HEURISTIC OPTIMISATION AND PARAMETER ESTIMATION METHODS FOR MODERN COSMOLOGICAL SURVEYS

E. C. ELSON

Thesis presented to the Faculty of Science
at the University of Cape Town
for the Degree of M.Sc.

Supervisors:

Assoc. Prof. Bruce Bassett[1]

Dr. Kurt van der Heyden[2]

Dr. Zeblon Vilakazi[3]

Applied Maths Department

UNIVERSITY OF CAPE TOWN

June 2006

[1]Applied Maths Department, University of Cape Town/South African Astronomical Observatory
[2]South African Astronomical Observatory
[3]Department of Physics, University of Cape Town/iThemba Labs

Abstract

This thesis deals with two important aspects of modern cosmology. Part 1 involves the testing and implementation of a new optimisation scheme named HYBRID. The new method is developed as part of this thesis and combines aspects of Monte Carlo Markov chain optimisation, simulated annealing and particle swarm optimisation. HYBRID is used to address the problem of selecting an optimal subset of targets from a larger astronomical catalog which, for massive current and future cosmological surveys, is becoming an increasingly important problem to solve. We find that HYBRID consistently outperforms the standard optimisation methods in terms of best results obtained, speed of convergence to those results and reliability. The method is found to perform extremely well in high dimensionality spaces with multiple characteristic curvature scales, implying significant advantages in solving complex optimisation problems. The results from part 1 have been submitted for publication as 'A New Approach to the Optimal Target Selection Problem' by E. Elson, B. Bassett, K. van der Heyden, and Z. Vilakazi, *astro-ph*/0602338, submitted to Astronomy and Astrophysics (2005).

Part 2 of the thesis deals predominantly with Monte Carlo Markov Chain parameter estimation as a means of extracting cosmological parameter estimates from type Ia supernova data. Specifically, the aim is to develop and test an analysis pipeline that may be used on the data of the SDSS-II supernova survey. The pipeline is tested on the Supernova Legacy Survey first-year data to estimate Ω_M, H_0 and w_0. We find that the results are broadly consistent with those in the literature. Finally, at the end of part 2 we carry out an investigation into the intrinsic error associated with the Monte Carlo Markov Chain parameter estimation method itself. The results we obtain verify that for simple likelihood surfaces the intrinsic error is negligibly small thereby validating the use of Monte Carlo Markov Chains for parameter estimation in these cases.

The pipeline developed in part 2 will be applied to year 1 of the SDSS supernova survey data once it has been released to the SDSS-II collaboration. In addition to the work presented in this thesis the author was a hand scanner for the first year (September to November 2005) of the SDSS-II survey. The first season of the survey lead to the discovery of 128 new type Ia supernovae.

Acknowledgments

Special thanks are extended first and foremost to my supervisors Bruce Bassett, Kurt van der Heyden and Zeblon Vilakazi for all their help, support and advice throughout the last year. I would like to acknowledge the University of Cape Town Department of Physics CERN cluster for use of their facilities for some of the numerical work in both the thesis and paper, specifically Gareth De Vaux for helping with the problems that occurred. The South African Astronomical Observatory is thanked for welcoming me as a masters student and allowing me to use their facilities. I am grateful also to the National Astrophysics and Space Science Programme for accepting me as part of their M.Sc. program and for the funding received from the Andrew W. Mellon Foundation and the National Research Foundation. Finally, very special thanks are in order for my family who have always encouraged me and provided me with everything that I need as well as my friends who force me to always keep one foot in the real world.

Contents

List of Figures

List of Tables

Part I

Optimisation Techniques

Chapter 1

Optimisation Introduction

This thesis focuses on methods of optimisation and data analysis for modern cosmological surveys. Recent years have seen a revolution in cosmology. The field is becoming a precision science very much due to modern cosmological surveys producing vast amounts of excellent data with which to test theoretical models and predictions. Surveys such as the Sloan Digital Sky Survey (SDSS) [5] and the 2dF Galaxy Redshift Survey [6] helped us quantify the large scale structure of galaxies and galaxy clusters. These two surveys alone have greatly furthered our understanding of structure formation and evolution in the Universe.

Of course, where would cosmology be without Cosmic Microwave Background (CMB) experiments such as the Wilkinson Microwave Anisotropy Probe (WMAP)? WMAP [7] single-handedly provided us with some of the most precise constraints on cosmological parameters to date [8, 9]. The experiment was able to the map the CMB at very small angular scales, exposing temperature fluctuations as small as one part in 10^5 [10]. Polarization analysis of the CMB data [11, 12] will allow us to further constrain the initial conditions and the nature of primordial density fluctuations. The way cosmologists perform this analysis is by using complex statistical techniques [13, 14] to extract from the data the information they require.

SDSS and WMAP are perhaps two of the most successful astronomical experiments of the last decade or so. Many future surveys are currently being planned. Surveys such as WFMOS (Wide-Field Multi-Object Spectrograph) [15] will obtain the spectra of thousands of galaxies so that we may further probe the dynamics of the mysterious dark energy. ALPACA (Advanced Liquid-mirror Probe for Astrophysics, Cosmology and Asteroids) [16], a proposed 8m liquid telescope surveying $\sim$ 1000 square degrees of the sky will provide multi-wavelength light curve data for approximately 10^5 supernovae per year as well as detect baryon acoustic oscillations. Other future cosmological surveys include the Large Synoptic Survey Telescope (LSST[1]), the Dark Energy Survey (DES[2]), the Large Sky Area Multi-Object Fiber Spectroscopic Telescope (LAMOST[3]), and numerous others. The point to be made is that current and future cosmological surveys require good, tested optimisation techniques in order to maximise their scientific output. WFMOS, as an example, will obtain thousands of spectra with each exposure. Exactly where to place each of the optical fibres so as to maximise the number of spectra obtained for a particular field of view is an optimisation problem, the type of problem that this thesis deals with.

November 2005 saw the inauguration of the new Southern African Large Telescope (SALT[4]). The telescope promises to elevate Southern African astronomy to world-leading standards, providing us

[1] http://www.lsst.org/lsst_home.shtml
[2] http://www.darkenergysurvey.org/
[3] http://www.lamost.org/en/
[4] http://www.salt.ac.za/

with all the scientific benefits of a 10 metre class telescope. Already some cosmological projects have been proposed, one of them being SCALPEL (SALT Cosmic Ages of Luminous Passive ELipticals Survey). The SCALPEL survey will obtain more than 1000 high signal to noise spectra to constrain the expansion rate of the Universe at a redshift of $z = 0.5$ to within 3%. SCALPEL will achieve this by utilizing one of the most promising methods of constraining dark energy: the method of age-dating luminous red galaxies, (LRGs) [17]. Typically LRGs are very massive, passive ellipticals sitting in clusters. The key idea of this Cosmic Chronometers [18] method is that the difference in cosmic age between two nearby redshifts is determined completely by the Hubble expansion rate at that redshift. Hence, if we know the age difference we know the Hubble expansion rate. The method has been recently tested [19], using SDSS data to produce an independent estimate of the Hubble constant today. The key problem in using Cosmic Chronometers to make the transition to the next level in precision constraints on dark energy is (as always) the need to understand the systematics associated with the problem. In this case we need to understand how LRG ages depend on environment and we need many independent estimates of the Hubble rate at each redshift. With limited observing time, optimal targets need to be pre-selected for spectroscopic follow-up so that the best results may be obtained from the survey.

Modern cosmological surveys clearly present the need for effective and efficient data analysis methods. Gone are the days when a cosmologist had a few hundred data points to which to fit a curve or from which to select an optimal subset. Current day surveys produce huge multi-dimensional data sets which need to be analysed with the aid of computers. The original aim of this project was to deal with the problem of optimal target selection given a catalog of thousands of objects, each with certain information, from which to choose. This type of problem is common. A general example of allocation of limited resources would be a city with a restricted power supply. Being forced to cut the power to certain areas, what are the best regions to leave powered up so that the most regular traffic flows may be maintained? Typically these real world problems are difficult to even formulate. Having formulated the predicament there is then the further task of actually producing a solution. This is where computers prove to be invaluable.

The first part of this project is ultimately motivated by the need for effective and efficient optimisation techniques which may be applied to a variety of problems. The specific optimal target selection problem addressed by this dissertation may be stated as follows: *Given right ascension and declination information for many objects, what are the best regions to place N fields of view (FoV) of fixed size so as to maximise the total number of objects observed?*. More generally, each catalog entry may have associated with it a figure of merit (FoM) which serves as a type of quality or weighting factor. The aim then is to optimally position the N FoV so as to maximise or minimise an objective function that depends on all the FoM. This optimal target selection problem is one of increasing importance in the planning of large future surveys. Given a catalog of say 10^5 objects, it is an impossible task to pick out, by eye, the optimal regions. Part 1 of this dissertation involves the development, testing and implementation of a new heuristic algorithm to try to find the optimal FoV. Specifically, the technique is a hybrid of Monte Carlo Markov Chain (MCMC) optimisation, simulated annealing (SA) and particle swarm optimisation (PSO) methods.

As an attempt to further probe the performance of the new method, hereafter called HYBRID, it is further tested on continuous functions, the minima of which standard methods struggle to locate. Standard methods often become stuck in local minima and fail to find the global minimum of interest. It is on these continuous functions that HYBRID really proves its worth by locating quickly and effectively regions very close to the global minimum.

As far as part 1 of this thesis is concerned, in chapter 2 the standard optimisation methods that make up HYBRID are explained. More precisely the particular aspects of all the methods that are used are listed and made clear. Thereafter, in chapter 3, the basic outline of the new method is described: the individual controlling functions as well as transition probabilities etc. Once the new technique has been discussed, the testing procedure is worked through. The method is tested both on discrete data sets as well as continuous functions, a chapter, including the results, being devoted

to each. The underlying theme of this project is that of modern cosmological surveys and so the second half of this dissertation, part 2, is centered around one particular such method–Monte Carlo Markov Chain parameter estimation in a maximum likelihood context. A separate introduction for this MCMC parameter estimation is included at the beginning of part 2.

Chapter 2

Standard Optimisation Methods

This chapter deals with standard optimisation methods. The basic underlying principles of each method are highlighted so that they may be easily referred to when describing the HYBRID algorithm in the next chapter. The general problem of having to optimally place N Fields of View (FoV) in order to maximise the number of objects within those FoV is non-trivial and depends very much on certain characteristics of the input catalog.

One may distinguish between deterministic and heuristic approaches. A deterministic method will produce exactly the same results for various runs on a particular data set provided that the same initial conditions and parameters are used for each run. An example would be the Newton-Raphson method [20] in which a function $f(x)$ is approximated by its tangent lines in order to iteratively locate its root. These deterministic methods often produce good results but at the expense of much computation time. A heuristic approach, on the other hand, incorporates certain random elements in order to find good solutions although there is no guarantee that they might be the best. The gain however is in computation time–the runs are quite fast. Out of the methods to be described below, the first might be labeled as deterministic while the others are heuristic.

2.1 The Grid Approach

As a starting point, a simple deterministic method against which all other heuristic techniques could be benchmarked was implemented. The first method came to be referred to as the grid method and was initially applied (as were the other methods) to a two-dimensional discrete data set, i.e. a collection of points in a plane. *It should be made clear to the reader* that in the end the grid method was actually discarded and only other heuristic methods were investigated thoroughly. The reasons for this are explained below. Nevertheless, the first part of this chapter describes the basic grid procedure that was thought out.

The basic idea is simple: Given a bounded two-dimensional data set, split the x-range into m equal parts and the y-range into n equal parts so that a grid of $m \times n$ elements is essentially superimposed on the data set. The procedure then is to simply locate the N most densely populated grid elements within which to place the centres of the fields of view[1]. To be more precise, a pseudo-code version of the entire program would be as follows:

1. Read in the data and determine the upper and lower bounds in the x and y directions. Call these x_{max}, y_{max}, x_{min}, y_{min}.

[1]Recall that the aim is to maximise the number of objects that are contained within N separately placed FoV.

2. Obtain from the user the values of m and n. The area of a single grid element will then be $dx \times dy$ where $dx = (x_{max} - x_{min})/m$ and $dy = (y_{max} - y_{min})/n$.
3. Place the grid over the data set and then count the number of points in each grid element.
4. Rank the grid elements according to the number of points in each one.
5. Having located the N best grid elements, calculate the average (x, y) position of all the points within a grid element and then place the centre of a FoV at that location.
6. Count the number of points in each FoV remembering not to double count points that may be shared by more than one FoV.

The method is extremely inelegant and inefficient in many ways. It uses brute force more than anything else. The computation time is proportional to the number of grid elements and usually proportional to the square of the number of objects and hence is unfeasible in general for very large data sets (which is the region we are mainly interested in). Furthermore, splitting the data box up into a grid proves to be very problematic. Firstly, the optimal number of grid elements to use is not obvious, it depends very much on the particular data set. A finer grid mesh is not always better. It might be the case that for a certain number of grid elements, a cluster small enough to fit into one grid element is shared evenly between a few grid elements. The over-density of this cluster will then not be as apparent as it would be if it were fortunate enough to lie only in one grid element. The grid method is also completely deterministic. It was decided, in the end, not to significantly develop and test the method and to simply move onto the heuristic algorithms. An improvement to this simple grid approach would be to make use of an adaptive grid in which the general grid method is iteratively applied to regions identified to be of interest, i.e. dense. This approach was also not investigated.

2.2 Monte Carlo Markov Chains

2.2.1 The Markov Process

A Markov chain [21] is the first heuristic method that we shall consider. It is a stochastic process meaning that a series of selections are made based upon numbers that are randomly extracted or determined from a probability distribution[2]. Furthermore, a Markov chain may be described as a type of random walk based upon the Markov process. In order to properly define the Markov process [22, 23], we should first lay down some basic nomenclature. All we really need is the idea that a system can be in a particular state at a particular time. We may speak, for example, of a system being in a state S_j at time t_j. Previous states of the system would be represented as S_{j-1}, S_{j-2}, ..., S_{j-n} and would correspond to times t_{j-1}, t_{j-2}, ..., t_{j-n} respectively. Similarly, future system states would be S_{j+1}, S_{j+2}, ..., S_{j+n} and would correspond, respectively, to times t_{j+1}, t_{j+2}, ..., t_{j+n}. With this terminology in mind, the Markov process simply states that the probability of the system being in state S_j at time t_j depends only on the state S_{j-1} that it was in at time t_{j-1}. In other words, the transition from one state to another depends only on the system's current state and is unaffected by the previous history.

Having defined the Markov process, a Markov chain is then simple to understand. A Markov chain is simply a random walk incorporating the Markov process at each step. Specifically, the transition probability with which the $j - th$ step in the random walk is taken depends only on the position of the chain at the previous step. One may speak of p_{mn} as being the probability that the system, which at time t_j is in the state S_m, will be in the state S_n at time t_{j+1}. The set of all these transition probabilities then forms a matrix which completely determines the properties of the chain.

[2]Given the probability distribution $f(x)$ for a variable x, the probability that $x_* < x < x_* + dx$ is $P(x_* < x < x_* + dx) = f(x)dx$.

2.2.2 The Metropolis-Hastings Transition Condition

Usually the transition probability with which a state change of the system is accepted is governed by the Metropolis-Hastings condition [22]. This condition simply states that the system should always make a transition that improves the objective function. The objective function is simply a function that we are attempting to optimise and which depends on the current and immediately previous system states. A bad step that might worsen the objective function is accepted with some, usually exponentially small, probability.

As a simple example, suppose we want to find the maximum of some one-dimensional function, $a(x)$. Given x_j, we should always accept a change Δx in x if $a(x_{j+1}) \geq a(x_j)$, where $x_{j+1} = x_j + \Delta x$. If $a(x_{j+1}) < a(x_j)$ then we should allow the move from x_j to x_{j+1} with probability

$$P \propto \exp(\Delta a), \tag{2.1}$$

for example. The reasoning behind accepting a small number of bad steps is to prevent a walker or a chain from getting stuck in local minima.

2.2.3 The Monte Carlo Component

So far, we have described what a Markov chain is and explained how, in many cases, the transition probability between system states is determined by applying the Metropolis-Hastings algorithm. Our only remaining task now is to explain how the Monte Carlo component is incorporated into the entire Markov chain. The answer to this question is quite simple–it is merely the random component of the process.

A Monte Carlo method is basically one which involves using randomly generated numbers to simulate some process. In terms of a random walker, such as a Markov chain, random increments in various dimensions need to be produced. The way this is usually done is to extract the random change, Δx_i in the $i-th$ dimension, from some distribution which is usually a Gaussian and then once all the increments in all the dimensions have been produced, the overall change is $\vec{\Delta x} = (\Delta x^1, \Delta x^2, ..., \Delta x^k)$, where k is the dimensionality of the space. It then needs to be decided whether or not to actually accept the step from x_j to $x_{j+1} = x_j + \Delta x$. What the reader should basically understand from all this is that *random* steps (and hence the Monte Carlo component) are taken between system states so that the entire process is stochastic rather than deterministic.

2.3 Simulated Annealing

The simulated annealing [24, 25] optimisation process is based upon the idea of thermal annealing in which a physical system is cooled adiabatically so that it becomes increasingly ordered with decreasing temperature so that at temperature T=0 it attains its lowest energy configuration. In terms of an optimisation problem, what one does is to generate a good current solution to the problem and then improve that solution by considering random variations of it. A worse solution is accepted with a probability which systematically decreases as the computation proceeds. This decreasing acceptance probability for worse solutions is called the 'cooling schedule' and the slower it is, the more likely it is that an optimal solution is reached. Typical cooling schedules vary logarithmically with step number. Basically the 'temperature' is made high in the early stages of the process for faster optimisation, then is reduced (according to the cooling schedule) for greater stability at later times. The method guarantees convergence to the optimal solution given an infinite amount of time.

If the reader thinks in terms of walkers exploring a space then the cooling schedule gradually relaxes all the walkers so that at later times they are all attempting small steps which should allow

then to scout their local extrema. Of course, there is the problem of the average step size being small even when the walkers are trapped in poor regions. For this reason the system is usually re-annealed regularly throughout the simulation. The simulated annealing approach to optimisation plays an important role in solving various types of astronomical problems [26].

2.4 Particle Swarm Optimisation

Briefly, particle swarm optimisation (PSO) is an optimisation process in which multiple walkers share information as they each execute a random walk. At each step the walkers change their position based on information from their own history, as well as information from the history of the other walkers. This sharing of information as the search is carried out allows local search methods to be based on more global results.

The method was developed by Eberhart and Kennedy [27] in an attempt to simulate the behaviour of flocks of birds. Individual walkers are usually called particles while collectively they are called a swarm. In an astronomical context, a particle could be called a FoV. In the general version of the scheme each particle has associated with it a position and a velocity. The velocities of the particles are then updated according to some objective function, the most successful particles 'attracting' the others toward them. Any change in velocity is obviously accompanied by an update in the particle's position. This process of updating velocities and positions according to the optimisation of some objective function is repeated until some performance criterion is met. For more information the reader is referred to [28, 29, 30, 31]

Chapter 3

HYBRID–A New Optimisation Algorithm

Having introduced the standard optimisation methods upon which HYBRID is based, this chapter provides a basic sketch of the new method. An outline is offered explaining how the standard methods are essentially special subcases of HYBRID. Thereafter, all the specific details of the way in which a HYBRID walker takes a step are discussed.

Before we proceed, let us set up some more specific nomenclature that will stream-line our discussions and let us clearly state the task to be carried out. Recall that the first task that we are concerned with is that of optimal target selection. Given a catalog of objects we need to to decide how to optimally position N fields of view so as to maximise the total number of objects observed, which serves as our objective function. For this reason, the words 'walker' and 'FoV' will be used interchangeable throughout this report[1]. FoM_j^i will be used to refer to the figure of merit of the $i-th$ FoV at the $j-th$ step in its chain. FoM_j, without any superscript, will then refer to the sum over i of the FoM of all the FoV:

$$FoM_j = \sum_{i=1}^{N} FoM_j^i. \tag{3.1}$$

This important quantity, the sum of the figures of merit of all the fields of view at step j of the simulation, will often be referred to as the survey count at step j.

3.1 Basic Outline

At the outset N FoV are randomly placed within the data set. Since we are dealing with N FoV moving simultaneously at each step in the simulation, overlapping of FoV becomes a problem and should be prevented to a certain extent, otherwise all FoV converge to a single dense region whereas we are interested in locating the N densest regions. One way in which this problem may be resolved is to make the objective function, which is to be maximised, decrease when overlapping occurs. When one of the N FoV needs to take a step the basic procedure is as follows:

[1] FoV is simply the astrophysical adaptation of a walker

1. Evaluate the current system state.

2. Extract, from a Gaussian, an increment for each of the dimensions of the space being explored (two-dimensional in our case).

3. Construct the step to be attempted.

4. Either accept or reject the step based on the comparison between the current system state and the proposed new system state.

What makes HYBRID different from standard MCMC [21] is the way in which a FoV takes a step. When a particular FoV is required to take a step, information from the other $N-1$ FoV is gathered to help decide upon a reasonable step to take. This makes up the particle swarm component of the HYBRID method which is necessary when dealing with data sets that have multiple clustering length scales. Let the radius of a FoV be R. If L_0 is the characteristic length scale of the data set and l_0 the characteristic length scale of a cluster, then the ratios R/L_0 and R/l_0 are crucial in deciding which method to use. When it comes to standard MCMC, the average step size of a walker should be chosen in such a way so that it may efficiently probe the large and small clustering scales of the data set. HYBRID eliminates the task of having to choose one length scale to focus on since it's heuristic in the sense that FoV evaluate their individual performance relative to the average of the others and then, based on that ratio, decide what size step to take. For the sake of further discussion we shall define X_j^i as the ratio of FoM_j^i to the average of the other FoM:

$$X_j^i \equiv \frac{FoM_j^i}{\sum_{k\neq i}^{N} FoM_j^k}. \tag{3.2}$$

The general performance of the entire system is also monitored to force the FoV to stay in or nearby regions which produce a good survey count. This is done by introducing a cooling schedule which steadily decreases the number of accepted bad steps as the simulation proceeds as well as the average step size of all the FoV as the objective function is optimised, thereby introducing a simulated annealing aspect to the entire method. The particle swarm and simulated annealing aspects of the method are controlled by the functions f and g respectively which, as we shall see, determine the variance of the Gaussian from which the increments in the various dimensions for a particular FoV at a particular step are extracted.

Having mentioned the functions f and g, we specify the variance of the distribution from which an increment in the $u-th$ dimension is chosen as

$$\sigma^u = \sigma_0^u \times f \times g, \tag{3.3}$$

where σ_0^u is constant. One may then identify four different cases:

1. **Case 1:** $f = g = 1$. When f and g are both 1, they have no effect and so the method reduces to that of standard MCMC.

2. **Case 2:** $f \neq 1$, $g = 1$. When f contributes, the particle swarm component of HYBRID comes into affect, allowing better regions to be located faster.

3. **Case 3:** $f = 1$, $g \neq 1$. As mentioned earlier, a cooling schedule g should force the FoV to 'stick' to regions, once found, that maximise the objective function.

4. **Case 4:** $f \neq 1$, $g \neq 1$. This is the case which should produce the best results since both the particle swarm and simulated annealing features are utilised. The results should be better both in terms of survey count as well as efficiency and variability of results.

3.2 Transition Probability

In chapter 2 the fundamental principles of a Markov chain were discussed. In particular it was mentioned that the usual transition probability between system states is governed by the Metropolis-Hastings algorithm [22]. HYBRID is no different in this respect.

Suppose that at step j in its Markov chain, FoV^i attempts a step. The step to be attempted may either increase or decrease (or possibly not affect) FoM^i_j. If $FoM^i_{j+1} \geq FoM^i_j$ then FoV^i should definitely take the step. When it is known that $FoM^i_{j+1} < FoM^i_j$, the step is accepted with probability

$$P\left(FoM_{j+1}|FoM_j\right) = \exp\left[-\alpha(FoM_j - FoM_{j+1})\right]. \tag{3.4}$$

α is simply the parameter that allows the user to fine tune the chance of accepting a bad step. Also, recall that omission of the superscript implies summation over that index, see eqn. 3.1.

3.3 Choosing a step.

This subsection explains in detail exactly how a step for a FoV to attempt is selected. Remember that in general a FoV can be used to explore an m-dimensional space. For this reason we speak of deciding upon an increment for each of the m dimensions and then combining them to produce the resultant step. For example, suppose that $\vec{x}_j$ represents the position of a particular FoV at step j in its chain. A FoV could then move from $\vec{x}_j = (x^1_j, x^2_j, ..., x^m_j)$ to $\vec{x}_{j+1} = (x^1_{j+1}, x^2_{j+1}, ..., x^m_{j+1}) = (x^1_j + \Delta x^1, x^2_j + \Delta x^2, ..., x^m_j + \Delta x^m)$. It is these Δx^1, Δx^2, ..., Δx^m that are drawn out of an m-dimensional Gaussian. The details of calculating a step are then as follows:

1. Suppose that the $i-th$ FoV is at step j in its Markov chain. The ratio $X^i_j \equiv FoM^i_j / \sum^N_{k\neq i} FoM^k_j$ is calculated. This is a crucial quantity since if the ratio is greater than 1 then we know that FoV^i_j is in a region which is on average more dense and it should take small steps to ensure that it does not move too far away, thereby properly 'exploring' the overdense region. Conversely, if the ratio is less than 1 then FoV^i_j is in a relatively underdense region and should take larger steps so that it may quickly move to a more optimal region which is of more interest to the survey.

2. At this stage a subroutine extracts a random number from a Gaussian distribution by performing the appropriate transformation on two uniform deviates supplied by a random number generator[2]. This number represents Δx^u where u =1, 2, ..., m. The variance of the Gaussian from which Δx^u is chosen is

$$\sigma^u = \sigma^u_0 \times f(X^i_j) \times g(FoM_j) \tag{3.5}$$

where σ^u_0 is the constant variance of the distribution associated with dimension u. In the case where f and g do not contribute the method reduces to that of standard MCMC. In general σ^u_0 is a type of scaling factor that can be different for each of the dimensions of the space. Typically, σ^u_0 is made to be a fixed fraction of the size of the data set or input catalog in dimension u.

The function g is the 'cooling' function that 'calms' the system down as FoM_j increases with j. In our simulations the typical functional form for g is

$$g = \left(\frac{FoM_1}{FoM_j}\right)^t. \tag{3.6}$$

g is therefore the ratio of the sum of the FoM at step 1 to the current sum at step j, all to some power t. Thus, as the survey count improves with time, the step sizes of all the FoV

[2]Given uniform deviates u and v, $w = (-2\log u)^{1/2}\cos(2\pi v)$ is normally distributed.

should steadily decrease. In many applications of simulated annealing the cooling schedules are typically logarithmic [24]. The problem with such a cooling schedule is that it depends only on step number, decreasing monotonically, and is therefore insensitive to the current performance.

Function f is the important function that should penalise a particular FoV if it is in a relatively underdense region by forcing it to take a big step. This 'performance' function should also reward a FoV for being in a relatively overdense region by allowing it to take a small step. It is the functional form of f which proves to be tricky to determine or decide upon. f is a function of the above mentioned ratio X_j^i since it is this ratio which communicates to the algorithm whether a FoV is in a relatively overdense or underdense region. This penalty function should drop off sharply when the ratio is greater than 1 in order to restrict the step size and increase quite steeply when the ratio is less than 1 to increase the step size. When the ratio is 1, f should not contribute, i.e. $f(1) = 1$. These requirements suggest that f should be a piecewise function with the above mentioned constraints. One possible general form of f is

$$f(X_j^i < 1) = -mX_j^i + c, \tag{3.7}$$

$$f(X_j^i \geq 1) = (X_j^i)^{-p}, \tag{3.8}$$

where m,c, and p are non-negative real numbers. Figure 3.1 shows some possible piecewise forms of f.

3. Once the penalty and cooling functions have been determined, the Gaussian with variance $\sigma^u = \sigma_0^u \times f \times g$ may be produced from which to obtain Δx^u. The entire process is repeated for $u = 1, 2, ..., m$ so that $\Delta\vec{x} = (\Delta x^1, \Delta x^2, ..., \Delta x^m)$. Of course, for two-dimensional discrete catalogs which is what we will be dealing with, $\Delta\vec{x} = (\Delta x, \Delta y)$ or $\Delta\vec{x} = (\Delta x^1, \Delta x^2)$.

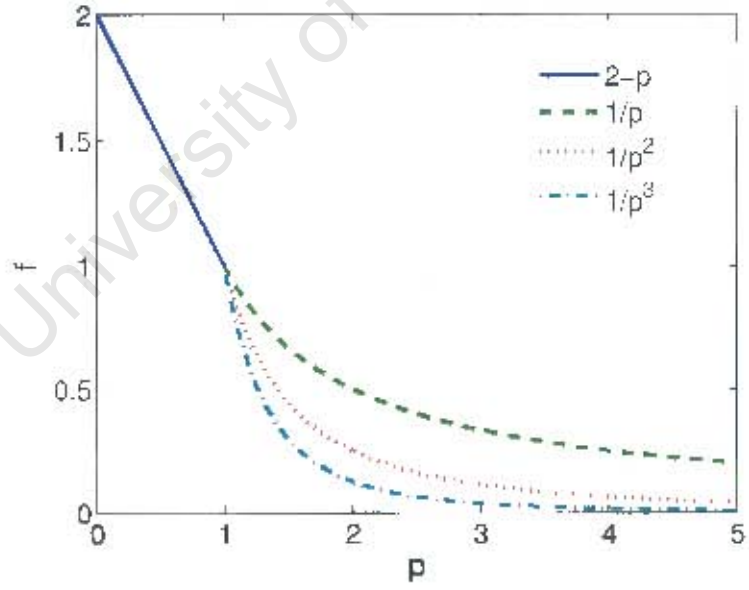

Figure 3.1: Various forms of the function f used in the simulations. f is a function of $X_j^i \equiv FoM_j^i / \sum_{k \neq i}^{N} FoM_j^k$. f enforces the requirement that the FoV take small steps in rich areas ($X_j^i \gg 1$) and large steps in poor areas ($X_j^i \ll 1$).

The merit in adopting the above mentioned approach is quite obvious. It eliminates the need to be concerned with various characteristic length scales since the step sizes are determined heuristically. In standard MCMC methods the distribution from which the Δx^u are chosen is constant and there is no sharing of information between FoV. If a particular FoV finds itself in an overdense region it

has no efficient way of exploring that region, it simply has to wait for the relevant numbers to be extracted from its distribution, relying on chance more than anything else. Thus, a standard MCMC optimisation scheme might take very long to converge to good local extrema. Simulated annealing is an improvement on basic MCMC. Early steps are large, and so, much more of the data set can be explored early on. Still, however, an individual *FoV* does not have a well-organised way of searching its local minimum and might in fact become trapped in a local minimum early on. The HYBRID approach is the best of both. In the chapters that follow, HYBRID is implemented and tested on various discrete data sets as well as continuous functions and the results are then analysed.

It is worth mentioning that when attempting to optimise a *continuous* FoM the overall performance of a stochastic optimistation technique such as MCMC may be improved by utilising the so called covariance matrix which is a way of representing the correlation between steps. Diagonalising the covariance matrix would ensure that walkers take bigger steps in the extended dimensions while ensuring smaller steps in the smaller dimensions. Clearly this would lead to a much more efficient searching method. The discontinuous nature of the FoM in the above applications, however, completely invalidates the use of the diagonalisation of the covariance matrix since the 'surface' we are exploring will not have a neat shape in general. Furthermore, were the diagonalisation of the covariance matrix approach implemented, the MCMC results may not have been improved relative to the HYBRID results since both optimisation schemes would have been affected by the covariance matrix in a similar manner, thereby not altering their relative performance significantly.

Chapter 4

The Discrete Test

This chapter focuses on the first implementation of HYBRID. The method is tested along with the standard methods on various discrete two-dimensional data sets. In using such discrete data sets we are essentially dealing with the classic problem of optimal target selection, the original aim of this investigation. It is demonstrated how HYBRID wins over MCMC and SA in three ways: (1) The maximum average FoM achieved is always higher. (2) The convergence to the average maximum is faster and (3) the HYBRID method is much more consistent in delivering good results so the spread around the maximum is much smaller. Combined this yields significant improvements in performance over SA and MCMC.

4.1 Test Data Sets

With any data set in which the elements are not simply uniformly distributed, there are at least two characteristic length scales which one may immediately identify: the clustering scale l_0 and the overall dimensions of the data set L_0. These length scales become important when performing a random walk. On one hand, a walker should have a large enough average step size to ensure that it efficiently explores the entire data set within a reasonable amount of time. Conversely, if there is clustering on much smaller scales, the average step size should be correspondingly small so that a walker may adequately examine a cluster. HYBRID overcomes the problem of having to decide upon a fixed average step size by continually changing the step size of a FoV based on its relative performance as well as the performance of all the FoV as a group.

To this end, three data sets (figures 4.1, 4.2 and 4.3) were used to test and compare HYBRID to the standard methods:

1. **Test data set 1:** A uniform background of points with small, dense clusters superimposed. 15000 data points in total.
2. **Test data set 2:** A background of points with a constant density gradient and, again, with small, compact clusters overlayed. 28000 data points in total.
3. **Test data set 3:** The 2SLAQ catalog [32] made up of the RA/Dec of 13510 luminous red galaxies (LRG's).

Data sets 1 and 2 were created by the author while data set 3 consists of modified real data. Each of these data sets should test a method's ability to locate optimal regions, the speed of convergence to these regions and variability in results.

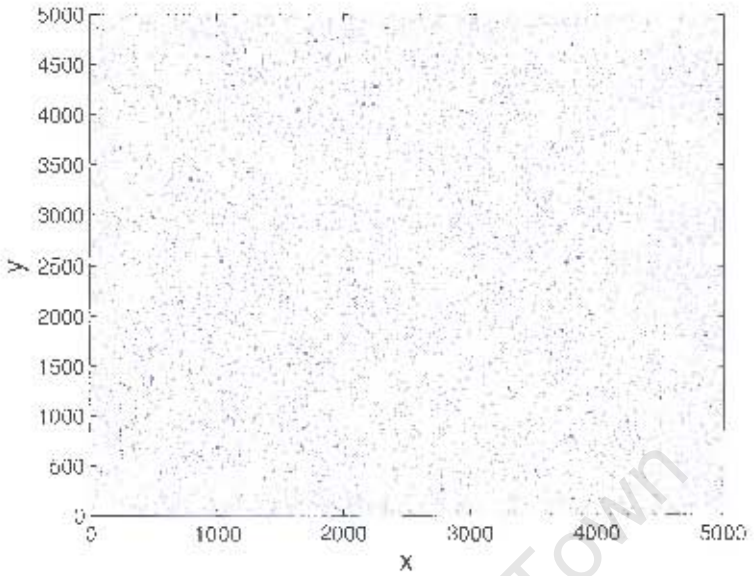

Figure 4.1: Test dataset 1: A uniform distribution of 10^4 points with 100 uniformly distributed dense clusters superimposed each containing 50 points. The data set tests HYBRID's ability to explore small, dense regions.

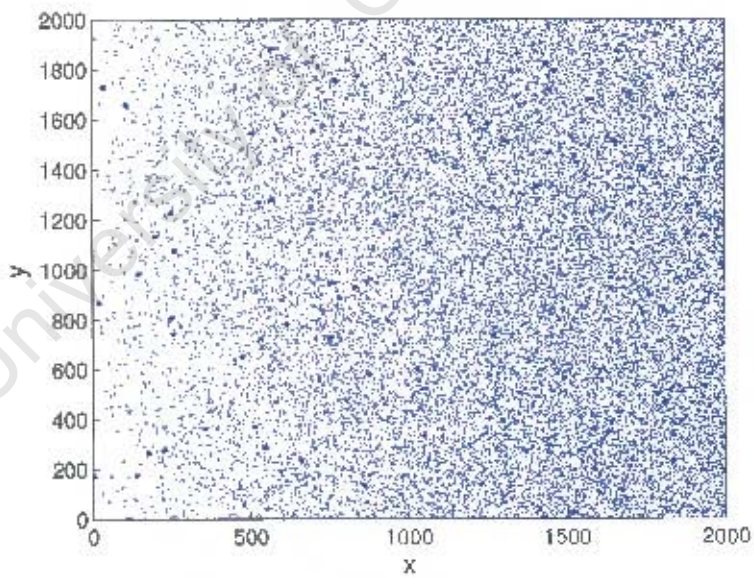

Figure 4.2: Test dataset 2: Gradient background of 25000 points with 60 superimposed dense clusters each containing 50 points. The data set tests HYBRID's ability to move quickly across the density gradient.

4.1.1 Details and Features of the Test Data Sets

Test data sets 1 and 2 were created in a simple enough manner. Data set 1 was made by randomly placing 10^4 points within the data box and then superimposing small, very dense clusters upon the

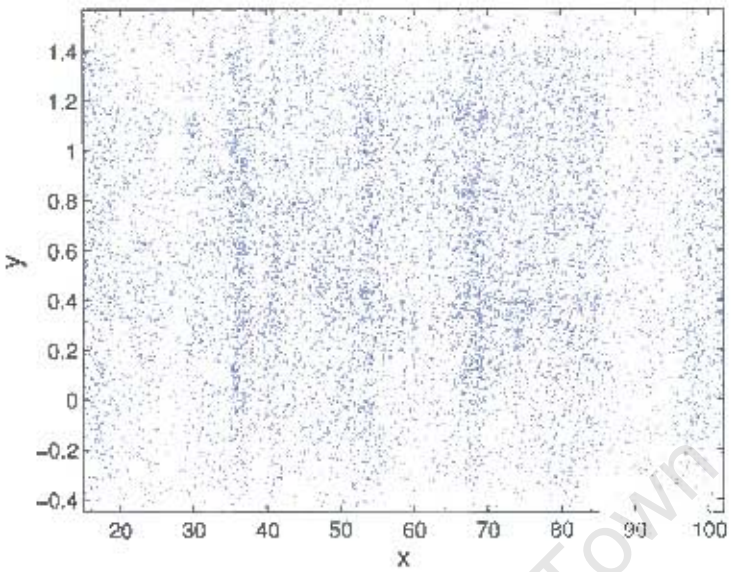

Figure 4.3: Dataset 3: The 2SLAQ LRG catalog with 13500 points and an inhomogeneous density along the x-axis signifying a partially completed survey.

uniform background. The clusters are all similar, being 'radially symmetric'. By radially symmetric it is meant that the probability of finding a point a certain distance from the centre of the cluster decreases linearly with distance out to a specified cut-off radius beyond which no points lie. Data set 2 is similar to data set 1 except that the uniform background is replaced by one with a steady density gradient. Data set 3 is the modified 2SLAQ galaxy catalog. The original catalog consisted of about ten strips of galaxies each largely separated in right ascension. The modification was to remove the gaps between the strips in order to created a catalog with no large voids and with many objects. In reality, HYBRID could be applied to a single galaxy strip at a time.

For data set 1 it was hoped that HYBRID would efficiently explore the clusters, using f and g, after having located them, thereby increasing FoM_j more rapidly with step number j. For the standard methods, a FoV should converge to higher FoM slower once in a cluster, relying on chance more than anything else. The same can be said for data set 2 except that it is now a race to see which method can move quickly its FoV across the density gradient to regions of higher background density and then also find and explore the clusters effectively. Data set 3, consisting of real data with no very dense, well defined clusters, is hoped to show that the HYBRID method could be used practically for a real cosmological survey. Table 4.1 lists some of the specific details of the test data sets.

data set	num of pts	x-range	y-range	num of clusters	cluster radii	num of BC pts
Uniform	15000	5000	5000	100	10	10000
Gradient	28000	2000	2000	60	10	25000
2SLAQ catalog	13510	90	2	N/A	N/A	N/A

Table 4.1: Some specific details of the test data sets. Data set 1 consists of a uniform background with dense clusters uniformly superimposed. Data set 2 is similar to data set 1 except that the uniform background is replaced by one with a density gradient. Data set 3 is modified real data. Data sets 1 and 2 were created by the author unlike data set 3. Some of the specifics of data set 3 are therefore not applicable, as indicated by N/A.

4.2 The Testing Procedure and Parameter Information

Now that the test data sets have been described and the specifics of the HYBRID method explained, we can take a look at how each method performed. Remember, the methods tested were MCMC, SA and HYBRID. Recall that MCMC and SA are subcases of HYBRID. The standard testing procedure was to execute a set of runs with $f = g = 1$ (MCMC), $f = 1$, $g \neq 1$ (SA), and then $f \neq 1$, $g \neq 1$ (HYBRID). At this stage let's clearly define some terminology to be used. For these test data sets, 20 FoV each executing a Markov chain with 10^4 steps constitutes a run. Remember that HYBRID is heuristic since multiple FoV search simultaneously so that they can share information and thereby locate, more effectively, the optimal regions. At each step in a run, the FoM (and other data) are recorded. In principle, after a single run of 10^4 steps, one may produce a plot of survey count versus step number. For a single run, however, the survey count is quite a volatile quantity and so, in order to produce results that are statistically significant, many (typically thousands) of runs are carried out for each method on each data set. Then, a 'smooth' plot of survey count versus step number, for example, may be produced with each value of the ordinate being the average value of that quantity from all the runs.

Let's call a set of a few thousand runs a simulation so that there are 3 simulations (one for each method) for each of the 3 data sets. Before each simulation commences, some parameters need to be decided upon and then initialized. These parameters are: p, m, c, t, α, σ_0^x, σ_0^y. For all 9 simulations, the general form of the penalty function f is

$$f(X^i \leq 1) = -mX_j^i + c, \tag{4.1}$$

$$f(X^i > 1) = (X_j^i)^{-p}. \tag{4.2}$$

The general form of the cooling function g is

$$g(FoM_j) = \left(\frac{FoM_1}{FoM_j}\right)^t. \tag{4.3}$$

The first four parameters (p, m, c, t) therefore control the functions f and g. α is simply the acceptance parameter for a bad step.

At this stage, think back to how a step size in a particular direction is chosen. In general, the data set may be m-dimensional. The test data sets, however, are two dimensional. The x-range and y-range of a data set need not be the same and so the average step size in a particular dimension/direction should scale with the range. This is ensured by the constant σ_0^u which is what the variance should reduce to when $f = g = 1$. Recall that for the general case of m-dimensions, a step may be written as $\Delta\vec{x} = (\Delta x^1, \Delta x^2, ..., \Delta x^m)$. In two dimensions this obviously reduces to $\Delta\vec{x} = (\Delta x^1, \Delta x^2)$. In the two dimensional data sets, therefore, σ_0^1 is the constant variance (when $f = g = 1$) of the Gaussian from which the Δx^1's are chosen and σ_0^2 the variance of the Gaussian from which the Δx^2's are chosen. Test data sets 1 and 2 have equal x and y ranges and so $\sigma_0^1 = \sigma_0^2$. Data set 3 has a very large x-range and a relatively small y-range, thus $\sigma_0^1 > \sigma_0^2$. Table 4.2 lists the p, m, c, t and α parameters used for each of the different methods for each of the data sets.

For all simulations on all data sets, 20 FoV are always used simultaneously. For data sets 1 and 2 the radius of each FoV is $R = 5.0$ while $R = 0.066$ for all simulations on data set 3. For the SA simulations on data sets 1 and 2 the σ_0^u are made significantly larger since it is known that the step size decreases with improved FoM. In this way the SA scheme is able to probe large regions at earlier times and then relax to smaller steps later on. With HYBRID, however, this is not needed since the step size is continuously varied based on a field of view's relative performance. Table 4.3 lists the number of runs in the simulations as well as the σ_0^u used.

Data set	p	m	c	t	α
uniform	0, 0, 2	0, 0, -2	1, 1, 3	0, 0.6, 0.6	1.2, 0.514, 1.2
gradient	0, 0, 2	0, 0, -49	1, 1, 50	0, 0.9, 0.2	0.4, 0.514, 0.4
2SLAQ catalog	0, 0, 0.5	0, 0, -5	1, 1, 6	0, 0.5, 0.5	1, 0.514, 2

Table 4.2: p, m, c, t parameters used for the various methods on the data sets 1, 2 and 3. For each data set, under each column label there are listed 3 entries. The first, second and third entries always correspond to MCMC, SA and HYBRID respectively. Note that the α values stated for SA are the initial α values which decrease logarithmically with step number. These initial α values were chosen in such a way that after 8000 steps the chances of a FoV accepting a bad step that decreases it's FoM by 1 or 2 is approximately 10%.

Data set	number of runs	σ_0^1	σ_0^2
uniform	9911, 5182, 7983	500, 500, 500	500, 500, 500
gradient	4640, 7958, 4948	5, 300, 5	5, 300, 5
2SLAQ catalog	2986, 5079, 3450	10, 10, 5	0.2, 0.2, 0.1

Table 4.3: Number of runs in each simulation and σ_0^1 and σ_0^2 parameters used for the various methods on test data sets 1, 2 and 3. For each data set, under each column label there are listed 3 entries. The first, second and third entries always correspond to MCMC, SA and HYBRID respectively.

4.3 Results

Figures 4.4, 4.5 and 4.6 show the performance of the three methods on each of the three test data sets. They show that HYBRID outperforms SA which in turn is a superior method to MCMC. It is expected that, given enough time, each method will *eventually* reach the optimal survey count. Two features that separate the methods, however, are the speed of convergence and variability in results. These are clearly important details on which to elaborate. Modern cosmological surveys typically produce catalogs of hundreds of thousands of objects, therefore requiring much computational time to analyse and so the rate at which optimal regions can be located is crucial. Furthermore, if there is low variability in the results then fewer runs need to be carried out in order to reach a particular confidence interval on the mean performance. Figures 4.4, 4.5 and 4.6 show that HYBRID noticeably dominates over SA and MCMC in all three departments.

4.3.1 Performance of f and g

Consider data set 2. The idea behind creating such a data set was to really test the efficiency of the performance function, f. Remember that one purpose of this function should be to force a FoV to quickly move away from an under dense region by taking large steps. This is clearly illustrated in figure 4.7 which shows the path of a particular FoV, call it FoV^1. The starting point of FoV^1 is (50, 1000)–the low density end of the data set. What can be seen is that FoV^1 initially takes big steps to move in the general direction of the density gradient. In this way it already outperforms the standard methods which would have needed many more steps for the FoV to migrate across. What we then see (figure 4.8) is that it reaches a cluster and therefore starts taking much smaller steps so as to properly explore the cluster.

The behaviour of a FoV is really dictated by the functions f and g. As a more quantitative analysis of the evolution of these functions with FoM the reader may refer to figure 4.9. The diagram shows the variation of f, g and the FoM of a single FoV (FoV^1 from above, in fact). Notice how in general f decreases in magnitude every time the FoM increases and vice versa. After about 200 steps the FoM of FoV^1 begins to level off with only small fluctuations from then onwards. Notice that thereafter, even though the FoM remains more or less constant, the magnitude of f increases by

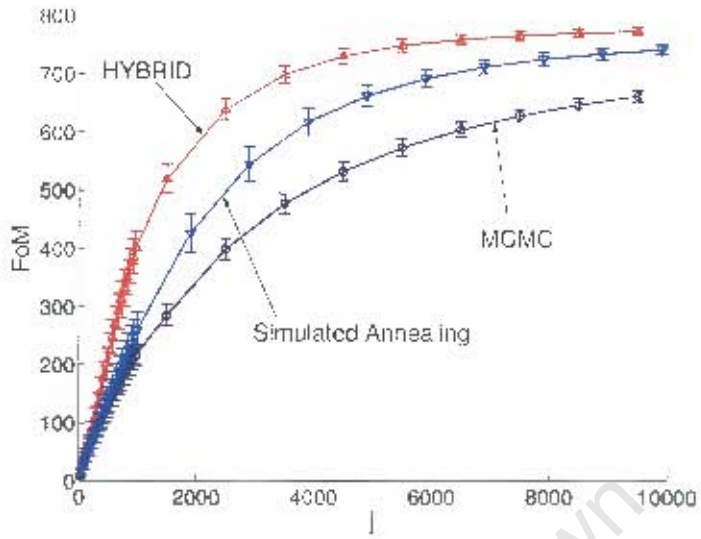

Figure 4.4: Performance on (uniform) test data set 1 (figure 4.1) of the various methods as a function of step j and averaged over all runs. The errorbars show 1σ variation in the FoM over all the runs.

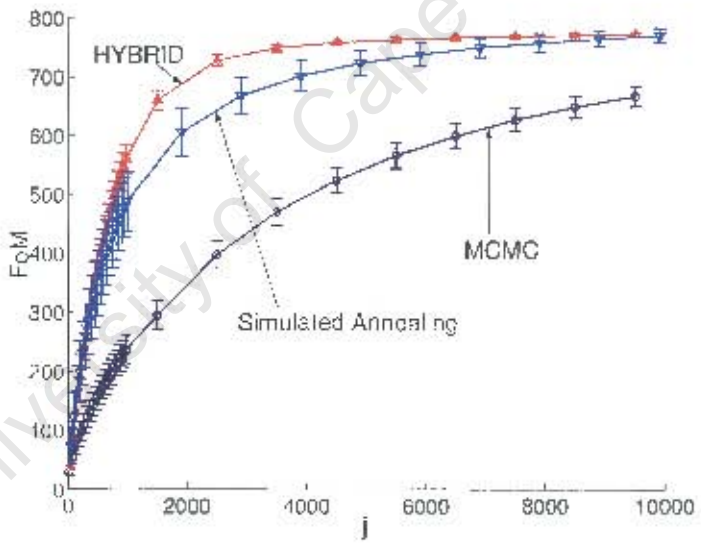

Figure 4.5: Performance on (gradient) test data set 2 (figure 4.2) of the various methods as a function of step j and averaged over all runs. The errorbars show 1σ variation in the FoM over all the runs.

a factor of about 2.5 so that after 10^4 steps it's back up to 1. The reason for this is that while FoV^1 is exploring its cluster, the other $N-1$ FoV are themselves moving to progressively better regions. This has the effect of diminishing the relative performance of FoV^1, i.e. $X_j^i \equiv FoM_j^i / \sum_{k\neq 1}^{N} FoM_j^k$ approaches a value of 1. But f is defined in such a way that it does not contribute to the variance of the distribution from which the step components are chosen when $X_j^i = 1$. This is when the simulated annealing component of HYBRID kicks in.

FoM^1 should still scout the cluster even though its performance relative to the other FoV is deteriorating. In order to ensure this, the cooling schedule, controlled by g, should decrease the step sizes of all FoV. But this is exactly what happens as the survey count increases since the cooling function is defined as $g(FoM_j) = (FoM_1/FoM_j)^t$, i.e. FoM larger than FoM_1(the sum of the figures

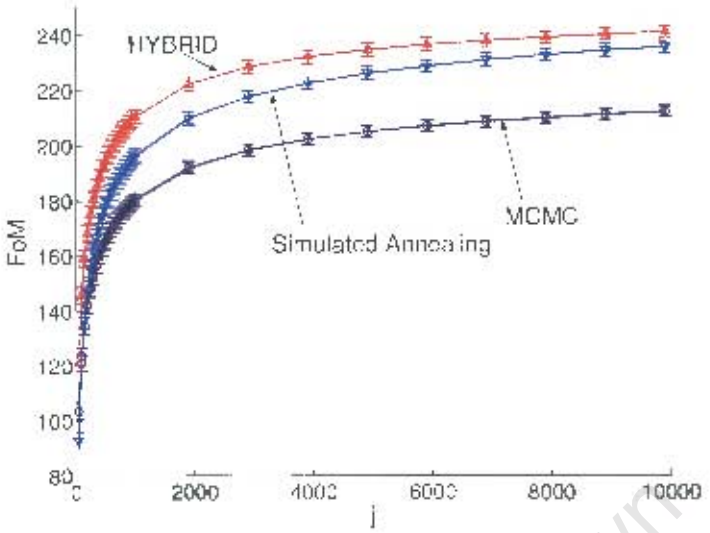

Figure 4.6: Performance on the 2SLAQ catalog (data set 3, figure 4.3) of the various methods as a function of step j and averaged over all runs. The errorbars show 1σ variation in the FoM over all the runs.

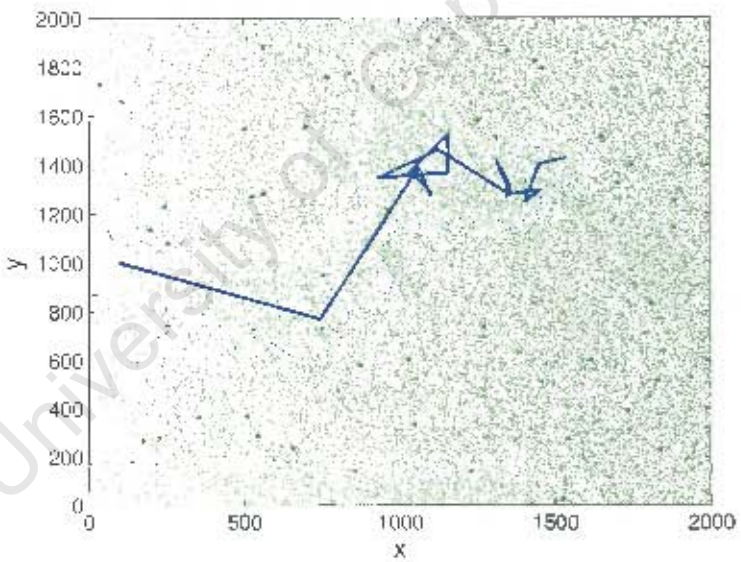

Figure 4.7: Path of one FoV (from left to right) across the gradient+clusters data set described above (figure 4.2). At this scale the path appears to end abruptly two-thirds the way across the dataset. Figure 4.8 shows how the FoV converges rapidly onto a rich cluster.

of merit at step 1 of a run) decrease the step sizes of all FoV. Thus, even though FoV^1 is on a par with the other FoV, all FoV are taking smaller steps due to the the higher FoM_j. By handing over control of the FoV to g at later times, each FoV can still effectively explore optimal regions by being forced to take smaller step sizes. This is exactly what is observed in figure 4.9: as f increases in magnitude while the FoM of FoV^1 remains approximately constant, g decreases in magnitude to compensate. g basically relaxes all the FoV when it is known that they are all performing well. Perhaps a better form of g would be $g(FoM_{j-100}, FoM_j) = (FoM_{j-100}/FoM_j)^t$ for $j > 100$. This would mean that the current performance would be evaluated relative to more recent results, thereby keeping closer

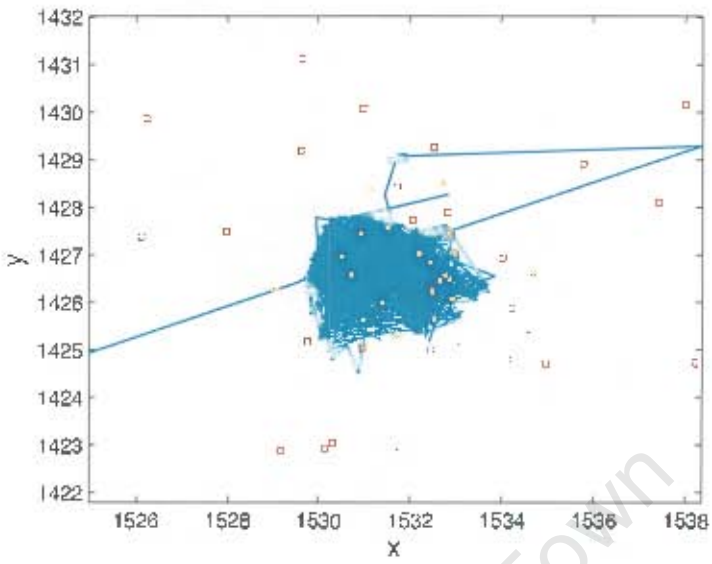

Figure 4.8: Zoom of figure 4.7 showing the final part of the trajectory of the FoV centre after locking onto a cluster. On finding the cluster the average step size plummets by a factor $\sim 10^3$ allowing the FoV to fully explore the cluster. With standard MCMC or SA methods the FoV would have evolved away from the cluster before exploring it. Cluster points are denoted by squares. On average, away from clusters, a region this size would contain less than one point. The radius of the FoV for this data set is $R = 5.0$ showing how the optimal FoV captures most of the points in the cluster.

tabs on overall performance. This functional form, however, was not used. The functions f and g are what allow the fast convergence rate to higher FoM of HYBRID.

4.3.2 Variability in Results

Another important feature of HYBRID is that it is much more stable than the standard methods. It is one thing knowing that a particular method produces good results but it is also important to know how consistently it does so. Again we shall consider the results from data set 2. By simply looking at figure 4.5 one can see that the errorbars for the HYBRID case are significantly smaller than the errorbars for MCMC and SA. It should be mentioned that the errorbars are the usual 1σ errorbars. This tells us that HYBRID is much less volatile than MCMC and SA. We expect this since the functions f and g make the entire method a much more controlled random process. To appreciate this think of what a HYBRID FoV does when in a cluster–it takes a high percentage of small steps thanks to the low-variance step size distributions insisted upon by f. Standard methods, when in a cluster, would be attempting to take steps of all sizes, big and small. A smaller fraction of steps would therefore be of the size required to explore the cluster. The cooling function g also ensures that the appropriately sized steps are attempted throughout the entire run. g is large when the survey count is low and vice versa.

To further appreciate the low-variability of HYBRID results relative to MCMC (SA is similar to MCMC), consider figure 4.10. The figure is a histogram of FoM_j at step $j = 5000$ for the MCMC and HYBRID simulations on data set 2. The first striking feature is that, at step 5000, the HYBRID FoM_j is typically 40% higher than the MCMC FoM_j. Furthermore, variance in FoM_j for HYBRID is much lower than it is for MCMC and so, relative to MCMC, HYBRID produces much better results and, very importantly, does so with more consistency.

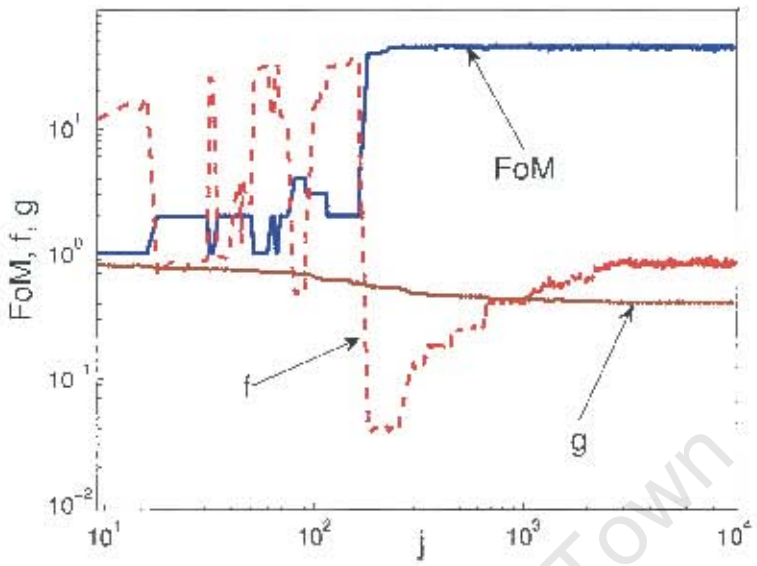

Figure 4.9: Behaviour of the FoM for the single FoV shown in Figs (4.7) and (4.8) as well as the corresponding f and g as a function of step number j. Note how f varies rapidly, anti-correlated with the FoM while g slowly decreases so as to relax the full system of FoV toward the global optimum. At $j \sim 200$ the FoV comes across the cluster and explores it to $j > 10^4$ illustrating how effective HYBRID is. The increase of f for $j > 200$ shows the improved performance of the remaining FoV in the survey which asymptotically match the performance of this FoV (since $f \to 1$). Notice how g decreases in compensation for $j > 200$.

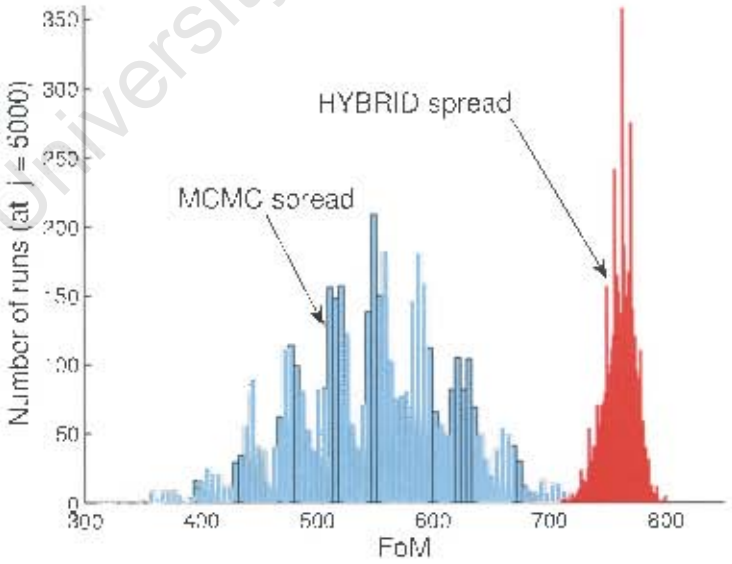

Figure 4.10: Histogram at step $j = 5000$ for the FoM for HYBRID and MCMC (SA is similar to MCMC and is not shown for clarity) on the gradient dataset of figure 4.2. Notice how the HYBRID runs are significantly more clustered than MCMC runs implying much enhanced consistency: HYBRID relies much less on luck than SA or MCMC do.

Chapter 5

Optimisation of Continuous functions

Chapter 4 highlighted the performance of the various methods on discrete two-dimensional data sets. As a natural extension of the testing procedure, the HYBRID algorithm is tested on some continuous functions. Such applications are important in order for the method to be able to be used as a general optimisation technique. To this end two continuous functions are used, each of which again highlight various aspects of the HYBRID method. In this chapter the various methods are each tested upon these continuous functions with the same performance criteria being used as for the discrete case to evaluate the performance of each method.

We now switch to the task of minimisation, i.e. the various methods are made to locate the global minimum of the test function. Of course, now that we are dealing with continuous functions, certain aspects of the simulations are slightly different. Firstly, we now speak of walkers (instead of FoV) which are simply points moving around in the high dimensional domain of the function. The FoM of a walker is just the value of the function at the walker's position. This is in contrast to the discrete examples in which we used FoV of finite size in which case, in order to determine the FoM of a FoV, the number of points within the FoV had to be counted up. This evaluation of a function at a point instead of counting points within a specified region greatly reduces computation time and makes it possible to execute many more runs with more steps in each Markov chain if desired. The second main difference is that for these continuous functions, the most optimal solution–the global minimum–is known analytically and so the performance of the various methods can be more easily examined and studied and also compared to the best possible solution.

5.1 The Test Functions

The first continuous function is a 50-dimensional hyperboloid, call it H, defined by the equation

$$H(\vec{x}) = \sum_{i=1}^{50} x_i^2, \tag{5.1}$$

where $\vec{x} = (x_1, x_2, ..., x_{50})$. The global minimum of zero clearly occurs when $\vec{x} = \vec{0}$.

The second continuous test function is Griewangk's function, often used to benchmark optimisation

algorithms. It is of functional form

$$G(\vec{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1 \tag{5.2}$$

with -600≤ x_i ≤600, i=1, 2. The function (figure 5.1) is highly oscillatory on small scales with many local minima within which walkers might get 'stuck'. Again, the global minimum is known analytically: 0 when $\vec{x} = (x_1, x_2) = \vec{0}$.

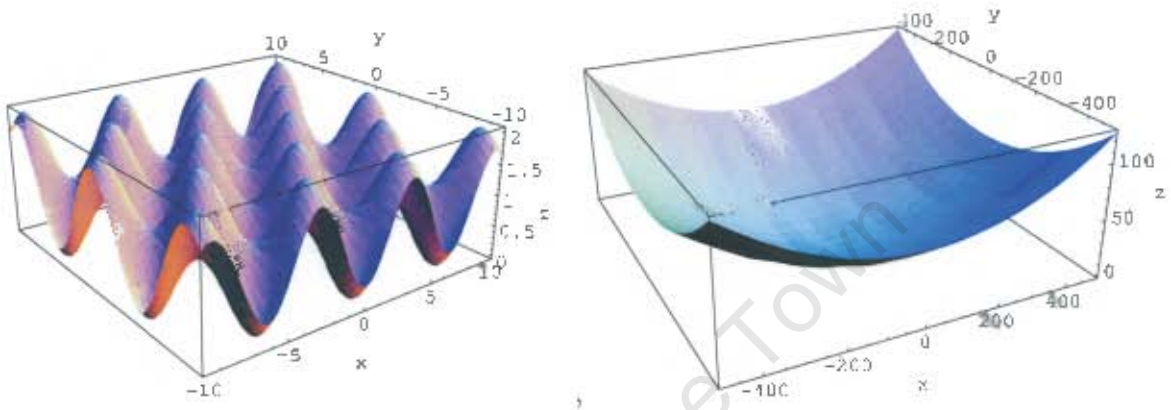

Figure 5.1: Griewangk's function. Note the two characteristic curvature scales-the function varies rapidly on small scales (left) and smoothly (parabolic shape) on large scales (right).

When applied to these continuous functions HYBRID dominates over the standard methods for a simple reason: FoV that have moved significantly toward the minimum will have small f and hence will take small steps since they are outperforming the average FoV. Since they are taking small steps, the probability of taking a step that improves the FoM increases. Conversely FoV doing badly try to take large steps. While this is an unsuccessful strategy on average, it works well in rare cases leading to significant improvements. Hence the system cascades down the slope with a mixture of large, high-risk steps and small, low-risk steps.

5.2 Testing Procedure and Results

The testing procedure for these two continuous functions is almost exactly identical to the procedure for the discrete cases. 20 walkers are made to search simultaneously with MCMC ($f = g = 1$) and SA ($f = 1, g \neq 1$) being sub-cases of the HYBRID method ($f \neq 1, g \neq 1$). For both functions σ_0^y was 1, 10 and 1 for MCMC, SA and HYBRID respectively. For each test function, each simulation consisted of 5000 runs with 10000 steps in each Markov Chain for the hyperboloid and 30000 for Griewangk's function. Note that, unlike before, the problem is now one of minimisation and so f should decrease sharply for ratios smaller than 1. The general piecewise form that was used was

$$f(X_j^i < 1) = (2 - X_j^i)^p \tag{5.3}$$

$$f(X_j^i \geq 1) = mX_j^i + c, \tag{5.4}$$

where p, m and c are real numbers. Figure 5.2 shows the form of f that was used for the HYBRID case on the continuous hyperboloid. Table 5.1 lists the parameters used for all the runs on both of the continuous functions. The parameters for all the simulations are chosen after some experimentation. A certian degree of fine tuning and external knowledge is required when deciding upon these parameters. As an example, when HYBRID was run on test data set 2 (dense clusters distributed

within a background of point with constant density gradient, see section 4.1.1) the slope of f for $X_j^i << 1$ was varied a few times until we were satisfied that the HYBRID walkers were traversing the background density gradient fast enough. As mentioned previously, it would be ideal to have an automated algorithm that could perform this task in a more effective, more systematic way.

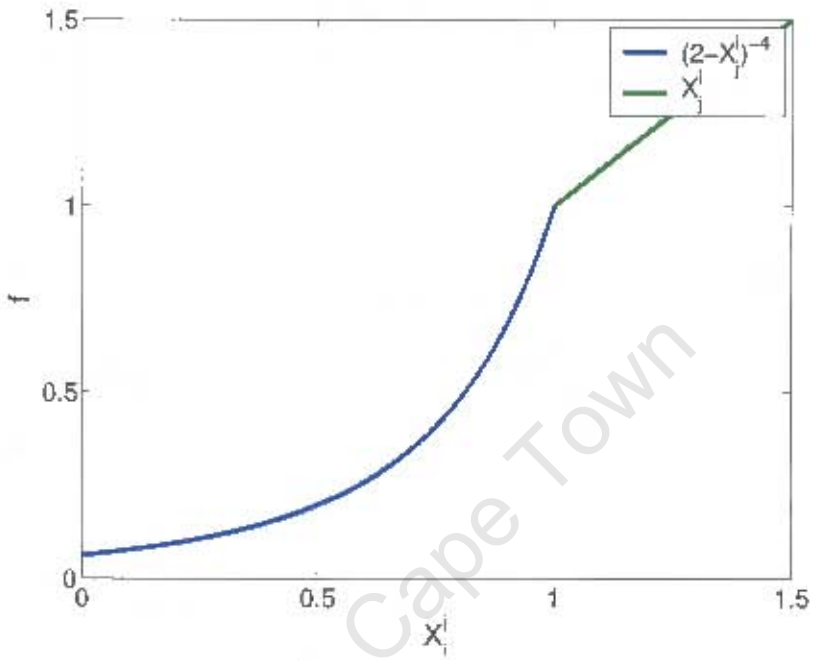

Figure 5.2: Form of the function f used in the simulations as a function of $X_j^i \equiv FoM_j^i / \sum_{k \neq i}^{N} FoM_j^k$. f enforces the requirement that the walkers take small steps in the region close to the global minimum ($X^i \ll 1$) and large steps in regions far away ($X^i \gg 1$).

function	p	m	c	t	α
Hyperboloid	0, 0, -4	0, 0, 1	1, 1, 0	0, 0.2, 0.2	5, 5, 5
Griewangk	0, 0, -6	0, 0, 10	1, 1, 0	0, 0.1, 0.1	1, 1, 1

Table 5.1: p, m, c, t, α parameters used for the various methods on the continuous functions. For each function, under each column label there are listed 3 entries. The first, second and third entries always correspond to MCMC, SA and HYBRID respectively.

5.2.1 50-Dimensional Hyperboloid

For the hyperboloid, the starting position for each of the walkers is always $x_i = 10$ for i =1, 2,..., 50. In the case of these test functions, the FoM of a walker at a particular point is simply the value of the function at that point. For each method on each of the two functions, the average FoM at step j from all 5000 runs was calculated and then plotted against step number. *Note* that for each step in a chain, the figure of merit that we quote for step j, FoM_j, is the lowest FoM out of all the walkers, i.e.

$$FoM_j = min_i(FoM_j^i) \tag{5.5}$$

where i=1, 2, ..., 20.

Looking at figure 5.3, HYBRID clearly dominates over the standard methods. For all methods $FoM_1 = 10^2 \times 50 \times 20 = 10^5$. After about 2000 steps the MCMC method already becomes saturated

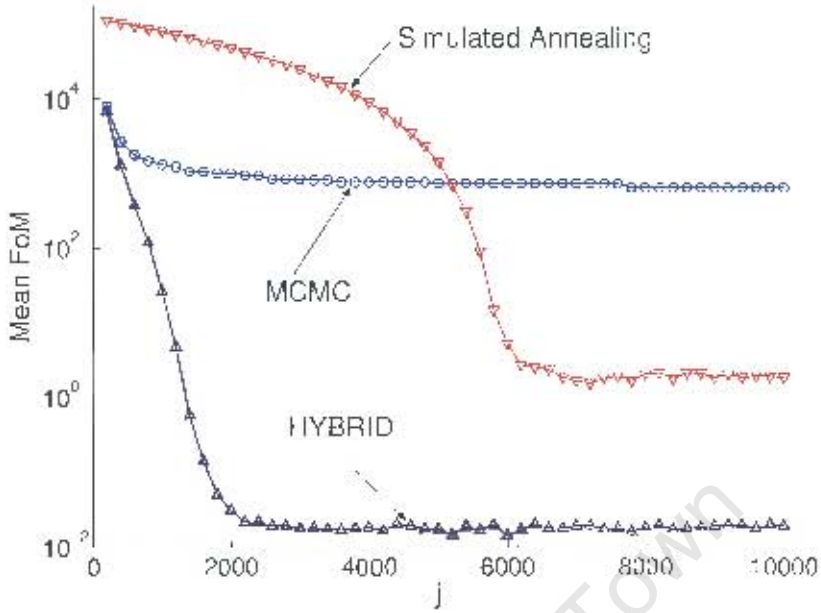

Figure 5.3: Mean performance averaged over 5000 runs as a function of step j on the continuous 50-dimensional hyperboloid given by eq. (5.1). HYBRID completely outperforms both SA and MCMC since the function f allows the FoV to cascade toward the minimum by taking large steps and then to properly explore the minimum by enforcing small steps.

at $FoM_j \sim \times 10^4$. SA performs poorly early on, converging very slowly. This is because it is initially taking large steps in order to explore the region, the difficulty with this may be attributed to the high dimensionality of the space. After about 5000 steps, however, it dips dramatically and starts to converge quickly until it levels off to $FoM_j \sim 2$. HYBRID, on the other hand, already outperforms both of the standard methods within 1000 steps and very quickly converges to a low FoM less than 0.1, close to the global minimum of 0. As for the variance in results, because of the log scale used on the y-axis in the plots, error bars are not included.

One might argue that the volume of the hyperboloid around the minimum is very small compared to the total volume. The affect that this might have on the MCMC results would be that the walkers would spend many more steps in regions of higher functional value since the time a walker would spend in a region is roughly proportional to the volume of that region. The very small relative volume near the minimum would therefore not be as thoroughly explored by the walkers which saturate out at higher FoM. This is a valid point and could be handled by using a log parameterisation which would effectively stretch out the region. The problem with such an argument is that the point at which the minimum occurs *is usually not known* and so the method will not be of any parctical help in general. The advantage in using HYBRID as an optimisation technique is that is continuously adjusts its average step size so that it can more effectively explore regions of varying volume, it takes care of the problem on its own.

5.2.2 Griewangk's Function

The second test function is Griewangk's function. As already mentioned, it is a highly variable function with many local minima. Figure 5.4 shows a slice through the function at y=0. The idea behind choosing this test function is that walkers in the standard methods should get stuck in the local minima and struggle to find the global minimum of zero. In this case, operations were carried out slightly differently. In some optimisation algorithms, the minimum of a function, usually the

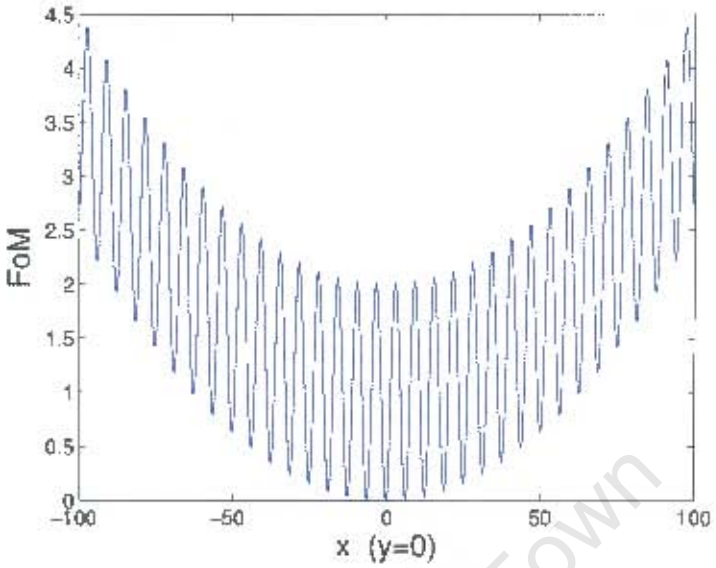

Figure 5.4: Slice through Griewangk's function at $x_2 = 0$. Note the two length scales of variability governed by the large-scale quadratic term and the small-scale oscillatory term.

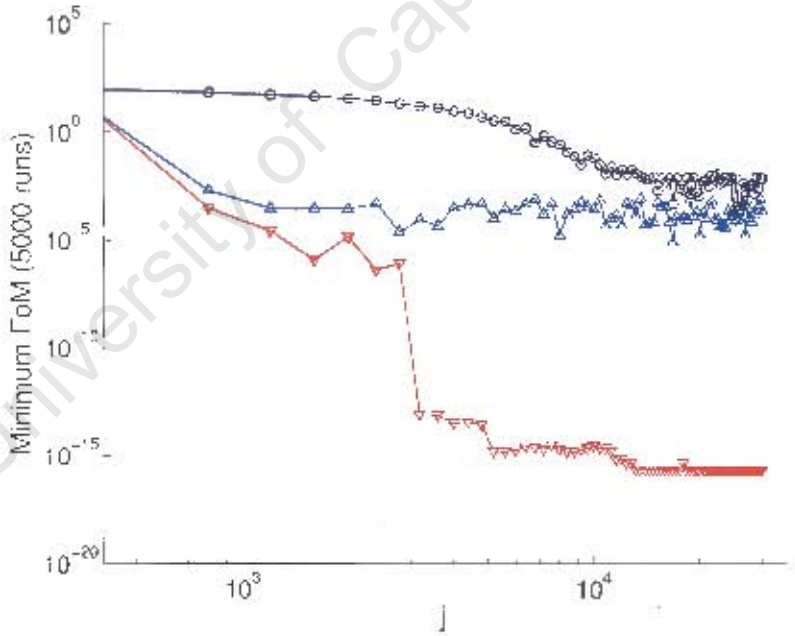

Figure 5.5: Best performance over 5000 runs as a function of step j on Griewangk's function. HYBRID significantly outperforms the other methods, reaching values of order 10^{10} smaller than either SA or MCMC. Each data point is the minimum FoM achieved by any FoV out of all the runs at step j. For very small FoM, f is made to depend only on the FoM allowing rapid convergence to very small values of the function. The best values achieved by SA or MCMC after 10^4 steps were achieved by HYBRID after only ~ 500 steps.

global minimum, is modeled as a paraboloid or some other suitable function. This allows for an easier search for the minimum. In our case, it was decided to let a FoV search its minimum region

independently of the others after having reached a sufficiently good level of relative performance. The way this is done is to say that for when a walker, i say, reaches a low, pre-defined X_j^i threshold, the penalty function takes on the form $f = \sqrt{FoM_j^i}$. In this way, f will continuously and independently decrease in magnitude as the FoM of a particular FoV increases. Furthermore, having implemented this strategy, to illustrate the performance of the methods, it was decided to plot (figure 5.5), against step number, the lowest FoM from all the runs in a simulation reached at each step.

Clearly HYBRID by far outperforms MCMC and SA. In 5000 runs, after 30000 steps, it is able to reach a minimum value of the order of 10^{-16}, a value that is approximately 10 orders of magnitude smaller than the lowest value reached by the standard methods. To remind the reader, the plot in figure 5.5 does not contain error bars since it is not average values that are being plotted against the step number but rather the minimum value reached out of 5000 runs.

Chapter 6

Optimisation Conclusions

We end this first part of the thesis off with a brief conclusion. The reader is reminded that part 2 of this thesis deals specifically with parameter estimation techniques used in cosmology, a separate conclusion for which may be found at the end of part 2. The conclusion below is only relevant to part 1.

Cosmological surveys have changed drastically over the decades. Modern surveys demand efficient and effective optimisation techniques to extract data subsets of interest, results, etc. In this first part of the thesis we address the particular problem of optimal target selection. To remind the reader, a statement of the task is: *Given right ascension and declination information for many objects, what are the best regions to place N fields of view (FoV) of fixed size*[1] *so as to maximize the total number of objects observed?*. The problem is an important one to be able to solve since large data sets produced by modern surveys demand automated optimisation methods. The investigation is based upon the creation, testing and implementation of a new heuristic optimisation scheme called HYBRID. The new method uses aspects of Monte Carlo Markov Chain optimisation (MCMC), simulated annealing (SA) and particle swarm optimisation (PSO). Given N FoV that are searching a space simultaneously, HYBRID is based upon the idea that the size of the step that a particular FoV attempts should be decided upon by its performance relative to the other $N-1$ FoV. Thus the multi-variate Gaussian from which the step that a walker attempts is selected is continuously varying so that the walker may either move away from a poor region or carefully explore a dense region of interest. This is accomplished by incorporating two controlling functions, f and g, that relax a particular FoV in to a good region and relax all the FoV when they are all performing well.

HYBRID was first tested, along with MCMC and SA, on various discrete data sets. The first two data sets, created by the author, test the method's ability to efficiently explore dense clusters as well as its ability to move quickly across a density gradient. The third data set consists of the 2SLAQ luminous red galaxies catalog. The aim of using such a data set is to demonstrate the practical application of HYBRID. On all three test data sets HYBRID outperforms the standard methods in terms of best regions located as well as speed of convergence to those regions. Furthermore, the variation in HYBRID output is significantly less than those of the standard methods meaning that fewer runs are required in order to reach a certain confidence interval of performance, an important benefit when computational resources are limited. In light of these promising results, it is the aim to use HYBRID to optimise forthcoming luminous red galaxy surveys, and others, on the new Southern African Large Telescope.

As a further probe of HYBRID's performance, it was used to minimise two continuous functions. The idea behind such tests was to show that HYBRID can be applied to more general optimisation

[1] As an example, the SALT has a FoV of size 8 arcminutes.

problems that may arise in industry etc. It was on these continuous functions that HYBRID outperformed the standard methods by the largest amount. On the first continuous test function, a 50 dimensional hyperboloid, $H(\vec{x})$, HYBRID converged very quickly to a value less than 0.1, while the other methods struggled to move in the $-\nabla H(\vec{x})$ direction through the high dimensional space. On a second, highly oscillatory continuous test function, HYBRID managed to obtain functional values as much as 10^{10} times lower than the best standard method results. The reason for this is that the penalty function, f, allows a HYBRID FoV to realise that is in a local minimum and that it should attempt large steps to escape that minimum while standard methods generally have no way of knowing that a FoV is in a relatively bad region and so the FoV spends much time in that local minimum. Furthermore, f may be used to make a FoV search its local minimum independently of the other FoV once a pre-determined performance threshold is reached.

Although there are further improvements that may be made to this new method, these initial results suggest that HYBRID is superior to the existing standard methods of optimisation. Future work may involve developing an adaptive technique that will calculate the best form of function f given a particular data and perhaps also function g so as to improve the results yielded by HYBRID.

Part II

Monte Carlo Parameter Estimation

Chapter 7

Parameter Estimation Introduction

Data from modern cosmological surveys demands very specialised analysis. Cosmologists need to be familiar with various statistical methods (see e.g. [13]) in order to extract the required information from their data. One such type of analysis is Monte Carlo Markov Chain parameter estimation in which a walker is made to execute a random walk through a multi-dimensional parameter space. Given a data set along with a theoretical model which we believe should describe that data set, the parameter estimation process allows us to determine the most likely model parameters [14, 33].

The author is privileged enough to have been a full external collaborator of the SDSS-II supernova survey[1] which started in the latter part of 2005. The survey is expected to discover approximately 200 new type Ia supernovae (SNIa) by 2008. Already, the discovery of ~ 130 new confirmed SNIa has been announced[2]. These newly discovered SNIa will help constrain dark energy parameters and allow us to further understand the ways in which SNIa may be used as standard candles. The second part of this thesis is based upon the development and testing of an analysis pipeline that will be used on the SDSS-II first year data. The necessary background theory is discussed in chapters 8 and 9. The tests of the pipeline on the Supernova Legacy Survey (SNLS) first-year data are presented in chapter 10. Finally we study the intrinsic accuracy of the MCMC method, the results of which appear in chapter 11.

Our standard model of the Universe at the moment is one dominated by two main energy components: dark energy and cold dark matter [34]. Astronomers have long known about dark matter. Some of the best evidence for dark matter lies in the flat rotation curves of spiral galaxies as well as gravitational lensing and structure formation in the Universe. At present we believe that the dark matter together with normal baryonic matter makes up about $\sim 27\%$ [8, 9] of the energy density of the Universe. The favoured candidate for dark matter is some sort of weakly interacting exotic particle which does not *yet* form part of our standard model of particle physics.

Arguably the greatest mystery in cosmology at present, and perhaps even of physics in general, is that of dark energy. Surprising results released in 1998 [35, 36], the outcome of two SNIa surveys, indicated that the expansion of the Universe has actually experienced a recent phase (we have no evidence at $z = 0$) of acceleration, thereby championing the notion of a significant component of the Universe that has associated with it a negative pressure that drives the acceleration. This mysterious component has been named dark energy, the name indicating that we have relatively little understanding of its properties. Recent observations [37, 8, 9] suggest that $\sim 70\%$ of the current energy density of the Universe may be made up of dark energy.

The simplest candidate for dark energy is the cosmological constant, Λ, first introduced by Einstein

[1] http://sdssdp47.fnal.gov/sdsssn/sdsssn.html
[2] CBET 229, 247, 254, 259, 268, 272, 280, 281, 304, 315, 339, IAUC 8625, 8621

so that his theory of general relativity could give a static Universe, the accepted view at the time. Λ is governed by the equation of state $P = -\rho$. It has a constant negative pressure P equal in magnitude to its energy density ρ. Λ's main contender is quintessence [34, 38, 39], a dark energy component of the Universe that can vary in both space and time. Thus far, there is no conclusive evidence for any variation in the dark energy. The main prize in examining the validity of a multi-parameter dark energy model (as opposed to a single parameter model such as that for Λ) is to prove that the dark energy is non-constant, thereby eliminating Λ as a possible candidate. Future surveys have already been planned [40, 41, 42, 43] that will hopefully collect enough data to allow us to further quantify the dark energy.

The application of the parameter estimation techniques in this thesis are therefore to SNIa data. While the cosmological constant is governed by a single parameter equation of state, i.e. $w_\Lambda = \rho/P$ where $w_\Lambda = -1$, we consider the more general case of $w_0 = \rho/P$ where $w_0 \neq -1$. Together with w_0 we also to estimate the best fit parameters for the amount of matter (dark and baryonic) in the Universe, Ω_M, as well as Hubble constant, H_0 at $z = 0$.

Chapter 8

Parameter Estimation

It is the job of scientists to observe the world around them and then produce theories which describe what they see. The predictions of the theory need to be compared with observations as a test of the theory's validity. This chapter deals with the statistical methods that allow us to extract from a data set the most likely parameters of the theory. We start with the description of the χ^2 test statistic which is what we use to minimize the discrepancy between our theory and observations. The idea of χ^2 is then adapted to the likelihood function, $\mathcal{L} \propto e^{-\chi^2/2}$. Thereafter we explain how the χ^2 statistic may be used in a Monte Carlo Markov Chain context in order to carry out the parameter estimation process.

8.1 The χ^2 Goodness of Fit Statistic

Suppose that we have made a series of N measurements, giving N data points, each consisting of a triple (x_i, y_i, σ_i), where σ_i is the error of the measurement y_i (we assume for simplicity that there is no error in x_i). Suppose also that we have some theory that, given the independent variable x_i, makes a prediction for y_i. In general the theory will contain in it some free parameters $\vec{\theta} = (\theta_1, \theta_2, ..., \theta_N)$ which ultimately govern the predictions of the theory. *Assuming* that the theory is correct, an important question is: How can we determine the best parameters $\vec{\theta}_*$ such that the theoretical predictions most accurately match the measurements? Assuming the data are uncorrelated, one answer to this question is to minimise the χ^2 goodness of fit statistic which is:

$$\chi^2(\vec{\theta}, x_i) = \sum_{i=1}^{N} \frac{\left(y_i - y_i^{th}(\vec{\theta}, x_i)\right)^2}{\sigma_i^2} \tag{8.1}$$

where $y_i^{th}(\vec{\theta}, x_i)$ is the predicted value from the theory, corresponding to x_i. We choose to deal with χ^2 rather than χ^n where $n \neq 2$ since the probability distribution of χ^2 is known[1], thereby allowing us to assign a probability $P(\chi^2)$ to a given set of data measurements. Note that the contribution of a measurement to χ^2 is larger if its associated error σ_i is small. This means that for a measurement with a large σ_i the only way to minimise the quantity $\left(y_i - y_i^{th}(\vec{\theta}, x_i)\right)^2 / \sigma_i^2$ is to minimise the difference between the actual measurement y_i and the theoretically predicted value $y_i^{th}(\vec{\theta}, x_i)$. This is exactly what we want from a statistic like χ^2.

[1] The χ^2-distribution with p degrees of freedom is the probability density function for the sum $z = x_1^2 + x_2^2 + ... + x_p^2$ where each x_i is extracted from a Gaussian.

One may define the reduced χ^2 statistic, χ^2_{red}, which is simply χ^2 divided by the number of degrees of freedom, df. df is defined as the difference between the number of data points N and the number of fitting parameters n_p:

$$df \equiv N - n_p. \tag{8.2}$$

The reduced χ^2 statistic is then defined as

$$\chi^2_{red} \equiv \frac{\chi^2}{df}. \tag{8.3}$$

A general rule of thumb for χ^2 fitting is that $\chi^2_{red} \approx 1$ corresponds to a good fit. This can be understood for large df since the χ^2 distribution tends to a normal distribution with a mean of df and a standard deviation of $2df$, hence $\chi^2_{red} \to \frac{df}{df} = 1$.

8.2 The Likelihood Function

We now need to introduce the likelihood function, $\mathcal{L}$. This is a function that is intimately linked to χ^2 so that by minimising χ^2 we maximise $\mathcal{L}$. In terms of our parameterized theory, $\mathcal{L}$ is defined as the probability that an experiment would yield a certain set of data, $\vec{y} = \{y_1, y_2, ..., y_N\}$, given the theory. This likelihood function provides us with a wealth of information. By locating the point in parameter space at which $\mathcal{L}$ peaks we may determine the best-fit free parameters $\vec{\theta_*}$ of our theory. Furthermore, by knowing the curvature of $\mathcal{L}$ around its peak we may also produce an error estimate for $\vec{\theta_*}$.

We will assume that each measurement y_i is distributed according to some probability distribution[2] $P(y_i)$ which we take to be Gaussian:

$$P(y_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp[-(y_i - \bar{y}_i)^2/2\sigma_i^2] \tag{8.4}$$

where $\bar{y}_i$ is the mean value of y_i and σ_i is the standard deviation of the statistical error in the measurement[3]. Assuming that there is *no correlation* between measurements, the joint probability that a single experiment or observation results in the set of values $\vec{y} = \{y_1, y_2, ..., y_N\}$ is the product of the individual distributions:

$$P(\vec{y}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_i} \exp[-(y_i - \bar{y}_i)^2/2\sigma_i^2] \tag{8.5}$$

$$= const \times \exp\left[\sum_{i=1}^{N} -(y_i - \bar{y}_i)^2/2\sigma_i^2\right]. \tag{8.6}$$

To make the transition from this joint probability to the idea of maximum likelihood we first replace the ideal mean values $\bar{y}_i$ of the data with the theoretically expected values $y_i^{th}(\vec{\theta}, x_i)$ which result from the set of parameters $\vec{\theta}$. It makes sense that we should do this since we want our theory to accurately predict the ideal mean values. Equation 8.5 then becomes the model-dependent conditional probability that a single experiment yields the values $\vec{y} = \{y_1, y_2, ..., y_N\}$ assuming parameters $\vec{\theta}$:

$$P(\vec{y}|\vec{\theta}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_i} \exp[-(y_i - y_i^{th}(\vec{\theta}, x_i))^2/2\sigma_i^2]. \tag{8.7}$$

This is in fact the way in which we define the likelihood function (see e.g. [44]):

$$\mathcal{L} \equiv P(\vec{y}|\vec{\theta}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_i} \exp[-(y_i - y_i^{th}(\vec{\theta}, x_i))^2/2\sigma_i^2]. \tag{8.8}$$

[2] Given the probability distribution $f(x)$ for a variable x, the probability that $x_* < x < x_* + dx$ is $P(x_* < x < x_* + dx) = f(x)dx$.

[3] The error quoted with a measurement is usually the standard deviation.

Notice that although each of the data points may be distributed according to a Gaussian, the likelihood function *is not* necessarily a Gaussian itself.

For the purposes of parameter estimation, however, we are actually interested in $P(\vec{\theta}|\vec{y})$ which is the probability that a certain set of theoretical parameters is correct given some data. The likelihood function is therefore not quite what we need. We now need to somehow relate $\mathcal{L}$ to $P(\vec{\theta}|\vec{y})$. The way to do this is to use Bayes' theorem:

$$\frac{P(B|A)}{P(A|B)} = \frac{P(B)}{P(A)}. \tag{8.9}$$

If we let proposition A represent $\vec{y}$ and let proposition B represent $\vec{\theta}$ then what we get is

$$P(\vec{\theta}|\vec{y}) = \frac{P(\vec{y}|\vec{\theta})P(\vec{\theta})}{P(\vec{y})}. \tag{8.10}$$

Bayes' theorem therefore links $P(\vec{\theta}|\vec{y})$, often called the posterior probability, to the likelihood. It turns out that we may ignore the denominator for the parameter estimation, it does not influence the point in parameter space at which $\mathcal{L}$ peaks or the curvature of $\mathcal{L}$ near that point. The reason for this is that when the posterior $P(\vec{\theta}|\vec{y})$ is integrated over $\vec{\theta}$ it must yield unity if we take our theory to be correct. The denominator is therefore a constant normalisation factor that is independent of $\vec{\theta}$. The second factor $P(\vec{\theta})$ in the numerator is called the prior probability and represents our previous beliefs or knowledge concerning the parameters. A common prior to use is a Gaussian centred about some parameter value that be believe is correct to high accuracy. *The priors that we use for the analysis carried out in this thesis are simply uniform priors as the simplest assumption.* When we use a uniform prior, $P(\vec{\theta})$ is simply a proportionality constant which may be ignored, yielding

$$P(\vec{\theta}|\vec{y}) \propto P(\vec{y}|\vec{\theta}) = \mathcal{L}. \tag{8.11}$$

Therefore in this case we may indeed attempt to maximise $\mathcal{L}$ in order to maximise the posterior, thereby obtaining the most probable set of parameters of a theory for some given data set. Ignoring the constant factor in eqn 8.8 we have

$$\mathcal{L} \propto \exp\left[\sum_{i=1}^{N} -(y_i - y_i^{th}(\vec{\theta}, x_i))^2/2\sigma_i^2\right]. \tag{8.12}$$

Using (8.1) this becomes

$$\mathcal{L} \propto \exp\left(-\frac{\chi^2(\vec{\theta}, x_i)}{2}\right). \tag{8.13}$$

Hence, as we said earlier, the likelihood function is maximised by minimising χ^2, thereby providing us with the parameters θ_* that allow our theory to best-fit the data. One may attempt to calculate the extrema of χ^2 by solving:

$$\nabla\chi^2(\vec{\theta}, x_i) = \vec{0}. \tag{8.14}$$

The problem with such an approach is that it becomes difficult when y_i^{th} has a non-linear dependence on the fitting parameters. Analytic techniques such as the method of steepest descent, in which steps of varying size are taken in the direction opposite to the gradient, suffer the possibility of becoming stuck in local extrema and are also computationally intensive. As will be seen, we use the stochastic method of MCMC to find parameters that produce low χ^2 values and also to estimate the curvature near the minimum, i.e. the errors associated with the best-fit parameters.

In general, if we are estimating p parameters then $\mathcal{L}$ will be p-dimensional. In order to estimate the error associated with a single free parameter we need a one dimensional likelihood function for that parameter. The way to do this is by the process of marginalisation. If the likelihood of θ_1,

for example, is required then we integrate the posterior over all the parameters, except for θ_1. The likelihood for θ_1 may therefore be written as

$$\mathcal{L}(\theta_1) = \int P(\vec{\theta}|\vec{y}) d\theta_2 d\theta_3 ... d\theta_p. \tag{8.15}$$

Clearly, for complex, high-dimensional posterior probabilities, marginalisation involves potentially computationally intensive numerical integration methods.

8.3 The Parameter Estimation Process

Having defined χ^2 and $\mathcal{L}$ let us now take a look at exactly how they are implemented in a Monte Carlo Markov Chain context in order to carry out the parameter estimation process. As shall be mentioned in further detail in chapter 10, when estimating Ω_M, H_0 and w_0 we consider three separate cases each differing in the way in which we fix certain parameters while allowing others to vary. For now we shall deal with the case in which we estimate all three parameters. The code for this case appears in appendix C.

8.3.1 Generating the Parameter Chains

First we need to generate the parameter chains, a parameter chain being the collection of parameter values from a Markov chain as it moves around the parameter space. A single program is dedicated to this task. Suppose the walker is at step j in its Markov chain. At this step it will have associated with it a specific set of parameters, call these $\vec{\theta}_j$. A step $d\vec{\theta}$, the components of which are extracted from a multi-variate Gaussian, is then generated which the walker should attempt. To determine whether or not the step is taken, $\chi^2\left(\vec{\theta}_j\right)$ is calculated as well as $\chi^2\left(\vec{\theta}_j + d\vec{\theta}_j\right)$ and then these two quantities are used in the Metropolis-Hastings algorithm (see section 2.2.2). This process of taking a step and calculating $\chi^2(\vec{\theta})$ is obviously repeated again and again, keeping track of the parameters $\vec{\theta}_*$ that produce the lowest χ^2, i.e. χ^2_{min}, in all the chains. When a walker has finished its Markov chain it would have produced a data set as shown in table 8.1. We speak of each of columns 2, 3, 4 and 5 as being a single parameter chain. This MCMC process forms the core of the data pipeline. The generated parameter chains are used to estimate the best-fit parameters along with an error estimate for those values.

8.3.2 Burning in the Chains

Table 8.1 shows an example of the data that is produced when a walker is made to execute a single Markov chain. In practice what we did was use a loop to make the walker execute numerous Markov chains thereby producing many parameter chains for a single free parameter. One we have the desired number of parameter chains (we typically used 3-5 chains), the next point of operation is to burn each of them in. All that this means is that the first S steps, below which the likelihood function $\mathcal{L}$ is less than some threshold likelihood, are thrown away. In our analysis the threshold is half the maximum likelihood, i.e $\frac{1}{2}\mathcal{L}_{max}$. In this way, if a chain starts in a very bad region of parameter space, the early steps that it takes in order to reach a relatively good region of the parameter space are discarded and are not used in calculating the best-fit values along with the errors.

step	χ^2	Ω_M	H_0	w_0
1	264.288635	0.616123378	56.1640816	-1.25681531
2	174.822311	0.655647635	57.4968605	-1.31792891
3	174.822311	0.655647635	57.4968605	-1.31792891
4	174.822311	0.655647635	57.4968605	-1.31792891
5	133.740570	0.651500165	58.8776016	-1.37459946
6	133.740570	0.651500165	58.8776016	-1.37459946
7	133.740570	0.651500165	58.8776016	-1.37459946
8	95.9190292	0.660857499	60.2408295	-1.37797964
9	95.9190292	0.660857499	60.2408295	-1.37797964
10	71.9717484	0.685378730	62.2889061	-1.42566955
11	71.9717484	0.685378730	62.2889061	-1.42566955
12	71.9717484	0.685378730	62.2889023	-1.42566955
13	71.9717484	0.685378730	62.2889023	-1.42566955
14	71.9717484	0.685378730	62.2889023	-1.42566955
15	71.9717484	0.685378730	62.2889023	-1.42566955
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

Table 8.1: An example of the data set produced by a walker after it has completed its Markov chain through the multi-dimensional parameter space. Column 1 is the step in the Markov chain. Columns 2, 3, 4 and 5 are then the goodness of fit test statistic, the matter density parameter, the Hubble parameter and the equation of state parameter of dark energy respectively, all corresponding to step j.

8.3.3 Finding the Most Likely Parameter Values

Finally, the burnt-in parameter chains are concatenated to form the so called 'master parameter chain' from which we estimate the best-fit values along with the associated errors of the parameters. For the master chain of a single free parameter, the best-fit parameter value that we quote is the median of the chain. We use the median and not the mean since it is not certain that the parameter chain entries will have a Gaussian distribution. We find that typically the mean and median of a chain are very close in the cases we consider. Having calculated the best-fit value of a parameter we need to give an error estimate along with it. Recall that it is the curvature around the maximum of $\mathcal{L}$ that provides us with an estimate of the associated errors of the best-fit values of the parameter.

The beauty of the MCMC parameter estimation method lies in the fact that marginalisation is trivial. In order to produce the 1-dimensional marginalised likelihood function of a particular parameter we simply histogram the various values of that parameter and then quote the spread around the peak of the histogram as the error estimate of the parameter. The reader is referred to [45] for further explanation. In order to quote the error, we therefore find the parameter values, call them p_1 and p_2, on either side of the median for which 34% of the data lies between p_1 and the median and 34% lies between p_2 and the median. In other words, approximately 68% of the data lies between p_1 and p_2. p_1 and p_2 are then quoted as the upper and lower error estimates of the best-fit value, i.e. the 68% confidence interval. Of course, for the case that the parameter values of a chain do have a Gaussian distribution, the median corresponds to the mean and p_1 and p_2 lie one standard deviation from that mean (by the Empirical rule). As an example, for the Supernova Legacy Survey first-year data (see section 10), figure 8.1 shows the first 2000 chain values of Ω_M (see chapter 9, section 9.4 for a definition of Ω_M) and the corresponding 1-dimensional marginalised likelihood of Ω_M for the case in which w_0 is fixed. The best-fit parameter along with the upper and lower errors for Ω_M is quoted as $\Omega_M = 0.277^{+0.092}_{-0.077}$.

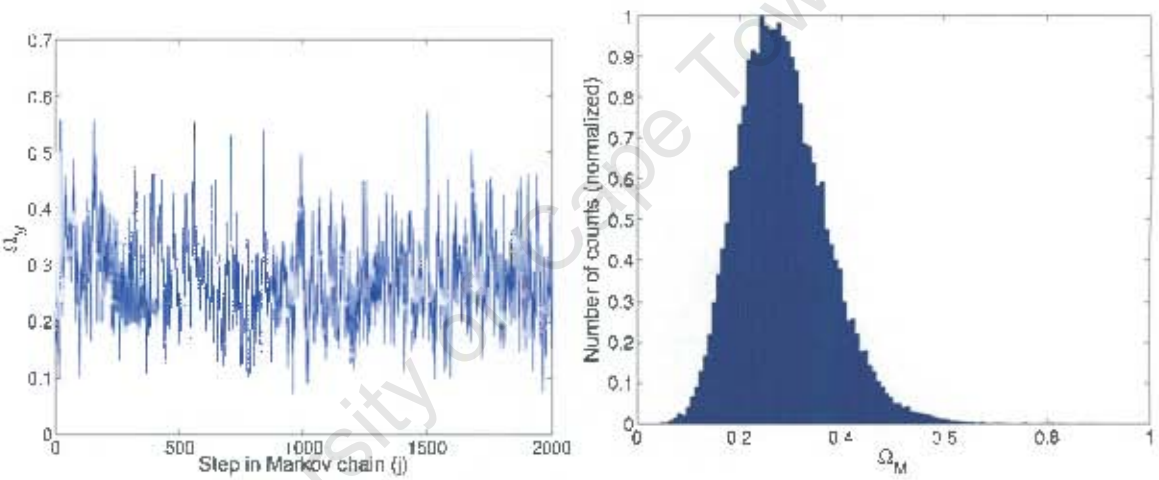

Figure 8.1: Parameter values (left) and likelihood function (right) of Ω_M for the case where Ω_M and H_0 are free while $w_0 = w_A = -1$ is fixed. Best fit parameter is $\Omega_M = 0.277^{+0.092}_{-0.077}$. The analysis is carried out on the SNLS first-year data (see section 10).

Chapter 9

Cosmological Theory

This chapter deals with the cosmological theory that builds up to the equations that we use to perform the parameter estimation. The cosmological equations that govern the evolution of our Universe as a whole as well as its constituent components are presented. Thereafter some of the parameters that we estimate are defined and then, finally, a redshift dependent form of the luminosity distance of an object is given. This equation will contain the free cosmological parameters that we will be adjusting so as to minimise the discrepancy between our theoretical predictions of the luminosities and the true observed luminosities.

9.1 The Friedmann Equation

Cosmologists adopt the cosmological principle which states that on sufficiently large spatial scales the Universe is both homogeneous and isotropic. Homogeneity implies that the Universe is the same at any point at constant time while isotropy implies that is looks the same in any direction. On small enough scales the assumption breaks down (we observe complex filamentary large scale galaxy structure in the nearby Universe, for example). The cosmological principle is the starting point for a certain set of solutions to Einstein's theory of gravity, known as Friedmann cosmologies.

In 1915 Albert Einstein put forward his theory of General Relativity. It is a geometric theory of gravity that describes the way in which objects move through curved spacetime and the way in which spacetime is affected by its energy-mass content. In particular, the spacetime metric is used to describe the curvature of spacetime while the stress-energy tensor describes the density and flux of energy and momentum in space. Einstein's Field Equations then relate the metric to the stress-energy tensor.

Friedmann, Robertson and Walker were three scientists who studied the solutions to the field equations under the assumption of a homogeneous and isotropic Universe. We use the FRW metric[1] to describe a Universe that is homogeneous and isotropic as well as either expanding or contracting. The spacetime interval, using the FRW metric, may be written as

$$ds^2 = -dt^2 + a^2(t)\left[\frac{dr^2}{1-kr^2} + r^2(d\theta^2 + \sin^2\theta d\phi^2)\right], \tag{9.1}$$

where t is the time variable, r, θ and ϕ are polar spatial coordinates. The constant spatial curvature is described by $k = 0$, $+1$, or -1 for a flat, closed or open Universe respectively. The FRW solutions

[1] A metric tensor for a certain curved space allows us to determine intervals in that space. In terms of GR, the FRW metric allows one to determine time and space intervals in a homogeneous and isotropic Universe.

to the field equations are extremely important to cosmologists since, using them, it is possible to understand how the Universe expands with time. The first equation that allows us to do this is the Friedmann equation which may be written as

$$\left(\frac{\dot{a}}{a}\right)^2 = H^2 = \frac{8\pi G}{3c^2}\rho - \frac{k}{a^2c^2}, \tag{9.2}$$

Let's take a moment to explain exactly what all the symbols represent.

1. $a(t)$ is the time dependent scale factor. It is a dimensionless quantity that describes the 'size' of the Universe. It is usually normalised so that it has a value of $a_0 = 1$ today and a value of $a = 0$ at time $t = 0$. $\dot{a}$ is then clearly the speed of the expansion of the Universe while $\ddot{a}$ is the acceleration.

2. c is the speed of light in vacuum.

3. H is the Hubble parameter and is equal to $\dot{a}/a$.

4. G is the usual gravitational constant.

5. ρ is the total mass density of the Universe. It may be decomposed into various contributions: (1) non-relativistic (simply referred to as the matter component from here on), (2) dark energy (of which the cosmological constant Λ is a special case) and (3) relativistic. The matter component of the Universe we take to be made up of baryonic matter and dark matter. Dark energy is the dominant component today whose properties we would like to quantify. It acts as a kind of anti-gravity, fueling the accelerated expansion of the Universe. Finally, the radiation contribution to ρ is made up of photons and neutrinos (at least in the early Universe). Today the Universe is dominated by contributions (1) and (2) [8, 9] and so for the purpose of this investigation we consider only these two contributions: $\rho = \rho_M + \rho_{DE}$.

6. k is the already mentioned curvature constant describing the spatial curvature of our Universe. k = 0, -1, +1 for a flat, closed and open Universe respectively. The term k/a^2c^2 may be thought of as an energy component of the Universe with equation of state parameter (defined below) equal to $-1/3$. For a particular component, -1/3 is the critical value of w that produces zero acceleration of the expansion in the absence of all other forms of energy. Usually, only for $k = 0$ can we demand that $a_0 = 1$. In a non-flat Universe there is a natural length scale given by the curvature radius and so we cannot set $a_0 = 1$ in general..

For the purpose of this investigation we limit ourselves to the case of a flat Universe (k=0) in which case the Friedmann equation reduces to

$$\left(\frac{\dot{a}}{a}\right)^2 = H^2 = \frac{8\pi G}{3c^2}\rho. \tag{9.3}$$

This is accurate to a good approximation given the WMAP results [8, 9].

9.2 The Acceleration and Energy Conservation Equations

The Friedmann equation cannot, all by itself, tell us how the scale factor evolves with time. It is a single equation in two unknowns, a(t) and $\rho(t)$, and so in order to solve for these quantities as functions of time we need $\rho(t)$. The second key equation is the energy conservation equation:

$$\dot{\rho} + 3\left(\frac{\dot{a}}{a}\right)\left(\rho + \frac{P}{c^2}\right) = 0. \tag{9.4}$$

One of the reasons this equation is important is because it relates the total pressure/density of the Universe to the scale factor via the equation of state parameter. The equation of state parameter is the dimensionless scalar w, equal to the ratio of a fluid's pressure to c^2 times its energy density, that characterises the perfect fluid:

$$w \equiv \frac{P}{\rho c^2}. \tag{9.5}$$

Combining (9.4) and (9.5) yields

$$\dot{\rho} + 3\left(\frac{\dot{a}}{a}\right)\rho\,(1+w) = 0. \tag{9.6}$$

Using the above equation one may show (see appendix B) that the density and equation of state parameter of a perfect fluid are related by

$$\rho(a) \propto a^{-3(1+w)}. \tag{9.7}$$

By combining the Friedmann and energy conservation equations, we can derive (see appendix B) an acceleration equation that tells us how the expansion of the universe speeds up or slows down with time, depending on the fluid density. The usual form of the acceleration equation is

$$\left(\frac{\ddot{a}}{a}\right) = -\frac{4\pi G}{3}(\rho + 3P). \tag{9.8}$$

P is the pressure of the perfect fluid that we take to fill the Universe. Again, P may be decomposed into various contributions. Collectively, equations (9.3), (9.4) and (9.8) are often referred to as the cosmological equations.

9.3 Implications of the Cosmological Equations

Cosmologists speak of the dark energy as behaving like anti-gravity: it forces masses to repel each other rather than attract one another. By considering the acceleration equation we can see how this could come to be. Rewriting the Acceleration equation in terms of w we have

$$\left(\frac{\ddot{a}}{a}\right) = -\frac{4\pi G}{3}\rho(1+3w). \tag{9.9}$$

Non-relativistic matter is essentially pressureless so that $w_M \simeq 0$. Substituting $w = 0$ into (9.9) we have $\ddot{a} < 0$. Thus the effect of baryons and dark matter is actually to decrease the expansion simply because of the usual gravitational attraction of the matter. Furthermore, the density of matter is diluted by the expansion as $\rho \propto a^{-3}$ according to (9.7). A Universe filled with enough non-relativistic matter would eventually stop expanding and then collapse back in on itself. As a point of interest, radiation has $w = 1/3$ and so again $\ddot{a} < 0$. For radiation, however, $\rho \propto a^{-4}$ and so, although the Universe was radiation dominated at very early times, the radiation density drops off much quicker with a than the matter density and, as we shall show, the dark energy density. The reason for the extra dilution factor of $1/a$ is because, besides being diluted by the increasing volume of the Universe, the radiation is also redshifted with the expansion[2]. For the purposes of this project the most interesting case occurs when $w < -1/3$. Clearly, if $w < -1/3$ then $\ddot{a} > 0$, i.e. the expansion is accelerated. Observations that tell us that the expansion is accelerating [35, 36, 46] allow us to infer the existence of a component with $w < -1/3$, the so-called dark energy, as a possible explanation for the acceleration.In this thesis, we estimate the best equation of state parameter w_0 for the dark energy, together with other cosmological parameters, using type Ia supernova data.

[2]A single photon has energy $E = h\nu = h\frac{c}{\lambda} \propto 1/a$, where ν and λ are the frequency and wavelength respectively.

9.4 Cosmological Parameters

Certain cosmological parameters determine the dynamics of our Universe. One set of such parameters is the densities of the various components since they affect the expansion rate. The total amount of matter/energy in the Universe determines the geometry of the Universe. For a spatially flat Universe we may define a critical density ρ_c above which the space will be positively curved and below which it is negatively curved. ρ_c is defined by the equation

$$H^2 = \frac{8\pi G}{3c^2}\rho_c \tag{9.10}$$

$$\Rightarrow \rho_c = \frac{3c^2H^2}{8\pi G}. \tag{9.11}$$

We may then express the density of any particular component relative to this critical density. This ratio, Ω_i for the $i-th$ component, then relates to us how dominant a particular component is:

$$\Omega_i \equiv \frac{\rho_i}{\rho_c}. \tag{9.12}$$

For a Universe containing only non-relativistic matter and dark energy we therefore have

$$\Omega_{NR} + \Omega_{DE} \div \Omega_k = 1 \tag{9.13}$$

where $\Omega_k = -kc^2/\dot{a}^2$ represents the spatial curvature.

Other important cosmological parameters that have already been mentioned are the Hubble parameter H_0 which describes the expansion rate of the the Universe today, as well as the equation of state parameter w_i of a particular component. In this investigation we are interested in the equation of state parameter of dark energy specifically and so Ω_M, H_0 and w_{DE} are the subset of cosmological parameters we estimate.

9.5 Luminosity Distance

Having discussed the equations and parameters that determine the basic properties and dynamics of our Universe we may now take a look at how they affect a particular physical observerable that is key to our analysis–the luminosity distance of an object. There are several distance measures that are used by cosmologists, all related by simple redshift factors [1]. The one that we use use in our analysis is the luminosity distance D_L defined by

$$F = \frac{L}{4\pi D_L^2} \tag{9.14}$$

$$\Rightarrow D_L = \sqrt{\frac{L}{4\pi F}} \tag{9.15}$$

where L is the intrinsic luminosity of a source and F its apparent flux. We use the luminosity distance since we measure the fluxes of supernovae.

From the Friedmann equation it is clear that H depends on the density of the various energy components which in turn vary with z and so, in general, H varies with redshift: $H = H(z)$. For our purposes we would therefore like a redshift dependent version of the Hubble parameter that reflects this dependence. Following Peebles (1993, pp310-321), we state, for a flat Universe with non-relativistic matter and dark energy, this dependence as

$$H(z) = H_0 E(z). \tag{9.16}$$

The function $E(z)$ is defined as

$$E(z) = \sqrt{\Omega_M(1+z)^3 + \Omega_{DE} \times f(z)} \tag{9.17}$$

where Ω_M and Ω_{DE} are the relative densities of non-relativistic matter and dark energy today respectively. H_0 is the current value of the Hubble parameter. The factor of $(1+z)^3$ in the first term is consistent with (9.7) which tells us that $\rho_M \propto a^{-3}$ (The relationship between z and a is $a/a_0 = 1/(1+z)$). The unspecified form of the factor $f(z)$ in the second term illustrates that the redshift dependence of the dark energy density is unknown. The reason for this is that we have not yet specified an equation of state parameter for the dark energy. In chapter 10, section 10.2 we will specify the functional form of f using (9.7).

Once we have $H(z)$ the luminosity distance of an object at redshift z in a flat FRW model is then given by

$$D_L(z) = (1+z)\frac{c}{H_0}\int_0^z \frac{dz'}{E(z')}. \tag{9.18}$$

The reader is referred to [1] for a derivation of this result. Equation 9.18 is the key equation that we will use in our analysis. It relates the luminosity distance of a source at redshift z, via the Hubble parameter, to the matter and dark energy content of the Universe. Having an expression for the luminosity distance D_L, we use it to define the distance modulus μ which is another common distance measure used by astronomers. Given the apparent and absolute magnitudes of an object, m and M respectively, the distance modulus is defined by

$$\mu \equiv m - M - K \tag{9.19}$$

where K is the K-correction [47] which allows us to account for the fact that sources observed at different redshifts are sampled at different rest-frame frequencies. We set $K = 0$[3] for simplicity so that $\mu = m - M$. The distance modulus is then related to the luminosity distance by

$$\mu = 5\log\left(\frac{D_L}{10pc}\right) \tag{9.20}$$

where pc stands for parsec. Figure 9.1 shows the behavior of D_L normalised with respect to the Hubble distance D_H ($D_H \equiv c/H_0$) for various cosmologies with different relative densities of matter and dark energy.

9.6 Type Ia Supernovae

Astronomers have always struggled to probe cosmological length scales. The task is a difficult one to work around. One may wonder why it is not simply a matter of measuring the redshift of an object and then using Hubble's Law to relate the recessional velocity to its distance. The answer is that Hubble's law, which predicts a linear relationship between the two observerables, is only accurate for low redshifts [1].

An object whose intrinsic luminosity is known is referred to as a standard candle. By comparing the known intrinsic luminosity of an object to its apparent flux on the sky we may work out the object's distance. Type Ia supernovae (SNIa), which have already been mentioned, are a particular type of standard candle. This section is dedicated to a brief discussion of why and how these objects may be used to probe cosmological distance scales and hence infer important dynamics of the Universe. For a more definitive discussion of the subject the reader is referred to the review by Perlmutter and Schmidt [48].

[3] It may not be entirely valid to set the K-correction to zero when analysing the SNLS data yet we simply do so since we are interested mainly in testing how well the analysis pipeline works. A non-zero K-correction term could easily be added in a later version of the code.

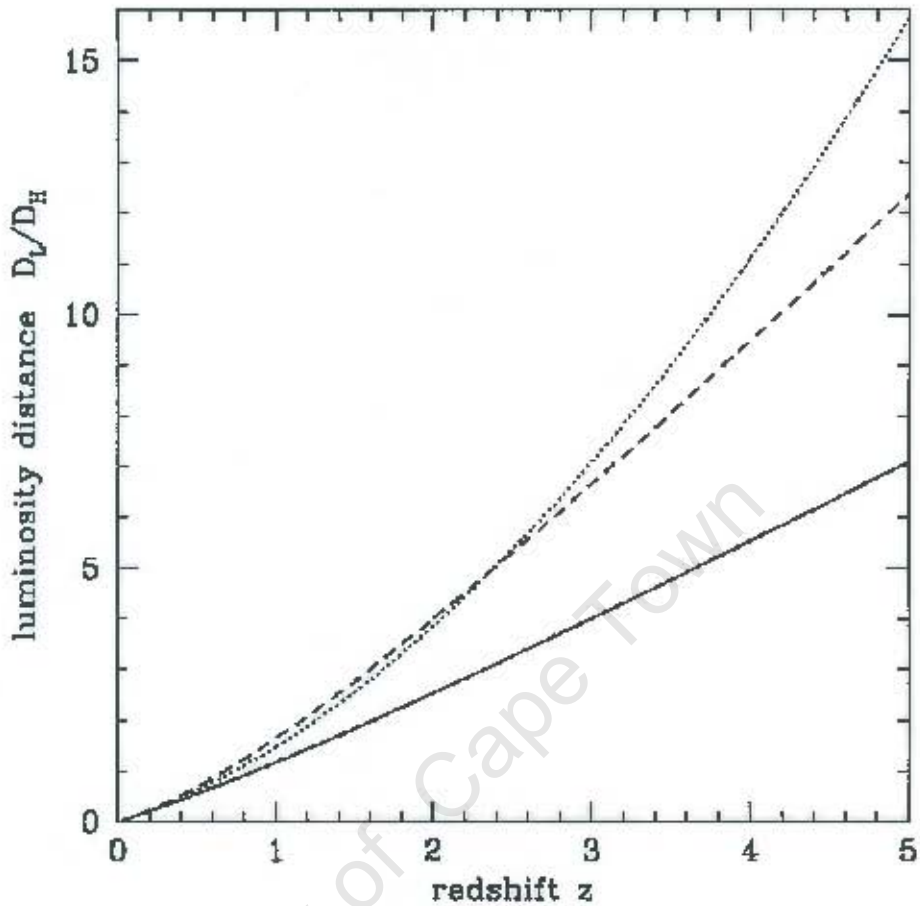

Figure 9.1: The dimensionless Luminosity distance D_L/D_H. The three curves correspond to cosmologies with: (Ω_M, Ω_{DE})=(1, 0), solid; (0.05, 0) (an open Universe), dotted; and (0.2, 0.8), dashed. Taken from [1].

The commonly accepted theory of SNIa is that of a binary system in which mass is transferred from a companion onto a white dwarf until the critical 1.4 solar mass Chandrasekhar limit is reached at which point the supernova explosion occurs. Since the Chandrasekhar limit is well defined and no supernova remnant is left, all SNIa have about the same explosion energy and therefore similar peak luminosity. The energy released in the explosion obviously greatly increases the luminosity of the supernova and it is this explosion that we observe from Earth or space.

In the 1990's it was realised that SNIa could be used as good distance indicators for a number of reasons [46]. They are extremely luminous, thereby allowing us to probe truly cosmological distance scales. There is a small dispersion among their peak luminosities. This is an extremely useful feature since it means that, on average, the fainter a SNIa appears to us, the further away it is. Little temporal evolution is expected in the peak luminosities since we do not expect the Chandrasekhar mass limit to evolve. Lastly, since it is expected that there is little temporal variation in these objects, we can compare different SNIa in different environments to test for various complications such as host galaxy contamination etc.

Although there is some scatter in the peak luminosities of SNIa, there is a way for us to determine, observationally, what the peak intrinsic luminosity of a SNIa is simply by examining its light curve. Phillips (1993)[49] found good evidence for a correlation between peak luminosity and the rate of decline of the light curve. Generally, intrinsically bright SNIa have slower decline rates than intrinsically

dim SNIa with the correlation being found to be stronger in B than in V or I. The parameter used by Phillips to characterise the decline rate is $\Delta m_{15}(B)$ and measures the the total drop in B-band magnitudes 15 days after B-band maximum. Basically the significance of Phillips' discovery is as follows: The rate at which the light dims from maximum is a pointer to how bright the SNIa was at that maximum. By measuring the decay rate, we may infer the peak intrinsic luminosity and hence calculate the distance to the SNIa.

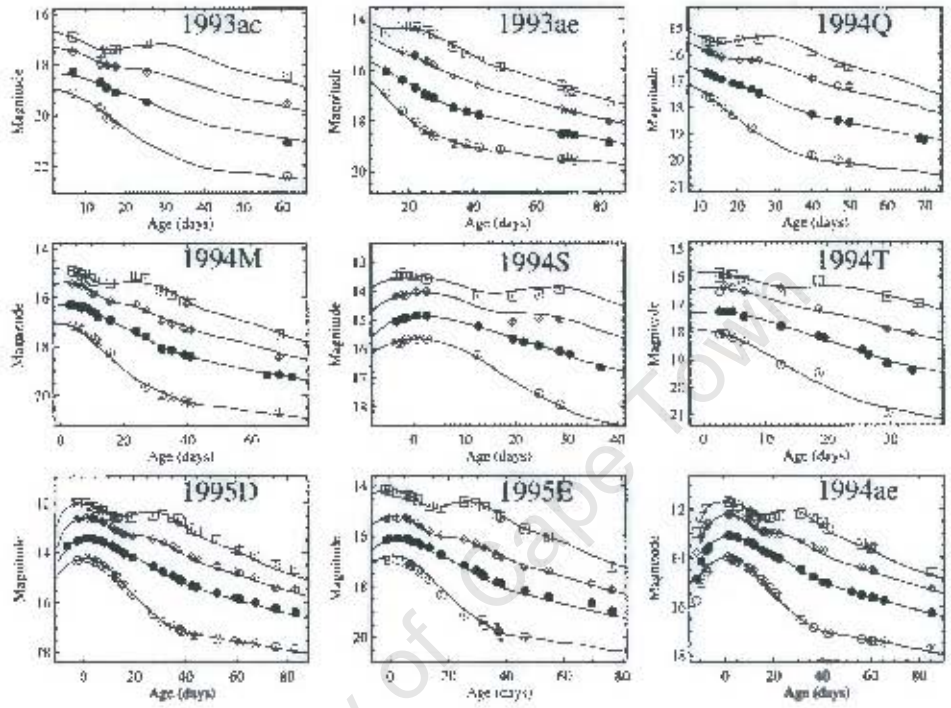

Figure 9.2: B, V, R, and I light curves of 9 SNe Ia. On the x-axis is plotted the number of days before or after maximum brightness. Notice the fast rise to peak brightness and then the slower decline thereafter. From [2]

In the 1990's two major teams were involved with supernova surveys. In 1998 both teams released results that flipped our understanding of the Universe on its head. The two teams were The 'Supernova Cosmology Project'(SCP[4]), led by Saul Perlmutter and the 'High-Z Supernova Search Team' (HZT[5]), led by Brian Schmidt. At the time it was generally believed that the expansion of the Universe should be slowing down due to the gravitational attraction of its non-relativistic matter content. The results released by SCP [36] and HZT [50] indicated that the Universe has been in a state of accelerated expansion during the last few billion years. See [51, 52, 53].

[4]http://supernova.lbl.gov/
[5]http://supernova.lbl.gov/

Chapter 10

Application to the SNLS data

Having discussed both the theory of parameter estimation as well as some of the cosmological theory, we study the Supernova Legacy Survey (SNLS) first-year data [4]. We attempt to determine the best fit values of Ω_M, H_0 and w_0 for a flat Universe. We state the results and then compare them with the corresponding results from Astier et al [4]. As was mentioned in the introduction, we aim to develop and test the analysis pipeline that will be used to analyse the SDSS-II first-year data. In this chapter we simply test the pipeline on the SNLS data and state the results.

10.1 The Supernova Legacy Survey First-Year Data

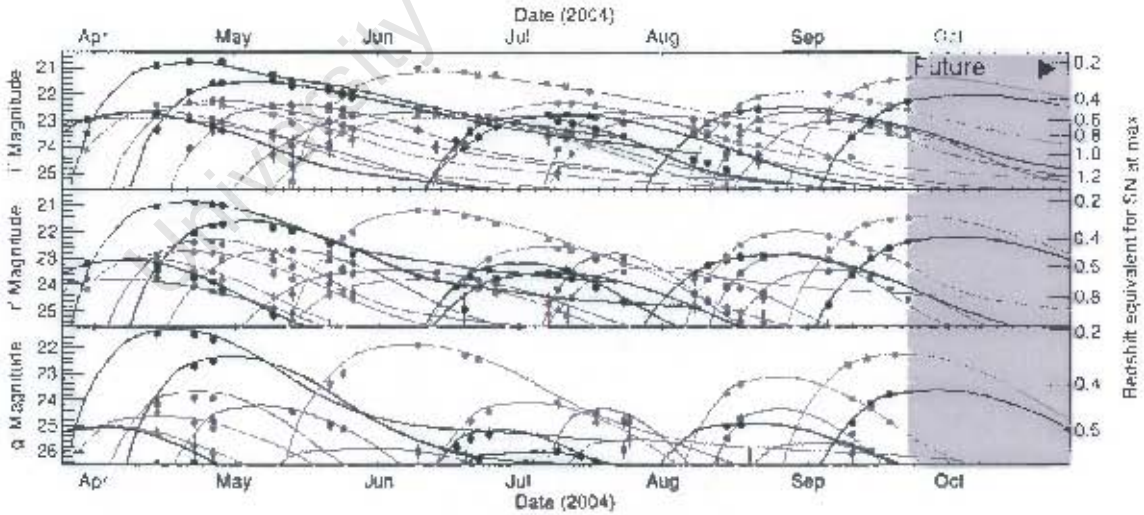

Figure 10.1: Some lightcurves in the g r i filters for confirmed SNIa discovered during 6 months in 2004, demonstrating the quality of the lightcurves produced by the SNLS. The figure includes only about a third of the SNIa that were monitored during this period. From [3]

The SNLS is one of the current surveys dedicated to probing the properties of the dark energy by determining its equation of state parameter, w_0. It is a five year project, which began in August 2003, that hopes to deliver ~ 1000 SNIa detections with light curves sampled in four pass bands. Using such a data set, the survey collaborators hope to determine the dark energy equation of state parameter to

high statistical precision order to perhaps exclude the hypothesis that the dark energy is a cosmological constant. The survey is carried out on the Canada-France-Hawaii Telescope (CFHT) which images in the four SDSS filters (g r i z)[1] but which also, very importantly, uses a very large imager field. 'Megacam' [54] is the square degree imager used on the CFHT to scan four pre-determined fields. High quality continuous light curves can thus be produced for each of the SNIa candidates. Figure 10.1 shows some such light curves. Successful SNIa candidates are followed up spectroscopically to determine their type and redshift.

The SNLS first-year data that we use for our analysis consists of a list of triples (z_i, μ_i, σ_i), i=1, 2, ..., 71, where z_i is the spectroscopically measured redshift of a supernova, μ_i its distance modulus and σ_i the error in the distance modulus. The data may be found in appendix C. At this stage what we can do is to assume a ΛCDM cosmology ($w_\Lambda = -1$) for a flat Universe ($\Omega_M + \Omega_\Lambda = 1$), fix the Hubble constant, and then produce a theoretical plot of $\mu(z)$ Vs. z for a specified value of Ω_M. Along with this theoretical curve we could also plot the μ's corresponding to each of the SNLS SNIa redshifts and then check for agreement. When the dark energy is a cosmological constant, (9.17) becomes

$$E(z) = \sqrt{\Omega_M(1+z)^3 + \Omega_\Lambda}. \tag{10.1}$$

Setting $\Omega_M = 0.27$ (and hence $\Omega_\Lambda = 0.73$) and $H_0 = 72$ km/s/Mpc produces the solid curve shown in figure 10.2. For comparative purposes the dashed curve corresponding to a totally matter dominated cosmology ($\Omega_M = 1$) is also plotted. The plot suggests that the data is consistent with a flat ΛCDM model. At this stage we do not bother calculating χ^2.

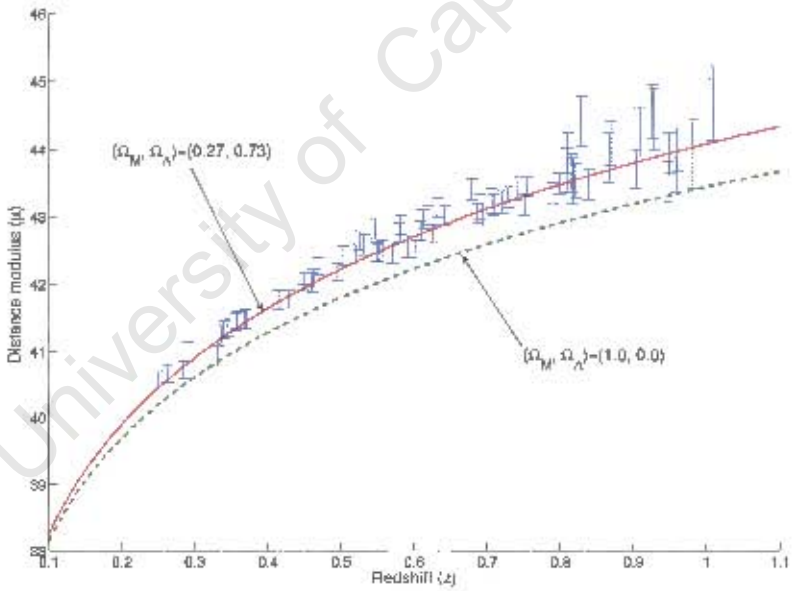

Figure 10.2: Plot of (1) theoretical (solid/red curve) $\mu(z)$ for a flat ΛCDM model with $\Omega_M = 0.27$ (and hence $\Omega_\Lambda = 0.73$) and $H_0 = 72$ km/s/Mpc and (2) theoretical (green/dashed curve) $\mu(z)$ for a flat, non-accelerating matter dominated cosmology with $\Omega_M = 1$ and (3) the observed distance moduli of 71 SNIa from the SNLS. A matter dominated Universe would clearly not have SNIa as distant as we observe them to be.

The reader should be made aware of the lack of a low z SNIa sample in this thesis. There exists an arbitrary additive constant which appears in the expression for the distance modulus. This nuisance

[1] The g, r, i, z filters correspond to green (4770Å), red (6231Å), near-infrared (7625Å) and infrared (9134Å) passbands respectively.

parameter is completely degenerate with H_0 and so a high z SNIa sample such as the SNLS data should actually be combined with a low z sample to estimate H_0. As an illustration of this effect, the lower dashed curve in figure 10.2 could be shifted upwards by adding this arbitrary constant, thereby allowing for a better fit between a $(\Omega_m, \Omega_\Lambda) = (1.0, 0.0)$ cosmology and the SNLS data. Since this SNLS application is intended primarily as a test of the analysis pipeline this is not a serious problem for this thesis.

For our analysis we attempt to minimize χ^2 with respect to only the three cosmological parameters Ω_M, H_0 and w_0 whereas the SNLS team minimize χ^2 with respect to some cosmological parameters as well as three other parameters that appear in their definition of the distance modulus μ. The extra parameters they use are related to their parameterisation of the SNIa light curves. More specifically, the SNLS team use the following form of the distance modulus:

$$\mu_B = m_B^* - M + \alpha(s-1) - \beta c \tag{10.2}$$

where m_B^*, s and c are derived from the fit to the light curves (s and c are shape and colour parameters respectively) and α, β and the absolute magnitude M are parameters which are fitted by minimising the residuals in the Hubble diagram. Furthermore, in the denominator of their expression for χ^2 they include a σ_{int}^2 term which is the intrinsic dispersion of the SNIa absolute magnitudes. The reader is referred to [4] for a full discussion of this analysis. Astier et al. also consider non-flat models whereas we limit ourselves to the analysis of only flat cosmologies. These are the ways in which our analysis differs from the SNLS team's analysis. Nevertheless we compare our results with their's. The results of the SNLS team's analysis with cosmological constant for dark energy are summarized in figures 10.3 and 10.4. The results seem to favour a flat cosmology although we state explicitly their results only when we present ours.

10.2 Estimating Ω_M, H_0 and w_0.

Having discussed the background theory as well as having introduced the SNLS we now apply the method of MCMC parameter estimation to the SNLS first-year data. We consider a flat Universe, as suggested by the WMAP results [8, 9], which has a dark energy component that has a constant equation of state parameter, w_0. (9.7) then becomes

$$\rho_{DE} \propto a^{-3(1+w_0)} = (1+z)^{3(1+w_0)}. \tag{10.3}$$

Using $w = w_0$ our expression (9.18) for the luminosity distance becomes

$$D_L = (1+z)\frac{c}{H_0}\int_0^z \frac{dz'}{\left[\Omega_M(1+z')^3 + \Omega_{DE}(1+z')^{3(1+w_0)}\right]^{0.5}}. \tag{10.4}$$

Thus when we perform our analysis, we vary Ω_M, H_0 and w_0 in order for (10.4) to be able to produce D_L's that match, as closely as possible, those of the data and therefore yield low χ^2 statistics.

When estimating Ω_M, H_0 and w_0 we consider 3 separate cases. Each case has a different combination of free and fixed parameters:

1. **Case 1:** Estimate Ω_M and H_0 for a cosmological constant ($w_0 = w_\Lambda = -1$) in which case the form of $E(z)$ is given by (10.1).

2. Case 2: Estimate Ω_M and w_0 with fixed $H_0 = 70$ km/s/Mpc.

3. **Case 3:** Estimate all three parameters Ω_M, H_0 and w_0.

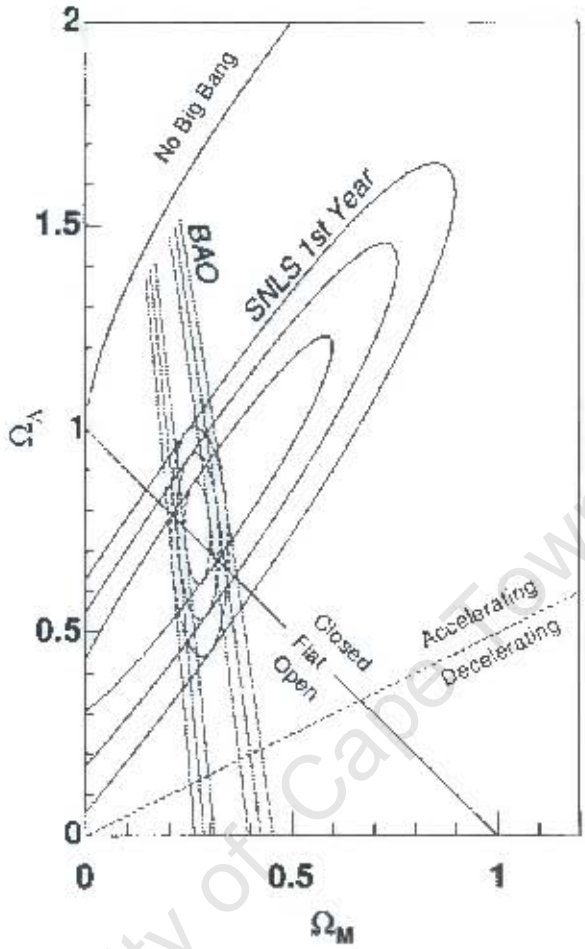

Figure 10.3: Contours at 68.3%, 95.5% and 99.7% confidence levels for the fit to an (Ω_M, Ω_Λ) cosmology from the SNLS plot of μ Vs. z (figure 10.2, solid contours), the SDSS baryon acoustic oscillation (Eisenstein et al, 2005, dotted lines), and the joint confident contours (dashed lines). From [4].

As is always good practice when writing a program, the code should be tested on some special/artificial data set (usually created by the programmer) which allows the programmer to check whether the code behaves as expected. The procedure is no different in this analysis. For each of the cases, a fake data set was created using specific values of the parameters. As an example we may consider case 3. Ω_M, H_0 and w_0 were set to 0.27, 72.0 km/s/Mpc and -1.3 respectively and then, for evenly spaced redshifts from 0.01 to 1.5, D_L's were produced using (10.4) and then converted to μ's. Together with each μ an error term was also created. The error term is extracted from a Gaussian that is centered on the μ with a standard deviation that is proportional to the redshift so that the errors increase with redshift. This simulates the way in which the distance moduli are less well determined at high redshifts. The code is then run on this test data set. The results it produced using 3 chains with 10^5 steps per chain were $\Omega_M = 0.268 \pm 0.007$, $H_0 = 72.10^{+2.23}_{2.21}$ and $w_0 = -1.314^{+0.176}_{-0.148}$. The estimates of Ω_M, H_0 and w_0 are therefore approximately 0.3, 0.04 and 0.1 standard deviations away from the accepted parameter values respectively, suggesting that the code works well.

Of course we have to mention what priors we use on each of the parameters when running the code. In the runs, a top hat prior is used for each of the free parameters. In other words, each parameter is assumed to fall equally likely at any point between some upper and lower bound. The upper and

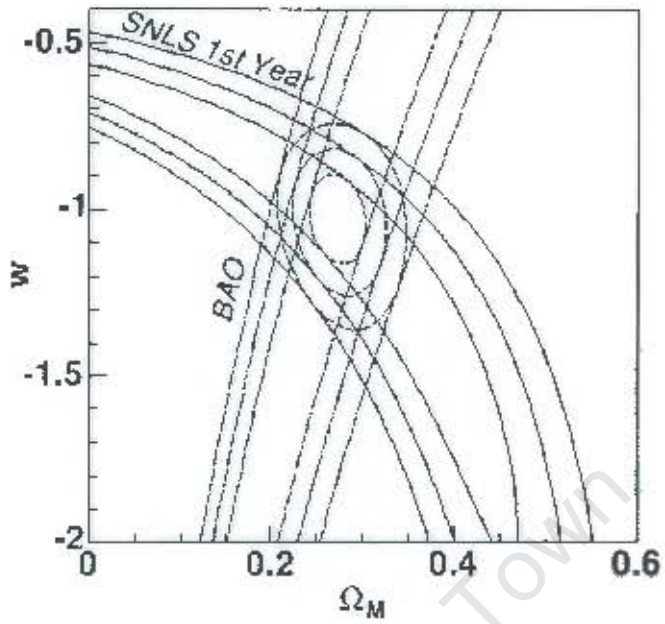

Figure 10.4: Contours at 68.3%, 95.5% and 99.7% confidence levels for the fit to a flat (Ω_M, w) cosmology, from the SNLS plot of μ Vs. z, from the SDSS baryon acoustic oscillations alone (Eisenstein et al. 2005), and the joint confidence contours. From [4].

lower bounds for Ω_M, H_0 and w_0 are the same for each case and are given in table 10.1.

	lower bound	upper bound
Ω_M	0.0	1.0
H_0	0.0	100.0
w_0	-10.0	10.0

Table 10.1: Cosmological parameter top hat (uniform) priors used for each of the three cases in the MCMC parameter estimation process.

For the sake of completeness it should be mentioned that when producing a step $d\vec{\theta} = (d\Omega_M, dH_0, dw_0)$ to attempt, each of the parameter increments are extracted from a Gaussian with zero mean and with a specified variance σ_p where $p = \Omega_M$, H_0, w_0. For all the runs these standard deviations were $\sigma_{\Omega_M} = \sigma_{w_0} = 0.05$, and $\sigma_{H_0} = 3.0$. By evoking the empirical rule for Gaussian distributions we know that the parameter increments will be less than $2\sigma_p$ in magnitude with ~95% probability.

10.3 Results

Cases 1 and 2 consist of 5 and 3 chains respectively with 10^5 steps in each chain. For case 3, in order to obtain better convergence of chains in the higher dimensional space, 5 chains are used with 3×10^5 steps per chain. Figure 10.5 is a graphical representation of the results, it shows the case 2 and 3 results relative to the case 1 results (ΛCDM model). Below we discuss separately the results from each of the cases. These results are summarised in table 10.2.

	Ω_M	H_0	w_0
Case 1	$0.277^{+0.092}_{-0.077}$	$69.60^{+2.234}_{-2.343}$	-1.000
Case 2	$0.336^{+0.112}_{-0.158}$	70.00	$-1.198^{+0.357}_{-0.532}$
Case 3	$0.319^{+0.082}_{-0.068}$	$78.17^{+2.16}_{-8.107}$	$-2.551^{+1.338}_{-1.692}$

Table 10.2: Best fit parameters together with 1σ error bounds for cases 1, 2 and 3. Top hat priors are used for each of the free parameters (see table 10.1).

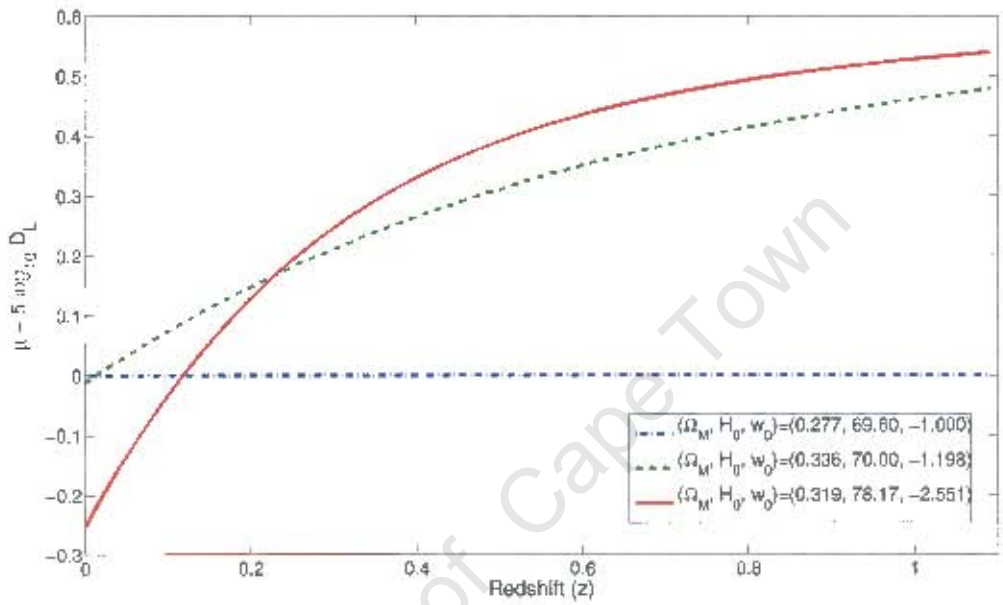

Figure 10.5: Residuals for the best fit to a Λ cosmology (case 1) of the theoretical distance modulus for the best fit parameters from cases 1, 2 and 3.

10.3.1 Case 1: Ω_M and H_0 free, w_0 fixed.

Case 1 results are quite consistent with the standard model of cosmology (ΛCDM model in which the dark energy is a cosmological constant). H_0 is slightly lower, however, than the generally accepted value of 71 km/s/Mpc$^{+0.04}_{-0.03}$ [8, 9] for such a model. The 1-dimensional likelihoods of these two parameters appear as figure 10.6 while the contour plot of the 2-dimensional likelihood surface of Ω_M and H_0 appears as figure 10.7. Notice the degeneracy that runs in the direction from bottom right to top left in figure 10.7. These degeneracies occur when a change in one parameter can be compensated by a change in another so that the overall affect on some function that depends on both the parameters is zero. In our case Ω_M and H_0 both occur in the expression for D_L. A higher value of Ω_M may be compensated by a lower H_0 value and vice versa so that χ^2 for both of the parameter combinations is the same, i.e. the likelihood of a higher Ω_M and a lower H_0 is the same as the likelihood of a lower Ω_M and a higher H_0. To overcome such degeneracies one may produce a contour plot of the likelihood surface for the same parameters using a different data set for which the parameter degeneracy runs in an orthogonal direction. One may then better constrain the parameters by combining the two data sets. As a means of checking that the indivíual burnt-in chains have converged (which we expect from the neat Gaussian histograms) tables 10.3 and 10.4 show the best-fit and error bound estimates for the burnt-in chain along with those of the master chain. Clearly there is high level of agreement.

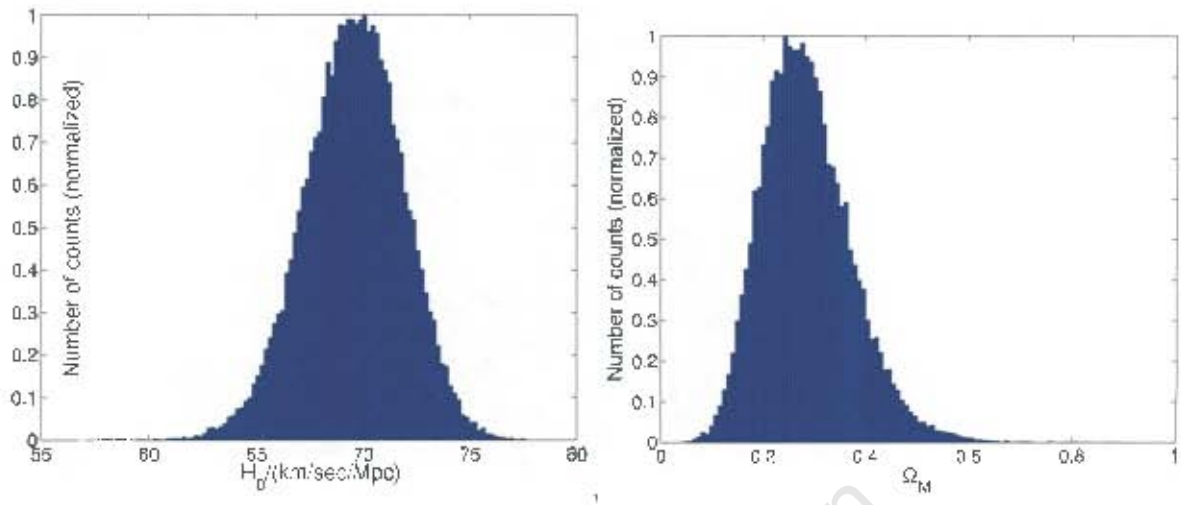

Figure 10.6: 1-dimensional likelihoods of Ω_M and H_0 for case 1 (Ω_M and H_0 free, $w_0 = w_\Lambda = -1$ fixed) which have best fit parameters of $\Omega_M = 0.277^{+0.092}_{-0.077}$ and $H_0 = 69.60^{+2.234}_{-2.343}$. See table 10.1 for priors used on the parameters.

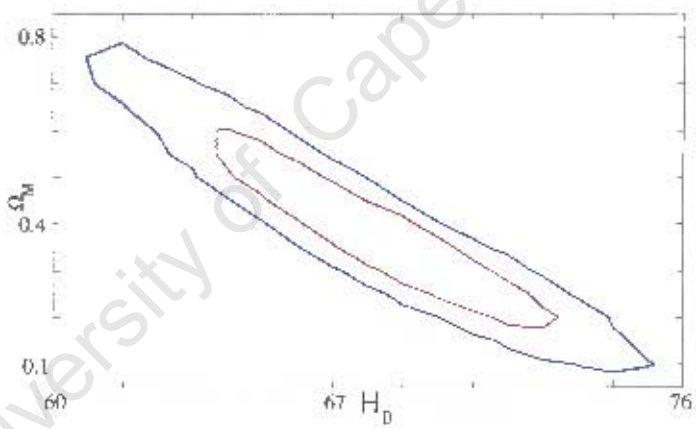

Figure 10.7: The contours at 68% and 95% confidence levels for the 2-dimensional likelihood surface of Ω_M and H_0. Notice the parameter degeneracy extending from the lower right corner to the upper left.

10.3.2 Case 2: Ω_M and w_0 free, H_0 fixed.

The results from case 2 may be compared with the results produced by Astier et al. A detailed comparison may not be made, however, since the SNLS team carried out an intrinsically different type of analysis with more free parameters [4]. Furthermore, the SNLS team have not yet released the error covariance matrices for their data[2]. However, since our main aim is to test the code in preparation for SDSS-II this is not a significant issue. The results produced by our code are Ω_M=$0.336^{+0.112}_{-0.158}$ and w_0 = $-1.198^{+0.357}_{-0.532}$ while the results produced by Astier et al. for their flat w_0 cosmology are Ω_M =0.271±0.021 and w_0 =-1.023±0.087. It should be stated that Astier et al. included the accurately determined baryon acoustic oscillation point, measured in the SDSS, which is very complemen-

[2]In general, if we use a vector $\vec{d}$, say, to represent the difference between observed quantities and the theoretically predicted values then χ^2 is the sum over the entries of $\vec{d}^T C^{-1} \vec{d}$ where C is the error covariance matrix and $\vec{d}^T$ is the transpose of $\vec{d}$. When the data errors are not correlated, C is diagonal and χ^2 reduces to the form used in this thesis.

tary to the SNIa data [55]. The SNLS results favour a flat cosmology when combined with the BAO result. Furthermore, the data set used by Astier et al includes a number of low redshift supernovae ($z \sim 0.05$) which we *do not use* for our analysis since we are mainly interested in testing the code for the SDSS-II pipeline. Such additional data will lead to better parameter constraints.

Looking at the likelihoods of Ω_M and w_0 in figure 10.8 we begin to observe deviations from neat Gaussian distributions. Specifically, the Ω_M distribution is skewed left while the H_0 distribution is skewed right. We would expect a left skewed Ω_M distribution to be accompanied by a right skewed w_0 distribution (or vice versa) simply because of our expression for our defined $E(z)$ function,

$$E(z) \equiv \left[\Omega_M(1+z')^3 + \Omega_{DE}(1+z')^{3(1+w_0)}\right]^{0.5} \tag{10.5}$$

which appears in the denominator of our expression for the luminosity distance. Clearly if Ω_M favours smaller values, thereby decreasing the magnitude of $E(z)$, then larger w_0 values are needed in order to compensate. Therefore a skewed left Ω_M distribution should be accompanied by a skewed right w_0 distribution. A contour plot of the likelihood surface appears in figure 10.9, again with parameter degeneracy.

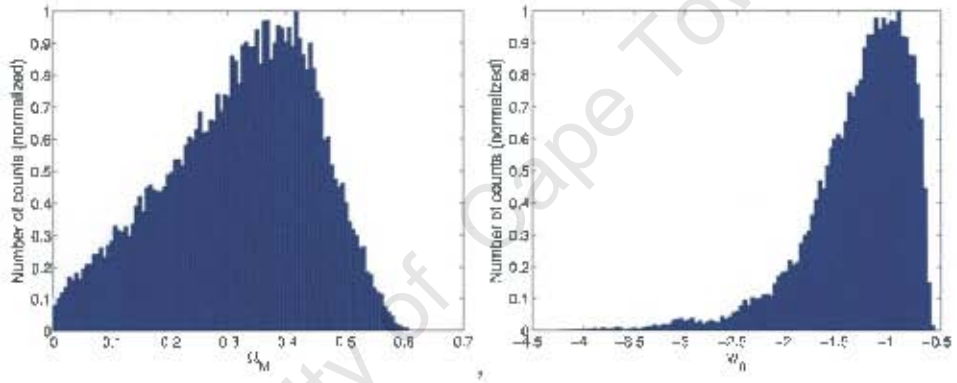

Figure 10.8: 1-dimensional likelihoods of Ω_M and w_0 for case 2 (Ω_M and w_0 free, $H_0 = 70$ km/s/Mpc fixed) which have best fit parameters of $\Omega_M = 0.336^{+0.112}_{-0.158}$ and $w_0 = -1.198^{+0.357}_{-0.532}$. See table 10.1 for priors used on the parameters.

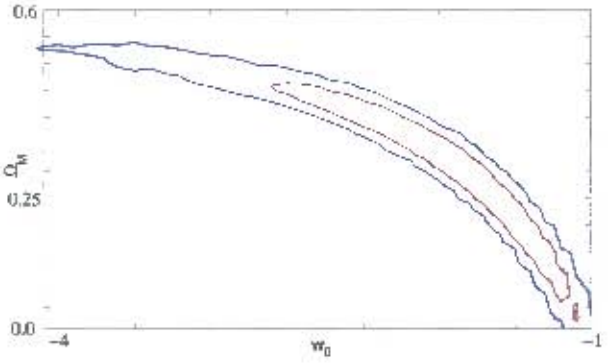

Figure 10.9: The contours at 68% and 95% confidence levels for the 2-dimensional likelihood surface of Ω_M and w_0. Notice how the contours are 'dragged' toward higher w_0 values as we would expect due to the highly skewed w_0 likelihood function.

10.3.3 Case 3: Ω_M, w_0 and H_0 free.

The case 3 results, presented in figures 10.10 and 10.11 are interesting. Ω_M and H_0 are both higher than the generally accepted values [8, 9] and w_0 is low. w_0 falls well within the so-called phantom energy regime in which w_0 <-1 and which ultimately causes the Universe to end in a big rip (since the energy density of the phantom energy actually increases with increasing scale factor). Such a component is unappealing since $w_0 = -1$ is a fixed point for $\dot{\rho}$ in the energy conservation equation (9.4). It is not clear how the cross over from $w_0 > -1$ to $w_0 < -1$ would occur. Looking at the likelihood surfaces of H_0 and w_0 we would have expected higher H_0 values (which decrease D_L) to be accompanied by lower w_0 values (which increase D_L when $w_0 - < 1$). We notice also that the likelihood surface of w_0 no longer has a single maximum but rather 3 or 4 maxima. When the likelihood surfaces of parameters that we are attempting to estimate become complex we have to start considering the MCMC error that is intrinsic to the MCMC method.

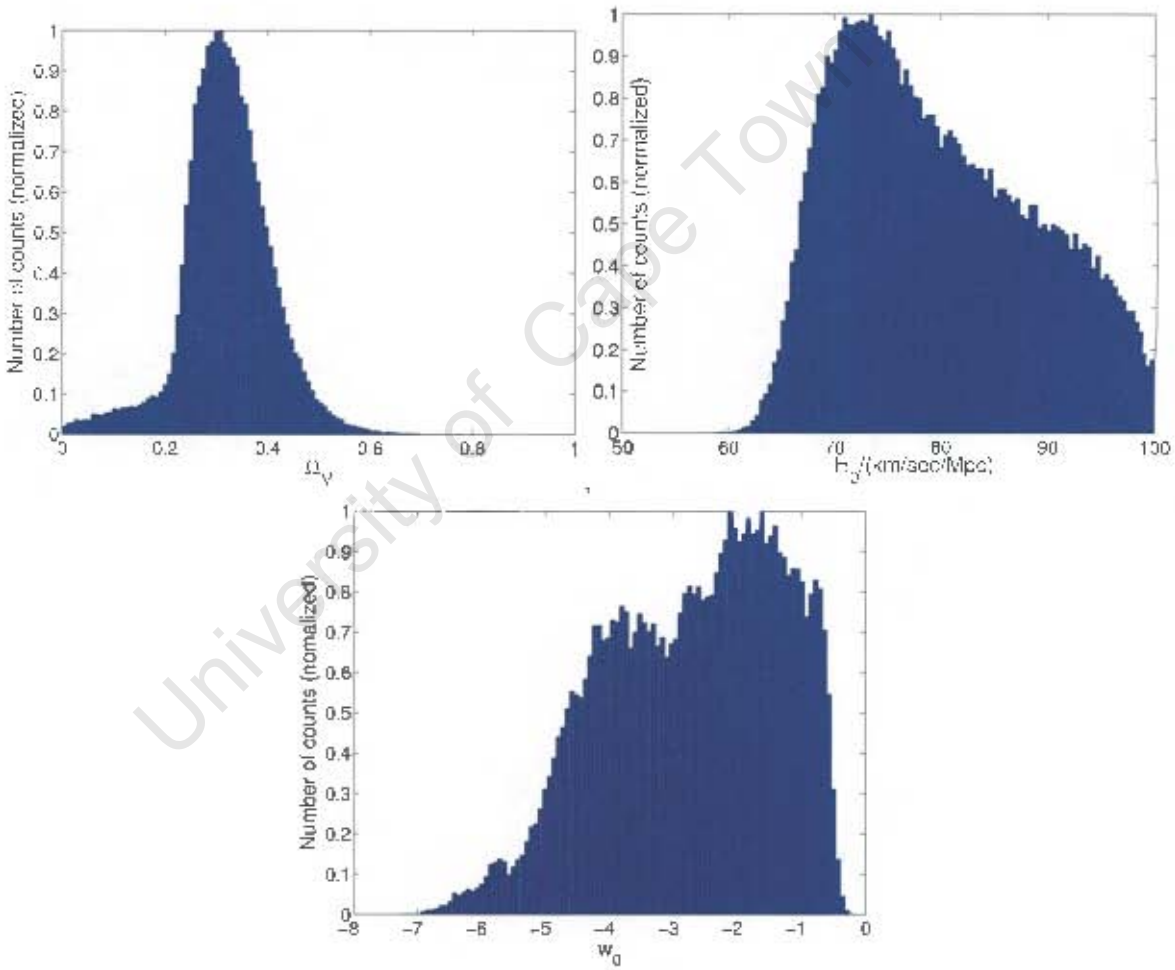

Figure 10.10: 1-dimensional likelihoods of Ω_M, H_0 and w_0 for case 3 (Ω_M H_0 and w_0 free) which have best fit parameters of Ω_M =$0.319^{+0.082}_{-0.068}$, H_0 =$78.17^{+12.16}_{-8.107}$ and w_0 =$-2.551^{+1.338}_{-1.692}$. See table 10.1 for priors used on the parameters. Notice how the highly skewed distributions lead to asymmetric error bars.

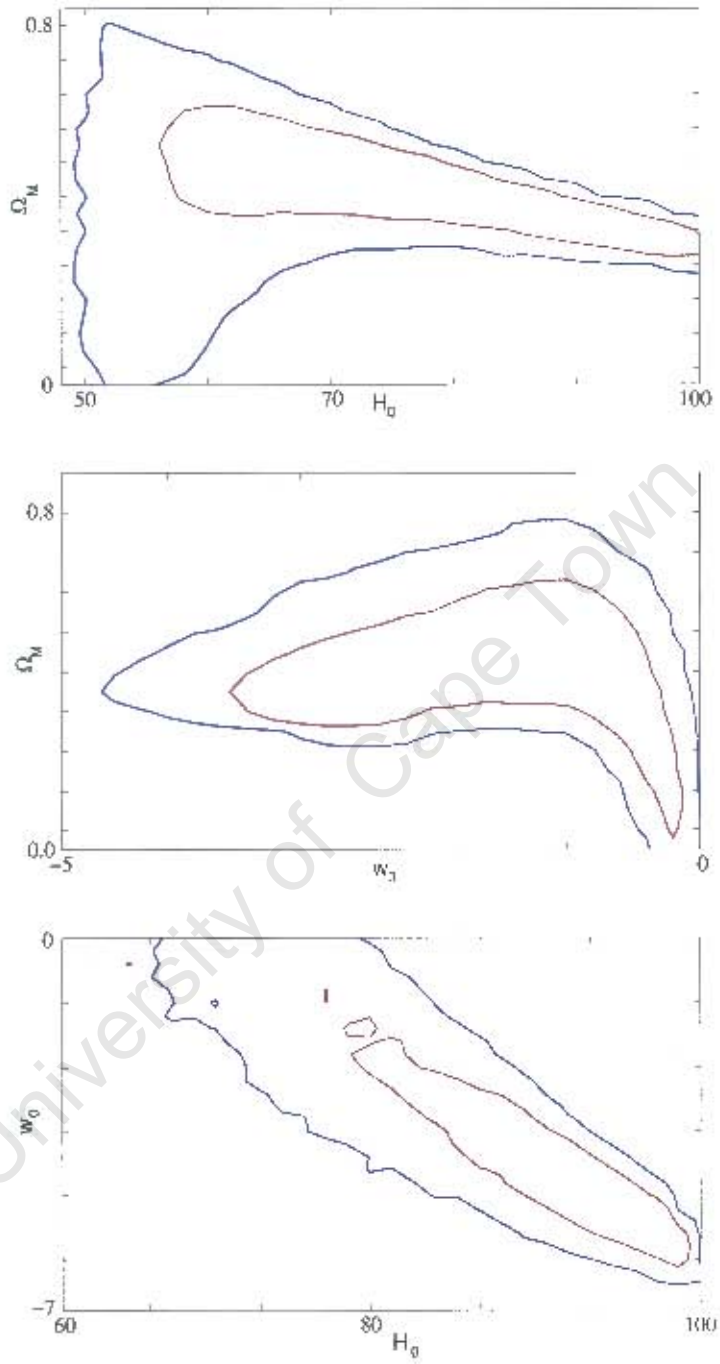

Figure 10.11: The contours at 68% and 95% confidence levels for the 2-dimensional likelihood surface of Ω_M and H_0 (top), Ω_M and w_0 (middle), H_0 and w_0 (bottom).

10.3.4 Convergence of Chains

When performing this type of parameter estimation it is crucial to know whether or not the parameter chains have converged. This obviously depends on the number of free parameters which in turn determines the complexity of the likelihood surface for which the walker should locate the maximum.

As a way of checking for convergence in the above stated results we now look at the best-fit and error bound estimates for each free parameters from each of the burnt-in chains and then compare those results with the corresponding master chain results.

In case 1 there is good agreement between burnt-in chain results and master chain results. We would expect this since fixing $w_0 = -1$ significantly simplifies the form of the luminosity distance expression thereby allowing for a simple likelihood surface that needs to be explored. In case 2 we still observe good agreement between burnt-in chain and the master chain for Ω_m but we begin to notice some small inconsistencies in the w_0 results. This is reflected in the anti-symmetric histogram of w_0 (figure 10.8). Finally in case 3 we begin to notice less agreement for H_0 and w_0 results. This is clearly due to the much more complex likelihood surface created by including all three free parameters in D_L. Even though there is the least agreement between burnt-in chain and master chain results in this third case, the results from the burnt-in chains always agree within 1 σ with the master chains. Hence while conververgence could no doubt be better in this case it suffices for the purposes of this thesis.

Ω_m	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	0.286	-0.078	0.093
burnt chain 1	0.288	-0.078	0.096
burnt chain 2	0.286	-0.077	0.090
burnt chain 3	0.283	-0.076	0.093
burnt chain 4	0.283	-0.078	0.093
burnt chain 5	0.289	-0.079	0.095

Table 10.3: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for Ω_m for case 1 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. Each of the burnt-in chains agrees well with the master chain results and so we may confident that each of the burnt-in chains have converged to the correct value.

H_0	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	68.78	-2.286	2.195
burnt chain 1	68.73	-2.330	2.184
burnt chain 2	68.77	-2.207	2.146
burnt chain 3	68.87	-2.295	2.183
burnt chain 4	68.88	-2.324	2.214
burnt chain 5	68.69	-2.287	2.226

Table 10.4: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for H_0 for case 1 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. Each of the burnt-in chains agrees well with the master chain results and so we may confident that each of the burnt-in chains have converged to the correct value.

Ω_m	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	0.363	-0.130	0.097
burnt chain 1	0.357	-0.131	0.096
burnt chain 2	0.361	-0.137	0.097
burnt chain 3	0.360	-0.126	0.109
burnt chain 4	0.355	-0.133	0.099
burnt chain 5	0.379	-0.118	0.088

Table 10.5: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for Ω_m for case 2 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. Each of the burnt-in chains agrees well with the master chain results and so we may confident that each of the burnt-in chains have converged to the correct value.

w_0	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	-1.366	-0.582	0.386
burnt chain 1	-1.340	-0.555	0.375
burnt chain 2	-1.359	-0.568	0.396
burnt chain 3	-1.354	-0.688	0.372
burnt chain 4	-1.336	-0.567	0.378
burnt chain 5	-1.435	-0.577	0.390

Table 10.6: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for w_0 for case 2 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. Most of the burnt-in chains agree well with the master chain results yet we do notice that some (chain 5 in particular) begin to deviate from the master chain results. This is expected if one notices how the histogram of w_0 for case 2 is not nicely symmetric as are the case 1 histograms were. There is still good agreement, however, between the burnt-in chains and master chain results.

Ω_m	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	0.313	-0.060	0.081
burnt chain 1	0.317	-0.064	0.082
burnt chain 2	0.299	-0.048	0.078
burnt chain 3	0.317	-0.059	0.078
burnt chain 4	0.311	-0.064	0.084
burnt chain 5	0.320	-0.065	0.082

Table 10.7: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for Ω_m for case 3 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. Most of the burnt-in chains agree well with the master chain results yet we do notice that some (chain 2 in particular) begin to deviate from the master chain results. This is expected if one notices how the histogram of Ω_m for case 3 is not nicely symmetric as are the case 1 histograms were. There is still good agreement, however, between the burnt-in chains and master chain results.

H_0	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	80.10	-9.810	12.42
burnt chain 1	78.49	-8.470	12.08
burnt chain 2	86.52	-15.78	9.390
burnt chain 3	80.59	-9.201	11.21
burnt chain 4	78.68	-9.242	13.02
burnt chain 5	78.55	-8.331	10.79

Table 10.8: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for H_0 for case 3 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. The burnt-in chain estimates now begin to fluctuate quite significantly about the master chain estimates suggesting that they have not all converged. To overcome this problem one may increase the chain length or incorporate the covariance matrix which will help the walkers to more effectively explore the complex likelihood surfuce.

w_0	best-fit parameter	lower 1σ error bound	upper 1σ error bound
master chain	-2.937	-1.802	1.542
burnt chain 1	-2.646	-1.544	1.263
burnt chain 2	-4.106	-2.008	2.684
burnt chain 3	-3.007	-1.607	1.429
burnt chain 4	-2.748	-1.788	1.543
burnt chain 5	-2.722	-1.265	1.309

Table 10.9: Check for convergence of parameter chains. Above appear the best-fit and error bound estimates for w_0 for case 3 from each of th burnt-in chains as well as the master chain which is used to produce the values in table 10.2. The burnt-in chain results now begin to oscillate quite severly against the master chain results indicating clearly that they have not converged properly. Again, this maybe overcome by incorporating the covariance matrix which will help the walkers to more effectively explore the complex likelihood surfuce. For simplicity, this was not done yet it clearly should be when performing the analysis on the SDSS-II SNIa data.

Chapter 11

Analysis of the Intrinsic Error Associated with Monte Carlo Markov Chains

As promised, this final chapter deals with the intrinsic error associated with the Monte Carlo Markov Chain method. We start this chapter by mentioning the systematic and statistical errors that usually accompany a scientific measurement. Thereafter we introduce the idea of MCMC error and describe the process whereby we attempt to measure the error as a function of chain length. Finally we present the results.

11.1 Systematic and Statistical Errors

As scientists we know that there is almost always some type of uncertainty associated with a measurement. Systematic uncertainty, for one, is as a result of the observational technique. One type of systematic error is calibration error. If, for example, the zero point of a certain measuring device is not correctly set, all measurements would be offset by the zero-point error. All measurements are therefore systematically affected in the same way by the offset. A second type of error which we may identify is statistical error–the error from the data. As an example we may consider weighing the same person a hundred times and calculating the average weight $\bar{x}$ as well as the standard deviation σ of the measured weights. We may then invoke the empirical rule (for a Gaussian distribution) by claiming that we are 68% confident that the person's weight lies somwhere in the interval $\bar{x} \pm \sigma$. It is σ that communicates to us the statistical uncertainty of $\bar{x}$ since if σ is very small then we are confident that most of the measurements lie very close to the mean and vice versa. Clearly the reliability of a measurement is affected by the quality of the data. For a given data set, there is a limit to accuracy with which we may infer certain quantities using statistical tools.

11.2 MCMC Error

Now we consider the intrinsic error associated with the MCMC method. When a scientist states a result, it is usually accompanied by some statistical error estimate, σ_{Ω_M} in the case of Ω_M. What we are investigating in this final chapter is whether or not it is necessary to also quote the MCMC error $\sigma_{\Omega_M^*}$ that represents the error in the best fit value. Including both the error terms, the best fit

parameter for Ω_M should be written as $\Omega_M^* \pm \sigma_{\Omega_M} \pm \sigma_{\Omega_M^*}$. Ω_M is the parameter that we estimate in this chapter in order to probe the MCMC error using the SNLS data described previously.

First we consider the error associated with a single Markov chain that has N steps in it. We allow the walker to start in some random place in the interval $0 < \Omega_M < 1$ and then at each of the N steps in its chain we record the value of Ω_M. Once the chain is complete we will have N values of Ω_M. We may then quote the standard deviation σ_{Ω_M} of these N measurements as the error associated with the mean Ω_M value. We may repeat the process for various values of N, calculate the standard deviation of each chain and then produce a plot of σ_{Ω_M} Vs. N. After a while we would expect σ_{Ω_M} to level off and to stop varying with N. The value that it levels off to tells us the accuracy to which we may estimate the parameter *for the given data set.* A data set with well controlled, small errors will allow us to better estimate parameters than a data set with large errors.

The above paragraph simply discusses the idea of the statistical error associated with a single chain of length N. To put forward the idea of MCMC error we consider running j chains of length N and then looking at how much the mean of each chain fluctuates. In our analysis we let $j = 1000$. For each of the j chains we calculate the mean and then take a look at how the means are distributed, i.e. the error in the mean. Carrying out this process will tell us how variable the results are for a chain of length N, i.e. we want to know how consistently we will obtain the same results if we run a chain many times. Given j means, each for a chain for length N, we again expect them to have a Gaussian distribution. Let's call the standard deviation of this distribution $\sigma_{\Omega_M^*}$ since it tells us how variable the best-fit value is. As before, we may vary N and then plot $\sigma_{\Omega_M^*}$ Vs. N.

Clearly it is quite useless for us to run a chain of length N and then state that the statistical error of the best fit parameter from that chain is σ_{Ω_M} if the MCMC error is larger than σ_{Ω_M}. So how volatile is the MCMC method? Specifically, *what is the critical number of steps in a chain, N, above which the MCMC error is negligably small?* Following Dunkley et al. [45], we may define the so called 'convergence ratio',

$$r = \left(\frac{\sigma_{\Omega_M^*}}{\sigma_{\Omega_M}}\right)^2. \tag{11.1}$$

To ensure that the MCMC error is sufficiently small we require r to be below some cut-off value, suggested to be 0.01 in [45].

11.3 Experimental Method and Results

In order to answer the question posed above, for N=N_1, N_2, ..., N_k, we simply run a single chain of length N and calculate the standard deviation of the parameter values. We then run 1000 chains of length N, for each chain calculate the mean, and then calculate the standard deviation of the 1000 means[1]. This is exactly the process outlined in the section above. The chain lengths we use are N=$\{100, 200, 500, 1000, 10000, 25000, 50000, 100000\}$. We then produce a plot with N on the x-axis and σ_{Ω_M}, $\sigma_{\Omega_M^*}$ and $\sqrt{r}$ on the y-axis. The plot appears in figure 11.1. For each of the values of N that were used, table 11.1 lists σ_{Ω_M}, $\sigma_{\Omega_M^*}$ and $\sqrt{r}$ while figure 11.2 shows how the spread of the 1000 means varies with N. In total we estimated the likelihood of Ω_M more than 10^8 times.

Figure 11.1 suggests that in this case the statistical error is always greater than the MCMC error. This is not surprising in retrospect since the likelihood surface of Ω_M has a single maximum (see figure 10.6) and so for long enough chain lengths convergence is assured. Furthermore, we notice that $r^{1/2} < 10^{-2}$ for $N > 4 \times 10^4$ meaning that when estimating Ω_M using the SNLS data set we should use chains with at least 4×10^4 steps to ensure that the MCMC error is sufficiently small. Figure 11.2 is a more graphic representation of how the variation of Gaussian distributed means decreases roughly as

[1]Although it is preferable to have j being very large, computational resources restrict us to $j = 1000$. This is also the reason we restrict our analysis to a single parameter, Ω_M.

N	$\sigma_{\Omega_M^*}/10^{-3}$	$\sigma_{\Omega_M}/10^{-2}$	$\sqrt{r}/10^{-1}$
100	4.5771319	1.241570	3.686568
200	3.1239716	1.618510	1.930153
500	1.8287925	1.959188	0.933115
10^3	1.2621037	2.010540	0.627744
10^4	0.41452161	1.939560	0.213719
25×10^3	0.26426566	1.914653	0.138023
50×10^3	1.8420511	1.924732	0.957043
100×10^3	0.1264585	1.927086	0.065623

Table 11.1: Tabulated results of the variation of σ_{Ω_M} (statistical error), $\sigma_{\Omega_M^*}$ (MCMC error) and the square root of the 'convergence ratio' with chain length, N.

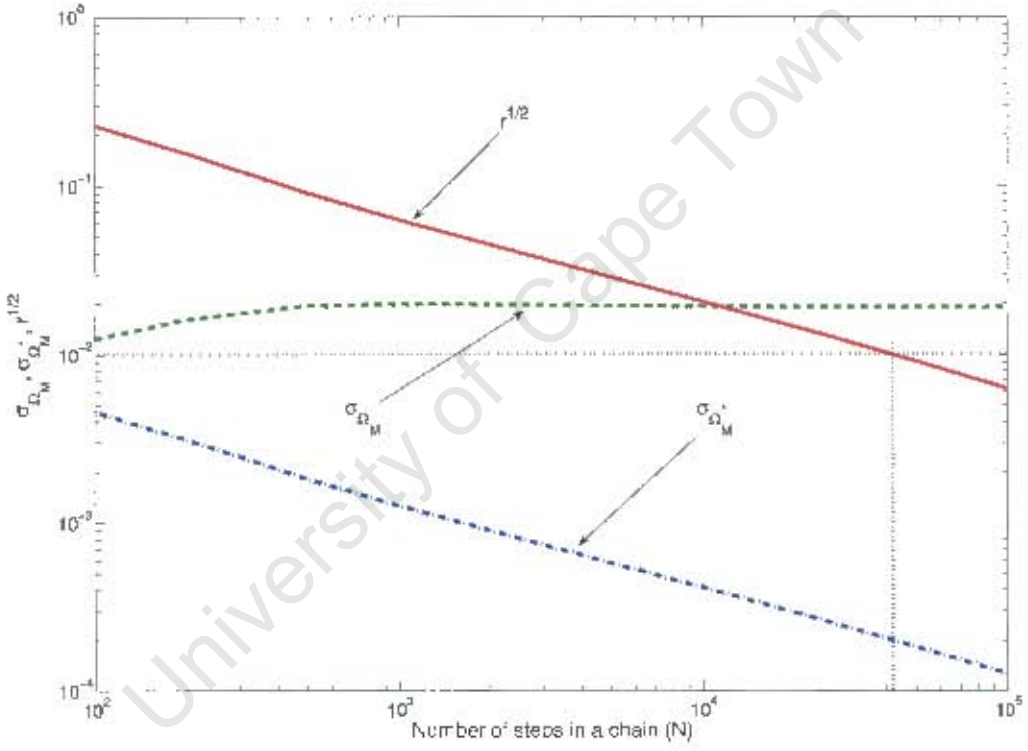

Figure 11.1: Variation of σ_{Ω_M}, $\sigma_{\Omega_M^*}$ and $\sqrt{r}$ with chain length, N. The plot shows that $\sigma_{\Omega_M^*} < \sigma_{\Omega_M}$ for $N > 100$. This may be attributed to the simple likelihood surface of Ω_M (see figure 10.6). For more complicated likelihood surfaces this will not necessarily be the case. For sufficiently complex likelihood surfaces we would not expect the MCMC error to be negligibly small. Notice that $r^{1/2} < 10^{-2}$ for $N > 4 \times 10^4$ approximately.

$1/\sqrt{N}$ as N is increased. By considering the mean and the standard deviation of a sample we would expect the error in the mean to drop off as $1/\sqrt{N}$, as observed. This simple statement will generally fail, however, for more complicated likelihood surfaces. Future work might involve increasing the number of free parameters and/or using highly variable analytic functions (e.g. Griewangk's function) as a likelihood surface.

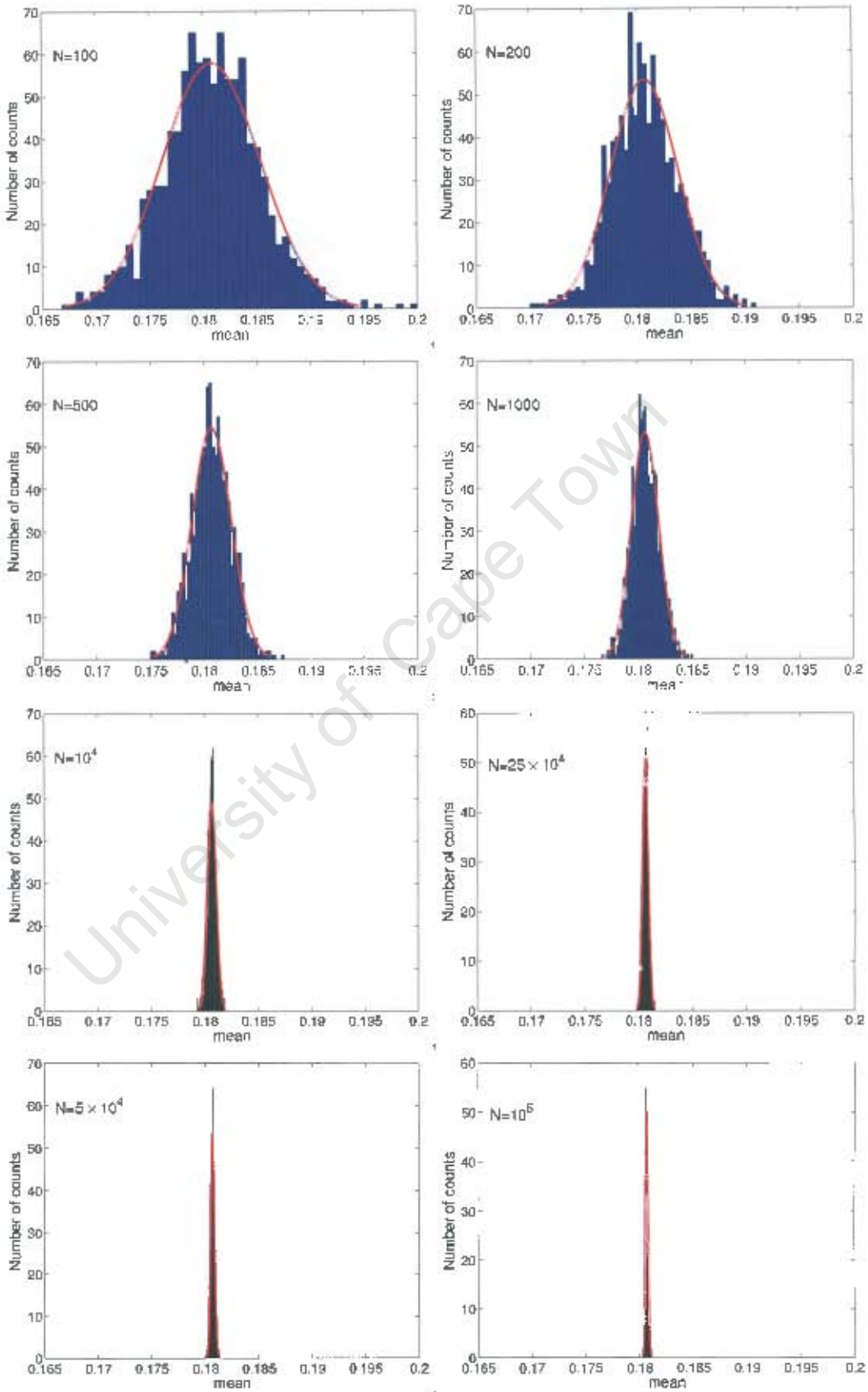

Figure 11.2: Histograms of the 1000 means together with the best fit Gaussians for various values of N. As we would expect, the spread of the means (a measure of the MCMC error, $\sigma_{\Omega^+_{24}}$) decreases with N roughly as $1/\sqrt{N}$.

Chapter 12

Parameter Estimation Conclusions

Finally, we end the thesis with a summary and conclusion of part 2 which deals specifically with methods of parameter estimation that will form part of the analysis pipeline for the forthcoming type Ia supernova data from the SDSS-II supernova survey of which the author is an external participant. This conclusion deals specifically with part 2 of the thesis. Conclusions for part 1 may be found at the end of chapter 6.

One of the statistical techniques that cosmologists use to extract specific information from a data set is the method of Monte Carlo Markov Chain parameter estimation. A summary of the basic procedure is as follows: We select a theoretical model that we assume is correct. This model will typically be controlled by some free parameters which ultimately dictate the predictions of the theory. The aim is to determine, from the data set, the most probable values of these parameters. To do so we allow a walker to execute a random walk through the multi-dimensional parameter space. Throughout the random walk we attempt to maximise the posterior distribution which is the product of the likelihood and the prior distribution. If the prior is uniform, as we assume throughout this thesis, we simply maximise the likelihood, $\mathcal{L}$ since the constant prior does not affect the shape of the posterior distribution. Furthermore, if the likelihood is Gaussian then $\mathcal{L} \propto \exp\left(\frac{-\chi^2}{2}\right)$ where is χ^2 is the statistic that communicates to us the discrepancy between our theoretical model, given a certain set of values for the free parameters, and the data.

Since 1998 it has been known that the Universe has experienced a phase of accelerated expansion. Since then, a huge effort has gone toward attempting to explain this accelerated expansion. One such explanation involves modified versions of general relativity, the claim being that GR fails on on sufficiently large length scales. An alternative, and perhaps more generally accepted explanation, is that of dark energy. Dark energy is believed to be the dominant form of energy in the Universe. It has the strange property that it actually behaves as a type of anti-gravity, fueling the accelerated expansion of the Universe. Supposing that dark energy is the cause of the acceleration, the next step is to attempt to quantify its properties. The simplest model of cosmology at present is one in which the dark energy is a cosmological constant, Λ. This Λ has the unique property that its energy density is not diluted by the expansion of the Universe, it remains constant. Of course, the dark energy density may in fact vary with time. With new data coming in all the time many people are attempting to detect some sort of variation with redshift of the density of dark energy thereby eliminating Λ as a dark energy candidate. Ultimately, it is the equation of state parameter w_0 of the dark energy that separates the different models from one another.

In part 2 of this thesis we apply the MCMC pipeline that we have developed (see appendix C for a copy of the code) to the Supernova Legacy Survey (SNLS) first-year data to determine the best-fit Ω_M, H_0 and w_0 parameters for a cosmology with zero curvature. Ω_M represents the ratio of the

amount of matter (both dark and baryonic) in the Universe to the critical density[1] while H_0 is the Hubble constant. w_0 is the important equation of state parameter of dark energy that will determine the properties of the dark energy[2]. These three parameters all form part of the theoretical expression for the luminosity distance of an object. This luminosity distance may then be converted to a distance modulus.

We considered three separate cases. In case 1, Ω_M and H_0 were allowed to vary while keeping w_0 fixed at -1[3]. The results obtained were $\Omega_M = 0.277^{+0.092}_{-0.077}$ and $H_0 = 69.60^{+2.234}_{-2.343}$, consistent with the WMAP results. Case 2 consisted of fixing H_0 at 70 km/sec/Mpc whilst allowing Ω_M and w_0 to vary. The case 2 results were $\Omega_M = 0.336^{+0.112}_{-0.158}$ and $w_0 = -1.198^{+0.357}_{-0.532}$. Finally, case 3 involved estimating all three parameters, fixing none of them. The results obtained were $\Omega_M = 0.319^{+0.082}_{-0.068}$, $H_0 = 78.17^{+12.16}_{-8.107}$ and $w_0 = -2.551^{+1.338}_{-1.692}$. The reader is reminded that upper and lower bounds for these results represent the parameter values on either side of the best fit parameter value between which 68% of the parameter values lie. In the case that the likelihood function of a parameter is a Gaussian, the upper and lower error bounds correspond to the 1σ error bounds. All the results are consistent with the WMAP [9] and SNLS [4] published data at the 68-95% confidence level. The minor discrepancies with the SNLS data can be ascribed to the fact that they used the full error covariance matrices for their data which have not been publically released as well as a set of low redshift ($z \sim 0.05$) which we did not use.

The final conclusion that may be drawn from the above mentioned results is that the code does seem to do its job and that, as a first version, may be used as part of the SDSS-II pipeline. Further work clearly still remains. Future work will involve upgrading the code so that it may handle cosmologies with non-zero curvature and more dark energy parameters.

The last chapter of this thesis deals with the intrinsic error associated with the MCMC method. Usually, a measurement is accompanied by error estimates. We were interested in determining whether or not there is a significant error associated with the MCMC method, the MCMC error, which should be quoted along with the other error estimates. As a means of measuring the MCMC error we carried out an analysis similar in some ways to that suggested by Dunkley et al. [45]. What we did was to run many chains of a fixed length and then check the variation in the best fit parameter values from all the chains. The MCMC error from 1000 chains of length N was then compared to the statistical error of a single chain of length N. The process was repeated for various values of N ranging from $N = 10^2$ to $N = 10^5$. Clearly, for a meaningful statement of a best fit parameter, the MCMC error should be sufficiently small what compared to the error in a single chain. The idea was to check above which chain lengths (N) this is true, thereby providing a lower-limit to the required length of a chain. In our analysis we estimated the best Ω_M value as a means of checking the MCMC error so as to assure the accuracy of the results produced by the MCMC parameter estimation analysis in this thesis. The results were that the MCMC error was always sufficiently low. In retrospect we would expect this to be the case since Ω_M, as we found, has a simple Gaussian likelihood with a single maximum. This will not always be the case when parameters with more complicated likelihood surfaces are estimated. Computational limitations prevented us from studying such cases, however, and this analysis is left to future work.

[1]The critical density is the total amount of matter the Universe requires in order to be spatially flat.

[2]Dark energy is basically the generic term used for any energy component of the Universe that has w_0 <-1/3, resulting in an accelerated expansion rate.

[3]Case 1 is therefore a flat cosmology with Λ as the dark energy.

Appendix A

HYBRID code

Below appears the 'HYBRID code'. This is the FORTRAN 90 code used to produce the data for figures 4.4, 4.5 and 4.6. The code allows 20 FoV of fixed radius to search a 2-dimensional catalog of objects in which each object has associated with it a figure of merit. For each step of the simulation, the code moves each walker based up on its performance relative to the other $N-1$ walkers. Individual subroutines are dedicated to (1) reading in the catalog data, (2) calculating the FoM of a FoV, (3) determining the size of the step (based on f and g) that a walker should attempt, (4) checking whether the step is favourable (the Metropolis-Hastings condition), (5) checking for overlapping of FoV's and then calculating the sum of the FoM's, (6) writing the results to file. Note that the input parameters that need to be specified by the user are provided in the form of an input file (not shown here).


```
module vec
type twoDvec
  real::x,y
end type

INTERFACE operator(.mag.)
   module procedure magnitude
End INTERFACE

CONTAINS

function Magnitude (s,t)
TYPE(twoDvec), intent (in)::s,t
real:: magnitude

magnitude=Sqrt((s%x-t%x)**2+(s%y-t%y)**2)

End function Magnitude

end module vec
!---------------------------------------
module fow_data
use vec

type fow_dat
  type(twoDvec)::p
```

```
  integer::c
end type

end module fow_data
!----------------------------------------
program singlemarkov
use vec
use fow_data
implicit none

type(twoDvec),dimension(:),allocatable::pts
type(fow_dat),dimension(:),allocatable::fows
type(fow_dat),dimension(:,:),allocatable::paths
real::xmax,ymax,xmin,ymin,R,dx,dy,acc_steps,acc_bs,bs,rn
real::sc_max,sc,sc_start,g,sigma0x,sigma0y,p,m,alpha,c,t
real::t1,t2
integer::num_pts,steps,i,N,a,b,amax,q,runs


xmin=2.e38
ymin=2.e38
xmax=-2.e38
ymax=-2.e38

call INPUT(R,N,steps,runs,p,t,sigma0x,sigma0y,alpha,m,c)

allocate(fows(1:N))
allocate(paths(1:N,1:steps))
open(1,file='/home/elson/SA-uni-2/data.dat')
!open(1,file='data.dat')
read(1,*)num_pts
allocate(pts(1:num_pts))
call TRUE_RANDOM()
call READ_IN_DATA(pts,xmax,ymax,xmin,ymin)

open(4,file='/home/elson/SA-uni-2/survey_count_max.dat')
!open(4,file='survey_count_max.dat')
!open(20,file='/home/elson/SA-grad/results.txt')
open(20,file='results.txt')
write(20,*)'# of pts in data set:',num_pts
write(20,*)'xrange:',xmax-xmin
write(20,*)'yrange:',ymax-ymin

do q=1,runs
acc_steps=0.
acc_bs=0.
bs=0.
sc_max=0.
!write(*,*)'Currently on simulation',q

do i=1,N
   call RANDOM_NUMBER(rn)
   rn=rn*(xmax-xmin)
   fows(i)%p%x=rn+xmin
```

```
   call RANDOM_NUMBER(rn)
   rn=rn*(ymax-ymin)
   fows(i)%p%y=rn+ymin
   call FOM_COUNT(fows(i)%p,fows(i)%c)
enddo!i
sc=SURVEY_COUNT(fows)
sc_start=sc
if (sc_start==0.) sc_start=1.
sc_max=sc
write(4,*)q,1,sc_max

do a=1,steps
   do b=1,N
   call SS(dx,dy,fows(b)%c,fows(b)%p,b,sc,a,g)
   call MOVE(fows,dx,dy,acc_steps,bs,acc_bs,b,sc,a)
   paths(b,a)%p=fows(b)%p
   paths(b,a)%c=fows(b)%c
   enddo!b
   if (sc>sc_max) then
      sc_max=sc
      amax=a
      !write(*,*)'Currently on step',a,'Highest sc so far:',sc_max
   endif
   if (a<1000.and.mod(a,5)==0) write(4,*)q,a,sc_max
   if (a.ge.1000.and.mod(a,100)==0) write(4,*)q,a,sc_max
enddo!a
!call RESULTS(q)
enddo!q

close(4)
close(20)
!call PLOTS(fows,paths)

deallocate(pts)
deallocate(fows)
deallocate(paths)
close(1)

CONTAINS
!-------------------------------------------------------
subroutine INPUT(R,N,steps,runs,p,t,sigma0x,sigma0y,alpha,m,c)
implicit none

real,intent(out)::R,p,t,sigma0x,sigma0y,alpha,m,c
integer,intent(out)::N,steps,runs


open(5,file='/home/elson/SA-uni-2/input.dat')
!open(5,file='input.dat')
read(5,*)R
read(5,*)N
read(5,*)steps
read(5,*)runs
read(5,*)p
```

```
read(5,*)m
read(5,*)c
read(5,*)t
read(5,*)sigma0x
read(5,*)sigma0y
read(5,*)alpha
close(5)
!write(*,*)R,N,steps,runs,p,m,c,t,sigma0x,sigma0y,alpha
end subroutine INPUT
!-----------------------------------------------------
subroutine TRUE_RANDOM()
implicit none

integer,dimension(2)::sval
integer::seed

call System_clock(seed)
sval=seed
call Random_seed(put=sval)

end subroutine TRUE_RANDOM
!-----------------------------------------------
subroutine READ_IN_DATA(pts,xmax,ymax,xmin,ymin)
implicit none

type(twoDvec),dimension(1:num_pts),intent(inout)::pts
real,intent(inout)::xmax,ymax,xmin,ymin
integer::k

open(2,file='/home/elson/SA-uni-2/data1.dat')
!open(2,file='data1.dat')
do k=1,num_pts
   read(1,*)pts(k)%x,pts(k)%y
   write(2,*)pts(k)%x,pts(k)%y
   if (pts(k)%x<xmin) xmin=pts(k)%x
   if (pts(k)%x>xmax) xmax=pts(k)%x
   if (pts(k)%y<ymin) ymin=pts(k)%y
   if (pts(k)%Y>ymax) ymax=pts(k)%y
enddo!k
close(2)
!write(*,*)'xmin,xmax,ymin,ymax:',xmin,xmax,ymin,ymax

end subroutine READ_IN_DATA
!-------------------------------------------
subroutine FOM_COUNT(centre,FOM)
implicit none

type(twoDvec),intent(in)::centre
integer,intent(out)::FOM
integer::i
real::d

FOM=0
do i=1,num_pts
```

```
      d=centre.mag.pts(i)
      if (d<R) FOM=FOM+1
   enddo!i

   end subroutine FOM_COUNT
   !-------------------------------------------------
   subroutine SS(dx,dy,FOM,centre,j,sc,a,g)
   implicit none

   real,intent(out)::dx,dy
   type(twoDvec),intent(in)::centre
   type(twoDvec)::centre_next
   integer,intent(in)::FOM,j,a
   real::rat,f,u,v,pi,avgFOM,epsilon
   integer::i
   real,intent(in)::sc
   real,intent(inout)::g

   pi=2.*asin(1.)
   epsilon=1.e-2

   avgFOM=0.
   do i=1,N
      if (i.ne.j) avgFOM=avgFOM+fows(i)%c
   enddo!i

   avgFOM=avgFOM/(N-1)
   rat=FOM/(avgFOM+epsilon)

   if (rat.ge.1.)f=1./rat**p
   if (rat<1.) f=m*rat+c
   if (sc.ne.0.) g=(sc_start/sc)**t
   if (sc==0.) g=sc_start**t

30 call RANDOM_NUMBER(u)
   if (u==0.) GOTO 30
   call RANDOM_NUMBER(v)
   dx=sqrt(-2.*log(u))*cos(2.*pi*v)
   dx=dx*f*g*sigma0x
   if (centre%x+dx>xmax.or.centre%x+dx<xmin) GOTO 30

40 call RANDOM_NUMBER(u)
   if (u==0.) GOTO 40
   call RANDOM_NUMBER(v)
   dy=sqrt(-2.*log(u))*cos(2.*pi*v)
   dy=dy*f*g*sigma0y
   if (centre%y+dy>ymax.or.centre%y+dy<ymin) GOTO 40

   end subroutine SS
   !-------------------------------------------------
   subroutine MOVE(fows,dx,dy,acc_steps,bs,acc_bs,j,sc,a)
   implicit none

   real,intent(in)::dx,dy
```

```
type(fow_dat),dimension(1:N),intent(inout)::fows
real,intent(inout)::acc_steps,bs,acc_bs,sc
type(twoDvec)::centre_next
real::chance,rn,sc_next
integer,intent(in)::j,a
type(fow_dat),dimension(1:N)::dummy

centre_next%x=fows(j)%p%x+dx
centre_next%y=fows(j)%p%y+dy

dummy=fows
dummy(j)%p=centre_next
call FOM_COUNT(dummy(j)%p,dummy(j)%c)
sc_next=SURVEY_COUNT(dummy)

if (sc_next.ge.sc) then
   fows=dummy
   acc_steps=acc_steps+1.
   sc=sc_next
else
   alpha=0.8*log10(a+1.)
   chance=exp(-alpha*abs(sc_next-sc))
   call RANDOM_NUMBER(rn)
   bs=bs+1.
   if (rn.le.chance) then
      fows=dummy
      bs=bs+1.
      acc_steps=acc_steps+1.
      acc_bs=acc_bs+1.
      sc=sc_next
   endif
endif

end subroutine MOVE
!-----------------------------------------------
real function SURVEY_COUNT(fows)
implicit none

type(fow_dat),dimension(1:N),intent(in)::fows
integer::i,j
real::d,dc

survey_count=0.
do i=1,N
   survey_count=survey_count+fows(i)%c
enddo!i

dc=0.
do i=1,N
do j=1,N
   d=fows(i)%p.mag.fows(j)%p
   if (i.ne.j.and.d<2.*R) call OVERLAP(dc,fows(j)%p,fows(i)%p)
enddo!j
enddo!i
```

```
dc=dc/2.
survey_count=survey_count-dc

end function SURVEY_COUNT
!-------------------------------------------------------------
subroutine OVERLAP(dc,fow1,fow2)
implicit none

real,intent(inout)::dc
type(twoDvec),intent(in)::fow1,fow2
real::e,f
integer::h

do h=1,num_pts
   e=pts(h).mag.fow1
   f=pts(h).mag.fow2
   if (e<R.and.f<R) dc=dc+1.
enddo

end subroutine OVERLAP
!-------------------------------------------------------------
subroutine RESULTS(q)
implicit none

integer,intent(in)::q

write(20,*)'-----------------------------------------------'
write(20,*)'run',q
write(20,*)'The best survey count was',sc_max,'at step',amax
write(20,*)'total # of steps acc:',acc_steps/(steps*N)*100.,'%'
write(20,*)'# of bad steps acc:',acc_bs/bs*100.,'%'
write(20,*)'ratio of acc_bs to acc steps:',acc_bs/acc_steps

end subroutine RESULTS
!-------------------------------------------------------------
subroutine PLOTS(fows,paths)
implicit none

type(fow_dat),dimension(1:N),intent(in)::fows
type(fow_dat),dimension(1:N,1:steps),intent(in)::paths
real::t,pi
integer::i

pi=2.*asin(1.)

open(9,file='fows.dat')
do i=1,N
do t=0.,2.*pi,0.01
write(9,*)paths(i,amax)%p%x+R*cos(t),paths(i,amax)%p%y+R*sin(t)
enddo!t
enddo!i
close(9)
```

```
end subroutine PLOTS
!-----------------------------------------------------------------------
end program singlemarkov
```

Appendix B

Derivations of some Cosmological Equations

B.1 $\rho \propto a^{-3(1+w)}$ for a Perfect Fluid.

We may combine the equation

$$\dot{\rho} + 3\frac{\dot{a}}{a}\left(\rho + \frac{P}{c^2}\right) = 0 \tag{B.1}$$

with the equation of state parameter

$$w = \frac{P}{\rho c^2} \tag{B.2}$$

to produce

$$\dot{\rho} + 3\frac{\dot{a}}{a}\rho(1+w) = 0. \tag{B.3}$$

Rearranging the above equation:

$$\frac{\dot{\rho}}{\rho} = -3(1+w)\frac{\dot{a}}{a}. \tag{B.4}$$

Integrating with respect to time:

$$\int \frac{\dot{\rho}}{\rho}dt = -3(1+w)\int \frac{\dot{a}}{a}dt. \tag{B.5}$$

Using the equality

$$\frac{\dot{x}}{x}dt = \frac{\left(\frac{dx}{dt}\right)}{x}dt = \frac{dx}{x}, \tag{B.6}$$

(B.5) becomes

$$\int \frac{d\rho}{\rho} = -3(1+w)\int \frac{da}{a} \tag{B.7}$$

$$\Rightarrow \ln\rho = -3(1+w)\ln a \tag{B.8}$$

$$= \ln a^{-3(1+w)}. \tag{B.9}$$

Exponentiating yields

$$\rho \propto a^{-3(1+w)}. \tag{B.10}$$

B.2 The Acceleration Equation for a Spatially Flat Universe

We are given the Friedmann equation for a flat (k=0) Universe,

$$\left(\frac{\dot{a}}{a}\right)^2 = \frac{8\pi G}{3c^2}\rho, \tag{B.11}$$

and the energy conservation equation,

$$\dot{\rho} + 3\frac{\dot{a}}{a}\left(\rho + \frac{P}{c^2}\right) = 0. \tag{B.12}$$

Taking the time derivative of the Friedmann equation yields

$$2\frac{\dot{a}}{a}\left(\frac{\ddot{a}a - \dot{a}^2}{a^2}\right) = \frac{8\pi G}{3c^2}\dot{\rho}. \tag{B.13}$$

From (B.12) we have

$$\dot{\rho} = -3\frac{\dot{a}}{a}\left(\rho + \frac{P}{c^2}\right). \tag{B.14}$$

Substituting (B.14) into (B.13):

$$2\frac{\dot{a}}{a}\left(\frac{\ddot{a}a - \dot{a}^2}{a^2}\right) = \frac{8\pi G}{3c^2}\left[-3\frac{\dot{a}}{a}\left(\rho + \frac{P}{c^2}\right)\right] \tag{B.15}$$

$$\Rightarrow 2\left[\frac{\ddot{a}}{a} - \left(\frac{\dot{a}}{a}\right)^2\right] = -\frac{8\pi G}{c^2}\left(\rho + \frac{P}{c^2}\right) \tag{B.16}$$

$$\Rightarrow \frac{\ddot{a}}{a} - H^2 = -\frac{4\pi G}{c^2}\left(\rho + \frac{P}{c^2}\right). \tag{B.17}$$

Using

$$H^2 = \frac{8\pi G}{3c^2}\rho \tag{B.18}$$

we have

$$\frac{\ddot{a}}{a} = \frac{8\pi G}{3c^2}\rho - \frac{4\pi G}{c^2}\left(\rho + \frac{P}{c^2}\right) \tag{B.19}$$

$$= \frac{4\pi G}{c^2}\left(-\rho - \frac{P}{c^2} + \frac{2}{3}\rho\right) \tag{B.20}$$

$$= -\frac{4\pi G}{3c^2}\left(\frac{3P}{c^2} + \rho\right) \tag{B.21}$$

$$= -\frac{4\pi G}{3c^2}\rho\left(\frac{3P}{\rho c^2} + 1\right) \tag{B.22}$$

$$= -\frac{4\pi G}{3c^2}\rho\left(1 + 3w\right). \tag{B.23}$$

Appendix C

Astronomical Data and Code Used for the Parameter Estimation

C.1 SNLS first-year data

Listed below is the redshift, distance modulus, and error in the distance modulus (columns 1, 2 and 3 respectively) for 71 confirmed SNIa from the SNLS first-year data. The data is used to estimate Ω_M, H_0 and w_0.

0.2490	40.5710	0.1328
0.2630	40.6820	0.1319
0.2850	40.7310	0.1321
0.2910	40.9990	0.1353
0.3310	40.9460	0.1318
0.3370	41.2590	0.1316
0.3400	41.3230	0.1317
0.3460	41.3670	0.1329
0.3570	41.4410	0.1324
0.3580	41.4740	0.1318
0.3690	41.4850	0.1321
0.3710	41.4880	0.1346
0.4150	41.7720	0.1339
0.4300	41.7760	0.1365
0.4490	41.8660	0.1383
0.4510	42.0380	0.1345
0.4610	42.0310	0.1434
0.4630	42.0890	0.1355
0.4700	42.2680	0.1350
0.4960	42.1800	0.1365
0.5040	42.4290	0.1368
0.5210	42.4390	0.1418
0.5260	42.6440	0.1422
0.5320	42.5920	0.1552
0.5480	42.8250	0.1536
0.5500	42.4870	0.1426
0.5520	42.4610	0.1357

0.5570	42.5110	0.1343
0.5710	42.4650	0.1486
0.5810	42.7610	0.1393
0.5820	42.8810	0.1418
0.5920	42.5620	0.1590
0.6040	42.5150	0.1343
0.6100	42.7960	0.1368
0.6130	42.9610	0.1446
0.6200	43.0300	0.1442
0.6270	42.7460	0.1357
0.6330	43.1330	0.1459
0.6430	43.0230	0.1437
0.6790	43.3980	0.1579
0.6880	43.0460	0.1399
0.6910	43.1440	0.1889
0.6950	43.0230	0.1568
0.7070	43.2370	0.1839
0.7100	43.1760	0.1520
0.7210	43.2090	0.1686
0.7300	43.2870	0.1510
0.7410	43.4270	0.1757
0.7520	43.1700	0.1590
0.7560	43.4530	0.1433
0.7910	43.3530	0.1515
0.8000	43.4900	0.1590
0.8100	43.6910	0.3283
0.8110	43.9500	0.2947
0.8170	43.6520	0.2858
0.8180	43.5320	0.3311
0.8220	43.5440	0.2544
0.8300	44.4140	0.3709
0.8400	43.4750	0.2267
0.8680	43.8710	0.3681
0.8700	44.0950	0.3283
0.9050	43.7080	0.2894
0.9100	44.2590	0.3700
0.9270	44.5530	0.4020
0.9300	44.4300	0.4495
0.9490	43.5070	0.3019
0.9500	43.9540	0.2983
0.9600	43.6220	0.2682
0.9610	44.0000	0.3182
0.9830	43.9410	0.5130
1.0100	44.6730	0.5489

C.2 The Code used to estimate Ω_M, H_0 and w_0.

The last part of this appendix consists of the three programs used to ultimately extract from the SNLS first-year data set (above) the best fit Ω_M, H_0 and w_0 parameters. Sections C.2.1, C.2.2 and C.2.3 contain, respectively, the code used to execute the Markov chains, burn in the Markov chains and then calculate the best fit parameters along with the upper and lower error bounds. See section 8.3 for a discussion of each program.

C.2.1 MCMC code

```
program mcmc_model
implicit none

real::D1,c,chi_sq1,chi_sq2,pi,chance,alpha,chi_min
real,dimension(:),allocatable::z_obs,mu_obs,sig_obs,mu_calc
!z_obs, mu_obs, sig_obs are arrrays containing the SNLS redshift,
!distance modulus and error data respectively
real,dimension(:),allocatable::p,dp,std_devs,best
!p=parameters, dp=change in parameters, std_devs=std dev of parameter distributions
real,dimension(:,:),allocatable::priors
integer::k,steps,num_p,i,chains,s,i_min,s_min
!steps=# of steps in a chain, chains=# of chains
character(len=50)::title

c=3.e5
pi=2.*asin(1.)
dp=0.
chi_min=2.e38
!chi_min is the minimum chi squared value obtained from any step from all that chains
i_min=1
!i_min is the step number of the chain at which chi_min occurs
s_min=1
!s_min is the chain in which chi_min occurs

call TRUE_RANDOM()
call GET_PARMS()
allocate(p(1:num_p))
call SYSTEM('clear')
call SYSTEM('date')
call GET_DATA(k)
allocate(dp(1:num_p))
allocate(mu_calc(1:k))
allocate(best(1:num_p))

do s=1,chains
call INITIALIZE(p)
call CALC_MU(z_obs,mu_calc)
call CHI_SQUARED(chi_sq1,mu_obs,mu_calc,sig_obs)
write(title,1000)s
1000    format('chain',i2.2,'.dat')

open(30,file=title)
!-----------------------LOOP-----------------------------------------------------
do i=1,steps
if (mod(i,1000)==0) write(*,*)'run:',s,'step:',i
call CALC_DP(dp,std_devs,priors)
p=p+dp
call CALC_MU(z_obs,mu_calc)
call CHI_SQUARED(chi_sq2,mu_obs,mu_calc,sig_obs)
call ATTEMPT_STEP(chi_sq1,chi_sq2,p)
```

```
if (chi_sq1<chi_min) then
    chi_min=chi_sq1
    best=p
    i_min=i
endif
write(30,*)i,chi_sq1,p
enddo!i
close(30)
enddo!s

open(40,file='chi_min.dat')
write(*,*)'chi_min:',chi_min
write(*,*)'best parms:',best,'step:',i_min,'run',s_min
write(40,*)chi_min
write(40,*)'best parms:',best,'step:',i_min,'run',s_min
close(40)

deallocate(std_devs)
deallocate(priors)
deallocate(z_obs)
deallocate(mu_obs)
deallocate(sig_obs)
deallocate(p)
deallocate(dp)
deallocate(best)
call system('date')
!-------------------------------------------------------------------------------
CONTAINS

!This subroutine sets the seed of the random number generator
subroutine TRUE_RANDOM()
implicit none

integer,dimension(2)::sval
integer::seed

call System_clock(seed)
sval=seed
call Random_seed(put=sval)

end subroutine TRUE_RANDOM
!-------------------------------------------------------------------------------
!This subroutine gets the the parameter values to be used by the code from an input file
subroutine GET_PARMS()
integer::s

open(50,file='input_parameters-model2.dat')
read(50,*)chains
read(50,*)steps
read(50,*)num_p
allocate(std_devs(1:num_p))
do s=1,num_p
   read(50,*)std_devs(s)
enddo!s
```

```
allocate(priors(1:2,1:num_p))
do s=1,num_p
   read(50,*)priors(1,s), priors(2,s)
enddo!s
read(50,*)alpha
close(50)
end subroutine GET_PARMS
!-------------------------------------------------------------------------------
!This subroutine reads in the SNLS z, mu and sigma data
!into 3 separate arrays called z_obs and mu_obs and sig_obs

subroutine GET_DATA(k)
implicit none

real::aa,ab,ac
integer::j
integer,intent(inout)::k

!find out how many lines of data are in the data file:
k=0
open(10,file='SNLS_data.dat')
100     read(10,*,end=200)aa,ab,ac
k=k+1
GOTO 100
200     close(10)

write(*,*)'# of observations:',k !# of observations
allocate(z_obs(1:k))
allocate(mu_obs(1:k))
allocate(sig_obs(1:k))
open(10,file='SNLS_data.dat')
do j=1,k
   read(10,*)aa,ab,ac
   z_obs(j)=aa !the redshifts
   mu_obs(j)=ab !the distance moduli
   sig_obs(j)=ac !errors of the dist moduli
enddo!j
close(10)

end subroutine GET_DATA
!-------------------------------------------------------------------------------
!This subroutine initializes the parameter values for the first step of the chain
subroutine INITIALIZE(p)
real,dimension(1:num_p),intent(inout)::p
call TRUE_RANDOM()
call RANDOM_NUMBER(p(1)) ! 0<O_m<1
call RANDOM_NUMBER(p(2)) !0<H0<1
p(2)=p(2)*80. !0<H0<80
p(2)=p(2)+20. !20<H0<100
call RANDOM_NUMBER(p(3)) !0<w0<1
p(3)=p(3)*(-4.66666666) !-4.6666<w0<0
p(3)=p(3)-0.33333333 ! -5<w0<-0.333333

end subroutine INITIALIZE
```

```
!-------------------------------------------------------------------------------
!This subroutine calculates mu's from Dl's corresponding to different redshifts
subroutine CALC_MU(z_obs,mu_calc)
implicit none

real,dimension(1:k),intent(in)::z_obs
real,dimension(1:k),intent(out)::mu_calc
integer::j
real::res

do j=1,k
   call COMP_TRAP_RULE(0.,z_obs(j),1000,res)
   Dl=(1.+z_obs(j))*(c/p(2))*res !in units of Mpc
   mu_calc(j)=5.*log10(Dl*1.e6/10.) !convert Dl to pc to make log10 argument dimensionless
enddo!j
end subroutine CALC_MU
!-------------------------------------------------------------------------------
!This subroutine use the trap rule for numerical integration
!of a function between the limits x0 and xn
subroutine COMP_TRAP_RULE(x0,xn,nsteps,res)
implicit none

real,intent(in)::x0,xn
integer,intent(in)::nsteps
real,intent(out)::res
real::h
integer::i

h=(xn-x0)/nsteps
res=(h/2.)*(func(x0)+func(xn))
do i=1,nsteps-1
   res=res+func(i*h)
enddo!i
res=res*h
end subroutine COMP_TRAP_RULE
!-------------------------------------------------------------------------------
!The function to be integrated above:
real function func(x)
implicit none

real,intent(in)::x
real::DE

DE=(1+x)**(3*(1+p(3)))
func=1./sqrt(p(1)*(1.+x)**3+(1.-p(1))*DE)

end function func
!-------------------------------------------------------------------------------
!A subroutine that produces a random step in each parameter dimension:
subroutine CALC_DP(dp,std_devs,priors)
implicit none

real,dimension(1:num_p),intent(out)::dp
real,dimension(1:num_p),intent(in)::std_devs
```

```
real,dimension(1:2,1:num_p),intent(in)::priors
real::u,v,random
integer::s

do s=1,num_p
1000        call RANDOM_NUMBER(u)
            call RANDOM_NUMBER(v)
!generate a gaussian number with mean 0 and variance 1
random=sqrt(-2.*log(u))*cos(2.*pi*v)
!change the variance:
dp(s)=random*std_devs(s)
!The parameter prior comes into play:
if(p(s)+dp(s)<priors(1,s).or.p(s)+dp(s)>priors(2,s)) GOTO 1000
enddo!s
end subroutine CALC_DP
!-------------------------------------------------------------------------------
!A subroutine that calculates the chi squared statistic:
subroutine CHI_SQUARED(chi_sq,mu_obs,mu_calc,sig_obs)
implicit none

real,intent(inout)::chi_sq
real,dimension(1:k),intent(in)::mu_obs,mu_calc,sig_obs
integer::j
real::denom

chi_sq=0.
do j=1,k
   denom=(sig_obs(j)**2)
   chi_sq=chi_sq+(mu_calc(j)-mu_obs(j))**2/denom
enddo!j
end subroutine CHI_SQUARED
!-------------------------------------------------------------------------------
!A subroutine that uses the Metropolis-Hastings algorithm to decide
!whether or not to take a step in the parameter space
subroutine ATTEMPT_STEP(chi_sq1,chi_sq2,p)
implicit none

real,intent(inout)::chi_sq1,chi_sq2
real,dimension(1:num_p),intent(inout)::p
real::expo,expo1

expo1=exp(alpha*(chi_sq1-chi_sq2))
expo=expo1

call RANDOM_NUMBER(chance)!0<chance<1
if (chance<min(1.0,expo)) then
    chi_sq1=chi_sq2
else
    p=p-dp
endif

end subroutine ATTEMPT_STEP
       !-------------------------------------------------------------------------------
end program mcmc_model
```

C.2.2 Burn-in Code

```
!Given a parameter chain this code should produce a burnt in version of that chain.
!If multiple runs are performed then, at the end, all of the burnt in chains of that
!parameter are combined to form one master burnt in chain from which the best parameter
!value together with error bars can be calculated.

program burn_in
implicit none
integer::steps,i,i_min,i_max,L_half,num_chains,s
real,dimension(:),allocatable::chi_sq,L,O_m,H0,w0
real::aa,ab,ac,ad,ae,chi_min,chi_max,L_best
character(len=50)::title_in,title_out

!get the minimum chi squared value from the MCMC chains:
open(40,file='chi_min.dat')
read(40,*)chi_min
close(40)
!Get the details of the chains:
open(10,file='input_parameters-model2.dat')
read(10,*)num_chains
read(10,*)steps
close(10)

allocate(chi_sq(1:steps))
allocate(L(1:steps))
allocate(O_m(1:steps))
allocate(H0(1:steps))
allocate(w0(1:steps))
open(50,file='master_chain.dat')

do s=1,num_chains
write(title_in,1000)s
1000    format('chain',i2.2,'.dat')
write(*,*)'chain:',s,title_in
open(20,file=title_in)
!read in chi_sq, Om, H0 and w0 values:
do i=1,steps
   read(20,*)aa,ab,ac,ad,ae
   chi_sq(i)=ab
   O_m(i)=ac
   H0(i)=ad
   w0(i)=ae
enddo!i
close(20)

!create likelihood array, normalised wrt L_max
do i=1,steps
   L(i)=exp(-chi_sq(i)/2.+chi_min/2.)
enddo!i

L_half=1
!Find the step in the chain at which the Likelihood rises above half
```

```
!the maximum likelihood:
do i=1,steps
   if (L(i)>0.5) then
   L_half=i
   GOTO 300
endif
enddo!i

write(title_out,2000)s
2000    format('burnt_chain',i2.2,'.dat')
open(30,file=title_out)
!write the burnt-in chain to file and add it to the master chain:
do i=L_half,steps
   write(30,*)i,O_m(i),H0(i),w0(i)
   write(50,*)O_m(i),H0(i),w0(i)
enddo!i
close(30)
enddo!s
close(50)
deallocate(chi_sq)
deallocate(L)
deallocate(O_m)
deallocate(H0)
deallocate(w0)
end program burn_in
```

C.2.3 Best-Fit Parameter Code

```
 !Given a master parameter chain, this code should find the mean, p_1 and p_2 values of the

program best_value_and_error
implicit none

character(Len=50)::title
integer::k,i,num_p,j
real::median,upper_bound,lower_bound,aa,ab,ac,ad,ae
real, dimension(:),allocatable::O_m,H0,w0

title='master_chain.dat'
!# of free parameters:
num_p=3

open(70,file='final_results-model2.dat')

!Find out how many elements are in the chain:
k=0
open(10,file=title)
100     read(10,*,end=200)aa,ab,ac
k=k+1
GOTO 100
200     close(10)
```

```
allocate(O_m(1:k))
allocate(H0(1:k))
allocate(w0(1:k))

open(10,file=title)
!read each of the parameter chains into an array:
do j=1,k
   read(10,*)O_m(j),H0(j),w0(j)
enddo!j
close(10)

call SORT_ARRAY(O_m,k)
!calculate the median:
if (mod(real(k),2.).ne.0.) then
    median=O_m(int(ceiling(k/2.)))
else
    median=(O_m(floor(real(k)/2.))+O_m(ceiling(real(k)/2.)))/2.
endif
!calculate the lower bound:
lower_bound=O_m(int(floor(0.1587*k)))
!calculate the upper bound:
upper_bound=O_m(int(ceiling(.8413*k)))
write(*,*)'-----------------------------------------------'
write(*,*)'results for O_m:'
write(*,*)median,'+',upper_bound-median,'-',median-lower_bound
write(70,*)'-----------------------------------------------'
write(70,*)'results for O_m:'
write(70,*)median,'+',upper_bound-median,'-',median-lower_bound
!---------------------------------------------------------------

call SORT_ARRAY(H0,k)
!calculate the median:
if (mod(real(k),2.).ne.0.) then
   median=H0(int(ceiling(k/2.)))
else
   median=(H0(floor(real(k)/2.))+H0(ceiling(real(k)/2.)))/2.
endif
!calculate the lower bound:
lower_bound=H0(int(floor(0.1587*k)))
!calculate the upper bound:
upper_bound=H0(int(ceiling(.8413*k)))
write(*,*)'-----------------------------------------------'
write(*,*)'results for H0:'
write(*,*)median,'+',upper_bound-median,'-',median-lower_bound
write(70,*)'-----------------------------------------------'
write(70,*)'results for H0:'
write(70,*)median,'+',upper_bound-median,'-',median-lower_bound
!---------------------------------------------------------------

call SORT_ARRAY(w0,k)
!calcualte the median:
if (mod(real(k),2.).ne.0.) then
   median=w0(int(ceiling(k/2.)))
else
```

```
   median=(w0(floor(real(k)/2.))+w0(ceiling(real(k)/2.)))/2.
endif
!calculate the lower bound:
lower_bound=w0(int(floor(0.1587*k)))
!calculate the upper bound:
upper_bound=w0(int(ceiling(.8413*k)))
write(*,*)'-----------------------------------------------'
write(*,*)'results for w0:'
write(*,*)median,'+',upper_bound-median,'-',median-lower_bound
write(70,*)'-----------------------------------------------'
write(70,*)'results for w0:'
write(70,*)median,'+',upper_bound-median,'-',median-lower_bound
!---------------------------------------------------------------------
CONTAINS
!This subroutine will sort an a array of real numbers into ascending order
subroutine SORT_ARRAY(u,size)
implicit none

integer,intent(in)::size
real,dimension(1:size)::u
real::temp,big
integer::m,s,i

m=size
do while (m>1)
!search for the smallest element:
   big=u(1)
   do i=1,m
   if (u(i).ge.big) then
      big=u(i)
      s=i
   endif
enddo

!make the swap:
temp=u(m)
u(m)=u(s)
u(s)=temp
m=m-1
enddo !while

m=0
end subroutine SORT_ARRAY
!-------------------------------------------------------------------------
end program best_value_and_error
```

Bibliography

[1] D. W. Hogg. Distance measures in cosmology. *astro-ph/9905116*, 2000.

[2] A. G. Riess et al. Bvri light curves for 22 type ia supernovae. *Astronomical Journal*, page 41, 1998.

[3] N. Palanque-Delabrouille for the SNLS collaboration. Supernova legacy survey (snls) : real time operations and photometric analysis. *To be published by EDP-Sciences*, page 4, 2005.

[4] P. Astier et al. The supernova legacy surevey: Measurement of ω_m, ω_Λ and w from the first year data set. *Astron.Astrophys.*, 447:31–48, 2006.

[5] D. G. York et al. (The SDSS Collaboration). The sloan digital sky survey: Technical summary. *AJ*, 120:1579–1587, 2000.

[6] M. Colless et al. The 2df galaxy redshift survey: Spectra and redshifts. *MNRAS*, 328:1039, 2001.

[7] C. L. Bennett et al. First-year wilkinson microwave anisotropy probe (wmap) observations: Preliminary maps and basic results. *Astrophys.J.Suppl.*, 148:1, 2003.

[8] D. N. Spergel et al. First year wilkinson microwave anisotropy probe (wmap) observations: Determination of cosmological parameters. *Astrophys.J.Suppl.*, 148:175, 2003.

[9] D. N. Spergel et al. Wilkinson microwave anisotropy probe (wmap) three year results: Implications for cosmology. *astro-ph/0603449 (submitted to ApJ)*, 2006.

[10] A. Kogut et al. Wilkinson microwave anisotropy probe (wmap) first year observations: Te polarization. *Astrophys.J.Suppl.*, 148:135, 2003.

[11] G. Hinshaw et al. First year wilkinson microwave anisotropy probe (wmap) observations: Angular power spectrum. *Astrophys.J.Suppl.*, 148:161, 2003.

[12] L. Page et al. Three year wilkinson microwave anisotropy probe (wmap) observations: Polarization analysis. *astro-ph/0603450 (submitted to ApJ)*, 2006.

[13] G. D'Agostini. Bayesian inference in processing experimental data: Principles and basic applications. *Reports on Progress in Physics*, 66:1383–1420, 2003.

[14] L. Verde et al. First year wilkinson microwave anisotropy probe (wmap) observations: Parameter estimation methodology. *Astrophys.J.*, 148:195, 2003.

[15] B. A. Bassett; R. C. Nichol; D. J. Eisenstein; the WFMOS Feasibility Study Dark Energy Team. Wfmos - sounding the dark cosmos. *October 2005 issue of the Royal Astronomical Society magazine A&G*, 2005.

[16] P. S. Corasaniti; M. LoVerde; A. Crotts; C. Blake. Testing dark energy with the advanced liquid-mirror probe of asteroids, cosmology and astrophysics. *astro-ph/0511632*, 2006.

[17] J. Simon; L. Verde; R. Jimenez. Constraints on the redshift dependence of the dark energy potential. *Phys.Rev. D*, 71:123001, 2005.

[18] R. Jimenez; A. Loeb. Constraining cosmological parameters based on relative galaxy ages. *Astrophys.J.*, 573:37–42, 2002.

[19] R. Jimenez; L. Verde; T. Treu; D. Stern. Constraints on the equation of state of dark energy and the hubble constant from stellar ages and the cmb. *Astrophys.J.*, 593:622–629, 2003.

[20] W. H. Press; W. T. Vetterling; S. A. Teukolsky; B. P. Flannery. *Numerical Recipes in C++–The Art of Scientific Computing*, volume 2. Cambridge University Press, 2002.

[21] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[22] G. S. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications.* Springer-Verlag, New York., 1996.

[23] N. Metropolis; A. W. Rosenbluth; M. N. Rosenbluth; A. H. Teller; E. Teller. Equation of state calculations by fast computingmachines,. *J. Chem. Phys.*, 21:1087–1092, 1953.

[24] S. Kirkpatrick; C. D. Gelatt; M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[25] P. J. M. van Laarhoven; E. H. L. Aarts. *Simulated annealing: Theory and applications, Mathematics and Its Applications.* Kluwer, Dordrecht, 1987.

[26] S. H. Hansen. Extraction of cluster parameters from sunyaev-zeldovich effect observations with simulated annealing optimization. *New Astron.*, 9:279–283, 2004.

[27] R. Eberhart; J. Kennedy. Particle swarm optimization. *Proc. IEEE Int. Conf. Neural Networks.*, 4:1942–1948, 1995.

[28] A. P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence.* Wiley and Sons, 2005.

[29] J. Kennedy; R. C. Eberhart. *Swarm intelligence.* Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2001.

[30] C. Skokos; K.E. Parsopoulos; P.A. Patsis; M.N. Vrahatis. Particle swarm optimization: An efficient method for tracing periodic orbits in 3d galactic potentials. *Mon.Not.Roy.Astron.Soc.*, 359:251–260, 2005.

[31] R. Mendes; J. Kennedy; J. Neves. The fully informed particle swarm: simpler, maybe better. *Evolutionary Computation, IEEE Transactions on*, 8:204– 210, 2004.

[32] G. T. Richards et al. The 2df-sdss lrg and qso survey: The z<2.1 quasar luminosity function from 5645 quasars to g=21.85. *Mon.Not.Roy.Astron.Soc.*, 360:825–838, 2005.

[33] M. Tegmark et al. Cosmological parameters from sdss and wmap. *Phys.Rev. D*, 69, 2004.

[34] E. J. Copeland; M. Sami; S. Tsujikawa. Dynamics of dark energy (review). *International Journal of Modern Physics D*, 2006.

[35] A. Riess et al. Observational evidence from supernovae for an accelerating universe and a cosmological constant. *Astron.J.*, 116:1009–1038, 1998.

[36] S. Perlmutter et al. Measurements of omega and lambda from 42 high-redshift supernovae. *Astrophys.J.*, 517:565–586, 1999.

[37] A. Riess et al. Type ia supernova discoveries at z>1 from the hubble space telescope: Evidence for past deceleration and constraints on dark energy evolution. *Astron.J.*, 607:665–687, 2004.

[38] R. R. Caldwell; E. V. Linder. The limits of quintessence. *Phys.Rev.Lett.*, 95:141301, 2005.

[39] P. H. Frampton. Dark energy - a pedagogic review. *Keynote talk to joint session of 5th Rencontres du Vietnam, Hanoi.*, 2004.

[40] *Supernova / Acceleration Probe, Studying the Dark Energy of the Universe.* http://snap.lbl.gov/.

[41] *The Atacama Cosmology Telescope.* http://www.hep.upenn.edu/act/.

[42] A. Kosowsky. The atacama cosmology telescope. *New Astron.Rev.*, 47:939–943, 2003.

[43] *South Pole Telescope.* http://spt.uchicago.edu/.

[44] S. Dodelson. *Modern Cosmology.* Elsevier, 2003.

[45] J. Dunkley et al. Fast and reliable mcmc for cosmological parameter estimation. *Mon.Not.Roy.Astron.Soc.*, Mon.Not.Roy.Astron.Soc.:925–936, 2005.

[46] S. Perlmutter et al. Cosmology from type ia supernovae. *BAAS*, 29:1351, 1997.

[47] D. W. Hogg; I. K. Baldry; M. R. Blanton; D. J. Eisenstein. The k correction. *astro-ph/0210394*, 2002.

[48] S. Perlmutter; B. P. Schmidt. Measuring cosmology with supernovae. *astro-ph/0303428*, 2003.

[49] M. M. Phillips. The absolute magnitudes of type ia supernovae. *Astrophys. J.*, 413,:L105–L108, 1993.

[50] B. P. Schmidt et al. The high-z supernova search: Measuring cosmic deceleration and global cur vature of the universe using type ia supernovae. *Astrophys.J.*, 507:46–63, 1998.

[51] A. Clocchiatti et al. Hubble space telescope and ground-based observations of type ia supernovae at redshift 0.5: Cosmological implications. *Astron.J.*, 642:1–21, 2006.

[52] J. L. Tonry et al. Cosmological results from high-z supernovae. *Astron.J.*, 594:1–24, 2003.

[53] A. G. Riess et al. The farthest known supernova: Support for an accelerating universe and a glimpse of the epoch of deceleration. *Astron.J.*, 560:49–71, 2001.

[54] O. Boulade; X. Charlot; P. Abbon et al. Instrument design and performance for optical/infrared ground-based telescopes. *Proceedings of the SPIE*, 4841:72–81, 2003.

[55] D. J. Eisenstein et al. Detection of the baryon acoustic peak in the large-scale correlation function of sdss luminous red galaxies. *Astrophys.J.*, 633:560–574, 2005.