	390±74
misclass-std	102±33	740±117	113±11	1362±89	415±63
straddle	41±12	963±136	106±5	1265±94	410±57

Table 2.1: Number of experiments required to obtain 99% classification accuracy for the 2D models and 95% classification accuracy for the 4D and SNLS models for various heuristics. Heuristics requiring more than 10,000 experiments to converge are labeled "DNF".

Sine and 2D Peak problems. Moreover, the fairly low dimension of the problem allowed us to densely sample the parameter space when estimating the classification accuracies of each of the heuristics.

SNLS The final target function we look at is the SNLS function. Unlike the previous four functions which were synthetic, this function arises from the statistical analysis of real data. Specifically, this function is the χ^2 statistic of a set of supernova data, and the cut off, t, is set such that the region interior to the level-set comprises the 95% confidence region of the χ^2 test. We will explain how the χ^2 test can be used with our active learning framework in Section 3.2. In 4.1.2, we will discuss the SNLS data in more detail, and the results of the 95% confidence intervals computed with this data will be described in 4.2.2.

Now let us look at how the heuristics described in Section 2.2.2 perform on the test target functions we just described. As mentioned above, we ran 20 trials, in each computing the number of samples to receive the desired classification accuracy. For the two dimension target functions, the desired classification accuracy was set at 99%, while the 4D Sine and SNLS results are for a 95% classification accuracy. Classification accuracy was computed by testing the learned GP at the points on a 141×141 grid for the 2D problem and a grid with 100 points per dimension for the other two models. For each point on the grid, θ_q, the model was judged to be classify correctly if the sign of $\phi(\theta_q) - t$ was equal to the sign of $g(x) - t$; that is, the GP and the target function agreed as to which side the threshold the point θ_q was.

In Table 2.1 we list the average number of samples needed by each heuristic to achieve

the desired classification accuracy along with the standard deviations among the 20 trials. Entries to the left of the vertical line indicate the number of samples required to obtain a 99% classification accuracy, while those on the right correspond to a 95% classification accuracy. Results are presented for the Random, Entropy, Variance, the product of Entropy and Variance, the product of Probability of Misclassification and the square-root of Variance and the Straddle heuristics. For simplicity, we will name these heuristics random, entropy, variance, ent-var, misclass-std, and straddle, respectively.

Note that picking points solely on entropy does not converge in many cases. This is because points near the boundary have high entropy, even when the boundary is well sampled (as the heuristics does not depend on the variance of the estimate). Thus, once they find a single boundary in the space, both the entropy and misclass-std would rather continue sampling this boundary than explore new regions of space, as can be seen in Figures 2.2(b) and 2.3(b). Meanwhile, both the straddle algorithm and misclass-std heuristic result in approximations that are significantly better than random and variance heuristics. Table 2.1 shows that the ent-var heuristics often performs as well as the straddle heuristic. However, the ent-var heuristic is much worse that either the straddle or misclass-std heuristics on the 2D DeBoor function. This may be due the discontinuities or the large semivariances inherent in this function. In general, the straddle heuristic seems to perform as well or better than all of the other heuristics considered over a wide range of target functions.

In Figures 2.2 and 2.3 we depict the the experiments selected by the various heuristics during a single trial for both the 2D Peak and 2D Sine level-set tasks. For the 2D Peak task we show results after selecting 30 experiments, while for the 2D Sine problem the results correspond to selecting 100 experiments. In each of these figures, we denote the samples that were chosen with black dots. The true level-set, $\mathcal{V}$, is marked with a red line, while the derived level-set — that given by sampling our Gaussian Process — is marked in blue. For these experiments we would like the blue curve to completely overlap the red curve.

However, as seen in Figures 2.2 and 2.3, there the red and blue curves deviate dramatically for some heuristics. This is especially apparent for the entropy heuristic. As previously noted, this heuristic favors exploitation of a known boundary over exploration of the parameter space. In fact, the entropy heuristic almost randomly samples points until it locates a boundary. Once a point on the boundary is discovered, it continues to sample points from the candidate set, $\mathcal{Q}$, that are as close as possible to the point that was previously found to be on the boundary. Note that in both cases, the heuristic has managed to sample along a single edge of the level set. This is a direct result of the limited number of candidates from which each heuristic get to choose; given an unlimited number of sam-

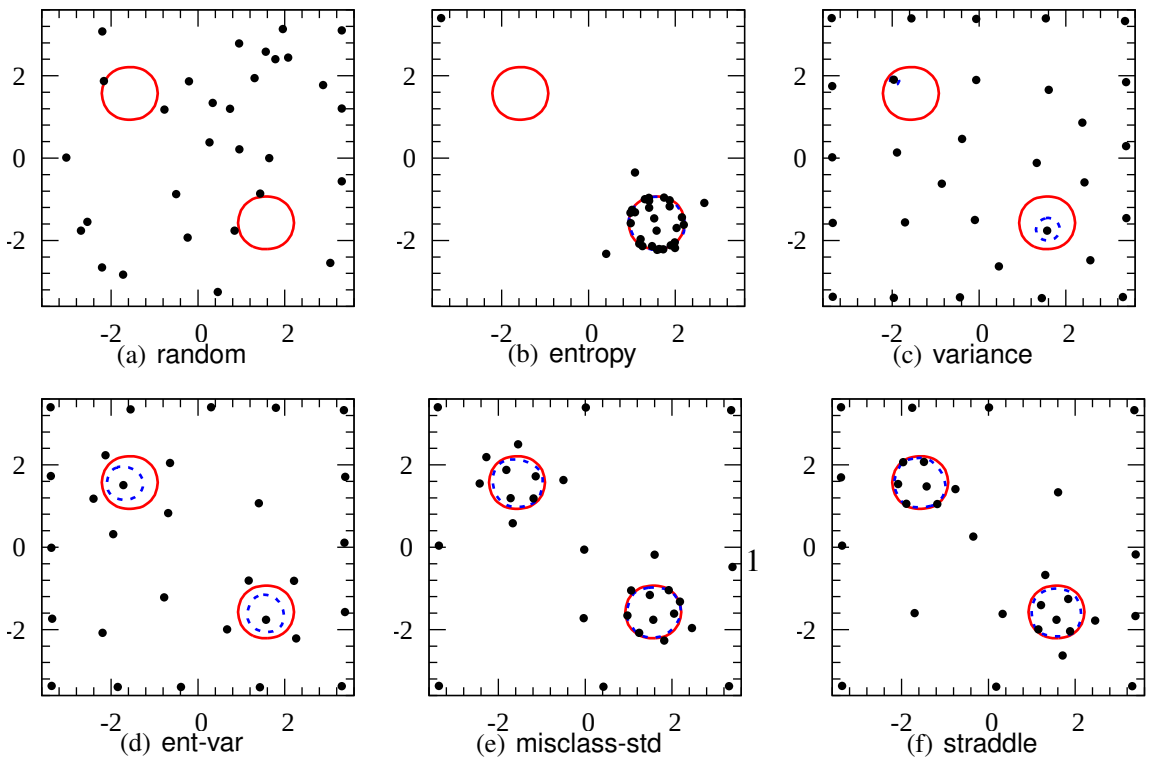


Figure 2.2: Predicted function boundary (blue), true function boundary (red) and experiments (dots) for the 2D Peak function after 30 experiments using various heuristics. The straddle heuristic outperforms the other heuristics by eliminating those regions of the input space that are unlikely to be near the boundary.

ples, the heuristic would converge to a single solution, and not even explore the piece of the level set it had discovered.

In contrast, the variance metric has effectively over sampled the parameter space. Note that the resulting sample pattern is almost grid-like, and irrespective of the boundary it is supposed to find. While the derived level set is much better than the entropy heuristic, the variance heuristic does not get the details of the true level set correct.

However, Figures 2.2 and 2.3 show that the heuristics which combine Variance and Entropy perform well. These heuristics — (straddle, ent-var and misclass-std — tend to emphasis the boundary, while still ensuring that the remaining parameter space is explored. While heuristics that are both the product and sum of Entropy and Variance appear to have sampled points allowing the underlying Gaussian Process to converge to the true function

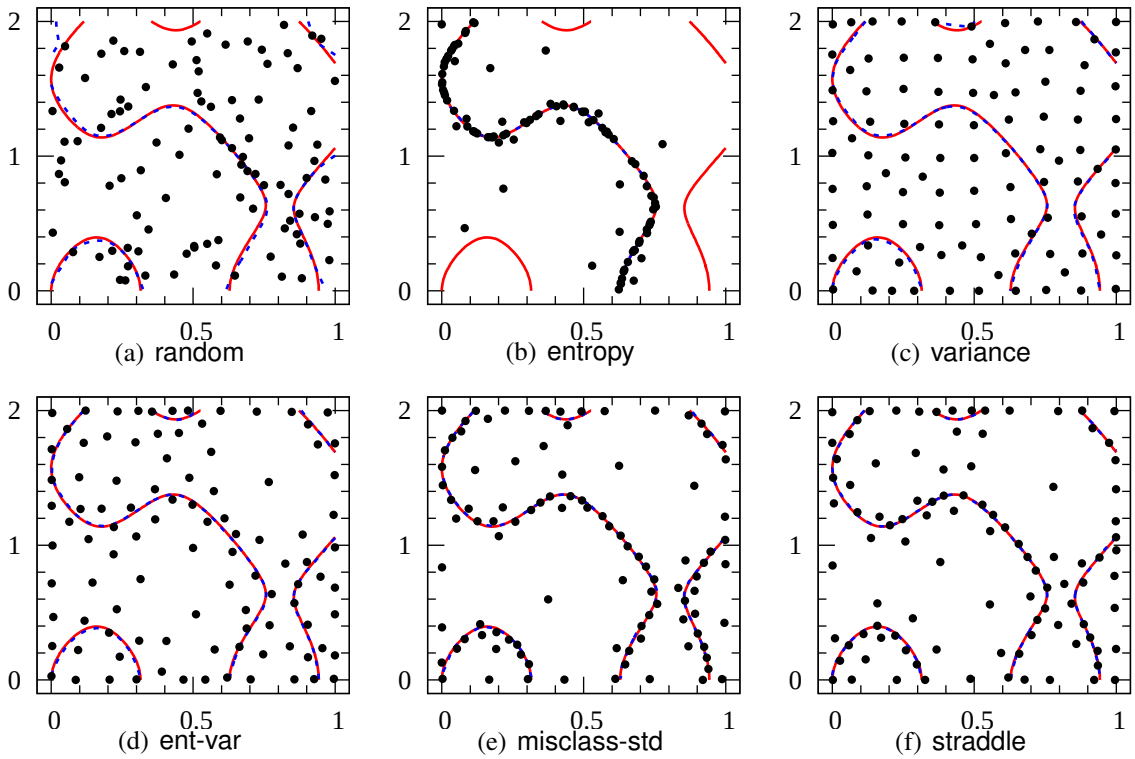


Figure 2.3: Predicted function boundary (blue), true function boundary (red) and experiments (dots) for the 2D Sine function after 100 experiments using various heuristics. Again the straddle heuristic is able to eliminate unlike regions of parameter space and outperform the other heuristics.

(at least around the boundary of interest), the straddle heuristic (the sum of Entropy and Variance) outperforms the other combinations. The straddle heuristic places many more points directly on the boundary as opposed to ent-var and misclass-std, which tends to place points slightly to either side of the boundary.

2.3 Subsampling Functions Below a Specified Level-Set

The second task we will look at is closely related to the first. Here we are interested in predicting a target function g in the regions where g is less than some specified threshold.[2]

[2]This work originally appeared in Bryan et al. [2007c] with co-authors Jeff Schneider and Chad M. Schafer. Additional results were taken from Bryan et al. [2007d] with co-authors Jeff Schneider Chad M.

Given a specified threshold t, we wish to learn g on the set $\Theta' = \{\theta \in \Theta | g(\theta) \leq t\}$. That is, we are interested in selecting q samples from Θ which minimize the estimated variance of g over all $\theta \in \Theta'$.

There are several applications where learning g on the restricted subset Θ' is beneficial. For instance, a geologist may be interested only in spatially modeling the concentration of some mineral or oil deposit where the deposit's concentration is above some financially viable threshold. Another case that we will see in Section 3.4.5 is that of computing confidence regions for model parameters. In this case, we desire to sample points from the parameter space where a specific test statistic is less than a certain value.

One straightforward strategy is to learn the level-set where $g(\theta) = t$ using the straddle heuristic from Section 2.2.2 and then use these boundaries to derive Θ'. However, this process requires additional information to determine how many samples should be chosen by the boundary finding heuristic before switching to an exploitation mode. It is not clear how many samples should be chosen based on the boundary finding heuristic before a sampling is done within the boundary. If we fail to sample enough points we may underestimate the size of Θ', while sampling too many points will result in wasted computation. Intuitively, we desire a heuristic which trades off exploitation for exploration throughout the selection process.

2.3.1 Active Learning Algorithm

As with the algorithm in Section 2.2.1, the active learning framework we use to learn g over Θ' is the one described in Section 2.1. The only difference between the algorithm describe here and the one described in Section 2.2.1, is that here we are interested in learning g over the entire space Θ', not just the level-set $\{\theta \in \Theta | g(t) = t\}$. Thus, the main difference between the algorithms will be the heuristic we employ to search the parameter space Θ.

2.3.2 Choosing Experiments

As mentioned in Section 2.3, we desire a heuristic which actively trades off exploitation for exploration. Here we compare the random and variance heuristics of 2.2.2 with the following heuristics:

Straddle & Variance: Instead of mapping g throughout Θ, a more practical solution is to concentrate on g where $g(\theta) \leq t$. One strategy is to first spend a fixed number of samples to locate the boundary of Θ' using the straddle heuristic mentioned in Section 2.2.2. For the remainder of the samples, the heuristic then selects points which have have the largest expected variance within the predicted boundary. If none of the points in the candidate set are predicted to be within the boundary region, we pick the point that most helps refine our boundary estimate: the point with the largest straddle value.

Entropy$\times$Variance & Variance: Similar to the last sampling heuristic, this heuristic first tries to estimate the location of the function's level-set and then sample within this level-set. However, instead of using the straddle heuristic, we use the ent-var heuristic of Ramakrishnan et al. [2005] mentioned in Section 2.2.2 to find the boundary. After spending a fixed number of samples locating the boundary, the heuristic switches to an exploitation mode and samples within the boundaries that have been located.

Variance Above Threshold The previous two heuristics require a parameter to determine how many samples should be chosen from the boundary finding heuristic before switching to an exploitation mode. However, as mentioned in Section 2.3, we are interested in a heuristic which trades off exploitation for exploration throughout the selection process. To this end, we propose a modified version of the straddle heuristic, Variance Above Threshold (threshvar) which performs this trade off:

$$\mathsf{threshvar}(\theta_q) = 1.96\sigma_{\theta_q} - \max\left\{0, \phi_{\theta_q} - t\right\}.$$

When $g < t$, the threshvar heuristic is solely a function of variance. When $g > t$, then the Variance Above Threshold heuristic reverts to the straddle heuristic of Bryan et al. [2005], selecting samples which are expected to be near the boundary and far from other samples.

2.3.3 Experiments

Let us now assess how these heuristics perform on several real and synthetic data sets by estimating the mean variance for points within Θ'. As the exact value of the mean variance is computationally prohibitive — it requires computing the variance at all $\theta \in \Theta'$ — we estimate the mean variance using a randomly selected subset of points from Θ'. Clearly, we desire the heuristics to have low variance within Θ', as this indicates that the kriging model has well approximated g within Θ'. Moreover, heuristics which have low

	2D Peak	2D DeBoor	2D Sin.	4D Sin.	SNLS
random	0.283±0.011	0.312±0.016	0.606±0.007	2.403±0.020	7923±200
variance	0.263±0.003	0.285±0.005	0.551±0.002	2.478±0.007	7670±55
entvar-thresh(0)	0.168±0.000	0.160±0.001	0.512±0.001	2.208±0.007	6937±34
straddlevar(0)	0.510±0.180	0.130±0.001	0.810±0.786	2.336±0.091	11791±2555
straddlevar(200)	0.139±0.000	0.136±0.001	0.493±0.001	2.275±0.046	7051±221
threshvar	0.146±0.000	0.139±0.002	0.493±0.001	2.093±0.053	6439±31

Table 2.2: Mean residual variance for points within Θ' of various heuristics after picking 500 samples. The threshvar heuristic has the minimal mean variance of those heuristics which do not require hand-tuning, and often performs nearly as well, or better, than those heuristics for which optimal parameters were selected.

variances within Θ' must have also sampled Θ' extensively in order to obtain that low variance. Thus, heuristics which minimize the mean variance are likely to produce dense quasi-random sets of points within Θ'.

Heuristics were tested on the same five data sets that we used in Section 2.2.3: 2D Peak, 2D Deboor, 2D Sine, 4D Sine, and SNLS. The 2D Peak function was particularly troublesome for the heuristics which only had a limited number of samples to detect the boundary before switching to an exploitation mode because of the two separated regions which were below the boundary.

Results for the mean variance among points within Θ' on these data sets after performing 500 samples each in 20 trials are shown in Table 2.2. We have denoted the Straddle & Variance, Entropy$\times$Variance & Variance, and Variance Above Threshold heuristics as straddlevar, entvar-thresh, and threshvar, respectively. The number in parentheses after straddlevar and entvar-thresh indicates how many samples were chosen to define the boundary before switching to an exploitation mode.

From Table 2.2, it is clear that some of the heuristics perform substantially worse than either random or variance sampling. These include entvar-thresh(0) and straddlevar(0). Both of these heuristics are based upon the idea of using some fixed number of samples to determine the location of the boundary and then use the remaining set of experiments to sample the interior region (Θ'). However, if the cutoff between exploration and exploitation is too small (in this case it is set to zero), then the heuristics will not completely map out the parameter space and are likely to miss large sections of Θ', especially if Θ' is disjoint. Given the similarity between entvar-thresh and straddlevar, in the following, we will refer only to straddlevar.

Figure 2.4 illustrates the fact that the straddlevar(0) heuristics searches only enough

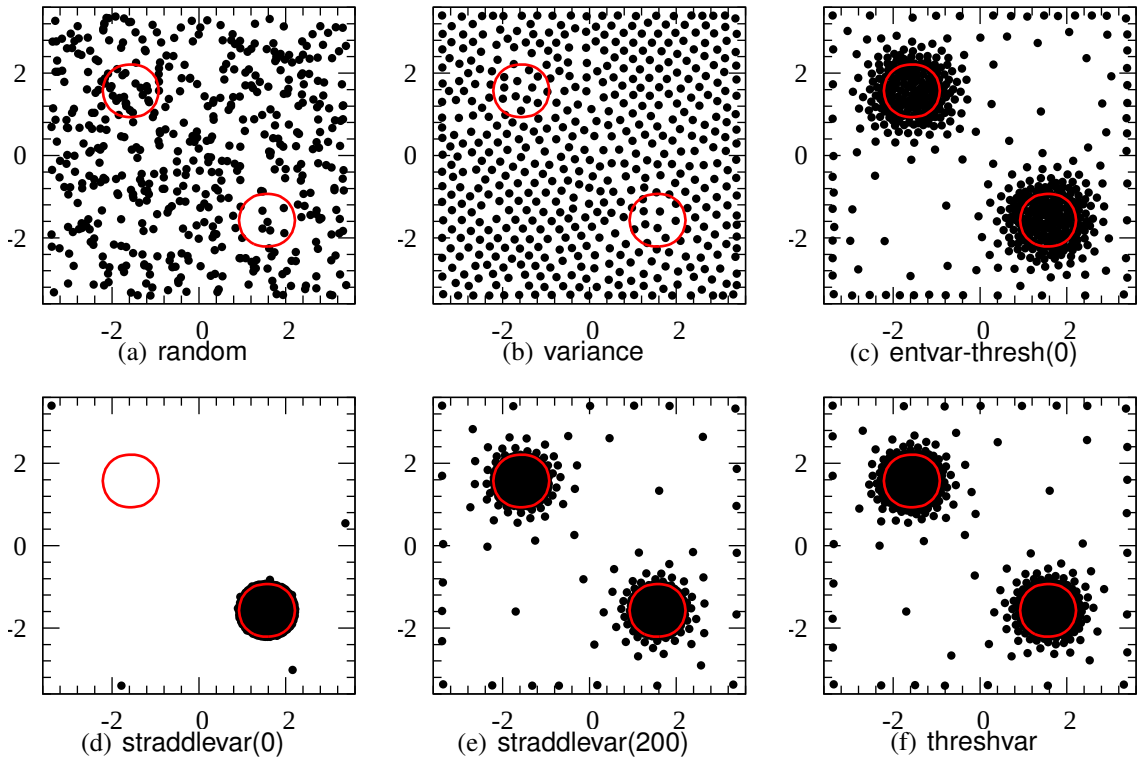


Figure 2.4: 500 samples chosen by various heuristics in a single trial, along with the target minima (open red circles). The variance heuristic focuses only on exploration, while the straddlevar(0) heuristic focuses only on exploitation, ignoring one of minima. The threshvar heuristic balances exploration with exploitation, both finding and frequently sampling both minima.

	2D Peak	2D DeBoor	2D Sin.	4D Sin.	SNLS	WMAP YR1
random	0.06±0.01	0.04±0.01	0.55±0.02	0.13±0.01	0.06±0.01	0.001
variance	0.05±0.00	0.04±0.00	0.53±0.01	0.11±0.01	0.06±0.01	—
entvar-thresh(0)	0.89±0.02	0.88±0.04	0.90±0.01	0.62±0.01	0.71±0.03	—
straddlevar(0)	0.89±0.03	0.89±0.03	0.90±0.02	0.62±0.01	0.71±0.04	—
straddlevar(200)	0.74±0.01	0.75±0.02	0.83±0.01	0.52±0.02	0.52±0.02	—
threshvar	0.57±0.01	0.67±0.03	0.83±0.01	0.39±0.01	0.16±0.01	0.32

Table 2.3: Efficiency of various heuristics after picking 500 samples. The threshvar heuristic has greater efficiency than all heuristics that remain in an exploration mode throughout the trial (random, variance), and is competitive with heuristics that switch to an exploitation mode (straddlevar(·), entvar-thresh(0)).

to find a single peak in the target function and spends the remainder of its samples within the peak. On the other extreme, the variance heuristic evenly spreads its samples over the entire space. The threshvar heuristic takes the middle route, both examining the entire space, but focusing a majority of the points in both local minima.

While we are primarily interested in achieving low variance within Θ', we are also interested in heuristics which heavily sample Θ', as these are the points that we will use for statistical inference techniques in Chapter 3. For each of the heuristics above, we measure their accuracy, which we define to be the fraction of samples selected from Θ, that were also in Θ'. In Table 2.3, we present efficiency results from the various heuristics on the 5 data sets. Additionally, we present some results from the Wilkinson Microwave Anisotropy Probe (WMAP) data set which depicts fluctuations in the temperature of the cosmic microwave background (CMB) (This data set will be described in more detail in Section 4.1.1.) However, due to the computational costs associated with running the WMAP experiments, we present results only for the random and threshvar heuristics.

As Tables 2.2 and 2.3 show, there is a strong correlation between efficiency and mean variance among the points in Θ'. Heuristics which favor exploitation over exploration have high efficiencies, but also high mean variances, while those that favor exploration have both lower efficiencies and mean variances. The most efficient heuristic on all of the experiments was straddlevar(0). However, this efficiency came at the cost of failing to locate all of the desired minima as seen in Figure 2.4(d). Random sampling and variance-weighted sampling performed equally well in terms of efficiency. However, variance-weighted sampling resulted in lower mean variance,.

As it generally takes only a few dozen to a hundred points to locate the minima in two dimensions, the straddlevar(200) heuristic appears to be superior for these low di-

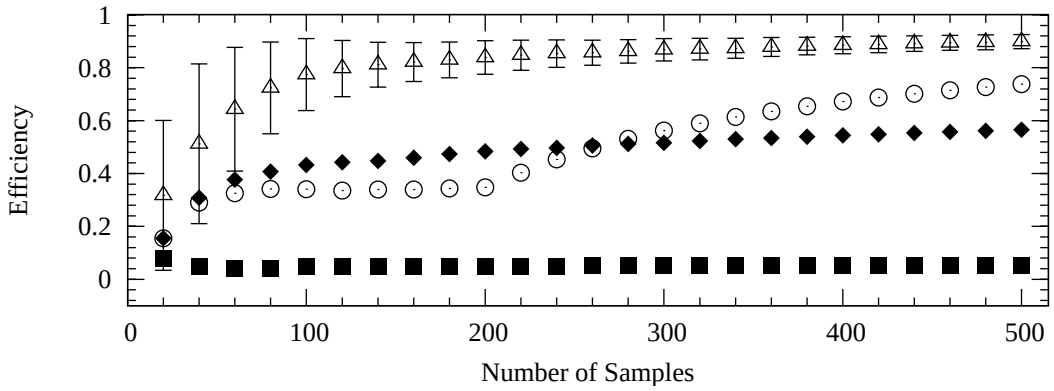


Figure 2.5: Efficiency of the straddlevar(0) (triangle), straddlevar(200) (open circle), threshvar (diamond) and variance (square) heuristics as a function of the number of samples chosen. Note the sharp increase in efficiency in the straddlevar(200) heuristic after 200 samples, when the heuristic switches from a exploration to an exploitation mode. The threshvar heuristic out-performs all other heuristics while they remain in their exploration mode.

mensional tasks. This is unsurprising, as the constant 200 was chosen to maximize the performance of the straddlevar heuristic on the 2D problems. After quickly finding the minima, the straddlevar(200) heuristic was able to switch to an exploitation mode (seen in Figure 2.5), allowing it to obtain both high efficiency and low mean variance. However, a major drawback of this heuristic is that one must specify the number of samples after which the heuristic should switch to an exploitation mode. Using too few samples results in failure to locate the desired minima (shown in Figure 2.4(d)), while using too many samples results in worse efficiency than the threshvar heuristic (as seen in Figure 2.5). While using 200 samples for boundary prediction for straddlevar results in a good trade off between discovery and exploitation for the 2D problems, it does not fare as well on the Sin4D or SNLS problems. In practice it is very difficult to choose this exploration constant without knowing the underlying function.

The threshvar heuristic performs well both in terms of efficiency and expected variance in Θ'. While it's efficiency was less than heuristics that focused on exploitation, its mean variance was near the minimal value for all the heuristics, and was significantly less than those heuristics which do not have tuning parameters. This indicates that the heuristic is correctly identifying and sampling the appropriate regions of the function. Moreover, its performance is similar to those heuristics which were optimized for the task using additional samples of the function. Thus, the threshvar heuristic performs nearly as well as the best heuristics for each problem without the need of hand-tuning, or switching to an

exclusively exploitation mode.

As a result, the threshvar heuristic can be directly applied to any problem, without first determining the approximate location and number of local minima that compose Θ'. This information is required by the heuristics that switch from an exploration to an exploitation mode (such as straddlevar) to ensure that all of the local minima are found before switching to an exploitation mode. Failure to ensure that all minima have been found results in an underestimation of Θ', as the one shown in Figure 2.5.

These experiments indicate that the Variance Above Threshold heuristic, threshvar, outperforms the other heuristics and significantly outperforms random sampling. We find that using the threshvar heuristic is up to two orders of magnitude more efficient than random sampling while the resulting mean variance is equal to or less than choosing points based upon the variance heuristic.

2.4 Learning Level Sets of Composite Functions

The third task that we will discuss is the problem of learning the level sets of a function which is a composition of several observable functions.[3] As with the level set detection algorithms discussed in Section 2.2, if the problem is only one dimensional, a variety of methods can be used to efficiently find a solution. However, there is no way to scale these solutions to problems with multiple input dimensions. Multi-dimensional problems often arise in scientific applications where several types of experiments or data sets may be available to test the validity of some hypothesis. In such cases, scientists, are interested in regions of the space which are statistically plausible for all of the experiments types or data sets.

One example is determining the spatial location of a disease outbreak using information derived from medical records (e.g. hospital admits), as well as sales of over the counter and prescription medications. In this case, public health officials are interested in regions which have over densities of both specific hospital admits and their corresponding medication sales. The presence of one or the other is not necessarily disturbing. Moreover, returning all cases where only one stream was unusual would overwhelm the health officials, due to their relative frequency. Another example is cost/benefit analysis of resource extraction where one must estimate the value of all resources to be obtained along with the extraction costs in terms of infrastructure and human resources required. In this section, we focus on a third application: simultaneous statistical analysis of multiple related data

[3]This work was original published in Bryan and Schneider [2007] with co-author Jeff Schneider.

sets for finding valid parameter ranges in scientific models.

When given several models, each associated with a different data set, computing confidence regions for the data sets as an ensemble often reduces to finding a level set of a function which is a composite of the evaluations of each data set for a particular parameter setting. A sampling algorithm for this problem must at each iteration select a parameter setting to be tested and decide which experiment type to use for the test. Often tests have different associated costs and accuracies. Moreover, a test on a single data set may be sufficient to reject a particular model or parameter setting without testing other data sets. Thus, an efficient algorithm must consider both the cost of each sample and the benefit of selecting that sample (on a particular data set) for learning the entire composite function.

Traditionally, experiment selection has been achieved in the sciences in a somewhat ad-hoc fashion where one scientist publishes plausible parameters derived from one type of experiment and another uses that information to guide the selection of parameters in future experiments. In Bayesian analysis, results from one experiment might form the priors for the next. A more rigorous and efficient approach is to consider the multiple experimental sources for evaluation simultaneously and choose evaluation samples in light of their contribution to the combined evaluation function. We now look at two methods to perform joint analyses in ways that form composite functions.

Joint Statistical Analysis

Joint analyses tend to take one of two forms. In the first, we create statistical model which simultaneously considers all data sets. For instance, when performing an analysis on two data sets using χ^2 tests, we will have one χ^2 test for data set A and a second for data set B. However, since the χ^2 test assumes that each of the data points have dependencies given by the covariance matrix, we can combine the two tests into a single χ^2 test. Let x_*, μ_*, and Σ_* be the associated test model, observed data and observed covariance for the model $*$ given some vector θ from the parameter space Θ. For instance $\mathbf{y}_A$ is the test model for data set A, while μ_B is the observed data for model B. If data set A and B have a and b degrees of freedom respectively, then the combined χ^2 test will be

$$\left[(x_A - \mu_A)^T, (x_B - \mu_B)^T\right] \begin{bmatrix} \Sigma_A & \Sigma_{AB} \\ \Sigma_{AB} & \Sigma_B \end{bmatrix}^{-1} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix} \sim \chi^2_{(a+b)}$$

where Σ_{AB} is the covariance of the data points between data sets A and B. If data sets A and B are independent, then all elements of Σ_{AB} are zero, and we can compactly write the above expression as:

$$(x_A - \mu_A)^T \Sigma_A^{-1} (x_A - \mu_A) + (x_B - \mu_B)^T \Sigma_B^{-1} (x_B - \mu_B) \sim \chi^2_{(a+b)}.$$

That is, the target function is merely the sum of the two observable functions: the variance weighted sum of squares for both data sets. This approach can be extended to any number of independent data sets, yielding a composite target function which is the sum over the χ^2 statistic for the k individual data sets.

Another approach to performing simultaneous joint analyses is to combine the models' p-values. There are many ways to combine test procedures, including using Bonferroni corrections [Bonferroni, 1936], the inverse normal method, and inverse logit methods [Hedges, 1985]. However, the most common method to combine p-values is Fisher's method [Fisher, 1932]. Fisher noted that since a p-value, p_i, has a Uniform distribution, then $-2\log(p_i)$ will have a $\chi^2_{(2)}$ distribution. Using the fact that the sum of independent χ^2 random variables has a χ^2 distribution, the test becomes: reject H_0 if and only if:

$$-2\sum_{i=1}^{k}\log(p_i) \geq C$$

where C is the critical value of a $\chi^2_{(2k)}$ distribution for some particular level α. Again, we see that the target function is the sum of two observable functions.

Active Learning Composite Functions

As we have seen, the target function that we are often interested in learning when combining statistical evidence is a composite of readily available observable functions. In particular, the previous two techniques rely on the sum of observable functions. It is clearly possible to sample all observable functions at each query point and then directly compute the value of the target function, effectively reducing the problem into a standard active learning problem. However, such an approach disregards any strong evidence provided by a single statistical test, and hence may result in extraneous sampling of the remaining statistical models.

Instead, we are interested in active learning algorithms which use information about each observable function to learn some composite target function. We propose a heuristic for actively learning level sets of composite functions of sums for continuous valued input spaces, without arbitrarily restricting the input space (e.g. imposing a grid).

2.4.1 Active Learning Algorithm

Suppose we are given a sample space (or parameter space) $\Theta \subseteq \mathbb{R}^W$ and a set of observable functions $g_i : \Theta \mapsto \mathbb{R}$ $(i = 1, 2, \ldots, k)$, such that $\sum_{i=1}^{k} g_i(\theta) = g(\theta)$, where g is the target

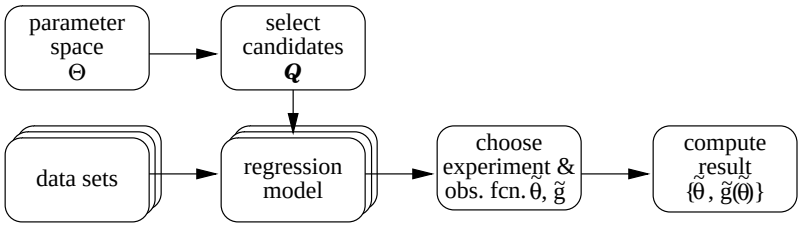


Figure 2.6: Outline of our sampling algorithm. Given an initial set of points (possibly empty), we randomly select a set of candidates, $\mathcal{Q}$, and score them using a set of Gaussian Processes. The best scoring point and observable function pair is chosen, and we evaluate the selected observable function at the given point. This data is added to the corresponding data set. The algorithm is nearly identical to the one given in Section 2.1 and shown in Figure 2.1, except that here we must choose both an experiment point from $\mathcal{Q}$ and an observable function, g_i, on which to perform that experiment.

function we are interested in learning and θ is an arbitrary element of Θ. Here we assume that we do not know the exact form of the observable functions. However, we assume that we can sample from these functions at any point in Θ, just as we did in Section 2.1. Given a threshold t, we want to find the set of points, Θ', where g is equal to or less than the threshold: $\mathcal{V} = \{\theta \in \Theta' | \theta \in \Theta, g(\theta) \leq t\}$. In general, computing the value of each $g_i(\theta)$'s may not incur the same cost. However, here we will assume that the costs are similar, and hence try to minimize the total number of samples of all observable functions, g_i, required to accurately estimate Θ'. Moreover, we assume that g cannot be directly sampled, and that neither g nor any of the g_i's is invertible. That is, the only way to estimate the level-sets of g is to sample points from the g_i's and infer g. This formulation accurately mimics combining p-values using Fisher's method, as the method for finding the individual p-values may be entirely unknown.

The approach we propose for the task of computing level sets of composite functions again leverages the active learning framework discussed in Section 2.1. The main difference between the approach used here and that detailed in Section 2.1, is that here we must not only choose the point $\tilde{\theta} \in \mathcal{Q}$, for the next experiment, but we must also choose the observable function, g_i, for which we will compute $g_i(\theta)$. Ideally, we want to sample the observable function g_i at the point which best increases our prediction accuracy (e.g. whether a point is above or below the threshold) over the combined target function, g. As we have seen before, the parameter space is typically continuous and multi-dimensional, and so we cannot afford to test all possible combination of points and observable functions to find the best pair. Instead, we model each of the observable functions using the current samples taken from that function as a Gaussian Process. Then, for each experiment, we

generate a candidate set $\mathcal{Q}$, as before. Each point in $\mathcal{Q}$ is scored using one of several heuristics, and the best point and observable function pair, $\{\tilde{\theta}, \tilde{g}\}$, is selected. We compute the value of the observable function at the selected point, $\tilde{g}(\tilde{\theta})$, and add it to the data set used to model that function, and the process is repeated. The active learning algorithm is illustrated in Figure 2.6.

2.4.2 Choosing Experiments

Using this greedy selection algorithm, we must now decide how to choose the sample / observable function pairs. We consider the following heuristics:

Random This heuristic selects one of the candidate points and the associated observable function, g_i, uniformly at random. This method serves as a baseline for comparison of the other heuristics.

Variance This heuristic selects the point which has the largest variance on any of the observable functions; our choice then, is the point with the maximum variance, and the observable function on which that variance was found. That is, the variance heuristic select s the candidate point, $\theta \in \mathcal{Q}$ and the observable function g_i, for which $\sigma_i^2(\theta)$ is maximal. Similar to the variance heuristic described in Section 2.2.2, this heuristic will map out the observable functions over the entire parameter space, and hence will learn the target function over the entire space.

Sequential-Straddle As noted in Section 2.4, the problem can be simplified to a standard active learning problem if one sequentially samples each of the observable functions in order to directly compute g. In Section 2.2 we showed that in a setting where experiments yield the (approximately) true values of the target function, the straddle heuristic efficiently identifies function level sets. The Sequential-Straddle heuristic, seq-straddle, leverages the searching ability of the straddle heuristic by choosing the candidate point from $\mathcal{Q}$ with the highest combined straddle score,

$$\text{combined-straddle}(\theta_q) = 1.96 \sum_{i=1}^{k} \sigma_i^2(\theta_q) - \left| \sum_{i=1}^{k} \phi_i(\theta_q) - t \right|, \tag{2.3}$$

and then sequentially sampling all k observable functions at the that point.

Variance-Straddle While Bryan et al. [2005] showed that the straddle heuristic works well when directly sampling the target function, we can hope to do better by considering the output from each observable function individually. For instance, if a sample point results in a very large value for one of the observable functions, it may be unlikely or impossible that the results of the other g_i's will be such that the resulting value of g is near the level-set. In particular, when dealing with the χ^2 models mentioned in the introduction, we know that $g_i(\theta) \geq 0$ for all i. Thus, if a single g_i is greater than the level-set boundary, the target function will also be greater than the level-set boundary, and hence it is likely that sampling elsewhere will be advantageous. The Variance-Straddle heuristic, var-straddle chooses the point, $\tilde{\theta}$ from $\mathcal{Q}$ with the largest combined-straddle score (given in Equation 2.3) and then selects the observable function, $\tilde{g}$, which has the largest variance at $\tilde{\theta}$.

Variance-MaxVarStraddle Finally, we consider a variant of the straddle heuristic. This heuristic tries to mimic the information gain of choosing a particular point and observable function pair. Specifically, note after observing a point, the variance of the Gaussian Process is effectively zero at that point (since we have set ζ to be a very small positive value). Thus, in the single data set case, the straddle heuristic can be seen as balancing the expected gain in the model fit ($\sigma(\theta_q)$) with the expected distance of the point to the level-set boundary. However, with the multiple model formulation, we do not expect the model variance to decrease by $\sigma(\theta_q) = \sum_{i=1}^{k} \sigma_i^2(\theta_q)$, but rather by $\sigma_i(\theta_q)$ where $\tilde{g} = g_i$ is the observable function selected. Thus, a more accurate estimate of the information gain of a candidate point and observable function pair is the Variance-MaxVarStraddle heuristic, var-maxvarstraddle

$$\text{var-maxvarstraddle}(\theta_q) = \max_i \left\{1.96\sigma_i^2(\theta_q)\right\} - \left|\sum_{i=1}^{k} \phi_i(\theta_q) - t\right|.$$

The candidate point, $\theta \in \mathcal{Q}$, and corresponding observable function g_i which maximize this heuristic are chosen as the next experiment.

2.4.3 Experiments

Now, let us assess how well each of the previously mentioned heuristics performs on the task of learning level sets of composite functions, again using classification accuracy as the test metric. Specifically, we compute the fraction of the test points in the predicted model , $\phi = \sum_{i=1}^{k} \phi_i$ and the target function g predict values which are on the same side of the threshold, t, after a fixed number of experiments. For the four following functions,

classification accuracies were assessed after 200 experiments. The sampling and testing process was repeated 20 times to account for variations due to the random nature of the candidate generation process. The first three target functions considered were sums of two observable functions, while the fourth was a sum of four observable functions. The considered functions are:

Gaussian This problem consisted of determining the 95% acceptance region of two axis aligned perpendicular two dimensional Gaussian distributions centered at the origin. Both Gaussians had diagonal covariance matrices with on diagonal elements of 1 and 16. Since working in probability space results in many near-zero values, the problem was considered in log-space. As such, the target function was a 2 dimensional symmetric quadratic function, and the level-set was a circle centered at the origin. The range of the parameter space ($x, y \in [-3.4, 3.4]$)

2D Sine The second problem consists of finding where the two 2D sinusoidal observable functions

$$\begin{aligned} g_1(x,y) &= \sin(10x) + \cos(4y) - \cos(3xy) \\ g_2(x,y) &= \sin(10y) + \cos(4x) - \cos(3xy) \end{aligned}$$

sum to zero where $x, y \in [0, 2]$. The first of these observable functions is identical to the 2D Sine test function used in Section 2.2.3, while the other is simply a rotation of the 2D Sine function. We chose this set of functions for the same reasons outlined in Section 2.2.3. In particular, these functions are interesting due to the discontinuities and extended regions in which they are very close to the boundary (making accurate classification difficult).

2D SimpleSine This problem is a simplified version of the previous problem, where the sinusoidal observable functions

$$\begin{aligned} g_1(x,y) &= \sin(4x) + \cos(4y) - \cos(3xy) \\ g_2(x,y) &= \sin(4y) + \cos(4x) - \cos(3xy) \end{aligned}$$

were chosen to reduce the problem's semi-variances (again $x, y \in [0, 2]$). Since problems with large semi-variances result in large model variance estimates in the kriging models, such problems require extensive sampling to correctly identify function level-sets. The smaller semi-variances of this task allow us to quickly distinguish which heuristics are utilizing the information in the Gaussian process to exclude portions of the sample space from further sampling.

	Gaussian	2D SimpleSine	2D Sine	2D Sine-4
random	> 1000	> 1000	> 1000	> 1000
variance	95.0±11.0	> 500	105.0±11.5	188.6±32.2
seq-straddle	76.2±3.5	150.3±6.5	87.0±7.3	98.1±14.0
var-straddle	89.5±5.0	157.9±12.3	90.4±9.0	72.5±12.0
var-maxvarstraddle	71.7±3.3	127.3±6.8	82.9±10.2	54.9±16.9

Table 2.4: Number of samples required to achieve a 99% classification accuracy on the Gaussian and 2D SimpleSine tests, and a 90% accuracy on the 2D Sine and 2D Sine-4 tests based on 20 trials. The Variance-MaxVarStraddle heuristic consistently performs better than competitors.

2D Sine-4 This task consisted of finding where four 2D sinusoids sum to -2. The sinusoids chosen for this problem are

$$\begin{aligned} g_1(x,y) &= \sin(4x) + \cos(2y) - \cos(3x) \\ g_2(x,y) &= \sin(2y-2) + \cos(2x) - \cos(3x) \\ g_3(x,y) &= \sin(3xy) + \cos(2x) + 1 \\ g_4(x,y) &= \cos(xy) - \sin(xy) \end{aligned}$$

where again $x, y \in [0, 2]$. The resulting target function contains both regions of high slope, as well as regions with low derivatives near the specified threshold. Moreover, there are regions of the input space in which a call to a single observable function would strongly suggest that further samples were unnecessary.

Classification accuracy results for the four tests are given in Table 2.4. Variance-MaxVarStraddle (var-maxvarstraddle) outperforms all of the other heuristics on each of the target functions. Not surprisingly, the straddle-based heuristics beat the random and variance heuristics, as both the random and variance heuristics sample the observable functions over the entire parameter space, while the straddle-based heuristics focus on the level-set of interest. Moreover, Variance-MaxVarStraddle beats out the Sequential-Straddle heuristic (seq-straddle); this validates our supposition that treating each of the observable functions individually allows for additional learning opportunities.

One surprising result of our experimentation is that the Sequential-Straddle performs as well as than the Variance-Straddle heuristic (var-straddle) on all test functions the Gaussian, 2D SimpleSine and 2D Sine tasks. We believe that this result illustrates the fact

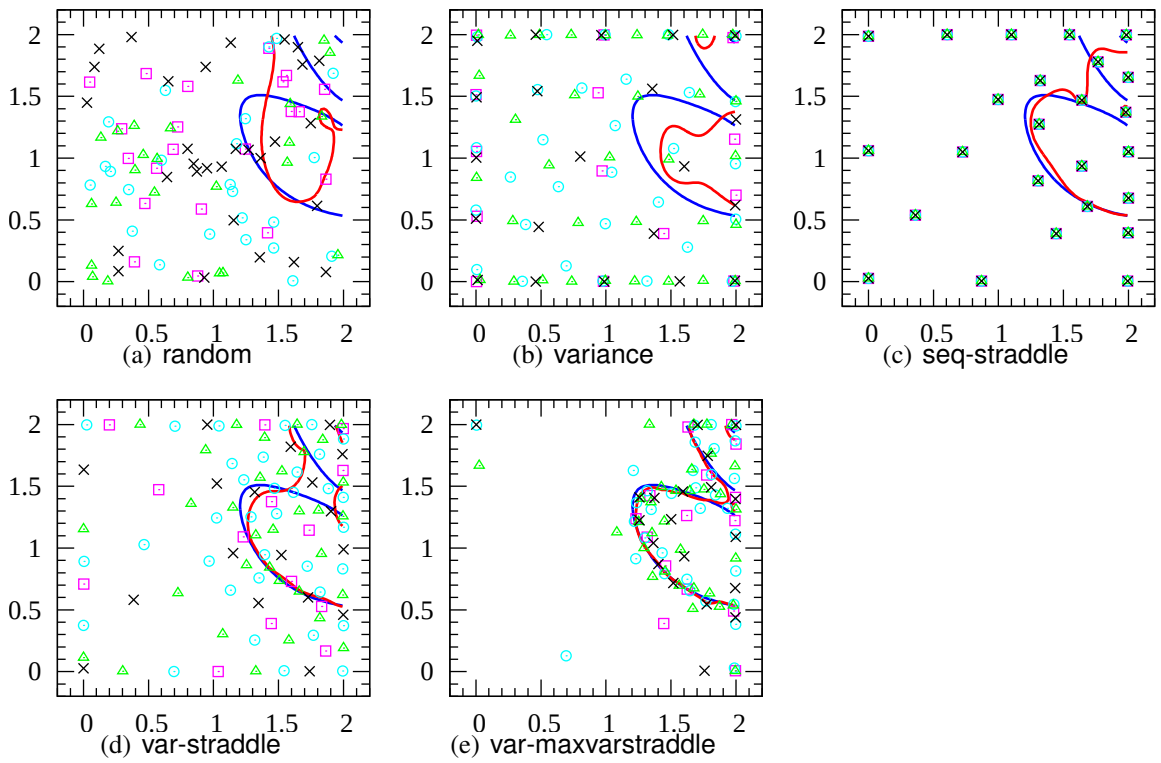


Figure 2.7: Predicted level-set (red), true level-set (blue) and experiments (squares, circle, triangles and $\times$'s) for the 2D Sine-4 function after sampling 100 points using the specified heuristics. The Variance-MaxVarStraddle heuristic outperforms the other heuristics by using information from a single observable function to quickly prune off portions of the parameter space that are extremely unlikely.

that the Variance-Straddle heuristic is over estimating the importance of the variance component of the candidate points to the information gain of a point. The Variance-Straddle heuristic will be as likely to choose a candidate point where one of the two Gaussian process models for the observable functions is large, and the other is zero as it is to choose a point where both estimated variances are equal. This is because the sum of the variances and estimated values for g_i are similar. However, the first candidate has much more information gain than the second, as selecting the first candidate will give us the (approximately) exact value of the target function, while selecting the second will only reduce the overall variance by a moderate amount. On the 2DSine-4 task the Variance-Straddle heuristic is able to make use of the individual observable functions, but still does not do as well as the Variance-MaxVarStraddle heuristic.

To illustrate the differences in sampling patterns between these heuristics, we plot the first 100 samples chosen for the observable functions (with squares, circles, triangles and $\times$'s, respectively) with the true (blue) and predicted (red) function level-sets for the 2D Sine-4 task in Figure 2.7. As seen in Table 2.4, the Variance-MaxVarStraddle heuristic is much better at picking points to aid in the location of the level-set of interest than the other heuristics. In particular, the Variance-MaxVarStraddle heuristic is able to learn that some regions of the space are poor without having to sample all observable functions in those regions. As such, the samples for the Variance-MaxVarStraddle heuristic lie much closer to the target level-set and the prediction accuracy after 100 samples is significantly better. This reinforces our hypothesis that modeling the observable functions separately results in additional learning opportunities.

Thus, our experimental results indicate that Variance-MaxVarStraddle outperforms both the random and variance heuristics typically applied to similar active learning tasks. Moreover, the Variance-MaxVarStraddle heuristic is better than both the Sequential- and Variance-Straddle heuristics, as it appears to better approximate the information gain of a candidate point.

2.5 Learning a Global Optimum

Finally, let us look at the task of learning the global optimum of a target function. As we have seen in Sections 2.2, 2.3 and 2.4, using heuristics which mimic information gain, but do not require the computation cost to compute the entire mutual information over the entire parameter space can be extremely powerful. One may well ask what would happen if we tried to modify the straddle heuristic to compute the global optimum. Now, instead of sampling points that are above a pre-specified boundary value t, we want the heuristic to sample points about the largest value of g currently observed; call this point $\tilde{t}$.[4] Moreover, we are only interested in points which are greater than $\tilde{t}$, so we replace $|\phi(\theta_q) - \tilde{t}|$ with $\tilde{t} - \phi(\theta_q)$, which effectively eliminates those points with large variances that are smaller than the threshold $\tilde{t}$, while emphasizing those points that are predicted to have values greater than $\tilde{t}$. The result is the global optimum heuristic:

$$\text{global-optimum}(\theta_q) = 1.96\sigma + \phi(\theta + q)$$

where the $\tilde{t}$ term was dropped as it is independent of θ_q and hence will not impact the selection process. This heuristic is identical to the IE-MAX heuristic of Moore and Schneider

[4]To eliminate ambiguities, we can select the first point randomly from the parameter space.

[1996], which was shown to work well for tasks such as factory optimization analysis.[5]

2.6 Summary

In this chapter we have described an active learning framework which can be used to learn features of a target function. We have used this framework to show how four different function features can be learned efficiently. Function level-sets can be learned with a fraction of the samples by using the straddle heuristic instead of heuristics which try to learn the target function over the entire parameter space. Moreover, the threshvar can be used to sample a function below a specified threshold t in a manner that both maximizes the number of samples in the subspace of Θ where $g < t$, Θ', and ensures that no regions of Θ' are neglected. Functions which are sums of other observable functions can be learned much more efficiently using the var-maxvarstraddle heuristic than either using the variance heuristic or the straddle over the target function. These results support our intuition: heuristics which focus on learning a specific feature of the target function, g, significantly outperform heuristics which try to learn g over the entire parameter space. In the next chapter we will see how we can harness these algorithms to solve statistical inference problems.

[5]The final heuristic used by Moore and Schneider [1996] added boundaries to ensure that sample experiments did not cause the factory machine to fail; these "soft boundaries" are similar to the constraints imposed by our parameter space, Θ.

Chapter 3

Confidence Region Procedures

In this chapter we discuss several techniques which can be used to compute $1-\alpha$ confidence regions. $1-\alpha$ confidence regions are possibly disjoint sub-regions of our parameter space, Θ, which contain the truth with probability $1-\alpha$. Most of the methods presented here are frequentist in nature, rather than Bayesian. While the selection of a frequentist or Bayesian procedure is often viewed as a matter of taste, it should be noted that the two statistical ideologies answer slightly different questions; we will discuss this issue in Section 3.6.1. As we shall see, frequentist methods are much better suited to active learning search algorithms, as there is no need to normalize the results (to form the posterior) either during or after sampling.

We begin this chapter by formalizing the definition of confidence regions. We then discuss several frequentist techniques to derive confidence regions: χ^2 tests, confidence balls, and the minimax expected size confidence procedure. Additionally, we discuss two ways to compute Bayesian credible intervals. We end this chapter with a comparison of the various confidence region techniques, pointing out their strengths and weaknesses.

3.1 Confidence Regions

We begin by assuming that we are given some data x, which can be modeled as the realization of a stochastic process, μ. The nature of this process is assumed to be known fully except for the values of W input parameters. We label the full vector of parameters θ. Thus, we assume that the observed data — consisting of N pairs (z_n, Y_n) — is given by $Y_n = \mu(z_n, \theta) + \varepsilon_n$, where ε_n is assumed to be Normal with known variance σ_n^2 (for $n = 1, 2, \ldots, N$. Further, we assume that it is known that the true value of the parameters,

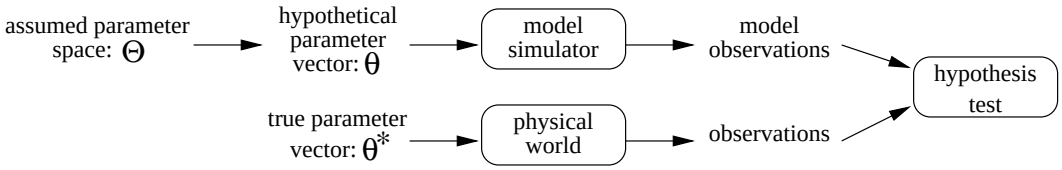


Figure 3.1: Pictorial overview of the statistical procedure to test hypothetical models against the observed data. Note that during this process, we assume that we are given a simulator which faithfully represents the physical world when given the true parameter vector $\theta^\star$.

which we label $\theta^\star$, falls in the space $\Theta \subseteq \mathbb{R}^W$; $\theta^\star$ is an unknown, unobserved random variable. Our goal then, is to take the observations x and reverse engineer the output of the model to determine $\theta^\star$. In general, it is impossible to decisively determine the single $\theta \in \Theta$ which is $\theta^\star$. Instead, we can compute confidence regions — a set of $\theta \in \Theta$ — which contains $\theta^\star$ with high probability.

A $1-\alpha$ confidence region, $\mathcal{C}$, for $\theta^\star$ is a subset of Θ that is produced using an approach known to correctly include the true value of the parameter with probability at least $1-\alpha$. Confidence regions are computed by applying a confidence procedure. A confidence procedure maps the observed data into $\mathcal{C} \subseteq \Theta$, forming one or more contiguous areas in the parameter space. We say that a confidence procedure has coverage probability $1-\alpha$ if, regardless of the true value $\theta^\star$, the probability that the resulting confidence region includes $\theta^\star$ is at least $1-\alpha$; that is $P_{\theta^\star}(\theta^\star \in \mathcal{C}) \geq 1-\alpha$. A confidence procedure is optimal if it is guaranteed to return the smallest possible confidence region which maintains $1-\alpha$ coverage. Optimal confidence procedures are generally known only for simple parametric models. General procedures for forming confidence regions are desirable for use with the complex models used in most scientific applications.

The general strategy for computing confidence regions is depicted in Figure 3.1. Parameter vector samples, θ, are chosen from Θ and then used to compute hypothetical observations given some model of the physical world. These hypothetical observations are compared to the true observations using a statistical test, resulting in a score that the hypothesis and true observations were drawn from the same source. Note that throughout this process, we assume that the simulator faithfully represents the underlying physical processes; all of the statistical inferences made are predicated on this fact. Thus, in the following, we will often refer to the hypothetical observations of our model simulator simply as our "model" given some value $\theta \in \Theta$.

While there are many methods for computing $1-\alpha$ confidence regions given the scores, it is natural to prefer approaches that produce small confidence regions, as small regions

correspond to tighter inferences. However, meaningful optimality results from considering procedures that are constructed without consideration of the observed data at hand. If a confidence procedure is tailored to the observed data x, then optimality is trivial, but meaningless, as a procedure can be designed such that the resulting $1-\alpha$ confidence region is small (or even empty) when x is observed and large otherwise.

For example, consider the decision rule which given the observed data x creates the confidence region, $\mathcal{C}(x)$, such that:

$$\mathcal{C}(x) = \begin{cases} \emptyset & \text{when the data is } x \\ \Theta & \text{otherwise.} \end{cases} \tag{3.1}$$

Since the probability of actually observing x is zero (as the observations could have been anything), the expected size of $\mathcal{C}(x)$ is Θ, even though the size of $\mathcal{C}(x)$ is zero given the observed data.

Instead, we look at confidence procedures which are constructed before the data is observed. These procedures can then be applied to the data to produce non-empty regions which contain the truth with high probability. The regions which are most valuable are those with the smallest size, but do not rely on a specific value of the data, as they are likely to yield tighter inferences on the input parameters of the physical model μ. We will now look at several techniques to compute $1-\alpha$ confidence regions.

3.2 χ^2 Tests

Often, χ^2 tests are used to produce $1-\alpha$ confidence regions due to the ease in which they can be both implemented and interpreted. In their simplest form, one fixes a candidate parameter value $\theta \in \Theta$ and assesses the fit of the data to the assumed model under parameter value θ using a sum of squares criterion. Typically one computes a variance weighted sum of squares between the test model and the data (where the variance at each point is given by error estimates from the data). However, this formulation assumes that the data points are independent, which is often incorrect. Instead, we can treat the data as coming from a multivariate Normal, and compute the sum of squares as $(x-\mu)\Sigma^{-1}(x-\mu)^T$, where x is our observed data, μ is our computed model, and Σ is a known covariance matrix for the data. In either case, we can the compare this sum of squares to the appropriate χ^2 distribution to assess if the deviation from expected is large enough to rule out that model at significance level α. If this is repeated for all $\theta \in \Theta$ the set of accepted parameter vectors will be a $1-\alpha$ confidence region for $\theta^\star$.

However, it is well known that such an approach is conservative [Wasserman, 2004] because it does not incorporate the available information regarding the full parameter space: In particular, some deviations from "expected" are due to noise, while others due to the fact that $\theta \neq \theta^\star$. By ignoring this distinction, χ^2 confidence regions are usually larger than necessary, resulting in sub-optimal inference.

3.2.1 Efficiently Computing χ^2 Confidence Regions

As mentioned in the previous section, exact computation of $1-\alpha$ confidence regions using χ^2 tests requires testing all $\theta \in \Theta$ to determine if the variance-weighted sum of squares is less than the $\chi^2_{(N)}$ statistic when the coverage is equal to $1-\alpha$. Since Θ is typically large or infinite, this is impossible. Instead we use the active learning framework of Section 2.1 to compute the $1-\alpha$ confidence regions.

Specifically, note that the task we are interested in is that of discovering $\Theta' = \{\theta \in \Theta | g(\theta) \leq t\}$, where here g is the function mapping parameter vectors through the model μ to the variance-weighted sum of squares, and t is the $\chi^2_{(N)}$ statistic which ensures that the coverage is $1-\alpha$. Thus, we could use the threshvar heuristic discussed in Section 2.3 to sample $\mathcal{V}$. However, a more efficient approach is to use the straddle heuristic described in Section 2.2, as we do not need samples from the confidence regions' interiors to determine their extent. If we can accurately discover the (possibly disjoint) boundaries of Θ', then we can determine the range of any of the input parameters either individually, or in combination with a subset of the other parameters.

3.3 Confidence Balls

Alternatively, Genovese et al. [2004] proposed the idea of confidence balls, constructed by first fitting the observed data non-parametrically and then comparing a proposed model to this nonparametric fit. This approach can be viewed as a generalization of the classic approach of reducing the full data down to maximum likelihood estimates of the parameters, and then using the approximate distribution of the maximum likelihood estimate to form a confidence region. Confidence balls correct for the power loss of χ^2 tests by mimicking the underlying function with the nonparametric fit, thereby reducing the noise inherent in the data. However as we shall see, computing the radius of the confidence ball is non-trivial, as it relies on both the fit to the data as well as the observational error.

3.3.1 The Non-Parametric Fit

Suppose we were given data in the form $\{z_\ell, Y_\ell\}$, for $\ell = L_{\min}, \ldots, L_{\max}$. That is, we suppose that the data is equally spaced along the z axis.[1] Let $N = L_{\max} - L_{\min} + 1$ be the total number of observed data points. We take $Y_\ell = \hat{C}_\ell$ to be the observations where $z_\ell = (\ell - L_{\min})/(L_{\max} - L_{\min})$ and let $\mu(z_\ell, \theta^\star) \equiv C_\ell$ denote the true value of the model at z_ℓ.

We then solve the nonparametric regression problem:

$$Y_\ell = \mu(z_\ell) + \varepsilon_\ell, \qquad \ell = L_{\min}, \ldots, L_{\max}, \tag{3.2}$$

where $\varepsilon = (\varepsilon_{L_{\min}}, \ldots, \varepsilon_{L_{\max}})$ are assumed Gaussian with known covariance matrix Σ. Henceforth, we will use $i = \ell - L_{\min} + 1$ as an index. Nonparametric analysis is based on the notion of estimating a function without forcing it to fit some finite-dimensional parameteric form, by smoothing the data in such a way to balance the bias and variance. In this work, we use orthogonal series regression to estimate μ, expanding μ as a cosine basis:

$$\mu(z) = \sum_{j=0}^{\infty} \rho_j \phi_j(z)$$

where

$$\phi_j(z) = \begin{cases} 1 & \text{for } j = 0 \\ \sqrt{2}\cos(\pi j z) & \text{for } j = 1, 2, 3, \ldots \end{cases}$$

and the ρ_j's are the coefficients for each basis component. If μ is smooth, which is common for many physical models, then ρ_j will decay rapidly as j increases. That is, if μ is smooth, then there are little or no high frequency fluctuations in μ and hence $\rho_j \simeq 0$. Thus, $\sum_{j=N+1}^{\infty} \rho_j^2$ will be negligible, and we can approximate the infinite sum as $\mu(z) \approx \sum_{j=0}^{N} \rho_j \phi_j(z)$. Let

$$\mathcal{Z}_j = \frac{1}{n} \sum_{i=1}^{N} Y_i \phi_j(z_i)$$

for $j = 0, 1, \ldots N$. Then $\mathcal{Z}$ is approximately Normally distributed with mean ρ and covariance $B/\sqrt{N} = U\Sigma U^T/\sqrt{N}$, where U is the cosine basis transformation matrix.

In order to obtain an even smoother estimate of μ, we damp out the higher frequencies using shrinkage estimators. We let $\hat{\rho}_j = \lambda_j \mathcal{Z}_j$ where $1 \geq \lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_N \geq 0$ are

[1] This assumption can be removed by performing a QR-factorization of the matrix ϕ.

shrinkage coefficients. The estimate of μ is now

$$\hat{\mu}(z) = \sum_{j=0}^{N} \hat{\rho}_j \phi_j(z) = \sum_{j=0}^{N} \lambda_j \mathcal{Z}_j \phi_j(z).$$

Following Genovese et al. [2004], we use a special case of monotone shrinkage in which

$$\lambda_j = \begin{cases} 1 & \text{for } j \leq J \\ 0 & \text{for } j > J \end{cases}$$

for some integer $J \in [0, N]$. We will show how to find J shortly. Using the monotone shrinkage scheme described above, the estimate of μ becomes

$$\hat{\mu}(z) = \sum_{j=0}^{J} \mathcal{Z}_j \phi_j(z).$$

The squared error loss as a function of $\hat{\lambda} = (\hat{\lambda}_0, \hat{\lambda}_1, \ldots, \hat{\lambda}_N)$ is

$$L_n(\hat{\lambda}) = \int_0^1 \left(\frac{\hat{\mu}(z) - \mu(z)}{\sigma(z)} \right)^2 dx \approx \sum_{j=1}^{N} \left(\frac{\rho_j - \hat{\rho}_j}{\sigma_j} \right)^2,$$

where $\sigma^2(z)$ is the variance of μ, and σ_j^2 are the observed variances of the power spectrum (the elements on the diagonal of Σ). Meanwhile, the risk is given by

$$R(\lambda) = \mathsf{E}\left[\int_0^1 \left(\frac{\hat{\mu}(z) - \mu(x)}{\sigma(z)} \right)^2 dx \right] \approx \frac{J}{N} + \sum_{j=J}^{N} \frac{\rho_j^2}{\sigma_j^2}$$

We choose J to minimize the Stein's unbiased risk estimate

$$\hat{R} = \mathcal{Z}^T \bar{D} W \bar{D} \mathcal{Z} + \text{trace}(DWDB) - \text{trace}(\bar{D} W \bar{D} B) \tag{3.3}$$

where D and $\bar{D} = 1 - D$ are diagonal matrices with 1's in the first J and last $N - J$ entries respectively, B is the covariance of $\mathcal{Z}$, and $W_{jk} = \sum_\ell \Delta_{jk\ell} / \sigma_\ell$ and

$$\begin{aligned} \Delta_{jk\ell} &= \int_0^1 \phi_j \phi_k \phi_\ell \\ &= \begin{cases} 1 & \text{if } \#\{j,k,l=0\} = 3 \\ 0 & \text{if } \#\{j,k,l=0\} = 2 \\ \delta_{jk}\delta_{0\ell} + \delta_{j\ell}\delta_{0k} + \delta_{k\ell}\delta_{0j} & \text{if } \#\{j,k,l=0\} = 1 \\ \frac{1}{\sqrt{2}}(\delta_{\ell,j+k} + \delta_{\ell,|j-k|}) & \text{if } \#\{j,k,l=0\} = 0 \end{cases}. \end{aligned}$$

Beran and Dümbgen [1998] showed that $\hat{R}(\lambda)$ is asymptotically, uniformly close to $R(\lambda)$ when using monotone shrinkage coefficients and $\sigma(z) = 1$. Genovese et al. [2004] extended this result to the heteroskedastic case used here.

3.3.2 The Confidence Ball

After we perform the non-parametric fit, we need to quantify the uncertainty to make statistical inferences. We use the Beran-Dümbgen pivot method [Beran and Dümbgen, 1998, Beran, 2000] to derive valid confidence intervals. This method relies on the weak convergence of the "pivot process" — $B_N(\hat{\lambda}) = \sqrt{N}(L_N(\hat{\lambda}) - \hat{R}(\hat{\lambda}))$ — to a Normal $(0, \tau^2)$ distribution for some $\tau^2 > 0$; a derivation of $\hat{\tau}_N$ can be found in Appendix A, taken from Appendix 3 of Genovese et al. [2004]. Using the convergence of the pivot process, we can compute a confidence ellipse for the basis coefficients with a "radius" given by:

$$\mathcal{D}_N = \left\{ \rho \,\middle|\, \sum_{i=1}^{N} \left(\frac{\hat{\rho}_i - \rho_i}{\sigma_i} \right)^2 \leq \frac{\hat{\tau}_N \, z_\alpha}{\sqrt{N}} + \hat{R}(\hat{\lambda}_N) \right\} \tag{3.4}$$

where the best fit to the data is represented by $\hat{\rho}_i$, the function being tested (whether it is within some confidence ball) is ρ_i, and the level of the confidence ball is determined by z_α, the upper α quantile of a standard Normal distribution.

Therefore, using the central limit theorem, we have

$$\mathcal{B}_N = \left\{ \mu(z) = \sum_{j=0}^{N} \rho_j \phi_j(z) \,\middle|\, \rho \in \mathcal{D}_N \right\} \tag{3.5}$$

is an asymptotic $1 - \alpha$ confidence region for μ.

By comparing the model to a fit of the data as opposed to the data themselves, the confidence ball technique is centered (approximately) on the true underlying function, μ, as opposed to the noisy realization, $\hat{\mu}$. The implication is that it is less affected by noise in the data. In particular, we have observed that χ^2 tests will reject all possible models in cases where there is a single outlier 4σ from the maximum likelihood estimate fit. By initially smoothing the data, and then fitting this smoothed estimate, we are much less susceptible to errors caused by noisy outliers.

Additionally, since the confidence ball method uses a smooth realization of the data, the size of the radius computed using the pivot process is smaller than the χ^2 radius, as shown in Figure 3.2. This is a result of the fact that the Gaussian errors of all points are considered as an ensemble, not individually as with χ^2 tests. This allows us to reject more points in the parameter space, and subsequently return tighter bounds on the confidence ball; that is, the confidence ball test has more statistical power than does the χ^2 test. A comparison of the relative widths of the confidence and χ^2 balls is shown in Figure 3.2. Note that as the confidence ball radius increases, so does the size of the confidence region (and α decreases). Thus, a 95% (or $\alpha = 0.05$) confidence region has a larger "radius" than

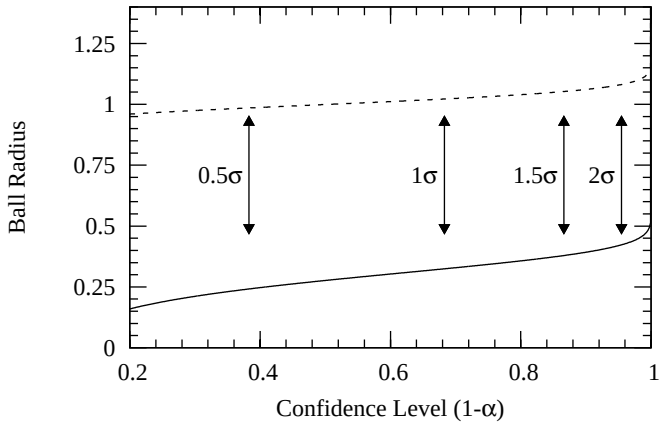


Figure 3.2: Radius of the non-parametric confidence ball as a function of confidence level (solid). The reduced χ^2 ball is shown for comparison (dashed). Arrows depict $\frac{1}{2}, 1, 1\frac{1}{2}$ and 2σ respectively.

does a 67% (or $\alpha = 0.33$) confidence region. Moreover, a $1 - \alpha$ confidence ball strictly contains all confidence balls with smaller values of $1 - \alpha$.

3.3.3 Efficiently Computing Confidence Ball Confidence Regions

As with χ^2 tests, the confidence ball techniques provides a method to test whether a particular $\theta \in \Theta$ can be rejected given the observed data. This can be done by computing $\mu(\theta)$ for any $\theta \in \Theta$, and then checking whether $\mu(\theta)$ is in $\mathcal{B}_N$. From Equation 3.4, checking whether $\mu(\theta) \in \mathcal{B}_N$ is equivalent to determining if the variance-weighted sum of squares of $\hat{\rho}$ and ρ is less than a constant given on the right-hand side of Equation 3.4. Thus, determining $1 - \alpha$ confidence regions using the confidence ball procedure results in a computational process that is very much like that used for the χ^2 tests; we must compute a variance-weighted sum of squares between the model and the non-parametric fit to the data, and then compare this sum of squares to a cutoff value, which given in Equation 3.4.

Compared with χ^2 tests, the confidence ball approach is only marginally more computationally expensive. The additional costs incurred by the confidence ball procedure are the computation of the non-parametric fit to the data and the derivation of the threshold based upon τ_N. These calculations need only be computed once for each data set and take only a couple of seconds. Thus, in general, the majority of computation used by both the χ^2 tests and the confidence ball procedure is in computing test models $\mu(\theta)$. This expense can be

minimized by efficiently choosing samples to locate the confidence region boundary.

Therefore, we use essentially the same algorithm described in Section 3.2.1, to compute the confidence ball $1-\alpha$ confidence regions. The only required changes are replacing $g(\theta)$ and t with their confidence ball counterparts. With these minor changes, we can then use the straddle heuristic in the active learning framework of Section 2.1 to compute the extent of the $1-\alpha$ confidence regions.

3.4 Minimax Expected Size Confidence Procedure

The third method for computing $1-\alpha$ confidence regions that we will look at is the Minimax Expected Size (MES) confidence procedure [Evans et al., 2005, Schafer and Stark, 2006].[2] This procedure actively tries to minimize the maximum expected size of the derived confidence region. Recall from Section 3.1, that small confidence regions, constructed by procedures which do not rely on the observed data, are the most useful as they impose tighter constraints on the model input parameters. While the MES procedure guarantees to minimize the maximum expected size, in general, the observed size is also approximately minimized, resulting in near optimal confidence regions.

Conceptually, the expected size of a confidence region equals

$$\sum_{x} \text{actual size of region when data is } x \times P(\text{observed data is } x)\,. \tag{3.6}$$

The expected size of the confidence region turns out to be a natural optimality criterion. If a confidence procedure simultaneously minimizes the actual size for all possible data values x, then that procedure will also minimize the expected size. However, such procedures will not exist in most situations. Looking more carefully at Equation 3.6, we note that the probability the observed data is x is, in fact, a function of the unknown true distribution. We seek the $1-\alpha$ confidence procedure that minimizes the maximum (over $\theta \in \Theta$) expected size of the confidence region.

The discussion of confidence regions in Section 3.1 alluded to a trade-off that must be made when constructing confidence procedures: region size versus coverage probability. In particular, a procedure which sets $\mathcal{C} = \Theta$ for all x is a trivial $1-\alpha$ confidence procedure for all values of α (as we assume that $\theta^\star \in \Theta$); however this confidence region is clearly of little use. Conversely, the set $\{\theta\}$ for some $\theta \in \Theta$ is a minimally sized confidence

[2]This work was originally published in Bryan et al. [2007a] and Bryan et al. [2007d] with co-authors H. Brendan McMahan, Chad M. Schafer, and Jeff Schneider.

region with probability zero of covering the true value. Intuitively this trade off can be thought of as an adversarial two person game. Indeed, in Section 3.4.3 we will show how the MES confidence procedure can be formulated as such a game. We now present the background and notation describing these games: matrix games and their generalization, convex games.

3.4.1 Game Theory Background

In this section we describe first matrix and then convex games. While there has been much research in this area, we present only the core concepts and ideas necessary to understand their relationship with and use within the MES confidence procedure. Interested readers are referred to Dresher and Karlin [1953], Osborne [2003], McMahan [2006].

Matrix Games

A zero-sum matrix (normal-form) game is played by two players, player row with strategies $R = \{1, \ldots, I\}$ and player column with strategies $C = \{1, \ldots, J\}$. An $I \times J$ matrix $\mathbf{A}$ specifies the payoffs. If row plays strategy $i \in R$ and column plays $j \in C$, the payment from column to row is the (i, j)th entry of $\mathbf{A}$, denoted a_{ij}. The players select their strategies simultaneously, without knowledge of the other player's choice.

We use $\Delta(\cdot)$ to denote the probability simplex over a finite set. For example, the probability simplex over the row player's strategy set is

$$\Delta(R) = \left\{ \mathbf{y} \in \mathbb{R}^I \Big| \sum_{i=1}^{I} y_i = 1 \text{ and } y_i \geq 0 \right\}.$$

A mixed strategy is an element $\mathbf{y} \in \Delta(R)$ for the row player or $\mathbf{z} \in \Delta(C)$ for the column player, corresponding to a distribution over the rows or columns, respectively. If the players select mixed strategies $\mathbf{y}$ and $\mathbf{z}$, the expected payoff $V(\mathbf{y}, \mathbf{z})$ from column to row is given by the bilinear form $\mathbf{y}^T \mathbf{A} \mathbf{z}$. A solution to the game is a minimax equilibrium $(\mathbf{y}^*, \mathbf{z}^*)$, a pair of strategies such that neither player has an incentive to play differently given that the other player selects their strategy from the pair. The minimax theorem [von Neumann and Morgenstern, 1944] states that if the players are allowed to select mixed strategies, there is no advantage to playing second:

$$\max_{\mathbf{y} \in \Delta(R)} \min_{\mathbf{z} \in \Delta(C)} \mathbf{y}^T \mathbf{A} \mathbf{z} = \min_{\mathbf{z} \in \Delta(C)} \max_{\mathbf{y} \in \Delta(R)} \mathbf{y}^T \mathbf{A} \mathbf{z}. \tag{3.7}$$

Thus, solving either the min max or max min optimization problem from (3.7) results in a minimax equilibrium for the matrix game. This problem can easily be converted to a linear program and solved via standard techniques.

An ϵ-approximate minimax equilibrium for a matrix game is a pair of strategies $(\mathbf{y}', \mathbf{z}')$ where neither player can gain more than ϵ value by switching to some other strategy. That is,

$$\max_{\mathbf{y} \in \Delta(R)} V(\mathbf{y}, \mathbf{z}') - \epsilon \leq V(\mathbf{y}', \mathbf{z}') \leq \min_{\mathbf{z} \in \Delta(C)} V(\mathbf{y}', \mathbf{z}) + \epsilon.$$

If $\epsilon = 0$, then $(\mathbf{y}', \mathbf{z}')$ is an exact minimax equilibrium.

Convex Games

Two-player zero-sum bilinear-payoff convex games (simply "convex games" for the sequel) are a natural generalization of matrix games.[3] This formulation was first introduced by Dresher and Karlin [1953], but despite the generality of the framework, convex games have received remarkably little treatment in the literature. Convex games allow arbitrary convex sets Y and Z in place of the probability simplexes $\Delta(R)$ and $\Delta(C)$ for matrix games. A convex game is specified by a tuple $(\mathbf{A}, Y, Z)$ where $Y \subseteq \mathbb{R}^I$ and $Z \subseteq \mathbb{R}^J$ are the strategy sets for the two players, and $\mathbf{A}$ is a $I \times J$ payoff matrix. The first player (who we will call y) selects an action $\mathbf{y} \in Y$, the second player (called z) simultaneously chooses $\mathbf{z} \in Z$ and the payoff from player z to player y is given by $V(\mathbf{y}, \mathbf{z}) = \mathbf{y}^T \mathbf{A} \mathbf{z}$. The concepts of equilibria and ϵ-approximate equilibria naturally generalize to convex games and it can be shown that the minimax theorem still holds (cf. McMahan [2006]).[4]

If the convex set Y is defined by a finite number of linear equality and inequality constraints, then the convex set Y is a polyhedron. If both Y and Z are polyhedra, then we say that the convex game $\mathcal{G} = (\mathbf{A}, Y, Z)$ is polyhedral. Polyhedral convex games can be solved in polynomial time via linear programming, as shown by Koller et al. [1994] in the context of extensive-form games. If Y is a bounded polyhedron and $\mathcal{E}(Y)$ is the set of extreme points (corners) of Y, then there is a natural mapping between $\mathcal{E}(Y)$ and R. When $\mathcal{G}$ is polyhedral, the sets Y and Z correspond exactly to the sets of possible mixed strategies in a certain matrix game. Whereas in the matrix game the mixed strategies are only implicitly considered, in the convex game formulation their representation is explicit.

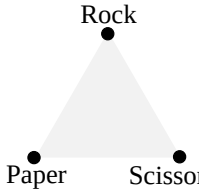
As an illustration, consider the matrix and convex game representations of "Rock-Paper-Scissors". Both representations use the same payoff matrix shown in Figure 3.3(a).

[3] Our convex games are non-cooperative and are unrelated to the super-modular coalitional games often called convex games in the cooperative game theory literature.

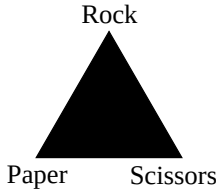
[4] Some mild technical assumptions are required.

	R	**P**	**S**
Rock	0	-1	1
Paper	1	0	-1
Scissors	-1	1	0

(a) Payoff matrix of "Rock-Paper-Scissors".



(b) Pure strategies in the matrix game representation (dots), along with implicit mixed strategies (shaded triangle).



(c) The convex game representation (solid triangle) explicitly includes both pure and mixed strategies.

Figure 3.3: The payoff matrix for "Rock-Paper-Scissors" (a), along with the matrix (b) and convex (c) representations of the players' strategy sets.

However, while the matrix game explicitly considers only the pure player strategies (e.g., always play "rock") shown as the dots in Figure 3.3(b), the convex game represents all possible strategies from which a player can choose, as illustrated in Figure 3.3(c).

While convex games are a simple generalization of matrix games, the ability to represent arbitrary convex strategy sets lets us take advantage of structure in many types of games, often yielding exponentially smaller representations. Notable examples include cost-paired Markov decision process games, extensive-form games (including poker), and the problem of computing an optimal oblivious routing [McMahan, 2006]. In the following two sections, we shall see that the MES confidence procedure can also be formulated as a convex game.

3.4.2 Formulating the MES Confidence Procedure

As in Section 3.1, we denote the set of possible model parameter settings as Θ ($\Theta \subseteq \mathbb{R}^W$), and let θ and $\tilde{\theta}$ be arbitrary members of Θ. For each $\theta \in \Theta$, there is a distribution P_θ on the space of possible observations $\mathcal{X} \subseteq \mathbb{R}^N$. Let X be a random variable and x be a generic observation of X. The distribution of X will be made clear by the context; for instance $P_\theta(X \in U)$ denotes the probability that X falls in the set U when it has distribution P_θ. Assume each distribution P_θ has a density $f(x|\theta)$ with respect to Lebesgue measure. We are interested in constructing a confidence region for the true value of the parameter, denoted $\theta^\star$, based on the observation that $X = x$ and the *a priori* constraint that $\theta^\star \in \Theta$.

Consider testing the hypothesis that $\theta^\star = \tilde{\theta}$ at level α for some arbitrary $\tilde{\theta} \in \Theta$. The associated acceptance region for the test, $\mathcal{A}(\tilde{\theta}) \subset \mathcal{X}$, is the set of data values for which the test will not reject the hypothesis $\theta^\star = \tilde{\theta}$. Since we are interested in tests with significance

level α, we require $P_{\tilde{\theta}}(X \in \mathcal{A}(\tilde{\theta})) \geq 1 - \alpha$.

The power of the test is the probability that the test correctly rejects the hypothesis that $\theta^\star = \tilde{\theta}$. Thus, power is a function of the true value of the parameter. Define the power function as

$$\beta(\theta, \tilde{\theta}) \equiv 1 - P_\theta(X \in \mathcal{A}(\tilde{\theta})).$$

The test has significance level α, so $\beta(\tilde{\theta}, \tilde{\theta}) \leq \alpha$. We are interested in small (i.e., precise) confidence regions and we see below that this is equivalent to choosing $\mathcal{A}(\tilde{\theta})$ to maximize $\beta(\theta, \tilde{\theta})$ over all θ subject to $\beta(\tilde{\theta}, \tilde{\theta}) \leq \alpha$.

The above can be repeated for all $\tilde{\theta} \in \Theta$ and the result is a family of acceptance regions $\mathcal{A}(\tilde{\theta})$. Inverting the hypothesis test $\theta^\star = \tilde{\theta}$ using acceptance region $\mathcal{A}(\tilde{\theta})$, gives us a confidence procedure, $d : \Theta \times \mathcal{X} \mapsto \{0, 1\}$, allowing us to compute the confidence region $\mathcal{C}_\mathcal{A}$. The confidence procedure d is defined as

$$d(\tilde{\theta}, x) = \begin{cases} 1, & \text{if } x \in \mathcal{A}(\tilde{\theta}) \\ 0, & \text{if } x \notin \mathcal{A}(\tilde{\theta}) \end{cases} .$$

Thus, we can either discuss the choice of $\mathcal{A}(\tilde{\theta})$ for all $\tilde{\theta}$, or the choice of d; in what follows it will be more natural to think of selecting d. Using the rule d,

$$\mathcal{C}_d(x) \equiv \{\tilde{\theta} \in \Theta | d(\tilde{\theta}, x) = 1\}.$$

is a $1 - \alpha$ confidence region for $\theta^\star$ based on the observed data x.

We wish to minimize the expected size of $\mathcal{C}_d(X)$ by choosing the confidence procedure d judiciously, since a small confidence set implies high precision in the estimate. However, since we do not know $\theta^\star$, we cannot choose our confidence procedure d to guarantee that $\mathcal{C}_d(x)$ includes the true value of the parameter. Instead we choose d to ensure that $P_\theta(\theta \in \mathcal{C}_d(X)) \geq 1 - \alpha$ for all $\theta \in \Theta$.

Define $\nu(\mathcal{C}_d(x))$ to be the size of $\mathcal{C}_d(x)$ using a measure ν:

$$\nu(\mathcal{C}_d(x)) = \int_\Theta d(\tilde{\theta}, x) \nu(d\tilde{\theta})$$

Any measure that is defined on a broad enough class of subsets of Θ that $\mathcal{C}_d(x)$ is ν-measurable for any value of x is permissible. In this paper we will assume ν is a Euclidean measure over the parameter space, although other choices could be justified. Choosing d to minimize $\nu(\mathcal{C}_d(x))$ for fixed data x' is trivial: simply define $d(\tilde{\theta}, x)$ as

$$d(\tilde{\theta}, x) = \begin{cases} 0 & \text{if } x = x' \\ 1 & \text{otherwise.} \end{cases}$$

This decision rule results in the confidence region given in Equation 3.1. Using the data to define the decision rule is equivalent to "data snooping" and is not statistically valid. Instead, we seek to choose d to make the expected size of the confidence region ($\mathsf{E}_\theta[\nu(\mathcal{C}_d(X))]$) small for all possible truths $\theta \in \Theta$.

Unfortunately, it is not usually true that a single d simultaneously minimizes $\mathsf{E}_\theta[\nu(\mathcal{C}_d(X))]$ over all θ. Instead, consider minimizing the weighted average

$$\mathcal{S}(\pi, d) \equiv \int_\Theta \mathsf{E}_\theta[\nu(\mathcal{C}_d(X))]\pi(d\theta), \tag{3.8}$$

with the weighting provided by a probability measure π defined on Θ. Pratt's theorem [Pratt, 1961] states

$$\mathsf{E}_\theta[\nu(\mathcal{C}_d(X))] = \int_\Theta (1 - \beta(\theta, \tilde{\theta}))\, \nu(d\tilde{\theta}).$$

This link between expected size and power allows us to apply the classic Neyman-Pearson lemma [Neyman and Pearson, 1933] to find the d that minimizes Equation 3.8: set $d(\tilde{\theta}, x) = 1$ if and only if $T_\pi(\tilde{\theta}, x) \leq c_{\tilde{\theta}}$ where

$$T_\pi(\tilde{\theta}, x) \equiv \frac{\int_\Theta f(x|\theta)\pi(d\theta)}{f(x|\tilde{\theta})} \tag{3.9}$$

and $c_{\tilde{\theta}}$ is a cutoff chosen large enough to ensure d has $1-\alpha$ coverage. Call this confidence procedure d_π.

The selection of the measure π is subjective, but there is a particular choice which can be justified using statistical decision theory. Let π_0 be a π that maximizes $\mathcal{S}(\pi, d_\pi)$. This π_0 is not necessarily unique, but Evans et al. [2005] show that, for any choice of decision procedure d,

$$\mathsf{E}_\theta[\nu(\mathcal{C}_d(X))] \geq \mathcal{S}(\pi_0, d_{\pi_0}) \quad \text{for some } \theta \in \Theta.$$

In other words, $\mathcal{S}(\pi_0, d_{\pi_0})$ is the smallest that the worst-case expected size could be. In what follows, we will see that we can use game theory and Monte Carlo simulations to construct π such that

$$\widehat{\mathsf{E}}_\theta[\nu(\mathcal{C}_{d_{\pi_0}}(X))] \leq \mathcal{M}$$

for all $\theta \in \Theta'$, where

$$\widehat{\mathsf{E}}_\theta[\nu(\mathcal{C}_{d_{\pi_0}}(X))] \approx \mathsf{E}_\theta[\nu(\mathcal{C}_{d_{\pi_0}}(X))],$$
$$\mathcal{M} \approx \mathcal{S}(\pi_0, d_{\pi_0}),$$

and Θ' is a finite approximation to Θ. The quality of these approximations improves as the size of the Monte Carlo simulations grow. In essence, Monte Carlo simulations

will be utilized to construct a finite-dimensional problem which well-approximates the full problem. This finite approximation can then be solved by utilizing techniques from game theory to find the desired MES confidence procedure which otherwise would be intractable. The following section provides detail.

3.4.3 Monte Carlo Approximations

Note that

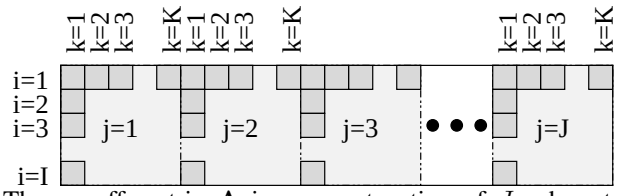
$$\begin{aligned}
\mathcal{S}(\pi, d) &= \int_\Theta \mathsf{E}_\theta[\nu(\mathcal{C}_d(X))]\pi(d\theta) \\
&= \int_\Theta \int_X \nu(\mathcal{C}_d(x)) f(x|\theta)\ dx\ \pi(d\theta) \\
&= \int_\Theta \int_X \int_\Theta d(\tilde{\theta}, x) f(x|\theta)\ \nu(d\tilde{\theta})\ dx\ \pi(d\theta) \\
&= \int_X \int_\Theta d_\pi(\tilde{\theta}, x) \frac{\int_\Theta f_\theta(x)\pi(d\theta)}{f_{\tilde{\theta}}(x)} f_{\tilde{\theta}}(x)\nu(d\tilde{\theta})dx \qquad (3.10)
\end{aligned}$$

We will approximate the integrals in the previous equation with finite samples. For instance,

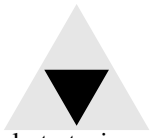
$$\int_\Theta \mathsf{E}_\theta[\nu(\mathcal{C}_d(X))]\pi(d\theta) \approx \sum_{i=1}^{I} \mathsf{E}_{\theta_i}[\nu(\mathcal{C}_d(X))]\pi(d\theta_i)$$

where $\theta_1, \theta_2, \ldots, \theta_I$ are chosen uniformly from Θ. Next, we approximate the integral $\nu(d\tilde{\theta})$ as the sum over values $\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_J$ sampled uniformly from Θ. Finally, for each $\tilde{\theta}_j$, a sample of K data values is simulated from distribution $P_{\tilde{\theta}_j}$ and labeled $x_{j1}, x_{j2}, \ldots, x_{jK}$. These are used in a Monte Carlo approximation to the integral against $f(x|\tilde{\theta}_j)$. We will replace the first integral with a discrete approximation of the expected value over X using K points. Thus,

$$\begin{aligned}
\mathcal{S}(\pi, d) &= \int_\Theta \mathsf{E}_\theta[\nu(\mathcal{C}_d(X))]\pi(d\theta) \\
&\approx \int_X \sum_{j=1}^{J} d_\pi(\tilde{\theta}_j, x) \frac{\sum_{i=1}^{I} f_{\theta_i}(x)\pi(\theta_i)}{f_{\tilde{\theta}_j}(x)} f_{\tilde{\theta}_j}(x) \\
&\approx \frac{1}{JK} \sum_{k=1}^{K} \sum_{j=1}^{J} \sum_{i=1}^{I} d(\tilde{\theta}_j, x_{jk}) \left(\frac{f(x_{jk}|\theta_i)}{f(x_{jk}|\tilde{\theta}_j)} \right) \pi(\theta_i), \qquad (3.11)
\end{aligned}$$



(a) The payoff matrix $\mathbf{A}$ is a concatenation of J sub-matrices, where each sub-matrix corresponds to a single sample point. Rows of the sub-matrices correspond to the same I prior points, while the columns correspond to K different simulations of the sub-matrix sample j.



(b) Pictorial strategies sets (Y and Z) for players y (light triangle) and z (dark triangle), respectively. Player y chooses strategies from an ordinary probability simplex, while player z must choose a decision rule, a subset of the probability simplex (when normalized by $1/K$.) Here $K = 3, \alpha = 0.33$.

Figure 3.4: Payoff matrix and player strategies for the convex game formulation of the MES confidence procedure.

where $x_{j1}, x_{j2}, \ldots, x_{jK} \sim P_{\tilde{\theta}_j}$. Equation 3.11 can be written compactly as

$$\mathcal{S}(\pi, d) \approx \frac{1}{JK} \pi^T \mathbf{A} \mathbf{d},$$

where

$$\begin{aligned} \mathbf{d}^T &= [\mathbf{d}_1^T, \mathbf{d}_2^T, \ldots, \mathbf{d}_J^T] \\ \mathbf{d}_j^T &= [d(\tilde{\theta}_j, x_{j1}), d(\tilde{\theta}_j, x_{j2}), \ldots, d(\tilde{\theta}_j, x_{jK})] \\ \pi^T &= [\pi(\theta_1), \pi(\theta_2), \ldots, \pi(\theta_I)] \end{aligned}$$

and the matrix $\mathbf{A}$ has elements given by

$$a_{i\ell} = \frac{f(x_{jk}|\theta_i)}{f(x_{jk}|\tilde{\theta}_j)} \tag{3.12}$$

where $\ell = (j-1)K + k$. Note that $\mathbf{A}$ can be viewed as the concatenation of J $I \times K$ sub-matrices, one for each sample point $\tilde{\theta}$, as shown in Figure 3.4(a). The sub-matrices share the same set of rows, corresponding to points used to derive the weighting function π. The columns of each sub-matrix correspond to different variance weighted perturbations of the sample point associated with that matrix. Thus, each element of the matrix is the ratio of the likelihood of the prior to that of the simulated sample.

Using the results of Evans et al. [2005], we find that the decision rule which minimizes the maximum expected size over $\theta \in \Theta$ can be found by solving the convex game

$$\mathcal{S}(\pi_0, d_{\pi_0}) \approx \mathcal{M} = \min_{\mathbf{d}} \max_{\pi} \frac{1}{JK} \pi^T \mathbf{A} \mathbf{d}. \tag{3.13}$$

The first player, y, has strategy set

$$Y = \Delta(\{\theta_i \mid 1 \leq i \leq I\}),$$

that is, a vector $\pi = \mathbf{y} \in Y$ is simply a probability distribution over the finite set of samples θ_i. The second player, z, has more complex constraints. Each sub-vector of $\mathbf{d}$, $\mathbf{d}_j$, must define a decision rule which gives probability $1 - \alpha$ to accepting $\tilde{\theta}_j$ into the confidence region when $\theta^\star = \tilde{\theta}_j$. This is guaranteed by requiring that the entries of $\mathbf{d}_j$ sum to $K(1-\alpha)$, with all of the entries bounded between zero and one. We can fully represent the set of allowed $\mathbf{d}_j$ as the polyhedron $\mathbf{D}_j$ using the linear constraints

$$\begin{aligned} \mathbf{1}_K^T \mathbf{d}_j &= K(1-\alpha) \\ (\forall k) \quad 0 &\leq d_{jk} \leq 1 \end{aligned}$$

where $\mathbf{1}_K$ is the vector of length K with each entry equal to one. Accounting for the normalization factor $1/JK$, we define the convex strategy set

$$Z = \left\{ \frac{1}{JK} \langle \mathbf{d}_1, \ldots, \mathbf{d}_J \rangle \in \mathbb{R}^{JK} \mid \mathbf{d}_j \in \mathbf{D}_j \right\}. \tag{3.14}$$

Thus, we have the convex game $\mathcal{G} = (\mathbf{A}, Y, Z)$, illustrated in Figure 3.4. The "nature" player, y, chooses a vector $\mathbf{y}$ corresponding to π and the "statistician" player, z, chooses $\mathbf{z}$ corresponding to $\mathbf{d}$. A comparison of Y and Z are shown in Figure 3.4(b), where, for simplicity of illustration, we let $I = K = 3$ and $\alpha = 0.33$. While y is free to choose any point from the probability simplex composed of the three row samples, y is restricted from playing any of the pure strategies. Indeed, as α is decreased, creating a more inclusive confidence region, the restriction on Z grows, until the point where $\alpha = 0$ and z is forced to play the uniform strategy over the K columns of each sub-matrix. In this case the statistician is forced to choose all K simulations, resulting in the confidence region $\mathcal{C} = \Theta$. While z is highly restricted when choosing decision rules from Z when $\alpha = 0.05$, there are still enough alternatives to typically yield confidence regions significantly smaller than Θ.

For fixed π, the statistician knows the ideal strategy formed by finding d_π and using it to find the entries of $\mathbf{d}$ (and hence $\mathbf{z}$). Moreover, the statistician assumes that nature acts in a way that maximizes her payoff (the size of the region, $\mathcal{S}(\pi, d)$, given by Equation 3.13). This is equivalent to assuming nature chooses π_0. Hence, the statistician's minimax strategy is $\mathbf{d}_{\pi_0}$.

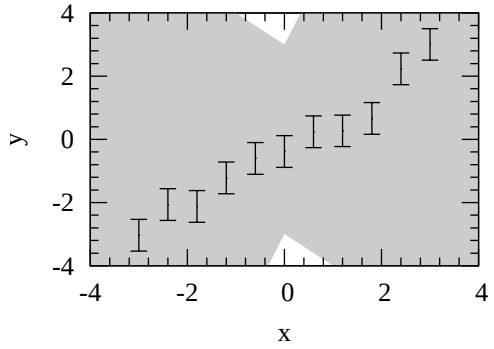
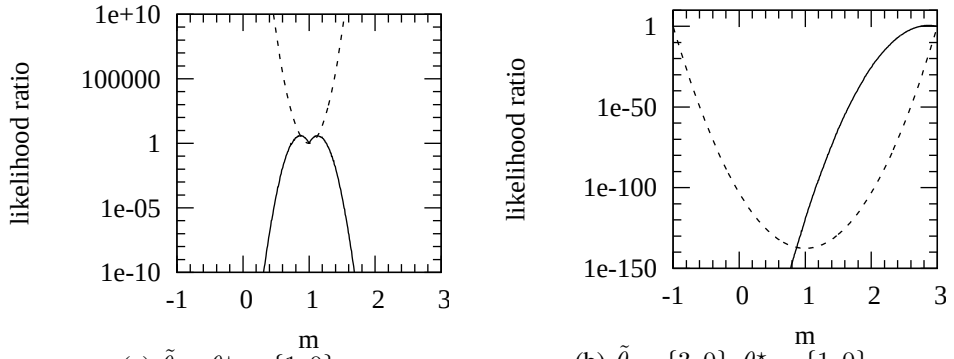


Figure 3.5: Random data generated from the linear model $s(x) = mx + b$ where each point was subject to independent random Gaussian noise with $\sigma = 0.5$. The shaded area corresponds to all possible models, $s(x)$, that can be generated by some $\theta \in \Theta$ for Θ with $m \in [-1 : 3]$ and $b \in [-3 : 3]$.

3.4.4 Simple Example

Before discussing the technical details on how to sample and solve the convex game denoted in Equation 3.13, let us construct MES confidence regions for a simple problem to gain some intuition about the procedure. A two player game is established between "nature" (y) and the "statistician" (z). Nature chooses the true value of the parameter, $\theta^\star$, but is allowed to use mixed strategies and hence nature's strategy space consists of distributions over Θ. The statistician chooses the decision procedure, but knows that the best strategy takes the form d_π for some choice of $\pi \in \Theta$. Conceptually, one can imagine that both players are choosing strategies from the same space — they each choose a distribution over Θ — with z transforming his choice, $\tilde{\theta}$ into the decision rule $d_{\tilde{\theta}}$. Unfortunately, from a computational standpoint, it is difficult to formulate a convex game in this manner. Instead, we use the game formulation of the previous section to model the interaction between y's choices of $\theta \in \Theta$ and z choice of $\mathbf{d} \in Z$.

As an illustration, consider a system governed by a simple linear model $s(x) = mx + b + \varepsilon$, where ε is some independent Gaussian noise with zero mean and known variance ($\varepsilon \sim N(0, 0.25)$), say due to measurement error. For instance, we can consider the system to be predicted observations of force exerted by a spring due to extension (Hooke's law). We will assume that $m \in [-1 : 3]$ and $b \in [-3 : 3]$, based on prior expert knowledge. Hence $\Theta = [-1, 3] \times [-3, 3]$ and $\theta = \{m, b\}$ is an element of Θ. Let the true values of the parameters underlying the model be $\theta^\star = \{1, 0\}$. In Figure 3.5, we plot 11 points drawn from the linear model with parameter vector $\theta^\star$, with some observational noise as described earlier. Clearly, Θ includes $\theta^\star = \{1, 0\}$ as well as many models which are



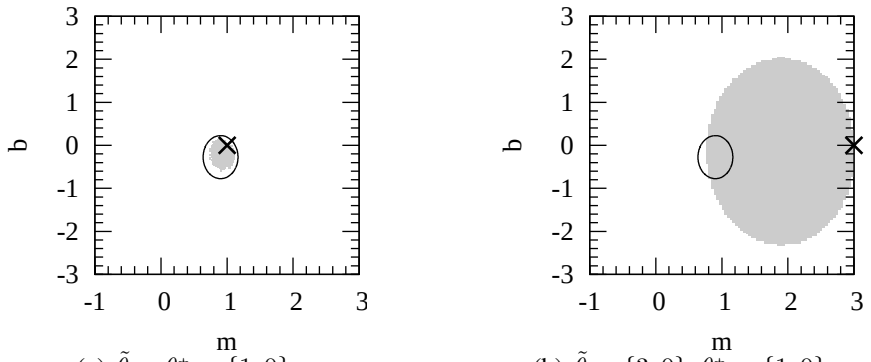
(a) $\tilde{\theta} = \theta^\star = \{1, 0\}$ (b) $\tilde{\theta} = \{3, 0\}$, $\theta^\star = \{1, 0\}$.

Figure 3.6: Comparison of the likelihood ratio (solid) versus the Neyman-Pearson cutoff value (dashed) when $\alpha = 0.05$, y chooses $\theta^\star = \{1, 0\}$ and z chooses $\tilde{\theta}$. $\tilde{\theta} = \{m, b\}$ is included in the $1 - \alpha$ confidence region if the likelihood ratio is greater than the cutoff value. Note that when $\tilde{\theta} = \theta^\star$ the resulting confidence region is much smaller than when $\tilde{\theta} = \{3, 0\}$. (See also Figure 3.7).

extremely poor fits to the observed data (for instance $\theta = \{-1, 3\}$). The set of all possible models is shown as the shaded region in Figure 3.5. Thus, in this setup, y (nature) is choosing the pure strategy that places all of the weight on $\theta = \theta^\star = \{1, 0\}$.

Now, let us observe what happens as z (the statistician) varies his choice of $\tilde{\theta}$ over Θ. Suppose that z is allowed to choose only a single $\tilde{\theta} \in \Theta$. The expected value of the payoff from z to y, the likelihood ratio given in Equation 3.12, is distributed as a Gaussian centered at $\tilde{\theta}$. However, the cutoff value used to determine $\mathbf{d}$ is an exponential function centered at $\theta^\star$, but dependent on $\tilde{\theta}$. The resulting confidence region, $\mathcal{C}$, will be the region of Θ where the Gaussian centered at $\tilde{\theta}$ is greater than the quadratic cutoff centered at $\theta^\star$.

In Figure 3.6, we show the interplay between the likelihood ratio and the cutoff values for two choices of $\tilde{\theta}$, $\tilde{\theta} = \theta^\star$ and $\tilde{\theta} = \{3, 0\}$, where b was fixed at zero. Since $\tilde{\theta}$ is included in the $1 - \alpha$ confidence region if the likelihood ratio is greater than the cutoff value, the derived confidence region will be much smaller when $\tilde{\theta} = \theta^\star$ than it when $\tilde{\theta} = \{3, 0\}$. This is confirmed by looking at the derived confidence regions, shown in Figures 3.7(a) and 3.7(b). In fact, the expected size of the confidence region is minimized when the z correctly "guesses" $\tilde{\theta} = \theta^\star$ and constructs the procedure as the best response to the true value of the parameter: $\mathbf{d}_{\theta^\star}$. In Figure 3.8(a), we plot the expected sizes of the confidence regions (over many realizations of the data) produced by z choosing a single value of θ over Θ. Darker regions correspond to larger confidence regions.

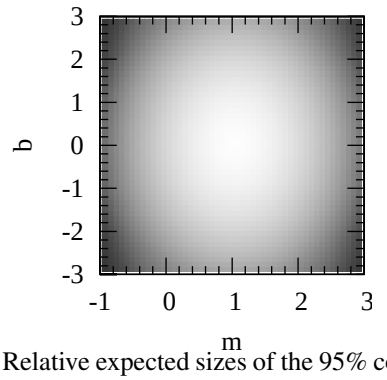


(a) $\tilde{\theta} = \theta^\star = \{1, 0\}$. (b) $\tilde{\theta} = \{3, 0\}$, $\theta^\star = \{1, 0\}$.

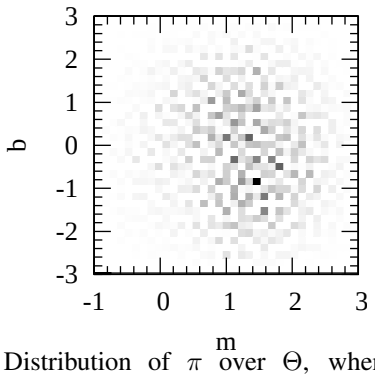
Figure 3.7: 95% confidence regions produced by the MES confidence procedure when $\alpha = 0.05$, y chooses $\theta^\star = \{1, 0\}$, and z chooses $\tilde{\theta}$ (shaded), as compared with the 95% confidence region derived using χ^2 tests (solid oval) for the linear model $s(x) = mx + b$. The $\times$ symbol denotes the location of $\tilde{\theta}$ in the space. When $\tilde{\theta} = \theta^\star$ the resulting confidence region is much smaller than when $\tilde{\theta} = \{3, 0\}$.

Thus, it is in z's best interest to always choose $\theta^\star$ and construct the corresponding decision rule. However, of course, z does not know $\theta^\star$. Moreover, z cannot use the data to try to estimate $\theta^\star$, because z must choose his strategy before observing any data (otherwise, z could select a confidence procedure which had zero size, such as the one described by Equation 3.1). Therefore, one reasonable strategy is to use the procedure d_π where π is a uniform distribution over Θ. The approach advocated here, though, involves finding the π that leads to the MES procedure. In Figure 3.8(b), we plot an approximation (using our Monte Carlo procedure) to the distribution over Θ that leads to the MES procedure. Dark regions correspond to areas of higher probability. Figure 3.8(b) shows that while π has positive probability over a wide range of Θ, the minimax solution for π_0 tends to place higher probability around the center. This is because if $\theta^\star$ is near the center of Θ, then the expected size of the confidence region will be larger than if $\theta^\star$ is near the boundary. Conceptually, a ball around $\theta^\star$ will intersect with more of Θ if $\theta^\star$ is near the center. Thus, the statistician (z) chooses the procedure to protect against this possibility.

One way to reduce risk to z of unjustifiably putting too much weight on $\tilde{\theta}$ far from $\theta^\star$ (possibly due to finite sampling effects) is to eliminate those strategies which greatly differ from $\theta^\star$ from Θ. Given that the $\tilde{\theta} \in \Theta$ which are distant from $\theta^\star$ result in large confidence regions (seen in Figure 3.7(a)), it is natural to want to restrict Θ to some smaller set. This can be done by first performing a χ^2 cut on all points in Θ, retaining only those with a



(a) Relative expected sizes of the 95% confidence regions computed with the procedure when z chooses a fixed (single point) strategy. In all cases, y chooses $\theta^\star = \{1, 0\}$. Dark areas correspond to larger regions.

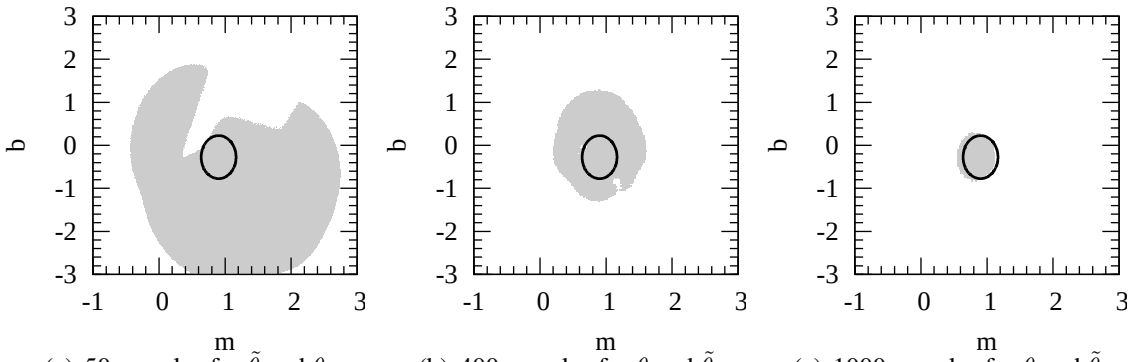


(b) Distribution of π over Θ, when y chooses $\theta^\star = \{1, 0\}$ and $\alpha = 0.05$. Dark regions correspond to higher probability. Pixelization is due to the binning of samples used to compute π.

Figure 3.8: Relative expected sizes of the resulting 95% confidence regions when z chooses a single strategy and the optimal mixed strategy z chooses when allowed to play a mixed strategy over Θ. In both cases, $\alpha = 0.05$ and y selects $\theta^\star = \{1, 0\}$. While the single optimal solution for z is to select $\theta^\star$ with probability 1, z does not know $\theta^\star$ and hence must also guard against other possible plays by y.

variance weighted sum of squares distance less than a specified cutoff. While conservative, the χ^2 test will exclude from Θ those models that are very poor fits to the data. In particular, we set the value of our χ^2 cut such that the probability of rejecting $\theta^\star$ is $\alpha/10$. We can then use a Bonferroni correction [Bonferroni, 1936] to ensure that the combination of the χ^2 cut and the MES procedure have the correct coverage. This is done by computing the $1-0.9\alpha$ level MES confidence regions on the restricted space. In practice, we have found that this two step procedure increases the chances of rejecting incorrect models, resulting in smaller confidence regions with the same $1 - \alpha$ coverage.

While we could perform the MES procedure without the χ^2 restriction, the MES procedure would be forced to compute the minimax expected size over even those parameter values which are extremely unlikely given the data, resulting in both a conservative estimate and a payoff matrix that would be orders of magnitude larger to obtain the desired accuracy. In Figure 3.9, we present 95% confidence regions for our linear model problem using differing number of samples for both players y and z (θ and $\tilde{\theta}$, respectively). As shown in Figure 3.9(a), using too few samples results in significantly irregular and larger confidence regions than when Θ is heavily sampled (Figure 3.9(c)).

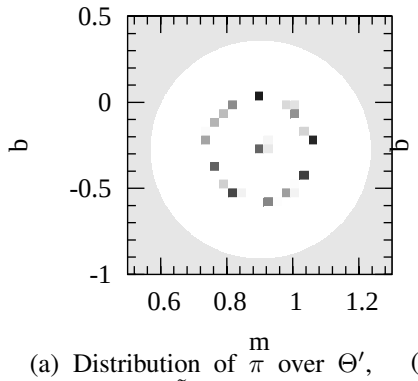


(a) 50 samples for $\tilde{\theta}$ and θ (b) 400 samples for θ and $\tilde{\theta}$ (c) 1000 samples for θ and $\tilde{\theta}$

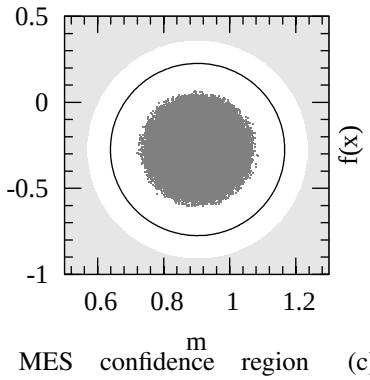
Figure 3.9: Derived MES 95% confidence regions (gray shaded regions) for the linear model $s(x) = mx + b$, using differing numbers of samples for both $\tilde{\theta}$ and θ, compared with the 95% confidence region derived by χ^2 tests (circles). Using too few samples results in extremely irregular and conservative regions.

Applying the two-stage procedure described above, where Θ is reduced to Θ' by using a χ^2 cut, the approximation to z's best procedure is shown in Figure 3.10(a). In this figure the shaded area denotes those models rejected by the χ^2 cut. The confidence region derived using this procedure is shown in Figure 3.10(b). Here, 500 candidate parameter combinations are tested and the accepted values are plotted. The slight irregularity in the boundary is due to the sampling. In practice this can be smoothed. The circle outside of this point cloud is the confidence region derived by a 95% χ^2 test applied to the original data. The gain in precision is clear. Finally, in Figure 3.10(c), we illustrate those models which cannot be rejected by both the MES confidence procedure as well as χ^2 tests. The greater statistical power of the MES test directly translates into better discriminative power in model space, allowing the MES procedure to reject more alternative models and potentially yield better scientific inferences.

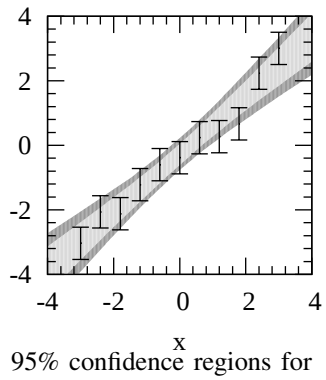
Thus, the restriction on Θ can be seen as performing two roles. First, it eliminates those $\tilde{\theta} \in \Theta$ that are clearly not good fits. Second, the χ^2 cut reduces the size of considered models in Θ, allowing us to compute accurate confidence regions with fewer samples. Using fewer samples not only results in greater data efficiency, but it reduces the size of the payoff matrix, which also reduces the time necessary to compute the minimax solution to the convex game. Indeed, the confidence region in Figure 3.10(b) was constructed with less than half of the samples and computation time than that used to construct the confidence region shown in Figure 3.9(c).



(a) Distribution of $\tilde{\pi}$ over Θ', when θ and $\tilde{\theta}$ are restricted to $\Theta' = \{\theta \in \Theta | P_{\chi^2}(\theta = \theta^\star) \geq 0.995\}$ (denoted by white disk).



(b) MES confidence region derived for the linear model $s(x) = mx + b$ (dark shading), using Θ' (white disk), compared with the confidence regions derived via χ^2 tests (circle).



(c) 95% confidence regions for both MES (light shading) and χ^2 tests (dark shading) projected into model space. Points represent random data generated from the linear model $s(x) = mx + b$ subject to independent Gaussian noise (see Figure 3.5).

Figure 3.10: Distribution of π_0 and 95% confidence regions produced when $\tilde{\theta}$ is restricted to Θ'. In all cases, $\alpha = 0.05$ and y selects $\theta^\star = \{1, 0\}$. Note that the MES confidence procedure produces confidence regions with more power (and hence smaller sizes) than a χ^2 test alternative.

3.4.5 Efficiently Computing MES Confidence Regions

Now that we have described the procedure in detail and looked at a simple example, let us determine how the MES procedure can be used in practice to compute confidence regions. Specifically, we address the questions of how to choose Monte Carlo samples, build the payoff matrix, solve the convex game, and derive $1 - \alpha$ confidence regions. In this section we will walk through the algorithm, describing each step and detailing how it can be computed efficiently.

Selection of Monte Carlo Samples

The first issue we need to address is that of selecting the Monte Carlo samples to approximate the MES confidence procedure. In Section 3.4.3 we saw that both θ and $\tilde{\theta}$ are sampled uniformly from Θ (while $x_{jk} \sim P_{\tilde{\theta}_j}$). However, as mentioned in Section 3.4.4, it is often desirable to limit Θ to just those parameter vectors which result in models accepted by a χ^2 test at some level much smaller than α. This restriction limits the MES procedure to

considering parameter vectors that are at least minimally supported by the data, reducing the number of samples required.

Let $\Theta \subseteq \mathbb{R}^W$ be the *a priori* parameter set and let Θ' be those parameter vectors from Θ that satisfy the χ^2 test. While we could choose models randomly from Θ and perform the χ^2 test to create Θ', for many problems, only a fraction of the parameter vectors in Θ result in models that lie within the χ^2 cutoff and therefore parameter vectors in Θ'. For computationally expensive models, this data inefficiency makes the MES procedure intractable.

Instead, we construct an active learning algorithm using the framework of Section 2.1 to efficiently sample vectors from Θ'. For the χ^2 and confidence ball search tasks mentioned in Sections 3.2.1 and 3.3.3, finding the boundaries of Θ' was sufficient to determining the constraints the data placed upon each parameter. However, here we are interested in actually obtaining samples from within Θ'. Thus, the threshvar sampling strategy discussed in Section 2.3 is a natural algorithmic choice.

Using the active learning framework of Section 2.1 with the threshvar heuristic, we can sample points from Θ and retain those which fall within Θ', in the set $\mathcal{T}$. A uniform sample of Θ' can then be obtained by randomly sampling points from $\mathcal{T}$. Since both θ and $\tilde{\theta}$ are sampled from Θ', we use points randomly sampled from $\mathcal{T}$ for both. In practice we generally use most, if not all, of the points in $\mathcal{T}$ for the MES procedure.

Note that a sample chosen in this manner will be slightly biased away from a truly uniform sample, as it is unlikely to include two points that are very close to each other (as such points would not be chosen by a variance-based algorithm). Our technique is similar to quasi-random sampling [Press et al., 1992, Sec 7.7]. Quasi-random sampling techniques trade the independence of consecutive samples for the ability to quickly cover the entire search space. They have been shown to be more efficient than uniform random sampling for many problems including numerical integration [Niederreiter, 1992]. For our application, we assume that parameter vectors which are close (in L_2 measure) result in models which are similar, a common property. Under this assumption, a quasi-random sample allows us to efficiently represent the space Θ' with a limited number of samples.

Unlike the samples of θ and $\tilde{\theta}$, samples for x_{jk} can easily be computed, as they depend only on the assumed noise model. For instance, in Section 3.1 we assumed the data were given by $\mu(\theta)$ plus some Gaussian noise with known covariance. Thus, sampling $P_{\tilde{\theta}_j}$ can be done efficiently as it is a multivariate Gaussian and no models (μ's) must be computed.

Constructing the Payoff Matrix

Given a set of Monte Carlo samples, we must now build the associated payoff matrix for the convex game. Note that all entries of the payoff matrix (defined by Equation 3.12) will be greater than zero, as $f(x|\theta) > 0$ for all x and θ. However, many values will be very close to zero as the parameter space Θ results in the majority of the model pairs being largely dis-similar. As we shall see in Section 3.4.5, a sparse representation of the payoff matrix will allow us to solve the convex game significantly faster. To obtain a sparse payoff matrix, $\tilde{\mathbf{A}}$, we take all entries $a_{i\ell} \leq \epsilon_{\mathrm{t}}$ and set them to zero in $\tilde{\mathbf{A}}$. Often, setting even a small value for ϵ_{t} will result in a fairly sparse matrix representation.

As an example, we consider the sparsity of the game matrix when constructing the MES procedure for the SNLS data mentioned in Section 2.2.3, and which will be discussed in detail in Section 4.1.2. For this data set, when $\epsilon_{\mathrm{t}} = 1 \times 10^{-4}$, the payoff matrix $\tilde{\mathbf{A}}$ is 96% sparse; even when $\epsilon_{\mathrm{t}} = 1 \times 10^{-32}$ (on the order of machine precision) the matrix is 85% sparse.[5] A plot of the sparsity of the SNLS payoff matrix as a function of ϵ_{t} is shown in Figure 3.11.

However, we could also construct $\tilde{\mathbf{A}}$ without first constructing $\mathbf{A}$. Assuming a Gaussian error model (as we did in Section 3.1), X has a multivariate Normal distribution with mean $\mu(\theta)$ and known variance, where $\mu(\theta)$ is the predicted model. Thus, the likelihood of an observation, x, given some $\theta_i \in \Theta$ is given by:

$$f(x|\theta_i) = \frac{1}{(2\pi)^{H/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}.$$

$f(x|\tilde{\theta}_j)$ has a similar form. If we further assume that the supernovae are independent, then for each element of x_{jk}, $x_{jkn} \sim N(\mu_n(\tilde{\theta}_j), \sigma_n)$. Thus, $x_{jkn} = \mu_n(\tilde{\theta}_j) + \sigma_n\varepsilon_{kn}$ where $\varepsilon_{kn} \sim N(0,1)$. Hence the elements of $\mathbf{A}$ are given by

$$\begin{aligned} a_{i\ell} &= \frac{f(x|\theta_i)}{f(x|\tilde{\theta}_j)} \\ &= \exp\left\{\sum_{n=1}^{N}\left[\frac{\varepsilon_{kn}^2}{2} - \frac{(\mu_n(\tilde{\theta}_j) + \varepsilon_{kn}\sigma_n - \mu_n(\theta_i))^2}{2\sigma_n^2}\right]\right\} \\ &= \exp\left\{\sum_{n=1}^{N} v\varepsilon_{kn} - \sum_{n=1}^{N}\frac{v^2}{2}\right\}, \end{aligned}$$

[5] Some convex games have much greater sparsity. For example, a payoff matrix for the poker game Rhode Island Hold'em is 99.994% sparse.

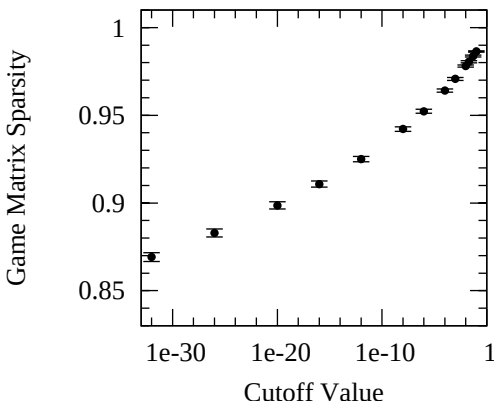


Figure 3.11: Sparsity of the game matrix for the SNLS data set as a function of the cutoff value ϵ_{t}. The cutoff value is the threshold below which non-zero entries are mapped to zero.

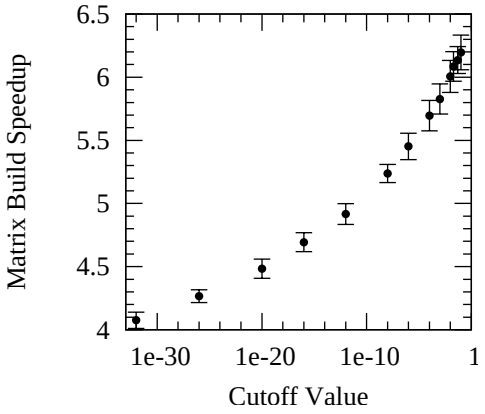


Figure 3.12: Speedup of the convex game matrix building on the SNLS data set for an algorithm that bounds matrix entries and prunes those shown to be smaller than the cutoff value over the naive algorithm as a function of the cutoff values. By pruning entries proven to be less than the cutoff value, the bound and prune algorithm can obtain a speedup of roughly 5 times over the naive algorithm.

where $\ell = (j-1)K + k$ and v is a vector with elements $(\mu_n(\theta_i) - \mu_n(\tilde{\theta}_j))/\sigma_n)$. The vector v dependents only on i and j, the index of the row and the sub-matrix, and is independent of k. Thus, we only need to compute this term once for each row of each sub-matrix. Moreover, ε_{kn} is the same for all entries of a particular column. Hence, by computing the maximum elements of ε_{kn} and v, we can bound the maximum magnitude of $a_{i\ell}$ for all i. Comparing this bound with $\log(\epsilon_{\text{t}})$, we can determine whether we can set $a_{i\ell}$ to zero without further computation. We find that by pruning those entries with maximal values less than the log of the zero cutoff value speeds up the matrix build process in proportion to the sparsity. When computing the payoff matrix for the SNLS data set, we observe a factor of 5 speedup when using the bound and prune approach over simply using Equation 3.12 to compute the matrix entries, as shown in Figure 3.12. A similar bound and prune technique can be used if Σ is not a diagonal matrix.

Solving the Convex Game

Given a payoff matrix $\mathbf{A}$ (or similarly $\tilde{\mathbf{A}}$) and the strategy sets Y and Z, defined in Section 3.4.3, we must now solve the resulting convex game: $\mathcal{G} = (\mathbf{A}, Y, Z)$. In this section we look at algorithms to solve convex games, along with a strategy pruning technique and the effect of sparse representations of $\mathbf{A}$ and approximate solutions on the resulting minimax equilibria.

Algorithms There are many approaches one could use to solve a convex game. Here we look at two standard approaches, fictitious play and linear programming, as well as the single and double oracle algorithms of McMahan and Gordon [2007].

Fictitious Play Fictitious play (FP), a classic algorithm for solving zero-sum matrix games, can also solve convex games. The FP algorithm, shown in Figure 3.13, simulates play of the game. On each iteration, both players select the action which is a best response to the average of the opponent's past actions. Standard, or synchronous, fictitious play (SFP) executes the updates independently in parallel for each player, as if the game was actually being played. Asynchronous fictitious play (AFP) does updates first for one player and then for the other, using the new average strategy computed for the first. As we shall soon see, AFP can be significantly faster.

Fictitious play relies on oracles BR_{y} and BR_{z} that calculate a best response to a fixed strategy of the opponent. As Y is a probability simplex, player y's best response to a fixed $\mathbf{z}$, $\mathsf{BR}_{\mathsf{y}}(\mathbf{z})$, is to compute the cost associated with choosing each row (by computing $\mathbf{Az}$) and then choose the row with the maximum payoff. If more than one row has the same maximal payoff, y can choose arbitrarily.

However, the best response for player z is not as straight forward. Note that the constraints on Z force player z to select a strategy that places positive probability on $K(1-\alpha)$ of the K columns per sub-matrix $\mathbf{A}_j$. In fact, given y's strategy $\mathbf{y}(= \pi)$, the choice of decision rules $\mathbf{d}_j$ are completely independent. Thus z's best response is to sequentially compute the cost vector $c_j = \pi\mathbf{A}_j$ and then select the $K(1-\alpha)$ smallest elements from c_j and set their corresponding values in $\mathbf{d}_j$ to $1/K$ (one may have to evenly distribute the remaining probability mass if there are multiple entries of c_j which are minimal). Finally we normalize the entire vector $\mathbf{d}$ by $1/J$ to ensure that it is an element of Z.

The time complexity of fictitious play is simply a product of the iteration cost and the number of iterations. As both best response oracles are approximately linear in the number of non-zeros of the resulting vectors, the per iteration computation cost of fictitious play is

$\bar{y} \leftarrow$ any strategy in Y

$\bar{z} \leftarrow$ any strategy in Z

$\text{lb} \leftarrow -\infty$ $\quad\quad$ $\text{ub} \leftarrow \infty$

$t \leftarrow 0$

while ($(\text{ub} - \text{lb}) > \epsilon$)

$t \leftarrow t + 1$

$y \leftarrow \mathsf{BR}_{\mathsf{y}}(\mathbf{A}\bar{z})$ $\quad\quad$ $z \leftarrow \mathsf{BR}_{\mathsf{z}}(\bar{y}^T\mathbf{A})$

$v_y = V(\bar{y}, z)$ $\quad\quad$ $v_z = V(y, \bar{z})$

$\text{lb} \leftarrow \max(\text{lb}, v_z)$ $\quad\quad$ $\text{ub} \leftarrow \min(\text{ub}, v_y)$

$\bar{y} \leftarrow \frac{t}{t+1}\bar{y} + \frac{1}{t+1}y$ $\quad\quad$ $\bar{z} \leftarrow \frac{t}{t+1}\bar{z} + \frac{1}{t+1}z$

end

return $(\bar{y}, \bar{z})$ corresponding to ub and lb, respectively

Figure 3.13: Fictitious Play Algorithm

dominated by computing the products $\mathbf{y}^T\mathbf{A}$ and $\mathbf{Az}$. At each step of the FP algorithm, we can compute a bound on the value of the convex game: $\mathcal{S}(\pi_0, d_{\pi_0})$. Thus, one generally chooses some error threshold, ϵ_{a}, and runs FP until this error tolerance has been met. Using the best response oracles BR_{y} and BR_{z}, the true value of the game, $(\mathbf{y}^\star)^T\mathbf{Az}^\star$, falls within the bounds

$$\mathbf{y}^T\mathbf{A}\mathsf{BR}_{\mathsf{z}}(\mathbf{y}) \leq (\mathbf{y}^\star)^T\mathbf{Az}^\star \leq \mathsf{BR}_{\mathsf{y}}(\mathbf{z})^T\mathbf{Az}$$

for $\mathbf{y} \in Y$ and $\mathbf{z} \in Z$, where $\mathbf{y}^\star$ and $\mathbf{z}^\star$ are a minimax solution to the convex game. Thus, we can run fictitious play until $\epsilon_{\text{a}} = \mathsf{BR}_{\mathsf{y}}(\mathbf{z})^T\mathbf{Az} - \mathbf{y}^T\mathbf{A}\mathsf{BR}_{\mathsf{z}}(\mathbf{y})$ is less than a specified threshold.

Interestingly, the initial choice for $\mathbf{y}$ and $\mathbf{z}$ can greatly affect the number of iterations required to reach a specified error tolerance, ϵ_{a}. In particular, it is desirable to initialize $\mathbf{y}$ and $\mathbf{z}$ to values close to the minimax solution. In practice, this is impossible, since the minimax solution is unknown.

Instead we tried several different easily computable initial distributions for the row (y) and column (z) players for both the synchronous and asynchronous fictitious play algorithms. For the row player, these distributions included: uniform distribution over Y, y's best response to a uniform distribution over Z, y selecting a random element of Y and y

selecting the single row that is the best response to z's best response to a uniform distribution over Y. The corresponding distributions were also tested for the column player. Most of these initial distributions can be either computed trivially, or with time approximately equal to a single FP interaction. In fact, much of the information that needs to be collected can be computed when the game matrix is being built with very little overhead. We find that different initial distributions for the row player do not significantly affect the overall convergence properties of the algorithms conditioned on the distribution chosen by the column player. While we find that using a uniform distribution for the row player's initial strategy is best, other strategies are very comparable.

The initial distribution selected for the column player significantly affects the observed convergence. In particular, selecting a uniform distribution for the column player results in extremely poor initial bounds for the game; these bounds are quickly and steadily improved upon. On the other hand, selecting a best response to the row player playing a uniform distribution results in significantly better initial bound on the game. However, unlike choosing the uniform distribution, the bounds on the game do not change much for the first hundred iterations. When the bounds do begin to change, they do not converge as rapidly as those computed using an initial uniform distribution for the column player. After several hundred iterations — roughly one minute of computation time — algorithms using the initial uniform distribution for the column player beat algorithms that initialize with a best response to a uniform distribution by the row player. We attribute these differences to the fact that uniform distributions allow the player to "observe" all of the opponents strategies — even though it is as an ensemble, — while best response to uniform distributions limit our knowledge of the other player's possibilities. This distinction seems to be particularly important for the column player, due to the restriction that he must place positive probability on $K(1-\alpha)$ of the elements for each sub matrix j. Thus, we initialize both the row and column players to the uniform strategy.

Linear Programming Another standard technique for solving matrix and polyhedral convex games is to write the game as a linear program (LP) and then employ standard LP software.

Recall, that the convex game in Section 3.4.3 can be written as:

$$\max_{\mathbf{y}\in Y} \min_{\mathbf{z}\in Z} \mathbf{y}^T \mathbf{A}\mathbf{z} \tag{3.15}$$

where Y is a probability simplex, and Z is the strategy set composed of J concatenated decision rules normalized by $\frac{1}{JK}$ given by Equation 3.14. These constraints can be written

in matrix form as:

$$\begin{array}{lll} Y: & \mathbf{y} \geq 0 & \qquad Z: \quad \mathbf{z} \geq 0 \\ & \mathbf{1}_I \mathbf{y} = 1 & \qquad\quad \mathbf{U}\mathbf{z} = K(1-\alpha)\mathbf{1}_J \\ & & \qquad\quad \mathbf{z} \leq \dfrac{\mathbf{1}_J}{JK} \end{array}$$

where $\mathbf{U}$ is the matrix of ones in entries $u_{i,J(i-1)+k}$ for $1 \leq k \leq K$:

$$\mathbf{U} = \begin{bmatrix} \mathbf{1}_K^T & \mathbf{0}_K^T & \dots & \mathbf{0}_K^T \\ \mathbf{0}_K^T & \mathbf{1}_K^T & \dots & \mathbf{0}_K^T \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_K^T & \mathbf{0}_K^T & \dots & \mathbf{1}_K^T \end{bmatrix}$$

and $\mathbf{0}_K$ is a vector of length K with all entries equal to zero. The U matrix ensures that the response to each sub-matrix is a decision rule. The diagonal structure of U corresponds to the fact that the optimal decision rule for each sub-matrix of $\mathbf{A}$ is independent given $\mathbf{y}$. Furthermore, the Lagrangian of Equation 3.15 problem can be written as:

$$L(\lambda, \delta, \rho, \mathbf{z}) = \mathbf{y}^T \mathbf{A}\mathbf{z} + \lambda^T \left(\mathbf{U}\mathbf{z} - K(1-\alpha)\mathbf{1}_J\right) - \delta^T \mathbf{z} + \rho^T \left(\mathbf{z} - \frac{1}{JK}\mathbf{1}_J\right)$$

where $\lambda, \delta, \rho \in \mathbb{R}$ and both δ and ρ are greater than or equal to zero.

Now suppose player y announces that she will be playing a fixed strategy $\mathbf{y} \in Y$. We can find a best response for player z by solving either $\min_{\mathbf{y} \in Y} \mathbf{y}^T \mathbf{A}\mathbf{z}$, or equivalently the $\min_{\mathbf{y} \in Y}$ of the Lagrangian:

$$\begin{aligned} \min_{\mathbf{z} \in Z} \mathbf{y}^T \mathbf{A}\mathbf{z} &= \min_{\mathbf{z} \in Z} \max_{\lambda, \delta \geq 0, \rho \geq 0} L(\lambda, \delta, \rho, \mathbf{z}) \\ &= \min_{\mathbf{z} \in Z} \max_{\lambda, \delta \geq 0, \rho \geq 0} \mathbf{y}^T \mathbf{A}\mathbf{z} + \lambda^T \left(\mathbf{U}\mathbf{z} - K(1-\alpha)\mathbf{1}_J\right) - \delta^T \mathbf{z} + \rho^T \left(\mathbf{z} - \frac{1}{JK}\mathbf{1}_J\right) \end{aligned}$$

Using the fact that strong duality holds for bounded polyhedral convex sets (cf. McMahan [2006]), the dual of the linear program above is given by

$$\begin{array}{ll} \max\limits_{\lambda, \rho \geq 0} & -K(1-\alpha)\lambda^T \mathbf{1}_J - \dfrac{1}{JK}\rho^T \mathbf{1}_J \\ \text{subject to} & \mathbf{y}^T \mathbf{A} + \lambda^T \mathbf{U} + \rho^T \geq 0 \end{array}$$

Therefore, we can write our original convex game as

$$\begin{aligned}
\max_{\mathbf{y}\in Y}\min_{\mathbf{z}\in Z}\ \mathbf{y}^T\mathbf{A}\mathbf{z} &= \max_{\mathbf{y}\in Y}\left[\begin{array}{ll} \min\limits_{\mathbf{z}} & \mathbf{y}^T\mathbf{A}\mathbf{z} \\ \text{subject to} & \mathbf{z}\geq 0 \\ & \mathbf{U}\mathbf{z} = K(1-\alpha)\mathbf{1}_J \\ & \mathbf{z}\leq \dfrac{1}{JK}\end{array}\right] \\
&= \max_{\mathbf{y}\in Y}\left[\begin{array}{ll} \max\limits_{\lambda,\rho\geq 0} & -K(1-\alpha)\lambda^T\mathbf{1}_J - \dfrac{1}{JK}\rho^T\mathbf{1}_J \\ \text{subject to} & \mathbf{y}^T\mathbf{A} + \lambda^T\mathbf{U} + \rho^T \geq 0\end{array}\right] \\
&= \left[\begin{array}{ll} \min\limits_{\mathbf{z},\lambda,\rho} & K(1-\alpha)\lambda^T\mathbf{1}_J + \dfrac{1}{JK}\rho^T\mathbf{1}_J \\ \text{subject to} & \mathbf{y}^T\mathbf{A} + \lambda^T\mathbf{U} + \rho^T \geq 0 \\ & \mathbf{1}_I^T\mathbf{y} = 1 \\ & \mathbf{y}\geq 0 \\ & \rho\geq 0\end{array}\right]
\end{aligned} \tag{3.16}$$

In this work, we use the CPLEX 10.0 commercial optimization package to solve the linear program, which proves to be very effective.

Single and Double Oracle Algorithms For very high dimensional and extremely sparse convex games like poker, the single or double oracle algorithms of McMahan and Gordon [2007] can be orders of magnitude faster than standard linear programming approaches. Here we briefly describe the double oracle algorithm; details on both algorithms can be found in McMahan [2006].

Like fictitious play, the double oracle algorithm is an iterative algorithm. The basic idea of the algorithm is to consider only the subset of the full strategy sets Y and Z that have positive support in the minimax solution. Let $Y' \subseteq Y$ and $Z' \subseteq Z$. The double oracle algorithm is initialized by picking Y' and Z' to be a fixed size random subset of Y and Z, respectively. At each iteration, the algorithm solves the convex game $\mathcal{G}' = (\mathbf{A}, Y', Z')$ and then adds a fixed number of the best response strategies to Y' and Z' based on the current game, using the oracles BR_y and BR_z. The algorithm terminates when there are no strategies in the sets $Y \setminus Y'$ and $Z \setminus Z'$ which improve the value of the game for either player.

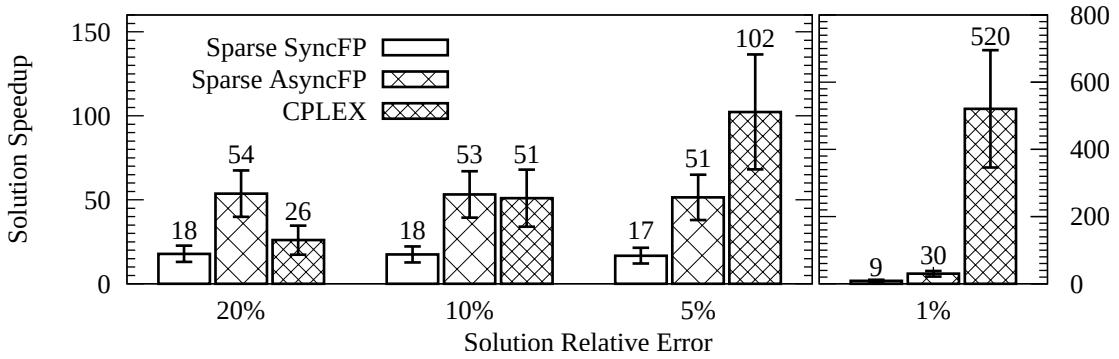


Figure 3.14: Convex game solution speedups obtained by the respective algorithms over synchronous fictitious play (SFP) using a dense matrix representation. The game matrix was composed of $I = 500$ rows and $JK = 1000 \times 200$ columns. Fictitious play (FP) is an iterative algorithm and can be terminated as soon as a certain relative error threshold has been met, while CPLEX solves the convex game exactly, up to the additive error ϵ_{t} induced by the sparse representation. Thus, the relative speedup of CPLEX over FP algorithms increases as the relative error decreases, since CPLEX's solve time is fixed while FP requires additional time to achieve lower relative errors. All of the bars for CPLEX correspond to a fixed run time of 61 seconds; SFP took 8.8 hours to solve the same game to a relative error of 1%.

Algorithm Comparison The experiments reported in Schafer and Stark [2006] used SFP on the dense game matrix $\mathbf{A}$; we use this approach as a baseline against which to compare our methods. Since FP is an iterative algorithm, we can imagine stopping the algorithm as soon as a specified error ratio has been met. For this work we consider relative error, ϵ_{a}, which we define to be the error of the MES confidence region (the difference between the upper and lower bounds on the value of the convex game) relative to the lower bound on the optimal size of the MES region. Figure 3.14 displays the average speedups, derived over 10 trials, obtained for 3 algorithms over SFP when solving for MES confidence regions for the SNLS data set, discussed in Section 4.1.2. The exact speedup depends on the random samples drawn. We used a fixed $\epsilon_{\mathrm{t}} = 1 \times 10^{-4}$ for all experiments (except for the baseline, which used the dense matrix). CPLEX took 61 seconds to solve this game. For CPLEX, $\epsilon_{\mathrm{a}} = 0$ and so the total error introduced was due to ϵ_{t}. This absolute additive error resulted in a solution with a relative error of 0.4%. For the FP implementations, we ran four experiments with relative total error stopping criteria of 20%, 10%, 5%, and 1%. Since the error due to ϵ_{t} was fixed for all of these runs, smaller total relative errors are achieved by running more iterations of FP. Thus, for all of the sets of columns in the figure, CPLEX is producing a higher-quality solution than

FP. For example, the right hand set of results shows CPLEX generating a solution with relative error 0.4% over 500 times faster than FP generates a solution with relative error 1%. Using CPLEX to solve the linear program becomes advantageous if we desire to find solutions with relative error rates less than $\sim 10\%$.

We ran preliminary experiments with both the single and double oracle algorithms. For generating approximate solutions, we were sometimes able to significantly outperform FP. However, the run times for a reasonable approximation using these algorithms were such that directly solving the linear program was preferable. For this reason, we do not report further on these results.

Thus, formulating the MES confidence procedure as a convex game and then solving it as a linear program results in two orders of magnitude speedup over SFP. In particular, the SNLS MES convex game with $I = 500$ rows and $JK = 1000 \times 200$ columns can be solved by CPLEX in roughly one minute, as opposed to almost ten hours with SFP.

Pruning Strategies from Y and Z Player z's strategy set is highly constrained and this makes it possible to prove that many of player y's pure strategies (rows of $\mathbf{A}$) cannot appear in a minimax solution. Let $\mathbf{y}(i) \in Y$ be the pure strategy that always plays row $i \in \{1, \ldots, I\}$. If there exists a row $j \neq i$ such that

$$(\forall \mathbf{z} \in Z) \qquad \mathbf{y}(j)^T \mathbf{A}\mathbf{z} \geq \mathbf{y}(i)^T \mathbf{A}\mathbf{z}, \tag{3.17}$$

then there exists a minimax solution that never plays row i. We say j dominates i if Equation 3.17 holds. If j dominates i, for any strategy $\mathbf{y}$ that sometimes plays i, the strategy $\mathbf{y}_{i \leftarrow j}$ that plays j every time $\mathbf{y}$ plays i must do at least as well against all opponent strategies.

Checking Equation 3.17 over all pairs of strategies would be prohibitive. Instead, we define

$$\text{lb} = \max_{1 \leq i \leq I} \min_{\mathbf{z} \in Z} \quad \mathbf{y}(i)^T \mathbf{A}\mathbf{z},$$

which can be computed by performing I best-response calculations. Then, for each i we check whether

$$\max_{\mathbf{z} \in Z} \quad \mathbf{y}(i)^T \mathbf{A}\mathbf{z} \quad < \quad \text{lb} \tag{3.18}$$

We can evaluate this expression by performing I "worst-response" calculations, which takes time approximately linear in the number of non-zeros in $\mathbf{A}$. If Equation 3.18 holds for some row i, then that row is dominated and can be removed from Y.

Running on different instances (caused by different random seeds) for the SNLS payoff matrix, we were able to eliminate from 20% to 60% of the rows in $\mathbf{A}$ in this manner. How-

ever, the overhead of computing this dominance approximately canceled out the speedup in the solution to the linear program. Nevertheless, this technique can be used as a preprocessing step for any convex game algorithm and we expect that on some domains improvements could be substantial. Further, more thorough direct checking of Equation 3.17 with respect to a small, diverse set of "good" strategies—perhaps the bundle maintained by the single oracle algorithm of McMahan et al. [2003]—could have substantial benefit.

Effects of Sparsity and Algorithm Induced Errors In the previous sections, we have seen that both a sparse representation of $\mathbf{A}$ and an approximate solution to $\mathcal{G}$ have the potential to significantly reduce computation times. However, it is possible that these errors may fundamentally alter the problem in such a way that the resulting solution is meaningless. Fortunately, approximately solving the approximated game $\tilde{\mathcal{G}} = (\tilde{\mathbf{A}}, Y, Z)$ gives an approximate minimax equilibrium for the original game $\mathcal{G}$. The theorem below, taken from Bryan et al. [2007a], quantifies these approximations.

Theorem 1 *Let $\mathcal{G} = (\mathbf{A}, Y, Z)$ be a confidence region game. Let $\tilde{\mathbf{A}}$ be a matrix such that $0 \leq a_{i\ell} - \tilde{a}_{i\ell} \leq \epsilon_t$ for all entries (i, l), and let $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ be an ϵ_a-approximate minimax equilibria to the convex game $(\tilde{\mathbf{A}}, Y, Z)$. Then, $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ is an $(\epsilon_t + \epsilon_a)$-approximate equilibria for the original game $(\mathbf{A}, Y, Z)$.*

Proof Let $V(\mathbf{y}) = V(\mathbf{y}, \mathsf{BR}_{\mathsf{z}}(\mathbf{y}))$ be the value of the game assuming player y chose strategy $\mathbf{y}$ and similarly for $V(\mathbf{z})$. For $\tilde{\mathcal{G}}$, define $\tilde{\mathsf{BR}}_{\mathsf{y}}$ and $\tilde{\mathsf{BR}}_{\mathsf{z}}$ by analogy to BR_{y} and BR_{z}, so for example $\tilde{\mathsf{BR}}_{\mathsf{y}}(\mathbf{z}) = \max_{\mathbf{y} \in Y} \mathbf{y}^T \tilde{\mathbf{A}} \mathbf{z}$. Then, define $\tilde{V}$ by analogy to V. It is sufficient to verify the inequalities that define an approximate minimax equilibrium. For all $\mathbf{y} \in Y$, $\|\mathbf{y}\|_1 = 1$. Each $\mathbf{z} \in Z$ is made up of J vectors $(1/JK)\mathbf{d}_j$ with $\|\mathbf{d}_j\|_1 = K(1-\alpha)$ (see Equation 3.14) and so we also have $\|\mathbf{z}\|_1 = 1 - \alpha$. Using these facts, it is straightforward to verify that

$$0 \leq V(\mathbf{y}, \mathbf{z}) - \tilde{V}(\mathbf{y}, \mathbf{z}) \leq \epsilon_{\mathrm{t}}. \tag{3.19}$$

Fix some $\mathbf{y}$ and let $\tilde{\mathbf{z}}_{\mathrm{br}} = \tilde{\mathsf{BR}}_{\mathsf{z}}(\mathbf{y})$, so

$$V(\mathbf{y}) \geq V(\mathbf{y}, \tilde{\mathbf{z}}_{\mathrm{br}}) \geq \tilde{V}(\mathbf{y}, \tilde{\mathbf{z}}_{\mathrm{br}}) = \tilde{V}(\mathbf{y}),$$

and letting $\mathbf{z}_{\mathrm{br}} = \mathsf{BR}_{\mathsf{z}}(\mathbf{y})$,

$$V(\mathbf{y}) = V(\mathbf{y}, \mathbf{z}_{\mathrm{br}}) \leq \tilde{V}(\mathbf{y}, \mathbf{z}_{\mathrm{br}}) + \epsilon_{\mathrm{t}} \leq \tilde{V}(\mathbf{y}) + \epsilon_{\mathrm{t}},$$

and so we conclude

$$0 \leq V(\mathbf{y}) - \tilde{V}(\mathbf{y}) \leq \epsilon_{\mathrm{t}}. \tag{3.20}$$

Now, let $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ be an ϵ_{a}-approximate solution to $\tilde{\mathcal{G}}$. Then, using the fact that $\tilde{V}(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}) \leq \tilde{V}(\tilde{\mathbf{z}}) + \epsilon_{\text{a}}$ from the definition of approximate minimax equilibria together with the inequalities 3.19 and 3.20, we have

$$V(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}) - \epsilon_{\text{t}} \leq \tilde{V}(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}) \leq \tilde{V}(\tilde{\mathbf{z}}) + \epsilon_{\text{a}} \leq V(\tilde{\mathbf{z}}) + \epsilon_{\text{a}},$$

from which we conclude

$$V(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}) \leq V(\tilde{\mathbf{z}}) + \epsilon_{\text{a}} + \epsilon_{\text{t}}.$$

A similar argument shows

$$V(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}) \geq V(\tilde{\mathbf{y}}) - (\epsilon_{\text{a}} + \epsilon_{\text{t}}),$$

and so we conclude $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ is a $(\epsilon_{\text{a}} + \epsilon_{\text{t}})$-approximate minimax equilibria for $\mathcal{G}$. ■

When we approximately solve $\tilde{\mathcal{G}}$ we introduce approximation error in two ways: both by finding an ϵ_{a}-approximate solution and by solving an ϵ_{t}-approximate game. Depending on the problem and the algorithm at hand, we can trade off these two sources of error for a fixed total error $\epsilon = \epsilon_{\text{t}} + \epsilon_{\text{a}}$. For example, if memory is tight, then we may prefer a larger ϵ_{t}, as this will result in a smaller memory footprint. This is principally a decision that must be made for anytime algorithms, such as fictitious play (FP) and the single and double oracle algorithms that can be stopped whenever a desired accuracy ϵ_{a} is reached. Other algorithms, such as linear programming, do not produce valid solutions until an exact solution is reached; hence $\epsilon_{\text{a}} = 0$ for these algorithms.

Locating Confidence Region Boundaries

The result of solving the convex game $\mathcal{G}$ (or similarly $\tilde{\mathcal{G}}$), is a strategy for nature π' which is nearly minimax optimal. Although π' may not equal π_0, it will still greatly reduce the maximum expected size of the confidence region relative to standard statistical approaches. This value of π' is then used in the underlying hypothesis tests (Equation 3.9). As each test has level α, the derived confidence regions will have $1-\alpha$ coverage probability, regardless of the discrete sampling approximations and approximate minimax solutions.

Given a solution to $\mathcal{G}$, we wish to approximate the confidence region $\mathcal{C}_{d_{\pi_0}}(x)$ utilizing our observed data x. To determine if the point $\tilde{\theta}$ should be included within the $1-\alpha$ confidence region, we must compute the cutoff values $c_{\tilde{\theta}}$ used in the hypothesis test given in Equation 3.9. Note that the cutoff values, $c_{\tilde{\theta}}$, have the property that $T_{\pi'}(\tilde{\theta}, x) \leq c_{\tilde{\theta}}$ with

probability $1-\alpha$. As in Section 3.4.3, we approximate $T_\pi(\tilde{\theta}, x)$ with a finite sample:

$$T_\pi(\tilde{\theta}_j, x) \equiv \frac{\int_\Theta f(x|\theta)\pi(d\theta)}{f(x|\tilde{\theta}_j)} \approx \sum_{i=1}^{I} \frac{f(x|\theta_i)}{f(x|\tilde{\theta}_j)}\pi(\theta_i) = \pi^T \mathbf{A}_j,$$

where $\mathbf{A}_j$ is the sub matrix of $\mathbf{A}$ associated with $\tilde{\theta}_j$ (see Figure 3.4(a)).

Since the x_{jk}s were chosen under the null distribution that $\theta^\star = \tilde{\theta}_j$ for all j, $(\pi')^T\mathbf{A}_j$ gives us the (approximate) distribution of $T_{\pi'}(\tilde{\theta}_j, x)$. Thus, the cutoff value $c_{\tilde{\theta}_j}$ is equal to the $(1-\alpha)K$th smallest element of the vector $(\pi')^T\mathbf{A}_j$ for all j. $c_{\tilde{\theta}_j}$ is the cutoff value of $\mathsf{BR}_\mathsf{z}(\pi)$ given in Section 3.4.5, the minimal sized decision procedure associated with the hypothesis test $\theta^\star = \tilde{\theta}_j$. Specifically, we reject the hypothesis that $\theta^\star = \tilde{\theta}_j$ given the observed data x if and only if

$$\sum_{i=1}^{I} \frac{f(x|\theta_i)}{f(x|\tilde{\theta}_j)}\pi(\theta_i) > c_{\tilde{\theta}_j}.$$

If the parameter space is small and the number of samples $\tilde{\theta}_j$ is large enough, the J samples used in the convex game may be enough to determine the $1-\alpha$ confidence region. However, it is often the case that classifying the J $\tilde{\theta}_j$ samples only gives a vague idea of the shape(s) of the $1-\alpha$ confidence region(s). In such cases, more samples are required to determine its exact boundary.

In particular, for a sample point θ, we can compute whether it is within our confidence region in a manner similar to that which we used to classify the $\tilde{\theta}_j$'s. Specifically, we compute the cost vector $c_\theta = \pi\mathbf{A}_\theta$, where $\mathbf{A}_\theta$ is the $I \times J$ sub-payoff matrix with entries given by Equation 3.12, where $\tilde{\theta}_j = \theta$, and θ_i are the same θ_i's used in the convex game $\mathcal{G}$. Let c_θ be the $K(1-\alpha)$th smallest element of c_θ. θ is an element of our $1-\alpha$ confidence region if and only if

$$\sum_{i=1}^{I} \frac{f(x|\theta_i)}{f(x|\theta)}\pi(\theta_i) \leq c_\theta.$$

While we could choose additional points randomly from Θ, a more efficient solution is to employ the active learning framework of Section 2.1 again. We define the function $g : \Theta \mapsto \mathbb{R}$ by

$$g(\theta) = c_\theta - \sum_{i=1}^{I} \frac{f(x|\theta_i)}{f(x|\theta)}\pi(\theta_i). \tag{3.21}$$

Note that those θ that result in $g(\theta) \geq 0$ will be elements of our confidence region.

This is similar to the situations that we encountered for both the χ^2 tests and confidence ball procedures (Sections 3.2.1 and 3.3.3). Again, we can use the straddle heuristic described in Section 2.2 to efficiently compute the boundaries of the confidence region, $\mathcal{C} = \{\theta | g(\theta) \geq 0)\}$.

Summary

As we have seen, the combination of active learning algorithms and linear programming can efficiently compute solutions which approximate the MES confidence procedure. The combination of the threshvar heuristic to choose samples to populate the payoff matrix and the straddle heuristic to locate the boundaries of the confidence region after the convex game is solved greatly reduces the sample complexity of the procedure. Moreover, the formulation of the MES procedure as a convex game, and the subsequent solution using linear programming reduce the computational complexity of the technique by orders of magnitude compared to either synchronous or asynchronous fictitious play.

However, even with these efficient computational and sampling techniques, the MES confidence procedure is still significantly more expensive than either χ^2 tests or the confidence ball procedure. The selection of samples to form the payoff matrix and the solution to the convex game, $\mathcal{G}$, are large, but one time costs. However, the value of $g(\theta)$, given by Equation 3.21, is itself not trivial and must be computed at all samples selected by the straddle heuristic. In particular, the computation of $g(\theta)$ involves creating a $I \times K$ submatrix, $\mathbf{A}_\theta$ and then finding $\mathbf{z}$'s best response to the vector $(\pi')^T \mathbf{A}_\theta$. Thus, while the MES procedure yields confidence regions which are generally substantially smaller than either χ^2 tests or the confidence ball procedure, (and hence results in tighter statistical inferences for the parameters), it does so at the expense of computational effort. However, in many cases, the cost associated with actually gathering the data far outweighs the complexity of statistical inference, making the MES approach an ideal candidate.

3.5 Bayesian Credible Regions

While the previous three sections have described frequentist confidence procedures, Bayesian approaches are also possible. Indeed, in many fields, Bayesian approaches far outnumber frequentist approaches, due to their (perceived) efficiency. Examples from the astronomical literature include Knox et al. [2001], Gupta and Heavens [2002], Spergel et al. [2003], Jimenez et al. [2004], Dunkley et al. [2005].

Bayesian techniques are concerned with computing the probability that the truth, $\theta^\star$,

equals some arbitrary $\theta \in \Theta$, given the observed data x: $P(\theta = \theta^\star | X = x)$. Since, we have a physical model that yields an observational hypothesis given $\theta \in \Theta$, we can compute $P(\theta = \theta^\star | X = x)$ using Bayes rule:

$$P(\theta = \theta^\star | X = x) = \frac{P(X = x | \theta = \theta^\star) P(\theta = \theta^\star)}{P(X = x)}. \tag{3.22}$$

Writing this statement in terms of density functions (using the same notation as Section 3.4) yields

$$f(\theta | x) = \frac{f(x|\theta)\pi(\theta)}{\int_\Theta f(x|\tilde{\theta})\pi(\tilde{\theta})\, d\tilde{\theta}}.$$

Again, assuming a Gaussian error model of Section 3.1, likelihood of observing x given an arbitrary $\theta \in \Theta$ is a multivariate normal given by

$$\mathcal{L}(\theta) = f(x|\theta) = \frac{1}{(2\pi)^W \left|\Sigma\right|^{1/2}} \exp\left\{ -\frac{1}{2}(x - \mu(\theta))^T \Sigma^{-1} (x - \mu(\theta)) \right\}, \tag{3.23}$$

where $\pi(\theta)$ is a prior probability distribution over $\theta \in \Theta$ arrived at before we observe any data. Up to a constant factor (given by the first term of Equation 3.23), this is the exponential of negative twice the χ^2 statistic: the variance-weighted sum of squares. Thus, we find that the Bayesian posterior is given by

$$f(\theta | x) = \frac{\mathcal{L}(\theta)\pi(\theta)}{\int_\Theta \mathcal{L}(\tilde{\theta})\pi(\tilde{\theta})\, d\tilde{\theta}} \propto \mathcal{L}(\theta)\pi(\theta). \tag{3.24}$$

The latter part of the previous equation shows a striking similarity to the expression describing the expected size of the $1 - \alpha$ MES confidence region (Equation 3.10). While the approach given here is Bayesian, and the approach given in Section 3.4 is a frequentist technique, they share many similarities. In particular, both methods assume a form of the likelihood function, $f(x|\theta)$ and a parameter space, Θ, which restricts the set of possible models. Changes to either of these assumptions will invalidate current inferences for both methods.[6] However, there are significant differences between the techniques. While the MES procedure assumes a range on the parameters to be searched, it does not assume a prior distribution over this space. Also, the MES procedure is constructed to have correct coverage probability in a frequency sense; the Bayesian approach has no coverage guarantees when it is used in repeated trials.

[6] Analyses performed by both the χ^2 and confidence ball procedures are independent of both the likelihood function and Θ.

Despite the lack of guaranteed coverage, Bayesian techniques have the benefit that they yield a probability distribution over $\theta \in \Theta$, not just point estimates or confidence regions. This posterior distribution can then be used to obtain point estimates corresponding to their frequentist counterparts. Specifically, the point $\theta \in \Theta$ which maximizes $f(\theta|x)$ is the maximum a posteriori (MAP) estimate, and corresponds to the maximum likelihood estimate (the most likely point given the data) in frequentist statistics. The Bayesian equivalent of $1-\alpha$ confidence regions are $1-\alpha$ credible regions; a subset of parameter space which contains $1-\alpha$ fraction of the probability mass of $f(\theta|x)$. While $1-\alpha$ confidence regions are theoretically guaranteed to contain the truth in $1-\alpha$ fraction of the instances in which they applied, $1-\alpha$ credible regions denote a region that a Bayesian believes has a $1-\alpha$ probability of containing the truth.

Generally, Bayesians report the $1-\alpha$ highest posterior density (HPD) credible regions, which are computed by returning the smallest set of Θ which covers $1-\alpha$ of the probability mass. While these HPD regions can be computed for $f(\theta|x)$ in its multi-dimensional setting, they are typically computed after marginalizing f. That is the HPD for a specific parameter θ_i is computed from the marginal distribution formed by integrating $f(\theta|x)$ over all parameters, except parameter i. This yields a 1 dimensional probability distribution over the parameter i, from which the HPD can be easily found.

The difficulty in Bayesian methods comes from trying to estimate $f(\theta|x)$ over the entire parameter space. If the distribution of π is chosen to be conjugate on the likelihood, then the integral in Equation 3.24 can be computed in closed form. However, since the prior distribution is often not conjugate on the likelihood, computing the posterior involves estimating an integral over the entire space spanned by the prior. Even if π is conjugate on f, we will still need to compute the posterior throughout Θ, in order to ascertain the $1-\alpha$ HPD credible regions. Unlike the frequentist techniques described in Sections 3.2 through 3.4, we cannot apply a search algorithm, such as the straddle algorithm directly to the posterior, as the threshold, t, which divides the $1-\alpha$ HPD from the remainder of Θ depends on the distribution $f(\theta|x)$ in the Bayesian case. The threshold, t, will only be known before sampling, if $f(\theta|x)$ is a known parametric distribution. However, if $f(\theta|x)$ is a known parametric distribution, the $1-\alpha$ HPD credible regions can be found directly without any sampling.

We now look at two methods to compute the posterior when it is not a known parametric form: direct integration based on samples chosen from a uniform grid, and the popular technique of Markov Chain Monte Carlo (MCMC). Both methods use a finite number of samples to approximate the posterior. However the selection of these samples is significantly different.

3.5.1 Direct Integration using a Grid

Perhaps the most straight-forward way to compute the integral in Equation 3.24 and compute the posterior, is to compute the value of $f(x|\theta)\pi(\theta)$ at each point on an evenly-spaced grid with s points per parameter. For this approach, one pre-specifies an W-dimensional grid (where W is the number of parameters of interest) and computes the posterior at the center of each grid cell. The integral is then (approximately) the sum of the unnormalized posterior at each grid cell, and the normalized posterior is given by the sampled value divided by the computed integral. While straight forward, this approach scales exponentially with dimension, and hence is infeasible for even moderate dimensions. Attempts have been made to formulate adaptive grids which shrink cell sizes in regions where the prior is high and, enlarge those cells where the prior is small [Tegmark et al., 2001]. However, even with adaptive grids, it is hard to cover the subset of the parameter space where the prior is large finely enough to provide strong inference results.

3.5.2 Markov Chain Monte Carlo

As a result of the dimensionality problem with grid-based sampling techniques, Markov Chain Monte Carlo has become an increasingly popular approach for estimating posteriors due to its (perceived) computational efficiency. MCMC is based on the idea of creating a Markov Chain, $X_1, X_2, \ldots, X_N$ with the stationary distribution equal to the posterior, $f(\theta|x)$. Let $g(\theta) = f(x|\theta)\pi(\theta)$, be the unnormalized posterior distribution. The basic intuition of the MCMC technique is if one were to use importance sampling to estimate the integral $c = \int_\Theta h(\theta)g(\theta)\,d\theta$, the points chosen by the sampling method weighted by $h(\theta)$ would produce $f(\theta|x)$.

Specifically, under certain conditions, the law of large numbers for Markov Chains yields

$$\frac{1}{N}\sum_{i=1}^{N} h(X_i) \xrightarrow{P} \mathsf{E}_g(h(X)) = c$$

Therefore, if we were able to sample points from the distribution of f, we could approximate c and hence compute f. While, this may seem like a circular argument, note that both $f(\theta|x)$ and $g(\theta) = \mathcal{L}(\theta)\pi(\theta)$ have the same distributional form. In fact they are equivalent up to the normalizing constant c. Thus we can sample points from $g(\theta)$ to estimate c and thereby obtain $f(\theta|x)$. Hence, the trick is to develop a Markov Chain which has the stationary distribution of $f(\theta|x)$ using $g(\theta)$. There are many ways that one could go about creating such a Markov Chain, including using the random-walk-Metropolis-

Hastings, independence-Metropolis-Hastings and Gibbs sampling algorithms. Here we consider the standard Metropolis-Hastings algorithm.

Let $q(\theta|\tilde{\theta})$ be an arbitrary proposal distribution, with the property that it is easy to generate samples from $q(\theta|\tilde{\theta})$. Now define $r(\theta, \tilde{\theta})$, the acceptance probability as:

$$r(\theta, \tilde{\theta}) = \min\left(\frac{g(\tilde{\theta})}{g(\theta)}\frac{q(\theta|\tilde{\theta})}{q(\tilde{\theta}|\theta)}, 1\right).$$

Now consider two arbitrary points $\theta, \tilde{\theta} \in \Theta$. Without loss of generality, let us assume that $g(\theta)q(\tilde{\theta}|\theta) > g(\tilde{\theta})q(\theta|\tilde{\theta})$. Thus,

$$r(\theta, \tilde{\theta}) = \frac{g(\tilde{\theta})}{g(\theta)}\frac{q(\theta|\tilde{\theta})}{q(\tilde{\theta}|\theta)},$$

while $r(\tilde{\theta}, \theta) = 1$. Let $p(\theta, \tilde{\theta})$ be the probability that the chain transitions directly from θ to $\tilde{\theta}$. This probability depends on two factors. First, the proposal distribution, q, must generate $\tilde{\theta}$ and secondly, r must accept $\tilde{\theta}$. Thus,

$$p(\theta, \tilde{\theta}) = q(\tilde{\theta}|\theta)r(\theta, \tilde{\theta}) = q(\tilde{\theta}|\theta)\frac{g(\tilde{\theta})}{g(\theta)}\frac{q(\theta|\tilde{\theta})}{q(\tilde{\theta}|\theta)} = \frac{g(\tilde{\theta})}{g(\theta)}q(\theta|\tilde{\theta}).$$

Therefore, we find that

$$p(\theta, \tilde{\theta})g(\theta) = q(\theta|\tilde{\theta})g(\tilde{\theta}).$$

Similarly, the probability of transitioning from $\tilde{\theta}$ to θ in one step is

$$p(\tilde{\theta}, \theta) = q(\theta|\tilde{\theta})r(\tilde{\theta}, \theta) = q(\theta|\tilde{\theta}),$$

since $r(\tilde{\theta}, \theta) = 1$. Thus, we have that

$$p(\tilde{\theta}, \theta)g(\tilde{\theta}) = q(\theta|\tilde{\theta})g(\tilde{\theta}) = p(\theta, \tilde{\theta})g(\theta). \tag{3.25}$$

Therefore, we see that the ratio of the probabilities of transitioning from θ to $\tilde{\theta}$ and from $\tilde{\theta}$ to θ is equivalent to the ratio of the densities of the posterior at $\tilde{\theta}$ and θ, respectively. Intuitively, the chain will transition to a point in parameter space in proportion to the posterior probability of that point, just as we desire.

More formally, the property of our derived chain given in Equation 3.25, is called detailed balance. Detailed balance indicates that g is proportional to a stationary distribution,

$i \leftarrow 0$
$\theta_0 \leftarrow$ arbitrary element from Θ
until converged
generate $\tilde{\theta}$ from $q(\tilde{\theta}, \theta_i)$
compute $r = r(\theta_i, \tilde{\theta}) = \min\left(\frac{g(\tilde{\theta})}{g(\theta_i)} \frac{q(\theta_i|\tilde{\theta})}{q(\tilde{\theta}|\theta_i)}, 1\right)$
set $\theta_{i+1} = \begin{cases} \tilde{\theta} & \text{with probability } r \\ \theta_i & \text{with probability } 1 - r \end{cases}$
$i \leftarrow i + 1$
end
return $\theta_0, \theta_1, \ldots, \theta_{i-1}$.

Figure 3.15: Metropolis-Hastings Algorithm

because, if detailed balance hold, then,

$$\begin{aligned} g(\theta) &= g(\theta) \int_\Theta p(\theta, \tilde{\theta})\, d\tilde{\theta} \\ &= \int_\Theta g(\theta) p(\theta, \tilde{\theta})\, d\tilde{\theta} \\ &= \int_\Theta g(\tilde{\theta}) p(\tilde{\theta}, \theta)\, d\tilde{\theta}, \end{aligned}$$

which is the definition of a stationary distribution. Therefore, choosing candidate points based upon the proposal distribution $q(\theta|\tilde{\theta})$ and then transitioning to these candidate points with probabilities given by $r(\theta, \tilde{\theta})$ yields a stationary distribution for g, that we can use to compute c, and hence derive $f(\theta|x)$. This algorithm is the oft used Metropolis-Hastings algorithm [Metropolis et al., 1953, Hastings, 1970]. An outline of the algorithm is given in Figure 3.15.

A common choice for $q(\tilde{\theta}|\theta)$ is the Normal distribution: $N(\theta, b^2)$, for some fixed $b > 0$. When q is chosen to be the normal distribution, then $q(\tilde{\theta}|\theta) = q(\theta|\tilde{\theta})$ and so $r(\theta, \tilde{\theta} = \min(g(\tilde{\theta})/g(\theta), 1)$. However, selecting b is nontrivial. Choosing b too small will result in the chain taking small steps. When this happens, it will fail to explore much of the parameter space in a fixed number of samples. As a result, it may appear that the chain has converged, when in fact it has yet to even explore some regions of parameter space.

Choosing b too large results in samples which are often in the tails of the distribution, making r small. As a result, the chain will remain in the same location for a long period, before jumping to the next location. Again, the convergence rate is poor. An optimal value of b allows for large steps, but also ensures that the chain does not get artificially stuck.

Thus, while theoretically MCMC using Metropolis-Hastings algorithm converges almost surely to the stationary distribution (the posterior) in the limit of infinite sampling, no definite statements can be made about general functions for cases where samples are limited to some finite number. In practice, the MCMC chain is run until convergence has approximately been met; that is, the distribution does not appear to be changing much with additional samples. Moreover, the first few tens or hundreds of samples are often removed from the chain to ensure that an initial choice of θ_0 does not bias the chain into regions of low probability.

However, even with these heuristics, it is quite difficult to determine if convergence has been met with a finite number of samples. In particular, if a posterior is comprised by two narrow, spatially separated Gaussians, then the probability of transition from one Gaussian to the other will be vanishingly small. Thus, after the chain has rattled around in one of the peaks for a while, it will appear that the chain has converged; however, after some finite amount of time, the chain will suddenly jump to the other peak, revealing that the initial indications of convergence were incorrect. As this example illustrates, if the Markov chain is run with too few examples, the resulting credible intervals will be too narrow, and thus will not truly contain $1 - \alpha$ of the probability mass. Thus, the consequence of lack of true convergence is artificially small credible intervals. This problem is usually skirted by assuming that there are no small isolated peaks, computing multiple independent chains and comparing the results to illustrate convergence. Additionally, Dunkley et al. [2005] and others have proposed alternative methods to detect convergence. However, none of these methods are able to prove convergence with a limited number of samples.

3.6 Comparison of Statistical Inference Techniques

Often, non-statisticians are confused by differences between Bayesian and frequentist techniques, and the advantages and limitations that each maintains. Particularly appealing with the Bayesian approach is the fact that one is computing a posterior distribution over the parameter space. Thus, not only does one obtain $1-\alpha$ credible intervals, but one gets a sense of where within the interval, the true value is expected to be. Frequentist approaches do not allow for one to compute the probability that the true value is equal to some particular parameter value. While choosing one technique over the other is a matter of personal

statistical philosophy, we believe that frequentist approaches hold important advantages over their Bayesian counterparts.

In this section we describe the differences between frequentist and Bayesian inferences, and to a lesser degree, the differences between the frequentist techniques described in Sections 3.2, 3.3, and 3.4. For illustration, we will be considering the task of computing confidence/credible regions for a simple linear model, similar to the one in Section 3.4.4. Here, we assume that $s(x) = mx$, as this will allow us to easily plot posterior distributions (since they will be one-dimensional). Here we will assume that $m \in [-3, 5]$, unless otherwise specified.

3.6.1 Philosophical Differences

One of the main differences between frequentist and Bayesian inference methods is how the two ideologies view the concept of probability. Frequentist statistics posits that probabilities are observed properties of the real world, that can be computed by looking at the relative frequencies of events occurring. In this framework, parameters are fixed, unknown constraints, and as such no useful probability statements can be made about them (their relative frequencies are either 0 or 1 depending on the question at hand). Frequentist statistical procedures are designed to have provable long-run behaviors. For instance, frequentist $1 - \alpha$ confidence procedures construct regions which, when applied to a large series of data sets, trap the truth in at least $1 - \alpha$ fraction of the cases.

Bayesian statistics, on the other hand, view probability as degrees of belief, independent of the underlying events' relative frequencies. Thus, even though parameters are fixed constants, probability statements can be made about them. As a result, we can compute a probability distribution over the parameter space, and then compute either point estimates or credible regions directly from this distribution, as discussed in Section 3.5.

Due to this differing opinion about the meaning of probability, it is not surprising that frequentist and Bayesian techniques answer different questions. In particular, frequentist inference methods are interested in computing confidence procedures which create regions that trap the true parameter $1 - \alpha$ fraction of the time, while Bayesian techniques are interested in determining those parameter vectors which most likely.

For parametric models with large sample sizes, Bayesian and frequentist approaches are known to result in similar inferences. However, for high dimensional and non-parametric problems, Bayesian credible intervals may trap the true value of the parameters close to zero percent of the time [Wasserman, 2004], resulting in inaccurate statistical inferences. That is, if Bayesian techniques are applied to a series of data sets, the fraction of the re-

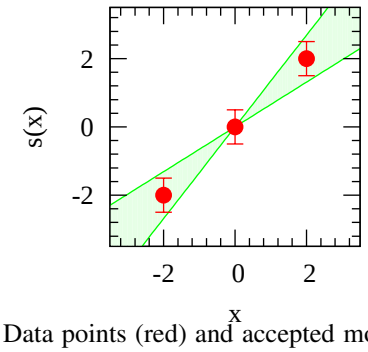
sulting $1 - \alpha$ credible intervals that contain the true values of the parameter will likely be less than $1 - \alpha$ and may be significantly less than $1 - \alpha$. This potential lack of coverage for Bayesian models makes their application to scientific inference disconcerting. Thus, as Wasserman [2004] notes, "to construct procedures with guaranteed long run performance, such as confidence intervals, use frequentist methods."

3.6.2 Effects of Priors

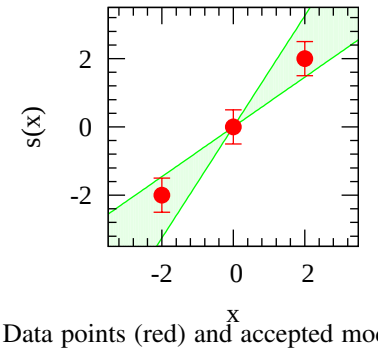
Another major difference between frequentist and Bayesian techniques is the use of priors. Given a model simulator, it is generally straightforward to compute the likelihood of some observed data as a function of input parameters. Computation of this likelihood value is sufficient for frequentist analyses. However, for Bayesian analyses, we are interested in the probability of the parameters given the data. This posterior probability can be obtained from the likelihood using Bayes rule (Equation 3.22), after first assuming a prior distribution over the parameters values. While in some cases a prior distribution can be justified, the formulation of a prior is often troublesome. Moreover, examples can be constructed in which the selection of a prior results in the posterior becoming independent of the observed data; that is the posterior, $f(\theta|x)$, depends solely on the prior [Wasserman, 2004].

Even in cases where the selection of the prior cannot result in a posterior which is independent of the data, selection of the prior can be difficult. In particular, in many cases the knowledge does not exist to form a reasonable prior. In these cases, an "uninformative prior," equivalent to a uniform distribution on some bounded range, is often assumed. However, such a prior is not uninformative; a uniform prior indicates that the practitioner believes that the true distribution of the parameter has equal probability throughout points in Θ, not that it is unknown. For instance, consider our simple linear model, $s(x) = mx$ for the data shown in Figure 3.16(a). A uniform prior for m on the range $[-3 : 5]$ would mean that the practitioner believed that observations from the model $s(x) = 5x$ were just as likely as $s(x) = x$. Given the observed data, and their small measurement error, this seems to be an unreasonable estimate. Additionally, if the practitioner truly believed that $m = 1$ and $m = 5$ were equally-likely, then s/he would certainly want to increase the range of the parameter space to ensure that the results were not biased by the prior to be artificially small.

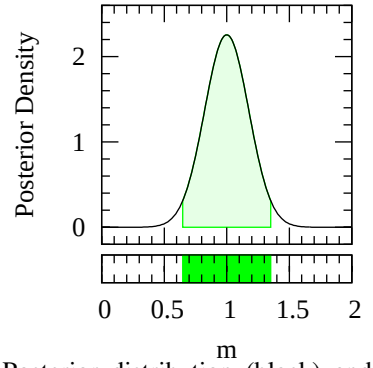
"Uninformative" priors are also parametrization dependent. Suppose, we reformulate our linear model as $s(x) = x/m'$, where $m' = 1/m$. Then, a uniform prior over m is not equivalent to a uniform prior over m', as m and m' are inversely related to each other. Choosing uniform prior distributions for m and m' results in markedly different posterior



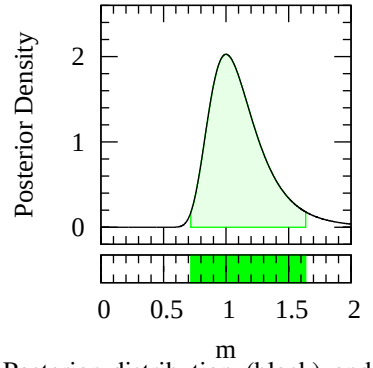
(a) Data points (red) and accepted models (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



(b) Data points (red) and accepted models (green) for Bayesian 95% credible intervals for m when π is a uniform prior on $m' = 1/m$ over the range $(-3, 5)$.



(c) Posterior distribution (black) and accepted values of m (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



(d) Posterior distribution (black) and accepted values of m (green) for Bayesian 95% credible intervals for m when π is a uniform prior on $m' = 1/m$ over the range $(-3, 5)$.

Figure 3.16: Resulting posteriors after choosing "uninformative" or uniform priors over m and $1/m$. Choosing a uniform distribution over a parameter has a large impact on the resulting posterior distribution, and hence the derived credible regions. Thus uniform priors cannot be viewed as "uninformative."

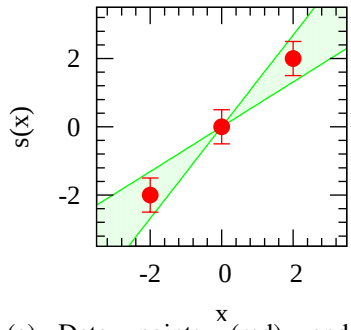
distributions, and hence different 95% credible regions, as shown in Figure 3.16(d).

While parametrization problems may seem trivial for the linear model example, it has significant effects in many scientific applications. For instance, the WMAP data set we shall describe in Section 4.1.1, has two parameters (H_0 and Ω_M) which are inversely related. The simulator which models the WMAP data takes Ω_M as an input parameter, but not H_0. Now suppose an astronomer is interested in computing the $1-\alpha$ credible interval for H_0. It is not clear whether the astronomer should put a uniform distribution over H_0 or Ω_M. What is clear is that the choice over which parameter to place a uniform distribution will affect the derived credible regions. Moreover, if the astronomer later decides s/he wants to compute credible regions for Ω_M, should s/he redo the analysis with the uniform prior on the other variable? Often, astronomers are interested in both H_0 and Ω_M; which parameter should have the uniform distribution in this case? The root problem in all of these cases is that there is no way to specify a distribution that is truly "uninformative". Any distribution that is chosen will place some constraints on the parameter and will affect the derived posterior distribution.

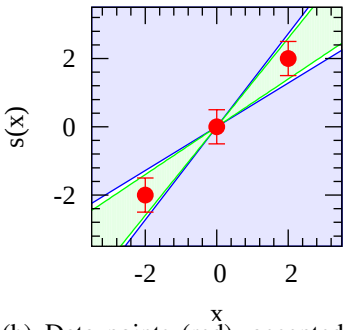
In cases where the prior is justified, say by another independent data set, it is often difficult to write the prior in a compact manner which allows it to be used when computing the posterior. Unless the prior can be formulated in a parametric form, calculating values of the prior over the parameter space will require computing the posterior from the independent data set. Often practitioners approximate the prior as a set of independent Normal distributions based upon the one dimensional marginalization of the independent data set. However, the one-dimensional marginals are hardly ever truly Gaussian. Additionally, this strategy completely ignores the correlations in the prior between the various parameters.

Another issue with priors, is that the choice of a prior can significantly affect the resulting posterior. In Figure 3.17, we illustrate the 95% credible regions derived for the simple linear model $s(x) = mx$ for three different prior distributions. In this case, the priors are formed by selecting a uniform distribution over the ranges $[-3:5]$, $[0.64, 1.36]$, and $[0.82, 1.18]$, respectively. The figure shows that restricting the prior restricts the derived credible regions. In particular, Bayesian techniques will never include points where the prior has zero probability in the $1-\alpha$ credible regions, as these points have zero posterior probability. Thus, a prior with regions which have zero probability will remove those regions from the posterior, even if those regions are supported by the data.

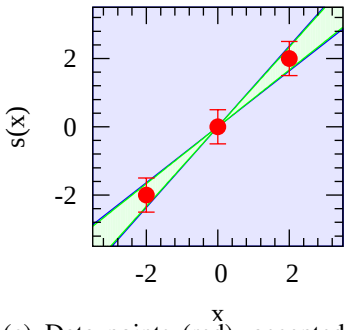
As a result, if we choose a prior which is too restrictive, we will underestimate the size of the $1-\alpha$ credible regions. However, it is difficult to determine if the prior is too restrictive directly from the returned credible regions. Because the Bayesian methods return the HPD which covers $1-\alpha$ fraction of the probability mass, they will return only a portion of the parameter space, even if all of the models corresponding to points that



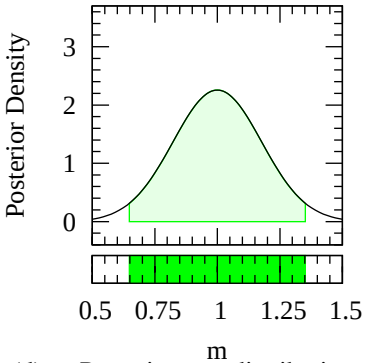
(a) Data points (red) and accepted models (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



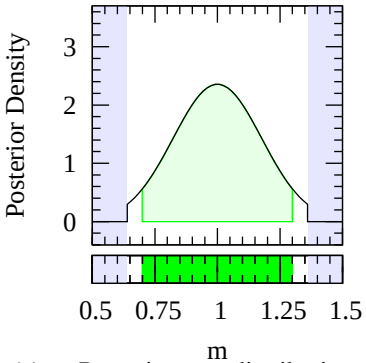
(b) Data points (red), accepted models (green) and models with zero posterior probability (blue) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(0.64, 1.36)$.



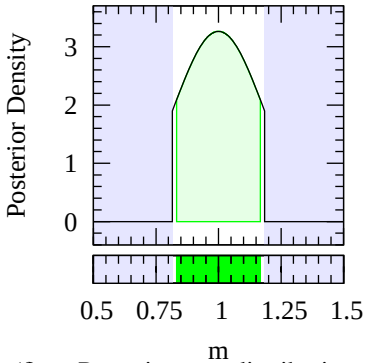
(c) Data points (red), accepted models (green) and models with zero posterior probability (blue) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(0.82, 1.18)$.



(d) Posterior distribution (black) and accepted values of m (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



(e) Posterior distribution (black), accepted values of m (green), and values of m with zero prior probability (blue) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(0.64, 1.36)$.



(f) Posterior distribution (black), accepted values of m (green), and values of m with zero prior probability (blue) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(0.82, 1.18)$.

Figure 3.17: 95% credible intervals for the grid-based Bayesian method with a uniform prior over various ranges. Blue regions correspond to regions where the prior, and hence the posterior, is zero. The choice of the prior greatly affects the posterior distribution.

parameter space are supported by the data (as is the case in Figures 3.17(c) and 3.17(f)). Thus, the resulting regions are likely to not include the entire parameter space, as shown in Figures 3.17(e) and 3.17(f). Thus, it may be tempting to believe that the prior correctly covered the nonzero probability regions of the likelihood function, when in fact it did not, resulting in artificially small credible regions (Figures 3.17(e) and 3.17(f)).

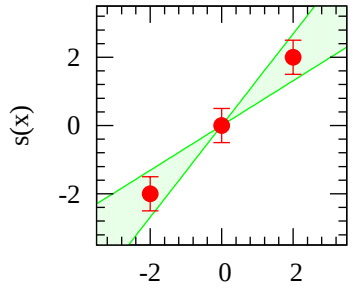
Finally, any change to the prior invalidates all of the current inferences. In particular, when one is using a uniform prior, merely changing parameter ranges will result in a different posterior with possibly different $1 - \alpha$ credible intervals. Thus analyses in which we change the range on one or more parameters to see the effects on the derived credible regions would have required us to recompute the entire chain (or set of chains) — an extremely expensive proposition — or somehow approximate the difference. These analyses are informative as they provide insights into how the various parameters are interconnected. However, for Bayesian techniques, the prior should be independent of the data, and hence it should not be changed after observing the data. By recomputing the posterior using a new prior (based upon a previous posterior), we open ourselves to errors incurred due to multiple hypothesis testing. Moreover, it is a small step from such repeated Bayesian inferences to data-dependent priors, which are incoherent, not Bayesian. Hence, data-dependent priors do not benefit from theoretical guarantees derived for Bayesian analyses, which assume priors are chosen before any data is observed.

Some Bayesians tend to believe, incorrectly, that assumptions on the parameter space used in frequentist techniques are essentially priors. However, note that most, if not all, physical models do not depend on the range of the input parameters; they are merely a function of the specific choice of a parameter vector $\theta \in \Theta$. Hence, assuming a parameter space does not affect either the likelihood calculations or the parameterization of the model, while a Bayesian prior does. Secondly, many frequentist inference techniques are completely independent of the parameter space, while the remaining methods can be made independent by first imposing a simple cut (such as the χ^2 cut used as a first pass in the MES method in Section 3.4.4).

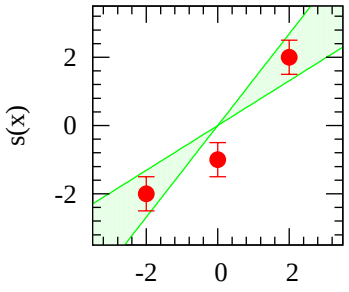
3.6.3 Model and Parameter Space Assumptions

As mentioned in Section 3.5, both the MES and Bayesian techniques are dependent on assumptions about the form of the likelihood function and the parameter ranges to be searched, while the χ^2 and confidence ball methods do not.

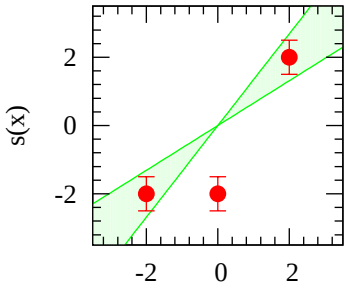
Because they assume the form of the likelihood function as part of the procedure, the MES and Bayesian techniques need only consider those parameter vectors which are



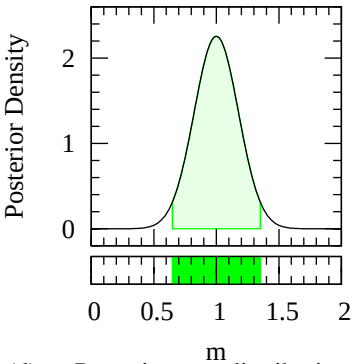
(a) Data points (red) and accepted models (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



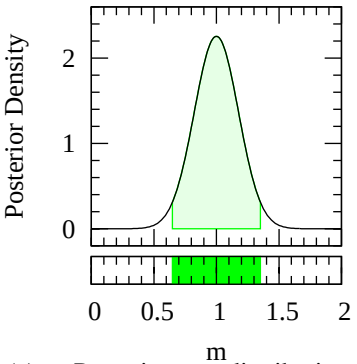
(b) Data points (red) and accepted models (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



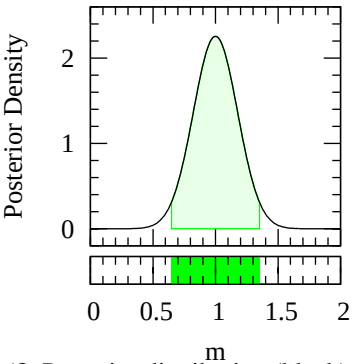
(c) Data points (red) and accepted models (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$.



(d) Posterior distribution (black) and accepted values of m (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$ using the data shown in (a).



(e) Posterior distribution (black) and accepted values of m (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$ using the data shown in (b).



(f) Posterior distribution (black) and accepted values of m (green) for Bayesian 95% credible intervals for m when π is a uniform prior on m over the range $(-3, 5)$ using the data shown in (c).

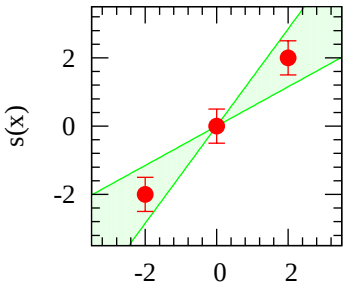
Figure 3.18: 95% credible intervals computed by a grid-based Bayesian technique with a uniform prior on m over the range $[-3 : 5]$. Note that as the data departs from a linear model intersecting the origin, the derived posterior, and hence the 95% credible region, does not change much. Because Bayesian techniques assume a form of the likelihood function and return the highest posterior region which covers 95% of the probability mass, they are susceptible to returning credible regions which correspond to models that are not supported by the data, as seen in panels (c) & (f).

possible given the model, and the assumed parameter range. As a result, they are able to efficiently reduce the size of the resulting confidence/credible regions. However, because the MES and Bayesian techniques assume that the model is correct, they have a tendency to return regions, even if the corresponding models are not well supported by the data.

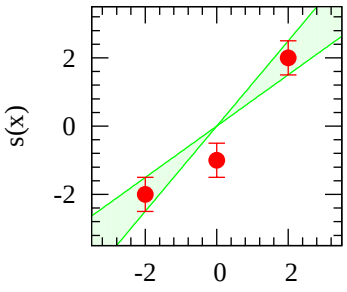
For example, in Figure 3.18, we plot three different data sets and for each compute the 95% credible regions (using a grid-based approach). As the data departs from the linear model through the origin (from Figures 3.18(a) to 3.18(c)), the likelihood of the data given the parameters decreases. However, the resulting credible region remains essentially the same, due to the fact that the entire likelihood function is decreasing. The Bayesian technique selects the highest posterior density region with $1 - \alpha$ fraction of the mass, regardless of how likely this region really is. Similarly, the MES procedure depends on the likelihood ratios of various points, not the absolute value of the likelihood. Thus, it also will always return a non-empty confidence region, even if the data wildly contradict the assumed model.

As with priors, inferences derived with techniques that make assumptions about the form of the likelihood function or the ranges of the parameter space will need to be recomputed if these assumptions change. This can be a computationally costly proposition.

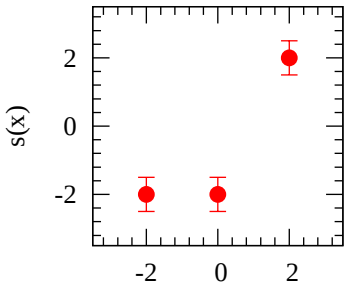
On the other hand, both the χ^2 and confidence ball procedures create W-dimensional ellipses centered on either the data (for the χ^2 tests) or a non-parametric fit of the data (for the confidence ball method) which contain all possible models which would be included within the $1 - \alpha$ confidence region. Both of these ellipses are completely independent of the likelihood function as well the the parameter space. Hence, if the combination of the physical model and the parameter space Θ result in models which do not sufficiently fit the data, it is possible that the $1 - \alpha$ confidence regions will be empty. Moreover, augmenting the parameter space by increasing the range of one or more parameters will not affect the inferences currently made. Consider again, the three data sets of Figure 3.18. In Figure 3.19, we plot the 95% confidence regions produced by the χ^2 test procedure. As the data conforms less and less to the linear model $s(x) = mx$ (intersecting the origin), the derived confidence region decreases, until it is empty in Figure 3.19(c). Importantly, note that the lack of fit illistrated in Figures 3.19(b) and 3.19(c) is not due to observational noise. In particular, the noise, depicted by the error bars in Figures 3.19(a) through 3.19(c), is the same for all three sets of data. The empty confidence intervals for m based on chi^2 tests is due to the fact that the assumption that the data was generated from a linear model (passing through the origin) is incorrect. Thus, while the χ^2 and confidence ball procedures produce confidence regions which are typically larger than either the MES or Bayesian approaches, they are not susceptible to returning spurious results when the physical model does not fit the data.



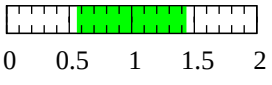
(a) Data points (red) and accepted models (green) for 95% confidence regions computed with χ^2 tests.



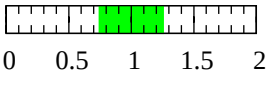
(b) Data points (red) and accepted models (green) for 95% confidence regions computed with χ^2 tests.



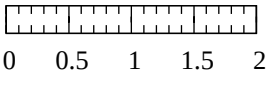
(c) Data points (red) and accepted models (green) for 95% confidence regions computed with χ^2 tests.



(d) Range of the 95% confidence interval for m computed by χ^2 tests on the data shown in (a).



(e) Range of the 95% confidence interval for m computed by χ^2 tests on the data shown in (b).



(f) Range of the 95% confidence interval for m computed by χ^2 tests on the data shown in (c).

Figure 3.19: 95% confidence intervals computed using χ^2 tests. Since χ^2 tests depend only on the observed data and measurement errors, as the data departs from a linear model intersecting the origin, the derived confidence intervals decrease. When none of the models fit the data well, the χ^2 procedure returns empty intervals, (as seen in panels (c) and (f)), unlike either the Bayesian or MES procedures (e.g. Figure 3.18(f)).

This argument further supports the idea of first performing a χ^2 cut on the parameter space before performing the MES procedure, as suggested in Section 3.4.4. By performing the χ^2 cut, we can ensure that the model has at least minimal support over the parameter range. We can then use assumptions about the form of the likelihood in the MES procedure to ensure that the size of the final confidence regions is (nearly) optimal.

3.6.4 Efficiency & Convergence

The last significant difference between the various statistical inference techniques is their sample efficiency and convergence properties. All of the methods we have described will converge in the limit of infinite sampling. χ^2 test, confidence ball, and MES procedures require infinite sampling in order to exactly determine the boundary of the confidence region. Additionally, the MES procedure will return the optimal minimal expected size regions when the payoff matrix in the convex game includes all of the possible truths, that is all $\theta \in \Theta$.[7] For the Bayesian techniques, infinite sampling allows us to either create a grid with infinitely small cells, or create infinite Monte Carlo chains which yield the stationary distribution of the posterior.

In practice, however, we never have infinite data. Moreover, data collection tends to be expensive, and hence it is sparse. None of the methods above have theoretical guarantees as to the optimality of the solutions obtained after a fixed number of samples. Nonetheless, we can make general observations about the efficiency and approximate convergence properties of the various methods.

For instance, one fundamental difference between Bayesian and frequentist methods is the need for Bayesian methods to compute the posterior over the parameter space in order to compute the normalization factor, c, and hence determine the boundary of the $1-\alpha$ highest posterior density (HPD) regions. This calculation is not needed for frequentist technique. Instead, for frequentist techniques the level-set which defines the $1-\alpha$ confidence region can be directly calculated by the procedure before any sampling takes place. Moreover, this boundary is not a function of either the number or particular samples chosen, as it is for the Bayesian techniques.

In particular, consider the example of the two separated Gaussians mentioned in Section 3.5.2, except now, let one of the Gaussians have twice the probability mass of the other (so that the two peaks are not exactly equivalent). If a Markov Chain were to rattle around in one of the peaks for some time, say the larger of the two peaks, then it would

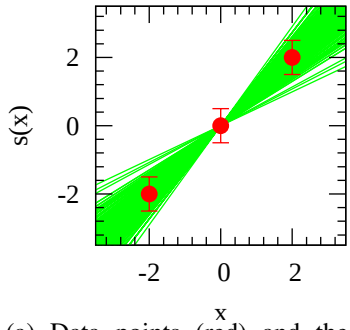
[7]Note that our approximate model still returns $1-\alpha$ coverage, but may return confidence regions which are slightly larger than the true minimax optimal regions.

appear that $1-\alpha$ HPD region should be at a particular value t. Once the chain was able to jump to the other peak, it would learn that the true threshold of the $1-\alpha$ HPD was $t' > t$. If the chain were able to jump between the peaks easily, then the chain would determine the true value of $1-\alpha$ HPD level-set quickly. However, in this case, we assume that the probability of transition from one peak to the other was vanishingly small, and hence the expected number of samples required to determine the true value of the $1-\alpha$ HPD level-set would be large. Thus, in Bayesian analyses, the selection of samples not only affects the derived credible regions, but also heavily influences the convergence rate.

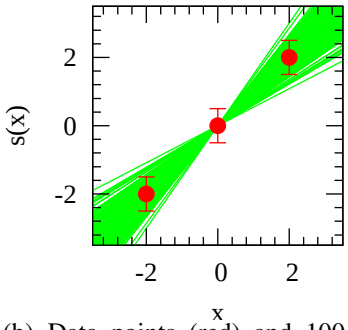
As a result, frequentist methods can perform a targeted search in parameter space for the desired function level-set, while Bayesian methods are required to sample and/or integrate the posterior distribution over the entire parameter space. As mapping a region of high likelihood points in parameter space can be thought of as a search problem, Bayesian methods do not necessarily translate into good search algorithms in practice. In particular, MCMC methods "represent" a high-likelihood region by heavily sampling that region. However, when we are computing $1-\alpha$ credible regions, it is the low-likelihood regions (those around the $1-\alpha$ HPD boundary) that we are interested in. Thus the regions which define the $1-\alpha$ credible region will receive roughly $1/\alpha$ fraction of the samples. For 95% credible regions, this corresponds to one twentieth of the samples apportioned to the highest likelihood regions of the parameter space. In contrast, a search algorithm that can directly observe the (normalized) likelihood of a sample will have no reason to spend samples in well sampled regions which are not on the $1-\alpha$ confidence region boundary.

This difference in sampling patterns is shown in Figures 3.20 and 3.21, where we compare the samples chosen by the MCMC algorithm (described in Section 3.5.2) with those chosen by the straddle heuristic when used in conjunction with χ^2 tests (see Section 3.2.1) on the task of computing 95% credible/confidence intervals for m. In both cases, the task was essentially that of computing 95% credible/confidence intervals for a normal distribution, due to the assumed Gaussian error model. For the Bayesian case, we assume that the observed data is a single point at the origin. As a result, the true posterior derived via sampling will be exactly the same as the true Normal distribution. This is done to ensure that both algorithms are sampling the same function, allowing us to accurately compare the sampling patterns of the algorithms.

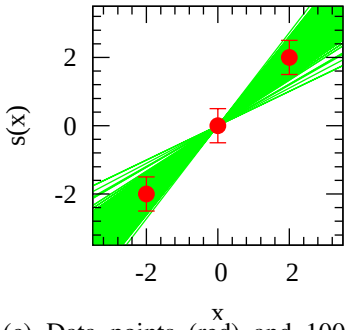
Both algorithms were constrained to samples of m chosen in $[-3:5]$. The MCMC algorithm was started at a randomly selected point, with a uniform prior over the range. In this figure we use a normal proposal distribution (with $\sigma^2 = 0.5$), although the sampling pattern is similar for other values of σ^2 that we tried. Examining the figures, several interesting differences become apparent. First MCMC has failed to converge in 100 samples, while our algorithm has converged nicely. The credible intervals given by MCMC are not



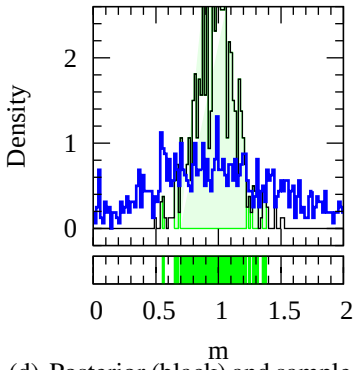
(a) Data points (red) and the 100 models sampled (green) by MCMC when computing 95% credible regions for m.



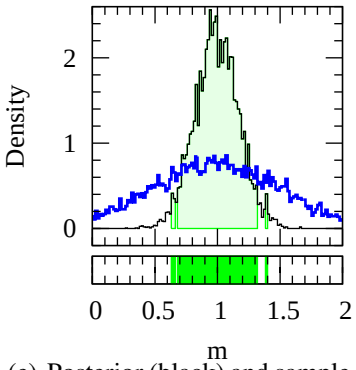
(b) Data points (red) and 100 randomly selected models (green) from 10,000 samples chosen by MCMC when computing 95% credible regions for m.



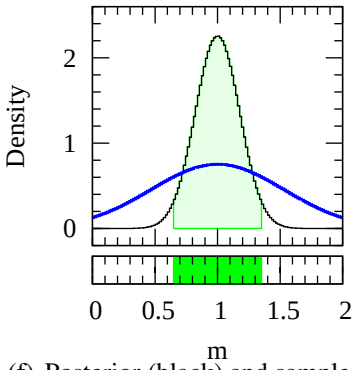
(c) Data points (red) and 100 randomly selected models (green) from 100,000 samples chosen by MCMC when computing 95% credible regions for m.



(d) Posterior (black) and sample (blue) densities after sampling 100 points using MCMC. Green regions denote 95% credible intervals for m.

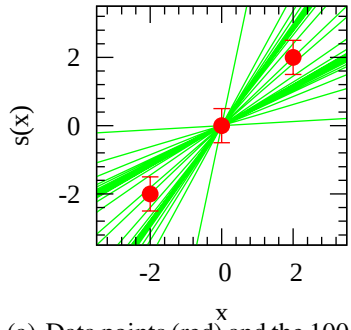


(e) Posterior (black) and sample (blue) densities after sampling 10,000 points using MCMC. Green regions denote 95% credible intervals for m.

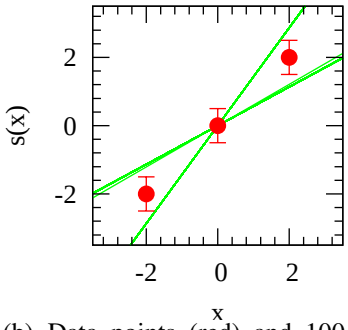


(f) Posterior (black) and sample (blue) densities after sampling 100,000 points using MCMC. The Green region denotes the 95% credible interval for m.

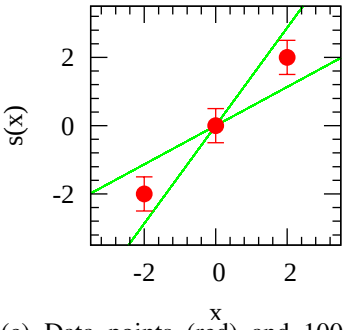
Figure 3.20: Sampling distribution of MCMC when trying to compute 95% credible intervals for m with various numbers of experiments. All chains were given an initial 100 extra samples to ensure that the random initial location did not adversely affect the overall results. MCMC samples points in proportion to the posterior distribution. Hence it favors high-likelihood regions of the parameter space to those regions which are on the $1-\alpha$ credible region boundary. As a result, the convergence rate is slower than χ^2 tests using the straddle heuristic (see Figure 3.21).



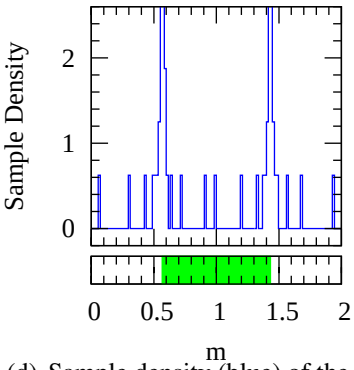
(a) Data points (red) and the 100 models sampled (green) by the straddle heuristic.



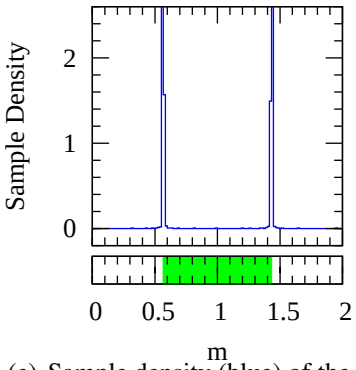
(b) Data points (red) and 100 randomly selected models (green) from the 10,000 samples chosen by the straddle heuristic.



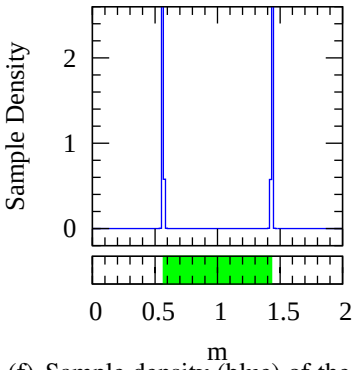
(c) Data points (red) and 100 randomly selected models (green) from the 100,000 samples chosen by the straddle heuristic.



(d) Sample density (blue) of the straddle heuristic after selecting 100 points to compute the 95% confidence intervals of m using χ^2 tests. Green regions denote the derived 95% confidence regions.



(e) Sample density (blue) of the straddle heuristic after selecting 10,000 points to compute the 95% confidence interval of m using χ^2 tests. Green regions denote the derived 95% confidence region.



(f) Sample density (blue) of the straddle heuristic after selecting 100,000 points to compute the 95% confidence interval of m using χ^2 tests. Green regions denote the derived 95% confidence region.

Figure 3.21: Sampling distribution of the straddle heuristic when trying to compute 95% confidence intervals for m using χ^2 tests with various numbers of experiments. Since the 95% confidence region level-set can be determined before any samples are chosen, the straddle heuristic can be used efficiently learn these confidence regions. The straddle heuristic quickly learns the approximate location of these boundaries panel (d), and then uses its remaining samples to determine the exact location of the boundary. As a result, the observed sampling density is composed of two delta function centered on the 95% confidence interval boundaries (panels (e) and (f)). The convergence rate is much faster than MCMC (see Figure 3.20).

only underestimated, but are also not centered on the true distribution's center, revealing a potential liability for interpreting MCMC chains which have not converged.

Second, notice that MCMC heavily samples the peak of the distribution, while our algorithm focus on those regions associated with the confidence interval boundaries. Even after 10,000 samples, the MCMC chain results in a ragged collection of disjoint credible intervals, while our algorithm returns a single interval in which the endpoints have been well determined.

Third, note that within the first 100 samples, our algorithm samples extreme points to ensure that it has not failed to observe additional peaks in the distribution which may contribute to the 95% confidence interval, while MCMC has not. As noted before, since MCMC is not a search algorithm, it may spend a large number of samples in a single distribution peak before jumping to another peak in the distribution. This sampling pattern may cause MCMC to appear to have converged, when in reality it has just failed to transition to the second peak, as in the case of the two distant Gaussians described previously.

Finally, we note that the MCMC algorithm is not data efficient. While Figures 3.20(c)-3.20(f) depict those experiments run by MCMC, the final MCMC chain consists of only those points that were accepted (in this case by the Metropolis-Hastings algorithm). As such, some of the points that MCMC samples are discarded immediately; these samples are never used to guide the chain in future steps, or to determine the $1-\alpha$ credible intervals. In addition, many MCMC practitioners remove all but every jth sample point (for some integer $j > 0$) to ensure that the points in the chain are truly independent. This significantly reduces data efficiency.

As these figures indicate, the combination of the straddle heuristic with χ^2 tests can be much more data efficient than the MCMC algorithm. Since the confidence ball and MES procedure use a search algorithm essentially identical to the one used for χ^2 tests (see Sections 3.3.3 and 3.4.5), they will also enjoy this increased data efficiency over their Bayesian counterparts.

3.7 Summary

In this chapter we have discussed the ideas of $1 - \alpha$ confidence and credible intervals. These intervals allow scientists to make meaningful inferences about the ranges of parameters which are fed into physical models of some observed phenomena. We have described several techniques which can be used to compute either $1 - \alpha$ confidence or credible regions.

For frequentist methods, we have seen that, in general, there is a trade off between computational effort to compute the threshold t of the $1-\alpha$ confidence region and resulting region size. The threshold for χ^2 tests can be quickly read from the $1-\alpha$ quantile of a $\chi^2_{(N)}$ distribution, while the threshold used by the MES procedure can only be computed after first solving a convex game. However, MES confidence regions can be shown to be approximately optimal, and are generally much tighter than regions derived using χ^2 tests.

Bayesian techniques can also be used to compute $1-\alpha$ credible regions. One particularly popular Bayesian method is Markov Chain Monte Carlo (MCMC), due to its perceived efficiency. However, we have seen that Bayesian techniques require sampling points throughout the parameter space in order to estimate the normalization constant of the posterior. As such, we cannot directly apply the search techniques developed in Chapter 2. While we could apply these algorithms after the normalization constant had been determined, by this time the entire posterior is already known and hence the $1-\alpha$ HPD regions can be found without further sampling. Moreover, we believe that Bayesian techniques suffer a number of shortcomings. These include the fact that the prior must be known (as there are no "uninformative priors"). Additionally, any changes to the parameter range to be searched, the prior, or the model being used invalidates all the inferences made to that point and the Markov chains must be restarted. Finally, we find the fact that Bayesian $1-\alpha$ credible regions may contain the true value of the parameter in substantially less than $1-\alpha$ fraction of the cases where the technique is used disconcerting.

Thus, we believe the combination of frequentist statistics with the active learning approaches discussed in Chapter 2 provides a much statically justified and efficient set of tools for computing scientific inferences. In the next chapter we apply these tools to a number of astronomical data sets.

Chapter 4

Astronomical Applications

In the previous two chapters we have discussed how active learning algorithms can be designed in order to efficiently compute statistical inferences. In this chapter, we demonstrate our algorithms and illustrate the advantages that our joint confidence regions have over standard techniques. While, we will restrict our analysis to astronomical sources, we note that the ideas and techniques developed here are useful in a wide range of scientific applications, including biology, geology, and physics.

In this chapter, we will be computing statistical inferences for a number of cosmological parameters in an attempt to determine physical properties of our Universe, such as age, composition and eventual fate. We begin by discussing the parameters for which we are interested in computing confidence regions, as well as three different types of data that can be used to constrain these cosmological parameters. We then demonstrate the active learning and statistical methods we have developed to compute jointly valid $1 - \alpha$ confidence regions for these parameters. We primarily consider frequentist inference techniques, but show some results for Bayesian alternatives. However, as discussed in Section 3.6, frequentist procedures are more easily integrated into our active learning framework.

4.1 Cosmological Data

Cosmology is the field of studying the Universe as a whole. In particular, cosmologists are interested in learning how it began, how it evolves, and what its eventually fate will be. The standard cosmological model begins with the "big bang": All of the matter and energy in the Universe is thought to have been compressed into a single small region, from which it then rapidly expanded.

What the Universe was like in the the initial 10^{-43} seconds after the big bang is unknown, as our understanding of general relatively is incomplete at the high densities present during this period. Between 10^{-43} seconds and 10^{-2} seconds, the Universe cooled to 10^{10} Kelvin. During this period, collisions resulted in subatomic particles to transitioning to and from photons. After 10^{-2} seconds, the Universe was filled with protons and neutrons, left from the high energy phase before, along with cooler photons. These photons were not energetic enough to form protons or neutrons through collisions, but did have enough energy to form electrons and positrons.

After the first few minutes, the Universe had cooled to 9×10^8K, at which point protons and neutrons were able to combine to form deuterium[1] nuclei. Deuterium nuclei then collided with protons or other deuterium nuclei to form helium, and lithium nuclei. However, during this period the Universe was still opaque to photons, as the energy in the photons was high enough to strip electrons from atoms. As a result, photons were constantly being scattered by free electrons.

It was not until roughly 300,000 years after the big bang before the Universe had cooled to 3000K that it became transparent. During this period, known as recombination, electrons combined with atomic nuclei to form stable atoms. As the energy contained in the photons was not enough to ionize electrons, the number of free electrons decreased rapidly. As a result, photons were able to stream through the Universe without interacting with the matter around them. Due to this separation, both matter and radiation subsequently evolved independently. Photons have flowed unencumbered through the Universe, slowly cooling due to the expansion of the Universe to about 3K today. Small over densities in the matter gravitationally collapsed, forming stars and galaxies 10^9 years after the big bang.

While the basic outline of how the Universe changed after the big bang is known, the exact details of when and how the specific events occurred is less well understood. For instance, it is unknown exactly how galaxies and clusters of galaxies formed in the Universe. If the density fluctuations in the matter after recombination were large, then galaxies could have formed in a top-down manner, in which large clouds of gas contracted, fragmented, and formed galaxies. However, if the fluctuations were smaller, then each of these over-densities could have collapsed into a star clusters, which then merged to form the galaxy clusters observed today. Moreover, n-body and observations experiments suggest that more than 95% of the Universe is in the form of dark matter (matter that does not strongly interact with photons), with a substantial proportion in the form of non-baryonic matter [Zwicky, 1937, Tegmark et al., 2001, Spergel et al., 2007].[2] Additional

[1] Deuterium is an atom with one proton and one neutron.

[2] Baryonic matter consists of matter formed by protons and neutrons — what we typically think of as

Parameter	Description	Ranges	Relations
τ	reionization optical depth	0.0 – 1.2	
Ω_Λ	dark energy / critical density fraction	0.0 – 1.0	$\Omega_\Lambda = \Omega_{\rm DE}$
$\Omega_{\rm M}$	matter / critical density fraction	0.1 – 1.0	
$\omega_{\rm DM}$	dark matter density	0.01 – 1.2	
$\omega_{\rm B}$	baryon density	0.001 – 0.25	
f_ν	dark matter neutrino fraction	0.0 – 1.0	
n_s	scalar spectral index	0.5 – 1.7	
A_s	scalar fluctuation amplitude	*0.6845*	
α	running of spectral index	*0.0*	$\alpha = dn_s/d\ln(k)$
b	galaxy bias	0.0 – 3.0	
Q_{nl}	non-linear correction	*30.81*	
$\Omega_{\rm k}$	spatial curvature	-1.0 – 0.9	$\Omega_{\rm k} = 1 - \Omega_\Lambda - \Omega_{\rm M}$
$\Omega_{\rm T}$	total density to critical density fraction	0.1 – 2.0	$\Omega_{\rm T} = \Omega_{\rm M} + \Omega_\Lambda = 1 - \Omega_{\rm k}$
$\omega_{\rm c}$	cold dark matter density	0.0 – 1.2	$\omega_{\rm c} = \omega_{\rm DM}(1 - f_\nu)$
$\omega_{\rm N}$	neutrino matter density	0.0 – 1.2	$\omega_{\rm N} = \omega_{\rm DM} f_\nu$
H_0	Hubble's Constant	10.5 – 380	$H_0 = 100\sqrt{\omega_{\rm DM} + \omega_{\rm B}}/\sqrt{\Omega_{\rm M}}$

Table 4.1: Meaning, ranges, and relationships among the cosmological parameters considered in this thesis. Parameters above the horizontal line are parameters for which confidence regions were explicitly computed as part of the analysis, while confidence regions for those parameters below the horizontal line can be ascertained based upon their dependencies to those parameters above the line. Ranges denote those values of each parameter that were considered as inputs to the physical models; the parameter space Θ is the cross product of the ranges of the those parameters above the horizontal line. All of the cosmological models rely upon, and hence yield inferences upon, on only a subset of these variables.

experiments, such as the supernovae data discussed in Section 4.1.2, predict that there is a negative gravitational force — a dark energy — which is splitting the Universe apart [Overbye, 2003, Scranton et al., 2003].[3]

Models which explain the evolution of the Universe are functions of many parameters. Here we look at a subset of these parameters, shown in Table 4.1. The effects of many of these parameters can be inferred from the previous discussion. For instance, the reion-

matter.

[3]Dark energy is a repulsive force that acts on a large scale, pushing two objects apart. While the objects may get farther away, the Universe will not actually be "split" into multiple fragments. Moreover, dark energy is quite weak at local scales, and cannot over power two objects which are bound by a strong force. For instance, dark energy will not cause gravitationally bound objects, such as the Earth or the Sun to explode.

ization depth, τ determines how far a photon will travel before it is scattered just before reionization.

The terms $\Omega_{\rm M}$ and Ω_Λ determine the relative fraction of the matter and dark energy in the Universe, as compared to the critical density, $\rho_0 = 3H_0^2/(8\pi G)$, where G is the gravitational constant. $\Omega_{\rm T}$ determines the geometry of the universe. If $\Omega_{\rm T} = \Omega_{\rm M}+\Omega_\Lambda = 1$, then the density of the Universe is equal to the critical density, and the universe is flat. If $\Omega_{\rm T} > 1$, then the Universe is closed, with a geometry similar to the surface of a sphere. If $\Omega_{\rm T} < 1$, the the universe is open, with a geometry similar to a saddle. The term $\Omega_{\rm T}$, in combination with the form of the dark energy component, also determines whether the universe will expand forever or collapse upon itself in a big crunch.

The density term $\omega_{\rm B}$ determines the contribution of baryons to the total matter density (where $\omega_{\rm M} = h^2\Omega_{\rm M}$). As mentioned before, baryonic matter includes all of the matter that we frequently observe, including stars, planets, and ourselves. Larger baryonic densities lead to higher pressures during recombination, and hence a more compressible matter/energy fluid. The terms $\omega_{\rm DM}$ and f_ν describe the density of dark matter, and determine if that dark matter is formed from massive neutrinos, or from other exotic particles, called cold dark matter. Cold dark matter, is needed to create large enough over densities in the baryons at the time of recombination to allow for the eventual collapse of these baryons into stars and galaxies. Without any cold dark matter, the over-densities in the baryonic matter during recombination are damped by diffusion effects.

Finally, the n_s and A_s describe the nature of primordial density perturbations, while α approximates the change of n_s with respect to the scale size on which it is measured. Q_{nl} and b describe the resulting large scale structure power spectrum, discussed in Section 4.1.3.

The parameter ranges considered in Table 4.1 are similar to those given in Tegmark et al. [2001] and Tegmark et al. [2006]. However, we have slightly increased the parameter space to include a secondary peak in the parameter space. Given this parameter space, Θ, let us now describe three data sets with which we can constrain the estimated ranges of each parameter.

4.1.1 Cosmic Microwave Background

The Cosmic Microwave Background (CMB) angular temperature power spectrum is the most widely utilized data set for constraining the cosmological parameters [Tegmark et al., 2001, Christensen et al., 2001, Verde et al., 2003, Spergel et al., 2003, Tegmark et al., 2004, Spergel et al., 2007]. This power spectrum, which statistically measures the distribution of

temperature fluctuations as a function of scale, is comprised of at least two peaks thought to have been formed by sound wave modes inherent in the primordial gas during recombination. The locations, heights, and height-ratios of the peaks and valleys in the power spectrum can provide direct information about fundamental parameters of the Universe, such as the space-time geometry, the fraction of energy density contained in the baryonic matter, and the cosmological constant [Miller et al., 2001]. However, it is more common for cosmologists to compare the observed CMB power spectrum to a suite of cosmological models (e.g. CMBFast [Seljak and Zaldarriaga, 1996] and CAMB [Lewis et al., 2000]). These models require as input some minimal number of cosmological parameters, d, — typically $d = 6$ or $d = 7$.

Here, we examine the CMB power-spectrum ($\hat{C}_\ell$) as measured by the Wilkinson Microwave Anisotropy Probe's first year data release [Bennett et al., 2003, Hinshaw et al., 2003, Verde et al., 2003][4], shown in Figure 4.1, as the third year data was uunavailable when we performed our analysis. While the third year data has significantly less noise than the first year data, the only parameters for which inferences which become substantially tighter are n_s and H_0 [Spergel et al., 2007]. However, in Section 4.2.3 we will see that the combination of other data sets with the WMAP first year data produces the same effect. Our approach is similar to that of other authors (e.g. Tegmark [1999], Tegmark et al. [2001], Spergel et al. [2003]), who fit the observed CMB power spectrum to a suite of cosmological models. These models, while sophisticated and detailed, have numerous free parameters, some of which are difficult to ascertain (e.g. ionization depth, contribution of gravity waves). However, there are many codes available to compute the CMB power spectrum, which trade off speed for accuracy and robustness.

Both CMBFast [Seljak and Zaldarriaga, 1996] and the related CAMB [Lewis et al., 2000] compute the CMB power spectrum by evolving the Boltzmann equation using a line of sight integration technique. While an order of magnitude faster than computing the full Boltzmann solution, this approach is still rather slow. Each individual model temperate spectrum takes between 30 seconds and 10 minutes to compute using CMBFast.

One approach for reducing the computation time of CMBFast is to split the Boltzmann computation into low and high multipole moment portions, as the low and high multipoles are mostly independent [Tegmark et al., 2001]. Using this method, ksplit, Tegmark et al. [2001] was able to reduce computation time by a factor of 10. Additionally, several approximate programs have been developed which are orders of magnitudes faster than CMBFast, including DASh [Kaplinghat et al., 2002], CMBWarp [Jimenez et al., 2004], and Pico [Fendt and Wandelt, 2006]. In general, these programs gain great speedups by approximating the power spectrum with a regression function fit to predetermined sample

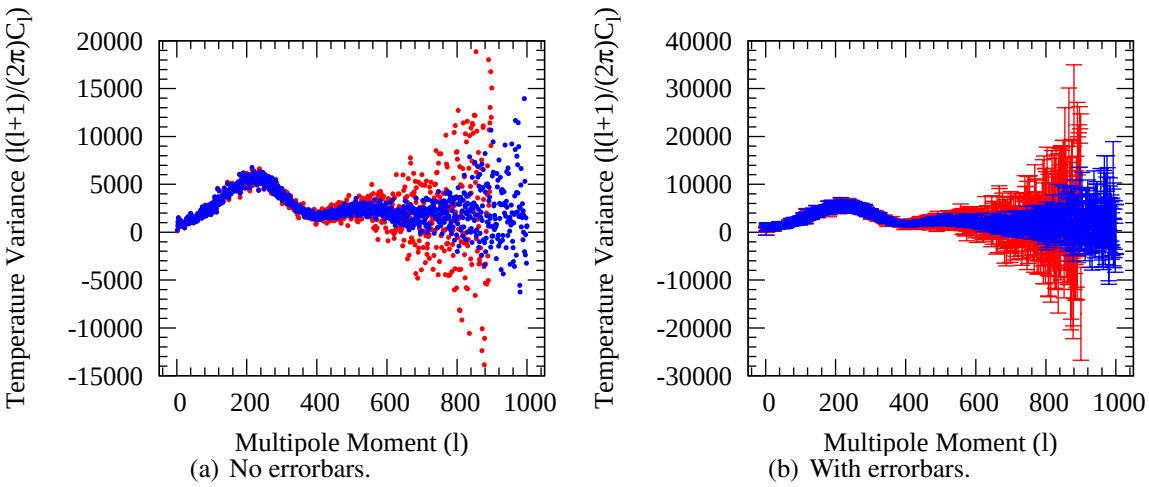
[4]Available at `http://lambda.gsfc.nasa.gov`

points generated from simulators such as CMBFast. As a result, generating a hypothesis spectrum for a new set of parameters is a simple function evaluation, foregoing the computation of the Boltzmann equation entirely.

While using any one of these approximate methods or ksplit may seem appealing due to their computational efficiency, they do not have the desired accuracy and robustness [Seljak et al., 2003]. These codes are only approximations. While fairly accurate around the concordance peak, their accuracy drops off drastically when computing models for parameter vectors slightly removed from the "accepted" cosmological models. Additionally, these codes are prone to failures when presented with parameter vectors that are not within a narrowly defined region around the concordance model [Fendt and Wandelt, 2006]. According to the Pico website: "Since Pico's purpose is to be part of parameter estimation codes, we are mainly concerned with having the regression coefficients defined around the region of parameter space allowed by the data (mainly the WMAP3 data). Pico will not be able to compute accurate spectra and likelihoods away from this region, but it will warn you about this." Similarly, in many instances ksplit will hang on parameter vectors that are a short distance from the concordance peak. Since we are interested in finding the tightest possible confidence intervals for all regions of parameter space that can possibly fit the data, we do not want to be artificially restricted by our CMB simulator. Thus, we choose to compute the model CMB power spectra using CMBFast; while not the fastest code available CMBFast is accurate and reliable.

Finally, the multipole covariance matrix can be estimated by using the covariance derived for the concordance model using code from Verde et al. [2003]. We find that the computed variances of the first year WMAP data match well with those found in the first year data release, with only a slight (roughly 1.15) multiplicative offset. This constant factor offset was hinted at by the sub unity slope of the quantile-quantile plot of the variance weighted deviations between the data and the concordance model prediction, using the variances given in the WMAP data.

Spergel et al. [2007] show that the WMAP third year data are well described by a simple 6 parameter model: $\tau, H_0, \Omega_{\rm M}, \Omega_{\rm B}, \sigma_8, n_s$. For the WMAP first year data set, we consider effectively the same model space as the simplified model in Spergel et al. [2007], except that we include the neutrino fraction and exclude σ_8. We made this change as we are not utilizing large-scale structure data, which is sensitive to σ_8. The resulting parameter vector $\mathbf{p} = (\tau, \Omega_\Lambda, \Omega_{\rm M}, \omega_{\rm DM}, \omega_{\rm B}, f_\nu, n_s)$ is similar to the model space searched by Tegmark et al. [2001].



(a) No errorbars. (b) With errorbars.

Figure 4.1: First-year (red), and third-year (blue) cosmic microwave background (CMB) temperature-temperature power spectrum data from the Wilkinson Microwave Anisotropy Probe (WMAP) data set shown with (a) and without (b) errorbars.

4.1.2 Supernovae

The second source of data that we will consider is type Ia supernovae. Type Ia supernovae result from the tightly regulated explosion of white dwarf stars. White dwarfs are stars that have reached the end of their lives, having burned all the nuclear fuel in their cores and blown off their atmospheres. Without the nuclear processing in their cores to counteract the pressure of gravity, these stars contract until they become electron degenerate, and then slowly cool off. However, in binary systems, white dwarfs can gravitationally "steal" matter from companion stars if the atmosphere of these neighboring stars gets too close during expansion phases of the neighboring star's life. If the accumulated mass on a white dwarf surpasses the 1.4 solar mass Chandrashakar limit, the electron degeneracy of the star is no longer enough to balance the gravitational force and the star collapses until the core becomes neutron degenerate. The reverberations of the white dwarf's atmosphere reflecting off the core tear the star apart.

Since the inputs and processes leading to type Ia supernovae are so highly constrained, their explosions follow predictable patterns, which allow astronomers to accurate estimate their maximum apparent magnitude (or brightness), m. Comparing this observed magnitude with the predicted magnitude for such explosions, M, yields the distance modulus, μ: $\mu = m - M$.

Because magnitudes are the negative logarithm of brightness, smaller values of m and

M correspond to brighter objects. Since all type Ia supernovae are (much) more distance than 10 parsecs[5], $\mu > 0$, allowing for a natural distance interpretation. Using the fact that energy flux falls off as the inverse of the squared distance, the supernova's luminosity distance, d_L, is given by

$$\mu = 5 \log_{10}(d_L) + 25. \tag{4.1}$$

where d_L is in the units of megaparsecs. Thus, the observation of the type Ia supernova explosion yields an estimate on the distance to the supernova, assuming that the Universe was static. However, we know that this assumption is false.

If instead, we assume a homogeneous, isotropic and flat Universe[6], the Robertson-Walker metric [Robertson, 1936], predicts that the distance to the observed supernova with redshift z is given by

$$d_L = \frac{c(1+z)}{H_0} \int_0^z \frac{dt}{\sqrt{\Omega_{\mathrm{M}}(1+t)^3 + \Omega_\Lambda}}. \tag{4.2}$$

The redshift z for a supernova is given by the change in the wavelength of emission and absorption lines, $\Delta\lambda$ relative to their values in the rest frame, λ: $z = \Delta\lambda/\lambda$. Moreover, an object's redshift is dependent on the expansion of the Universe between the time that the light was emitted and when it was observed. Thus, since the distance computed based on the observed magnitude of the supernova does not depend on expansion of the Universe, while the distance computed from the redshift does, we can isolate the effects due to expansion as a function of redshift, or equivalently as a function of time.

Specifically, we combine Equations 4.1 and 4.2 to produce a model of μ as a function of H_0, Ω_{M}, and Ω_Λ:

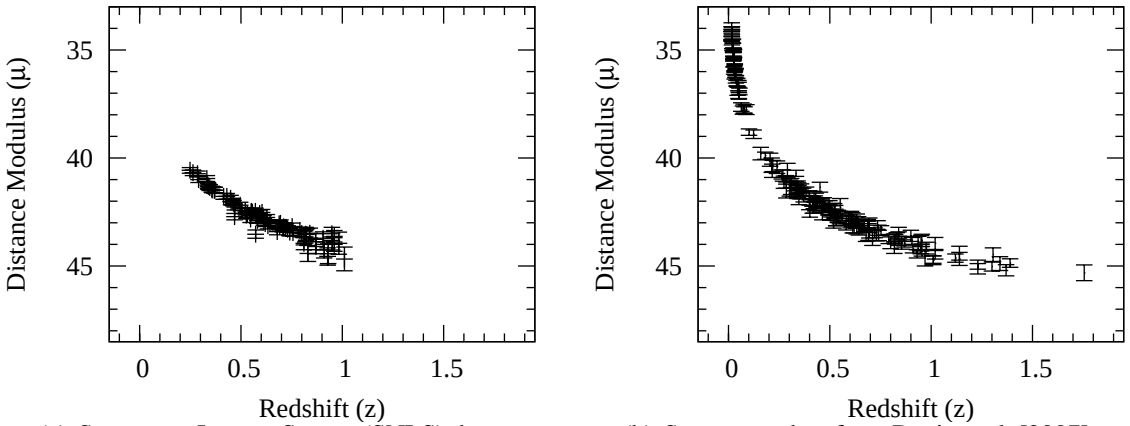
$$\mu_i = 5 \log_{10} \left(\frac{c(1+z_i)}{H_0} \int_0^{z_i} \frac{du}{\sqrt{\Omega_{\mathrm{M}}(1+u)^3 + \Omega_\Lambda}} \right) + 25, \tag{4.3}$$

The error in the estimates of the distance moduli is assumed to have the Gaussian distribution with mean zero and specified variance σ_i^2. Figure 4.2 displays the data; the error bars illustrate the magnitude of σ_i in each case.

Comparing the distance moduli models predicted by Equation 4.3 with the observations in Figure 4.2, we can make inferences about the true values of the unknown parameters $H_0, \Omega_{\mathrm{M}}, \Omega_\Lambda$. Note that while we cannot compute the integral given in Equation 4.3,

[5] A supernova within 10 parsecs would be a *very* bad thing for life here on Earth.

[6] The assumptions of homogeneity and isotropy are common in astrophysics. The flat assumption, seems to be supported by the CMB data.



(a) Supernova Legacy Survey (SNLS) data.
(b) Supernova data from Davis et al. [2007].

Figure 4.2: Observed distance moduli as a function of redshift for the SNLS data (a), and the Davis et al. [2007] data (b). Since distance modulus is the negative log of brightness, the distance modulus increases with distance and hence redshift.

we can quickly approximate it using Romberg's method, a generalization of Simpson's method [Press et al., 1992, Section 4.3]. Thus, unlike the CMBFast experiments of Section 4.1.1, the supernova experiments can be computed in a fraction of a second. As a result, the supernovae data sets provide an excellent real-world testbed for our algorithms.

While there are many supernovae data sets to choose from, we consider two: the Supernova Legacy Survey (SNLS) data set [Astier et al., 2006] comprised of 73 supernova and the larger Davis et al. [2007] survey composed of 192 supernova taken from Riess et al. [2007] and Wood-Vasey et al. [2007]. The two data sets are shown in Figure 4.2. Figure 4.2 shows that the Davis et al. [2007] data covers a much wider range of redshift and contains much less noise. Intuitively, the Davis et al. [2007] data is preferable, as it results in tighter constraints on the cosmological parameter, and hence more powerful scientific inferences. However, in some of the analyzes in the subsequent chapters we use the SNLS data, due to its earlier availability.

4.1.3 Large Scale Structure

The final data set we consider is based on observations of the large scale structures. After recombination, the baryonic matter in the Universe collapsed to form galaxies and stars. However, galaxies are not distributed uniformly throughout the Universe [Shectman et al., 1996, Falco et al., 1999, Saunders et al., 2000, Tegmark et al., 2006]. Instead, galaxies

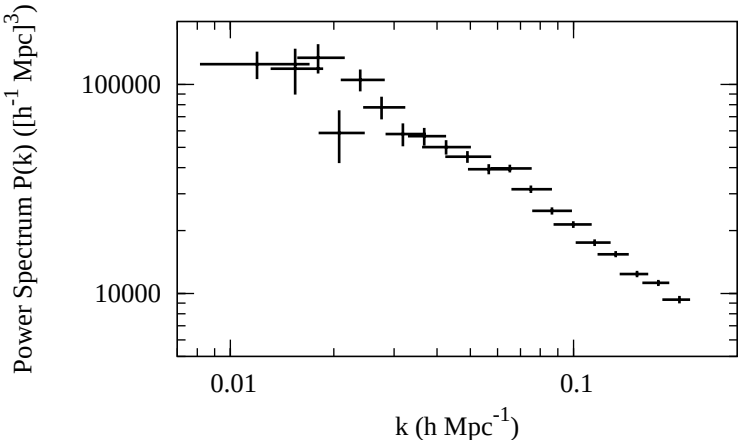


Figure 4.3: Observed galaxy power spectrum. The data shown here is from Tegmark et al. [2006].

preferentially lie in clusters, filaments, bubbles and wall like structures. To ascertain the degree of clustering, astronomers compute the two-point correlation function which measures the probability that a galaxy is a specific distance from a random neighbor, in excess of a uniform distribution. As with the CMB data, we look at the Fourier transform of the data to obtain the galaxy power spectrum. This spectrum, plotted in Figure 4.3, presents the probability of two galaxies within a specified size scale as a function of size scale. At large scales the WMAP data suggests that the spectral index, n_s, is slightly above 1.0. However, at small scales, the correlations function has a negative slope, as can be seen in Figure 4.3. Theoretical models suggest that cold dark matter is required to produce this power spectra with this "turn over" point.

Thus, the galaxy power spectrum provides useful insight into a number of cosmological parameters. Importantly, Tegmark et al. [2006] shows that the LSS data can be used to break degeneracies in the accepted parameter ranges in the WMAP data (see Section 4.2.1). For instance, while the WMAP data is able to determine $\Omega_{\rm T}$ with some accuracy, the relative contributions of $\Omega_{\rm M}$ and Ω_Λ are not well constrained. The galaxy power spectrum data provides constraints on $\Omega_{\rm M}$ allowing us to break this degeneracy. Moreover, the LSS data produces strong constraints on n_s and H_0, as we will see in Section 4.2.3. Here, we use the theoretical models of Tegmark et al. [2006] to compute likelihood estimates of the data given the parameter vector $\mathbf{p} = (\Omega_{\rm k}, \Omega_\Lambda, \omega_{\rm c}, \omega_{\rm B}, n_s, A_s, \alpha, b, Q_{nl})$. These models take a fraction of a second to compute, on par with the supernovae calculations, and significantly faster than CMBFast computations. However, in order to reduce the parameter space of our entire collection of cosmological parameters to a more manageable

size, we fix A_s, α, and Q_{nl} to their maximum likelihood estimate values given in Table 4.1. These variables are either weakly constrained by the data, or are "nuisance" parameters, introduced to describe the non-linearity at small scales [Tegmark et al., 2006]. As a result, we are left with an eight dimensional parameter space which can be used to describe the three different data sources.

4.2 Statistical Inference Results

In the previous section, we described three independent data sets which can be used to constrain the values of the cosmological parameters listed in Table 4.1. Let us now use the inference methods described in Chapter 3 to compute statistically valid joint $1 - \alpha$ confidence regions for the parameters. In this section we look at the tasks of computing $1 - \alpha$ confidence regions for the WMAP first year data and the supernovae data sets independently, and then demonstrate how our techniques can be used to compute $1 - \alpha$ regions using combination of all three data sources. We conclude by comparing our results with those from the literature.

4.2.1 Cosmic Microwave Background Result

The first analysis we will examine uses the WMAP first year CMB temperature-temperature power spectrum data described in Section 4.1.1.[7] Most CMB power spectrum parameter estimations to date have been done via Bayesian techniques (e.g., Knox et al. [2001], Gupta and Heavens [2002], Spergel et al. [2003], Jimenez et al. [2004], Dunkley et al. [2005]). However, there have also been undertakings to estimate cosmological parameters using frequentist techniques, such as χ^2 tests [Gorski et al., 1993, White and Bunn, 1995, Padmanabhan and Sethi, 2001, Griffiths et al., 2001, Abroe et al., 2002] and Bayes risk analyses [Schafer and Stark, 2003]. Here, we show how statistical inferences can be computed using the confidence ball procedure in combination with our active learning algorithms (as described in Section 3.3.3).

The first step of the confidence ball procedure is to compute the non-parameter fit, and then determine the radius of the resulting confidence ball. Here we use the results from Genovese et al. [2004]. In Figure 4.4, we compare the non-parametric fit of Genovese et al. [2004] to a model-based fit from Spergel et al. [2003]. Points in the figure depict the

[7]This work was originally published in Bryan et al. [2005] and Bryan et al. [2007b] with co-authors Jeff Schneider, Christopher J. Miller, Robert C. Nichol, Christopher R. Genovese, and Larry Wasserman.

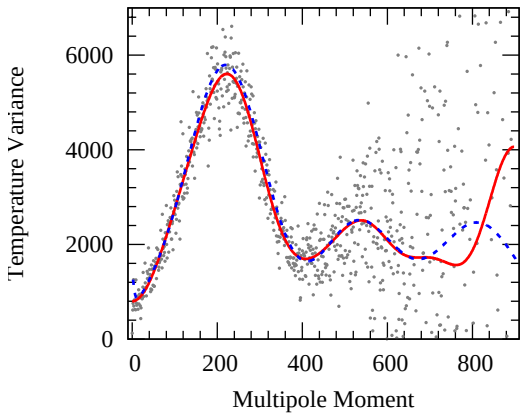


Figure 4.4: Comparison of our nonparametric fit of the CMB power-spectrum (solid) with Spergel et al. [2003] parametric fit (dashed). First-year WMAP data (dots) are shown without errors for clarity.

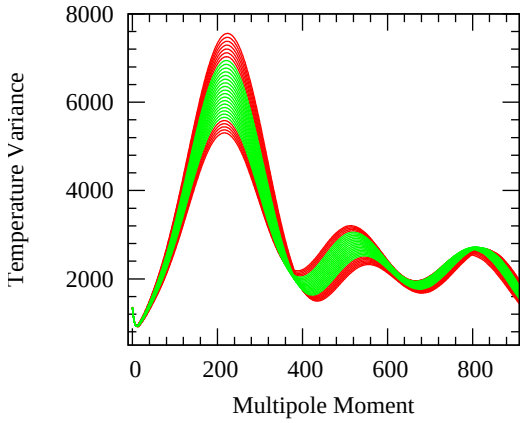


Figure 4.5: A "ribbon" plot depicting the effect of varying ω_{B} while all other parameters remain fixed (at concordance values). Green lines indicate those models which are contained within a 95% confidence ball, while the red lines indicate those models rejected by the hypothesis that the model and the regressed fit are the same.

first year WMAP data. Error bars are omitted for clarity. The full estimated covariance, Σ, is used in both the Spergel et al. [2003] model fit and the Genovese et al. [2004] nonparametric fit. In Table 4.2, we compare our maximum likelihood estimate based on the Genovese et al. [2004] nonparametric fit, with the mean estimate computed by Spergel et al. [2003] using χ^2 tests. Empirically, we find that similitude between the studies is remarkably close, given that the studies used different statistical tests and parameters of interest.

The combination of the non-parametric fit and the $1-\alpha$ confidence ball radius forms a high-dimensional ellipse; parameter vectors which result in models that are contained within this ball will be accepted, while those outside will be rejected at a confidence level of α. As mentioned in Section 4.1.1, we consider the parameters space spanned by the parameters $\tau, \Omega_\Lambda, \Omega_{\mathrm{M}}, \omega_{\mathrm{DM}}, \omega_{\mathrm{B}}, f_\nu$, and n_s. Since the dimensionality of our space is large, it is difficult to visualize the confidence region that surrounds the non-parametric fit. However, we can show examples of functions which live inside (or outside) our confidence region by calculating their distance from the nonparametric fit to the data. In Figure 4.5, we show a "ribbon" plot for ω_{B} around the concordance model. This figure is generated

Parameter	Our MLE Model	Spergel et al. [2003] Mean Model
τ	0.177	0.166
Ω_Λ	0.73	0.71
Ω_M	0.27	0.29
ω_DM	0.1161	0.116
ω_B	0.0238	0.024
f_ν	0.0	0.0
n_s	1.0	0.99

Table 4.2: Our maximum likelihood estimate parameter estimate compared with the mean parameter estimate from Spergel et al. [2003]. The models generally agree quite well, considering that different assumptions and parameters were used in the analysis.

by setting all of the cosmological parameters to their maximum likelihood estimate values and then slowly evolving ω_B from 0.012250 to 0.036750 to depict the range of temperature spectra allowed due to uncertainty of ω_B. The green curves correspond to cosmological models which live within the 95% confidence ball, while the red curves are models that do not. As can be seen in this figure, the shape of the confidence region is not simply a band of constant width surrounding the best fit. It is, in fact, a very complicated, possibly disconnected surface in our high-dimensional parameter space.

Using our active learning algorithm with the straddle heuristic as described in §3.3.3, we have sampled just over 1.2 million CMBFast models to learn the surface of this $1-\alpha$ confidence region, while $\alpha = 0.61, 0.32, 0.13,$ and 0.05. These various levels of α correspond roughly to $0.5, 1.0, 1.5$ and $2.0-\sigma$ confidence regions. The result of our 1.2 million models was a "primary" data set used to compute the $1-\alpha$ confidence regions.

Additionally, we sampled another 100 thousand models uniformly at random throughout the parameter space. From the randomly sampled data, we find that less than 0.1% of the parameter space searched is within the 2σ confidence ball; that is, our set of acceptable models (those within 2σ) exclude 99.97% of all possible models defined in Table 4.1. However, the method we use to generate parameter vectors results in only 54% of the points being rejected by the hypothesis that the model and the regressed fit are the same. Note that a sampling efficiency around 50% is optimal, as points need to be selected on both the interior and the exterior of the function level-set in order to accurately determine the level-set's location. Thus, by actively searching through the space, we are able to identify and efficiently map regions of interest, while ignoring large areas of parameter space that result in models below the 2σ level.

Confidence Region Projections

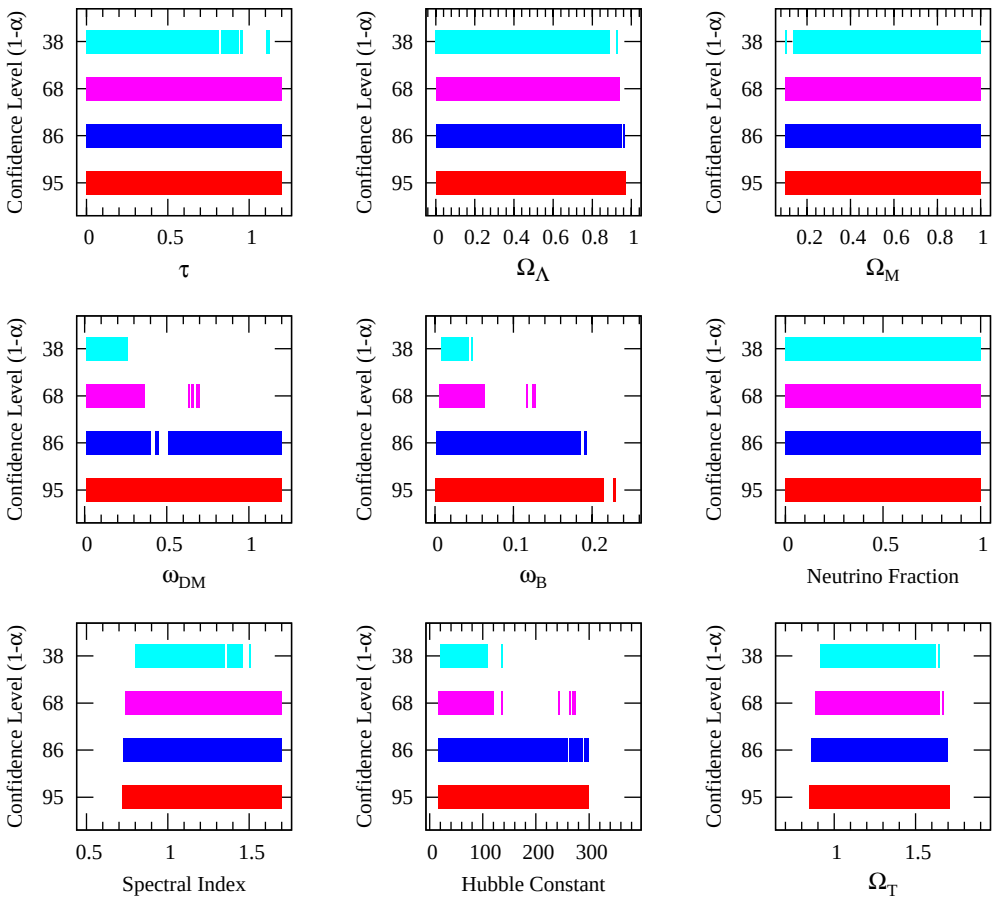


Figure 4.6: Jointly valid confidence intervals for our cosmological parameters for four values of $1-\alpha$, corresponding to $\frac{1}{2}\sigma, \sigma, 1\frac{1}{2}\sigma$ and 2σ confidence levels, respectively. Areas of solid color indicate values for the given parameter that contain the true value of cosmological parameter with probability $1-\alpha$, regardless of the values of the remaining 6 parameters.

The result of running the 1.2 million models contained in the primary data set is a set of disjoint, seven dimensional "confidence regions" in parameter space which contain all models that fall within our $1-\alpha$ confidence ball. In each of these regions, the confidence interval for a particular parameter is given by the range of values that parameter takes in that region. Thus, the confidence interval for a particular parameter will be a function of which sets of regions we consider.

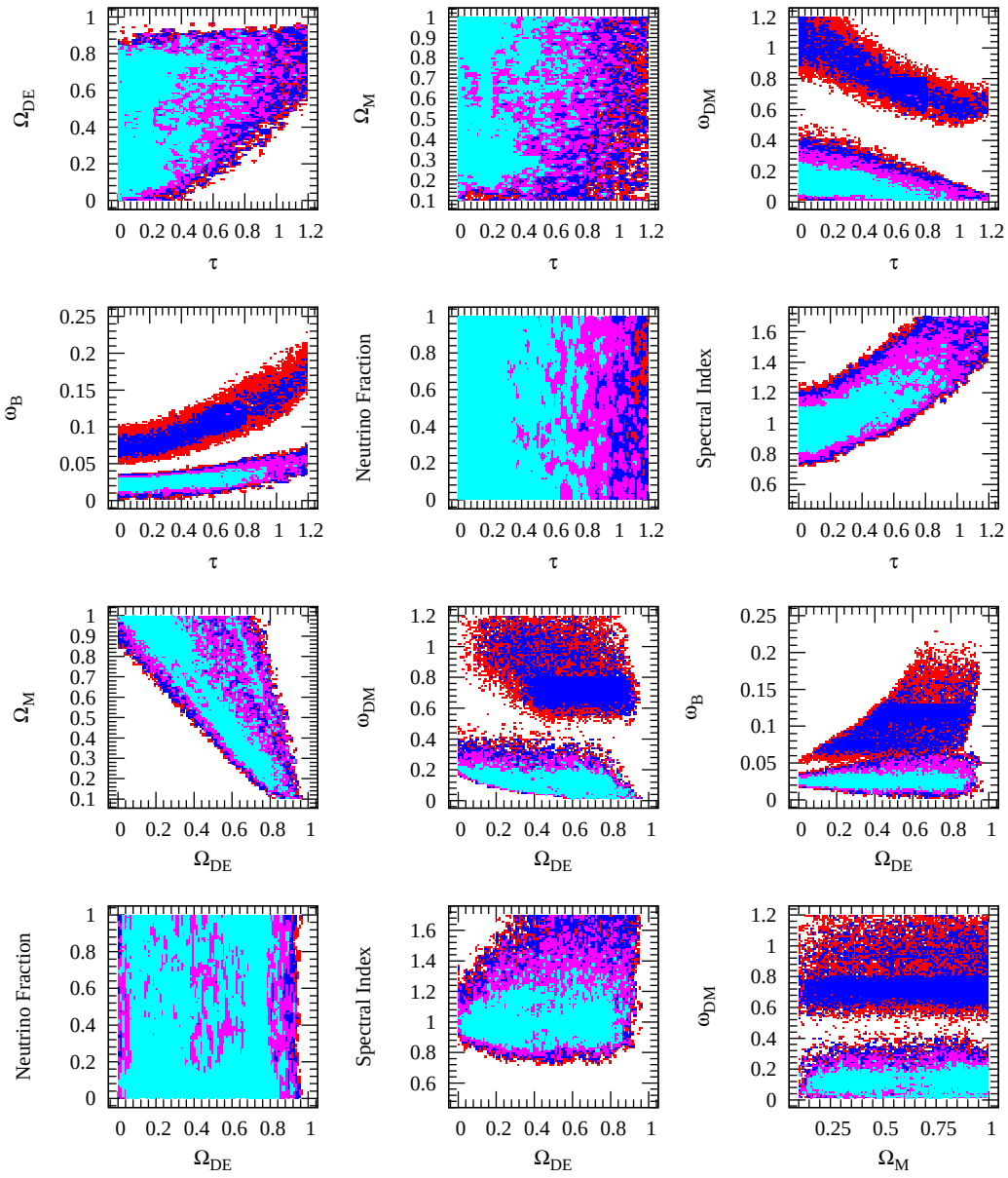
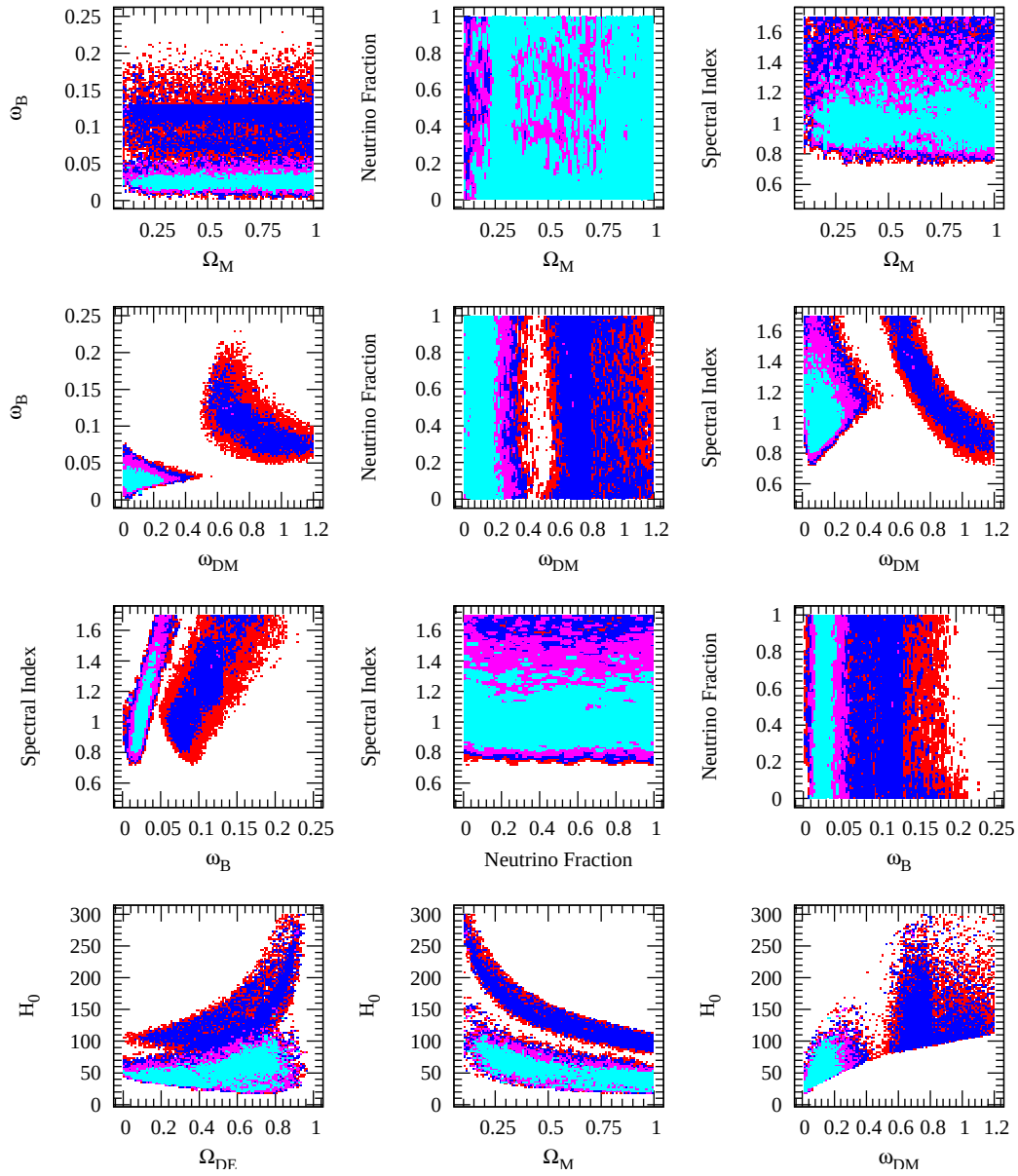


Figure 4.7: Jointly valid confidence regions for pairs of cosmological parameters, where the colors cyan, magenta, blue and red correspond to $\frac{1}{2}\sigma, \sigma, 1\frac{1}{2}\sigma$ and 2σ, confidence levels respectively. Areas of solid color indicate values for the given pair of fixed (plotted) parameters that contain the true value of cosmological parameter with probability $1 - \alpha$, regardless of the values of the remaining 5 parameters. Note there are two disjoint regions in parameter space which are above the 2σ confidence interval.



Continuation of Figure 4.7.

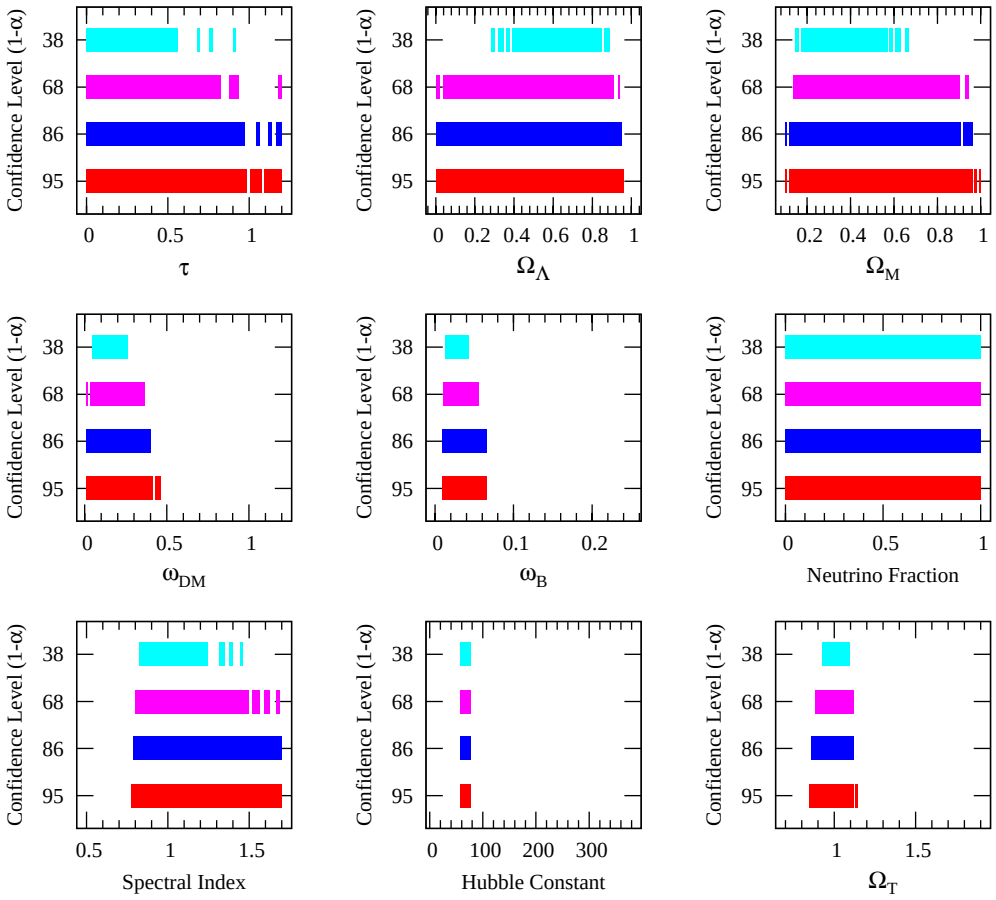


Figure 4.8: Jointly valid confidence intervals for our cosmological parameters, where we assume that that the value of H_0 is between 60 and 75$\frac{\text{km/s}}{\text{Mpc}}$. Areas of solid color indicate values for the given parameter that contain the true value of cosmological parameter with probability $1-\alpha$, regardless of the values of the remaining 6 parameters.

If we put no restrictions on the values of the other 6 parameters, then the confidence interval of a parameter will be the union of the confidence intervals for that parameter for all confidence regions. We plot these unrestricted confidence intervals in Figure 4.6 for the four values of $1-\alpha$ corresponding to 0.5, 1.0, 1.5 and $2.0-\sigma$ confidence regions. Intuitively, Figure 4.6 can be interpreted as stating that for any value of a parameter that lies within the depicted $1-\alpha$ confidence interval, there exists at least one combination of the remaining six parameters such that the resulting parameter vector lies within one of the $1-\alpha$ confidence regions.

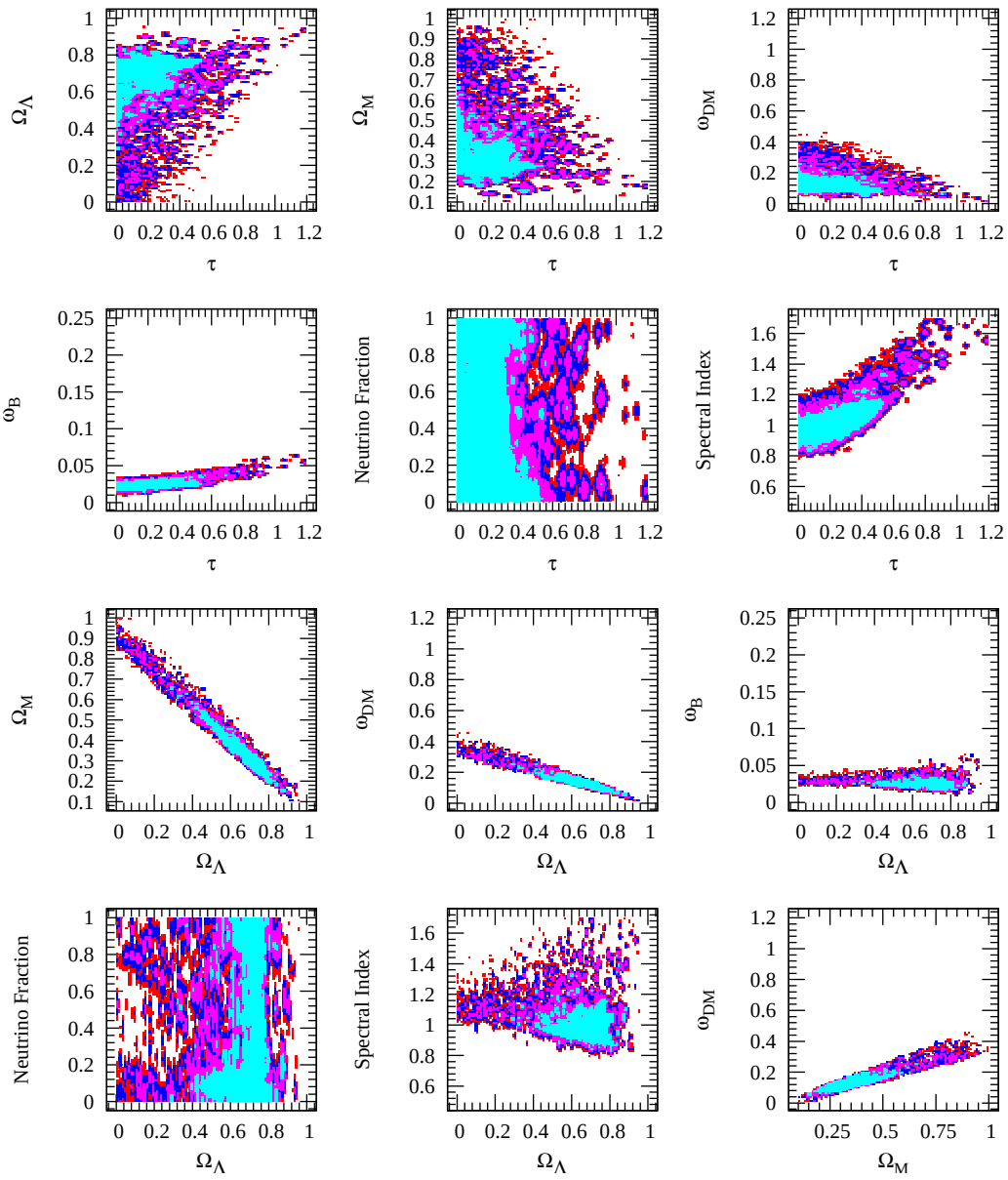
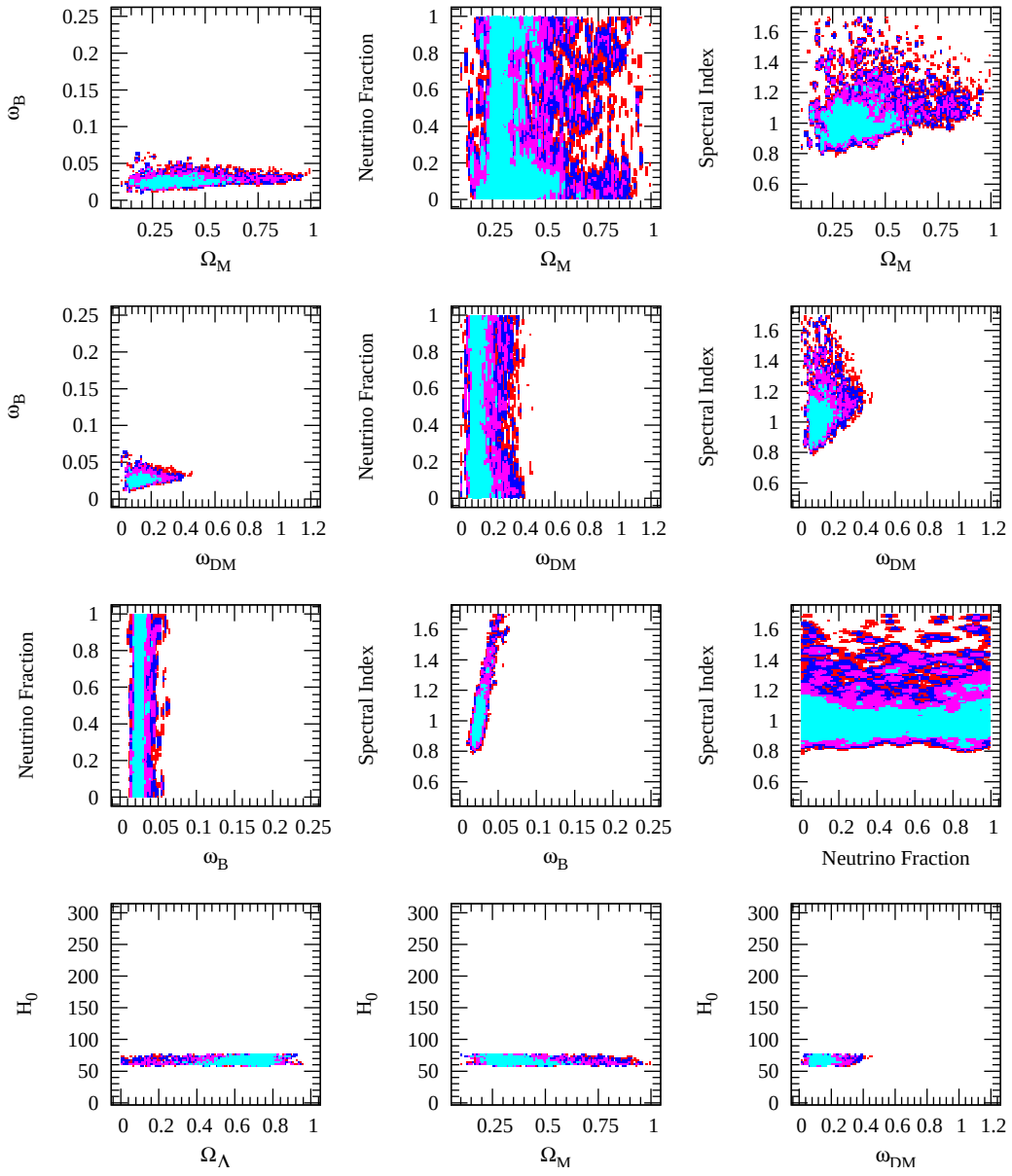


Figure 4.9: Jointly valid confidence regions for pairs of cosmological parameters, where we assume that that the value of H_0 is between 60 and 75$\frac{\text{km/s}}{\text{Mpc}}$. The colors cyan, magenta, blue and red correspond to $\frac{1}{2}\sigma, \sigma, 1\frac{1}{2}\sigma$ and 2σ, confidence levels, respectively. Areas of solid color indicate values for the given pair of fixed (plotted) parameters that contain the true value of cosmological parameter with probability $1 - \alpha$. Note that the constraint on H_0 eliminates the secondary confidence region found in Figure 4.7.



Continuation of Figure 4.9

In Figure 4.7 we depict results of interactions between pairs of parameters on the computed confidence regions. As with the one dimensional projections in Figure 4.6, points in Figure 4.7 which are denoted to be within the $1-\alpha$ confidence ball, are points where given the particular values of the two fixed cosmological parameters — those being explicitly plotted on the x and y axes, — there exists some values for the other 5 parameters such that the resulting parameter vector is within the $1-\alpha$ confidence region. While some plots show that most combinations of the fixed parameters are within the 95% confidence ball providing minimal constraints on parameters describing the Universe, others, such as $\omega_{\rm DM}$ versus $\omega_{\rm B}$ ($4^{\rm th}$ row, $4^{\rm th}$ column), show strong constraints.

Areas in Figure 4.7 which are blank (white), are areas that are rejected at the 95% confidence level; for these combinations of fixed parameters, there exists no combination of the other five parameters, such that the resulting vector is within any of our confidence regions. In particular, the plot of Ω_Λ versus $\Omega_{\rm M}$ ($2^{\rm nd}$ row, $3^{\rm rd}$ column) illustrates that $\Omega_{\rm Total} \gtrsim 0.9$, while the plot of $\omega_{\rm DM}$ versus $\omega_{\rm B}$ shows that there are at least two disjoint confidence regions in our seven dimensional space. These disjoint regions in Figure 4.7 correspond directly to the split confidence intervals observed in Figure 4.6. We defer further discussion of the disjoint confidence regions to Section 4.2.1. Smaller splits in the confidence intervals observed in nearly every plot in Figure 4.6 are a result of the fact that CMBFast does not return models which are perfectly continuous in the parameter space. While one may expect the derived confidence level to be smooth in parameter space, this is not the case. We observe small discretizations and inconsistencies in the power spectrum model, which result in the confidence ball having a jagged, nebulous surface (as observed in Figure 4.7), rather than a perfectly smooth one. We will elaborate on this observation in Section 4.2.1.

As illustrated in Figure 4.6, the confidence intervals for most parameters are not well constrained by the WMAP data alone. In particular, the constraint on the Hubble constant, H_0, is so weak as to allow values between 15 and 300 at the two sigma level; even at the one sigma level, H_0 ranges between 15 and 150 with additional fits at $H_0 \sim 250$. The confidence intervals derived here cover the Bayesian credible intervals found in the literature using a variety of techniques (e.g. Tegmark et al. [2001], Spergel et al. [2003, 2007]), as shown in Table 4.3. The results in Table 4.3 show that the parameter ranges derived by both the frequentist and Bayesian analyses are approximately centered on the same values. However, we are not in any way attempting to argue that the allowed parameter ranges are better, or worse, than those derived from alternative methods, as the comparison of credible (Bayesian) and confidence (frequentist) parameter regions is non-trivial.

While this assessment may appear bleak, there is underlying structure to the confidence regions, hinted at by the disjoint regions in Figure 4.7. Suppose we restrict the range of

Parameter	No Constraints	$60 \leq H_0 \leq 75$	$n_s < 1$ $60 \leq H_0 \leq 75$	Spergel et al. (2003)	Spergel et al. (2006)
τ	0 – 1.2	0-0.94, 1.17-1.2	0 – 0.4	0.095 – 0.242	0.058 – 0.117
Ω_Λ	0 – 0.94	0 – 0.94	0.39 – 0.9		
$\Omega_{\rm M}$	0 – 1.0	0.13 – 0.95	0.13 – 0.59	0.22 – 0.36	0.199 – 0.273
$\omega_{\rm DM}$	0 - 0.36, 0.62 - 0.70	0.0 – 0.36	0.03 – 0.2		
$100\omega_{\rm B}$	0.5 - 6.2, 11.5 - 12.7	1.3 – 5.5	1.3 – 3.2	2.26 – 2.51	2.15 – 2.31
f_ν	0 – 1	0 – 1	0 – 1		
n_s	0.73 – 1.7	0.8 – 1.7	0.84 – *1.0*	0.95 – 1.03	0.944 – 0.978
σ_8				0.82 – 1.02	0.71 – 0.81
H_0	17 - 135, 243 - 272	*60 – 75*	*60 – 75*	67 – 77	70.3 – 76.7

Table 4.3: Derived 68% confidence intervals. Those to the left of the solid line are derived from Figures 4.6, 4.8 and 4.10 respectively, while those to the right are quoted from referenced literature.

a subset of our parameters and then compute the confidence intervals for the remaining parameters. Since our statistical model is independent of the ranges searched, we can compute these conditional confidence intervals without re-running any models. For any restriction of our parameter space, the confidence interval for a parameter of interest will be the union of the confidence intervals for that parameter over those confidence regions which obey our restriction. For example, in Figures 4.8 and 4.9 we show the effect on the confidence intervals and regions, respectively, of imposing the restriction that H_0 is between 60 and $75\frac{\rm km/s}{\rm Mpc}$. Note that with this restriction on H_0, the confidence intervals agree much better with the current estimate of the cosmological matter/energy budget and strongly suggest that $\Omega_{\rm Total} = 1$.

This analysis exhibits the power of our statistical inference technique: we can test constraints on one parameter, and see their effects on the remaining parameters without additional CMBFast computation or invalidation of our previous statistical inferences. To this end, we have created a graphical interface that can be used to apply constraints and view the resulting effects in real time. This tool, along with the necessary data files, can be downloaded from `http://gs3636.sp.cs.cmu.edu/visualizer/`.

In the Bayesian view, the tightening of the allowable regions between Figures 4.6 and 4.8, and Figures 4.7 and 4.9 is analogous to what would occur when priors (either informative or non-informative) are applied. Such priors are universally applied in CMB cosmological analyses. As an example of how we can use this technique to better understand the cosmological confidence surface, we focus in on one or two parameters and utilize the graphical interface described above.

WMAP Three Year data show that a scale invariant spectra ($n_s = 1$) is not a good fit to the WMAP Three Year data alone. If we place both the constraint that $n_s < 1$ and that $60\frac{\mathrm{km/s}}{\mathrm{Mpc}} \leq H_0 \leq 75\frac{\mathrm{km/s}}{\mathrm{Mpc}}$ on the WMAP One Year data, we see in Figure 4.10 that $\tau, \omega_{\mathrm{B}}$, and ω_{DM} are much better constrained. More importantly, we see that the allowable ranges on ω_{DM} are forced into a single confidence range, in agreement with previous studies [Spergel et al., 2003].

Exploring the high ω_{DM} space shown in Figure 4.6, we find that models consistent with high ω_{DM} have large values of ω_{B} (> 0.05), as well as large Hubble constants ($> 100\frac{\mathrm{km/s}}{\mathrm{Mpc}}$). Both of these parameters are much better constrained in the WMAP Three Year data. This leads us to predict that the second confidence surface peak in the WMAP Three Year Data is less significant than in the WMAP One Year data (although this has yet to be shown).

Convergence

Ideally, one would like to prove that our mapping from confidence ball radius to parameter space has converged. This could be done, for instance, by proving that our approximating model of spectrum distance as a function of cosmological parameters – that is our Gaussian process – has converged to the true values in those areas where the true values are near the radius of the $1 - \alpha$ confidence ball. However, this effort has been confounded by a lack of continuity in the results returned by CMBFast. The method presented here is not more susceptible to discontinuities than other techniques. Indeed, the convergence of most, if not all, inference methods will be adversely effected by the discontinuities of CMBFast models we observe in parameter space.

One standard assumption of function approximators is that of smoothness; that is that the underlying function to be modeled is continuous and differentiable. For Gaussian processes, this assumption motivates the usage of a covariance matrix in determining the relative weights of known samples when estimating values for unknown points. In this paper, we have also assumed that the covariance function is fixed over the entire space – that is that the underlying covariance is isotropic and homogeneous. These assumptions allow us to compute error bounds for each point in space, and enable us to determine when the model has converged to the underlying function.

However, experimentation shows that the underlying CMBFast function does not fulfill the continuous and differentiable assumptions, as shown in Figures 4.11 and 4.12. Both figures were produced by plotting the resulting model distance as we varied one parameter and kept the other six parameters fixed. Figure 4.11 shows a discretization effect that we believe is a result of integral approximations done by CMBFast. Discretization effects are common in simulated environments and it is reasonable to assume that the true function

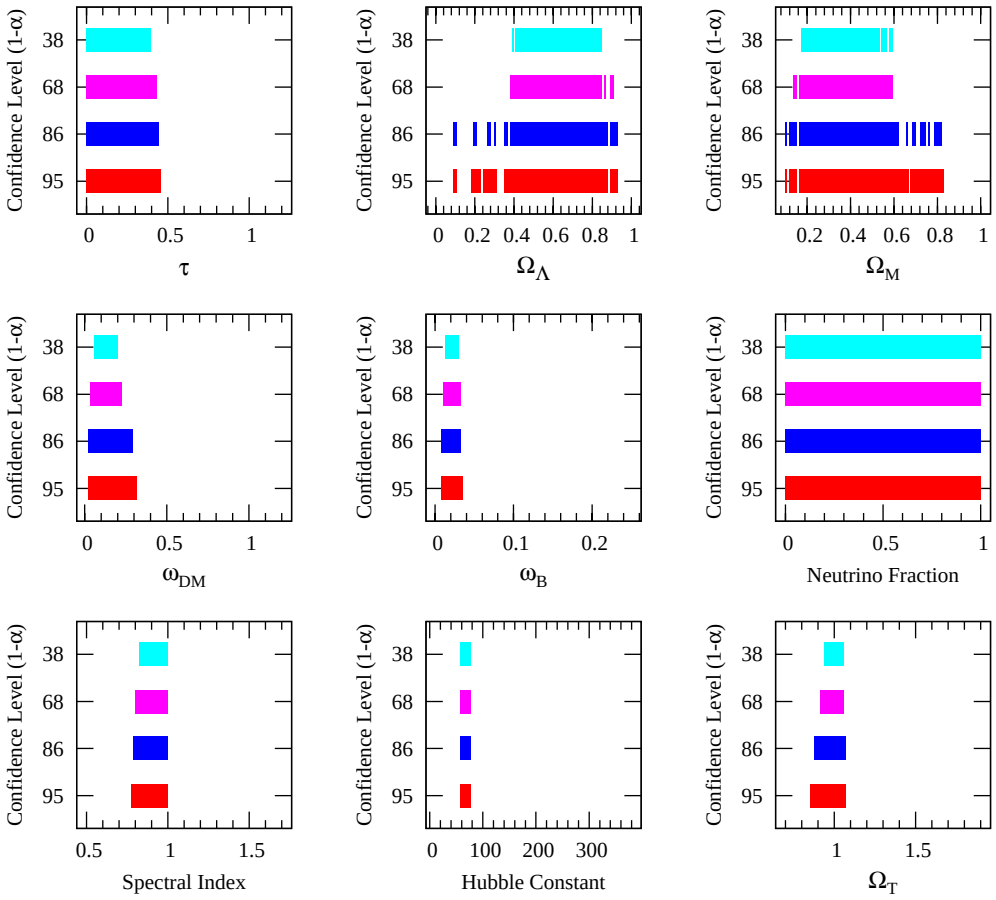


Figure 4.10: Jointly valid confidence intervals for our cosmological parameters, where we assume that $60\frac{\text{km/s}}{\text{Mpc}} \leq H_0 \leq 75\frac{\text{km/s}}{\text{Mpc}}$ and $n_s < 1$. Areas of solid color indicate values for the given parameter that contain the true value of cosmological parameter with probability $1 - \alpha$, regardless of the values of the remaining 6 parameters.

varies smoothly. More startling are the discontinuities revealed in Figure 4.12. Figure 4.12 shows that while on a broad scale the CMBFast function appears smooth, when one looks closer and closer, the function begins to act quite erratically. Of particular interest are the large discontinuity at $\Omega_\Lambda = 0.446516$ and the seemingly random deviations from a smooth function throughout the entire range. These fluctuations in distance are not caused by random noise from CMBFast; CMBFast's output is deterministic given an input parameter vector.

There are two important implications of the results in Figures 4.11 and 4.12. First, we

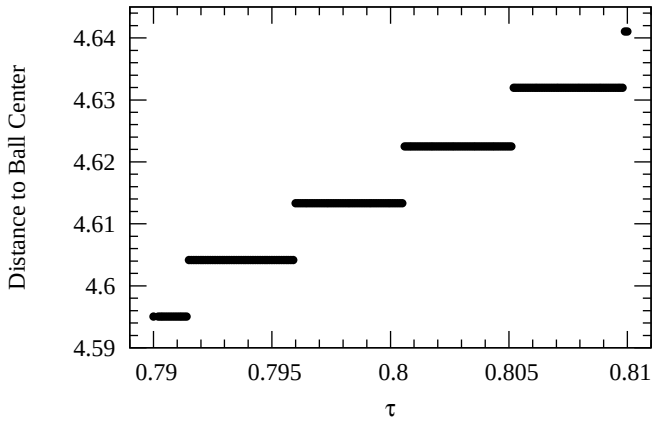
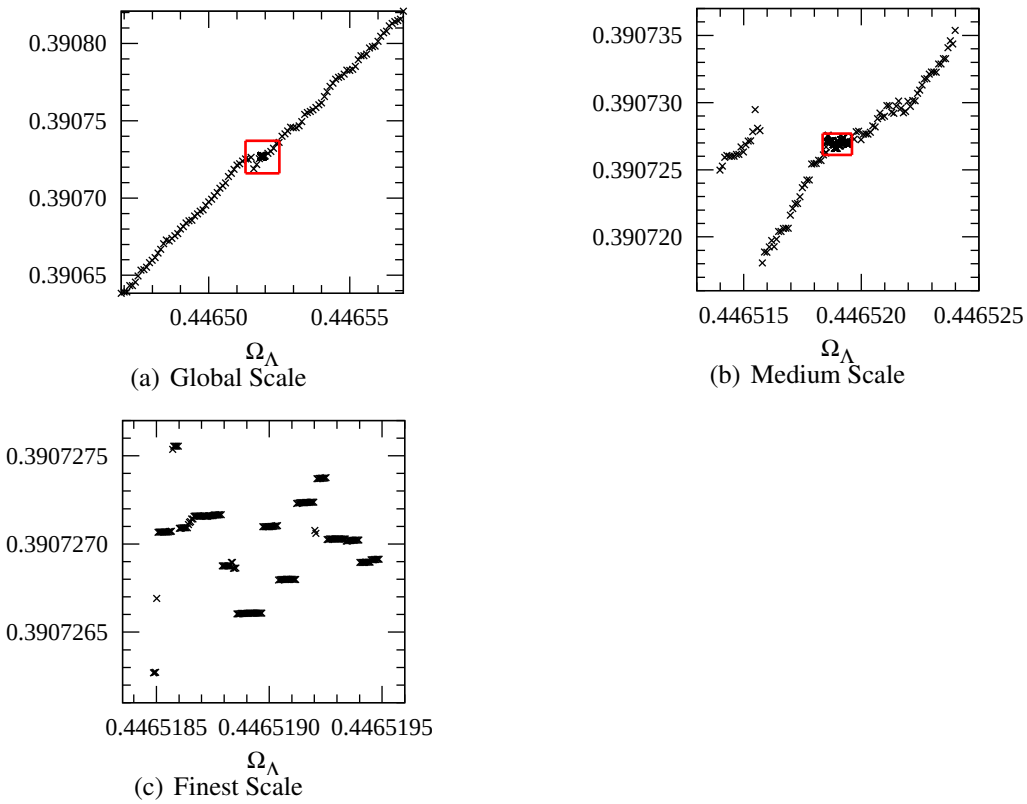


Figure 4.11: A plot of spectra distance as a function of τ, with all other parameters fixed, showing the discretization of CMBFast. For these experiments $\vec{x} = \{\tau, \Omega_\Lambda, \Omega_\mathrm{M}, \omega_\mathrm{DM}, \omega_\mathrm{B}, f_\nu, n_s\} = \{\tau, 0.0, 0.2, 0.8, 0.003, 0.0, 1.2\}$.

note that when parameter values result in spectra that are very close to the confidence ball radius, it is impossible to predict which side of the boundary a given point will be on, due to the inherent noise in CMBFast. For regions where many points are near the confidence ball radius, we will obtain spotty, jagged boundaries between those areas in the ball and those not. Second, the effects plotted in Figures 4.11 and 4.12 do not appear on the same range scales. This makes it more difficult to determine the correct level of smoothing, and hence discover the true underlying function. Thus, while it is still possible to deduce approximate covariances among the variables, it becomes impossible to ensure the model has correctly converged to the true model.

In Section 2.2, we noted that our active learning framework was able to learn the level-sets of the 2D Deboor model, a discontinuous target function. However, the primary difference between the CMBFast output and the 2D Deboor function is that the 2D Deboor function is discontinuous in specific regions away from the level-set of interest, while the CMBFast models are discontinuous over the entire parameter space. Thus, while we can use our active learning framework to learn an approximate level-set corresponding to the $1 - \alpha$ confidence regions, we cannot learn the exact level-set.

We note that this lack of continuity will adversely effect the convergence of any model that relies on the smoothness of the underlying function, be it MCMC or Gaussian processes. In the case of MCMC, the discontinuities in the variance weighted sum of squares between the models computed by CMBFast and the data require that comprehensive sampling of the posterior be performed to ensure that the peaks and valleys in any local region



(a) Global Scale (b) Medium Scale

(c) Finest Scale

Figure 4.12: A plot of spectra distance as a function of Ω_Λ, with all other parameters fixed. The square boxes in each of the left two plots denotes the area enlarged in the neighboring plot to the right. Note that while on the global scales, (a), the mapping appears to be smooth, closer inspection (b), (c) reveal numerical errors resulting from approximations used in CMBFast.

are correctly averaged out, allowing the integral over the posterior to be correctly computed. While we can run both methods in a mode that smooths over these discontinuities (by effectively ignoring them), we must realize that the resulting algorithms will converge to a solution that is incorrect. Additionally, increasing the sampling of either algorithm would eventually turn up the existence of these discontinuities, and the system would jump from an apparent convergence in the smoothed case, to a new convergence where discontinuities are considered.

Connectivity

As Figure 4.7 show, there are two main peaks that lie above the 1σ confidence ball radius. As a test of the function approximate convergence, we conducted focused tests to see if these peaks were truly connected. In particular, we used the semi-variance matrix of the Gaussian process to compute the maximal influence distance from a given point one could travel before possibly encountering the $1 - \alpha$ confidence ball radius. We then created clusters of points above the 68% confidence ball radius using a friends-of-friends algorithm; that is, a point is added to an existing group if it is within the maximal influence distance of any point currently in the group. Starting with all points in their own groups, we first passed through the data, merging groups where possible. Then, additional points were sampled between existing groups, using an A* like algorithm [Hart et al., 1968]. For two groups A and B, we found the point, x, in A that was closest to any point in B. We then created a set of candidate points within the influence distance of x, and add them to a queue, $\mathcal{P}$, sorted according to their distances to B. We then take the point p from $\mathcal{P}$ that is closest to B run it through CMBFast and compare to our confidence ball. If p is within our confidence radius, then we create candidate points for p (just as we did for x) and add them to $\mathcal{P}$. Otherwise, we remove p from $\mathcal{P}$. This procedure is repeated until either B is within the influence distance of p or we exhaust $\mathcal{P}$.

The primary data set contained roughly 2000 distinct groups, which were quickly merged using the friends-of-friends algorithm. This left us with 2 major clusters shown in Figure 4.7. Using the algorithm noted above, we were unable to find connections between the main peak and the secondary peak, even after multiple attempts starting from different locations. We believe that there exists no smooth transition of variable parameters that leads from the concordance to the secondary peak. The second peak is not just an extension of the concordance peak that appears disjoint due to under sampling or projection effects.

Comparison to Grid Based Approaches

Finally, let us compare the efficiency of our active learning algorithm with grid-based approaches. While it would be interesting to compare our algorithm with MCMC as well, the efficiency of MCMC is hard to calculate in high-dimensions, as the truth depends on the samples chosen (see Section 3.6.4); we will discuss this issue more in Section 4.2.2. Therefore, let us restrict our comparison to the efficiencies of our algorithm and grid-based approaches.

Given the high dimensionality of the parameter space, and the thirty second to ten

	Peak Center		# Points in Effective Radius	
	$\omega_{\rm DM}$	$\omega_{\rm B}$	Grid	Straddle
MLE Peak	0.116	0.024	4943	25689
Secondary Peak	0.665	0.122	0	5488
Total Points			5613300	603384

Table 4.4: Number of points found in the two peaks for the grid based approach of [Tegmark et al., 2001] and our straddle algorithm.

minute cost of computing a single CMBFast model, a simplistic grid based approach with only 10 samples per dimension would require between 10 and 200 CPU years to compute.[8] However, a 10 point per dimension grid is far to sparse to be scientifically useful. A more informative grid with 50 points per dimension would take over a million CPU years to compute. Even utilizing parallel computing infrastructures, this computational requirement is clearly unfeasible. For comparison, the 1.2 million CMBFast models used here took roughly 4 years of CPU time to compute.

Instead, Tegmark et al. [2001] suggests the use of an adaptive grid. This grid contains between 7 and 11 points per dimension, and samples for each dimension are placed according to an assumed Bayesian prior. Grid points are more densely concentrated where the the marginalized prior is large, and are sparely distributed where the prior is near zero. While this adaptive grid creates a dense mesh in the region of the maximum likelihood estimate model (see Table 4.2), — hence should produce scientifically useful constraints — the fact that it ignores large regions of space limits its value. In particular, we note that the adaptive grid used by Tegmark et al. [2001] fails to sample a single point with the secondary peak shown in Figure 4.7. As a result, a posterior derived using this adaptive grid would underestimate the normalization constant c. Hence, Bayesian analyses would underestimate the size of the credible regions about the concordance peak, as well as miss the contributions of the secondary peak to the $1 - \alpha$ credible regions.

On the other hand, our algorithm samples this secondary peak frequently, while our statistical procedure is independent of the actual experiments chosen. A comparison of the number of samples within both the maximum likelihood estimate and secondary peak for our active learning algorithm and the grid based approach used by Tegmark et al. [2001] is shown in Table 4.4. Even with only 10% of the experiments used in the grid approach, we sampled the concordance peak five times more frequently than the grid based approach. Moreover, Table 4.4 shows that the the grid completely ignored the secondary peak, while

[8]This calculation is based on benchmarks made on 1.6Ghz Opteron machines.

our method sampled it over 5000 times. These results dramatically illustrate the power of our active learning algorithm, and show how it is significantly more efficient that adaptive grid-based approaches.

4.2.2 Supernovae Results

We now turn to the second statistical analysis, this time using supernovae data. In the previous section, we computed $1 - \alpha$ confidence regions for the WMAP first year data using the confidence ball procedure. For variety, here we use the minimax expected size (MES) confidence procedure described in Section 3.4.

While the MES technique yields near optimal $1 - \alpha$ confidence regions in practice, it requires much more upfront computation. Besides performing a search of the confidence region boundary, as we did for the confidence ball approach, we must also construct and solve a convex game. Moreover, as seen in Section 3.4.4, it is generally preferable to perform a χ^2 cut, to further restrict the size of the parameter space, Θ. This restriction allows us to solve a smaller convex game and eliminate from consideration those parameter vectors which are very unlikely.

In this section, we will compute 95% confidence regions for both the SNLS data set [Astier et al., 2006] and the data set compiled by Davis et al. [2007], using the MES algorithm given in Section 3.4.5. For both analyses, we will use a χ^2 cut at level $\alpha = 0.005$ to create the restricted parameter space Θ'. Active learning is used to learn both Θ', and the resulting confidence region boundary. The experiments shown here use a sparse matrix representation, with a cutoff value, ϵ_t, of 1×10^{-32}. As we saw in Section 3.4.5, even with this small cutoff value, the payoff matrix of the convex game is 85% sparse, leading to a large computational speed-up.

We begin by discussing the analysis using the SNLS data, and then present the results from the Davis et al. [2007] data.

Results Based on the SLNS Data Set

In Figure 4.13 we plot the 95% confidence regions derived from the SNLS data set using the MES approach described in Section 3.4.5 (blue), along with 95% confidence/credible regions derived using χ^2 tests (red), and the grid-based Bayesian technique (green) described in 3.5.1. Bayesian intervals were estimated by numerically integrating the likelihood function over the entire space (using a finely spaced grid) and then computing the maximum a posterior 95% credible region, as discussed in Section 3.5. As with the re-

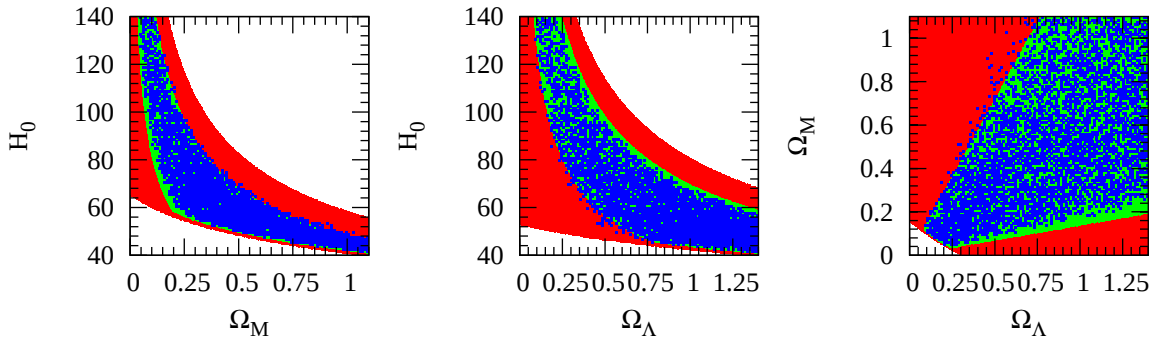


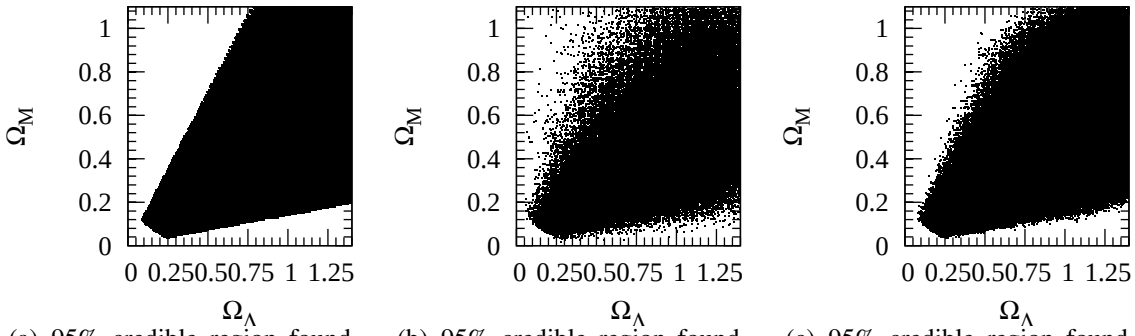
Figure 4.13: 2D projections of the confidence regions of the parameters H_0, Ω_{M}, and Ω_Λ based on the Supernova Legacy Survey data set, using χ^2 tests (red), a Bayesian technique (with a uniform prior) (green), and MES (blue). MES has roughly the same power as the Bayesian technique and significantly more power than χ^2 tests.

gions presented in Section 4.2.1, a region of color indicates that for some value of the third (non-plotted) parameter, the resulting model is accepted by the specified inference technique.

Figure 4.13 illustrates that the confidence region derived by MES is much smaller than the associated χ^2 confidence region and similar in size to the 95% Bayesian credible interval. While the MES and Bayesian intervals seem comparable, recall that the MES interval guarantees 95% coverage, while the Bayesian interval does not. In general, the confidence regions produced by the MES procedure are similar to the credible regions of a Bayesian analysis when the Bayesian approach uses a prior distribution which is similar to the likelihood of the parameter space given the observed data.

Taken together, the panels of Figure 4.13 suggest that the parameters may be correlated. In fact, looking at Equation 4.3, we can clearly see this dependence. Note that if we increase Ω_{M} and Ω_Λ by a constant factor, then decreasing H_0 by the square of that factor results in the exact same model. Thus, there are not really three independent parameters, but two. This dependence results in the clear correlation between Ω_{M} and Ω_Λ and the inverse correlation between H_0 and Ω_{M} (and Ω_Λ). These correlations can easily be broken by including data from other astronomical observations, such as the CMB data described earlier.

MCMC versus Grid Based Numerical Integration In Figure 4.13, we compared the results of the MES 95% confidence region with that of the Bayesian 95% credible region. Bayesian credible regions are most commonly produced by using MCMC to sample the



(a) 95% credible region found using numerical integration with 8 million experiments

(b) 95% credible region found using MCMC (Metropolis-Hastings) with 8 million experiments.

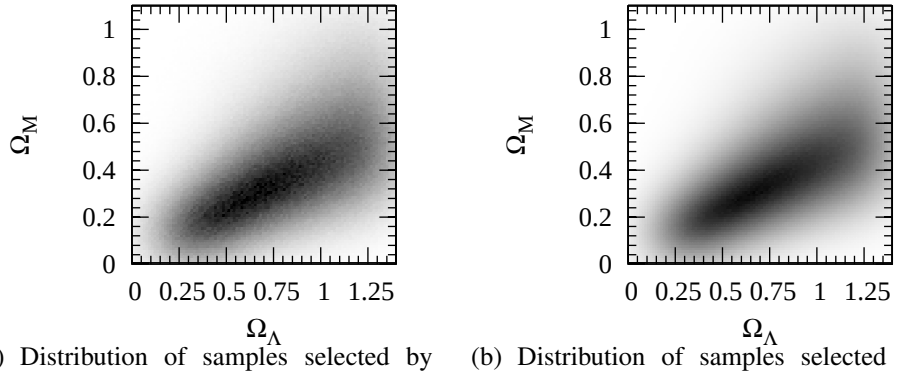
(c) 95% credible region found using MCMC (Metropolis-Hastings)with 125 million experiments.

Figure 4.14: Derived 95% credible regions for the SNLS data set using numerical integration (a) and MCMC (b & c). Note that MCMC is not necessarily data efficient for finding credible region boundaries.

parameter space and then choosing the highest probability density (HPD) regions which enclose 95% of the probability mass. However, for the reasons discussed in Sections 3.5.2 and 3.6.4, MCMC is actually a fairly inefficient algorithm for computing credible regions. Specifically, MCMC is prone to selecting samples in regions of high probability, virtually ignoring parts of parameter space which are on the boundary of the credible region.

In Figure 4.14 we compare the 95% credible regions derived using MCMC sampling (with the Metropolis-Hastings algorithm) with those found using numerical integration where the samples were taken from a uniform grid in parameter space. For both approaches, we used 8 million samples. This corresponds to a grid with 200 points per dimension for the numerical integration technique, a fairly fine sampling. Figure 4.14 shows that the numerical integration has "converged" to a reasonable 95% credible region, while MCMC has not. This indicates that at least on the SNLS data, MCMC is less efficient than grid-based sampling. Indeed, if we were to allow MCMC to sample an order of magnitude more points, the resulting credible regions are still not as well defined as the grid based approach, as shown in Figure 4.14(c).

Figure 4.15 depicts the distribution of experiments used by the MCMC algorithm for both 8 and 125 million experiments. These are the points that are either accepted or rejected by the Metropolis-Hastings algorithm (not the points that become part of the Markov chain). The figure shows the MCMC algorithm heavily over-sampling regions of high-probability, leading to the unrefined credible region boundaries seen in Figure 4.14(b),



(a) Distribution of samples selected by MCMC (Metropolis-Hastings) with 8 million experiments

(b) Distribution of samples selected by MCMC (Metropolis-Hastings) with 125 million experiments

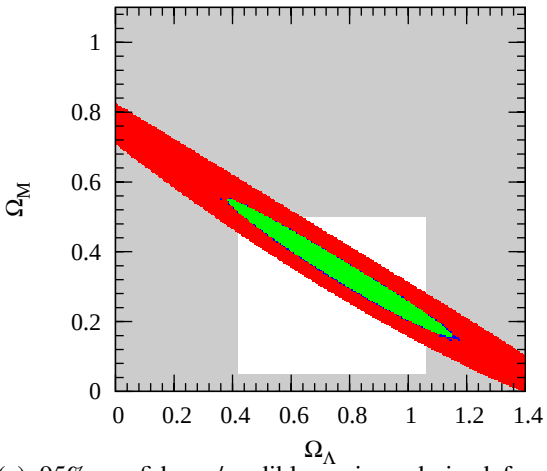
Figure 4.15: Distribution of experiments computed by MCMC (not samples chosen for inclusion in the Markov chain). Note that MCMC heavily samples the high-likelihood areas of the parameter space, leaving relatively few samples for the regions defining the 95% HPD credible region. Thus, the HPD regions computed by MCMC are less refined that those computed using a grid based approach (See Figure 4.14).

again illustrating the fact that MCMC is not an efficient search algorithm; it is an algorithm for computing an entire posterior distribution.

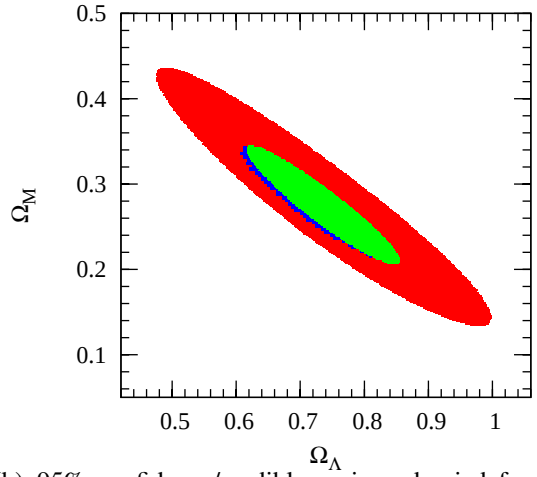
Thus, the MES $1 - \alpha$ confidence procedure described in Section 3.4 has two major advantages over the MCMC approach. First it produces $1 - \alpha$ confidence regions that are guaranteed to have correct coverage (in a frequency sense). Second, it does so in a data efficient, dimension independent manner.

Results based on the Davis et al. [2007] Data Set

Now, let us preform an analysis of the Davis et al. [2007] data, again using the MES confidence procedure. However, as noted in Section 4.2.2, there are really only two free parameters in the supernova model. As such, we fix $H_0 = 65\frac{\mathrm{km/s}}{\mathrm{Mpc}}$, and compute the 95% confidence regions for Ω_M and Ω_Λ. For comparison, we plot the 95% confidence/credible regions derived using MES (blue), the grid-based Bayesian technique (green) and χ^2 test methods (red) for both the SNLS data and the Davis et al. [2007] data. Since the ranges of the two figures are different, we depict the range of Figure 4.16(b) as a white box in Figure 4.16(a).



(a) 95% confidence/credible regions derived from the SNLS data Astier et al. [2006].



(b) 95% confidence/credible regions dervied from the Davis et al. [2007] data.

Figure 4.16: 95% confidence/credible regions computed using χ^2 tests (red), Bayesian techniques (green), and the MES procedure (blue) for the SNLS (a) and Davis et al. [2007] (b) data sets. The Davis et al. [2007] data allows for much tighter inferences for all techniques; note that the range of panel (b) corresponds to the white box in panel (a).

Unsurprisingly, the relative sizes of the derived confidence/credible regions remain constant between the data sets. The MES 95% confidence region is again significantly smaller than the corresponding χ^2 region, and similar in size to the 95% Bayesian credible region.

However, the sizes of all the 95% confidence/credible regions decrease significantly when the Davis et al. [2007] data is used. This is a direct consequence of the fact that the Davis et al. [2007] data set has both better redshift coverage and smaller measurement errors, as seen in Figure 4.2. In particular, note that the expected brightness of a supernova due to Ω_{M} is highly dependent on the redshift, while Ω_λ is not. Thus, the sharp increase in brightness at low redshifts observed in the Davis et al. [2007] data, but not in the SNLS data, allows us to disambiguate the relative effects of Ω_{M} and Ω_Λ. Hence, the statistical inferences are much tighter for the Davis et al. [2007] data.

Let us now use this data set to compare some of the properties of our active algorithms with MCMC. This comparison is possible for the supernova data, as the size of the parameter space, Θ, is only two dimensional when we fix H_0 as above, and the computation of the expected model given a parameter vector, $\theta \in \Theta$, is extremely quick. Thus, we can easily compute many instance of all of the inference techniques to obtain insights into

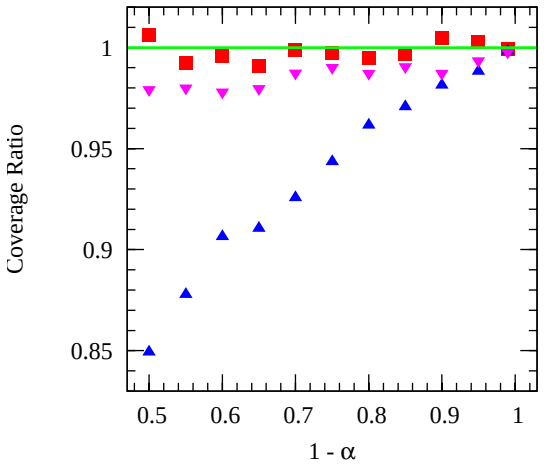


Figure 4.17: Coverage ratio (coverage fraction divided by $1-\alpha$) for χ^2 tests (red), and grid-based Bayesian technique using credible regions derived from the 2D posterior (purple) and from the 1D marginals of $\Omega_{\rm M}$ and Ω_Λ (blue).

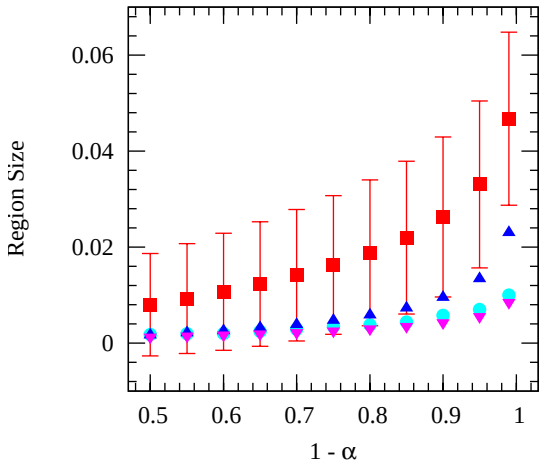


Figure 4.18: Confidence/credible region size as a function of α for χ^2 tests (red), the MES confidence procedure (cyan), and the grid-based Bayesian method using credible regions derived from the 2D posterior (purple) and from the 1D marginals (blue).

their coverage probabilities and convergence properties which we discussed on simulated data in Section 3.6.

Coverage First, let us look at the coverage properties of the χ^2, MES, and Bayesian inference techniques. Note that in order to compute the coverage probability of the truth, $\theta^\star$, we need to know $\theta^\star$. Since $\theta^\star$ is not known for the Davis et al. [2007] data set, we pick a plausible value for $\theta^\star$: $\theta^\star = \{\Omega_{\rm M}, \Omega_\Lambda\} = \{0.27, 0.73\}$. (Recall that we have already fixed $H_0 = 65 \frac{\rm km/s}{\rm Mpc}$.) Using Equation 4.3, we can compute the expected observation given $\theta^\star$ for the Davis et al. [2007] data, $\mu^\star$. We then derive simulated data by adding the observed measurement errors of the Davis et al. [2007] data to $\mu^\star$ to obtain $\tilde{\mu}$. Repeating this procedure thousands of times results in a series of hypothetical observations, given $\theta^\star$, to which we apply the three inference techniques. The coverage fraction is then the fraction of the confidence/credible regions produced by each method for the simulated values of $\tilde{\mu}$.

In Figure 4.17, we plot the ratio of the coverage fraction of each to the expected coverage fraction: $1-\alpha$. Thus, if the statistical method truly have $1-\alpha$ coverage, we expect the coverage ratio to be one for all values of $1-\alpha$ in the plot (denoted by a green line). As

expected, statistical inference based on χ^2 tests (red) results in correct coverage (considering the error). Regions derived by the MES procedure (not shown) also guaranteed $1-\alpha$ coverage. In purple, we plot the results of Bayesian inference using a grid with 300 points per dimension, where the credible regions are computed directly from the two dimensional posterior. While the coverage from the Bayesian technique using the posterior is correct when $\alpha = 0.05$, it is slightly overestimating the coverage (underestimating the size of the credible regions to obtain the correct coverage) at smaller values of $1-\alpha$.

However, that bias is small compared with the $1-\alpha$ credible regions derived using the marginal distributions for the two parameters (using the same posterior derived from a 300×300 grid.), shown in blue. When $1-\alpha = 0.5$, the credible regions derived from the marginal distributions claims to cover the truth 50% of the time, when in reality they cover the truth less than 43% of the time. Thus, credible regions derived from marginal distributions can be have coverage significantly less than expected.

Based on this example, it may seem that the obvious solution to the coverage problem for Bayesian methods would be to compute the credible regions using the entire posterior. Note that in order to compute the credible regions in such an manner, we must compute the HPD regions of the posterior. In general, this can be done by binning the data into a fine mesh. However, the size of this mesh will be exponential in the number of parameters used to define the parameter space. For even moderate sized problems, such as the WMAP data discussed in Section 4.2.1, creation of this mesh is impossible. While sparse representations can be used, these representations will generally also increase exponentially with the dimension. Thus, for many real-world problems, computing $1-\alpha$ credible regions directly from the posterior is impossible. Moreover, as noted in Section 3.6.1, credible regions derived from the entire posterior may also result in credible regions with less than expected coverage.

In addition to coverage probability, we can also compute the average size of the derived confidence/credible regions, as a function of α for the simulated observations, $\tilde{\mu}_i$. These confidence/credible region sizes are shown in Figure 4.18. As expected, based on the plots in Figure 4.16, χ^2 confidence regions are significantly larger than those produced by any of the other methods. Meanwhile, the sizes of the confidence regions produced by MES are similar to those produced by the grid-based Bayesian method using the full posterior, which was also seen in Figure 4.18, where the Bayesian regions (computed from the entire posterior) almost completely covered the MES confidence regions. Indeed, the MES confidence regions are typically similar to the credible regions derived from the posterior of a Bayesian analysis when the Bayesian approach uses a prior distribution which is similar to the likelihood of the parameter space given the observed data. Finally, note that credible regions based on the marginal distributions are typically larger than

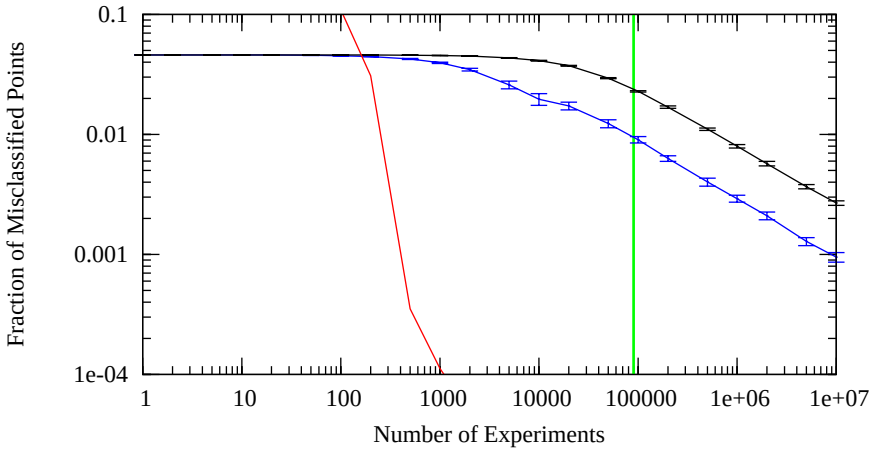
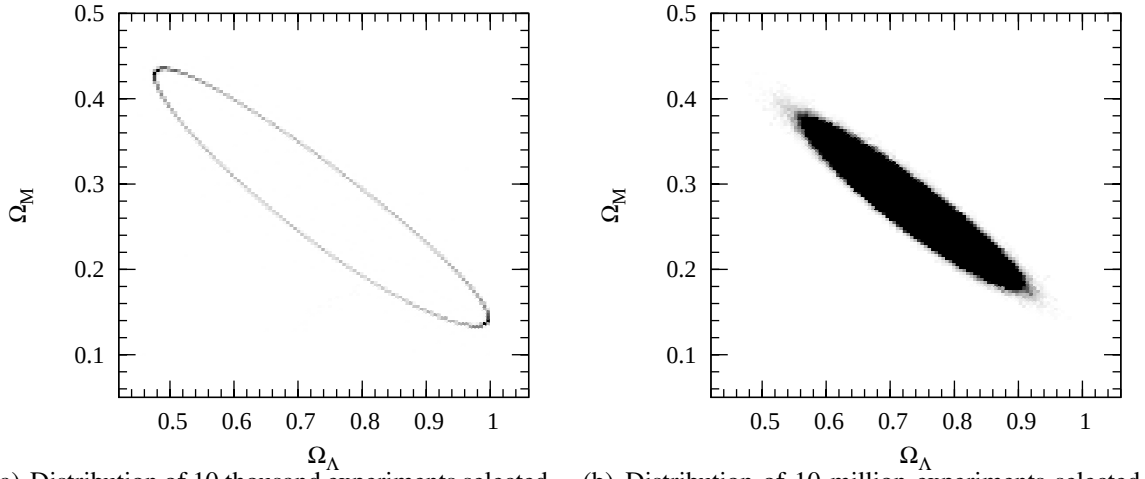


Figure 4.19: Convergence comparison between our active learning algorithm using on χ^2 tests (red) and MCMC where $b = 0.005$ (blue) and $b = 0.1$ (black). For each test, we compare the points included by the method in the 95% confidence/credible region to a baseline computed using a grid with 300 points per dimension. We plot the fraction of the points each method incorrectly classifies. The size of the grid used to compute the baseline is shown by the green line. MCMC is orders of magnitude more sample inefficient than both our straddle based method and the naive grid-based method.

those computed from the full posterior. Thus, computing credible regions directly from the posterior should be preferred over using marginal distributions, as the region size will be smaller and the coverage fraction will be closer to $1 - \alpha$.

Convergence Now, let us consider the convergence properties of MCMC and χ^2 tests using the active learning method described in Section 3.2.1; the convergence of the MES method described in Section 3.4.5 will be similar as both use the straddle heuristic to located the boundaries of the confidence regions.

For χ^2 tests, we can calculate convergence by first computing whether or not each point on an evenly spaced 300×300 baseline grid is included in the 95% confidence region; the ranges of the grid are equal to those of Figure 4.16(b). We then employ the our active learning algorithm to sample points. At specified intervals, we compare the 90,000 grid samples with the values predicted using our Gaussian Processes, and return the fraction of points incorrectly classified. That is, we return the fraction of points in which the true grid and the GP model disagree as to whether they should be included in the 95% confidence region.



(a) Distribution of 10 thousand experiments selected by our active learning algorithm using χ^2 tests.

(b) Distribution of 10 million experiments selected by MCMC.

Figure 4.20: Distribution of samples selected by our active learning algorithm using χ^2 tests (a) and MCMC (b). While our algorithm focuses on the 95% confidence region boundary, MCMC samples points in proportion to the posterior density. MCMC favors points that are likely over those which define the 95% credible region boundary. Hence, MCMC is significantly less data efficient than our algorithm (see Figure 4.19).

Convergence for MCMC is calculated in a similar way. First, we use the 300×300 evenly spaced grid to compute the baseline 95% credible regions (using numerical integration). We then run MCMC with different values of b, the variance of the proposal distribution. As with the active sampling method based on χ^2 tests, at specified intervals, we calculate the fraction of grid samples on which the baseline and MCMC 95% credible regions disagree.

In Figure 4.19, we show the the classification error as a function of number of experiments for our active learning algorithm (based on χ^2 tests) (red), along with MCMC where $b = 0.005$ (blue) and $b = 0.1$ (black). Choices of b were selected to present the ranges in convergence rates for MCMC. The number of points used to create the baseline grids is depicted as the green vertical line.

As the Figure shows, using our active learning algorithm is significantly more efficient that using either MCMC or the naive grid-based approach. In fact, our approach is roughly two order of magnitude more data efficient that the naive grid and more than four orders of magnitude more efficient than MCMC. Note that we tried many values of b for MCMC; the blue line in Figure 4.19 represents the optimal choice. As we discussed in Section 3.5.2,

the choice of b can have large impact on the convergence rate of the algorithm. However, it is difficult to choose b *a priori*. If a poor choice of b were selected, say $b = 0.1$, then the convergence rate of MCMC could be another order of magnitude worse than if the optimal value of b were selected.

Thus in practice, MCMC is not an efficient algorithm for computing $1 - \alpha$ credible regions. As discussed in Section 3.6.4 and seen in the SNLS data in Section 4.2.2, MCMC selects points in proportion to their posterior density. As a result, the points defining the 95% credible region boundary are sampled an order of magnitude less than those which define the peak of the posterior. In Figure 4.20, we illustrate the sampling patterns for our algorithm and MCMC. As expected, MCMC samples points through the space which have likelihood, while our algorithm focuses on the boundary, agreeing with the example of Section 3.6.4. Since MCMC is an algorithm to sample a posterior, not to find confidence regions, it can be orders of magnitude less efficient than both our active learning algorithm and the naive grid-based sampling. Moreover, the result of lack of convergence, is that the resulting $1 - \alpha$ credible intervals are underestimated. The majority of the classification error shown in Figure 4.19 is a result of the MCMC chain returning $1 - \alpha$ credible regions which are too small.

4.2.3 Combined CMB + SN + LSS Results

In the previous two statistical analyses, we examined only a single data source. For the third study, we examine three data sets together: the WMAP first year data [Verde et al., 2003], the supernova data from Davis et al. [2007], and LSS data based on luminous red galaxies [Tegmark et al., 2006]. By simultaneously computing the confidence region of the combined result, we are able to ensure that the resulting regions are statistically accurate.

While there are many approaches that could be used to compute these joint confidence regions, we use a frequentist approach based on p-values. Specifically, for each parameter vector in our parameter space, we compute the associated p-values for each of the three data sets and then combine the resultant p-values using Fisher's method [Fisher, 1948]. Fisher's method trades discrimination power in cases where the p-values are drastically different (e.g. one is 0 and another is 1), for those cases when the p-values are nearly equivalent. As the latter case is generally more common, Fisher's method for combining p-values typically results in tighter confidence regions than other methods (e.g. Bonferroni corrections [Bonferroni, 1936]).

Combining p-values using Fisher's method has two major advantages. First, since the data sets are assumed to be independent, the p-values for the three data sets can be

computed independently, possibly using different statistical tests for each data set as appropriate. Second, as we saw in Section 2.4, Fisher's method can be easily decomposed into a sum of observable functions (one for each data set). Moreover, each data set can be individually modeled allowing for increased efficiency.

Recall from Section 2.4, that Fisher's method rejects the hypothesis θ if

$$-2\left[\ln(p_{\text{cmb}}) + \ln(p_{\text{sn}}) + \ln(p_{\text{lss}})\right] \geq C,$$

where C is set to the value at which a $\chi^2_{(6)}$ cumulative distribution function equals $1-\alpha$, ensuring the test has correct $1-\alpha$ coverage.

Now let us look at how we can compute the p-values. For the WMAP data, we use the same confidence ball procedure we developed in Section 4.2.1. After computing the hypothetical model for qθ, we can solve Equation 3.4 for z_α and hence compute p-value of the hypothesis θ. For both the supernova and the LSS data sets, we use χ^2 test to compute the p-values. This is done by computing the variance-weighted sum of squares between the hypothetical model for θ and the observed data, and then comparing this statistic with the appropriate χ^2 distribution to compute α. For the LSS data, we will use the code of Tegmark et al. [2006] to compute the variance weighted sum of squares. While in Section 4.2.2 we showed that the MES procedure provides an tighter $1-\alpha$ confidence region, it is hard to compute a p-value from the MES procedure. In practice, if you want to compute two confidence regions of different levels for the same data set, the MES procedure will require that two convex games be solved. Thus, we restrict ourselves here to the confidence ball and χ^2 procedures.

We have now defined the three observable functions g_{cmb}, g_{sn}, and g_{lss}. Thus, we can use the var-maxvarstraddle heuristic of Section 2.4 to learn the threshold $t = -2C$ of g, where $g = g_{\text{cmb}} + g_{\text{sn}} + g_{\text{lss}}$. However, note that the running times of the model simulators for the different data sets are vastly asymmetric. In particular, CMBFast takes several minutes to run, while computing models for both the supernova and LSS data sets takes only a fraction of a second. Thus, we use a modified version of our algorithm with the var-maxvarstraddle heuristic.

In this modified algorithm, we determine the candidate point and observation function pair, $\{\tilde{\theta}, \tilde{g}\}$, which maximizes the var-maxvarstraddle heuristic. If $\tilde{g}$ is either g_{sn} or g_{lss}, we compute it, just as we did in Section 2.4. However, if $\tilde{g} = g_{\text{cmb}}$, we first compute the p-values for the supernova and LSS data sets, to see if the p-values from these two data sets alone can reject $\tilde{\theta}$. That is, we determine if $g_{\text{sn}}(\tilde{\theta}) + g_{\text{lss}}(\tilde{\theta}) \geq C$; if this is the case, we can reject $\tilde{\theta}$ without having to compute $g_{\text{cmb}}(\tilde{\theta})$, which is an enormous computational savings. If we determine that $\tilde{\theta}$ is rejected, then we place an approximate value for $g_{\text{cmb}}(\tilde{\theta})$ in the data set used to model the WMAP samples.

Since we compute $g_{\rm sn}$ and $g_{\rm lss}$ whenever we compute $g_{\rm cmb}$, but not visa versa, the data sets used to model the supernova and LSS functions generally will have more points, and hence lower var-maxvarstraddle scores. Thus, in practice, the var-maxvarstraddle heuristic always selects $\tilde{g} = g_{\rm cmb}$, and the algorithm is much like the seq-straddle heuristic with the ability to skip computations of $g_{\rm cmb}$ when warranted. While in Section 2.4.3 we showed that the var-maxvarstraddle heuristic is much more efficient than the seq-straddle heuristic when the models (g_i) have similar costs, some preliminary results indicate that when the models have divergent costs, the seq-straddle heuristic performs better than a cost-weighted var-maxvarstraddle heuristic. We leave this idea for future work.

As the computational cost of CMBFast is so high, we initialized the active algorithm with the CMBFast models computed in Section 4.2.1. These results included the cosmological parameters $\tau, \Omega_\Lambda, \Omega_{\rm M}, \omega_{\rm DM}, \omega_{\rm B}, f_\nu$ and n_s, which allowed us to also compute $g_{\rm sn}$ for each of the experiments. However, the experiments did not include galaxy bias, b. Thus, for each experiment from Section 4.2.1, we compute several values of $g_{\rm lss}$ for different values of b and retained the best and worst fits. While in principle, we could have retained all fits, doing so would have increased the size of our data set by two orders of magnitude and have prevented it from being sorted in RAM.

Thus, we seed the algorithm with 2.2 million p-values from the 1.2 million CMBFast models in Section 4.2.1, and their corresponding supernova and LSS p-values. Using the algorithm described above, we selected another $\sim 600,000$ models, including 300,000 CMBFast models. Results are shown in Figures 4.21 and 4.22. As with the figures from Section 4.2.1, regions of solid color indicate values for the parameter(s) on the axes for which some combination of the remaining parameters result in a model that is within the $1 - \alpha$ confidence region.

Comparing Figures 4.21 and 4.22, with Figures 4.6, and 4.7, it is apparent that combining the WMAP data with the supernova and LSS data provides strong constraints on all of the parameters. Note that the inclusion of additional data sets has completely eliminated the secondary peak that we observed in Figures 4.7. This peak corresponded to parts of parameter space that indicated values of the Hubble's constant far in excess of the commonly accepted values. As we saw in Figures 4.8 and, 4.9, restricting the values of Hubble's constant can dramatically reduce the size of the resulting confidence regions.

In Figure 4.23, we plot confidence regions in $\Omega_{\rm M}$ versus Ω_Λ space for all three data sets individually, along with a combined analysis for all three data sets. While the 95% confidence regions (represented as those regions of any color) for all three models individually are large, the resulting 95% confidence region of the combined model is quite small. In particular, the 95% confidence region of the combined model is smaller than what one would obtain by intersecting the two dimensional projections of the 95% confidence re-

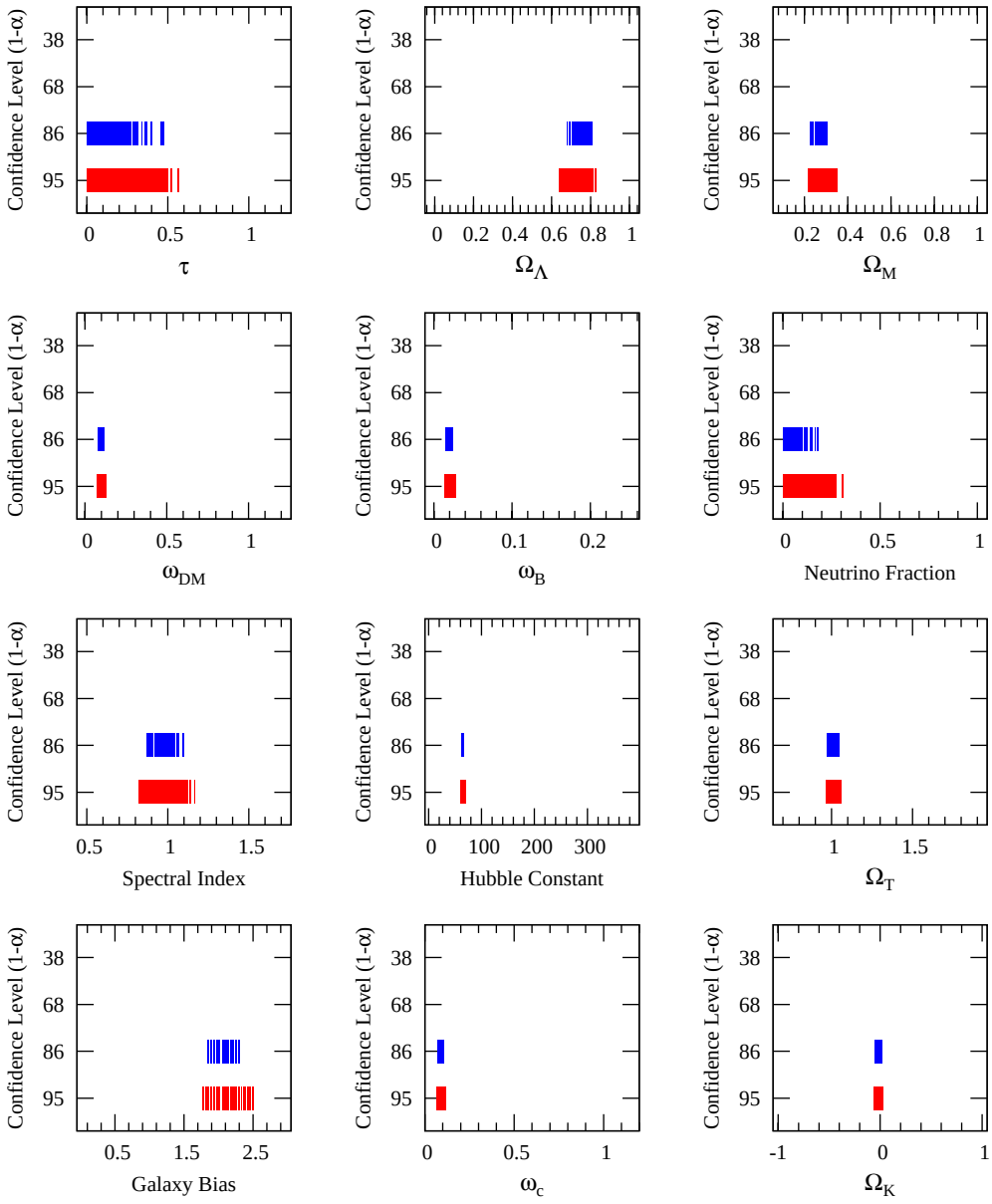


Figure 4.21: Jointly valid confidence intervals based on an combined analysis of WMAP, supernova, and LSS data for our cosmological parameters for four values of $1-\alpha$, corresponding to $\frac{1}{2}\sigma, \sigma, 1\frac{1}{2}\sigma$ and 2σ confidence levels, respectively. Areas of solid color indicate values for the given parameter that contain the true value of cosmological parameter with probability $1-\alpha$, regardless of the values of the remaining 7 parameters.

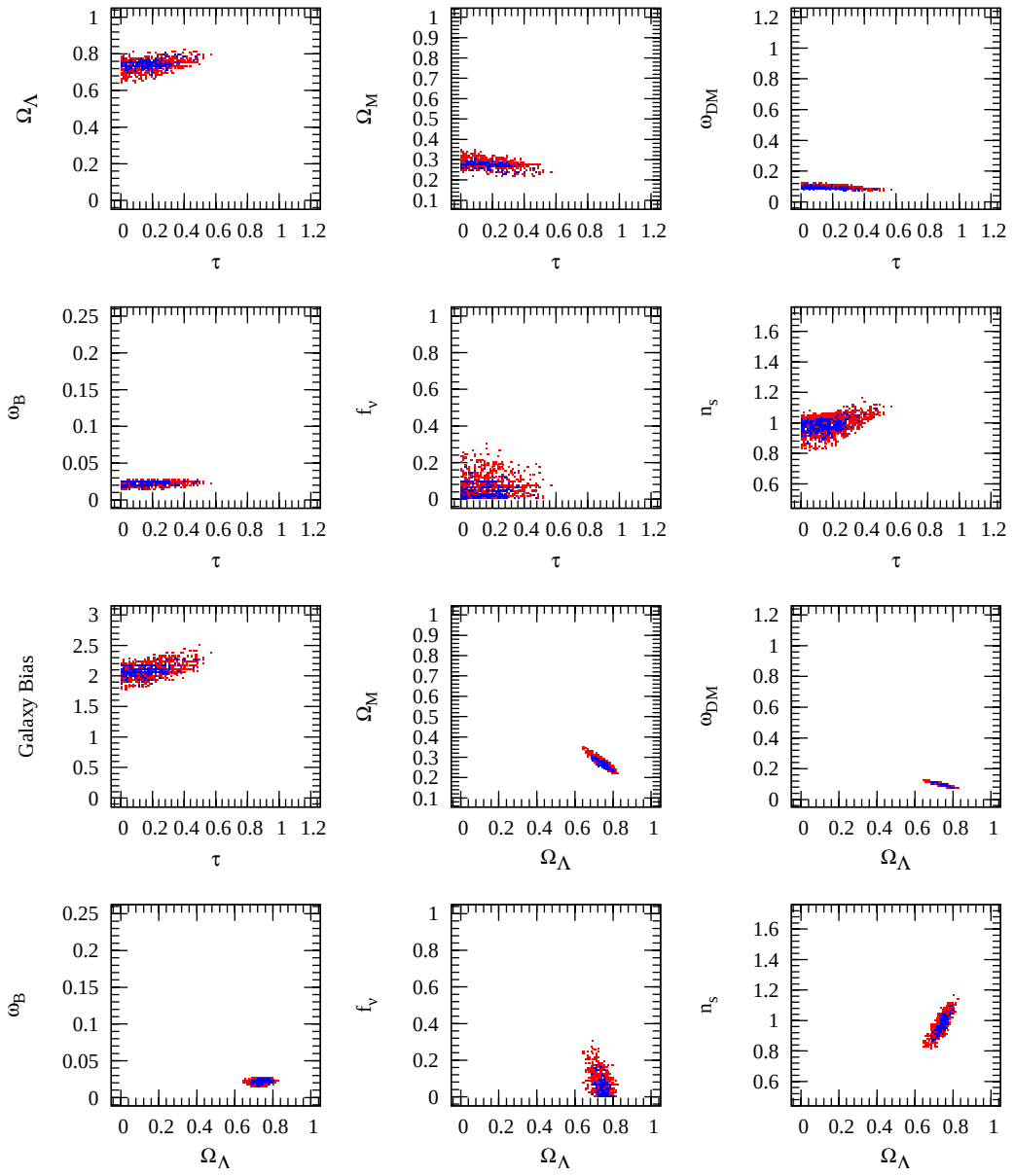
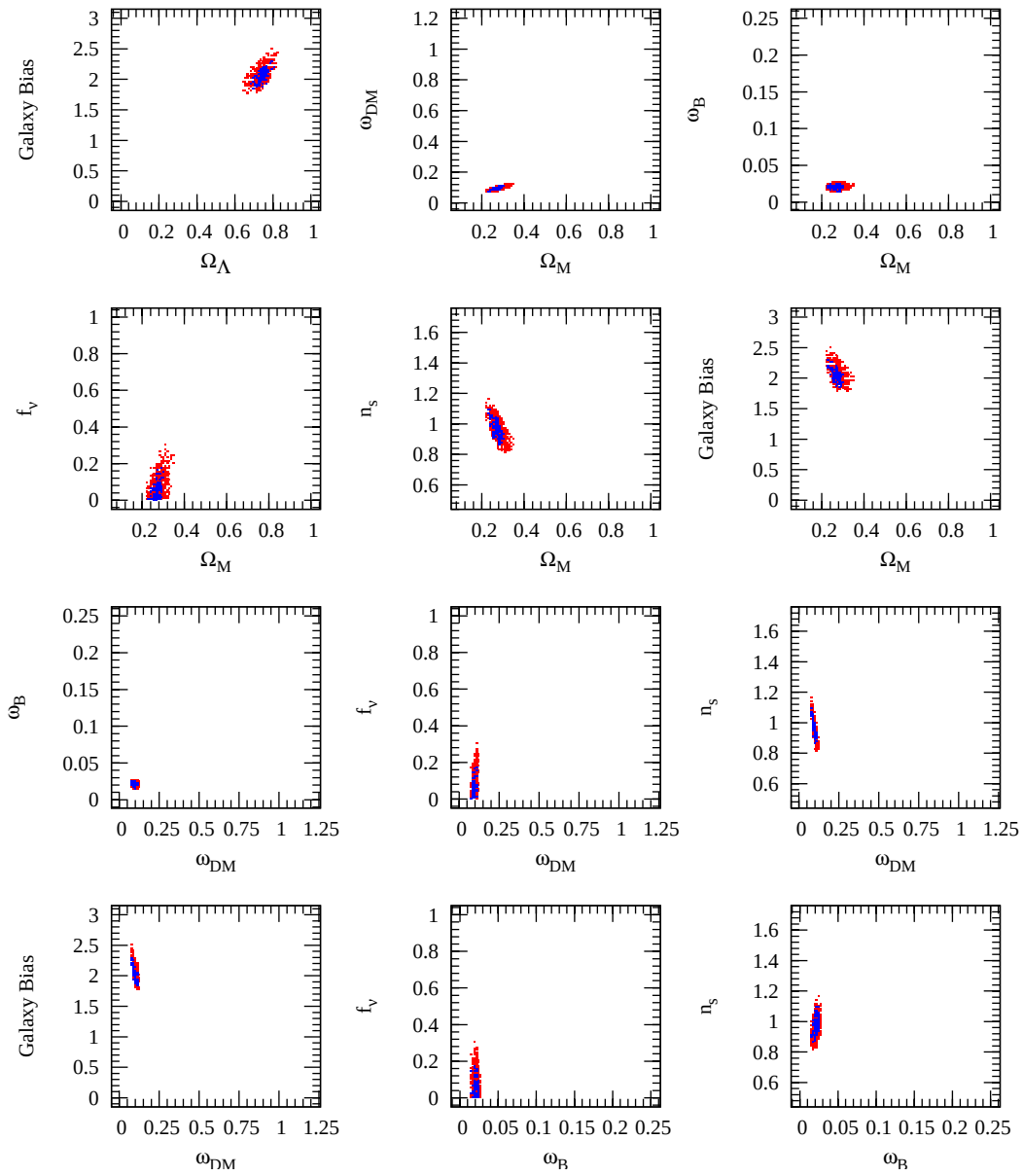
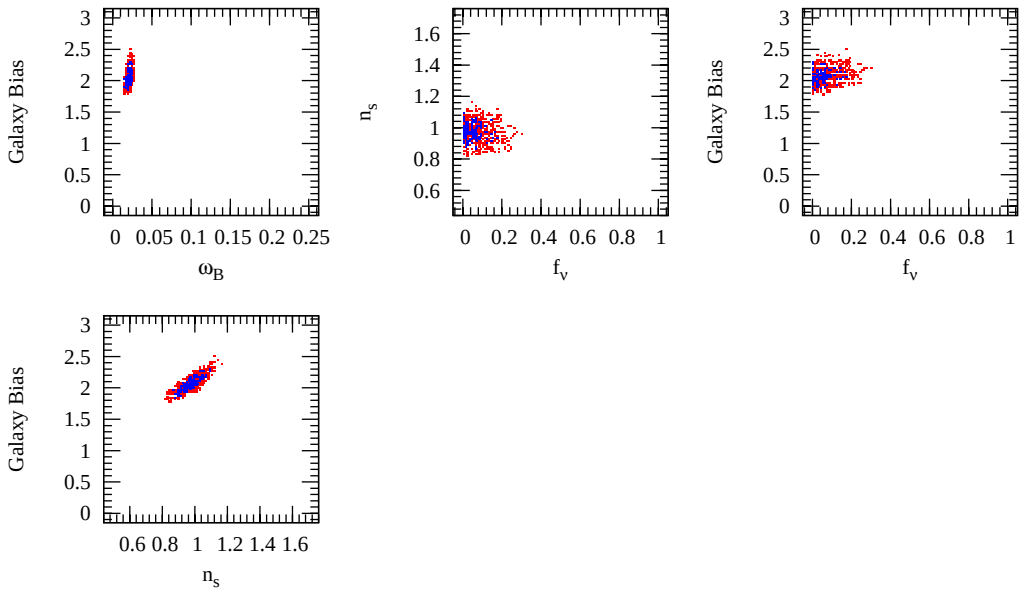


Figure 4.22: Jointly valid confidence regions for pairs of cosmological parameters, where the colors cyan, magenta, blue and red correspond to $\frac{1}{2}\sigma, \sigma, 1\frac{1}{2}\sigma$ and 2σ, confidence levels respectively. Areas of solid color indicate values for the given pair of fixed (plotted) parameters that contain the true value of cosmological parameter with probability $1 - \alpha$, regardless of the values of the remaining 6 parameters.



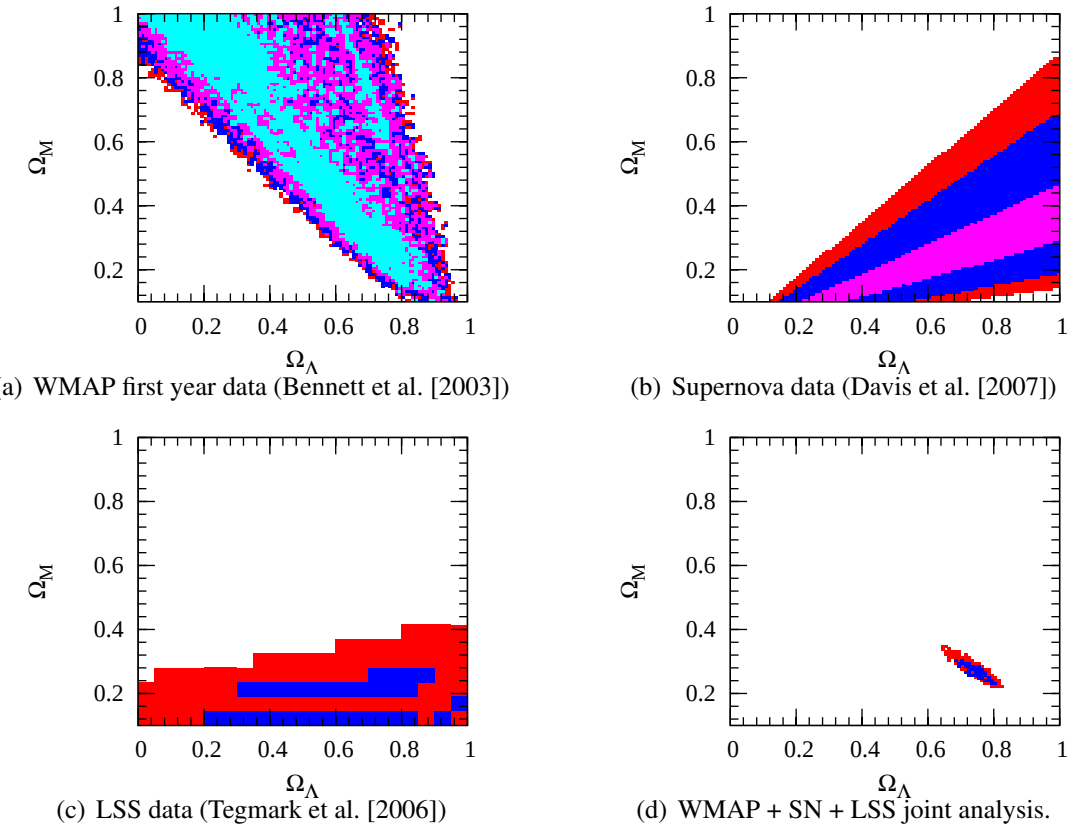
Continuation of Figure 4.22.



Continuation of Figure 4.22.

gions for the three models individually. This indicates that the confidence surfaces are complex surfaces embedded in their associated parameter spaces. By maintaining these surfaces, we are able to gain better insights as to the interdependencies of the parameters. Moreover, we can exploit this structure to obtain tighter confidence regions than could be obtained from marginalizations of the data.

Finally, in Table 4.5, we present the 95% confidence intervals derived for each of the cosmological parameters from Table 4.1 for the WMAP first year data, the Davis et al. [2007] supernova data set, and the Tegmark et al. [2006] LSS data set both individually and combined together. The combination of the three data sets results in constraints that are much tighter than those provided by any of the data sets individually. In particular, note that the constraint on Ω_Λ for all three data sets is quite weak; for all three data sets, Ω_Λ is nearly unconstrained. However, in the joint analysis, Ω_Λ is constrained between 0.64 and 0.83. Moreover, the parameters with the weakest constraints in Table 4.5 (see also Figure 4.21) such as τ and n_s are typically those which are model inputs for only one of the three data sets. Thus, a joint analysis of many independent data sets which share the same model parameters can result in tighter constraints on the $1-\alpha$ confidence region, and provide results which are much much more enlightening scientifically.



(a) WMAP first year data (Bennett et al. [2003])

(b) Supernova data (Davis et al. [2007])

(c) LSS data (Tegmark et al. [2006])

(d) WMAP + SN + LSS joint analysis.

Figure 4.23: Comparison of the confidence regions derived for WMAP (a), supernova (b), and LSS (c) data sets with those derived using all three data sets together (d). Regions of solid color indicate values for Ω_M and Ω_Λ for which some combination of the remaining parameters results in a model with probability greater than $1 - \alpha$. The WMAP and LSS models are 7 parameter models, while the supernova is a 3 parameter model, and the combination model is an 8 parameter model.

Comparison to Previous Work

Unfortunately, as far as we know, there have been no cosmological joint analyses involving the same three data we used in Section 4.2.3. However, most cosmological works include joint analyses with some subset of the available data sets (c.f. Tegmark et al. [2001], Verde et al. [2003], Spergel et al. [2007]). Here we compare our results with Spergel et al. [2007].

Spergel et al. [2007] computes joint (Bayesian) analyses for the WMAP third year data with the SNLS data set [Astier et al., 2006], as well as a joint analysis of the WMAP third

Parameter	WMAP	Supernova	LSS	Combined
τ	0.0 – 1.2	——	——	0.0 - 0.5, 0.52 - 0.53, 0.56-0.57
Ω_Λ	0.0 – 0.94	0.12 – 1.0	0.0 – 1.0	0.64 – 0.83
$\Omega_{\rm M}$	0.0 – 1.0	0.1 – 1.0	0.1 – 0.42	0.217 – 0.352
$\omega_{\rm DM}$	0.0 - 0.36, 0.62 - 0.70	——	0.07 – 1.2	0.069 – 0.129
$100\omega_{\rm B}$	0.5 - 6.2, 11.5 - 12.7	——	0.1 – 25	1.34 – 2.84
f_ν	0.0 – 1.0	——	0.0 – 0.95	0.0-0.28, 0.30-0.31
n_s	0.73 – 1.7	——	0.5 – 1.22	0.81 – 1.17
b	——	——	0.75 – 3.0	1.77 – 2.49
$\Omega_{\rm k}$	-0.7 – 0.2	-1.0 – 0.78	-0.8 – 0.93	-0.06 – 0.035
$\Omega_{\rm T}$	-0.8 – 1.7	0.22 – 2.0	0.07 – 1.8	0.965 – 1.06
$\omega_{\rm c}$	*0.0 – 0.70*	——	0.0 – 1.2	0.066 – 0.12
$\omega_{\rm N}$	*0.0 – 0.70*	——	*0.0 – 1.14*	*0.0 – 0.063*
H_0	17 - 135,243 - 272	10.5 – 380	*41.1 – 380*	60 – 70

Table 4.5: 95% confidence intervals for the cosmological parameters using the WMAP first year, the Davis et al. [2007] supernova, and the Tegmark et al. [2006] LSS data individually, and then combined. Intervals in italics were estimated from the ranges computed for the other parameters. The combination of the three data sets results in constraints that are much tighter than those provided by any of the data sets individually.

year data with the LSS data from Eisenstein et al. [2005]. For comparison, we combine the confidence ball results on the WMAP first year data from Section 4.2.1 with results from the SNLS and (subsequently) the LSS data of Tegmark et al. [2006] using χ^2 tests. While the data sets are not exactly identical, they are fairly similar. For instance, the mean values of the WMAP data do not drastically change between the first and third year compilations. While the third year data has significantly less noise (as observed in Figure 4.1), only n_s and H_0 become significantly more constrained [Spergel et al., 2007].

In Table 4.6, we present results between our analysis (using frequentist methods) and the analysis of Spergel et al. [2007] (using Bayesian techniques). When combining the WMAP and LSS data sets, we observe that our results are quite comparable with those of Spergel et al. [2007]. For several of the parameter, the derived confidence intervals are nearly identical. The remaining parameters are slightly better constrained by the Spergel et al. [2007] analysis.

Analyses using the combination of the WMAP and SNLS data, however, show larger discrepancies. For most of the parameters, our analyses returns confidence intervals which are significantly larger than those reported by Spergel et al. [2007]. This difference may be attributed to our use of the WMAP first year data, (as opposed to the WMAP third year data), as well as the use of (surprisingly narrow) Gaussian priors on many of the parame-

	WMAP + SNLS		WMAP + LSS	
Parameter	this work	Spergel et al. [2007]	this work	Spergel et al. [2007]
τ	0.000 – 1.056	0.055 – 0.115	0.096 - 0.144, 0.288-0.3	0.050 – 0.109
Ω_Λ	0.200 – 0.930	——	0.72 – 0.78	——
Ω_M	0.100 – 0.964	0.223 – 0.272	0.235 – 0.271	0.235 – 0.295
ω_dm	0.034 – 0.296	*0.100 – 0.114*	0.093 – 0.117	*0.103 – 0.117*
$100\omega_\mathrm{b}$	1.096 – 6.076	2.160 – 2.309	1.843 – 2.341	2.160 – 2.301
f_ν	0.000 – 1.000	——	0.05 – 0.140	——
n_s	0.788 – 1.7	0.933 – 0.966	0.920 – 1.016	0.933 – 0.964
b	——	——	1.920 – 2.100	——
H_0	54.8 – 77.0	70.1 – 74.7	69.6 – 73.3	68.4 – 73.6
σ_8	——	0.717 – 0.799	——	0.731 – 0.812

Table 4.6: Comparison of 68% confidence/credible intervals between this work and Spergel et al. [2007] for nine cosmological parameters. The first two columns give a joint analysis using WMAP and SNLS data, while the second two columns use a joint analysis of WMAP and LSS data to compute confidence/credible intervals. Intervals in italics were estimated from the ranges computed for the other parameters.

ters. Spergel et al. [2007] notes that the effect of the prior in some cases is large, especially using lensing data. For this reason, Spergel et al. [2007] cautions against interpreting these joint credible regions as being the true parameter ranges. Conversely, our results make no use of a prior distribution over each parameter and are independent of the parameter ranges searched. Moreover, our frequentist techniques guarantee $1 - \alpha$ coverage for the intervals given in Table 4.6.

4.3 Summary

In this chapter, we have discussed several data sets which can be used to determine the values of fundamental constants from cosmology. These parameters have physical meanings and their values describe the age, composition, and eventual fate of our Universe. Moreover, by determining the values of these parameters, we can better model how matter in Universe collapsed to form stars, galaxies and large scale structures.

Using the active learning methods from Chapter 2 combined with the statistical inference techniques from Chapter 3 we have computed confidence regions for these data sets, individually and together as an ensemble. We find that our frequentist-based techniques allow us to easily perform both joint analyses and investigate the interactions of the cos-

mological parameters by placing constraints upon any number of the parameters. As we saw in Section 4.2.1, we can restrict the ranges of any number of parameters and then recompute the associated $1 - \alpha$ confidence regions without performing additional model experiments (e.g. CMBFast models). Thus, we can efficiently determine the interactions among the parameters.

Moreover, we have seen that because we maintain a representation of the entire $1 - \alpha$ confidence surface in the embedding parameter space, analyses using multiple data sets may lead to $1 - \alpha$ confidence regions which are substantially smaller than those obtained for any of the data sets individually. In particular, we note that combining the confidence intervals derived from a one or two dimensional marginalization of the data will result in confidence regions which can be much larger than those obtained by a joint analysis.

Comparing our methods with Bayesian methods in Section 4.2.2, we saw that observationally our methods have correct coverage (in a frequency sense) while Bayesian methods do not. In fact frequentist methods are guaranteed theoretically to have correct coverage. Bayesian methods do not have theoretically coverage guarantees. In particular, the results from Section 4.2.2 indicated that the coverage of Bayesian methods can be substantially lower than $1 - \alpha$. We note that the supernova inference task in Section 4.2.2 was only a two dimensional problem. In higher dimensions, Bayesian methods can result in credible intervals which trap the true value of the parameters close to zero percent of the time.

However, the coverage guarantee does not need to come at a cost to region size. In Section 4.2.2, we saw that 95% confidence regions produced by the MES confidence procedure were similar in size to those derived by Bayesian inference. In general, the confidence regions produced by the MES procedure are similar to the credible regions of a Bayesian analysis when the Bayesian approach uses a prior distribution which is similar to the likelihood of the parameter space given the observed data.

Finally, we have seen that our active learning method is much more data efficient that either grid based approaches or MCMC. While adaptive grids can be formed over the parameter space, there is always the risk of creating the grid cells such that an unknown interesting peak is missed (such as the secondary peak of the WMAP data — see Section 4.2.1). MCMC, on the other hand, can be orders of magnitude more data inefficient than even simplistic grid based approaches. As we saw in Section 4.2.2, even on the Davis et al. [2007] supernova data set with a smooth likelihood and only one peak, MCMC was over four orders of magnitude less efficient than our active learning algorithms. We expect this discrepancy only to increase as we consider higher dimensional tasks and those with multiple peaks in the likelihood surface (such as the WMAP first year data). However, testing converge of high-dimensional problems is problematic, as the base line is difficult to obtain.

Thus, we believe that our active learning algorithms coupled with frequentist statistical techniques provides a framework for learning $1 - \alpha$ confidence regions which is more computationally and data efficient that MCMC. Our techniques are able to derive inferences which are just as tight, while still ensuring correct coverage. Nevertheless, while the algorithms we used in this chapter have many desirable properties, they are first and foremost search algorithms. In particular, they are very good at locating parameter vectors which are on the boundary of the $1 - \alpha$ confidence regions. However, they cannot guarantee that there are no additional regions in the parameter space which have yet to be found. Hence they cannot guarantee that they have converged to the correct solution. In the next chapter we discuss an algorithm to prove convergence.

Chapter 5

Convergence Algorithms

In Chapters 2 through 4, we have developed and demonstrated active learning algorithms for efficiently learning $1 - \alpha$ confidence regions using a variety of frequentist statistical inference techniques. However, all of the algorithms that we have discussed have been search algorithms. That is, these algorithms are concerned with *finding* $1 - \alpha$ confidence region boundaries, not *proving that no other confidence regions exist* within the given parameter space.

For instance, using the results of our search algorithm, we can create figures such as Figure 4.23. The data supporting these figures indicate that there exists at least one 95% confidence region centered at $\Omega_\Lambda = 0.75$ and $\Omega_\mathrm{M} = 0.25$. However, the data does not necessarily rule out another peak or peaks at different values of Ω_Λ or Ω_M. While the presence of additional peaks is unlikely, due to the exploitative nature of our algorithms, it cannot be ruled out.

In this chapter we consider the problem of convergence. We will see that proving pointwise convergence is computationally infeasible. Instead, we define convergence in terms of the largest region our algorithms could have missed, and then seek to minimize this quantity. We present two algorithms which can be used either after the fact to compute, or with our learning framework to actively minimize, the maximum size of the largest unsampled region in parameter space.

5.1 Proving Convergence

To prove that a point in Figure 4.23(d) is within the $1-\alpha$ confidence region, we need to find one setting of the remaining six parameters ($\tau, \omega_{\mathrm{DM}}, \omega_{\mathrm{B}}, f_\nu, n_s$, and b) for which the corresponding model is accepted by the level α statistical test (e.g. χ^2 tests, confidence ball procedure, or MES). However, to prove that a point in Figure 4.23(d) is not within the $1-\alpha$ confidence region, we must show that no matter what values are selected for the remaining six cosmological parameters, no model is accepted by our level α statistical test.

Proving this for every point in a plot such as Figure 4.23(d) is infeasible, as there are an infinite number of combinations of Ω_Λ and Ω_{M}. Even binning the parameter space and then trying to prove that the center of each bin is either accepted or rejected is computationally infeasible for a reasonable number of bins per dimension. In fact, the problem of proving convergence in this manner is nearly equivalent to computing the $1-\alpha$ Bayesian credible regions directly from the d dimensional posterior. In both cases, the grids require 20 to 100 points per dimension to be scientifically useful. However, the size of the gridded region increases exponentially with dimension, making griding infeasible for even moderate dimensional problems like computing confidence regions for the WMAP data (see Section 4.1.1).

We have looked at using tree based structures, such as red-black trees and variable spatial trees [Kubica et al., 2005]. However, data distributions can be formed which force the trees to require exponential (in dimension) storage size. More importantly, even with our moderate dimension confidence region tasks, we were not able to either store the trees, or use the trees to compute the covered pixels in an efficient manner. Attempts to use either tree or grid based methods required days to prove that a single combination of Ω_Λ and Ω_{M} was not covered.

Instead of trying to prove that no other peaks in the confidence surface exist on a pixel by pixel basis, we consider minimizing the maximum size of an unobserved peak. Naively, the maximum size of an unobserved peak could be minimized by selecting points uniformly throughout the space, either using a grid or the variance heuristic (from Section 2.2), as this would assure that the samples were nearly equidistant from each other. However, as we saw in Section 2.2, sampling based on the variance heuristic is much less efficient than the straddle heuristic. This is because the straddle heuristic uses information about the values of the experiments, as well as their locations to pick subsequent samples, while variance and grid based techniques completely ignore the results of the experiments after they are performed.

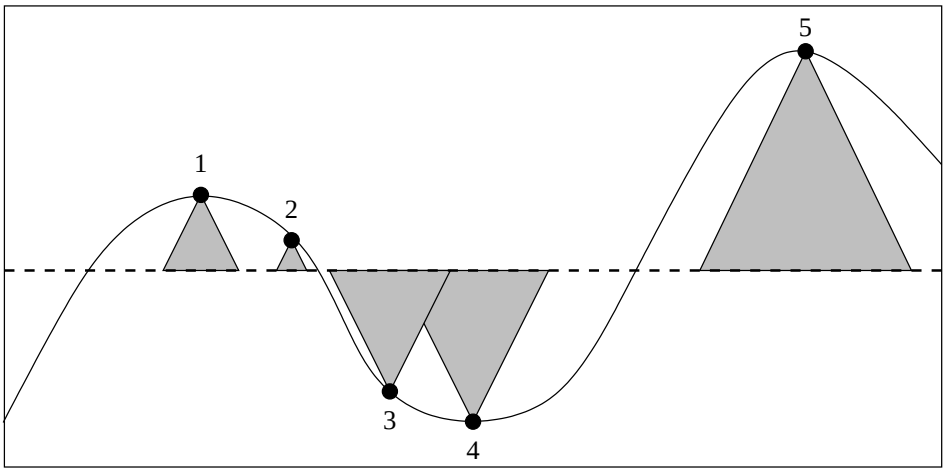


Figure 5.1: A pictorial example of using the derivative information to compute influence regions (gray) for a one dimensional target function (solid line). Currently sampled point are denoted by the black dots, while the dashed line indicates the threshold defining the level-set we seek. Using the maximum derivative of the target function and the values of the sampled experiments we can provable exclude regions of the parameter space, reducing the space that needs to be search for level-set members.

However, the value of the target function at specific locations alone does not give us enough additional information to select samples which minimize the maximum unobserved confidence region size better than variance or grid based techniques. Instead, suppose that we knew the maximum derivatives of the target function for each dimension. Then, we could compute regions in which the sampled point could prove that target function was either above or below the level-set boundary of interest.

For example, consider the one dimensional target function shown as a solid line in Figure 5.1. The dashed line indicates the level-set of the target function in which we are interested, while the five points indicate samples that have already been chosen, along with their values. We typically assume that the target function is almost deterministic, so the observed values appear nearly on the target function. The largest interval without samples is between samples 4 and 5. This distance corresponds to the maximal size of any missed confidence region in the parameter space when no assumptions are made about the target function's derivatives. However, if we do have information about the target function's derivatives, then we can construct "influence" regions around each point which specify locations where the target function cannot intersect our level-set boundary. Using these influence regions, the largest unknown region is now between the left-hand boundary and

sample point 1, and is roughly half the size of the maximal confidence region size when no derivative information is used.

Thus, the combination of both experimental values along with derivative information about the target function can significantly reduce the maximal size of any unobserved peak of the target function in our parameter space. Intuitively, to minimize the maximum size of a hypothetical unobserved peak, we want to sample points such that no point in the parameter space is far from the influence region resulting experimental points. Hence, we desire sample points that result in large influence regions which do not overlap the influence regions of other experiments, as these sample points with will have influence regions which "fill" the parameter space.

More formally, let $\mathcal{H}$ be a set of h experiments from parameter space Θ. Then we are interested in selecting the h experiments which minimize

$$\max_{\theta \in \Theta} \min_{\tilde{\theta} \in \mathcal{H}} \mathcal{D}(\theta, \tilde{\theta}) \tag{5.1}$$

where $\mathcal{D}(\cdot, \cdot)$ computes the minimal distance between θ and the influence region of $\tilde{\theta}$. Since the parameters composing the parameter space may not be independent, the shape of the influence regions around an experiment are hyperdiamonds with their vertices on the dimensional axes. Without loss of generality, we will assume that the space is scaled so that the derivatives of the target function are unity for all dimensions. Thus, the hyperdiamond becomes a hypercube.

Moreover, we assume that the influence region of $\tilde{\theta}$ can be modeled as a hypersphere, where the radius of the hypersphere is selected to maximally inscribe the original hyperdiamond; the radius of such a hypersphere is $\sqrt{2}/2$. This assumption leads us to underestimate the size of the influence region for a point, resulting in a conservative estimate of the largest possible missed peak. Specifically, the ratio of the volume of the inscribed hypersphere to the volume of the enclosing hyperdiamond as a function of dimension is given by

$$r = \frac{\text{Volume}_{\text{hypersphere}}}{\text{Volume}_{\text{hyperdiamond}}} = \frac{(2\pi)^{d/2}}{\Gamma(d/2+1)}$$

where $\Gamma(\cdot)$ is the gamma function. As d goes to infinity, the ratio of r goes to zero. However, for small d, $r \sim 1$. Moreover, we can practically remove this assumption later. Thus, assuming a Euclidean distance measure, Equation 5.1, can be written as

$$\theta' = \max_{\theta \in \Theta} \min_{\tilde{\theta} \in \mathcal{H}} \|\theta - \tilde{\theta}\|_2 - \mathcal{R}(\tilde{\theta}) \tag{5.2}$$

where $\|\cdot\|_2$ denotes the L_2 norm, and $\mathcal{R}(\tilde{\theta})$ is the radius of the influence region for $\tilde{\theta}$. Intuitively, this computation is similar to the minimax process used in the MES confidence

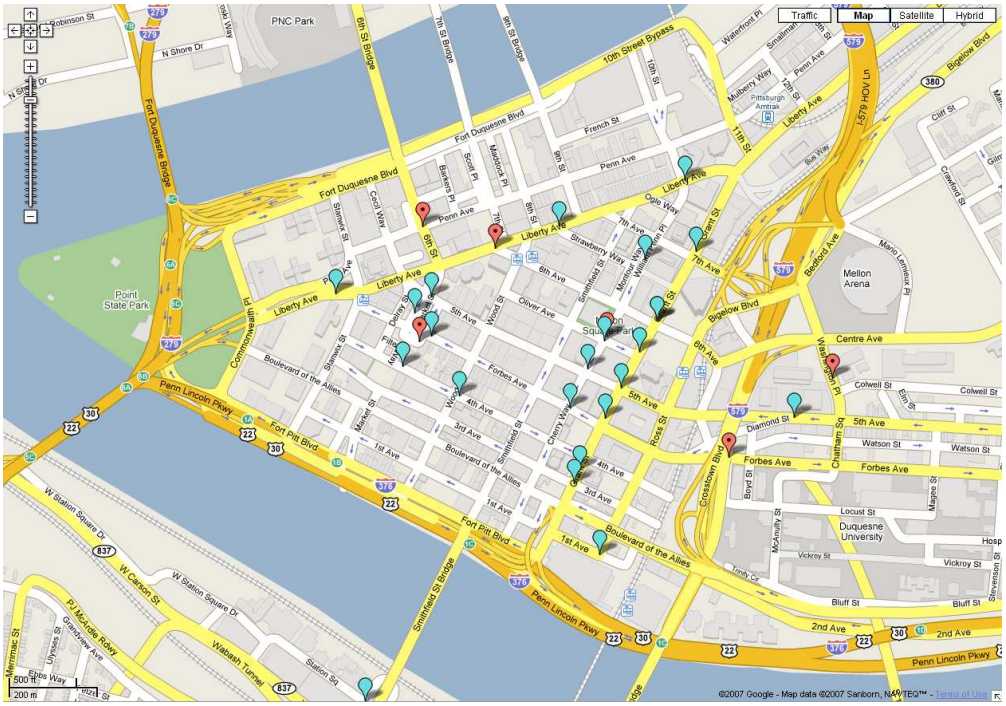


Figure 5.2: Coffee shops located in downtown Pittsburgh. Starbucks locations are denoted with red pins (with dots), while all other shops are denoted with cyan pins. Each shop is located at the bottom end of the pin — where it "touches" the map.

procedure. However, here the parameter space is not convex. To see this note that there can be multiple local maxima in the space, corresponding to the locations which are equidistant from a set of $d+1$ experimental points. Thus, we cannot solve the above minimax problem with either linear or quadratic programming. Instead we consider an approach using Voronoi diagrams.

5.2 Voronoi Diagrams

In two dimensions, Voronoi diagrams are sets of $d-1$ dimensional faces which decompose a d dimensional metric space into disjoint regions. In order to more easily describe the terminology and gain some intuition, let us look a simple two dimensional example. In Figure 5.2, we plot all of the coffee shops in downtown Pittsburgh as of October 2007 (as listed by Google and Yahoo!). As with most metropolitan cities, there are a plethora

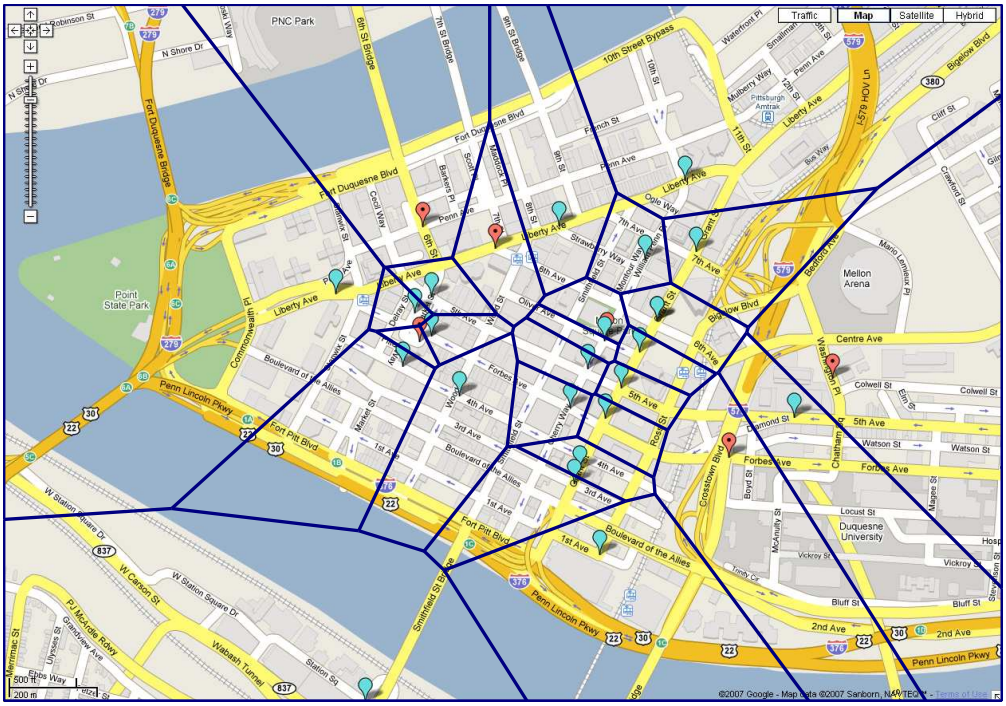


Figure 5.3: Equidistant boundaries between coffee shops located in downtown Pittsburgh based on Euclidean measure (ignoring buildings, etc). Blue lines indicate the lines which bisect the nearest two coffee shops. Region bounded by blue lines indicate the customer base for the coffee shop in that region, assuming customers go to the nearest shop. Each coffee shop has its own influence region.

of Starbucks coffee houses (red pins with dots), as well as some non-Starbucks coffee locations (cyan pins).

Suppose we are in downtown Pittsburgh with the Figure 5.2 map, and — coffee being coffee — we decide to head for the nearest coffee house, regardless of the store's corporate affiliation. Here we define distance to be Euclidean distance, and do not worry with such trivialities as rivers or buildings. Depending on where we were, we would head for a different store location. Suppose we were to divide the downtown area into a set of regions which describe the coffee house we would head to given our starting location.

The resulting Voronoi diagram is given in Figure 5.3. The dark blue lines indicate parts of the downtown that are equidistant to two adjacent coffee houses. In two dimensions, we will refer to the blue lines as "edges", the regions which they enclose as "faces", and the point where edges meet as "vertices". In higher dimensions, we will refer to all objects as

d-faces. Hence, a vertex will be a 0-face, and an edge will be a 1-face.

In two dimensions, faces correspond to the area from which a coffee house draws its customers (assuming that everyone goes to the nearest coffee shop). Thus, each face has one coffee house, which will generally be call a "site". Faces are composed of all those points which are nearer to the site of that face than the site of another face. Hence, the Voronoi diagram can be thought of as the classification boundary of an algorithm which uses only the single nearest neighbor to classify test points.

Meanwhile, vertices denote points which are equal distance to three (or more) coffee shops. Vertices are locations which are maximal distant from any of the surrounding coffee houses. Thus, if you were to open a new coffee shop, choosing a vertex would maximize your customer base (assuming a uniform distribution of people over the city). However, not all vertices are the same distance from their corresponding coffee houses, thus you would want to check all possible vertices and select the one which was the farthest from its associated sites; by construction, all of the associated sites will have the same distance to the vertex.

Now suppose that you are a Starbucks aficionado. As such, you are willing to walk an extra block to get your Eggnog Latte[1], over taking the plain coffee from the local stand. As a result of your preference of Starbucks over other coffee locales, the decision boundary as to which coffee house to visit will have changed to that shown in Figure 5.4. As in Figure 5.3, blue lines indicate the demarcations between coffee houses. The red circles are centered on the locations of the Starbucks, and their radii indicate the extra distance you are willing to travel to visit them. Thus, you will never visit any of the four coffee houses that are within one of these red circles, as the nearest Starbucks is less than a block away.

The preference for the Starbucks sites results in a drastic change to the Voronoi diagram. In addition to the four coffee houses that will never get your business, the fraction of the city from which you would visit Starbucks has increased dramatically. Moreover, the edges between sites are no longer necessarily straight. When you have equal preference to visit two sites, θ_1 and θ_2, the edge between them will simply be their a straight line: $\{\theta \in \Theta : \|\theta - \theta_1\|_2 = \|\theta - \theta_2\|_2\}$. However when one of the sites has a larger preference, the edge between the two sites becomes curved to reflect the fact that the site with a larger radius is closer to more points in $\mathbb{R}^2$. This curved edge is given by

$$\{\theta \in \Theta : \|\theta - \theta_1\|_2 - \mathcal{R}(\theta_1) = \|\theta - \theta_2\|_2 - \mathcal{R}(\theta_2)\},$$

where $\mathcal{R}(\theta_i)$ gives the radius of site θ_i. Since sites with larger radii have more influence, the edges between them and other sites bend away from them. These resulting generalized

[1]Eggnog lattes are apparently one of the hardest Starbucks drinks to make.

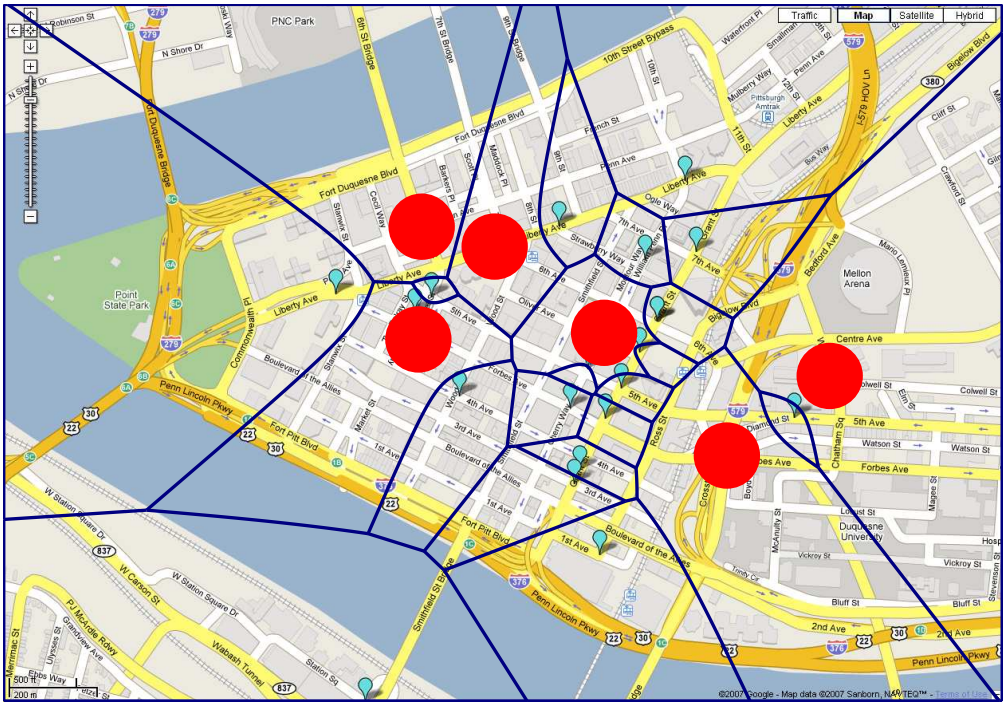


Figure 5.4: Equidistant boundaries between coffee shops located in downtown Pittsburgh based on Euclidean measure (ignoring buildings, etc), where Starbucks locations (red) are preferred over other locations. Red circles indicate the extra influence that Starbucks has over other vendors; vendors within these circles will receive no business, as customers are willing to walk the extra block to the nearest Starbucks. Thus, there are fewer "coffee shop regions" here than in Figure 5.3. When coffee shops have different attractive influences, the equidistant boundaries are no longer straight lines.

Voronoi diagrams with spherical sites are known as "Apollonius diagrams" (c.f. Okabe et al. [2000], Jeyakumar and Rubinov [2005], Boissonnat and Teillaud [2006]). In Figure 5.4, the sites have only one of two distinct radii. However in general, each site can have its own unique radius.

Now, reconsider the task of computing the size of the maximum unobserved peak discussed in Section 5.1. Each sampled experiment corresponds to a site in an Apollonius diagram. The radius of the site θ_i will be given by $|g(\theta_i) - t|$, where $g(\theta_i)$ is the evaluation target function at θ_i and t is the level-set boundary of interest. Here we retain the assumption that the parameter space is scaled so that the maximum derivative of g is unity in all dimensions. As with the coffee house example, the points in parameter space which

are furthest from their adjacent sites are the vertices of the corresponding Apollonius diagram. Thus, to find the maximum possible size of an unobserved peak, we need only iterate through the vertices of the Apollonius diagram and select the vertex which is the farthest from any site. Let us now look at this idea in more detail.

5.2.1 Related Work

Several algorithms exist to efficiently construct (standard) Voronoi diagrams and their duals, Delaunay triangulations [Delaunay, 1934]. As we saw earlier, the edges of Voronoi diagrams are the classification boundaries of a single nearest neighbor algorithm. Moreover, Voronoi diagrams can be used in physics and biology to model situations such as particle interactions in gasses and foams [Zanin et al., 2002, Li et al., 2006], and understanding the living arrangements of communities of organisms [Grant, 1968, Wiens, 1969, Barlow, 1974, Byers, 1992].

A Delaunay triangulation is a triangulation of a set of points in which no circumcircle of any of the triangles contains one of the points. That is, all of the points used to construct the triangulation lie on the circumference of the circumcircles of the triangulation. Additionally, they maximize the minimum angle over all angles of all triangles in the triangulation, virtually eliminating long, thin triangles. Delaunay triangulations are used in fields such as graphics and physics to construct meshes [Rebay, 1993, Beyer et al., 2005, Zhao et al., 2005].

Given their importance to the research community, there are many efficient algorithms to compute both Voronoi diagrams and Delaunay triangulations Shamos and Hoey [1975], Fortune [1986], Inagaki et al. [1992], Sugihara and Iri [1994]. For instance, Fortune [1986] gives an algorithm which, given a set of sites, "sweep"s over these sites from left to right, to construct a Voronoi diagram in $\mathcal{O}\left(n \lg(n)\right)$ time.

Similarly generalized Voronoi diagrams, including Apollonius diagrams, are found in many places in the literature, such as robot planning [Mahkovic, 1999, Kalra et al., 2006], representing sounds and music [McLean et al., 2007], and crystal growth [Kobayashi and Sugihara, 2002] [2] Algorithms for computing generalized Voronoi diagrams include those proposed by Hoffmann [1990], J. M. Vleugels [1995], Karavelas and Yvinec [2002], Emiris and Karavelas [2006] Here, we modify the online algorithm of Shamos and Hoey [1975], as described by Mulmuley [1994] to construct Apollonius diagrams, as this algorithm will allow us to dynamically add and remove new sites. This algorithm is essentially

[2] `http://www.voronoi.com/applications.htm` gives a list of hundreds of applications of Voronoi diagrams in real world systems.

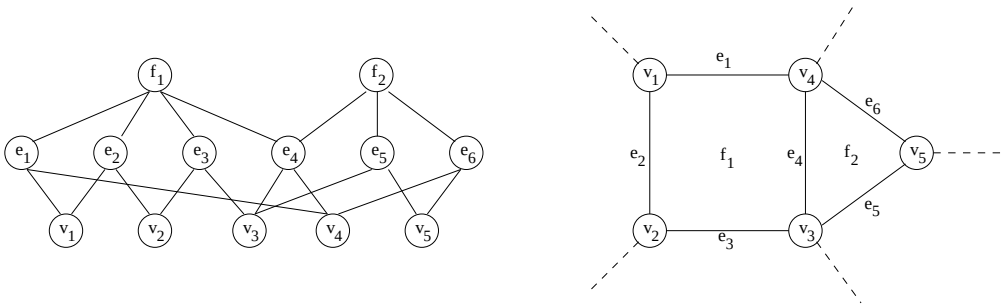


Figure 5.5: Example facial lattice (left) and the associated Voronoi diagram (right). The facial lattice stores relationships between vertices, edges, and faces allowing for efficient insertions, accesses and deletions.

identical to the one given in Karavelas and Yvinec [2002] and requires $\mathcal{O}(n \lg(n))$ computational complexity.

5.3 Facial Lattice

Before discussing the algorithm, let us first discuss a data structure to maintain the relationships between vertices, edges and faces in two dimensions, and arbitrary d-faces in multiple dimensions. This structure, called a facial lattice, contains a node for each d-face. Each node contains auxiliary information, such as pointers to adjacent $d-1$ and $d+1$-faces. For example, consider a portion of the facial lattice for the two dimensional case shown in Figure 5.5. Each edge is terminated by two vertices, on at either end. Moreover, each edge is adjacent to two faces, unless it is on the boundary of the region to be searched. Vertices, are adjacent to two edges if they are on the boundary, or three edges if they are not. Finally, faces are bounded by a set of at least three edges.

In order to facilitate the retrieval of relationships between vertices, edges, and faces, in $\mathcal{O}(1)$, this relationship information is stored in structures which allow constant time lookup. The set of edges adjacent to a vertex is stored in a hashset, while the edges adjacent to a face are stored in a circular doubly linked-list structure with a hashmap to allow us to jump to any specific edge entry in constant time. Thus, by carefully designing the nodes of the the facial lattice, we can add, find, or remove elements of the facial lattice in constant time. This property is important, as it allows us to construct a dynamic algorithm for computing Apollonius diagrams in $\mathcal{O}(n \lg(n))$ time.

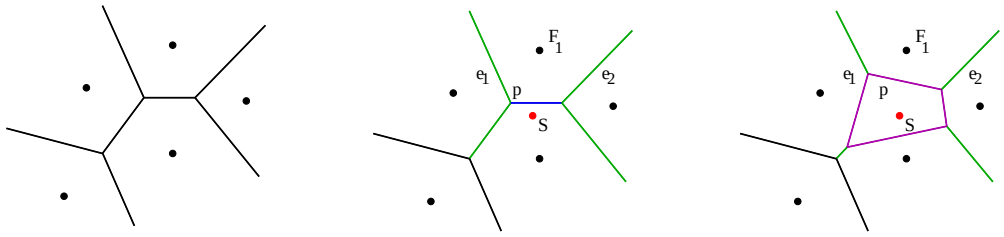


Figure 5.6: Pictorial insertion of S (red dot, right two figures) into the Voronoi diagram (left). In the middle figure, we illustrate S, along with a conflicting point p. Green lines indicate that the edge is partially conflicting, while blue lines indicate that the edge is completely conflicting. Note that e_1 and e_2 for F_1 are both partially conflicting edges. In the right figure, we show the resulting Voronoi diagram after S has been added. All totally conflicting edges (blue) are removed. Purple lines denote the new edges created in the insertion process; these edges are bisectors between S and the sites corresponding to the adjacent faces. Partially conflicting edges (green) are updated to meet with the new (purple) edges.

5.4 Algorithm in Two Dimensions

Suppose we are given a set of sites, N, in a two dimensional plane. The algorithm of Shamos and Hoey [1975] sequentially inserts a random ordering of these points. Specifically, let N^i be the first i sites from this random ordering, and $V(N^i)$ be the resulting Apollonius diagram. We are now interested in inserting site $S = S_{i+1}$ into $V(N^i)$. Figure 5.6 gives a pictorial representation of the insertion process.

Insertion of S into $V(N^i)$ is performed by first locating a vertex, p, in $V(N^i)$ which conflicts with S; that is we are interested in finding a vertex which is closer to S than it is the sites corresponding the adjacent faces. With (standard) Voronoi diagrams, such a vertex always exists. In fact, there may be many vertices which conflict with S. These vertices can be found in $\mathcal{O}(\lg(n))$ time by performing a nearest neighbor search over the vertices using a spatial tree structure, such as R-trees [Guttman, 1984]. However, with Apollonius diagrams, such a vertex many not exist; we return to this possibility later.

Once we have located p, we do a depth first search through the graph formed by the edges and vertices of $V(N^i)$ to find an edge which has one conflicting vertex, and one non-conflicting vertex; we call such edges "partially conflicting", while edges with two conflicting vertices are considering "conflicting". When a partially conflicting edge, e_1, is located, we find one of the two associated faces. Call this face F_1. Using the facial lattice described in Section 5.3, we obtain the edges of F_1, e_{F_1}, and jump in the list to e_1. We set

(a) Before inserting S (red dot) into the Apollonius diagram (black).

(b) After inserting S (red dot) into the Apollonius diagram. Purple indicates new edges.

Figure 5.7: Insertion of a site S into an Apollonius diagram when no conflicting vertices are present. In this case, the new face is sandwiched between two adjacent faces, which share edges on either side of the new face.

the iteration direction of the edges e_{F_1} (which are stored in a doubly linked list) such that the next edge is either totally or partially conflicting. We now iterate through e_{F_1} starting at e_1 and remove all conflicting edges until we arrive at a non-conflicting edge, e_2. We construct the bisector between S and the site corresponding to F_1: e_S. The endpoints of e_S are determined by the intersection of e_S with e_1 and e_2, respectively. e_S is then added to e_{F_1} between e_1 and e_2. Finally, we update the conflicting endpoints of e_1 and e_2 with the corresponding endpoints from e_S. We then move to the face adjacent to e_2 (which is not F_1) and repeat the update step on that face. This process continues until we return to F_1.

Now suppose that there is no conflicting vertex p associated with the insertion of S. Figure 5.7 shows an example in which there are no conflicting vertices. As indicated in Figure 5.7, cases in which there are no vertices that are closer to S than any other site already in the Apollonius diagram, lead to situations in which the new face, F, corresponding to S is sandwiched in between two faces: F_1 and F_2. Note that F shares edges only with F_1 and F_2. Moreover, the edges immediately preceding and succeeding the edge between S and F_1 in F_1's edge list are truncated copies of the edge that was originally

between F_1 and F_2.

Thus, to insert S into $V(N^i)$ in the case where there are no conflicting vertices, we first search through the set of sites N^i to find the site which is closest to S. Let us call this site S_1 and its associated face F_1. We know that S will result in a face F which will split one of the edges of F_1. To determine which edge will be split, we iterate through the edges of F_1 until we find the edge that intersects the ray starting at S_1 and going through S: e_{12} This edge is adjacent to faces F_1 and F_2. Once we have found the correct edge, we create the bisections between S_1 and S as well as S_2 and S (where S_2 is the site corresponding to F_2): e_{1s} and e_{2s} respectively. The end points of e_{1s} and e_{2s} are determined by where these curved segments intersect e_{12}. Note that the endpoints of e_{1s} and e_{2s} will be the same. Finally, we split e_{12} int two pieces with end points corresponding to those of e_{1s}, and insert both pieces of e_{12} into the edge sets of F_1 and F_2, along with e_{1s} for face F_1 and e_{2s} for face F_2. Thus, the most expensive part of this operation is the location of the edge e_{12} from among the edge set of F_1.

Mulmuley [1994] proves that the online algorithm for constructing (standard) Voronoi diagrams of Shamos and Hoey [1975], described above, has a computational complexity of $\mathcal{O}(n \lg(n))$. Let us now show that the generalization to Apollonius diagrams also has a computational complexity of $\mathcal{O}(n \lg(n))$.

We begin by citing two important lemmas from Mulmuley [1994], which do not change when we consider Apollonius diagrams.

Lemma 1 *The number of vertices in an Voronoi diagram with n sites is $\mathcal{O}(n)$.*

Proof The proof follows directly from Euler's relation. Specifically, if μ, ϵ, and ϕ are the number of vertices, edges, and regions of a convex polytope, then $\mu - \epsilon + \phi = 2$. Note that the number of edges is less than 2μ. Moreover, $3(\phi - 1) \leq 2\epsilon \leq 4\mu$, since each edge is adjacent to two vertices and each region is adjacent to at least three vertices. Thus, the size of a three dimensional convex hull is $\mathcal{O}(n)$, where n is the number of points defining the hull. Taking the dual of the convex hull, we obtain a three dimensional convex polytope of size $\mathcal{O}(n)$, where n denotes the number of half-spaces defining the polytope. Finally, there is a one-to-one relationship between half-spaces of a three dimensional polytope and vertices of a two dimensional Voronoi diagram.[3] Thus the number of vertices of a planar Voronoi diagram is $\mathcal{O}(n)$. ■

[3]For details see Mulmuley [1994, Sec 2.5] .

Lemma 2 *The expected number of newly created vertices in the $i+1$th random addition is $\mathcal{O}(1)$.*

Proof Following Mulmuley [1994], we estimate the the expected cost of adding S, given N^i. That is, we imagine deleting S from N^{i+1}. Since each site, S_j, is equally likely, to occur as S, it follows that the conditional expected number of vertices is

$$\frac{1}{i+1}\sum_{S\in N^{i+1}} m(S_{i+1}, N^{i+1}), \tag{5.3}$$

where $m(S_{i+1}, N^{i+1})$ denotes the number of vertices removed when S is removed from N^{i+1}. Note that, in two dimensions, each vertex is adjacent to at most three edges.[4] Therefore, each vertex of $V(N^{i+1})$ is adjacent to a bounded number of edges. Thus, the terms in the sum in Equation 5.3 are bounded by three times the number of vertices in $V(N^{i+1})$. From Lemma 1, the number of vertices in $V(N^{i+1})$ is $\mathcal{O}(i)$ in two dimensions. Thus, $m(S_{i+1}, N^{i+1}) = \mathcal{O}(i)$, and hence the expected number of newly created vertices is $\mathcal{O}(1)$. As N^{i+1} is arbitrary, the bound holds unconditionally as well. ■

In addition to the previous two lemmas, let us also show that the the intersection of two edges (whether curved or not) can be located in constant time.

Lemma 3 *The intersection of two edges can be computed in $\mathcal{O}(1)$ time.*

Proof There are three cases which we must consider: both edges are straight, one edge is straight, the other is curved, and both edges are curved. Note that edges between two sites of equal radii will be straight, while all other edges will be curved.

First, let us consider the intersection of two straight lines. Here we assume that the lines are not equal, or parallel as in the first case there are infinite solutions, while in the latter there are no solutions. Let us write the lines as $y = m_a x + c_a$ and $y = m_b x + c_b$, where m_a and m_b are the slopes of the two lines, and c_a and c_b are their y-offsets, respectively. Let $\{x_{a1}, y_{a1}\}$ and $\{x_{a2}, y_{a2}\}$ be the endpoints of edge a, and similarly for edge b. Then $m_a = (y_{a2} - y_{a1})/(x_{a2} - x_{a1})$, while

$$c_a = y_{a1} - m_a x_{a1} = y_{a1} - \frac{y_{a1} - y_{a2}}{x_{a2} - x_{a1}} x_{a1}.$$

[4] In some cases multiple vertices may lie close to one another giving the impression that more than three edges are adjacent to a single vertex.

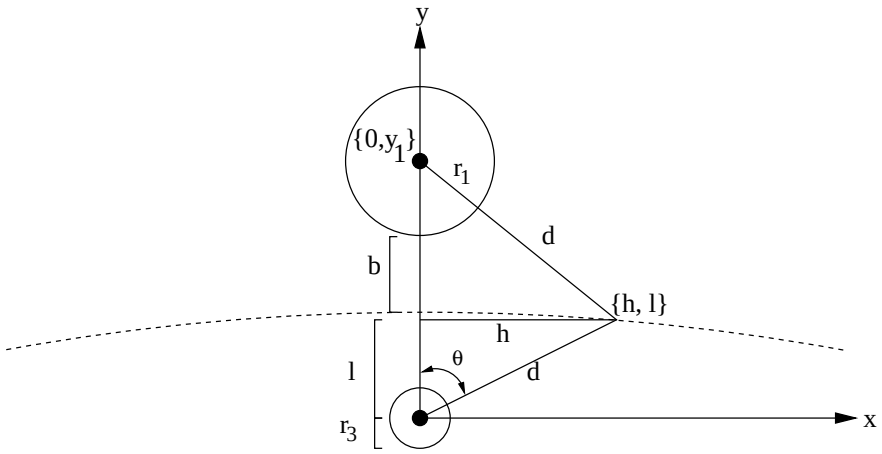


Figure 5.8: Pictorial representation of the bisector between two curved edges.

and similarly for line b.

Now suppose that either $m_a = 0$ or $m_b = 0$. Without loss of generality, assume that $m_a = 0$. As the lines are not parallel, $m_a \neq m_b$, and $y = c_a = y_{a1}$. Thus, $x = (y_{1a} - c_b)/m_b$. Similarly, if $m_a = \infty$ ($x_{2a} - x_{1a} = 0$), then $x = x_{1a}$ and $y = m_b x_{1a} + c_b$. Finally, when m_a and m_b are nonzero and finite, then $m_a x + c_a = m_b x + c_b$, so $x = (c_b - c_a)/(m_a - m_b)$, and $y = m_a(c_b - c_a)/(m_a - m_b) + c_a$. All of these expressions can be computed in constant time.

Now consider the case where both edges are curved. Since the two edges are intersecting at a point in the Apollonius diagram, they must share a face. Let us label the sites of the three faces as S_1, S_2, and S_3. Without loss of generality, let us assume that $\mathcal{R}(S_1) \geq \mathcal{R}(S_2) \geq \mathcal{R}(S_3)$. Let us assume that S_3 is at the origin, and S_1 is along the y axis. For simplicity, let r_i, be the influence radii of S_i: $r_i = \mathcal{R}(S_i)$, and let $\{x_i, y_i\}$ be the location of site S_i. Then S_1 is located at $\{0, y_1\}$, while S_3 is located at $\{0, 0\}$. Let $b = (y_1 - r2 - r1)/2$ and let d_{13} be the distance between $\mathcal{R}(S_3)$ and the bisector of S_1 and S_3. Let $\{h, \ell\}$ be a point on the bisector of S_1 and S_3, shown in Figure 5.8. Then $h = (r_3 + d_{13})\sin(\theta)$ and $\ell = (r_3 + d_{13})\cos(\theta)$, where the angle θ is measured from the y-axis.

Then

$$\begin{aligned} r_1^2 - 2r_1 d_{13} + d_{13}^2 &= (r_1 + d_{13})^2 \\ &= (r_3 + d_{13})^2 \\ &= (\ell - y_1)^2 + h^2 \\ &= d_{13}^2 + 2r_3 d_{13} + r_3^2 - 2y_1 r_3 \cos(\theta) - 2y_1 d_{13} \cos(\theta) + y_1^2. \end{aligned}$$

Solving for d_{13} yields,

$$\begin{aligned} d_{13}(r_1 - r_3 + y_1 \cos(\theta) &= \frac{1}{2}\left[r_3^2 - r_1^2 - 2y_1 r_3 \cos(\theta) + y_1^2\right] \\ &= \frac{1}{2}\left[r_3^2 - r_1^2 - 2y_1 r_3 \cos(\theta) + (r_1^2 + r_3^2 + 4b^2 + 2r_1 r_3 + 4r_1 b + 4r_3 b)\right] \\ &= r_3(r_3 + r_1 + 2b) - y_1 r_3 \cos(\theta) + 2r_1 b + 2b^2 \\ &= r_3 y_1(1 - \cos(\theta)) + 2r_1 b + 2b^2 \end{aligned}$$

Hence,

$$d_{13} = \frac{r_3 y_1(1 - \cos(\theta)) + 2r_1 b + 2b^2}{r_1 - r_3 + y_1 \cos(\theta)} \tag{5.4}$$

Moreover,

$$\begin{aligned} d_{13} + r_3 &= r_3 + \frac{r_3 y_1(1 - \cos(\theta)) + 2r_1 b + 2b^2}{r_1 - r_3 + y_1 \cos(\theta)} \\ &= \frac{r_1 r_3 - r_3^2 + r_3 y_1 \cos(\theta) + r_3 y_1 - r_3 y_1 \cos(\theta) + 2r_1 b + 2b^2}{r_1 - r_3 + y_1 \cos(\theta)} \\ &= \frac{r_1 r_3 - r_3^2 + r_3(r_1 + r_3 + 2b) + 2r_1 b + 2b^2}{r_1 - r_3 + y_1 \cos(\theta)} \\ &= \frac{2(b + r_1)(b + r_3)}{r_1 - r_3 + y_1 \cos(\theta)} \end{aligned}$$

Similarly, the distance from the origin to the bisector of S_2 and S_3 is given by

$$d_{23} + r_3 = \frac{2(b + r_1)(b + r_3)}{r_1 - r_3 + \sqrt{x_2^2 + y_2^2}\cos(\theta)}$$

where β is measured from the y-axis. Let γ be the angle between the y-axis and the line between S_2 and S_3. Then $\beta = \theta - \gamma$.

Note that the intersection of the bisectors of S_1 and S_3 and S_2 and S_3 will result in a vertex in the Apollonius diagram. This vertex will be equidistant from the influence regions of S_1, S_2 and S_3. Thus,

$$\begin{aligned} 0 &= d_{13} - d_{23} \\ &= \frac{r_1 - r_3 + y_1 \cos(\theta)}{(b + r_1)(b + r_3)} - \frac{r_2 - r_3 + \sqrt{x_2^2 + y_2^2}\cos(\theta - \gamma)}{(b + r_2)(b + r_3)} \end{aligned} \tag{5.5}$$

Unfortunately, we cannot solve the above equation analytically. However, the derivative can easily be computed. Thus, we can numerically solve the above equation using the Newton-Raphson's method [Press et al., 1992, Sec 9.4]. Once we have obtained θ, then we can easily compute d_{13} using equation 5.4, and obtain the intersection $\{x, y\}$ by noting that $x = (r_3 + d_{13})\sin(\theta)$ while $y = (r_3 + d_{13})\cos(\theta)$.

Finally, when edge a is curved but edge b is straight, we can again solve Equation 5.5, with either $r_1 = r_2$ or $r_2 = r_3$. When $r_2 = r_3$, Equation 5.5 becomes

$$0 = \frac{r_1 - r_3 + y_1 \cos(\theta)}{(b + r_1)(b + r_3)} - \frac{\sqrt{x_2^2 + y_2^2}\cos(\theta - \gamma)}{(b + r_3)^2},$$

which is no easier to solve than Equation 5.5. Again we resort to numerical solutions. When $r_1 = r_2$, then Equation 5.5 becomes

$$\begin{aligned} 0 &= \frac{r_2 - r_3 + y_1 \cos(\theta)}{(b + r_2)(b + r_3)} - \frac{r_2 - r_3 + \sqrt{x_2^2 + y_2^2}\cos(\theta - \gamma)}{(b + r_2)(b + r_3)} \\ &= y_1 \cos(\theta) - \sqrt{x_2^2 + y_2^2}\cos(\theta - \gamma). \end{aligned}$$

Solving for θ yields

$$\theta = \arctan\left(\frac{y_1 - \sqrt{x_2^2 + y_2^2}\cos(\gamma)}{\sqrt{x_2^2 + y_2^2}\cos(\gamma)}\right).$$

Again, once we have found θ, we can obtain the intersection by computing first d_{13} and using it to find $\{x, y\}$.

Note that throughout this discussion we have assumed a rotation and shift such that S_3 is at the origin, and S_1 is along the y-axis. However, this affine transform can be easily computed in $\mathcal{O}(d^2) = \mathcal{O}(1)$ time, by using a transformation and a shift matrix. The resulting intersection can be translated back into "real-world" coordinates by applying the reverse transformation: subtracting off the shift matrix and multiplying by the inverse of

the rotation matrix. Thus, we can obtain the intersection of any two edges in $O(1)$ time.[5] ■

Using these lemmas, we can now prove that we can construct an Apollonius diagram in $\mathcal{O}(n \lg n)$ time.

Theorem 2 *A Apollonius diagram can be constructed in $\mathcal{O}(n \lg n)$ time.*

Proof The proof follows the description of the algorithm that we gave earlier. Consider adding the element $S = S_{i+1}$ to the Apollonius diagram $V(N^i)$. The first step of the algorithm is to find a conflicting vertex p, if one exists. Assuming the vertices are stored in a tree structure, this search takes $\mathcal{O}(\lg(i))$ time.

Suppose that we find a conflicting vertex. Then we need to iterate through each face that is adjacent to a conflicting or partially conflicting edge. Lemma 2 states that there are only a constant number of vertices which conflict with S. Since each vertex is adjacent to at most 3 edges, then there are at most a constant number of conflicting or partially conflicting edges. Moreover, since each face that must be updated is adjacent to a conflicting or partially conflicting edge, there are also only a constant number of faces which must be updated.

Updating a face, requires that we iterate through the conflicting edges of that face. However, using our facial lattice structure, we can jump to the first conflicting edge immediate (since we know it from either the previous update or from the search for a partially conflicting edge). We then iterate through the conflicting edges of the face until we reach the first partially conflicting edge, removing conflicting edges as we go. Finally we update the two partially conflicting edges, and add a new bisector between the S and the site corresponding to the face. Note that because of the cyclic way that we move through the edges, each conflicting edge gets visited at most three times (once on the depth-first search for a partially conflicting edge, and once for each when updating the two adjacent faces. Thus, we touch at most $\mathcal{O}(1)$ edges during the update. Since the number of faces that we encounter is $\mathcal{O}(1)$, we add at most a constant number of new edges. As addition and deletion from the facial lattice can be performed in constant time, then the cost of all the additions and deletions is also $\mathcal{O}(1)$. Thus, if a conflicting vertex exists, then the expected update cost is $\mathcal{O}(\lg(i))$ (from the initial search).

Now suppose that we do not find a conflicting vertex. Lemma 1 notes that the total number of vertices in $V(N^i)$ is $O(i)$. Thus, the expected number of edges per face is

[5]The intersection between two lines, curved or straight, can also be computed without first applying the transform matrix. The derivations are similar, but more cumbersome because of the additional location variables that result from the (typically) off-axis alignment of the sites.

$\mathcal{O}(1)$. Thus we can find the correct edge in which to insert S in constant time. This edge needs to be split, two new edges need to be created, and two faces need to be updated. Using the facial lattice structure all of these operations can be done in constant expected time. Thus, even if a conflicting vertex does not exist, then the expected update cost is $\mathcal{O}(\lg(i))$.

Since the insertion of one element into $V(N^i)$ can be done in $\mathcal{O}(\lg(i))$ time, then the insertion of all n elements into the Apollonius diagram has expected computational complexity

$$\mathcal{O}\left(\sum_{i=1}^{n} \lg(i)\right) = \mathcal{O}(n \lg(n)).$$

■

Using a Apollonius diagrams, we can now show that we can efficiently solve the problem described in Section 5.1. Specifically, we can find the point θ' from Equation 5.2 which is maximal far from the influence region of any experiment in $\mathcal{O}(n \lg n)$ time.

Corallary 1 *The point in a bounded (but possible infinite) parameter space which is the furthest from all of the spheres of influence of the currently sampled experiments, θ', can be located in $\mathcal{O}(n \lg(n))$ time. That is, if $\mathcal{B}$ is the set of experiments which already have be observed, and $\mathcal{R}(\tilde{\theta})$ is the influence radius of $\tilde{\theta} \in \mathcal{B}$, then the solution to*

$$\theta' = \max_{\theta \in \Theta} \min_{\tilde{\theta} \in \mathcal{B}} \|\theta - \tilde{\theta}\|_2 - \mathcal{R}(\tilde{\theta})$$

can be computed in $\mathcal{O}(n \lg(n))$ time.

Proof This result follows directly from Theorem 2 and Lemma 1. Specifically, Theorem 2 states that we can compute an Apollonius diagram of a set of n points in $\mathcal{O}(n \lg(n))$ time. We know the vertices of this diagram correspond to the set of points which are the farthest from the sites contained by the adjoining faces. Thus, the vertex which is the furthest from the sites of its associated faces is θ'. Therefore to compute θ', we need to iterate through the vertices of the Apollonius diagram and select the one with the largest value of $\min_{\tilde{\theta} \in \mathcal{B}} \|\theta - \tilde{\theta}\|_2 - \mathcal{R}(\tilde{\theta})$. By Lemma 1, there are only $\mathcal{O}(n)$ vertices in the Apollonius diagram, so this search takes $\mathcal{O}(n)$ time. Hence, we can find the point which is maximally far from the influence region of any experiment in $\mathcal{O}(n \lg(n))$ time. ■

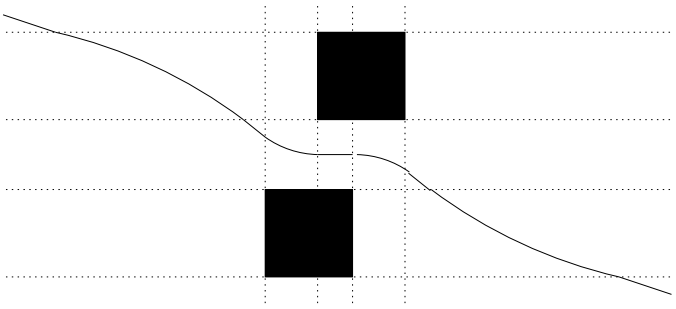


Figure 5.9: Illustration of the edge between two square sites (black). Dashed lines indicate the transitions between subedges. Edges between two square sites can consist of up to 9 subedges, which are either straight or curved.

In the proceeding, we have assume that the influence region is a circle. However, as mentioned in Section 5.1, the influence regions of experiments are typically squares. Unlike the circular case, when using squares, the edge between two sites is a composite of sub-edges, some which are straight and some which are curved, as seen in Figure 5.9. There are at most nine different subedges which compose a single edge between two squares. Since the curved sections occur when the edge is closest to a corner on one square and an edge on the second square, the shape of the curved subedges is parabolic. Thus, the intersections of the curved subedges either with other curved subedges or with straight subedges can be computed analytically. The followings result shows that the assumption of circular influence regions is not necessary in two dimensions.

Lemma 4 *Edges between two aligned square sites are composed of only a constant number of sub-edges.*

Proof An edge between two faces is the region of the Apollonius diagram in which the two faces are equidistant from each other. Moreover, the endpoints of the edge are the only two points in which another site is as close or closer to the edge than the two sites defining the edge. Thus, the edge between two vertices is influenced only by its two adjacent faces.

Now consider an arbitrary edge, E, between two square sites. The points where two subedges of E meet are points where the nearest element of one of the squares transitions from a corner to a side, or visa versa. In Figure 5.9, these locations are illustrated with dotted lines. The number of subedges is bounded by the number of combinations of sides and corners of one square with sides and corners and sides of the second square. Since each square has four sides and four corners, the maximal number of combinations is 64.[6]

[6]Not all of these combinations are possible. Figure 5.9, shows that there are only 25 distinct regions

Thus, the number of subedges is $\mathcal{O}(1)$. ■

Lemma 5 *The intersection of two square edge (edges between square sites), can be computed in constant time.*

Proof First, let us consider the cost of intersecting two subedges. If both subedges are straight, then Lemma 3 states that their intersection can be computed in constant time.

Now suppose subedge e_1 is a curved subedge. Curved subedges occur when the closest point on one square is a corner, c, and the closet point on the other square is on a side, s. Let us orient the space, such that s is along the x-axis, and c falls on the y-axis; that is c is located at $\{0, y_c\}$. Then since e_1 bisects c and s, then $y^2 = x^2 + (y - y_c)^2$, or

$$y = \frac{x^2}{2y_c} + \frac{y_c}{2}$$

Since e_1 has a closed form solution, we can compute the intersection of e_1 with either a straight or curved subedge analytically, just as we did in Lemma 3. Specifically, if e_2 is straight, then

$$m_2 x + b_2 = y = \frac{x^2}{2y_c} + \frac{y_c}{2}$$

Solving the quadratic, we find that

$$x = y_c m_2 \pm c\sqrt{m_2^2 - 1 - \frac{2b_2}{y_c}}$$

and y can easily be found by substituting x into $y = m_2 x + b$.

If e_1 and e_2 are both curved then

$$\frac{x^2 + y_{c1}^2}{2y_{c1}} = y = \frac{x^2 + y_{c2}^2}{2y_{c2}}$$

Solving for x, we find that $x = \sqrt{y_{c1} y_{c2}}$. Hence $y = 1/2(y_{c1} + y_{c2})$.

Thus we can compute the intersections of subedges in constant time. Since there are at most a constant number of subedges per edge (Lemma 4, we can compute the intersections of all pairs of subedges and return the intersections that are within the corresponding

where the edge changes subedges. Moreover, a single edge cannot visit all 25 of these regions.

subedges' endpoints, in $\mathcal{O}(1)$ time. ■

Using these two lemmas, and our analysis from Theorem 2, let us now show that generalized Voronoi diagrams with square sites can also be computed in polynomial time.

Theorem 3 *A generalized Voronoi diagram where the sites are aligned squares can be constructed in $\mathcal{O}(n \lg n)$ time.*

Proof Constructing generalized Voronoi diagrams where the sites are mutually aligned squares uses the same algorithm that was described for circular sites. The only difference comes in computing the intersection of two edges. However, from Lemma 5, we know that we can intersect two edges that result from square sites in constant time.

However, since the intersection of two edges can be computed in constant time, then the complexity of the algorithm using square sites is identical (within a constant) to that of computing an Apollonius diagram. That is, the complexity of computing a generalized Voronoi algorithm with n square sites is $\mathcal{O}(n \lg(n))$. ■

Thus, for two dimensions, we have an efficient algorithm for computing the point in some bounded regions which is furthest from a set of circles. In practice, computing this point is much quicker than computing an estimate of the Gaussian Process used in the active learning framework in Section 2.1.

5.4.1 Experimental Results

Let us now assess the greedy algorithm which selects the point that is furthest from the influence region of any of the previously sampled points and compare it with active learning algorithms in Chapter 2. In particular, we consider the random, entropy, variance, ent-var, and straddle heuristics of Section 2.2, to the following heuristics:

Maximum Distance to Influence Region Suppose we are given a set of sampled points $\mathcal{B}$ and a candidate set, $\mathcal{Q}$, as in Section 2.1. This heuristic, selects the point from $\mathcal{Q}$ which is the furthest from any of the influence regions of any of the points in $\mathcal{B}$. That is, this heuristic selects θ', where

$$\theta' = \max_{\theta \in \mathcal{Q}} \min_{\tilde{\theta} \in \mathcal{B}} \|\theta - \tilde{\|}_2 - \mathcal{R}(\tilde{\theta}).$$

Note that this heuristic provides the approximate solution to Equation 5.2, finding the best solution among the set $\mathcal{Q}$. This heuristic does not require any estimates from Gaussian Process. Thus, the complexity of this heuristic is $\mathcal{O}\left(|\mathcal{Q}|\lg(|\mathcal{B}|)\right)$, where $|\cdot|$ yields the sizes of the associated set. We refer to this heuristic as maxdist.

Maximum Voronoi Vertex The second heuristic, voronoi, computes an Apollonius diagram from the sampled points, and returns the vertex which is maximally distant from the (circular) influence region of all of the sampled points. In theory, the result of voronoi and maxdist should be equivalent if the candidate set is large enough, or happened to contain the true solution to Equation 5.2.

For each of these heuristics, we compute three measures of performance. First we measure their classification accuracies after 100 samples, as we did in Section 2.2. Secondly, we compute the fraction of the space that is covered by the influence regions of the 100 selected samples. This is computed by randomly selecting 10,000 points and determining what fraction of the points lie within the influence regions of the sampled experiments. Finally, we compute the size of the largest hole in parameter space, where largest hole is defined to be the maximum distance from any point in the parameter space to the nearest influence region. This metric is computed by building an Apollonius diagram out of the 100 sampled points and then iterating among the vertices to find the point which is maximally distant from the nearest influence region.

Each of these performance measures was tested on the 2D Peak, 2D DeBoor, 2D Sine, and 4D Sine target functions from Section 2.2. Note that the voronoi algorithm only works for two dimensional tasks, so results for the 4D Sine function are not included. Moreover, since the largest hole metric relies on the Apollonius diagram code, we did not compute this metric for any of the heuristics on the 4D Sine data set.

Results are shown in Table 5.1, 5.2, and 5.3. Unsurprisingly, the straddle out performs all heuristics in terms of classification accuracy. However, it does not perform well in terms of minimizing the largest hole (Table 5.3). Indeed, the size of the largest hole using the straddle was typically larger than the one found after using the variance heuristic. In one of the three cases, the largest hole derived from the samples chosen by the straddle heuristic was larger than any other heuristic except the entropy heuristic; recall that the entropy heuristic tends to sample points only from a specific region in parameter space.

On the other hand, the voronoi heuristic performs well on all tasks. While not always the best heuristic to maximize classification accuracy and coverage fraction, the voronoi algorithm tended to be near the top. Moreover, the voronoi algorithm was the clear winner on the largest hole metric, beating the other heuristics by significant margins.

	2D Peak	2D DeBoor	2D Sine	4D Sine
random	0.978±0.007	0.991±0.005	0.958±0.007	0.940±0.003
entropy	0.971±0.000	0.994±0.015	0.808±0.189	0.914±0.044
variance	0.989±0.002	0.996±0.001	0.988±0.001	0.930±0.003
ent-var	0.998±0.000	1.000±0.000	0.992±0.001	0.950±0.002
straddle	0.999±0.000	1.000±0.000	0.996±0.000	0.971±0.001
maxdist	0.994±0.002	0.992±0.005	0.992±0.001	0.952±0.002
voronoi	0.999±0.000	0.998±0.000	0.992±0.001	——

Table 5.1: Classification accuracy for various heuristics after selecting 100 samples from the corresponding target function.

	2D Peak	2D DeBoor	2D Sine	4D Sine
random	0.819±0.028	0.000±0.000	0.507±0.023	0.041±0.009
entropy	0.327±0.208	0.000±0.000	0.118±0.062	0.003±0.002
variance	0.867±0.015	0.061±0.010	0.578±0.015	0.012±0.005
ent-var	0.871±0.013	0.000±0.000	0.456±0.025	0.012±0.004
straddle	0.683±0.021	0.000±0.000	0.285±0.023	0.008±0.003
maxdist	0.522±0.046	0.000±0.000	0.516±0.025	0.022±0.007
voronoi	0.866±0.006	0.160±0.013	0.499±0.025	——

Table 5.2: Influence region coverage fraction for various heuristics after selecting 100 samples from the corresponding target function.

	2D Peak	2D DeBoor	2D Sine
random	0.633±0.120	6.309±0.913	2.075±0.400
entropy	1.718±1.046	17.138±4.932	6.422±2.512
variance	0.435±0.047	3.127±0.142	1.031±0.117
ent-var	0.239±0.011	6.792±1.058	0.869±0.056
straddle	0.258±0.043	11.296±1.162	1.325±0.191
maxdist	0.496±0.081	9.320±0.438	0.725±0.048
voronoi	0.183±0.010	2.578±0.075	0.703±0.024

Table 5.3: Maximal distance to the nearest influence region for various heuristics after selecting 100 samples from the corresponding target function.

Finally, note that while the maxdist heuristic and voronoi heuristic optimize nearly the same function, the maxdist heuristic performs substantially worse than the voronoi heuristic. This is a result of the fact that the maxdist heuristic is limited to the small number of elements in $\mathcal{Q}$. We suspect that if we were to increase the size of $\mathcal{Q}$, then the performance of maxdist would be more equivalent to voronoi.

5.5 Computing Derivatives

Throughout the previous discussion, we have assumed that the derivatives of the target function were known. However, this is hardly ever the case. Nonetheless, we can compute approximate values of the derivatives in one of many ways. Perhaps the simplest way is to assume some (reasonable) value for the derivatives, and then sample a certain number of points. Using these samples, we can then compute a better estimate for the derivatives, which we can use to select another set of points.

Note that if we underestimate the derivatives, we will over smooth our estimate of the target, and hence miss portions of the target function's level-set. However, if we overestimate the derivatives, our sampling method will resort to variance weighted sampling, as the variance term will overpower all other factors in all of the heuristics.

Thus, the best approach is probably to overestimate the derivatives by multiplying the observed derivatives of the currently sampled points by a constant factor. Moreover, if after every j experiments, we sample a point at random, we can be assured that the observed derivative estimates are not being biased by our assumed derivatives.

5.6 Extending the Algorithm to Multiple Dimensions

Finally, let us consider extending the Voronoi algorithm to multiple dimensions. As noted in Section 5.1 the problem cannot be formulated as either a linear or quadratic program. However, there are other strategies that we can use to (at least approximately) compute solutions to Equation 5.2 in multiple dimensions.

First, Bowyer [1981] and Watson [1981] give poly-time algorithms for computing d-dimensional Delaunay tessellations. As Delaunay tessellations are the duals of Voronoi diagrams, we can thus compute the portions of the Voronoi diagram which are interior to the convex hull of the sites in $\mathcal{O}\left(n^2\right)$ time.

Similarly, we can extend the techniques above for Apollonius diagrams to multiple

dimensions. Computing intersections of curved edges in d dimensions can be calculated in constant time using the same techniques as in Section 5.4. Thus, algorithms exist for computing a multi-dimensional Apollonius diagram interior to the convex hull of its n sites in $\mathcal{O}(n^2)$ time [Emiris and Karavelas, 2006, Boissonnat et al., 2006]. Moreover, multi-dimensional generalized Voronoi diagrams can be computed using tree-based grid methods [J. M. Vleugels, 1995, Hoff III et al., 1999].

In order to ensure that all points in the parameter space are considered by the (standard or generalized) Voronoi diagram algorithm, we must first select the 2^d corners of the d-dimensional parameter space. This ensures that every point in the parameter space is within the convex hull of the sampled points. As a result complexity for the Voronoi heuristic using Apollonius diagrams becomes $\mathcal{O}\left(2^d + n^2\right)$. While this selection of corners results in a non-polynomial algorithm (assuming $\mathrm{P} \neq \mathrm{NP}$), for the moderate dimensional problems the algorithm is computationally feasible. For instance, consider the task of computing $1 - \alpha$ confidence intervals for the WMAP data in Section 4.1.1. This seven dimensional data set would require only 128 points to sample all of the corners of parameter space. Compared with the ~ 1.3 million models that were run, this number is inconsequential.

The second approach for computing the point which is maximally distant from the influence regions of the currently sampled points is to use the maxdist heuristic. As we saw in Section 5.4.1, this heuristic did not perform well when the size of the candidate set was highly restricted. However, increasing the size of the candidate set may drastically increase performance. Moreover, the cost of computing the maxdist heuristic only grows linearly with the number of dimensions, instead of exponentially. On the other hand, exponentially more points are need to cover the space with random samples as the dimension increases.

Finally, note that the true influence regions of the sampled points are hypercubes, not hyperspheres. As seen in Section 5.1, the ratio of the size of a d-dimensional hypersphere and the d-dimensional hypercube in which it is inscribed goes to zero as d increases. When $d = 7$, the hypersphere encloses less than 4% of the volume of the hypercube. Thus, we could modify the maxdist heuristic to compute the distance from the candidate point to the influence region as a hypercube rather than a hypersphere. This change cannot be easily made for the curved Voronoi diagrams, as the cost to compute the intersection of $j - 1$ faces when the sites are hypercubes is exponentially dependent on the dimension.

5.7 Summary

In this chapter we have discussed algorithms for computing convergence. We noted that computing point-wise convergence is computationally infeasible. Instead, we calculate the largest uncovered hole in the parameter space, and seek to minimize its size.

In order to take advantage of the values of the experiments chosen (instead of just their locations in the space), we need to make assumptions as to the derivatives of the target function. With these derivative assumptions, we can compute "influence regions" for each sample; any point within a sample's influence region is provably not part of the target function's level set.

Using the influence regions of the sampled experiments, we can compute the size of the largest hole in parameter space. Specifically, the size of this hole is equal to the maximum distance from any point in the parameter space to the nearest influence region of the sampled points. The size of this hole corresponds to the largest possible size of a unobserved peak in the target function.

In order to compute and minimize the size of this unobserved peak, we have proposed two heuristics. In the first, we have showed that using a generalized Voronoi diagram — in this case an Apollonius diagram — the point which is maximally distance from the nearest influence region in a bounded two dimensional region can be found in $\mathcal{O}(n \lg(n))$ time. Current ideas of extending the Voronoi diagram based solution to multiple dimensions do not yield polytime algorithms.

The other approach to computing the largest hole is to randomly sample points throughout the parameter space and compute their distance to the nearest influence region based on the currently available samples. While an approximate solution, this algorithm scales linearly with dimension, rather than exponentially. Experiments show that the Voronoi diagram based solution works better than the approximate solution for a number of problems. Moreover, the Voronoi diagram based algorithm is nearly as good at finding function level-sets as the straddle heuristic. We believe that the random sampling algorithm would perform nearly as well as the Apollonius diagram based solution if the number of random candidates were increased.

Chapter 6

Conclusions

We have developed an active learning framework which can be used to efficiently learn specific properties of target functions. In particular, we show that the straddle heuristic can be used to learn function level-sets much more efficiently than techniques which try to learn the target function over the entire domain. In addition, we have developed the threshvar heuristic to learn subsets of a target function (where it is below a specified value), as well as the var-maxvarstraddle heuristic to learn the level-sets of functions which are sums of sub-functions, which can be independently queried.

While these active learning techniques can be applied to many domains, we have shown how they are useful for efficiently computing statistical inferences. We described how our algorithms can be used to increase the efficiency of χ^2 tests, the confidence ball method, and minimax expected size (MES) confidence regions. For each of these procedures, a function level-set search must be preformed to determine the extent of the $1 - \alpha$ confidence regions.

When using the MES confidence procedure, additional savings can be achieved by first restricting the parameter space based on a χ^2 test and then sub-sampling this restricted region using the threshvar heuristic. Once we have obtained the necessary samples, we have shown how the MES procedure can be formulated as a convex game with a sparse payoff matrix. Solving this game using linear programming was found to be orders of magnitude faster than standard fictitious play algorithms. The solution of the MES game can then be translated into a target function where the boundaries of the $1 - \alpha$ confidence regions corresponds to a specific level-set.

While Bayesian procedures can be used to compute credible intervals, our techniques cannot be applied to Bayesian inference techniques, due to the fact that the boundary of

the $1 - \alpha$ credible region cannot be computed until after the parameter space has been sampled. Additionally, Bayesian inferences have many troubling aspects, including the fact that $1 - \alpha$ credible regions may not contain the true parameter vector $1 - \alpha$ fraction of the time, in a frequency sense. Moreover, we showed that the size of $1 - \alpha$ confidence regions derived by the MES procedure are similar in size to those computed by Bayesian methods. Finally, we showed that Bayesian methods are extremely data inefficient. In particular, our methods were more than four orders of magnitude more data efficient than MCMC and two orders of magnitude more efficient than simple grid-based techniques. Thus, we assert that scientific anaylses should consider using frequentist methods based on our active learning techniques.

Using the combination of our active learning heuristics and frequentist inference techniques, we computed $1 - \alpha$ confidence regions for several astronomical data sets. We observed that our technique explicitly retains the d dimensional confidence surface, allowing for extremely powerful analyses. For instance, we can compute the effects of adding restrictions to the parameter ranges after the fact without needing additional samples. Moreover, we can combined multiple data sets and produce confidence regions which are much tighter than a simple intersection of the one or two dimensional views. Such analyses are not possible with Bayesian methods, as changes to the priors invalidate the derived inferences.

Finally, we looked at two techniques which can be used to determine the level of convergence of our sampling heuristics. Since computing convergence point-wise is computationally infeasible, we suggest computing the size of the maximum object that could still be observed. This size is equivalent to the size of the maximum distance from any point in the parameter space to the influence regions of the currently sampled points, where the influence region of a point is determined by the product of the experiment's distance to the boundary and the maximum derivatives of the target function. When the maximum derivatives are unknown, they can be approximated using the samples selected as well as additional random samples.

We give two algorithms to compute these "largest holes." The first generalizes Voronoi diagrams to include sites with non-zero radii. The second uses random sampling to approximately compute the maximum distance to the nearest influence region. Experimental results suggest that the Voronoi diagram based method outperforms the random sampling technique, but this may be due to using too few points in the random sampling technique. We saw that both the Voronoi based diagram approach and the random sampling approach can be extended to multiple dimensions. However, the Voronoi diagram algorithm is likely exponential in dimension, while the random sampling method scales linearly with dimension.

6.1 Future Work

There are several areas where this research could be furthered. First, it would be interesting to determine if an exact solution of the point in a bounded parameter space which is maximally distance from a set of spheres can be computed in polynomial time. As mentioned in Section 5.1, the minimax formulation can not be written as either a linear or quadratic program. However, there may be other formulations that will allow us to solve this problem, or at least an approximation. On such solution is to use the Voronoi diagram based algorithm. The multidimensional Voronoi diagram algorithms considered here only examine those points which are within the convex hull of the samples; thus we must sample all of the corners to ensure the entire space is actively considered. However, there may be ways to relax this constraint.

Secondly, it would be interesting to derive some theoretical guarantees for our algorithms. One common approach is to prove the function is submodular and then invoke the results of Nemhauser et al. [1978], which states that a submodular greedy algorithm performs near optimally. However, this result only holds if the algorithm cannot observe the values of the experiments until all experiments are sampled. Formulations of this framework for active learning cases are more difficult, as the optimal solution can be significantly better than the case where all experiments are unobserved until the end.

Moreover, note that many of our heuristics are not even submodular. For instance, the Voronoi diagram algorithm is not submodular. Consider the case of one point in the middle of a two dimensional parameter space. Since the distance to all four of the corners is equivalent, selecting any of these corners will not reduce the maximum distance to the nearest influence region. It is not until all four corners have been sampled that the size of the largest hole decreases. Thus, choosing the fifth point fifth has significantly more value than choosing that same point second. While it is possible to change the heuristic to be more akin to mutual information (rather than variance), this change forces us to essentially perform an integration over all space, which is infeasible.

Finally, it would be interesting to apply this technique to a host of other data sets. In particular, we are interested in scaling up the active learning techniques here to a situation where we have multiple data sets, all of which share the same model and parameter space. In such a setting, it would interested to determine how best to choose samples. Should samples be chosen which best help identify the worst performing data set (thereby ensuring all data sets achieve some minimum standard — "no data set left behind"), or should we save our efforts for computing models which are useful in determining the $1-\alpha$ confidence intervals for a large number of models. Since the data sets all represent the same objects — here galaxies — we expect that using the data sets to guide which models are chosen will

be significantly more efficient than first trying to learn about the entire parameter space, and then applying this knowledge to all of the galaxies.

Appendix A

Estimating τ, the radius of the confidence ball

The following derivation is taken from Genovese et al. [2004]. Recall from §3.3.1 that the cosine basis is defined on $[0, 1]$ by

$$\phi_j(z) = \begin{cases} 1 & \text{for } j = 0 \\ \sqrt{2}\cos(\pi j z) & \text{for } j = 1, 2, 3, \ldots \end{cases}$$

If j and k are distinct, positive integers, then

$$\begin{aligned} \phi_j \phi_k &= 2\cos(\pi j z)\cos(\pi k z) \\ &= \cos(\pi(j+k)z) + \cos(\pi(j-k)z) \\ &= \frac{1}{\sqrt{2}}(\phi_{j+k} + \phi_{|j-k|}). \end{aligned}$$

Moreover, if $j > 0$, then $\phi_j^2 = 2\cos^2(\pi j z) = \cos(2\pi j z) + 1 = \frac{1}{\sqrt{2}}\phi_{2j} + \phi_0$. Therefore, as mentioned in §3.3.1,

$$\Delta_{jk\ell} = \begin{cases} 1 & \text{if } \#\{j, k, l = 0\} = 3 \\ 0 & \text{if } \#\{j, k, l = 0\} = 2 \\ \delta_{jk}\delta_{0\ell} + \delta_{j\ell}\delta_{0k} + \delta_{k\ell}\delta_{0j} & \text{if } \#\{j, k, l = 0\} = 1 \\ \frac{1}{\sqrt{2}}(\delta_{\ell,j+k} + \delta_{\ell,|j-k|}) & \text{if } \#\{j, k, l = 0\} = 0 \end{cases}.$$

Let $w(z) = 1/\sigma^2(z)$, such that $w^2(z) = \sum_j w_j \phi_j(z)$. As in §3.3.1, we let $\hat{\rho}_j = \lambda_j \mathcal{Z}_j$, where

$$\mathcal{Z}_j = \frac{1}{n}\sum_{i=1}^{n} Y_i \phi_j(z_i)$$

and $1 \geq \lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_n \geq 0$ are shrinkage coefficients. In this work, we use a special case of monotone shrinkage in which

$$\lambda_j = \begin{cases} 1 & \text{for } j \leq J \\ 0 & \text{for } j > J \end{cases}$$

for $J \in [0, 1, 2, \ldots, n]$ such that J minimizes Stein's unbiased risk estimate given in Equation 3.3. With these definitions, the loss can be written as

$$\begin{aligned} L(\mu, \hat{\mu}) &= \int_0^1 \left(\frac{\hat{\mu}(z) - \mu(z)}{\sigma(z)} \right)^2 dz \\ &= \sum_{j,k,\ell} (\rho_j - \hat{\rho}_j)(\rho_k - \hat{\rho}_k) w_\ell \int_0^1 \phi_j \phi_k \phi_\ell \\ &= \sum_{j,k} (\rho_j - \hat{\rho}_j)(\rho_k - \hat{\rho}_k) \sum_\ell w_\ell \Delta_{jk\ell} \\ &= (\rho - \hat{\rho})^T W (\rho - \hat{\rho}), \end{aligned}$$

where $W_{jk} = \sum_\ell w_\ell \Delta_{jk\ell}$. As in §3.3.1, let D and $\bar{D} = 1 - D$ be diagonal matrices with 1's in the first J and last $n - J$ entries respectively. Then $\hat{\rho} = D\mathcal{Z}$, where $\mathcal{Z}$ is again assumed to be Normal (ρ, B). Thus, $\mathsf{E}[\hat{\mu}] = D\rho$, $\mathsf{Cov}(\hat{\mu}_j, \hat{\mu}_k) = \lambda_j \lambda_k B_{jk}$ and $\mathsf{Var}(\hat{\mu}) = DBD$. The risk then becomes

$$\begin{aligned} R = \mathsf{E}[L] &= \mathsf{E}\left[(\rho - \hat{\mu})^T W (\rho - \hat{\mu}) \right] \\ &= \operatorname{trace}(DWDB) + \rho^T \bar{D} W \bar{D} \rho \\ &= \operatorname{trace}(DWDB) + \sum_{j,k} \rho_j \rho_k \bar{\lambda}_j \bar{\lambda}_k W_{jk} \end{aligned}$$

An unbiased estimate can be obtained by replacing $\rho_j \rho_k$ with $\mathcal{Z}_j \mathcal{Z}_k - B_{jk}$. The result is

$$\hat{R} = \mathcal{Z}^T \bar{D} W \bar{D} \mathcal{Z} + \operatorname{trace}(DWDB) - \operatorname{trace}(\bar{D} W \bar{D} B)$$

It follows that

$$\hat{L} - \hat{R} = \rho^T W \rho - \mathcal{Z}^T C + \mathcal{Z}^T A \mathcal{Z} + \operatorname{trace}(A\mathcal{Z})$$

where $A = DW + WD - W$ and $C = 2DW\rho$. Moreover,

$$\begin{aligned} \mathsf{Var}(\hat{L} - \hat{R}) &= \mathsf{Var}(\mathcal{Z}^T A \mathcal{Z} - \mathcal{Z}^T C) \\ &= \mathsf{Var}(\mathcal{Z}^T A \mathcal{Z}) + \mathsf{Var}(\mathcal{Z}^T C) - 2\,\mathsf{Cov}(\mathcal{Z}^T A \mathcal{Z}, \mathcal{Z}^T C) \\ &= 2\operatorname{trace}(ABAB) + \rho^T Q \rho \end{aligned}$$

where $Q = ABA + WDBDW - 2ABDW$. Plugging in unbiased estimates of the linear and quadratic forms involving ρ, we get the following estimate for the variance of the pivot process:

$$\hat{\tau}^2 = 2\,\mathrm{trace}(ABAB) + \mathcal{Z}^T Q \mathcal{Z} - \mathrm{trace}(QB).$$

Bibliography

M. E. Abroe, A. Balbi, J. Borrill, E. F. Bunn, S. Hanany, P. G. Ferreira, A. H. Jaffe, A. T. Lee, K. A. Olive, B. Rabii, P. L. Richards, G. F. Smoot, R. Stompor, C. D. Winant, and J. H. P. Wu. Frequentist estimation of cosmological parameters from the MAXIMA-1 cosmic microwave background anisotropy data. *Monthly Notices of the Royal Astronomical Society*, 334:11–19, July 2002. doi: 10.1046/j.1365-8711.2002.05383.x. 4.2.1

D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988. 2

P. Astier, J. Guy, N. Regnault, R. Pain, E. Aubourg, D. Balam, S. Basa, R. G. Carlberg, S. Fabbro, D. Fouchez, I. M. Hook, D. A. Howell, H. Lafoux, J. D. Neill, N. Palanque-Delabrouille, K. Perrett, C. J. Pritchet, J. Rich, M. Sullivan, R. Taillet, G. Aldering, P. Antilogus, V. Arsenijevic, C. Balland, S. Baumont, J. Bronder, H. Courtois, R. S. Ellis, M. Filiol, A. C. Gonçalves, A. Goobar, D. Guide, D. Hardin, V. Lusset, C. Lidman, R. McMahon, M. Mouchet, A. Mourao, S. Perlmutter, P. Ripoche, C. Tao, and N. Walton. The Supernova Legacy Survey: measurement of Ω_M, Ω_Λ and w from the first year data set. *Astronomy & Astrophysics*, 447:31–48, February 2006. 4.1.2, 4.2.2, 4.16(a), 4.2.3

L. Atlas, D. Cohn, R. Ladner, M. A. El-Sharkawi, and II R. J. Marks. Training connectionist networks with queries and selective sampling. In *Advances in Neural Information Processing Systems*, volume 2, pages 566–573, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc. 2.1

G. W. Barlow. Hexagonal territories. *Animal Behavior*, 22:876–878, 1974. 5.2.1

E. Baum. Neural net algorithms that learn in polynomial time from examples and queries. *IEEE Transactions on Neural Networks*, 2(1), January 1991. 2

C. L. Bennett, R. S. Hill, G. Hinshaw, M. R. Nolta, N. Odegard, L. Page, D. N. Spergel, J. L. Weiland, E. L. Wright, M. Halpern, N. Jarosik, A. Kogut, M. Limon, S. S. Meyer, G. S. Tucker, and E. Wollack. First-Year Wilkinson Microwave Anisotropy Probe

(WMAP) Observations: Foreground Emission. *Astrophysical Journal Supplement Series*, 148:97–117, September 2003. doi: 10.1086/377252. 4.1.1, 4.23(a)

R. Beran. React scatterplot smoothers: superefficiency through basis economy. *Journal of the American Statistical Association*, 95(449):155–171, March 2000. 3.3.2

R. Beran and L. Dümbgen. Modulation of estimators and confidence sets. *Annuals of Statistics*, 26:1826–1856, 1998. 3.3.1, 3.3.2

T. Beyer, G. Schaller, A. Deutsch, and M. Meyer-Hermann. Parallel dynamic and kinetic regular triangulation in three dimensions. *Computer Physics Communications*, 172(2): 86–108, November 2005. 5.2.1

J.-D. Boissonnat and M. Teillaud, editors. *Effective Computational Geometry for Curves and Surfaces*, volume 12 of *Mathematics and Visualization*. Springer, 2006. 5.2

J.-D. Boissonnat, C. Wormser, and M. Yvinec. Curved Voronoi diagrams. In Jean-Daniel Boissonnat and Monique Teillaud, editors, *Effective Computational Geometry for Curves and Surfaces*, pages 67–116. Springer-Verlag, Mathematics and Visualization, 2006. 5.6

C. E. Bonferroni. Teoria statistica delle classi e calcolo delle probabilità. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, 8:3–62, 1936. 2.4, 3.4.4, 4.2.3

A. Bowyer. Computing dirichlet tessellations. *The Computer Journal*, 24(2):162–166, 1981. 5.6

B. Bryan and J. Schneider. Actively learning level-sets of composite functions. Technical Report CMU-ML-07-121, Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, December 2007. 3

B. Bryan, J. Schneider, C. J. Miller, R. C. Nichol, C. R. Genovese, and L. Wasserman. Active learning for identifying function threshold boundaries. In *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, 2005. 2.2, 1, 2.3.2, 2.4.2, 7

B. Bryan, H. B. McMahan, C. M. Schafer, and J. Schneider. Efficiently computing minimax expected-size confidence regions. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pages 97–104, New York, NY, USA, 2007a. ACM Press. 2, 3.4.5

B. Bryan, J. Schneider, R. C. Nichol, C. J. Miller, C. R. Genovese, and L. Wasserman. Mapping the cosmological confidence ball surface. *Astrophysical Journal*, 665:25, August 2007b. 1, 2.2, 7

B. Bryan, J. Schneider, and C. M. Schafer. Active learning for subsampling functions below a specified level-set. Technical Report CMU-ML-07-120, Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, December 2007c. 2

B. Bryan, J. Schneider, C. M. Schafer, and H. B. McMahan. Computational tools for efficiently constructing confidence regions. Submitted to JMLR, October 2007d. 1, 2, 2

J. A. Byers. Dirichlet tessellation of bark beetle spatial attack points. *Journal of Animal Ecology*, 61(759-768), 1992. 5.2.1

N Christensen, R Meyer, L Knox, and B Luey. Bayesian methods for cosmological parameter estimation from cosmic microwave background measurements. *Classical and Quantum Gravity*, 18:2677–2688, June 2001. 4.1.1

D. A. Cohn, L. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201, 1994. 2.1

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129, 1996. 2.1

N. A. C. Cressie. *Statistics for Spatial Data*. Wiley, New York, 1991. 2.1

T. M. Davis, E. Mörtsell, J. Sollerman, A. C. Becker, S. Blondin, P. Challis, A. Clocchiatti, A. V. Filippenko, R. J. Foley, P. M. Garnavich, S. Jha, K. Krisciunas, R. P. Kirshner, B. Leibundgut, W. Li, T. Matheson, G. Miknaitis, G. Pignata, A. Rest, A. G. Riess, B. P. Schmidt, R. C. Smith, J. Spyromilio, C. W. Stubbs, N. B. Suntzeff, J. L. Tonry, W. M. Wood-Vasey, and A. Zenteno. Scrutinizing Exotic Cosmological Models Using ESSENCE Supernova Data Combined with Other Cosmological Probes. *Astrophysical Journal*, 666:716–725, September 2007. doi: 10.1086/519988. (document), 4.2(b), 4.2, 4.1.2, 4.2.2, 4.2.2, 4.2.2, 4.16(b), 4.16, 4.2.2, 4.2.3, 4.2.3, 4.23(b), 4.5, 4.3

B. Delaunay. Sur la sphre vide. *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk*, 7:793–800, 1934. 5.2.1

J. Denes and A. D. Keedwell. *Latin Squares and Their Applications*. Academic Press, Inc., New York, December 1974. 2

M. H. Dickinson, F. O. Lehmann, and W. P.Chan. The control of mechanical power in insect flight. *American Zoology*, 38(4):718–728, 1998. 1

M. Dresher and S. Karlin. Solutions of convex games as fixed-points. In H. W. Kuhn and A.W. Tucker, editors, *Contributions To The Theory of Games: Volume 2*, number 28 in Annals of Mathematics Studies, pages 75–86. Princeton University Press, 1953. 3.4.1, 3.4.1

J. Dunkley, M. Bucher, P. G. Ferreira, K. Moodley, and C. Skordis. Fast and reliable Markov chain Monte Carlo technique for cosmological parameter estimation. *Monthly Notices of the Royal Astronomical Society*, 356:925–936, January 2005. doi: 10.1111/j.1365-2966.2004.08464.x. 3.5, 3.5.2, 4.2.1

D. J. Eisenstein, I. Zehavi, D. W. Hogg, R. Scoccimarro, M. R. Blanton, R. C. Nichol, R. Scranton, H.-J. Seo, M. Tegmark, Z. Zheng, S. F. Anderson, J. Annis, N. Bahcall, J. Brinkmann, S. Burles, F. J. Castander, A. Connolly, I. Csabai, M. Doi, M. Fukugita, J. A. Frieman, K. Glazebrook, J. E. Gunn, J. S. Hendry, G. Hennessy, Z. Ivezić, S. Kent, G. R. Knapp, H. Lin, Y.-S. Loh, R. H. Lupton, B. Margon, T. A. McKay, A. Meiksin, J. A. Munn, A. Pope, M. W. Richmond, D. Schlegel, D. P. Schneider, K. Shimasaku, C. Stoughton, M. A. Strauss, M. SubbaRao, A. S. Szalay, I. Szapudi, D. L. Tucker, B. Yanny, and D. G. York. Detection of the Baryon Acoustic Peak in the Large-Scale Correlation Function of SDSS Luminous Red Galaxies. *Astrophysical Journal*, 633: 560–574, November 2005. doi: 10.1086/466512. 4.2.3

I. Z. Emiris and M. I. Karavelas. The predicates of the apollonius diagram: algorithmic analysis and implementation. *Comput. Geom. Theory Appl.*, 33(1-2):18–57, 2006. ISSN 0925-7721. doi: http://dx.doi.org/10.1016/j.comgeo.2004.02.006. 5.2.1, 5.6

S. N. Evans, B. B. Hansen, and P. B. Stark. Minimax expected measure confidence sets for restricted location parameters. *Bernoulli*, 11(4):571–590, 2005. 3.4, 3.4.2, 3.4.3

E. E. Falco, M. J. Kurtz, M. J. Geller, J. P. Huchra, J. Peters, P. Berlind, D. J. Mink, S. P. Tokarz, and B. Elwell. The Updated Zwicky Catalog (UZC). *Publications of the Astronomical Society of the Pacific*, 111:438–452, April 1999. 4.1.3

W. A. Fendt and B. D. Wandelt. Pico: Parameters for the Impatient Cosmologist. *ArXiv Astrophysics e-prints*, June 2006. 4.1.1

R. A. Fisher. Combining independent tests of significance. *The American Statistician*, 2 (5):30, October 1948. 4.2.3

R. A. Fisher. *Statistical Methods for Research Workers*. Oliver and Boyd, London, 4 edition, 1932. 2.4

S. Fortune. A sweepline algorithm for voronoi diagrams. In *SCG '86: Proceedings of the second annual symposium on Computational geometry*, pages 313–322, New York, NY, USA, 1986. ACM. ISBN 0-89791-194-6. doi: http://doi.acm.org/10.1145/10515.10549. 5.2.1

C. Genovese, C. J. Miller, R. C. Nichol, M. Arjunwadkar, and L. Wasserman. Nonparametric inference for the cosmic microwave background. *Statistical Science*, 19(2):308–321, 2004. 3.3, 3.3.1, 3.3.1, 3.3.2, 4.2.1, A

S. Gordon and M. H. Dickinson. Role of calcium in the regulation of mechanical power in insect flight. *PNAS*, 103:4311–4315, 2006. 1

K. M. Gorski, R. Stompor, and R. Juszkiewicz. Cold dark matter and degree-scale cosmic microwave background anisotropy statistics after COBE. *Astrophysical Journal Letters*, 410:L1–L5, June 1993. doi: 10.1086/186865. 4.2.1

P. R. Grant. Polyhedral territories of animals. *American Naturalist*, 102:75–80, 1968. 5.2.1

L. M. Griffiths, J. Silk, and S. Zaroubi. Bumpy power spectra and ΔT/T. *Monthly Notices of the Royal Astronomical Society*, 324:712–716, June 2001. doi: 10.1046/j.1365-8711.2001.04372.x. 4.2.1

C. Guestrin, A. Krause, and A. P. Singh. Near-optimal sensor placements in gaussian processes. In *ICML '05: Proceedings of the 22nd International Conference on Machine learning*, page 265, New York, NY, 2005. ACM Press. 2.1, 2.2.2, 2.2.2

S. Gupta and A. F. Heavens. Fast parameter estimation from the cosmic microwave background power spectrum. *Monthly Notices of the Royal Astronomical Society*, 334:167–172, July 2002. doi: 10.1046/j.1365-8711.2002.05499.x. 3.5, 4.2.1

Antonin Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD '84: Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*. ACM Press, 1984. ISBN 0-89791-128-8. doi: http://doi.acm.org/10.1145/602259.602266. 5.4

P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. In *IEEE Trans. on Systems Science and Cybernetics*, volume 4, pages 100–107. IEEE, 1968. 4.2.1

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. 3.5.2

L. V. Hedges. *Statistical Methods for Meta-Analysis*. Academic Press, 1985. 2.4

D. Higdon, J. Gattiker, B. Williams, and M. Rightley. Computer model calibration using high dimensional output. Technical report, Los Alamos National Laboratory, 2005. 1

G. Hinshaw, D. N. Spergel, L. Verde, R. S. Hill, S. S. Meyer, C. Barnes, C. L. Bennett, M. Halpern, N. Jarosik, A. Kogut, E. Komatsu, M. Limon, L. Page, G. S. Tucker, J. L. Weiland, E. Wollack, and E. L. Wright. First-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: The Angular Power Spectrum. *Astrophysical Journal Supplement Series*, 148:135–159, September 2003. doi: 10.1086/377225. 4.1.1

K. E. Hoff III, J Keyser, M. Lin, D. Manocha, and T. Culver. Fast computation of generalized Voronoi diagrams using graphics hardware. *Computer Graphics*, 33(Annual Conference Series):277–286, 1999. URL `citeseer.ist.psu.edu/hoff99fast.html`. 5.6

C. F. Hoffmann. How to construct the skeleton of csg objects. In A. Bowyer and J. Davenport, editors, *The Mathematics of Surfaces*, volume 4. Oxford University Press, 1990. 5.2.1

H. Inagaki, K. Sugihara, and N. Sugie. Numerically robust incremental algorithm for constructing three-dimensional voronoi diagrams. In *In Proc. 4th Canad. Conf. Comput. Geom.*, pages 334–339, 1992. 5.2.1

M. H. Overmars J. M. Vleugels. Approximating generalized voronoi diagrams in any dimension. Technical Report UU-CS-1995-14, Department of Information and Computing Sciences, Utrecht University, 1995. 5.2.1, 5.6

V. Jeyakumar and A. Rubinov, editors. *Continuous Optimization: Current Trends and Modern Applications*, volume 99 of *Applied Optimization*. Springer, 2005. 5.2

R. Jimenez, L. Verde, H. Peiris, and A. Kosowsky. Fast cosmological parameter estimation from microwave background temperature and polarization power spectra. *Physical Review D*, 70(2):023005–+, July 2004. doi: 10.1103/PhysRevD.70.023005. 3.5, 4.1.1, 4.2.1

N. Kalra, D. Ferguson, and A. Stentz. Incremental reconstruction of generalized voronoi diagrams on grids. In *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS)*, March 2006. 5.2.1

M. Kaplinghat, L. Knox, and C. Skordis. Rapid Calculation of Theoretical Cosmic Microwave Background Angular Power Spectra. *Astrophysical Journal*, 578:665–674, October 2002. doi: 10.1086/342656. 4.1.1

M. I. Karavelas and M. Yvinec. Dynamic additively weighted voronoi diagrams in 2d. In *ESA '02: Proceedings of the 10th Annual European Symposium on Algorithms*, pages 586–598, London, UK, 2002. Springer-Verlag. ISBN 3-540-44180-8. 5.2.1

K. Kersting, C. Plagemann, P. Pfaff, and W. Burgard. Most likely heteroscedastic gaussian process regression. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pages 393–400, New York, NY, USA, 2007. ACM Press. 2.1

L. Knox, N. Christensen, and C. Skordis. The Age of the Universe and the Cosmological Constant Determined from Cosmic Microwave Background Anisotropy Measurements. *Astrophysical Journal Letters*, 563:L95–L98, December 2001. doi: 10.1086/338655. 3.5, 4.2.1

K. Kobayashi and K. Sugihara. Approximation of multiplicatively weighted crystal growth voronoi diagram and its application. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 85(6):1042–967, 2002. 5.2.1

D. Koller, N. Megiddo, and B. von Stengel. Fast algorithms for finding randomized strategies in game trees. In *"Proceedings of the 26th Annual ACM Symposium on the Theory of Computing"*, 1994. 3.4.1

J. M. Kubica, J. Masiero, A. W. Moore, R. Jedicke, and A. J. Connolly. Variable kd-tree algorithms for spatial pattern search and discovery. In *Neural Information Processing Systems*, December 2005. 5.1

S. R. Kulkarni, S. K. Mitter, and J. N. Tsitsiklis. Active learning using arbitrary binary valued queries. *Machine Learning*, 11(1):23–35, 1993. 2.1

F. O. Lehmann and M. H. Dickinson. The control of wing kinematics and flight forces in fruit flies (drosophila spp.). *Journal of Experimental Biology*, 201(3):385–401, 1998. 1

A. Lewis, A. Challinor, and A. Lasenby. Efficient Computation of Cosmic Microwave Background Anisotropies in Closed Friedmann-Robertson-Walker Models. *Astrophysical Journal*, 538:473–476, August 2000. doi: 10.1086/309179. 4.1.1

K. Li, X. L. Gao, and G. Subhash. Effects of cell shape and strut cross-sectional area variations on the elastic properties of three-dimensional open-cell foams. *Journal of the Mechanics and Physics of Solids*, 54(4):783–806, April 2006. 5.2.1

A. Linden and F. Weber. Implementing inner drive by competence reflection. In *In Proceedings of the 2nd International Conference on Simulation of Adaptive Behavior*, Cambridge, MA, 1993. MIT Press. 2.1

D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992. 2.1, 2.2.2

R. Mahkovic. Construction of the generalized voronoi diagram of the unknown environment. In *ISIE '99. Proceedings of the IEEE International Symposium on Industrial Electronics*, volume 3, pages 984–989, Bled, Solvenia, July 1999. 5.2.1

M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000. 2.1

A. McLean, F. Fol Leymarie, and G. Wiggins. Apollonius diagrams and the representation of sounds and music. In *ISVD '07: Proceedings of the 4th International Symposium on Voronoi Diagrams in Science and Engineering*, pages 276–281, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2869-4. doi: http://dx.doi.org/10.1109/ISVD.2007.7. 5.2.1

H. B. McMahan. *Robust Planning in Domains with Stochastic Outcomes, Adversaries, and Partial Observability*. PhD thesis, Carnegie Mellon University, December 2006. 3.4.1, 3.4.1, 3.4.1, 3.4.5, 3.4.5

H. B. McMahan and G. J. Gordon. A fast bundle-based anytime algorithm for poker and other convex games. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007. 3.4.5, 3.4.5

H. B. McMahan, G. J. Gordon, and A. Blum. Planning in the presence of cost functions controlled by an adversary. In *ICML '03: Proceedings of the Twentieth International Conference on Machine Learning*, 2003. 3.4.5

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 1953. 3.5.2

C. J. Miller, R. C. Nichol, C. Genovese, and L. Wasserman. A non-parametric analysis of the cmb power spectrum. *Bulletin of the American Astronomical Society*, 33:1358, December 2001. 4.1.1

A. Moore and J. Schneider. Memory-based stochastic optimization. In D. Touretzky, M. Mozer, and M. Hasselm, editors, *Neural Information Processing Systems 8*, volume 8, pages 1066–1072. MIT Press, 1996. 2.2, 2.5, 5

Ketan Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms*. Prentice-Hall, Inc., New Jersey, 1994. 5.2.1, 5.4, 3, 5.4

G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978. 6.1

J. Neyman and K. Pearson. On the problem of the most efficient test of statistical hypotheses. *Phil. Trans. of Royal Soc. of London*, 231:289–337, 1933. 3.4.2

H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992. 3.4.5

A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, 2nd edition, 2000. 5.2

M. J. Osborne. *An Introduction to Game Theory*. Oxford University Press, USA, August 2003. 3.4.1

D. Overbye. Astronomers report evidence of 'dark energy' splitting the universe. *The New York Times*, July 22 2003. 4.1

T. Padmanabhan and S. K. Sethi. Constraints on $\Omega_{\rm B}$, $\Omega_{\rm M}$, and h from MAXIMA and BOOMERANG. *Astrophysical Journal*, 555:125–129, July 2001. doi: 10.1086/321469. 4.2.1

J. W. Pratt. Length of confidence intervals. *Journal of the American Statistical Association*, 56(295):549–567, September 1961. 3.4.2

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992. 2.2, 3.4.5, 4.1.2, 5.4

N. Ramakrishnan, C. Bailey-Kellogg, S. Tadepalli, and V. N. Pandey. Gaussian processes for active data mining of spatial aggregates. In *Proceedings of the SIAM International Conference on Data Mining*, 2005. 1, 2.2, 2.2.2, 2.2.3, 2.3.2

S. Rebay. Efficient unstructured mesh generation by means of delaunay triangulation and bowyer-watson algorithm. *J. Comput. Phys.*, 106(1):125–138, 1993. ISSN 0021-9991. doi: http://dx.doi.org/10.1006/jcph.1993.1097. 5.2.1

A. G. Riess, L.-G. Strolger, S. Casertano, H. C. Ferguson, B. Mobasher, B. Gold, P. J. Challis, A. V. Filippenko, S. Jha, W. Li, J. Tonry, R. Foley, R. P. Kirshner, M. Dickinson, E. MacDonald, D. Eisenstein, M. Livio, J. Younger, C. Xu, T. Dahlén, and D. Stern. New Hubble Space Telescope Discoveries of Type Ia Supernovae at $z \geq 1$: Narrowing Constraints on the Early Behavior of Dark Energy. *Astrophysical Journal*, 659:98–121, April 2007. doi: 10.1086/510378. 4.1.2

H. P. Robertson. An interpretation of page's "new relativity". *Phyiscal Review*, 49(10): 755–760, May 1936. 4.1.2

W. Saunders, W. J. Sutherland, S. J. Maddox, O. Keeble, S. J. Oliver, M. Rowan-Robinson, R. G. McMahon, G. P. Efstathiou, H. Tadros, S. D. M. White, C. S. Frenk, A. Carramiñana, and M. R. S. Hawkins. The PSCz catalogue. *Monthly Notices of the Royal Astronomical Society*, 317:55–63, September 2000. 4.1.3

C. M. Schafer and P. B. Stark. Using what we know: Inference with physical constraints. In L. Lyons, R. Mount, and R. Reitmeyer, editors, *PHYSTAT 2003: Statiscal Problems in Particle Physics, Astrophysics, and Cosmology*. SLAC, September 2003. 4.2.1

C. M. Schafer and P. B. Stark. Constructing Confidence Sets of Optimal Expected Size. Technical Report 836, Department of Statistics, Carnegie Mellon University, 2006. URL `http://www.stat.cmu.edu/tr/tr836/tr836.html`. 3.4, 3.4.5

J. Schmidhuber and J. Storck. Reinforcement driven information acquisition in nondeterministic environments, 1993. 2.1

R. Scranton, A. J. Connolly, R. C. Nichol, A. Stebbins, I. Szapudi, D. J. Eisenstein, N. Afshordi, T. Budavari, I. Csabai, J. A. Frieman, J. E. Gunn, D. Johnson, Y. Loh, R. H. Lupton, C. J. Miller, E. S. Sheldon, R. S. Sheth, A. S. Szalay, M. Tegmark, and Y. Xu. Physical Evidence for Dark Energy. *ArXiv Astrophysics e-prints*, July 2003. 4.1

U. Seljak and M. Zaldarriaga. A Line-of-Sight Integration Approach to Cosmic Microwave Background Anisotropies. *Astrophysical Journal*, 469:437–+, October 1996. 4.1.1

U. Seljak, N. Sugiyama, M. White, and M. Zaldarriaga. Comparison of cosmological boltzmann codes: Are we ready for high precision cosmology? *Phys. Rev. D*, 68(8): 083507, October 2003. doi: 10.1103/PhysRevD.68.083507. 4.1.1

M. I. Shamos and D. Hoey. Closest-point problems. In *In Proceedings of the 16th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 151–162, 1975. 5.2.1, 5.4, 5.4

S. A. Shectman, S. D. Landy, A. Oemler, D. L. Tucker, H. Lin, R. P. Kirshner, and P. L. Schechter. The Las Campanas Redshift Survey. *Astrophysical Journal*, 470:172–+, October 1996. doi: 10.1086/177858. 4.1.3

P. Sollich. Query construction, entropy, and generalization in neural-network models. *Physical Review E*, 49:4637–4651, May 1994. doi: 10.1103/PhysRevE.49.4637. 2.1

D. N. Spergel, L. Verde, H. V. Peiris, E. Komatsu, M. R. Nolta, C. L. Bennett, M. Halpern, G. Hinshaw, N. Jarosik, A. Kogut, M. Limon, S. S. Meyer, L. Page, G. S. Tucker, J. L. Weiland, E. Wollack, and E. L. Wright. First-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Determination of Cosmological Parameters. *Astrophysical Journal Supplement Series*, 148:175–194, September 2003. doi: 10.1086/377226. 3.5, 4.1.1, 4.2.1, 4.2.1, 4.4, 4.2, 4.2.1, 4.2.1

D. N. Spergel, R. Bean, O. Doré, M. R. Nolta, C. L. Bennett, J. Dunkley, G. Hinshaw, N. Jarosik, E. Komatsu, L. Page, H. V. Peiris, L. Verde, M. Halpern, R. S. Hill, A. Kogut, M. Limon, S. S. Meyer, N. Odegard, G. S. Tucker, J. L. Weiland, E. Wollack, and E. L. Wright. Three-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Implications for Cosmology. *Astrophysical Journal Supplement Series*, 170:377–408, June 2007. doi: 10.1086/513700. (document), 1, 4.1, 4.1.1, 4.1.1, 4.2.1, 4.2.3, 4.2.3, 4.6

A. Stein, F. van der Meer, and B. Gorte, editors. *Spatial Statistics for Remote Sensing*. Kluwer Academic Publishers, Boston, 1999. 1, 2.2

K. Sugihara and M. Iri. A robust topology-oriented incremental algorithm for voronoi diagrams. *Int. J. Comput. Geometry Appl.*, 4(2):179–228, 1994. 5.2.1

K. K. Sung and P. Niyogi. Active learning for function approximation. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 593–600. The MIT Press, 1995. 2.1

B. Tang. Orthogonal array-based latin hypercubes. *Journal of the American Statistical Association*, 88(424):1392–1397, December 1993. 2

M. Tegmark. Cosmological Constraints from Current Cosmic Microwave Background and Type IA Supernova Data: A Brute Force, Eight-Parameter Analysis. *Astrophysical Journal Letters*, 514:L69–L72, April 1999. doi: 10.1086/311943. 4.1.1

M. Tegmark, M. Zaldarriaga, and A. J. Hamilton. Towards a refined cosmic concordance model: Joint 11-parameter constraints from the cosmic microwave background and

large-scale structure. *Physical Review D*, 63(4), February 2001. 1, 3.5.1, 4.1, 4.1, 4.1.1, 4.1.1, 4.2.1, 4.4, 4.2.1, 4.2.1, 4.2.3

M. Tegmark, M. A. Strauss, M. R. Blanton, K. Abazajian, S. Dodelson, H. Sandvik, X. Wang, D. H. Weinberg, I. Zehavi, N. A. Bahcall, F. Hoyle, D. Schlegel, R. Scoccimarro, M. S. Vogeley, A. Berlind, T. Budavari, A. Connolly, D. J. Eisenstein, D. Finkbeiner, J. A. Frieman, J. E. Gunn, L. Hui, B. Jain, D. Johnston, S. Kent, H. Lin, R. Nakajima, R. C. Nichol, J. P. Ostriker, A. Pope, R. Scranton, U. Seljak, R. K. Sheth, A. Stebbins, A. S. Szalay, I. Szapudi, Y. Xu, J. Annis, J. Brinkmann, S. Burles, F. J. Castander, I. Csabai, J. Loveday, M. Doi, M. Fukugita, B. Gillespie, G. Hennessy, D. W. Hogg, Ž. Ivezić, G. R. Knapp, D. Q. Lamb, B. C. Lee, R. H. Lupton, T. A. McKay, P. Kunszt, J. A. Munn, L. O'Connell, J. Peoples, J. R. Pier, M. Richmond, C. Rockosi, D. P. Schneider, C. Stoughton, D. L. Tucker, D. E. vanden Berk, B. Yanny, and D. G. York. Cosmological parameters from SDSS and WMAP. *Physical Review D*, 69(10): 103501–+, May 2004. doi: 10.1103/PhysRevD.69.103501. 4.1.1

M. Tegmark, D. J. Eisenstein, M. A. Strauss, D. H. Weinberg, M. R. Blanton, J. A. Frieman, M. Fukugita, J. E. Gunn, A. J. S. Hamilton, G. R. Knapp, R. C. Nichol, J. P. Ostriker, N. Padmanabhan, W. J. Percival, D. J. Schlegel, D. P. Schneider, R. Scoccimarro, U. Seljak, H.-J. Seo, M. Swanson, A. S. Szalay, M. S. Vogeley, J. Yoo, I. Zehavi, K. Abazajian, S. F. Anderson, J. Annis, N. A. Bahcall, B. Bassett, A. Berlind, J. Brinkmann, T. Budavari, F. Castander, A. Connolly, I. Csabai, M. Doi, D. P. Finkbeiner, B. Gillespie, K. Glazebrook, G. S. Hennessy, D. W. Hogg, Ž. Ivezić, B. Jain, D. Johnston, S. Kent, D. Q. Lamb, B. C. Lee, H. Lin, J. Loveday, R. H. Lupton, J. A. Munn, K. Pan, C. Park, J. Peoples, J. R. Pier, A. Pope, M. Richmond, C. Rockosi, R. Scranton, R. K. Sheth, A. Stebbins, C. Stoughton, I. Szapudi, D. L. Tucker, D. E. V. Berk, B. Yanny, and D. G. York. Cosmological constraints from the SDSS luminous red galaxies. *Physical Review D*, 74(12):123507–+, December 2006. doi: 10.1103/PhysRevD.74.123507. 4.1, 4.1.3, 4.3, 4.2.3, 4.2.3, 4.23(c), 4.5, 4.2.3

S. B. Thrun and K. Möller. Active exploration in dynamic environments. In *Advances in Neural Information Processing Systems*, volume 4, pages 531–538. Morgan Kaufmann Publishers, Inc., 1992. 2.1

S. Tong and D. Koller. Active learning for parameter estimation in bayesian networks. In *Advances in Neural Information Processing Systems*, pages 647–653, 2000. URL `citeseer.ist.psu.edu/tong01active.html`. 2.1

L. Verde, H. V. Peiris, D. N. Spergel, M. R. Nolta, C. L. Bennett, M. Halpern, G. Hinshaw, N. Jarosik, A. Kogut, M. Limon, S. S. Meyer, L. Page, G. S. Tucker, E. Wollack, and

E. L. Wright. First-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Parameter Estimation Methodology. *Astrophysical Journal Supplement Series*, 148:195, September 2003. doi: 10.1086/377335. 4.1.1, 4.2.3, 4.2.3

J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944. 3.4.1

L. Wasserman. *All of Statistics*. Springer-Verlag, New York, 2004. 3.2, 3.6.1, 3.6.2

D. F. Watson. Computing the n-dimensional delaunay tesselation with application to voronoi polytopes. *The Computer Journal*, 24(2):167–172, 1981. 5.6

M. White and E. F. Bunn. The COBE Normalization of CMB Anisotropies. *Astrophysical Journal*, 450:477–+, September 1995. doi: 10.1086/176158. 4.2.1

S. Whitehead. A study of cooperative mechanisms for faster reinforcement learning. Technical report, University of Rochester, Rochester, NY, 1991. 2.1

J. A. Wiens. An approach to the study of ecological relationships among grassland birds. *Ornithological Monographs No. 8.*, pages 1–93, 1969. 5.2.1

W. M. Wood-Vasey, G. Miknaitis, C. W. Stubbs, S. Jha, A. G. Riess, P. M. Garnavich, R. P. Kirshner, C. Aguilera, A. C. Becker, J. W. Blackman, S. Blondin, P. Challis, A. Clocchiatti, A. Conley, R. Covarrubias, T. M. Davis, A. V. Filippenko, R. J. Foley, A. Garg, M. Hicken, K. Krisciunas, B. Leibundgut, W. Li, T. Matheson, A. Miceli, G. Narayan, G. Pignata, J. L. Prieto, A. Rest, M. E. Salvo, B. P. Schmidt, R. C. Smith, J. Sollerman, J. Spyromilio, J. L. Tonry, N. B. Suntzeff, and A. Zenteno. Observational Constraints on the Nature of Dark Energy: First Cosmological Results from the ESSENCE Supernova Survey. *Astrophysical Journal*, 666:694–715, September 2007. doi: 10.1086/518642. 4.1.2

A. L. Zanin, A. .W. Liehr, A. S. Moskalenko, and H. G. Purwins. Voronoi diagrams in barrier gas discharge. *Applied Physics Letters*, 81(18):3338–3340, October 2002. 5.2.1

Y. Zhao, F. E. H. Tay, F. S. Chau, and G. Zhou. A nonlinearity compensation approach based on delaunay triangulation to linearize the scanning field of dual-axis micromirror. *Journal of Micromechanics and Microengineering*, 15:1972–1978, 2005. 5.2.1

F. Zwicky. On the Masses of Nebulae and of Clusters of Nebulae. *Astrophysical Journal*, 86:217–+, October 1937. 4.1

Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Carnegie Mellon