Parameter Estimation for ODEs using a Cross-Entropy Approach

by

Bo Wang

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Abstract

Parameter Estimation for ODEs using a Cross-Entropy Approach

Bo Wang
Master of Science
Graduate Department of Computer Science
University of Toronto
2012

Parameter Estimation for ODEs and DDEs is an important topic in numerical analysis. In this paper, we present a novel approach to address this inverse problem. Cross-entropy algorithms are general algorithm which can be applied to solve global optimization problems. The main steps of cross-entropy methods are first to generate a set of trial samples from a certain distribution, then to update the distribution based on these generated sample trials. To overcome the prohibitive computation of standard cross-entropy algorithms, we develop a modification combining local search techniques. The modified cross-entropy algorithm can speed the convergence rate and improve the accuracy simultaneously.

Two different coding schemes (continuous coding and discrete coding) are also introduced. Continuous coding uses a truncated multivariate Gaussian to generate trial samples, while discrete coding reduces the search space to a finite (but dense) subset of the feasible parameter values and uses a Bernoulli distribution to generate the trial samples (which are fixed point approximation of the actual parameters) . Extensive numerical and real experiments are conducted to illustrate the power and advantages of the proposed methods. Compared to other existing state-of-the-art approaches on some benchmark problems for parameter estimation, our methods have three main advantages: 1) They are robust to noise in the data to be fitted; 2) They are not sensitive to the number of observation points (in contrast to most existing approaches) ; 3) The modified versions exhibit faster convergence than the original versions, thus they are more efficient, without sacrificing accuracy.

Acknowledgements

I am deeply thankful to my supervisor, Wayne Enright, whose encouragement, guidance and support from the initial to the final level enabled me to develop a through understanding of this project. I also thank Christina Christara for reviewing my work throughout the process of conducting this experiment and for her suggestions given.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the thesis.

Contents

Chapter 1

Introduction

1.1 Motivation

Ordinary differential equations (ODEs) are widely used in modeling dynamic processes in many areas including physics, chemistry and biology. Numerical investigations often involve simulating with various parameter values and estimating the sensitivity of solutions to small changes in parameters. Scientists are now able to use more complex ODE models to study real life problems due to the increasing development and availability of computers with significant processing power.

A key task in systems biology is to learn about biological systems using mathematical models and this task is hampered by the fact that parameters of the models, such as rate constants, are not known or easily available. The same situation arises in chemical engineering, which is also in need of efficient methods for parameter estimation when trying to fit parameter-dependent models to noisy industry measurements.

The challenging problems related to parameter estimation in ODEs are numerous. One of the most important issues is the unpredictable and inevitable existence of noise in measurements. Some parameters are very sensitive to noise which can make their estimation difficult and sometimes even impossible. Another difficulty results from the nonlinearity of the most relevant ODE models, which complicates the adoption of most optimization techniques. All methods for parameter estimation involve an interplay between numerical trajectory simulation of ODEs and a global optimization technique. The simulation of an ODE for a particular choice of the parameters is usually done by a reliable ODE solver (whose accuracy is assumed) and the optimization technique employed assumes that the error associated with a particular simulation is small compared to the other sources of error that arise during the optimization.

The optimization techniques used can be classified as a mainly global or a primarily local procedure. Typical examples of methods based on global optimization are adaptive stochastic methods [1, 35], genetic programming methods [31], and evolutionary methods [30]. A detailed comparison of these global optimization methods is given in [23]. The main disadvantage of a stochastic optimization method is its prohibitive computational cost. On the other hand, methods based on local optimization such as Newton methods [8], or sequential quadratic programming [37] can be more efficient than global

optimization methods. At the same time, however, they suffer from two main obstacles: one is a possible discontinuity of derivatives of the associated functions that arise and another is that they converge to local optima.

There is a trade-off between computational efficiency and robustness when estimating parameters in ODEs. Good methods for parameter estimation should exhibit the following properties: 1) The computational cost needed for reliably estimating parameters should not be too large. This property makes the methods practical and easily adopted in real applications; 2) The methods must be stable and robust to noisy measurements. This is needed in real applications due to the fact that most measurements are inevitably susceptible to unpredictable noise.

Another class of differential equations that arise when modeling real systems are, delay differential equations (DDEs) . Parameter estimation for systems of DDEs is more challenging due to frequent discontinuities in the solution caused by the delay terms. Investigations related to parameter estimation for DDEs are rare. Baker and Paul [2] have argued that there is a need for a systematic method for these problems. Zivaripiran [38] develops a systematic collection of tools related to DDE simulations, sensitivity analysis and parameter estimation. However, the tools for parameter estimation that he developed cannot deal with a large number of parameters.

The goal of this paper is to develop a unified framework for parameter estimation for both ODEs and DDEs. We pursue both effectiveness and efficiency. Robustness with respect to noise and the ability to cope with large number of parameters are also objectives of the approaches we investigate.

1.2 Mathematical Setting for Parameter Estimation of ODEs

Suppose that a dynamical system is represented by a state variable $\mathbf{y}(t) \in \mathbb{R}^d$ for $t \in [t_0, t_f]$, which is the unique and differentiable solution of an initial value problem,

$$\mathbf{y}' = \mathbf{g}(t, \mathbf{y}(t), \mathbf{p}), \tag{1.1}$$

$$\mathbf{y}(t_0) = \mathbf{y}_0. \tag{1.2}$$

The right-hand side of (1.1) depends on a constant vector of parameters $\mathbf{p} \in R^{n_p}$. Note that here we do not assume that $\mathbf{g}$ is continuously differentiable with respect to $\mathbf{p}$. Then the complete set of parameters $\mathbf{\Theta}$ of the model is defined by the vector $\mathbf{\Theta}^T = (\mathbf{p}^T, \mathbf{y}_0^T)$ having dimension $s = n_p + d$.

Let $\bar{Y}_i$, $i = 1, \ldots, n$, denote a set of observed data vectors, with components $\bar{Y}_{ij}$, $j = 1, \ldots, d$, i.e., $\bar{Y}$ is a $n \times d$ matrix, where n represents the total amount of data. Let Y_{ij} denotes the "true" solution of component j of the state at time t_i of the solution of (1.1) with exact parameters $\tilde{\mathbf{\Theta}}$. We assume the data $\bar{Y}_{ij}$ satisfy the following observation equation

$$\bar{Y}_{ij} = Y_{ij} + \sigma_j \epsilon_{ij} \tag{1.3}$$

where $\sigma_j > 0$ measures the variance of the noise associated with the j-the component and is related to the scale of the expected magnitude of the jth component, $|\mathbf{y}_j(t)|$. The

ϵ_{ij}s are independent and standard Gaussian distributed random variables. Observation points t_i are ordered such that $t_0 \leq t_1 < \ldots < t_{n-1} < t_n$. On the basis of knowing only the measurements $\bar{Y}_{ij}$, the task is to estimate the initial state y_0 and parameters $\mathbf{p}$. The principle of maximum-likelihood yields an appropriate cost function which should be minimized with respect to the parameters $\mathbf{\Theta}$ to yield an approximation, $\bar{\Theta}$ to the true value $\tilde{\mathbf{\Theta}}$. We define the cost function or objective function by

$$\mathcal{L}(\mathbf{\Theta}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d}\frac{(Y_{ij}(\mathbf{\Theta}) - \bar{Y}_{ij})^2}{2\sigma_j^2}, \tag{1.4}$$

and we seek a $\bar{\Theta}$ that satisfies:

$$\bar{\Theta} = \arg\min_{\mathbf{\Theta}} \mathcal{L}(\mathbf{\Theta}) \tag{1.5}$$

Note that $\bar{\Theta}$ need not be unique. A direct minimization of $\mathcal{L}$ with respect to $\mathbf{\Theta}$ leads to the so-called initial value approach (see, for example, [25]) . Instead of direct minimization, we will adopt an equivalent optimization formulation to (1.5) as follows: Let $\phi : R^+ \rightarrow R$ be a strictly decreasing function and $S(\Theta) = \phi(\mathcal{L}(\Theta))$. Then an equivalent optimization problem (1.5) is,

$$\max_{\mathbf{\Theta}} S(\mathbf{\Theta}) = \phi(\mathcal{L}(\mathbf{\Theta})) = \phi(\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d}\frac{(Y_{ij}(\Theta) - \bar{Y}_{ij})^2}{2\sigma_j^2}) \tag{1.6}$$

Clearly from (1.4) , $\mathcal{L}(\mathbf{\Theta}) \geq 0$ and since $\phi(x)$ is strictly decreasing, $\mathcal{L}(\bar{\Theta})$ is a minimum if and only if $S(\bar{\Theta})$ is a maximum. In this paper, we will focus on formulation (1.6) with $\phi(x) = \exp(-x)$ and we will call the function $S(\Theta)$ the *performance function*.

1.3 A Review of Related Work

As we have shown, the parameter estimation problem can be interpreted as an optimization problem. Most methods for these optimization problems use a initial value approach which is summarized in Fig. (1.1) .

1. Choose an initial guess for the parameters, $\hat{\Theta}$.
2. Solve model (1.1) using an effective IVP solver with $\Theta = \hat{\Theta}$ and evaluate the objective function $\mathcal{L}(\hat{\Theta})$ or the performance function $S(\hat{\Theta})$.
3. Check stopping conditions, if satisfied, stop
4. Otherwise choose a better value for $\hat{\Theta}$ and return to step (2) .

Figure 1.1: Summary of the initial value approach

The key steps in the initial value approach is how to solve the model equations (1.1) in step 2 and how to improve the approximation $\hat{\Theta}$ in step 4. Simulation of (1.1) is done by a reliable numerical method, such as a Runge-Kutta method, to approximate the solution given a trial set of parameter values.

There is an extensive literature on numerical methods for solving constrained optimization problems such as (1.6) (see Biegler and Grossman [3] for an excellent review) . The particular optimization methods used will determine the stopping criteria (step 3) and the updating rules for step 4. The optimization methods we consider can be categorized into two classes. The first class is composed of local optimization (LO) methods. This class of optimization methods usually require the evaluation of derivatives of the objective function. The assumption that the function g in (1.1) is continuously differentiable almost everywhere with respect to $\mathbf{\Theta}$ is required. The gradient of the objective function (1.4) is computed using

$$\frac{\partial \mathcal{L}(\Theta)}{\partial \Theta} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d}\frac{(Y_{ij}-\bar{Y}_{ij})}{\sigma_j^2}(\frac{\partial Y_{ij}(\Theta)}{\partial \Theta}) \tag{1.7}$$

The partial derivatives, appearing in the right side of (1.7) , $\frac{\partial Y_{ij}(\Theta)}{\partial \Theta}$ can be approximated by solving the sensitivity IVPs: $(\frac{\partial \mathbf{y}}{\partial \mathbf{p}})$ is an $d \times n_p$ matrix which is the solution of the IVP,

$$\frac{d}{dt}(\frac{\partial \mathbf{y}}{\partial \mathbf{p}}) = \frac{\partial \mathbf{g}}{\partial \mathbf{p}} + \frac{\partial \mathbf{g}}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{p}}, \tag{1.8}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{p}}|_{t=0} = 0; \tag{1.9}$$

and $\frac{\partial \mathbf{y}}{\partial \mathbf{y_0}}$ is a $d \times d$ matrix which is the solution to the IVP,

$$\frac{d}{dt}(\frac{\partial \mathbf{y}}{\partial \mathbf{y_0}}) = \frac{\partial \mathbf{g}}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{y_0}}, \tag{1.10}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{y_0}}|_{t=0} = \mathbf{I}. \tag{1.11}$$

Note that the gradient $\frac{\partial S(\Theta)}{\partial \Theta}$, required for the optimization formulation (1.6) , can easily be computed (when $\phi(x) = \exp(-x)$) by:

$$\frac{\partial S(\Theta)}{\partial \Theta} = -\phi(\mathcal{L})\frac{\partial \mathcal{L}(\Theta)}{\partial \Theta}. \tag{1.12}$$

Edsberg et. al [8] developed a toolbox, called *diffpar*, which uses a Gauss-Newton type algorithm for directly minimizing the objective function (1.4) . Despite its simplicity, the Gauss-Newton method has slow convergence for a large residual initial guess, and, to obtain convergence it is essential to have a good initial guess. A trust region Levenberg-Marquardt method to minimize the objective function (1.5) is used in [39, 6] where finite differences are used to approximate the required derivatives. This method suffers from the high cost of calculation of the derivatives and the fact that the method must be started from many initial guesses in order to achieve convergence and avoid local minima. The sequential quadratic programming (SQP) method is a modification of the Gauss-Newton method [5, 37] which has also been used for solving this minimization problem (1.5) and can take advantage of the least-squares structure of the objective function.

A few methods have attempted to solve parameter estimation and compute a numerical solution to (1.1) simultaneously. Tjoa et.al. [33] combine a numerical solution of the collocation equations with an optimization over parameters to obtain a single constrained optimization problem. Similar idea can be found in [4] where a *multiple shooting method* was proposed that partitions the fitting interval $[t_0, t_n]$ into many subintervals, each having its own initial values. The parameters are held constant for all subintervals. Horbelt et.al [15] showed the effectiveness of a multiple shooting method by applying it to identify physical properties of a CO_2 laser.

The other class of optimization methods is composed of global optimization (GO) methods. Global optimization methods can be roughly classified as deterministic [17, 14] or stochastic [1, 35]. One weakness of stochastic GO methods is the lack of strong theoretical guarantees of convergence to the global optima, since stochastic techniques for global optimization ultimately rely on probabilistic arguments. Deterministic GO methods are, on the other hand, generally able to achieve a level of assurance that the global optimum will be located. However, one should note that no algorithm can solve general GO problems with certainty in finite time [23]. In general, the computational effort most GO methods need to achieve convergence increases very rapidly with the problem size, in our case quantified by s, the dimension of $\mathbf{\Theta}$. In this paper, we will focus on the use of a stochastic GO optimization method applied to formulation (1.6) .

1. Choose a mechanism parameterized by $\mathbf{v}$ which can be used to generate candidate values $\Theta \in \mathbb{R}^s$, and prescribe a value v_0.
2. Generate N different trial parameters $\Theta_i, i = 1, \ldots, N$ from the chosen mechanism parameterized by $\mathbf{v}$. For $i = 1, \ldots, N$, solve model (1.1) with Θ_i using an effective IVP solver and evaluate the objective function $\mathcal{L}(\Theta_i)$ or the performance function $S(\Theta_i)$.
3. Check stopping conditions, if satisfied, stop
4. Otherwise choose a better value for $\mathbf{v}$ based on the observations $\mathcal{L}(\Theta_i)$ or $S(\Theta_i)$ and return to step (2) .

Figure 1.2: Overview of the initial value approach for stochastic GO optimization

For stochastic GO methods, often a modified version of the initial value approach (Fig. (1.1)) is used (see Fig. (1.2)) . To better illustrate this modified version, we will introduce some basic notation and results from probability theory which are needed for the motivation and description of a stochastic GO method. First, a multivariate Gaussian distribution can be used to describe a particular distribution of vectors in $\mathbb{R}^s$. The corresponding probability density is

$$f(\mathbf{X}, \mathbf{v} = \{\mu, \sigma\}) = \prod_i^s \frac{1}{\sqrt{2\pi}\sigma_i} \exp(-\frac{(\mathbf{X}_i - \mu_i)^2}{2\sigma_i^2}) \tag{1.13}$$

where $\mathbf{v}$ is the set of $2s$ parameters used to specify the probability density and μ is the mean vector in which μ_i represents the mean of the i-th component of $\mathbf{X}$. Similarly, σ is the standard deviation vector where σ_i represents the standard deviation of the i-th component of $\mathbf{X}$. Here we assume all the components of data $\mathbf{X}$ are independent. We use this multivariate density distribution to generate a set of trial parameters required for Step 2 of Fig. (1.2) .

This multivariate Gaussian distribution is the most common example of the "mechanism" for Step 1 of Fig. (1.2) . The stochastic GO method we will consider, on each iteration, identifies a subset of the trials that are thought to be close to being optimal (called the "elites") , and then obtains an estimate of the mean and standard deviation vectors of the elites to be used to generate the trials on the next iteration. The hope is that the means of each component of the elite trials at each iteration will converge to the optimal parameter $\bar{\Theta}$ of the optimization problem (1.6) .

Here we review some related examples of stochastic GO methods and the key strategies they adopt for step 4 of Fig. (1.2) . 1) The early stochastic GO methods were adaptive stochastic methods, originally developed for engineering in the 1960s [22]. 2) Clustering methods [34] are based on the concept of multistart methods, that is, local methods started from different initial points. Because clustering methods aim to identify the vicinity of local optima, they can be more efficient and robust than standard multistart methods by avoiding repeated determination of the same local solutions. However, they often fail when there are a large number of parameters. 3) Another popular class of stochastic methods are evolutionary computation (EC) methods, also known as biologically inspired methods. These methods generate better and better solutions by iteratively simulating biological evolution. Classic examples are Genetic Programming (GP) [13], Evolutionary Programming (EP) [12] and Evolution Strategy (ES) [30]. 4) Other meta-heuristics stochastic GO methods have been also proposed like Taboo Search, Ant Colony Optimization, and particle swarm methods.

A Cross-Entropy method is a newly developed stochastic GO method. The Cross-Entropy method was motivated by an adaptive algorithm for estimating probabilities of rare events in complex stochastic networks and it involves variance minimization. It was realized that a simple modification of this approach could be applied to solve difficult combinatorial optimization as well. This paper is an attempt (the first that we are aware of) that to investigate the use of a Cross-Entropy approach in the framework of parameter estimation in ODEs and DDEs. We modify the general approach to be efficient and effective for this class of problems. The approach can be broken down into two key strategies: 1. Generate a random set of trial parameters according to a specified distribution. 2. From the values of the performance function associated with each trial parameter, update the probability distribution used to generate the random trials based on the principle of "importance sampling".

Until now we have focused on presenting a review of parameter estimation techniques for systems of ODEs (1.1) . We will now present a brief review of parameter estimation techniques for DDEs. The main difficulty, (see Baker and Paul [2]) , in extending an initial value approach to this class of problems, is that discontinuities arising from the initial point $\mathbf{y}_0$ may propagate into the objective function through the solution values that contain delay terms. It is not surprising that the assumption of the smoothness of the objective function may not hold in this case.

There have been some investigations (see [16], [19]) which address this difficulty by introducing splines to define the initial values and delay parameters. By carefully matching the nodes of the splines to the locations of the discontinuities, they are able to improve the effectiveness of this approach. Recently, Zivaripiran [38] developed a set of tools for investigating DDEs including simulation, sensitivity analysis and parameter

estimation. To avoid the potential non-smoothness difficulty, the points of discontinuity are automatically located. This is accomplished using a continuous extension of $y(t, \Theta)$ and its partial derivatives for evaluations of the objective function. However, this remedy procedure is only applicable in the event of a few parameters. When dealing with a large number of parameters, this procedure fails due to prohibitive computation burden of estimating the derivative information at all the potential discontinuities.

One advantage of cross-entropy methods is that they do not require any derivative information, so they can handle potential discontinuities that may arise in ODEs and DDEs. Based on the importance sampling strategy, cross-entropy methods update the parameterized probabilistic distribution function at each step by using only a subset of those generated approximations, the "elites", whose associated performance function values are near the best value observed so far.

In order to overcome the slow convergence rate and large computational cost, we modify an existing cross-entropy method by using a local evolution procedure. This process records the best samples so far and penalizes the "bad" samples based on the best one. The focus on the best samples can increase the convergence rate, while the evolution of bad ones is helpful to explore the whole parameter space and avoid the optimums already identified. In this way, the cross-entropy approach can be applied when there are a large number of parameters. A comparison with other cross-entropy strategies is presented in Chapter 4, from which we can see our modification can achieve both better accuracy and better efficiency.

We develop two versions of our algorithm. The first (the continuous version) assumes each component of $\Theta \in \mathbb{R}^s$ is in the range $[-M, M]$. The second version (the discrete version) assumes each $\Theta \in \mathbb{R}^s$ is approximated by its closest fixed point value. Since there are only a finite number of fixed point values in the range of $[-M, M]$, our discrete version will generally require less time to convergence. We will refer to the continuous version as the continuous coding scheme and the discrete version as the discrete coding scheme.

1.4 An Outline of the Thesis

In Chapter 2, we first give an overview of the basic idea of a cross-entropy method, then we provide a brief introduction to the concept of importance sampling and the Kullback-Leibler Distance, which play an important role in defining our cross-entropy methods. Then we give a detailed description on how cross-entropy can be used for parameter estimation. We review how cross-entropy is used in a combinatorial optimization setting, then modify the standard cross-entropy approach to gain robustness and improvement in efficiency when applied to our problems. The key idea is the local evolution procedure, which can make use of the best sample so far to speed the convergence, and which also takes advantage of "bad" samples to attempt to avoid local optimums. In the last section of chapter 2, we discuss in detail the two coding schemes: continuous cross-entropy and discrete cross-entropy. In Chapter 3, we provide details on the implementation of the proposed methods. In Chapter 4, we present numerical results on real-world problems involving both ODEs and DDEs, and compare our methods with other existing methods

with respect to both efficiency and accuracy. Also, we present a short discussion of the effect of sampling size N and choices of objective function. In Chapter 5, we summarize the thesis and discuss future work.

Chapter 2

The Cross-Entropy Algorithm

2.1 Kullback-Leibler Distance

Let f and h be two probability distribution functions associated with a vector $\mathbf{x}$, and $h(\mathbf{x})$ is zero only when $f(\mathbf{x})$ is zero. The Kullback-Leibler cross-entropy (CE) distance between f and h is defined as

$$\begin{aligned} \mathcal{D}_{KL}(f,h) &= \mathbb{E}_f \ln \frac{f(\mathbf{x})}{h(\mathbf{x})} \\ &= \int f(\mathbf{x}) \ln f(\mathbf{x}) \mu(dx) - \int f(\mathbf{x}) \ln h(\mathbf{x}) \mu(dx), \end{aligned} \tag{2.1}$$

where $\mathbb{E}_f$ means the expectation under the probability density f. Note that $\mathcal{D}_{KL}(f,h)$ is not a strict "distance" since $\mathcal{D}_{KL}(f,h) \neq \mathcal{D}_{KL}(h,f)$. Nonetheless, it is often useful to think of $\mathcal{D}_{KL}(f,h)$ as a distance because

$$\mathcal{D}_{KL}(f,h) \geq 0, \tag{2.2}$$

and $\mathcal{D}_{KL}(f,h) = 0$ if and only if $f(x) = h(x)$.

2.2 The Cross-Entropy for Parameter Estimation

In this section, we discuss how to apply a cross-entropy method for parameter estimation problems. We assume that $-M \leq \Theta_i \leq M, i = 1, \ldots, s$. We search for an optimal Θ in this hypercubic (denoted as $\Im$) in $\mathbb{R}^s$ for a known value $M > 0$. That is, we wish to find the maximum of S over $\Im$, and the corresponding states at which this maximum is attained. Let us denote the maximum by γ^*. Thus,

$$S(\bar{\Theta}) = \gamma^* = \max_{\mathbf{X} \in \Im} S(\mathbf{X}). \tag{2.3}$$

The goal is to estimate $\{\bar{\Theta}, \gamma^*\}$. As mentioned before, a set of candidate Θ (denoted as $\mathbf{X}_i, i = 1, \ldots, N$) is usually generated from a specific distribution, $f(X, \mathbf{v})$. So instead of directly searching for $\{\bar{\Theta}, \gamma^*\}$, we aim at estimating $\{\mathbf{v}^*, \gamma^*\}$, where $\mathbf{v}^*$ is associated

with an "optimal" distribution that is centered at the optimal parameter $\bar{\Theta}$. An iterative algorithm is proposed to solve this estimation problem. The idea is to construct a sequence of $\mathbf{v}_t$ and a sequence of levels γ_t and iterate in both γ_t and $\mathbf{v}_t$. We initialize by choosing a not very small $\rho \in (0,1)$, say $\rho = 0.01$ and by defining $\mathbf{v}_0 = \mathbf{u}$, for example, we can initialize the mean of the multivariate Gaussian as $\mu = \{0, .., 0\}$ and the standard deviation $\sigma = \{1, .., 1\}$. Next, we let γ_1 be such that, under the original density $f(\mathbf{X}; \mathbf{u})$, the probability $\mathbb{P}_{\mathbf{u}} I_{\{S(\mathbf{X}) \geq \gamma_1\}}$ is at least ρ. We then update v_1 based on the N trial parameters X_i generated by $f(\mathbf{X}; \mathbf{v_0})$, and repeat the last two steps iteratively with the goal of estimating the pair $\{\mathbf{v}^*, \gamma^*\}$. In other words, each iteration of the algorithm consists of two main phases. In the first phase, γ_t is updated, in the second $\mathbf{v}_t$ is updated. Specifically, starting with $\mathbf{v}_0 = \mathbf{u}$ we obtain the subsequent γ_t and $\mathbf{v}_t$ as follows:

1. **Adaptive Updating of γ_t.** For a fixed $\mathbf{v}_{t-1}$, we first generate N random sample parameters $\mathbf{X}_i, i = 1, \ldots, N$ according to the density function $f(X, v_{t-1})$, then with these trial parameters, we calculate the associated performance function $S(\mathbf{X}_i), i = 1, \ldots, N$ using a reliable IVP solver. These N values are then sorted in an increasing order $S(\mathbf{X}_{j_1}) \leq S(\mathbf{X}_{j_2}) \leq \ldots \leq S(\mathbf{X}_{j_N})$, where $j_1, j_2, \ldots, j_N$ is a permutation of $1, 2, \ldots, N$. We call $\mathbf{X}_{j_{\lceil (1-\rho)N \rceil}}, \ldots, \mathbf{X}_{j_N}$ the "elites", and the remaining $N - \lceil (1-\rho)N \rceil$ samples "bad samples". Let γ_t be a $(1-\rho)$-quantile of $S(\mathbf{X})$ under $\mathbf{v}_{t-1}$:

$$\gamma_t = S(\mathbf{X}_{j_{\lceil (1-\rho)N \rceil}}). \tag{2.4}$$

That is, γ_t satisfies

$$\mathbb{P}_{\mathbf{v}_{t-1}}(S(\mathbf{X}) \geq \gamma_t) \geq \rho, \tag{2.5}$$

$$\mathbb{P}_{\mathbf{v}_{t-1}}(S(\mathbf{X}) \leq \gamma_t) \leq 1 - \rho \tag{2.6}$$

where $\mathbf{X}$ follows the probability distribution $f(\cdot; \mathbf{v}_{t-1})$ (we denote it as $\mathbf{X} \sim f(\cdot; \mathbf{v}_{t-1})$) .

2. **Adaptive Updating of $\mathbf{v}_t$.** For fixed γ_t and $\mathbf{v}_{t-1}$, the updating of $\mathbf{v}_t$ is a key step. The important assumption here is that the "elite" samples are close to $\bar{\Theta}$. So the following truncated normal distribution can be seen as a trustful one that can generate samples near $\bar{\Theta}$:

$$f_{t-1}^* = I_{\{S(\mathbf{X}) \geq \gamma_t\}} f(\mathbf{X}, \mathbf{v}_{t-1}). \tag{2.7}$$

The hope is if $f(\mathbf{X}, \mathbf{v}_t)$ is close to this truncated distribution f_{t-1}^*, we can get a set of better trial samples which are closer to $\bar{\Theta}$. It is well known that, KL distance (cross-entropy) is a good measure of distance between distributions, hence, we try to solve the following minimization problem:

$$\mathbf{v}_t = \arg\min_{\mathbf{v}} \mathcal{D}_{KL}(f_{t-1}^*, f(\mathbf{X}, \mathbf{v})) \tag{2.8}$$

Using (2.1) and (2.7) , we can obtain an equivalent optimization problem to (2.8) :

$$\begin{aligned} \mathbf{v}_t &= \arg\max_{\mathbf{v}} \int I_{\{S(X) \geq \gamma_t\}} f(\mathbf{X}, \mathbf{v}_{t-1}) \ln f(\mathbf{X}, \mathbf{v}) dx \\ &= \arg\max_{\mathbf{v}} \mathbb{E}_{f(\mathbf{X}, \mathbf{v}_{t-1})} I_{\{S(X) \geq \gamma_t\}} \ln f(\mathbf{X}, \mathbf{v}) \end{aligned} \tag{2.9}$$

We may estimate $\mathbb{E}_{f(\mathbf{X},\mathbf{v}_{t-1})} I_{\{S(X)\geq\gamma_t\}} \ln f(\mathbf{X},\mathbf{v})$ by

$$\bar{E}(\mathbf{v}) = \frac{1}{N}\sum_{i=1}^{N} I_{\{S(\mathbf{X_i})\geq\gamma_t\}} \ln f(\mathbf{X_i};\mathbf{v})dx \tag{2.10}$$

where $\mathbf{X} \sim f(\cdot;\mathbf{v}_{t-1})$. In typical applications, the function $\bar{E}_(\mathbf{v})$ in (2.10) is convex and differentiable with respect to $\mathbf{v}$ [29]. Thus, $\mathbf{v}_t$ may be readily obtained by solving the system of equations

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(\mathbf{X_i})\geq\gamma\}} \nabla \ln f(\mathbf{X_i};\mathbf{v})dx = \mathbf{0}, \tag{2.11}$$

where the gradient is with respect to $\mathbf{v}$. The advantage of this approach is that the solution of (2.11) can often be calculated analytically, especially when the distributions of the random variables belong to a natural exponential family (NEF) [28].

Instead of updating the parameter vector $\mathbf{v}_{t-1}$ to $\mathbf{v}_t$ directly via (2.11) , we use a *smoothed* updating procedure as has been suggested in [29]:

$$\mathbf{v}_t = \alpha\mathbf{w}_t + (1-\alpha)\mathbf{v}_{t-1} \tag{2.12}$$

where $\mathbf{w}_t$ is the vector derived via (2.11) . This smooth updating rule can prevent the algorithm from being trapped at local optimums. We found empirically that a value of α such that $0.6 \leq \alpha \leq 0.9$ gives the best results.

As to the stopping rule, we have two similar methods to decide when to terminate the iterations. First, we keep record of γ_t and stop when $\gamma_t \cong \gamma_{t-1} \cong \ldots \cong \gamma_{t-k}$ for a certain number k. Note that $\gamma_t \cong \gamma_{t-1}$ means $|\gamma_t - \gamma_{t-1}| \leq tol$, where tol is a small positive number. The second way is to stop the iterations when $\mathbf{v}_t \cong \mathbf{v}_{t-1} \cong \ldots \cong \mathbf{v}_{t-k}$. In this investigation, we choose the first criterion, and set $k = 5$, $tol = 10^{-6}$.

We summarize the CE algorithm in Fig. (2.1) .

1. Define $\mathbf{v}_0 = \mathbf{u}$. Set $t = 1$, and pick ρ and α.
2. Generate a sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $f(\cdot;\mathbf{v}_{t-1})$ and compute the sample $(1-\rho)$-quantile γ_t of the performance according to (2.4) .
3. Calculate $\mathbf{v}_t$ using the **same** samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ by solving the system of equations (2.11) and using the smoothing update rule in (2.12) .
4. If, for some $t \geq 5$, $\gamma_t \cong \gamma_{t-1} \cong \ldots \cong \gamma_{t-5}$, then **stop**; otherwise, set $t = t + 1$ and return to step 2.

Figure 2.1: Overview of the CE Algorithm for GO Optimization

2.3 Modifications of the Cross-Entropy Algorithm

As with most heuristic global optimization methods, CE suffers from a large computational cost associated with each iteration. Generally speaking, to make CE useful, the sample size N usually lies in the interval $100 \leq N \leq 1000$. This means that to solve the parameter estimation problem for an ODE, we have to run the ODE solver at least TN

times, where T is the number of the iterations before convergence. For large systems which have a large number of unknown parameters, the time needed to use CE methods is prohibitive. Looking into the core idea of CE methods, we find that only partial information in the N samples is used. The key step of CE methods is to update the parameter vector $\mathbf{v}_t$ via solving the following system:

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(\mathbf{X_i})\geq\gamma\}} \nabla \ln f(\mathbf{X_i};\mathbf{v})dx = \mathbf{0}, \tag{2.13}$$

We can see that only those "elites" samples which satisfy $\{S(\mathbf{X_i}) \geq \gamma\}$ are used. The remaining samples (which are not good enough) are ignored. One straightforward idea to speed up the CE method is to make use of the ignored samples. We proposed a method to speed up convergence, called *local evolution*. The core idea behind our "speed-up" heuristic is to evolve the "bad" (abandoned) samples to good ones. Every iteration, we record the best-so-far sample, denoted $\mathbf{X}^{best}$. For any "bad" sample $\mathbf{X}^{bad}$, we shift it toward $\mathbf{X}^{best}$ using the updated value:

$$\mathbf{X}^{bad} \leftarrow (\delta * \mathbf{X}^{bad} + (1-\delta) * \mathbf{X}^{best}). \tag{2.14}$$

where $\delta \in [0,1]$ is a weight for the "bad" samples. In this paper, we fix $\delta = 0.2$. After performing the procedure of local evolution, we can update the $\mathbf{v}_t$ by solving the new system

$$\frac{1}{N}\sum_{i=1}^{N}(I_{\{S(\mathbf{X_i})\geq\gamma\}} + \zeta * I_{\{S(\mathbf{X_i})<\gamma\}}) \bigtriangledown \ln f(\mathbf{X_i};\mathbf{v})dx = \mathbf{0}, \tag{2.15}$$

where ζ is also a weight parameter. In this paper, we fix $\zeta = 0.5$.

The advantage of the use of local evolution is two-fold: 1) to aid convergence and reduce the sample size because we employ the information contained in both abandoned samples and "best-so-far" sample; 2) to avoid local optima since local evolution expands the exploration of the solution space and encourages the search to move to regions that contain additional local or global optima. An overview of the resulting modified Cross-Entropy method is presented in Fig. (2.2) .

1. Define $\mathbf{v}_0 = \mathbf{u}$. Set $t = 1$ and pick ρ and α.
2. Generate a sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $f(\cdot; \mathbf{v}_{t-1})$ and compute the sample $(1-\rho)$-quantile γ_t of the performance according to (2.4) .
3. For those samples satisfying $\{S(\mathbf{X}_i) \leq \gamma_t\}$, perform local evolution using (2.14) .
4. Calculate $\mathbf{v}_t$ using the trial samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ and solving the system of equations in (2.15) .
5. If for some $t \geq 5$, $\gamma_t \cong \gamma_{t-1} \cong \ldots, \cong \gamma_{t-5}$, then **stop**; otherwise, set $t = t+1$ and return to step 2.

Figure 2.2: Overview of the Modified Cross-Entropy (MCE) Algorithm

2.4 Continuous Cross-Entropy Method

In this section, we discuss how to associate a continuous probability density to express the pdfs for each of the parameters. As pointed out above, to apply the CE method to such a problem, we have to specify the sampling distribution and the updating rules for its parameters. The choice of the sampling distribution is quite arbitrary. In this paper, we use the truncated Gaussian with independent components, as was done in [7, 20]. The s-dimensional normal distribution with independent components, mean vector $\mu = (\mu_1, \ldots, \mu_s)$ and variance vector $\sigma^2 = (\sigma_1^2, \ldots, \sigma_s^2)$ is denoted by $\mathcal{N}(\mu, \sigma^2)$.

Using this parameterized distribution, it can be shown that the updating rule that corresponds to (2.11) is:

$$\begin{aligned} \widetilde{\mu}_t &= \frac{\sum_{i=1}^N I_{\{S(\mathbf{X}_i)\geq\gamma\}}\mathbf{X}_i}{\sum_{i=1}^N I_{\{S(\mathbf{X}_i)\geq\gamma\}}} \\ \widetilde{\sigma^2}_t &= \frac{\sum_{i=1}^N I_{\{S(\mathbf{X}_i)\geq\gamma\}}(\mathbf{X}_i - \widetilde{\mu}_t)^2}{\sum_{i=1}^N I_{\{S(\mathbf{X}_i)\geq\gamma\}}} \end{aligned} \tag{2.16}$$

We use the smooth updating rule (2.12) to "smooth" the mean:

$$\mu_t = \alpha\widetilde{\mu}_t + (1-\alpha)\mu_{t-1}. \tag{2.17}$$

To prevent the smoothed algorithm for the variance vectors from being trapped in local optima, Rubinstein and Kroese [28] proposed the use of dynamic smoothing where, at each iteration, the variance is updated using a smoothing parameter β_t by

$$\begin{aligned} \beta_t &= \beta_0 - \beta_0(1-\frac{1}{t})^c, \\ \sigma_t^2 &= \beta_t\widetilde{\sigma_t^2} + (1-\beta_t)\sigma_{t-1}^2, \end{aligned} \tag{2.18}$$

where β_0 and c are fixed numbers. In our experiments, we set $\beta = 0.95$ and $c = 1$. There are many possible stopping criteria; we use the stopping rule to stop when the best value over 5 iterations does not change significantly. We summarize the continuous cross-entropy (CCE) method in Fig. (2.3)

For the modified version, the main difference lies in the calculation of $\{\widetilde{\mu}_t, \widetilde{\sigma^2}_t\}$: First, we perform the local evolution procedure on those samples that satisfy $\{S(\mathbf{X}_i \leq \gamma)\}$, and the updating rule is:

$$\begin{aligned} \widetilde{\mu}_t &= \frac{\sum_{i=1}^N(\zeta I_{\{S(\mathbf{X}_i)\leq\gamma\}} + I_{\{S(\mathbf{X}_i)\geq\gamma\}})\mathbf{X}_i}{\sum_{i=1}^N(\zeta I_{\{S(\mathbf{X}_i)\leq\gamma\}} + I_{\{S(\mathbf{X}_i)\geq\gamma\}})} \\ \widetilde{\sigma^2}_t &= \frac{\sum_{i=1}^N(\zeta I_{\{S(\mathbf{X}_i)\leq\gamma\}} + I_{\{S(\mathbf{X}_i)\geq\gamma\}})(\mathbf{X}_i - \mu_t)^2}{\sum_{i=1}^N(\zeta I_{\{S(\mathbf{X}_i)\leq\gamma\}} + I_{\{S(\mathbf{X}_i)\geq\gamma\}})} \end{aligned} \tag{2.19}$$

We summarize the modified continuous cross-entropy (MCCE) algorithm in Fig. (2.4)

1. Initialize $\{\mu_0, \sigma_0^2\}$, $\beta_0 = 0$. Set $t = 1$ and pick ρ and α.
2. Generate a sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $\mathcal{N}(\mu_{t-1}, \sigma_{t-1}^2)$
3. Solve N ODE systems with the N generated samples, and calculate the objective functions in (1.6) .
4. Compute the sample $(1-\rho)$-quantile γ_t of the performance according to (2.4) .
5. Obtain $\{\widetilde{\mu}_t, \widetilde{\sigma^2}_t\}$ using (2.16) , then update β_t as in (2.18) .
6. Update μ_t as in (2.17) and σ_t^2 as in (2.18).
7. If for some $t \geq 5$, $\gamma_t \cong \gamma_{t-1} \cong \ldots, \cong \gamma_{t-5}$, then **stop**; otherwise, set $t = t + 1$ and return to step 2.

Figure 2.3: Overview of the algorithm of Continuous Cross-Entropy (CCS)

1. Initialize $\{\mu_0, \sigma_0^2\}$. Set $t = 1$ and pick ρ and α.
2. Generate a sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $\mathcal{N}(\mu_{t-1}, \sigma_{t-1}^2)$
3. Solve N ODE systems with the N generated samples, and calculate the objective functions in (1.6) .
4. Compute the sample $(1-\rho)$-quantile γ_t of the performance according to (2.4) , and perform evolution on those samples $\{\mathbf{X}_i | S(\mathbf{X}_i) \leq \gamma_t\}$.
5. Obtain $\{\widetilde{\mu}_t, \widetilde{\sigma^2}_t\}$ using (2.19) , and update β_t using (2.18) .
6. Update μ_t as in (2.17) and σ_t^2 as in (2.18).
7. If for some $t \geq 5$, $\gamma_t \cong \gamma_{t-1} \cong \ldots, \cong \gamma_{t-5}$, then **stop**; otherwise, set $t = t + 1$ and return to step 2.

Figure 2.4: Overview of the algorithm of Modified Continuous Cross-Entropy (MCCE)

2.5 Discrete Cross-Entropy Method

One drawback with the continuous cross-entropy method is the high computation cost due to the high dimensional search space. To reduce the search space of continuous cross-entropy methods, we try to search for the optimal parameters over a finite subset of $\Im$. As above, we assume all the parameters needed to be estimated are bounded in a hypercubic $\Im$ in which $|\Theta|_\infty \leq M$. For any $\mathbf{\Theta} \in \Im$, we can approximate each component $\mathbf{\Theta}_j, j = 1, \ldots, s$ of $\mathbf{\Theta}$ by its finite fixed point approximation. Specifically, we can approximate $\mathbf{\Theta}_j, j = 1, \ldots, s$ by

$$\hat{\mathbf{\Theta}}_j = b_0^j b_1^j b_2^j \ldots b_K^j \bullet b_{K+1}^j b_{K+2}^j \ldots b_{K+L}^j \tag{2.20}$$

where $b_i^j, i = 0, 1, \ldots, K+L$ is either 0 or 1. b_0^j represents the sign of $\mathbf{\Theta}_j$, and both K and L are fixed integers. For our implementation, we set $K = \lceil \log_2 M \rceil$, and $L = 6$. With this representation system, we have a bounded relative error:

$$\frac{|\hat{\mathbf{\Theta}}_j - \mathbf{\Theta}_j|}{|\mathbf{\Theta}_j|} \leq 2^{-(1-L)} \tag{2.21}$$

With this finite fixed point system, we can approximate any s-dimension $\mathbf{\Theta} \in \mathbb{R}^s$ with a binary vector $\hat{\mathbf{\Theta}}$ whose dimension is $s \times (K + L + 1)$:

$$\hat{\mathbf{\Theta}} = [\hat{\mathbf{\Theta}}_1^T, \hat{\mathbf{\Theta}}_2^T, \ldots, \hat{\mathbf{\Theta}}_s^T]^T \tag{2.22}$$

where $\hat{\mathbf{\Theta}}_i$ is a finite fixed point binary approximation of i-th parameter value $\mathbf{\Theta}^i, i = 1, \ldots, s$.We call the variant of cross-entropy method using this finite number floating representation system a discrete cross-entropy (DCE) method. On each iteration of our DCE method, we need to generate N trial values, $\hat{\mathbf{\Theta}}$, in this fixed point floating system. We denote these trial samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$, where each sample $\mathbf{X}_i, i = 1, \ldots, N$ is a binary $s(K + L + 1)$-dimensional vector. For each binary $s(K + L + 1)$-dimensional vector $\mathbf{X}$, we denote the l-th component of the vector $\mathbf{X}$ as $X_{(l)}$. Since $\mathbf{X}$ is a binary vector, we can set its density pdf to be a Bernoulli distribution parameterized by $p \in \mathbb{R}^{s(K+L+1)}$ (Ber (p)), that is,

$$f(\mathbf{X}; \mathbf{p}) = \Pi_{l=1}^{s(K+L+1)} p_l^{X_{(l)}} (1 - p_l)^{1 - X_{(l)}} \tag{2.23}$$

and since each $X_{(l)}$ can only be 0 or 1,

$$\frac{\partial}{\partial p_l} \ln f(\mathbf{X}; \mathbf{p}) = \frac{X_{(l)}}{p_l} - \frac{1 - X_{(l)}}{1 - p_l} = \frac{X_{(l)} - p_l}{(1 - p_l) p_l}. \tag{2.24}$$

Substituting (2.24) into (2.11) , we have

$$\frac{\partial}{\partial p_l} \sum_{i=1}^{N} I_{\{S(\mathbf{X_i}) \geq \gamma\}} \ln f(\mathbf{X_i}; \mathbf{p}) = \frac{1}{(1 - p_l) p_l} \sum_{i=1}^{N} I_{\{S(\mathbf{X_i}) \geq \gamma\}} (X_{i(l)} - p_l) = 0, \tag{2.25}$$

where $X_{i(l)}$ represents the l-th component of the vector $\mathbf{X_i}$. Thus, we have

$$p_l = \frac{\sum_{i=1}^{N} I_{\{S(\mathbf{X_i}) \geq \gamma\}} X_{i(l)}}{\sum_{i=1}^{N} I_{\{S(\mathbf{X_i}) \geq \gamma\}}} \tag{2.26}$$

for $l = 1, \ldots, s(K + L + 1)$.

We can now present our discrete Cross-Entropy (DCE) algorithm (Fig. (2.5)) and corresponding Modified Discrete Cross-Entropy (MDCE) algorithm (Fig. (2.6)) for parameter estimation of ODEs.

1. Initialize $\mathbf{v}_0 = [\frac{1}{2}, \frac{1}{2}, \ldots]$. Set $t = 1$.
2. Generate N trial samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $Ber(\mathbf{v}_{t-1})$
3. Obtain N possible parameters $\{\hat{\mathbf{\Theta}}_i\}_{i=1}^N$ by decoding the samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ into decimal values.
4. Calculate the objective functions (1.6) with the decoded N parameters $\{\hat{\mathbf{\Theta}}_i\}_{i=1}^N$, and compute the sample $(1 - \rho)$-quantile γ_t of the performance according to (2.4) .
5. Update $\mathbf{v}_t$ by (2.12) and (2.26) .
6. If, for some $t \geq 5$, $\gamma_t \cong \gamma_{t-1} \cong \ldots, \cong \gamma_{t-5}$, then **stop**; otherwise, set $t = t + 1$ and return to step 2.

Figure 2.5: Overview of the algorithm of Discrete Cross-Entropy (DCE)

1. Initialize $\mathbf{v}_0 = [\frac{1}{2}, \frac{1}{2}, \ldots]$. Set $t = 1$.
2. Generate N trial samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $Ber(\mathbf{v}_{t-1})$
3. Obtain N possible parameters $\{\hat{\mathbf{\Theta}}_i\}_{i=1}^N$ by decoding the samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ into decimal values.
4. Calculate the objective functions (1.6) with the decoded N parameters $\{\hat{\mathbf{\Theta}}_i\}_{i=1}^N$, and compute the sample $(1 - \rho)$-quantile γ_t of the performance according to (2.4) .
5. For those satisfying $\{S(\mathbf{X}_i) \leq \gamma_t\}$, perform local evolution using (2.14) .
6. Update $\mathbf{v}_t$ by (2.12) and (2.26) .
7. If, for some $t \geq 5$, $\gamma_t \cong \gamma_{t-1} \cong \ldots, \cong \gamma_{t-5}$, then **stop**; otherwise, set $t = t + 1$ and return to step 2.

Figure 2.6: Overview of the algorithm of Modified Discrete Cross-Entropy (MDCE)

Chapter 3

Implementation Details

In this section, we discuss some implementation details of the proposed methods.

In many applications, we often encounter stiff ODEs. We use different solvers for stiff and non-stiff ODEs. For non-stiff ODEs, we apply a 4th order Runge-Kutta method with a variable time step for efficient computation (implemented in Matlab as *ode45*) ; For stiff ODEs, we use an implicit method (implemented in Matlab as *ode15s*) . For DDE parameter fitting, we used the second order method (*dde23* in Matlab) . We set the precision of the ODE and DDE solvers to be 10^{-6}. One challenge in parameter estimation problems is the possibility that, the IVPs may be stiff only for some of the trial parameter values. This means we have to design an adaptive mechanism to change the solver according to the parameters. We address this difficulty by setting non-stiff solver as the default and monitoring the computation time of every updating iteration. If the computation cost is too high, we switch to the stiff solver.

As discussed earlier, the proposed methods are insensitive to the parameter settings. We adopt a fixed setting of parameters in all experiments. For the DCE method, the parameters are $N = 200, \rho = 0.05, \alpha = 0.8$; for the MDCE method, the parameters are: $N = 30, \rho = 0.4, \alpha = 0.8$; for the CCE method, the parameters are $N = 200, \rho = 0.05, \alpha = 0.8, \beta = 0.95, q = 0.925$; for the MCCE method, the parameters are $N = 30, \rho = 0.4, \alpha = 0.8, \beta = 0.95, q = 0.925$. We will observe that, for the modified version of CE, we typically use about 85% less samples than for CE methods to obtain the same accuracy.

Chapter 4

Numerical Experiments

In this section, we report numerical results on four test problems. The first one is a classic problem often cited in the literature on parameter estimation [8, 32, 36]. The second and third ones are application problems. The fourth problem is a DDE problem.

To better evaluate the quality of the proposed method on parameter estimation problem, we use two measures of performance: 1) $J = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d}(\bar{Y}_{ij} - Y_{ij}(\Theta))^2$. This quantity measures the consistency between all the observation points and the estimated ones. 2) CPU_T: Time required to converge, which reflects the convergence rate and the time complexity of the iterations.

4.1 Barnes' problem

First, we compare our modified cross-entropy methods to the standard cross-entropy methods on a classic test problem [8, 32, 36]. The problem consists of 2 ODE's with 3 parameters in the definition of $g(\cdot)$:

$$\begin{aligned} y_1' &= k_1 y_1 - k_2 y_1 y_2, \quad y_1(0) = a \\ y_2' &= k_2 y_1 y_2 - k_3 y_2, \quad y_2(0) = b. \end{aligned} \tag{4.1}$$

This is a small non-stiff problem but is interesting since there are several local minima of the objective function [8]. Some existing methods for parameter estimation, e.g. Gauss-Newton algorithm in [8], Sequential Quadratic Programming in [37] suffer from an extreme sensitivity to the initial guess, that is, most starting values for the parameters will lead to local minimum for those methods.

Our methods, however, are insensitive to the initial guess of parameters. For DCE and MDCE, we set the initial probability distribution to be 0.5 for all components of Θ; for CCE and MCCE, we set the initial guess of the distribution ($v_0 = \{\mu_0, \sigma_0\}$) to be 10 for every values in μ_0, and 1 for every value in σ_0. First, we generate 100 data points on the interval $[0, 10]$ with no noise. We conduct 3 different experiments: 1) Fix initial values a, b first, estimate parameters $\Theta = (k_1, k_2, k_3)$; 2) Fix the parameters (k_1, k_2, k_3) and estimate the initial values (a, b); 3) Keep $\Theta = (k_1, k_2, k_3, a, b)$ unknown. We independently run each experiment 100 times and report the mean value and variance for each parameter value of convergence and the associated values of J and CPU_T.

Table 4.1: Results for Barnes' Problem with (k_1, k_2, k_3) unknown and no noise is added.

Methods	$\theta = \{k_1, k_2, k_3\} \quad (\theta_{True} = \{1, 1, 1\})$	J	CPU_T
DCE	$\{0.924 \pm 0.033, 1.116 \pm 0.040, 1.105 \pm 0.031\}$	0.028 ± 0.008	16.79 ± 0.12
MDCE	$\{1.004 \pm 0.013, 1.001 \pm 0.018, 0.998 \pm 0.011\}$	0.008 ± 0.001	5.94 ± 0.05
CCE	$\{1.097 \pm 0.033, 1.045 \pm 0.023, 0.905 \pm 0.028\}$	0.013 ± 0.005	50.79 ± 0.52
MCCE	$\{1.031 \pm 0.011, 1.005 \pm 0.009, 1.000 \pm 0.008\}$	0.006 ± 0.001	30.00 ± 0.45

Table 4.2: Results for Barnes' Problem with initial values (a, b) unknown and no noise is added.

Methods	$\{a, b\} \quad (\{a, b\}_{True} = \{1, 0.3\})$	J	CPU_T
DCE	$\{0.950 \pm 0.037, 0.326 \pm 0.045\}$	0.025 ± 0.008	12.48 ± 0.11
MDCE	$\{1.001 \pm 0.018, 0.305 \pm 0.011\}$	0.010 ± 0.001	3.87 ± 0.04
CCE	$\{1.058 \pm 0.024, 0.312 \pm 0.020\}$	0.013 ± 0.004	45.46 ± 0.47
MCCE	$\{1.001 \pm 0.014, 0.304 \pm 0.009\}$	0.005 ± 0.001	27.15 ± 0.35

Table 4.3: Results on Barnes' Problem with all the parameters (k_1, k_2, k_3, a, b) unknown.

Methods	$\boldsymbol{\Theta} = \{k_1, k_2, k_3, a, b\} \quad (\boldsymbol{\Theta}_{True} = \{1, 1, 1, 1, 0.3\})$	J	CPU_T
DCE	$\{1.124 \pm 0.051, 1.094 \pm 0.048, 0.943 \pm 0.061, 0.904 \pm 0.047, 0.351 \pm 0.085\}$	0.059 ± 0.012	34.48 ± 0.41
MDCE	$\{1.012 \pm 0.024, 1.021 \pm 0.023, 0.992 \pm 0.040, 0.989 \pm 0.031, 0.319 \pm 0.043\}$	0.019 ± 0.009	10.14 ± 0.28
CCE	$\{1.117 \pm 0.038, 1.087 \pm 0.031, 0.956 \pm 0.049, 0.954 \pm 0.034, 0.334 \pm 0.070\}$	0.027 ± 0.010	186.04 ± 0.95
MCCE	$\{1.009 \pm 0.018, 1.012 \pm 0.014, 0.994 \pm 0.029, 0.994 \pm 0.027, 0.3081 \pm 0.031\}$	0.012 ± 0.003	32.08 ± 0.57

The results (see Table. (4.1) ,Table. (4.2) ,Table. (4.3)) show that the proposed methods can estimate the parameters. Specifically, we can see that the modified version can obtain both high accuracy and efficiency. Also, the continuous version is more accurate but slower to converge than discrete version.

4.2 FitzHugh-Nagumo Problem

Our second problem is a real example arising in modeling the behavior of spike potentials in the giant axon of squid neurons. The FitzHugh-Nagumo equations were developed by FitzHugh [26] and Nagumo et al. [24], as a simplification of a description of the behavior of spike potentials in the giant axon of squid neurons.

$$\begin{aligned} V' &= c(V - \frac{V^3}{3} + R), \\ R' &= -\frac{1}{c}(V - a + bR). \end{aligned} \tag{4.2}$$

The system describes the reciprocal dependencies of the voltage V across an axon membrane and a recovery variable R representing outward currents. Although not intended to provide a close fit to neural spike potential data, solutions to the FitzHugh-Nagumo ODEs do exhibit qualitative features that are common to elements of biological neural networks [27].

We set the initial known conditions as $(V(0), R(0)) = (-1, 1)$. The parameters are $\theta = \{a, b, c\}$, to which we assign the values $(0.25, 0.5, 3.5)$ respectively to define the

reference solutions. As discussed in [11], the existence of many local optimum makes the search for optimal parameters difficult. Algorithms such as simulated annealing, have been proposed to overcome this problem, although these are very computationally demanding. If we fix the value of c and vary the values of a and b, we can show the corresponding value of sum of squares of error ($SSE = \sum_i^n (Y_{ij}(a, b, c = 3.5) - Y_{ij}(a = 0.25, b = 0.5, c = 3.5))$) in Fig. (4.1) . As we can see from Fig. (4.1) , the features of the surface of SSE include "ripples" due to changes in the shape and period of the limit cycle and breaks due to bifurcation, or sharp changes in behavior.

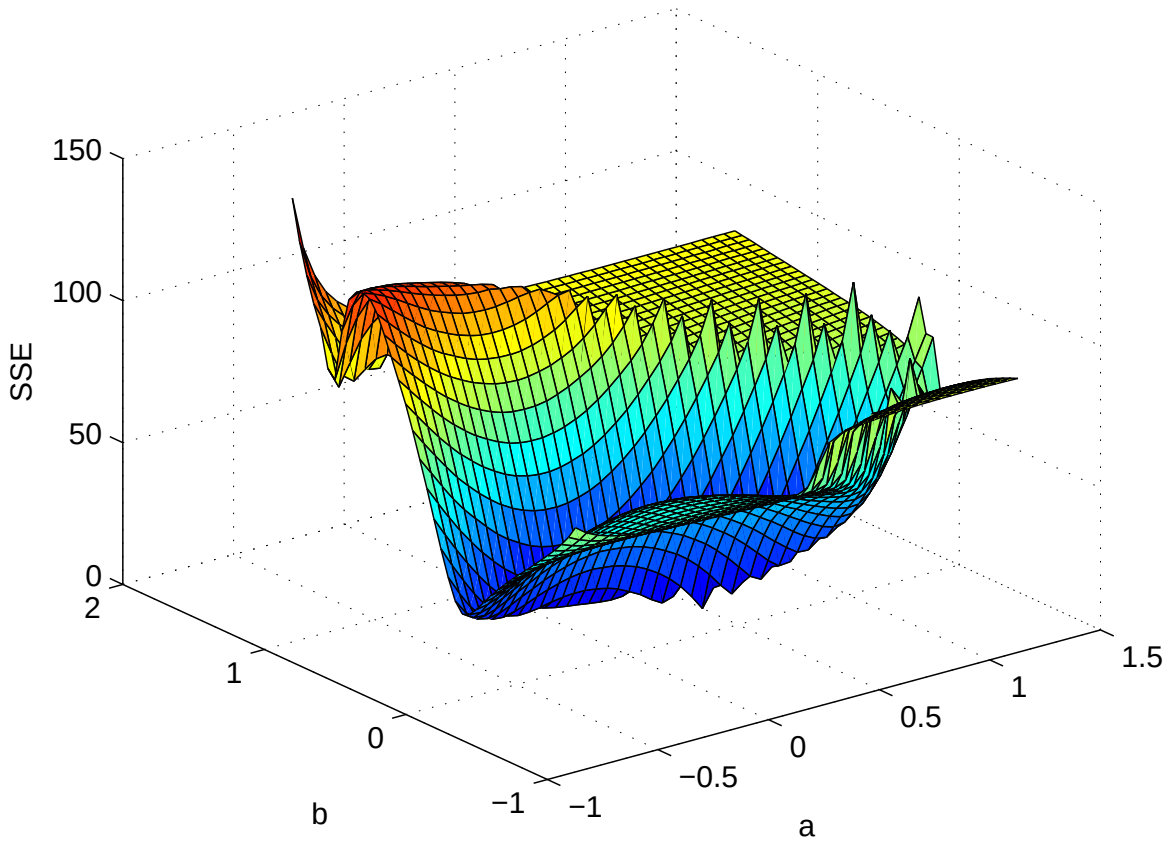

Figure 4.1: Response surface for solutions of the FitzHugh-Nagumo equations as parameters a and b are varied. $c = 3.5$ and initial conditions $(V(0), R(0)) = (-1, 1)$ are held constant.

Various scales of noise are added to the experiments. Measurements are generated by adding random noise $N(0, \sigma^2)$ with standard deviations σ of 1.0,0.1,0.01. We generate 200 equally distributed observations in the time range $[0, 20]$. We also provide some comparisons with other existing state-of-the-art methods for parameter estimation: 1) *diffpar* (differential equations with unknown parameters) [8], which uses a Gauss-Newton method to search for the optimal solution; 2) A multilevel Coordinate Search (*MCS*) algorithm by Huyer and Neumaier [18] which is an intermediate approach between a

Table 4.4: Values of J at convergence for FitzHugh-Nagumo Problem with different scales of noise, (200 observation points) .

Methods	$\sigma = 1$	$\sigma = 0.1$	$\sigma = 0.01$
MDCE	0.0820	0.0194	0.0022
MCCE	0.0814	0.0172	0.0016
diffpar	0.6521	0.3175	0.1010
MCS	0.4708	0.1291	0.0316
uES	0.3140	0.0932	0.0227

Table 4.5: CPU_T required for methods on FitzHugh-Nagumo Problem with different scales of noise, (200 observation points) .

Methods	$\sigma = 1$	$\sigma = 0.1$	$\sigma = 0.01$
MDCE	4.52	4.02	3.85
MCCE	24.6	20.9	19.4
diffpar	28.6	26.7	25.9
MCS	186	180	178
uES	317	305	294

Table 4.6: Values of J on FitzHugh-Nagumo Problem with different scales of noise, (50 observation points) .

Methods	$\sigma = 1$	$\sigma = 0.1$	$\sigma = 0.01$
MDCE	0.1529	0.0302	0.0029
MCCE	0.1417	0.0293	0.0020
diffpar	0.8520	0.594	0.3815
MCS	0.7687	0.4914	0.1586
uES	0.6170	0.3682	0.1763

purely heuristic method and one that allows an assessment of the quality of the minimum obtained. It has an initial global phase after which a local procedure, based on an SQP algorithm, is used; 3) The unconstrained Evolution Strategy (*uES*) which is a (μ, λ)-ES evolutionary optimization algorithm developed by Schwefel [30].

We report the results in Table. (4.4) and Table (4.5) , in which, accuracy and efficiency are demonstrated, respectively. In this test, only one run was made fore each problem.

We can see that both MDCE and MCCE are effective and efficient. Also, the comparisons shows that *diffpar* requires less time, but is less accurate than the heuristic methods *MCS* and *uES*. However, our modified methods can achieve both accuracy and efficiency.

As is well-known, optimization methods may perform poorly when too few samples are provided. We generate less dense samples (50 observations) , and also add noise. Fig. (4.2) shows the results of fitted trajectories with different scales of noise using MDCE (It is almost the same using MCCE so we omit it) . We can see that our methods can capture the underlying structure of the true trajectories, even when the observations are sparse and noisy. Table. (4.6) shows that the other methods perform considerably worse when there are only a few sample observations. However, our methods can still obtain reliable results in this case.

4.3 Calcium Ions Simulation Problem

Calcium ions are an important second messager substance in eucaryotic cells. It is known that, calcium ions plays a significant role in the cellular information processing system. It has been observed that the concentration of the cytoplasmatic calcium ions may exhibit oscillations. This has been modeled in [21] for a specific set of parameters and reported complex or chaotic behaviour. The main stages of the calcium signalling pathway are activation of the phospolipase C (PLC) enzyme by the activated G_α unit of a G-Protein linked receptor.

For the following simulation study we used the most complex mathematical model presented in [21, 25]. This model consists of four state variables representing the concentrations of: 1) the active G_α unit, G^*_α; 2) the active PLC, PLC^*; 3) the free calcium in the cytoplasm, Ca_{cyt} and 4) the calcium in the endoplasmatic reticulum, Ca_{er}. The dynamics of the model is then given by the following system of differential equations:

$$\begin{aligned}
\frac{d}{dt}G^*_\alpha &= k_1 + k_2 G^*_\alpha - k_3 PLC^* \frac{G^*_\alpha}{G^*_\alpha + Km_1} - k_4 Ca_{cyt} \frac{G^*_\alpha}{G^*_\alpha + Km_2} \\
\frac{d}{dt}PLC^* &= k_5 G^*_\alpha - k_6 \frac{PLC^*}{PLC^* + Km_3} \\
\frac{d}{dt}Ca_{cyt} &= k_7 PLC^* Ca_{cyt} \frac{Ca_{er}}{Ca_{er} + Km_4} + k_8 PLC^* + k_9 G^*_\alpha - k_{10} \frac{Ca_{cyt}}{Ca_{cyt} + Km_5} - k_{11} \frac{Ca_{cyt}}{Ca_{cyt} + Km_6} \\
\frac{d}{dt}Ca_{er} &= -k_7 PLC^* Ca_{cyt} \frac{Ca_{er}}{Ca_{er} + Km_4} + k_{11} \frac{Ca_{cyt}}{Ca_{cyt} + Km_6}
\end{aligned} \tag{4.3}$$

where the 17 parameters are chosen in the following manner: $k_1 = 0.09, k_2 = 2, k_3 = 1.27, k_4 = 3.73, k_5 = 1.27, k_6 = 32.24, k_7 = 2, k_8 = 0.05, k_9 = 13.58, k_{10} = 153, k_{11} = 4.85, Km_1 = 0.19, Km_2 = 0.73, Km_3 = 29.09, Km_4 = 2.67, Km_5 = 0.16, Km_6 = 0.05$. For this specific parametrization, the solution of (4.3) can exhibit a limit cycle. As initial values we use $G^*_\alpha(0) = 0.12, PLC^*(0) = 0.31, Ca_{cyt}(0) = 0.0058$ and $Ca_{er}(0) = 4.3$ to be fixed. We choose interval $[0, 20]$ with the sampling interval set to $\triangle t = 0.1$. This leads to 200 observation points. As in [25], we chose some of the parameters to be estimated. We keep k_6, k_{10}, Km_3 fixed since their values are quite large compared to the others which makes them difficult to estimate. We estimate the remaining 14 unknown parameters.

We give results when there is no noise (see Fig. (4.3)) . It is clear that, even with so many parameters unknown, our methods can still capture the underlying structure of the complex system. Also, we add different scales of noise. Similar to the section (4.2) , we set the variance of the noise $\sigma = \{1, 0.1, 0.01\}$. In Table (4.7) we also provide a comparison with the other three methods. We observe that our methods perform better even when the number of parameters is large. Other methods can easily break down when dealing with complex system due to the fact that they either need good approximation of derivative which is sometimes impossible in complex system, or they require a prohibitive amount of CPU time.

Table 4.7: Values of J on Calcium Ions Problem with different scales of noise, (14 parameter values and 200 observation points) .

Methods	$\sigma = 1$	$\sigma = 0.1$	$\sigma = 0.01$
MDCE	0.1002	0.0497	0.0304
MCCE	0.0957	0.0413	0.0215
diffpar	1.0740	0.4130	0.2885
MCS	0.5678	0.2674	0.1359
uES	0.8752	0.5681	0.2478

Table 4.8: Comparison on the evaluation metric J in the Kermack-McKendrick model

Methods	$\sigma = 1$	$\sigma = 0.1$	$\sigma = 0.01$
MDCE	0.1829	0.1202	0.0825
MCCE	0.1495	0.1026	0.0547
SQPDSA	0.2864	0.2340	0.1457

4.4 Kermack-McKendrick Problem

In this section, we apply our methods to a delay differential equation (DDE) . Since our methods require no information about the partial derivatives of the differential equations, we only need to apply an existing reliable DDE solver. We use the solver *dde23*. In the literature of parameter estimation in differential equations, there is little to address the problem for DDEs. Here we give a comparison with a recently developed method of parameter estimation of DDEs [38]. The method is deterministic and uses sequential quadratic programming complied with delayed sensitivity analysis of the DDEs (SQPDSA).

We test our methods on a classic Kermack-McKendrick model of an infectious disease with periodic outbreak [10, 9]. The problem is

$$\begin{aligned} y_1' &= -y_1(x)y_2(x-\tau) + y_2(x-\rho), \\ y_2' &= y_1(x)y_2(x-\tau) - y_2(x), \\ y_3' &= y_2(x) - y_2(x-\rho), \end{aligned} \tag{4.4}$$

with constant history function

$$\begin{aligned} y_1' &= a, \\ y_2' &= b, \\ y_3' &= c \end{aligned} \tag{4.5}$$

for $t \leq 0$. We set the delay terms $\tau^* = 1$ and $\rho^* = 10$, for t in $[0, 55]$ and the history function $a = 5, b = 0.1, c = 1$ for $t \leq 0$ to specify the exact reference solution. The unknown parameters are $\Theta = \{\tau, \rho, a, b, c\}$.

The exact solution of this problem is unknown. Fig. (4.4.a) shows the approximate solutions which is periodic. Fig. (4.4.b) shows the fitted curve output by the proposed methods with no noise added. Also, we add different scales of noise and give a comparison with *SQPDSA* [38] (see Table (4.8)).

4.5 Discussions

In this section, we discuss the effect of the parameter tuning in the proposed methods, as well as the choice of objective function in (1.5) .

4.5.1 The effect of N

In this section, we investigate the effect of the number of trial samples N, which has a significant effect on the efficiency of the methods. We conduct an experiment with the Barne's problem. We keep other parameter fixed, and only vary the number of samples N. Then $\Theta = (k_1, k_2, k_3, a, b)$ is estimated. We compare the four methods in Fig. (4.5) . We can see that, as N increases, the accuracy of parameter estimation is improved. This is expected for the initial cross-entropy methods (DCE,CCE) . However, our modified versions can obtain high accuracy, even when N is small. This property, on the other hand, can reduce the computational cost needed to achieve a desired accuracy.

4.5.2 The effect of ρ

We also test the effect of ρ (the parameter used to define the "elites") on both accuracy and efficiency. An similar experiment is conducted: we keep $N = 40$ fixed for MDCE and MCCE, $N = 100$ for DCE and CCE; and we vary ρ from 0.1 to 0.9. The results presented in Fig. (4.6) demonstrate the sensitivity of ρ. It is not surprising that DCE and CCE produce a better accuracy as ρ increases. This is because DCE and CCE rely on the "elite" samples to update the parameters. For MDCE and MCCE, the situation is different, as ρ increases, the accuracy improves for a while and after a certain point, the accuracy does not improve. This is easy to understand. When ρ is very small and increases, the local evolution procedure will put more weight on the "best" samples, and the "bad" samples will enlarge the search space. All of these will increase the accuracy. But MDCE and MCCE degenerate to DCE and CCE, respectively, when ρ is large enough. In addition, we also test the effect of ρ on the CPU time. We can see that ρ can boost the convergence rate of DCE and CCE. However, when ρ is large for MDCE and MCCE, the convergence rate is slow due to the fact that we have to explore more "bad" samples in order to achieve good accuracy.

4.5.3 The effect of choices of objective function

Most methods choose the objective function in (1.5) as the square difference of the observations and fitted data. The main reason is that this objective function is differentiable and easy to analyze. Since we don't need any derivative information, the square form is not necessary for the proposed methods. Now we exploit the effect of different choices for the cost function in (1.4) or (1.6) . The generic p-norm is defined as

$$\|x - y\|_p = (\sum_i (x_i - y_i)^p)^{1/p} \tag{4.6}$$

Table 4.9: The effect of choices of distance functions on Barnes' Problem.

p	0.1	0.2	0.5	1	2	5
MDCE	**0.0110**	0.0154	0.0185	0.0185	0.0196	0.0235
MCCE	**0.0057**	0.0083	0.0102	0.0105	0.0121	0.0157

Table 4.10: Results on Barnes' Problem in the case of partial observation with all the parameters (k_1, k_2, k_3, a, b) unknown.The true parameters are $\{1, 1, 1, 1, 0.3\}$

Observed Variable	MDCE	MCCE
y_1	$\{0.868, 1.550, 1.498, 1.00, 0.258\}$	$\{1.126, 1.148, 1.579, 1.00, 0.287\}$
y_2	$\{0.875, 0.754, 0.056, 1.00, 0.244\}$	$\{1.056, 1.873, 0.226, 1.00, 0.276\}$

and the corresponding objective function should be:

$$\mathcal{L}_p(\mathbf{\Theta}) = \frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{d}(Y_{ij}(\mathbf{\Theta}) - \bar{Y}_{ij})^p)^{1/p}, \tag{4.7}$$

We pick different p to see the effect of distance function. We can see that when p is small, we can achieve better accuracy. This is not difficult to understand that small p can enlarge the difference of small y_i numerically.

4.5.4 The case of partial observation

We also investigate our methods when only partial observation is available. Parameter estimation in the case of partial observation is a challenging problem and few researchers have addressed this problem. Discussion on the identifiability of the parameters in the case of partial observation is beyond the scope of this paper. Here we report the performance on the Barnes' problem, which is known to be well-posed. Table (4.10) shows the estimated values of the complete parameters. With limited observations of a single variable, the proposed methods can predict a good set of parameters. And our methods are very accurate in estimating the initial values.

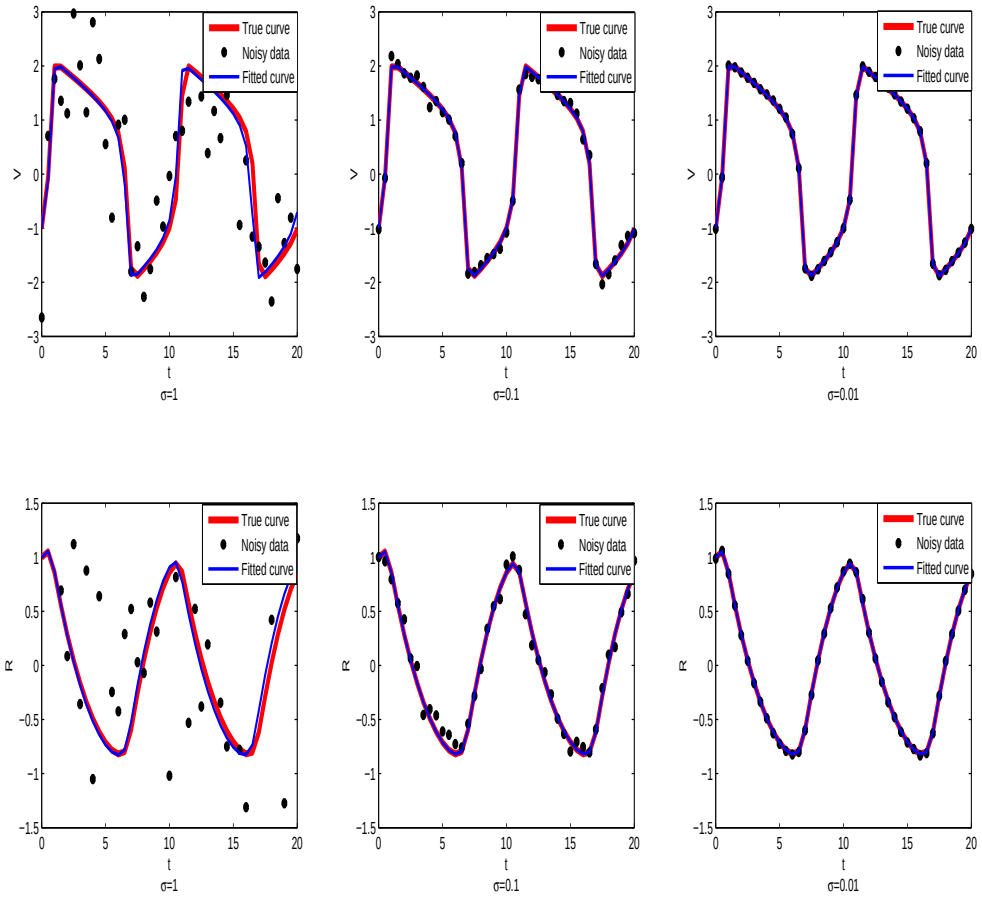

Figure 4.2: The fitted curves of solutions of FitzHugh-Nagumo equations using MDCE with different scales of noise and 50 observation points. The upper half shows the exact solution and "fitted" approximation to $V(t)$ and the lower half shows the exact solution and "fitted" approximation to $R(t)$

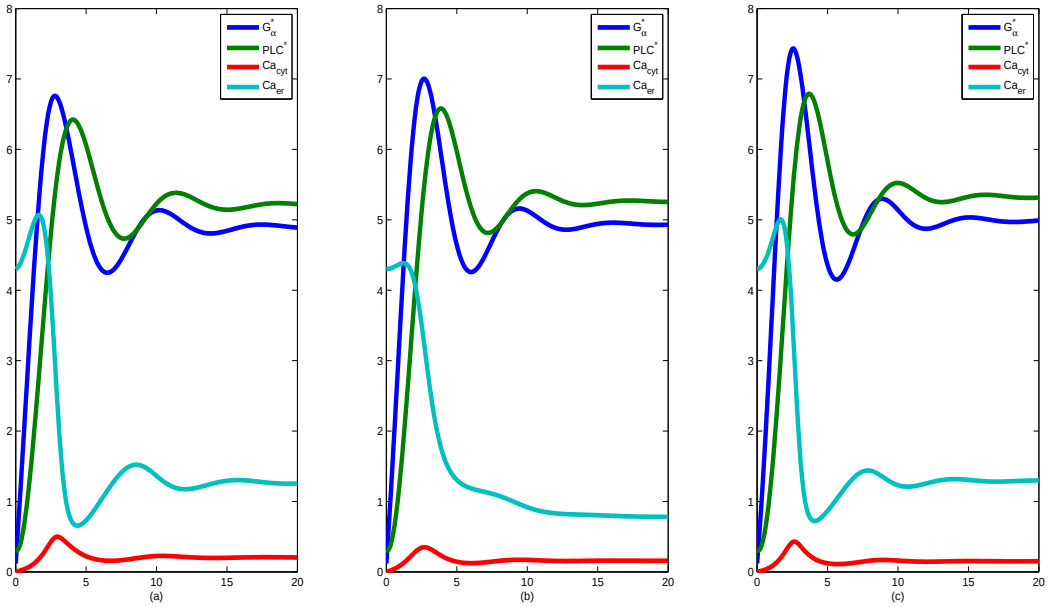

Figure 4.3: Response of Calcium Ions Simulation. (a) shows the true curve of all variables. (b) shows the fitted curve using MDCE. (c) shows the fitted curve using MCCE.

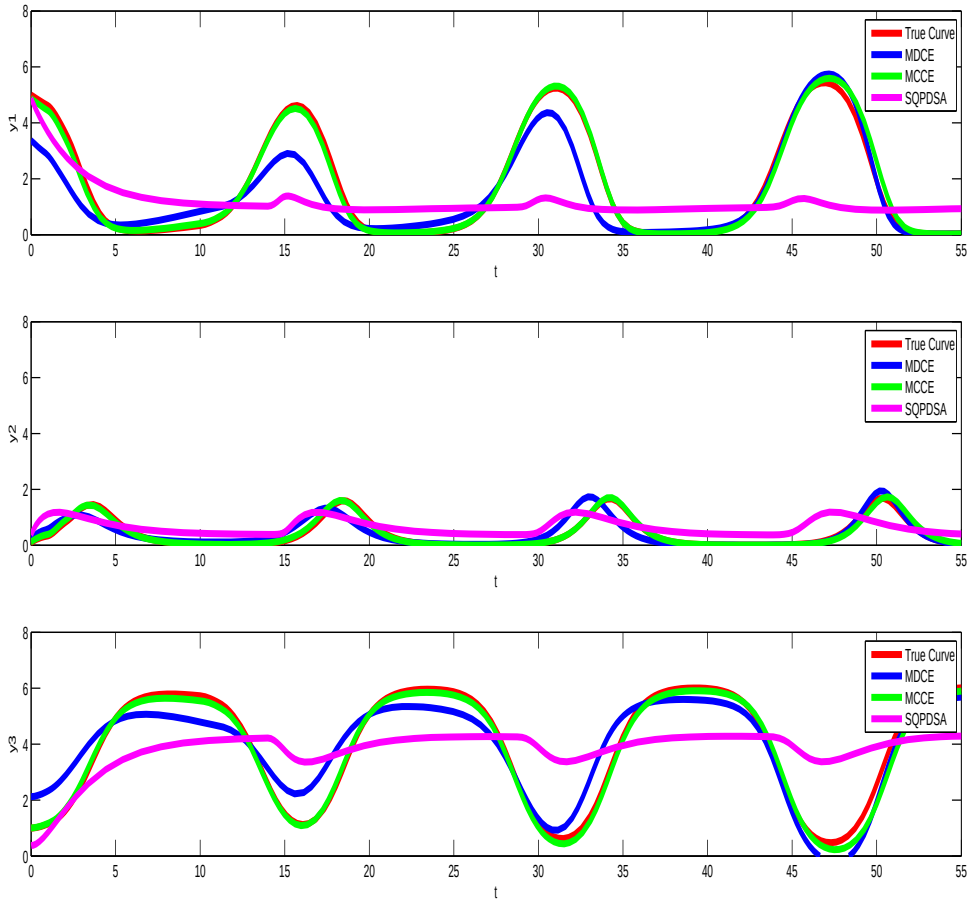

Figure 4.4: Response of y_1, y_2, y_3 in Kermack-McKendrick model for $t \in [0, 55]$ with no noise added. The red curve represents the underlying solution of Kermack-Mckendrick models. 55 observations points were used.

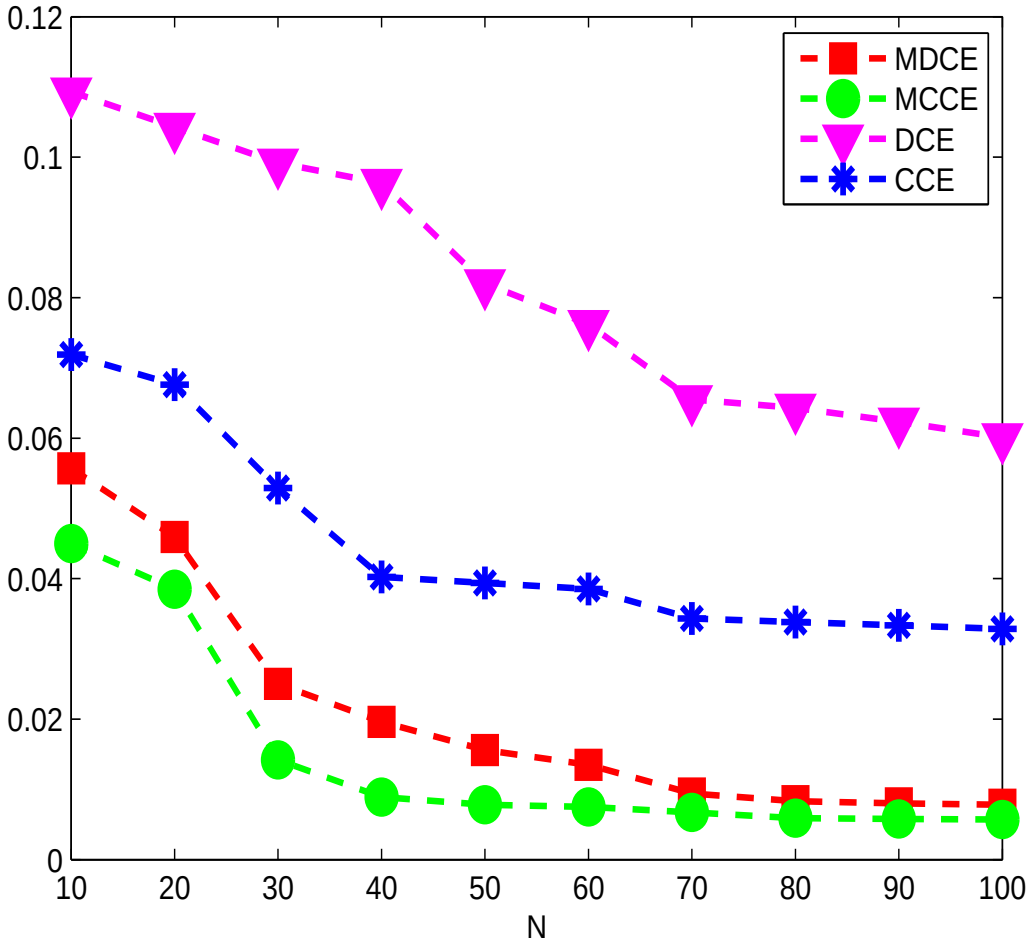

Figure 4.5: The effect of the values of N on J for Barnes' Problem with no noise added.

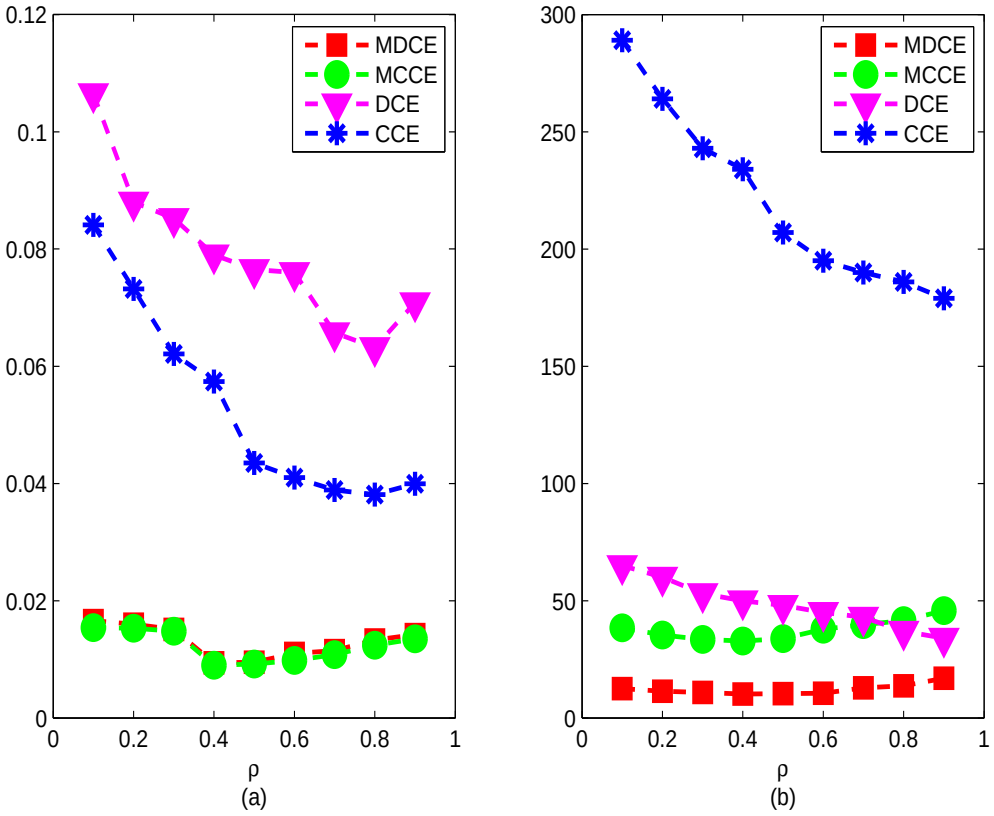

Figure 4.6: (a) The effect of ρ on J for Barnes' Problem with no noise added. (b) The effect of ρ on CPU_T for Barnes' Problem with no noise added.

Chapter 5

Conclusion

In this paper, we present a novel method for parameter estimation problem of ODEs and DDEs. We apply the cross-entropy algorithms in the context of differential equations for the first time, to our best knowledge. To overcome the prohibitive computation of cross-entropy algorithms, we provide a modification combining a local search technique. The modified cross-entropy algorithm can speed up the convergence rate and improve the accuracy simultaneously. We also design two different coding schemes:1) Discrete coding, which is easy to implement, and also fast to converge; 2) Continuous coding, which is more robust and accurate, though it suffers from relatively slow convergence rate. Extensive numerical experiments on real-world problems illustrate the effectiveness of the proposed methods. Our methods are comparative to most existing state-of-the-art approaches on some benchmark problems for parameter estimation. The experiments show three main advantages of the proposed methods: 1) They are robust to noise of observation points; 2) They are not sensitive to the number of observation points, while most existing approaches are; 3) The proposed methods can be generalized to large number of parameters. We also briefly discuss the effectiveness of the proposed methods when partial observations are available. Future work will focus on the extension of the use of cross-entropy methods to stochastic differential equations with parameters.

Bibliography

[1] M.M. Ali, C. Storey, and A. Törn. Application of stochastic global optimization algorithms to practical problems. *J. Optim. Theory Appl.*, 95:545–563, 1997.

[2] C.T.H. Baker and C.A.H. Paul. Pitfalls in parameter estimation for delay differential equations. *SIAM J. Sci. Comput.*, 18:305–314, 1997.

[3] L. Biegler and I. Grossman. Retrospective on optimization. *Comput. Chem. Engng.*, 28:1169–1192, 2004.

[4] H.G. Bock. Recent advances in parameter identification techniques for ODE. In P. Deuflhard and E. Hairer, editors, *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, pages 95–121. Birkhäuser, Boston, 1983.

[5] H.G. Bock and J.P. Schlöder. Recent progress in the development of algorithm and software for large-scale parameter estimation problems in chemical reaction systems. In P. Kotobh, editor, *Automatic Control in Petrol, Petrochemical and Desalination Industries*. IFAC Congress, Pergamon, Oxford, 1986.

[6] P.T. Boggs, R.H. Byrd, and R.B. Schnabel. A stable and efficient algorithm for nonlinear orthogonal distance regression. *SIAM J. Sci. Stat. Comput.*, 8(6):1052–1078, 1987.

[7] Abderrahmane Boubezoul, Sebastien Paris, and Mustapha Ouladsine. Application of the cross entropy method to the glvq algorithm. *Pattern Recognition*, 41:3173–3178, 2008.

[8] Lennart Edsberg and Per-Ake Wedin. Numerical tools for parameter estimation in ode-systems. *Optimization Methods and Software*, 6:193–217, 1995.

[9] E.Hairer, S.P.Norsett, and G.Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problem*. Springer Series in Computational Mathematics., New York, 1993.

[10] W. H. Enright. Software for ordinary and delay differential equations: Accurate discrete approximate solutions are not enough. *Appl. Numer. Math.*, 56:459–471, March 2006.

[11] W.R. Esposito and C. Floudas. Deterministic global optimization in nonlinear optimal control problems. *J. Glob. Optimization*, 17:97–126, 2000.

[12] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley, New York, USA, 1966.

[13] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, 1 edition, 1989.

[14] I.E. Grossmann. *Global Optimization in engineering design*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[15] W. Horbelt, J. Timmer, Bünner, Meucci R., and Ciofini M. Identifying physical properties of a co2 laser by dynamical modeling of measured time series. *Physical Review E*, 64, 2001.

[16] W. Horbelt, J. Timmer, and Voss H.U. Parameter estimation in nonlinear delayed feedback systems from noisy data. *Physical Letters A*, 299:513–521, 2002.

[17] R. Horst and H. Tuy. *Global Optimization: Deterministic approaches*. Springer-Verlag, Berlin, 1990.

[18] W. Huyer and A. Neumaier. A global optimization by multilevel coordinate search. *J. Global Optim.*, 14:331–355, 1999.

[19] Murphy K.A. Estimation of time- and state-dependent delays and other parameters in functional differential equations. *SIAM J. Scient. Comput.*, 50:972–1000, 1990.

[20] D. Kroese, S. Porotsky, and R. Bubinstein. The cross-entropy method for continuous multi-extremal optimization. *Methodol. Comput. Appl. Probab.*, 8:383–407, 2006.

[21] Ursula Kummer, Lars F. Olsen, C. Jane Dixon, Anne K. Green, E. Bornberg Bauer, and Gerold Baier. Switching from simple to complex oscillations in calcium signaling. *Biophysical Journal*, 79(3):1188–1195, 2000.

[22] J. Matyas. Random optimization. *Automat. Remote Control*, 26:246–253, 1965.

[23] Carmen G. Moles, Pedro Mendes, and Julio R. Banga. Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome research*, 13:2467–2474, 2003.

[24] J.S. Nagumo, S. Arimoto, and S. Yoshizawa. An active pulse transmission line simulating a nerve axon, 1961.

[25] M Peifer and J Timmer. Parameter estimation in ordinary differential equations for biochemical processes using the method of multiple shooting. *IET System Biology*, 1:78–88, 2007.

[26] FitzHugh R. Impulses and physiological states in models nerve membrane. *Biophys. J.*, 1:445–466, 1961.

[27] J. O. Ramsay, G. Hooker, D. Campbell, and J. Cao. Parameter estimation for differential equations: a generalized smoothing approach. *Journal Of The Royal Statistical Society Series B*, 69(5):741–796, 2007.

[28] R.Y.Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer-Verlag, New York, 2004.

[29] R.Y.Rubinstein and A. Shapiro. *Discrete Event Systems:Sensitivity Analysis and Stochastic Optimization via the score function method*. Wiley Series in Probability and Statistics, 1993.

[30] H.P. Schwefel. *Evolution and optimum seeking*. Wiley, New York, 1995.

[31] Masahiro Sugimoto, Shinichi Kikuchi, and Masaru Tomita. Reverse engineering of biochemical equations from time-course data by means of genetic programming. *BioSystems*, 80:155–164, 2005.

[32] I. Tjoa and L. Biegler. Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *Industrial Engineering Chemistry Research*, 30:376–385, 1991.

[33] I.B. Tjoa and L.T. Biegler. Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *Ind. Engng Chem. Res.*, 30:376–385, 1991.

[34] A. Törn. Global optimization as a combination of global and local search. In *Proceedings of computer simulation versus analytical solutions for business and economic models*, pages 191–206, 1973.

[35] A. Törn, M. Ali, and S. Viitanen. Stocastic global optimization: problems classes and solution techniques. *J. Glob. Optim.*, 14:437, 1999.

[36] J.M. Varah. A spline least squares method for numerical parameter estimation in differential equations. *SIAM, Journal of Scientific and Statistic Computation*, 3:28–46, 1982.

[37] Michael R.Osborrne ZhengFeng Li and Tania Prvan. Parameter estimation of ordinary differential equations. *IMA, Journal of Numerical Analysis*, 25:264–285, 2005.

[38] H. Zivaripiran. *Efficient Simulation, Accurate Sensitivity Analysis and Reliable Parameter Estimation for Delay Differential Equations*. PhD thesis, Computer Science Department, Univeristy of Toronto, 2009.

[39] Jason W. Zwolak, John J. Tyson, and Layne T. Watson. Parameter estimation for a mathematical model of the cell cycle in frog eggs. *J. of Comput. Bio.*, 12(1):48–63, 2005.