APPLICATION OF ODSA TO POPULATION CALCULATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NAURAL AND APPLIED SCINECES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MUSTAFA ULUKAYA

IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN ELECTRICAL AND ELECTRONIC ENGINEERING

APRIL, 2006

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan ÖZGEN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science

Prof. Dr. İsmet ERKMEN
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Kerim DEMİRBAŞ
Supervisor

Examining Committee Members:

Prof.Dr. Kemal LEBLEBİCİOĞLU (METU, EE)

Prof.Dr. Kerim DEMİRBAŞ (METU, EE)

Yard. Doç. Dr. Çağatay CANDAN (METU, EE)

Doç.Dr. Fahrettin ASLAN (ANKARA UNIV, STAT)

Dr. Ümit BİNGÖL (ANKARA UNIV, BIO)

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Mustafa ULUKAYA

Signature :

ABSTRACT

APPLICATION OF ODSA TO POPULATION CALCULATION

Ulukaya, Mustafa

M.S., Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Kerim Demirbaş

April 2006, 102 pages

In this thesis, Optimum Decoding-based Smoothing Algorithm (ODSA) is applied to well-known Discrete Lotka-Volterra Model. The performance of the algorithm is investigated for various parameters by simulations. Moreover, ODSA is compared with the SIR Particle Filter Algorithm. The advantages and disadvantages of the both algorithms are presented.

Keywords: Discrete Lotka-Volterra, Trellis Diagram, ODSA, SIR Particle Filter

ÖZ

POPULASYON HESAPLAMALARINA ODSA UYGULANMASI

Ulukaya, Mustafa

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Kerim Demirbaş

Nisan 2006, 102 sayfa

Bu tezde Optimum Kodlamaya Dayalı Yumuşatma Algoritması bilinen Sayısal Lotka-Volterra modeline uyarlandı. Algoritmanın performansı çeşitli değişkenlere göre benzetimler gerçekleştirilerek incelendi. Ayrıca ODSA SIR Parçacık Filtresi Algoritması ile karşılaştırıldı. İki Algoritmanın da avantajlı ve dezavantajlı yönleri gösterildi.

Anahtar Kelimeler: Sayısal Lotka-Volterra, Kafes Diyagramı, ODSA, SIR Parçacık Filtresi

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my supervisor Prof. Dr. Kerim DEMIRBAŞ for his guidance, advice, criticism, encouragements, and patience throughout the research.

I am very grateful to my family for their assistance and encouragements.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Interaction between species has been studied by scientists for several centuries. There are numerous scientists spent their lives on studying the interaction of the species.

Competition and mutualism are two important interactions among species. According to definitions by Krebs [9], competition occurs when two species use the same resources or harm each other when sharing resource; mutualism is defined as living of two species in close association with one another with the benefit of both. Competition also occurs among individuals or groups within species, often more furiously because they use the very similar resources. Within species, mutualism is called cooperation [10], which is commonly seen in social animals and in human society.

The famous competition model was proposed independently by Lotka (1925) [2] and Volterra (1926) [2] in Italy. In this model, coexistence occurs only when the crowd-tolerability and competitive capacity of species are well balanced. Otherwise, low crowd-tolerable and low competitive species (inferior competitor) will be removed by the superior competitor. Many competitive models or community models were further developed by slightly modifying the Lotka-Volterra (LV) model, often by introducing non-linear isoclines, but without changing the monotonous and negative relationship between the growth rate of the focal species (e.g. Paine, 1966[11]; May, 1973 [12]; Renshaw, 1991 [13]; Zhang and Hanski, 1998 [14]).

Leslie [21] gave the general Lotka-Volterra Model that investigates the interaction between species. The species are classified as competitors, predator-prey and symbiosis via the model parameters. Since these relations occur not only between species in biology, but also in economy, political sciences, food engineering, etc. as well. Actually, Lotka Volterra Model is used in various fields where competition is involved. First of all, in biology, Lotka Volterra Model is applied to investigate the relationship between the species of animals by several scientists [22, 23, 24, 25] in addition the relationship between plants is also investigated by several scientists [26, 27, 28] using Lotka Volterra Model. On the other hand, S.J Lee and Y.Louzoun applied LV model to economical systems, which investigate the stock market analysis [15] and market volatility [16], respectively. Moreover, Lotka Volterra Model is used in Political Sciences; Francisco modeled the interaction between coercion and protest using Lotka Volterra Model [17]. In addition, Lotka Volterra Model is used in Chemistry [18], Food Engineering [19], Computer Sciences [20], etc. Briefly, as soon as interaction between two phenomena that have competition, predator-prey or symbiosis relationship; Lotka Volterra Model is a powerful tool to analyze the interaction.

In some applications, measurements are taken in discrete time. In order to use discrete time data, it is necessary to convert the Lotka-Volterra equation into discrete time version. Dubious [7] and Murray [8] give the discrete time version of the Lotka Volterra Model. Then, Discrete Lotka Volterra Model can be used for finding the population of the species in an ecosystem. If the parameters of the system that exposes the interaction of the species, and the previous number of the species are known, by using Discrete Lotka Volterra Model, the present value of the population of the species can be predicted.

At this point, the problem of predicting the new species population can be regarded as a discrete-time state estimation problem. Let the populations of two species size is regarded as state variables, since the previous values and the system parameters are known, the future state values can be predicted. To solve the state estimation problem Optimum Decoding-based Smoothing Algorithm (ODSA) [1] is a powerful algorithm that gives suboptimum prediction. ODSA obtains a Trellis

Diagram for the state values and estimates the new values for the state values of the using Viterbi Algorithm [3, 4].

ODSA can be used for both linear and nonlinear models. In addition ODSA is capable to estimate both new states and the new parameters of the system whenever system parameters either unknown or changing.

In this thesis, ODSA is applied to Stochastic Discrete Lotka-Volterra Model to estimate the population of two species. The algorithm is implemented in MATLAB® environment and some simulations are performed in order to evaluate the state estimation performance. Also ODSA is compared with SIR Particle Filter Algorithm and advantageous and disadvantageous parts of the both algorithm are presented. In addition, Lotka Volterra Model is compared with the Lanchester War Model and similarities and differences between the models are explained.

In Chapter 2, the well-known Lotka-Volterra model is presented; the Discrete Lotka-Volterra Model and Stochastic Discrete Lotka-Volterra Model are given.

In Chapter 3, Lanchester War model is given and the comparison between Lanchester War Model and Lotka Volterra Model is made. Similarities and differences between the two models are presented.

In Chapter 4 ODSA Algorithm is explained in detail. Gazioğlu [5] used ODSA in one-dimensional problems; since in Discrete Lotka Volterra Model there are populations of two species, the two-dimensional ODSA is discussed. In addition, the complexity analysis of ODSA is done.

In Chapter 5 Particle Filter Algorithm is introduced. For Lotka-Volterra model, SIR type Particle Filter is used. In addition, sample run of Particle Filter algorithm is given for Stochastic Discrete Lotka Volterra Model.

In Chapter 6 simulation results of ODSA are presented and the effects noise variances, initial state variance and ODSA parameters are discussed.

In Chapter 7, simulations of Particle Filter algorithm are done for various N_s values and resultant error performance is plotted. Moreover, the comparison of

ODSA and Particle Filter Algorithm is done in this chapter. The advantageous and disadvantageous parts of the both algorithm is presented.

In Chapter 8 Comments on the simulation results of the ODSA algorithm on Lotka-Volterra model is given. The advantages and the disadvantages of the algorithm are discussed by considering the simulation results.

In Appendix A, the possible values and the corresponding probabilities of the discrete random variable which approximates the Gaussian distributed continuous random variables up to 50 possible values are given. These values are used by the ODSA algorithm while obtaining the trellis diagram for the target motion model.

CHAPTER 2

2.1 LOTKA VOLTERRA MODEL

The interaction between two species can be expressed in general terms via the Lotka Volterra Model. When two species, *X* and *Y*, interacting in the same environment, we may write as follows [6]:

$$\frac{dX}{dt} = (a_1 - c_1 Y)X = a_1 X - c_1 XY \tag{2.1}$$

$$\frac{dY}{dt} = (a_2 - c_2 X)Y = a_2 Y - c_2 YX \tag{2.2}$$

Where X is the population size of the species X and Y is the population size of the species Y. This system of equations contains all fundamental parameters that impact the rate of growth of both species. Namely, a_i is the logistic parameter for the species i when living alone, c_i is the interaction parameter with the other species. A simulation of Lotka-Volterra Model is given in Figure 1

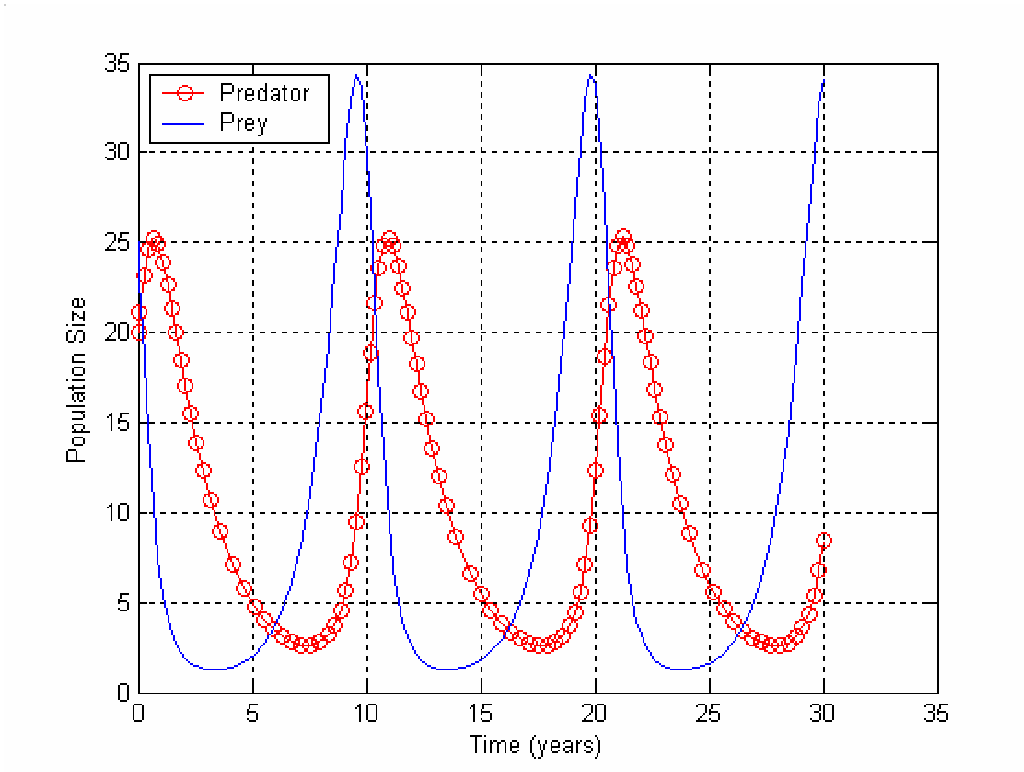

Figure 1. *Lotka-Volterra Predator Prey Population Model*

The parameter a_i gives the birth and/or death rate of the species when it is living alone. Therefore a_i can be modeled as

$$a_1=1+br; \quad (2.3)$$

$$a_2=1-dr; \quad (2.4)$$

where br gives you the birth rate of the species and dr gives the death rate of the species, respectively.

The parameter c_i gives the interaction between the species. Especially, by the sign of c_1 and c_2, we can determine the type of the competitive roles as shown in Table 1[15].

Table 1.*The type of competitive roles according to the signs of c*1 *and c2*

c_1	c_2	Type	Explanation
+	+	Pure Competition	Occurs when both species suffers from each other's existence
+	–	Predator – Prey	Occurs when one of them serves direct food to the other
–	–	Mutualism	Occurs in case of symbiosis

2.2 DISCRETE LOTKA VOLTERRA MODEL

This thesis considers discrete time Lotka-Volterra Model which is given by Dubious [7] and Murray [8] as follows:

$$X[k+1]=X[k]+brX[k]-a_1X[k]Y[k] \quad (2.5)$$

$$Y[k+1]=Y[k]-drY[k]-a_2Y[k]X[k] \quad (2.6)$$

where,

X[k]: The number of first species in the ecosystem

Y[k]: The number of second species in the ecosystem

br: The annual birth rate of the first species, including partial death rate

dr: The annual death rate of the second species, including partial death rate

a_1: The interaction parameter between first species and second species

a_2: The interaction parameter between second species and first species

2.2.1 Stochastic Discrete Lotka-Volterra Model

Since the life is not perfect and the nature is not deterministic, some of the terms cannot be fully explained by the deterministic model. Always there are noises that affect the system. In Lotka Volterra model, we assumed that there are only two species interacting between each other, which is not the usual case. Moreover, there are several effects that have role in the population of the species in the ecosystem. Although these effects have not too much deviation in the population of the species in the ecosystem, they should be taken into account. These effects are modeled as Gaussian disturbance noises w_1 and w_2 where w_1 is normal distributed with zero mean and var_{w1} variance and w_2 is normal distributed with zero mean and var_{w2} variance. In addition for the observations X[k] and Y[k], there are always observation errors for specie populations. These errors are modeled as observation noise, namely, v_1 is counting errors for species 1, i.e., X; v_2 is counting errors for species 2, i.e., Y. The observation noise is also modeled as Gaussian distributed white noise.

When these noises appended to the system, the state equations (2.5) and (2.6) of the model become:

$$X[k+1]=X[k]+brX[k]-a_1X[k]Y[k]+w_1[k] \quad (2.7)$$

$$Y[k+1]=Y[k]-drY[k]-a_2Y[k]X[k]+w_2[k] \quad (2.8)$$

The observation equation is,

$$Z[k]=\begin{bmatrix} X[k] \\ Y[k] \end{bmatrix}+\begin{bmatrix} v_1 k] \\ v_2[k] \end{bmatrix} \quad (2.9)$$

where

$$v_1 \sim N(0,\ \sigma_{v_1}^2) \text{ and} \quad (2.10)$$

$$v_2 \sim N(0,\ \sigma_{v_2}^{\ 2}) \tag{2.11}$$

That is, v_1 is a Gaussian distributed random noise with zero mean and $\sigma_{v_1}^{\ 2}$ variance, and v_2 is also a Gaussian distributed random noise with zero mean and $\sigma_{v_2}^{\ 2}$ variance.

Moreover, since the birth and death rate of the species are not constant and also they deviate from year to year due to several effects e.g. migration, unexpected diseases etc., a noise term should also be added on these parameters. Let us define the br and dr as follows:

$$br \sim N(\mu_{br},\ \sigma_{br}^{2}) \tag{2.12}$$

$$dr \sim N(\mu_{dr},\ \sigma_{dr}^{2}) \tag{2.13}$$

where μ_{br} is the mean of the birth rate of the first species, σ_{br}^{2} is the variance of the birth rate of the first species, μ_{dr} is the mean if the death rate of the second species and σ_{dr}^{2} is the variance of the death rate of the second species. A simulation of Stochastic Discrete Lotka Volterra Model is given in Figure 2. For this simulation, the following model parameters are chosen:

- Mean of initial state X0 ($mean_{X0}$) : 20
- Mean of initial state Y0 ($mean_{Y0}$) : 25
- Variance of initial state X0 (var_{X0}) : 4
- Variance of initial state Y0 (var_{Y0}) : 9
- Mean of birth rate of X ($mean_{br}$) : 0.7
- Mean of death rate of X ($mean_{dr}$) : 0.5
- Variance birth rate of X (var_{br}) : 0.1
- Variance death rate of X (var_{dr}) :0.1
- The interaction parameter between X and Y (a_1) : 0.007
- The interaction parameter between Y and X (a_2) : 0.006
- Variance of disturbance noise w_1 (var_{w1}) : 4
- Variance of disturbance noise w_2 (var_{w2}) : 4

- Variance of observation noise v_1 (var_{v1}) : 1
- Variance of observation noise v_2 (var_{v2}) : 4

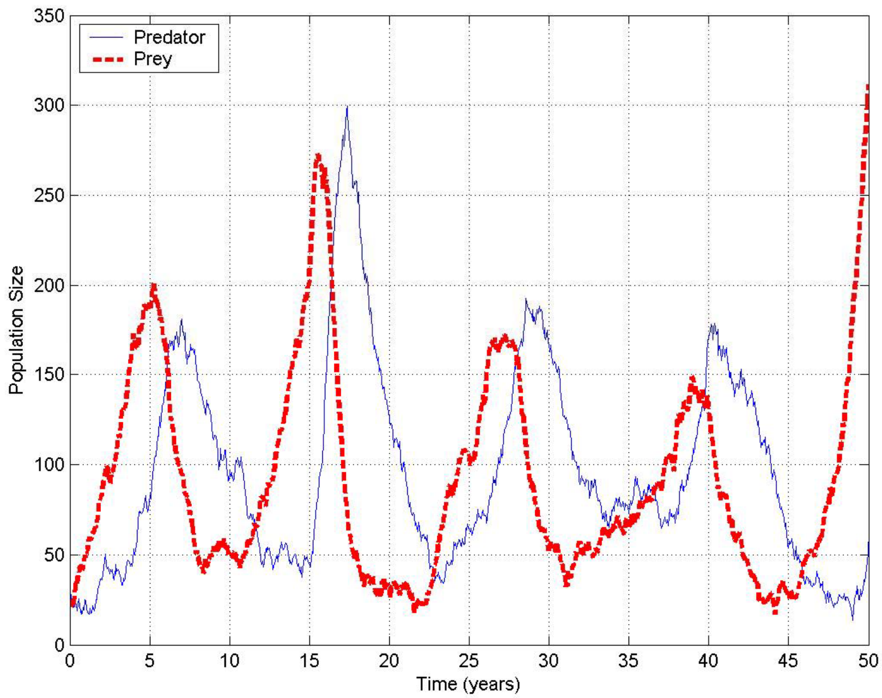

Figure 2. *Stochastic Discrete Lotka-Volterra Population Model*

CHAPTER 3

LANCHESTER WAR MODELLING

3.1 Lanchester War Modeling

Lanchester War modeling is first introduced by British Scientist F. W. Lanchester (1868 - 1946). German scientist, Bernard Koopman (1900-1981) developed and modified the Lanchester's law into Lanchester's strategy model, and widely used in World War II.

Lanchester War model consists of two general differential equations, including both conventional and guerilla combat terms. Özdemir [34] derived this general model, seen in equation (3.1).

$$\begin{aligned} \frac{dx(t)}{dt} &= -ax(t) - \alpha_x v_{cx} r_y p_y y(t) - (1-\alpha_x) v_{gx} r_y p_y x(t) y(t) + RR_x(t) \\ \frac{dy(t)}{dt} &= -dy(t) - \alpha_y v_{cy} r_x p_x x(t) - (1-\alpha_y) v_{gy} r_x p_x x(t) y(t) + RR_y(t) \end{aligned} \tag{3.1}$$

where

Operational Loss Rates:

- *-ax(t)*: operational losses of General X,
- *-dy(t)*: operational losses of General Y,

Combat Loss Rates:

- $-\alpha_x v_{cx} r_y p_y y(t)$: General X is combating conventionally and incurring a loss where
  - α_x is the ratio of General X conventional combat forces over whole forces of General X, ($0 \le \alpha_x \le 1$),
  - v_{cx} is the visibility coefficient of a General X's soldier, combating conventionally.
  - r_y is the firing rate, rate of ammunition that is utilized by the army (shots/combatant/day),
  - p_y is the probability that a single shot kills an opponent.
  - $y(t)$ is the number of General Y force at time t.
- $-(1-\alpha_x) v_{gx} r_y p_y x(t) y(t)$: General X is guerilla combating and incurring a loss where
  - $1-\alpha_x$ is the ratio of General X forces guerilla combating over whole forces of General X, ($0 \le (1-\alpha_x) \le 1$),
  - v_{gx} is the visibility coefficient of a General X's soldier, guerilla combating. This parameter is a dynamic one depending on the total number of guerillas in the area and the predefined maximum amount of guerillas that the area can hold.
  - r_y is the firing rate, rate of ammunition that is utilized by the army (shots/combatant/day),
  - p_y is the probability that a single shot kills an opponent.
  - $x(t)$ is the number of General X force at time t.

Reinforcement Rates:

- $RR_x(t)$ is the reinforcement rate in numbers of combatants per day. It is controlled by General X.

As it can be seen in the equation (3.1), Lanchester War Model is a deterministic, differential model. Without loss of generality, the model is very similar to the Lotka-Volterra Model in Chapter 2. To see this fact clearly, the Lanchester War model can be written as:

$$\frac{dx(t)}{dt} = a_1 x(t) + \beta_1 y(t) + \gamma_1 x(t) y(t) + f_x(t)$$
$$\frac{dy(t)}{dt} = a_2 x(t) + \beta_2 y(t) + \gamma_2 x(t) y(t) + f_y(t) \quad (3.2)$$

On the other hand, LV model can be written as:

$$\frac{dX(t)}{dt} = X(t) + brX(t) - a_1 X(t) Y(t) + w_1(t)$$
$$\frac{dY(t)}{dt} = Y(t) - drY(t) - a_2 Y(t) X(t) + w_2(t) \quad (3.3)$$

As it can be seen from the equations (3.2) and (3.3), the equations are very similar in type. However, we derived Stochastic Lotka Volterra Model to explain the noisy variations as well as use ODSA and Particle Filter, the Lanchester War model is again deterministic and to use these models, stochastic version should be derived to use ODSA and Particle Filter Algorithm. In the following table (Table.2), the Lanchester War Model and Stochastic Lotka Volterra Model are compared. To make the comparison more clear, without changing the structure, let us change the name of the parameters of Lotka Volterra Model (3.4).

$$\frac{dX(t)}{dt} = \alpha_1 X(t) + \gamma_1 X(t) Y(t) + w_1(t)$$
$$\frac{dY(t)}{dt} = \alpha_2 Y(t) + \gamma_2 Y(t) X(t) + w_2(t) \quad (3\ 4)$$

Table 2.*Similarities and Differences Between Lotka Volterra Model and Lanchester War Model*

	Lotka Volterra Model	Lanchester War Model
1. α_1 and α_2	Varying and noisy	Constant
2. β_1 and β_2	Not exist	Varying as a function of both x and y
3. γ_1 and γ_2	Constant	Varying as a function of both x and y
4. $f_x(t)$ and $f_y(t)$	Gaussian Disturbance Noise	Varying with respect to x and y, respectively
5. Observation	There exist observation equation	There is no observation equation

CHAPTER 4

OPTIMUM DECODING BASED SMOOTHING ALGORITHM

Optimum Decoding-Based Smoothing Algorithm [1] is used for the target tracking problems have the following form

$$\textit{Motion model,} \quad x(k+1) = f(k, x(k), u(k), w(k)), \tag{4.1}$$

$$\textit{Observation model,} \quad z(k) = g(k, x(k), v(k)),$$

where $x(0)$ is an $nx1$ initial state random vector which determines the considered target location at time 0. $x(k)$ is an $nx1$ state vector at time k which determines the considered target location at time k. $u(k)$ is a $qx1$ input vector at time k. $w(k)$ is a $px1$ disturbance noise vector at time k with zero mean and known statistics. In the observation model, $v(k)$ is an $lx1$ observation noise vector at time k with zero mean and known statistics. $z(k)$ is an $rx1$ observation vector at time k. Time k is time t_0+kT_0 where t_0 and T_0 are the initial time and the observation interval, respectively. Furthermore, the random vectors $x(0)$, $w(j)$, $w(k)$, $v(l)$ *and* $v(m)$ are assumed to be statistically independent for all j, k, l, m. The aim is to estimate the state sequence $\{x(0), x(1), x(2), \ldots x(L)\}$ by using the observation sequence $\{z(1), z(2), z(3), \ldots z(L)\}$ where L is a chosen integer. The estimation algorithm presented is capable not only the target tracking algorithms but also the whole estimation problems that can be modeled in equation (4.1).

4.1 Quantization of the States and Transition Probabilities

First of all, we are going to make some definitions before proceeding further. Let us consider the state $x(k)$, it is a random vector whose range is in the space R^n. Let us divide R^n into non-overlapping subspaces R_i^n and assign each subspace, R_i^n , a unique value x_{qi}. Note that subscript q refers to quantization.

Definition 4.1: A function $x_q(.) \triangleq Q\{x(.)\}$ is a quantizer for the state $x(.)$ if the following hold:

1) A function $x_q(.) \triangleq Q\{x(.)\} = x_{qi}$ whenever $x(.) \in R_i^n$; and

2) x_{qi} is unique for each R_i^n

Definition 4.2: The function $x_q(.)$ is the quantized state vector at time $(.)$, and its possible values are called quantization levels of the state $x(.)$.

Definition 4.3: Subspace R_i^n is called gate (or sometimes called cell) R_i^n.

Definition 4.4: The value x_{qi} is called the quantization level for the gate R_i^n.

Quantization means that whenever a random state vector falls into a gate of R_i^n the state is quantized to the value x_{qi}.

Definition 4.5: The transition probability $\pi_{jm}(k)$ is the probability that the state $x(k+1)$ will lie in the gate R_m^n when the state $x(k)$ is in the gate R_j^n; i.e.,

$$\pi_{jm}(k) \triangleq Prob\{ x(k+1) \in R_m^n \mid x(k) \in R_j^n \} \qquad (4.2)$$

The transition probability $\pi_{jm}(k)$ is a conditional probability. Hence it can be written as :

$$\pi_{jm}(k) = \frac{\Pr ob\{x(k+1) \in R_m^n, x(k) \in R_j^n\}}{\Pr ob\{x(k) \in R_j^n\}}$$

$$= \frac{\iint_{R_j^n R_m^n} p(x(k+1), x(k)dx(k+1)dx(k))}{\int_{R_j^n} p(x(k))dx(k)}$$

$$= \frac{\left\{ \int_{R_j^n} \left[\int_{R_m^n} p(x(k+1) \mid x(k))dx(k+1) \right] p(x(k))dx(k) \right\}}{\int_{R_j^n} p(x(k))dx(k)}$$

It is not usually easy to evaluate transition probability $\pi_{jm}(k)$ analytically. Evaluation of the integral above is not usually possible even if the system is linear. If the system is nonlinear, difficulty of evaluation increases. Therefore, numerical evaluation will be needed for $\pi_{jm}(k)$. Even this may be difficult, therefore it will be discussed that an approximate target motion model by approximating the disturbance noise vector *w(k)* and the initial state vector *x(0)* by discrete random vectors. In the following section this approximation of the continuous random vectors by discrete random vectors will be explained.

4.2 Approximation of an Absolutely Continuous Random Vector by a Discrete Random Vector

We want to approximate an absolutely continuous random vector by a discrete random vector. Let $F_x(.)$ be the distribution function of an absolutely continuous random vector *x*. And let $F_{y_0}(.)$ be the approximation of $F_x(.)$. To make $F_{y_0}(.)$ the best approximation of $F_x(.)$ the following objective function *J(.)* should be minimized by $F_{y_0}(.)$.

$$\begin{aligned} J(F_{y_0}(.)) &= \min_{F_y(.)} J(F_y(.)) \\ &= \min_{g(.)} J(g(.)) \end{aligned} \tag{4.3}$$

where

$$J(F_y(.)) = \int_{-\infty}^{\infty} \left[F_x(a) - F_y(a)\right]^2 da \tag{4.4}$$

and *g(.)* is a step function that minimizes *J(g(.))*.

$$\begin{aligned} J(g(.)) = &\int_{-\infty}^{y_1} F_x^{\ 2}(a) da + \int_{y_1}^{y_2} \left[F_x(a) - P_1\right]^2 da + \int_{y_2}^{y_3} \left[F_x(a) - P_2\right]^2 da + \ldots \\ &+ \int_{y_{n-1}}^{y_n} \left[F_x(a) - P_{n-1}\right]^2 da + \int_{y_n}^{\infty} \left[F_x(a) - 1\right]^2 da \end{aligned} \tag{4.5}$$

$$g_0(x) = \begin{cases} 0, & x < y_{1,0}, \\ P_{i,0}, & y_{i,0} \le x < y_{i+1,0}, \quad i = 1,2,...,n-1 \\ 1, & x \ge y_{n,0}, \end{cases} \tag{4.6}$$

If $g_0(x)$ is a step function which minimizes (4.5), it must satisfy the following set of equations:

$$\begin{aligned} &P_{1,0} = 2F_x(y_{1,0}); \\ &P_{i,0} + P_{i+1,0} = 2F_x(y_{i+1,0}), \qquad i = 1,2,...,n-2; \\ &1 + P_{n,0} = 2F_x(y_n); \\ &P_{i,0}(y_{i+1,0} - y_{i,0}) = \int_{y_{i,0}}^{y_{i+1,0}} F_x(a)da \qquad i = 1,2,...,n-1 \end{aligned} \tag{4.7}$$

If the mean (μ) and the variance (σ) of the random variable are different than 0 and 1 respectively, it maps the new discrete values according to the mean and variance of the random variable by using the formula given in (4.9).

$$y' = \sigma y_{i,0} + \mu, \qquad P'_{i,0} = P_{i,0} \qquad i = 1,2,...,n \tag{4.9}$$

4.3 Finite State Observation Model

The gates are assumed to be generalized rectangles such that the zero vector 0 (origin) is located in the center of a generalized rectangle, say R_0^n (see Figure 3). Let the lengths of the sides of a generalized rectangle say R_i^n, be g_{i1}, g_{i2}, ... g_{in}. These lengths are the sizes of the gate R_i^n. On the other hand, for the sake of simplicity, we are going to choose the length of the gates equal. In addition, the quantization levels for gates are assumed to be the center points of the gates.

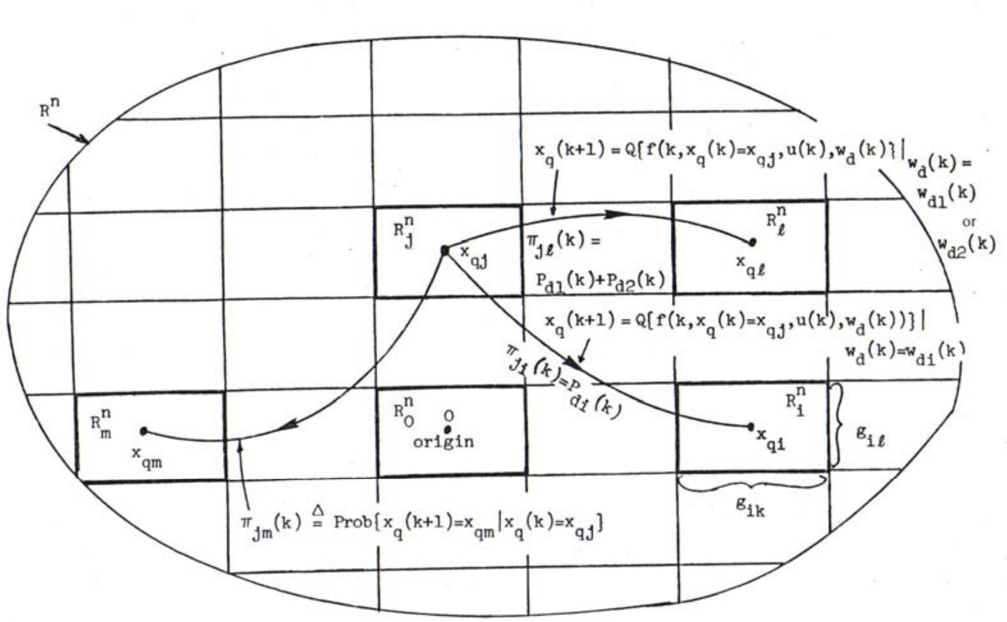

Figure 3. *Quantization with Generalized Rectangles*

The flowchart of the finite-state model is in Figure 4. For each *k*, the disturbance noise vector *w(k)* is approximated by discrete random vector $w_d(k)$ whose possible values are $w_{d1}(k)$, $w_{d2}(k)$, ... , $w_{dm_k}(k)$. The corresponding probabilities are $p_{d1}(k)$, $p_{d2}(k)$, ... , $p_{dm_k}(k)$, where $i=1,2, \ldots, m_k$. Also the initial stat vector *x(0)* is approximated in the same manner with w(k) by n_0 possible values. m_k and n_0 is chosen so that *w(k)* and *x(0)* are satisfactorily approximated by discrete random vectors $x_d(0)$ and $w_d(k)$ for the considered estimation problem, respectively.

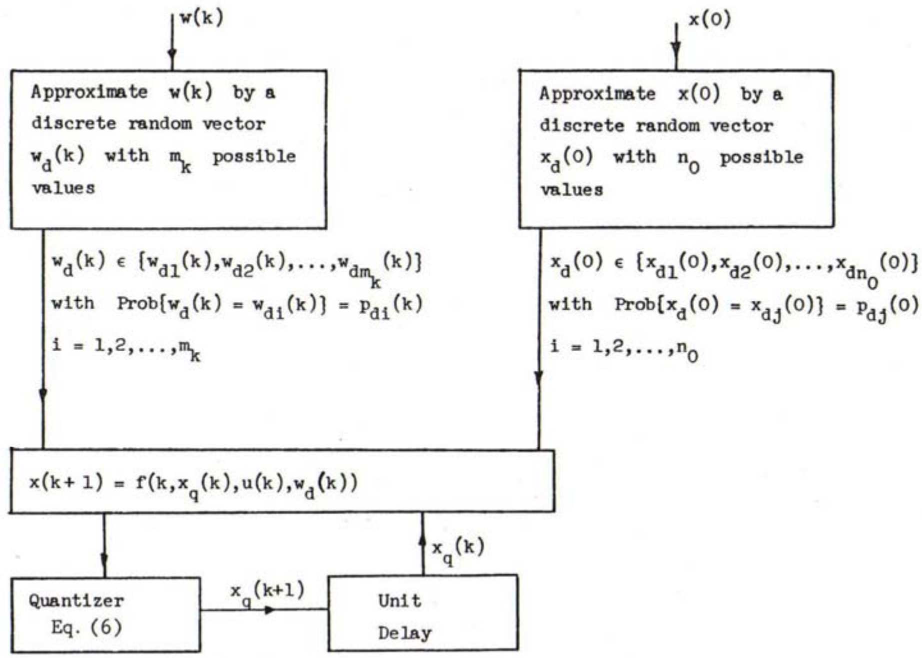

Figure 4. *Flowchart of the Finite State Model*

After $x(0)$ and $w(k)$ is approximated by $x_d(0)$ and $w_d(k)$, respectively, the model can be written as:

$$x_q(k+1) = Q\{f(k, x_q(k), u(k), w_d(k))\} \qquad (4.8)$$

In equation (4.8) the variables and the functions are explained as follows:

- $Q\{.\}$ is the quantizer function
- $x_q(k)$ is quantized state vector at time k and its possible values are $\{ x_{q1}(k), x_{q2}(k), \ldots, x_{qn_k}(k). \}$ where n_k is the number of possible quantization levels of the state vector $x(k)$
- $x_q(0) \triangleq x_d(0)$ [by definition, $x_{qi}(0) \triangleq x_{di}(0), i=1, 2, \ldots n_0$]
- $u(k)$ is the input vector to the system.
- $w_d(k)$ is the approximated discrete disturbance noise random vector of the system.

The transition probability $\pi_{jl}(k)$, which is defined by the conditional probability that the quantized state vector $x_q(k+1)$ will be equal to the quantization level x_{ql} for gate R_l^n, given that the quantized state vector $x_q(k)$ is equal to the quantization level x_{qj} for gate R_j^n, namely,

$$\pi_{jl}(k)= Prob\{x_q(k+1)=x_{ql} \mid x_q(k)=x_{qj}\} \quad (4.9)$$

is determined as follows (Figure 3):

Let us assume that $x_q(k)$ is x_{qj} at time k. The transitions from this level to next state is decided by the random vector $w_d(k)$ and the function $Q\{f(k, x_q(k)= x_{qj}, u(k), w_d(k))\}$. Discrete random vector $w_d(k)$ can take any value in the set $\{w_{d1}(k), w_{d2}(k), \ldots , w_{dm_k}(k)\}$ with corresponding probabilities $p_{d1}(k), p_{d2}(k), \ldots , p_{dm_k}(k)$. Thus the quantized $x_q(k+1)$ can be equal to at most m_k various quantization levels. Then the transition probability of being $x_q(k+1)$ given $x_q(k)$ will be equal to the corresponding $w_{di}(k)$'s probability. That is, $\pi_{ji}(k)$ will be equal to $w_{di}(k)$'s probability, $p_{di}(k)$. However, if the function $Q\{f(k, x_q(k)= x_{qj}, u(k), w_d(k))\}$ maps x_{qj} into another gate, say R_l^n, for more than one possible value, say $w_{d1}(k)$ and $w_{d2}(k)$ of $w_d(k)$, the transition probability $\pi_{jl}(k)$ from gate R_j^n to gate R_l^n is the probability that the discrete random vector $w_d(k)$ is equal to either of the possible values $w_{d1}(k)$ or $w_{d2}(k)$, i.e., $\pi_{jl}(k)=\sum_n P_{dn}(k) =P_{d1}(k)+P_{d2}(k)$, where the summation is over all n such that $Q\{f(k, x_q(k)= x_{qj}, u(k), w_d(k))\}= x_{ql}$. Having determined the finite-state model, we can present the target motion by a Trellis Diagram.

4.4 A Trellis Diagram For the Target Motion

Since we approximated the state values and noises by discrete random vectors, we can arrange the system as a Trellis Diagram. To present the system by a graph we adopt the following conventions:

1. Each possible value of $x_q(k)$ is represented on the k^{th} column by a point (sometimes called node) with the corresponding quantization level so that the k^{th} column contains the possible quantization levels of x(k) (in other words, the possible gates in which the target can lie a time (k)) where k=0, 1, 2, … .

2. A line having a direction indicating the direction of the target motion represents the transition from one quantization level to another

Therefore, the target motion from time 0 to L can be represented by a directed graph (Trellis Diagram) shown in the following figure (Figure 5)

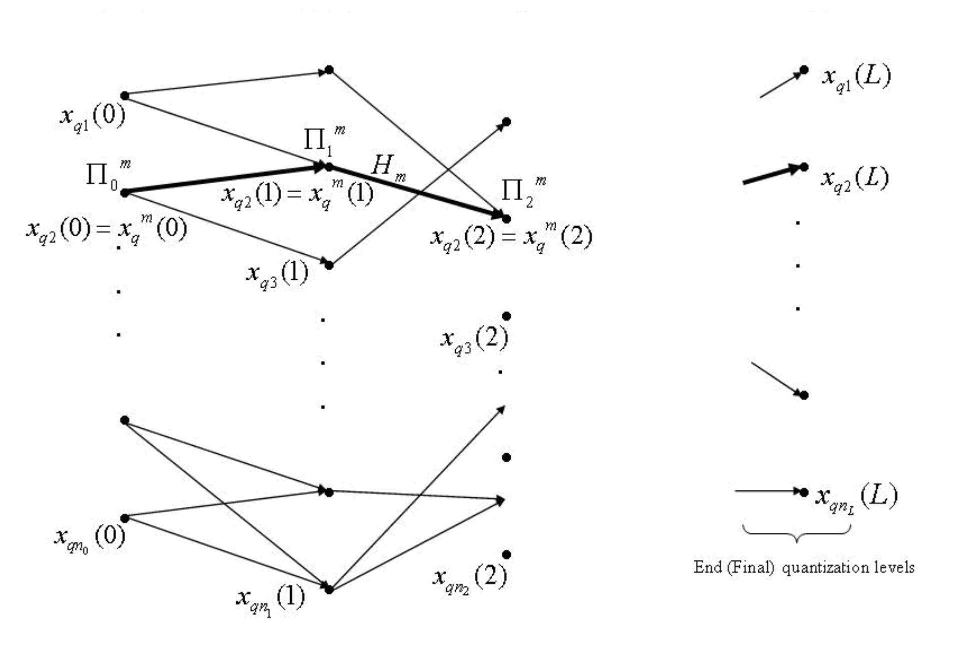

Figure 5. *The Trellis Diagram For The Target Motion*

4.5 Approximate Observation Models

Since in previous sections we reduced the system model to a finite-state model, which uses quantized $x_q(k)$, we should use the reduced state in also observation model. When we use the reduced model in observation model, the observation model becomes:

$$z(k) = g(k, x_q(k), v(k)) \tag{4.10}$$

In further analysis, we will use the equation (4.10) as observation model. At this point, it will be better to define the some symbols that will be used in our further analysis.

n_i Number of quantization levels for the gates in which the target may lie at time i.

H_m The m^{th} path through the Trellis Diagram, indicated by a bold line in Figure 5

$x^m{}_q$ Quantization level for the gate in which the target lies at time i when it follows the path H_m. In other words, the possible value of the quantized state vector $x_q(i)$ through which the m^{th} path passes. For example, in Trellis Diagram of Figure 5, $x^m{}_q(0)= x^m{}_{q2}(0)$; $x^m{}_q(1)= x^m{}_{q2}(1)$, $x^m{}_q(2)= x^m{}_{q2}(2)$

$\pi_0{}^m$ Probability that the m^{th} path passes through initial state $x_d(0)$. That is, $\pi_0{}^m=Prob\{x_d(0)= x^m{}_q(0)\}$

$\pi_i{}^m$ Transition probability from the $(i\text{-}1)^{th}$ gate (i.e., the gate for the target passes at time i-1 when it follows the path H_m) to the i^{th} gate for the m^{th} path. That is, $\pi_i{}^m=Prob\{x_q(i)=x_q{}^m(i)|x_q(i\text{-}1)=x_q{}^m(i\text{-}1)\}$

$\underset{\sim L}{x^m}$ $\underset{\sim L}{x^m} \triangleq \{ x^m{}_q\ (0),\ x^m{}_q\ (1),\ \ldots\ x^m{}_q\ (L)\}$ The sequence of the quantization levels (nodes) which the mth path passes through.

z^L $z^L \triangleq \{z(1), z(2), z(3), \ldots z(L)\}$ The observation sequence from time 1 to time L

It is obvious that, the target will follow one of the m paths that can be seen in the Trellis Diagram. Our aim is to find the most likely path that can be followed by the target. Because of the randomness in the model, our approach must be statistical. That is, statistical optimization problem. The most likely path should be chosen from the possible m paths; therefore to find the best path among the others, a criterion is needed. Our criterion will be the Minimum Error Probability Criterion, which is a special case of the Bayes' criterion in detection theory. Using this criterion reduces the problem of finding the path most likely followed by the target to a multiple hypothesis-testing problem.

4.6 Minimum Error Probability Criterion

Let we have M possible paths in the Trellis Diagram for the target to follow. We can refer these paths as M hypotheses to be determined. We would like to decide which hypothesis is true, that is, which path is most likely followed by the target.

To do this analysis, we should develop a decision rule that assigns each point in the observation space D to one of the hypotheses. Therefore, the decision rule divides the whole observation space into M subspaces. We must choose the decision regions D_1, D_2, … , D_M in such a way that overall error probability is minimized.

The overall error probability, sometimes called Bayes' risk, R is defined by:

$$R \triangleq \sum_{j=1}^{M} \sum_{\substack{i=1 \\ i \neq j}}^{M} \left\{ \int_{z_L \in D_i} p(H_j) p(z^L \mid H_j) dz^L \right\} \tag{4.11}$$

where

$p(H_j)$: Probability that the hypothesis H_j (path H_j) is true,

$p(z^L|H_j)$: Conditional probability of the observation sequence z^L ($z(1), z(2), \ldots, z(L)$) given that hypothesis H_j is true.

In order to find the optimal decision rule, we vary the decision regions D1, D_2, …, D_M so that the risk R is minimized. The optimum decision rule is:

Choose H_i if $p(H_i)p(z^L \mid H_i) > p(H_j)p(z^L \mid H_j)$ for all $j \neq i$, (4.12)

4.7 Optimum Decision Rule

For the motion model (4.8) and the observation model (4.10) the a priori probability of hypothesis H_i can be rewritten as:

$$p(H_i) = \prod_{k=0}^{L} \pi_k{}^i \qquad (4.13)$$

Since the disturbance noise vector w(k) is assumed to be independent of w(j) and x(0) for all j≠k, where π_k^i is as defined in the section 4.5.

Since the observation noise v(k) is independent from sample to sample, we can write

$$p(z^L | H_i) = \prod_{k=1}^{L} prob\{z(k) | x_q{}^i(k)\} \qquad (4.14)$$

Please note that k starts from 1, not 0 since for k=0, no decision is made. The only thing done is quantize x(0) and approximate it with n_{x0} possible values.

Since *z(k)* is a Gaussian distributed random variable, $z(k)|x_q{}^i(k)$ is also a Gaussian distributed random variable with mean $g(k, x_q{}^i(k), 0)$ and variance $\sigma^2{}_{v(k)}$. So, $p(z(k)|x_q{}^i(k))$ can be computed according to the formula given in equation (4.15).

$$prob\{z(k)|x_q{}^i(k)\} = (2\pi)^{n/2}\left[\det(R_i)\right]^{-1/2} \qquad (4.15)$$

$$\text{H}\exp\left(-\frac{1}{2}\left[z(k) - g(k, x_q{}^i(k), 0)\right]^T R_i^{-1}\left[z(k) - g(k, x_q{}^i(k), 0)\right]\right)$$

Substituting equation (4.13) and (4.14) into the optimum decision rule of equation (4.14), we obtain the following:

Choose H_i if

$$\pi_0^i \prod_{k=1}^{L} \pi_k^i \, prob(z(k) \mid x_q^i(k)) \rangle \pi_0^j \prod_{k=1}^{L} \pi_k^j \, prob(z(k) \mid x_q^j(k)) \tag{4.16}$$

for all j≠i.

Obviously, using summation will be more useful than the multiplication. Since the natural logarithm function is monotonically increasing, taking the natural logarithm of both sides of the inequality will not affect the validity of the inequality. After we take the natural logarithm of both sides, we get the following:

Choose H_i if

$$\ln(\pi_0^i) + \sum_{k=1}^{L} \left\{ \ln(\pi_k^i) + \ln(prob(z(k) \mid x_q^i(k)) \right\} > \tag{4.17}$$

$$\ln(\pi_0^j) + \sum_{k=1}^{L} \left\{ \ln(\pi_k^j) + \ln(prob(z(k) \mid x_q^j(k)) \right\}$$

for all j≠i.

4.8 Optimum Decoding-Based Smoothing Algorithm

Optimum Decoding-Based Smoothing Algorithm (ODSA) is based on following logic:

Since the target motion is represented by a Trellis Diagram, using optimum decision rule, path that has maximum probability will be found. The operation will be done by Viterbi [3,4] algorithm. This method finds the most probable path along the Trellis Diagram from time 0 to time L by comparing the metric values. Let us present some definitions that will be used throughout the chapter.

*Definition 4.6: A*n initial node is a quantization level at time zero. The metric, denoted by $MN(x_{qi}(0))$, is defined by

$$MN(x_{qi}(0))=ln[prob\{x_q(0)=x_{qi}(0)\}] \tag{4.18}$$

Therefore, $MN(x_{qi}(0))=ln(\pi_0^m)$.

Definition 4.7 The metric, denoted by $MNT[x_{qj}(k\text{-}1) \rightarrow x_{qj}(k)]$ of the branch, which connects the quantization level (node) $x_{qj}(k\text{-}1)$ to node $x_{qj}(k)$.

$$M[x_{qj}(k\text{-}1) \rightarrow x_{qj}(k)] = ln[prob\{x_q(k)=x_{qi}(k) \mid x_q(k\text{-}1)=x_{qj}(k\text{-}1)\}] \qquad (4.19)$$

This metric is called *transition metric* of node $x_{qj}(k\text{-}1)$ to node $x_{qj}(k)$.

Definition 4.8 The metric denoted by $MNO[x_{qi}(k)]$ is called the *output metric* that comes by the difference between output value and the estimated output value. That is,

$$MNO[x_{qi}(k)]=ln[p\{z(k)|x_{qi}(k)\}] \qquad (4.20)$$

Definition 4.9. The metric of a path from time 0 to time i is the summation of the metric of the initial node from time, which the path starts, and the metrics of the branches of which the path consists.

Please note that, the metric of a node $x_q(k)$ has three components which are

1. The metric comes from previous node. $MN(x_q(k\text{-}1))$
2. The metric comes from the transition from $x_q(k\text{-}1)$ to $x_q(k)$ which is transition metric.
3. The metric comes from the difference between output value and estimated value of $z(k)$, which is output metric.

To find the metric of a node, all three components of the metrics should be added up.

Definition 4.10. The error probability of a path, say H_m, through a Trellis Diagram with M possible paths $H_1, H_2, \ldots, H_M$ is the probability of deciding that a path which is different from H_M is the one most probably followed by the target when the target actually followed the path H_M. This error probability is denoted by $P_{E_M}(H_1, H_2, \ldots, H_M)$. Hence,

$$P_{E_M}(H_1, H_2, \ldots, H_M) \triangleq Prob\{z^L \in \overline{D} \mid H_M\} = \int_{z^L \in \overline{D}_M} p(z^L \mid H_M)dz^L \qquad (4.21)$$

where $\overline{D}_M$ is the complement of the decision region D_M for the path H_M. And $p(z^L|H_M)$ is the probability density function of the observation sequence z^L when the target actually followed the path H_M. Therefore, from equation (4.11), the overall error probability for the detection of the path most likely followed by the target can be expressed in terms of the path error probabilities as follows:

$$P_E=\sum_{m=1}^{M} p(H_m)P_{E_m}(H_1, H_2, ..., H_m) \tag{4.22}$$

After presenting the definitions, we can begin the implementation steps of the ODSA algorithm:

Step 0.(Initialization step) First of all, reduce the motion model to a finite state model and obtain the Trellis Diagram for the target motion from time *0* to time *L*. At time *0*, obtain the approximated initial random vector x_0 by using the algorithm explained in section 4.2. Also, using equation (2.18) assign each node to its metric value.

Step 1. At time 1, using the initial node's metric, transition probability and the observation $z(1)$, evaluate the metrics of the new nodes. Take a fixed number of the nodes that have larger metric to limit the number of states since otherwise the number of the states might be blown up and make the algorithm useless.

Step k. For each node at time *k*, using the previous node's metric, transition probability and observation at time *k*, $z(k)$, evaluate the metrics of the $(k+1)^{th}$ nodes. Take a fixed number of the nodes that have larger metric to limit the number of the states since otherwise the number of the states would be blown up and make the algorithm useless.

Step Final. After *k* reaches to *L*, choose the node that has maximum metric value. Then, by going backward to the parent nodes of the chosen node, the chain of the most probable states will be gotten.

4.9 An Example of ODSA Algorithm

To understand the ODSA algorithm better it will be very useful to apply the algorithm on a model for few steps. Let us apply the ODSA to Lotka-Volterra Model for $L=2$.

Initial step. First of all, we decide the program parameters n_{x0}, n_{y0}, n_{w1}, n_{w2}, n_{br} and n_{dr} which are the quantization levels of the random vectors x_0, y_0, w_1, w_2, br and dr. Let we have $n_{x0}=5$, $n_{y0}=5$, $n_{w1}=3$, $n_{w2}=3$, $n_{br}=5$ and $n_{dr}=5$. Then, at time 0, we have $n_{x0}Hn_{y0}=5x5=25$ nodes and the initial metrics to all nodes which are the logarithm of the corresponding probabilities.

Moreover, we have to decide the other system parameters such *as* $mean_{X0}$, $mean_{Y0}$, var_{x0}, var_{y0}, var_{w1}, var_{w2}, $mean_{br}$, $mean_{dr}$, var_{br}, var_{dr}, $\sigma_{v_1}^{2}$, $\sigma_{v_2}^{2}$, *GS*(gate size) and *MaxState*(maximum number of states). Let us clearly define these parameters and assign them their values:

- $mean_{X0}$: The mean of the initial state $X0$, which is the mean of the initial value of the Rabbits in the ecosystem. Let it be 20, i.e. $mean_{X0}=20$.
- $mean_{Y0}$: The mean of the initial state $Y0$, which is the mean of the initial value of the Foxes in the ecosystem. Let it be 25, i.e. $mean_{Y0}=25$.
- var_{X0}: The variance of the number of existing Rabbits initially in the ecosystem. Let it be 9, i.e., $var_{X0}=9$.
- var_{Y0}: The variance of the number of existing Foxes initially in the ecosystem. Let it be 4, i.e., $var_{Y0}=4$.
- var_{w1}: The variance of the disturbance noise to the system. That is the effects on the number of Rabbits other than foxes, which cannot be explained by the model, such as eaten rabbits by some animals other than Foxes or etc. Let it be 4, i.e., $var_{w1}=4$.
- var_{w2}: The variance of the disturbance noise to the system. That is the effects on the number of Foxes other than rabbits, which cannot be explained by the model, such as foxes nourished by some animals other than Rabbits. Let it be 4, i.e., $var_{w2}=4$.

- $mean_{br}$: Average of the usual birth rate of the Rabbits. . Let it be 0.7, i.e., $mean_{br}=0.7$.
- $mean_{dr}$: Average of the usual death rate of the Foxes. Let it be 0.5, i.e., $mean_{dr}=0.5$.
- var_{br}: Variance of the birth rate of the Rabbits. Let it be 0.1, i.e., $var_{br}=0.1$.
- var_{dr}: Variance of the dearth rate of the Foxes. Let it be 0.1, i.e., $var_{dr}=0.1$.
- $\sigma_{v_1}^2$: Variance of the observation noise of the number of the Rabbits. Let it be 1, i.e., $\sigma_{v_1}^2=1$.
- $\sigma_{v_2}^2$: Variance of the observation noise of the number of the Foxes. Let it be 4, i.e., $\sigma_{v_2}^2=4$.
- *GS:* The gate size of the quantizer function. Let it be 0.01, i.e., $GS=0.01$.
- *MaxState:* The maximum number of the states allowed that could be taken into account while passing to next time. Let it be 100, i.e., $MaxState=100$.

At this moment, we can evaluate and write down the initial nodes. In the evaluation of x_0 and y_0, we will use the following equations:

$$x_q(0) = Q\left\{meanX_0 + \left[x_{qi}(0)\right] \times \sqrt{\operatorname{var} x_0}\right\} \tag{4.23}$$

$$y_q(0) = Q\left\{meanY_0 + \left[y_{qi}(0)\right] \times \sqrt{\operatorname{var} y_0}\right\} \tag{4.24}$$

Let us define $(XY)_{k,l}$ as the l^{th} node at time k.

$(XY)_{0,1}$= Quantize {($mean_{X0}$+1st value of n_{X0} approximation points)×square root of variance of *X0}*

For example

$(XY)_{0,1}$= Q{20+ (-1.3767) ×√9}=Q{20- 4.1301}=Q{15.8699}=15.87

The other nodes can be evaluated using (4.23) and (4.24).

Let us form the nodes of the initial state of the Trellis Diagram:

- $(XY)_{0,1}=(15.87\ ,\ 22.25\ )\ ;\ MN((XY)_{1,0})=ln(0.169)+ln(0.169)=-3.5605$
- $(XY)_{0,2}=(15.87\ ,\ 23.82\ )\ ;\ MN((XY)_{2,0})=ln(0.169)+ln(0.216)=-3.3095$
- $(XY)_{0,3}=(15.87\ ,\ 25.00\ )\ ;\ MN((XY)_{3,0})=ln(0.169)+ln(0.230)=-3.2521$
- $(XY)_{0,4}=(15.87\ ,\ 26.18)\ ;\ MN((XY)_{4,0})=ln(0.169)+ln(0.216)=-3.3095$
- $(XY)_{0,5}=(15.87\ ,\ 27.75)\ ;\ MN((XY)_{5,0})=ln(0.169)+ln(0.169)=-3.5605$
- $(XY)_{0,6}=(18.22\ ,\ 22.25)\ ;\ MN((XY)_{6,0})=ln(0.216)+ln(0.169)=-3.3095$
- ...
- $(XY)_{0,13}=(20.00\ ,\ 25.00)\ ;\ MN((XY)_{13,0})=ln(0.230)+ln(0.230)=-2.9437$
- ...
- $(XY)_{0,25}=(24.13\ ,\ 27.75)\ ;\ MN((XY)_{25,0})=ln(0.169)+ln(0.169)=-3.5605$

Once we prepared the initial nodes $(XY)_0$ and corresponding metrics, we can pass to step 1.

Step 1. $(XY)_1$ is determined by 4 unknown variables, namely, w_1, w_2, br and dr. It is obvious that also $(XY)_0$ determines $(XY)_1$.

Let us start evaluating the nodes of the $(XY)_1$ from $(XY)_{0,1}$. Since there are 4 variables that determines $(XY)_1$, and according to the assumptions made in the previous step, that is, $n_{w1}=3,\ n_{w2}=3,\ n_{br}=5$ *and* $n_{dr}=5$, we can have $3\times3\times5\times5=225$ different values while coming from $(XY)_{1,0}$. The values depend on the variables n_{w1}, n_{w2}, n_{br} and n_{dr}.

After we assign all possible values to whole $(XY)_1$ pairs, we are ready to metric evaluation. As stated in the section 4.8, metrics have 3 components:

1. Metric coming from the previous node, $MN((XY)_{0,1})=-3.5605$
2. Metric due to the transition, which is the sum of the natural logarithms of $prob(w_1=w_{1,1})$, $prob(w_2=w_{2,1})$, $prob(br=br_1)$ and $prob(dr=dr_1)$. That is, $ln(0.315)+ln(0.315)+ln(0.169)+ln(0.169)=-5.8721$
3. The metric comes from the observation. Natural logarithm of the $prob(z(k)|(XY)_{1,1})$. Since observation equation is

$$z(k) = \begin{bmatrix} x(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{4.25}$$

and the observation noises v_1 and v_2 are independent from each other,

$$prob(z(k) | (XY)_{1,1}) = \tag{4.26}$$

$$prob(x(1) + v_1(1) | (XY)_{1,1}) \times prob(y(1) + v_2(1) | (XY)_{1,1})$$

Then,

$$prob(x(1) + v_1(1) | (XY)_{1,1}) = N(x_{1,1}, \sigma_{v_1}^{\ 2})$$

$$= \frac{1}{\sqrt{2\pi\sigma^2_{v_1}}} e^{-\frac{\{x_{1,1} - x(1)\}^2}{2\sigma^2_{v_1}}} \tag{4.27}$$

where $x(1)$ is the observation of x at time 1, $x_{1,1}$ is approximated value of x at time 1 and at node 1 and $\sigma_{v_1}^{\ 2}$ =1, also

$$prob(y(1) + v_2(1) | (XY)_{1,1}) = N(y_{1,1}, \sigma_{v_2}^{\ 2})$$

$$= \frac{1}{\sqrt{2\pi\sigma^2_{v_2}}} e^{-\frac{\{y_{1,1} - y(1)\}^2}{2\sigma^2_{v_2}}} \tag{4.28}$$

where $y(1)$ is the observation of x at time 1, $y_{1,1}$ is approximated value of y at time 1 and at node 1and $\sigma_{v_2}^{\ 2}$ $=4$

Consequently, metric due to observation is

$$MNO((XY)_{1,1}) = ln\{ prob(z(k) | (XY)_{1,1}) \} = \tag{4.29}$$

$$ln\{ prob(x(1) + v_1(1) | (XY)_{1,1}) \} + ln\{ prob(y(1) + v_2(1) | (XY)_{1,1}) \}$$

Let $x(1)=19.29$ and $y(1)=26.41$, then

$$ln\{ prob(z(k)|(XY)_{1,1})\}= -7.7720+(-5.3799)= -13.1519 \qquad (4.30)$$

Now, to find the metric of $(XY)_{1,1}$, we will add up the tree metrics above. Then,

$$MN((XY)_{1,1})= MN((XY)_{0,1})_{previous}+ MN((XY)_{1,0}\rightarrow(XY)_{1,1})_{transition}+MNO((XY)_{1,1}) \qquad (4.31)$$

$$= -3.5605+(-5.8721)+(-13.1519)$$

$$= -22.5845$$

The same operation should be done for $(XY)_{1,2}$, $(XY)_{1,3}$, … , $(XY)_{1,225}$ by changing the value of the variables at each turn. Then, we have 225 $(XY)_1$ pairs that are coming from $(XY)_{1,0}$. At this point, (XY) pairs should be quantized with the proper *gate size*. Note that after quantization, some (XY) pairs may be equal. When (XY) pairs are equal, the probabilities of the variables that give same (XY) pair should be added and then total probability is taken in the calculation of the transition metric. After this operation, sometimes the number of "children nodes" of $(XY)_{0,1}$ may decrease from *225*. Please note that, if the gate size is large, it causes more reduction in the number of nodes on behalf of increase in the error. On the other hand, if the gate size is too small fewer nodes will fall into same gate, which causes increase in computation time. Obviously, in this case the error will decrease. Therefore, compensation between gate size *GS* and *Error* should be done.

The operation done to node $(XY)_{0,1}$ should be done for all nodes at time 0. That is, $(XY)_{0,2}$, $(XY)_{0,3}$, … , $(XY)_{0,25}$. Each will produce *225* (or less) nodes. In case of being same $(XY)_1$ pair coming from different parent nodes, the one having larger metric will be chosen and other having smaller metric will be thrown out.

Totally, we will have at most $225x25=5625$ different nodes for $(XY)_1$. At the next time, say $(XY)_2$, the number of the nodes will reach to $5625x225= 1265625$ which is an enormous number. However, as mentioned in the section 4.8, the number of states will be limited. Since for this example we assumed that the maximum number of states is *MaxState*=1*00*, the number of states will not exceed 1*00.* Obviously we will choose the states 100 states which have the largest metric. Therefore, at the end of step 1, we will have *at most* 100 different nodes and their associated metric values.

Step 2.The same operation should be done as step 1, the only difference is the number of initial states. At step1, we had 25 initial nodes, however, in this step, we have 100 different initial nodes. The whole operation is same with step 1.

After the operations are done, we will have again at most 100, the number of maximum allowed states, and the associated metric values.

Step Final. Since $k=2$ is reached, note that $L=2$ for this example, we should check the metric values *of* $(XY)_2$ node. Then, the node that has largest metric value is our most probable state at time 2. For $k=1$, we should go to the parent of the node chosen as most probable node at time 2. For $k=0$, again we should go to the parent of the node chosen at time 1. Then, the most probable state chain is

$\{(XY)_{0,i}, (XY)_{1,j}, (XY)_{2,l}\}$ where

$(XY)_{2,l}$ is the node that has largest metric value.

$(XY)_{1,j}$ is the parent of the $(XY)_{2,l}$ node.

$(XY)_{0,i}$ is the parent of the $(XY)_{1,j}$ node.

4.10 Complexity Analysis of ODSA

The runtime of the program written for ODSA is determined by the maximum number or states, *MaxState,* the time L, numbers of quantization levels n_{X0}, n_{Y0}, n_{w1}, n_{w2}, n_{br}, n_{dr} and the gate size *GS*.

If the number of states were not limited, the program will be useless for the Stochastic Discrete Lotka Volterra Model, since the runtime will blow up. Each time step the possible maximum time consumption for will be $n_{X0}\, n_{Y0}(n_{w1}n_{w2}n_{br}n_{dr})^k$ at time k. Let the maximum time consumption at each state be t_s, then the time consumption at each time step k will be $t_s n_{X0}\, n_{Y0}(n_{w1}n_{w2}n_{br}n_{dr})^k$. Then the total runtime will be equal to:

$$\text{Runtime} = (t_s n_{X0}\, n_{Y0} + t_s n_{X0}\, n_{Y0} n_{w1}n_{w2}n_{br}n_{dr} + t_s n_{X0}\, n_{Y0}(n_{w1}n_{w2}n_{br}n_{dr})^2 + t_s n_{X0} n_{Y0}(n_{w1}n_{w2}n_{br}n_{dr})^3 + \ldots + t_s n_{X0}\, n_{Y0}(n_{w1}n_{w2}n_{br}n_{dr})^L) = t_s n_{X0} n_{Y0} \frac{(n_{w1}n_{w2}n_{br}n_{dr})^{L+1} - 1}{(n_{w1}n_{w2}n_{br}n_{dr}) - 1} \tag{4.32}$$

where

t_s : Maximum time consumption at each state

n_{X0} : The number of quantization level of state $X(0)$

n_{Y0} : The number of quantization level of state $Y(0)$

n_{w1} : The number of quantization level of disturbance noise $w_1(k)$

n_{w2} : The number of quantization level of disturbance noise $w_2(k)$

n_{br} : The number of quantization level of birth rate $br(k)$

n_{dr} : The number of quantization level of death rate $dr(k)$

Since $(n_{w1}n_{w2}n_{br}n_{dr})^{L+1} >> 1$, $(n_{w1}n_{w2}n_{br}n_{dr})^{L+1} - 1 \approx (n_{w1}n_{w2}n_{br}n_{dr})^{L+1}$. Therefore,

$$\text{Runtime} \approx \frac{t_s n_{X0} n_{Y0}}{(n_{w1}n_{w2}n_{br}n_{dr}) - 1}(n_{w1}n_{w2}n_{br}n_{dr})^{L+1} \tag{4.33}$$

The program run time increases exponentially as time L increases. So, in case of unlimited number of states, the program complexity will be $O((n_{w1}n_{w2}n_{br}n_{dr})^L)$.

If the number of states is limited by some value at each time k, the program runtime for the worst case will be:

$$\text{Runtime} = t_s\, n_M L \qquad (4.34)$$

where

t_s : Maximum time consumption at each state

n_M : The maximum number of states allowed.

From 4.31 we can say that program run time increases linearly as time L increases. Therefore, if the maximum number of states is limited, the program complexity will be $O(L)$.

On the other hand, the gate size effects the program run time since due to quantization, some of the states will fall into same gate and these states will be discarded. Therefore, larger gate size will decrease the program run time.

4.1.10 Program Runtime Simulation

In this section, the changes in the program runtime will be studied via the simulation plot. If the maximum number of states is limited, the Figure 6 shows the one step runtime versus iterations where one step runtime is the amount of time passes in one time-step, that is from time k to time k+1. The model parameters are chosen as same values in the simulation Chapter 2, and the algorithm parameters are as follows:

- Maximum number of states (*MaxState*) : 25
- Quantization number of Initial State X0 (n_{X0}) : 5
- Quantization number of Initial State Y0 (n_{Y0}) : 5
- Quantization number of the disturbance noise w_1 (n_{w1}) : 3
- Quantization number of the disturbance noise w_2 (n_{w2}) :3
- Quantization number of the birth rate (n_{br}) : 5
- Quantization number of the death rate (n_{dr}) : 5

The program runtime is plotted for various Gate Size (*GS*) values.

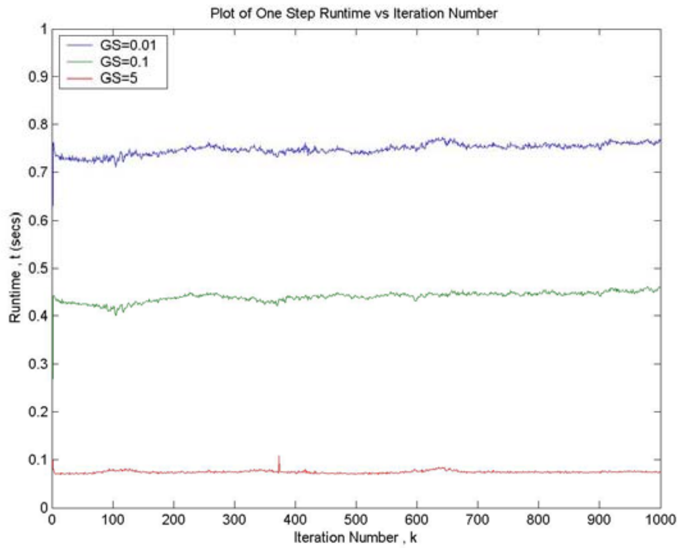

Figure 6 *Plot of One Step Runtime vs. Iteration Number*

It is clear in the plot that when the gate size, *GS*, increases, the program runtime decreases since some of the nodes fall down into the same gates and the those states are discarded. Consequently, this yields decrease in the runtime.

CHAPTER 5

PARTICLE FILTER

5.1 Particle Filters

Particle filtering is a sequential Monte Carlo methodology where the basic idea is the recursive computation of relevant probability distributions using the concepts of importance sampling and approximation of probability distributions with discrete random measures. The earliest applications of sequential Monte Carlo methods were in the area of growing polymers [29], [30], and later they expanded to other fields including physics and engineering. Sequential Monte Carlo methods found limited use in the past, except for the last decade, primarily due to their very high computational complexity and the lack of adequate computing resources of the time. The fast advances of computers in the last several years and the outstanding potential of particle filters have made them recently a very active area of research. Their potential for parallel implementation represents additional impetus for their development. The current interest in particle filtering for signal processing applications was brought on by Gordon, Salmond and Smith [31].

Let the system equation be in form:

$$x_k = f(x_{k-1}, w_{k-1}) \tag{5.1}$$

where, $f : \Re^{n_x} \times \Re^{n_w} \rightarrow \Re^{n_x}$ is a possibly nonlinear function of x_k and w_k. The state variable $x_k \in \Re^{n_x}$ and the noise $w_k \in \Re^{n_w}$ is independent and identically distributed (i.i.d) process noise sequence. n_x and n_w are the dimensions of the state and the process noise vectors, respectively. The aim is to recursively estimate x_k using the measurements:

$$z_k = h(x_k, v_k) \tag{5.2}$$

where, $h : \Re^{n_x} \times \Re^{n_v} \rightarrow \Re^{n_z}$ is possibly nonlinear function of x_k and v_k. The state variable $x_k \in \Re^{n_x}$ and the noise $v_k \in \Re^{n_v}$ is (i.i.d) measurement noise sequence. n_x and n_v are the dimensions of the state and the measurement noise vectors, respectively. In particular, particle filter seeks filtered estimates of x_k based on the set of all available measurements $z_{1:k} = \{z_i, i = 1,2,\ldots,k\}$, up to time k.

5.1.1 Sequential Importance Sampling (SIS) Algorithm

The Sequential Importance Sampling (SIS) [31] algorithm is a Monte Carlo (MC) method that forms the basis for most sequential MC filters developed in the past decades. The key idea of the SIS algorithm is to represent the required posterior density function by a set of random samples with associated weights and to compute estimates based on these samples and weights.

Let the probability density function (pdf) of the x_k approximated with the weighted samples where $\sum_{i} s_k^i = 1$. Then the pdf at k can be approximated as:

$$p(x_{0:1} \mid z_{1:k}) \approx \sum_{i=1}^{N_s} s_k^i \delta(x_{0:k} - x_{0:k}^i). \tag{5.3}$$

where N_s is the number of sample points approximating the pdf. The weights are chosen using *importance sampling*. That is,

$$s_k^{\mathrm{i}} \propto \frac{p(x_{0:k}^i \mid z_{1:k})}{q(x_{0:k}^i \mid z_{1:k})} \tag{5.4}$$

where, q(.) is importance density.

The algorithm of the SIS is following:

- FOR i=1:N_s
  - Draw $x_k^i \sim q(x_k \mid x_{k-1}^i, z_k)$
  - Assign the particle a weight, s_k^i, according to

$$s_k^i \propto s_{k-1}^i \frac{p(z_1 \mid x_k^i) p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, z_{1:k})} \qquad (5.5)$$

- END FOR

On the SIS algorithm, degeneracy problem can occur in some modes. To overcome this problem, Resampling can be made. Therefore Sampling Importance Resampling Filter SIR [31] is developed.

5.1.2 Sampling Importance Resampling (SIR) Algorithm

SIS algorithm sometimes suffers from degeneracy problem which is after few iterations, all but one particle will have negligible weight. The variance of the importance weights can only increase over time; therefore, it is impossible to avoid the degeneracy phenomenon. As degeneracy problem occurs, large computational effort is devoted to updating particles whose contribution to approximation of pdf is almost zero.

To overcome degeneracy problem, resampling will be used. The basic idea of resampling is to eliminate particles that have small weight and concentrate on the particles that have large weights. The resampling step involves generating a new set $\{x_k^{i*}\}_{i=1}^{N_s}$ by *resampling* (with replacement) Ns times from an approximate discrete representation of $p(x_k^i \mid z_{1:k})$ given by

$$p(x_k \mid z_{1:k}) \approx \sum_{i=1}^{N_s} s_k^i \delta(x_k - x_k^i). \qquad (5.6)$$

so that, $\Pr(x_k^{i*} = x_k^i) = s_k^i$. The resulting sample is in fact an i.i.d. sample from the discrete density (5.6); therefore, weights are now reset to $s_k^i = 1/N_S$

The algorithm of Resampling is as follows:

- Initialize the constant $c_0=0$
- FOR $i=1:N_s$
  - $c_i = c_{i-1} + s_k^i$
- End FOR
- Draw a starting point $u_1 \in \left[0, 1/N_s\right]$, uniformly distributed.
- FOR $j=1:N_s$
  - $i=1$
  - $u_j = u_1 + \frac{j-1}{N_s}$
  - WHILE $u_j > c_i$
    - $i=i+1$
  - END WHILE
  - Assign $x_k^{j*} = x_k^j$, $s_k^{j*} = 1/N_s$
- END FOR

Then the SIR algorithm becomes;

$$x_k = f(x_{k-1}, w_{k-1}) \qquad (5.7)$$

$$z_k = h(x_k, v_k) \qquad (5.8)$$

- Generate N_s values from pdf of x_0
- Generate N_s values from pdf of v_k

- Calculate $x_k^i = f(x_{k-1}^i, w_{k-1}^i), i = 1, \ldots, N_s$
- Calculate $s_k^i = \dfrac{p(z_k \mid x_k = x_k^i)}{\sum_{j=1}^{N_s} p(z_k \mid x_k = x_k^j)}$
- $E\{x_k \mid z_{1,k-1}\} = \dfrac{\left(\sum_{i=1}^{N_s} x_k^i\right)}{N_s}$ (Prediction)
- RESAMPLING $\left[\{x_k^i, s_k^i\}_{i=1}^{N_s}\right] = \left[\{x_k^{j*}, s_k^{j*}\}_{i=1}^{N_s}\right]$
- $E\{x_k \mid z_{1,k}\} = \dfrac{\left(\sum_{i=1}^{N_s} x_k^{i*}\right)}{N_s}$, $z_{1,\mathrm{k}} = \{z_1, z_2, \ldots, z_k\}$ (Filtering)

When this SIR algorithm is applied to Stochastic Discrete Lotka Volterra Model in section 2.2.1,

For *Ns*=25, the resultant filtering output becomes:

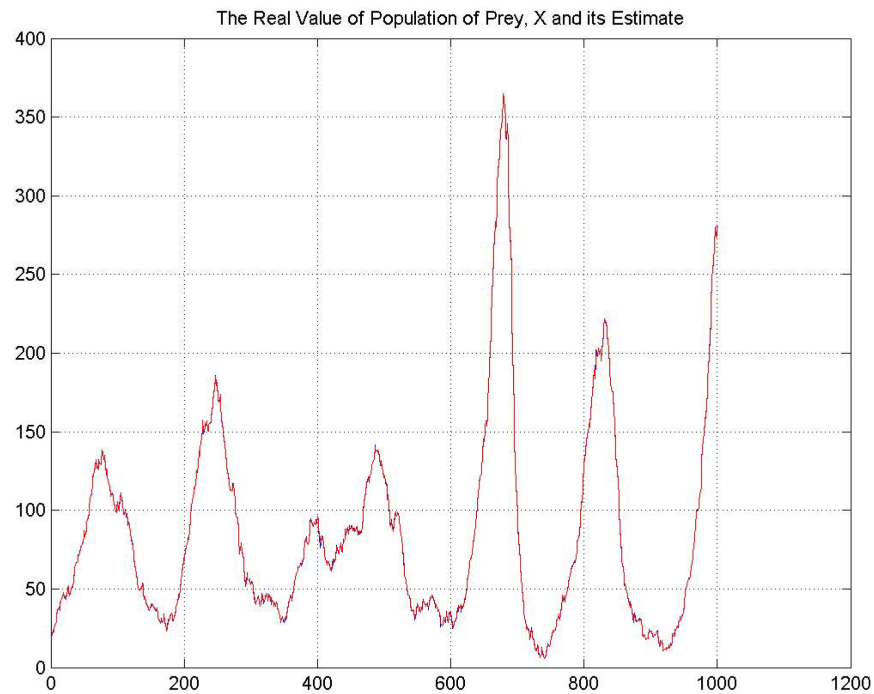

Figure 7 *The Real Value of Population of Prey, X, and Its Estimate*

The Error is :

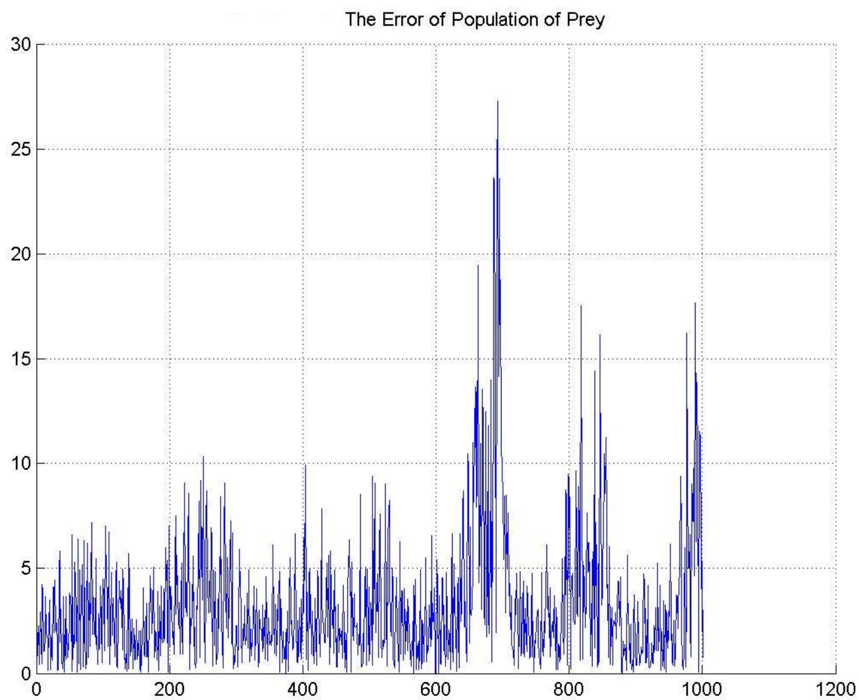

Figure 8. *The Error of Population of Prey, X*

The system model is:

$$X[k+1]=X[k]+brR[k]-a_1X[k]Y[k]+w_1[k] \qquad (5.9)$$

$$Y[k+1]=Y[k]-drY[k]-a_2Y[k]X[k]+w_2[k] \qquad (5.10)$$

The system parameters are chosen as:*Ns*=1*00;*

L=1*000;*

*a*1*=0.007;*

a2=0.006;

meanBR=0.7;

meanDR=0.5;

meanX0= 20;

meanY0= 25;

varBR=0.1;

varDR=0.1;

varw1=4;

varw2=4;

varv1= 1;

varv2= 4;

varX0= 1;

varY0= 1;

The resultant RMS mean of the error for this sample run is:

RMS Error Mean = 4,585

RMS Error Variance = 10,8227

Run Time= 4.136 secs

where RMS error is defined as:

RMS Error $\triangleq \sqrt{\dfrac{\sum_{j=1}^{N}(z_j - \overline{z}_j)^2}{N}}$,   where N is number of turns and $\overline{z}_j$ is estimated value of z_j.

The error performance of the algorithms will be given in detail in the CHAPTER 7.

CHAPTER 6

SIMULATION RESULTS FOR THE ODSA

Simulations of ODSA applied to the Stochastic Discrete Lotka-Volterra Model are done by using MATLAB®. In order to simulate the Discrete Lotka-Volterra Model, state and observation vectors should be generated. The noises of the model are generated at MATLAB® using *randn(.)* function of MATLAB® to generate Gaussian distributed random vectors.

The Stochastic Discrete Lotka-Volterra Model is given in the following:

$$X[k+1]=X[k]+brX[k]-a_1X[k]Y[k]+w_1[k] \qquad (6.1)$$

$$Y[k+1]=Y[k]-drY[k]-a_2Y[k]X[k]+w_2[k] \qquad (6.2)$$

The observation equation is,

$$Z[k]=\begin{bmatrix} X[k] \\ Y[k] \end{bmatrix}+\begin{bmatrix} v_1 k] \\ v_2[k] \end{bmatrix} \qquad (6.3)$$

At the whole simulations done in Chapter 6, unless otherwise stated, the model parameters are chosen as in the following:

- Mean of initial state X0 ($mean_{X0}$) : 20

- Mean of initial state Y0 ($mena_{Y0}$) : 25
- Variance of initial state X0 (var_{X0}) : 4
- Variance of initial state Y0 (var_{Y0}) : 9
- Mean of birth rate of X ($mean_{br}$) : 0.7
- Mean of death rate of X ($mean_{dr}$) : 0.5
- Variance birth rate of X (var_{br}) : 0.1
- Variance death rate of X (var_{dr}) :0.1
- The interaction parameter between X and Y (a_1) : 0.007
- The interaction parameter between Y and X (a_2) : 0.006
- Variance of disturbance noise w_1 (var_{w1}) : 4
- Variance of disturbance noise w_2 (var_{w2}) : 4
- Variance of observation noise v_1 (var_{v1}) : 1
- Variance of observation noise v_2 (var_{v2}) : 4

Moreover, unless otherwise stated, the algorithm parameters are the following:

- Maximum number of states (*MaxState*) : 25
- Gate Size (*GS*) : 0.01
- Quantization number of Initial State X0 (n_{X0}) : 5
- Quantization number of Initial State Y0 (n_{Y0}) : 5
- Quantization number of the disturbance noise w_1 (n_{w1}) : 3
- Quantization number of the disturbance noise w_2 (n_{w2}) :3
- Quantization number of the birth rate (n_{br}) : 5
- Quantization number of the death rate (n_{dr}) : 5

Simulations are done according to the algorithm given in sections 4.8 and 4.9. At each execution, the state and the observation vectors have been changed.

First of all, 100 run Monte Carlo simulations are done to see the error performance and the course of error is plotted. Secondly, the effects of the model parameters are investigated via simulations and the results are plotted. Finally, the effects of ODSA parameters are investigated via simulations and the results are plotted.

6.1 The Error Performance of the Algorithm

In this section, the plots of the simulation results are given. For each parameter, the RMS (Root-Mean-Square) errors of the state values X and Y are plotted. After the plots, the mean of the RMS error, variance of the RMS error are given in a table. Moreover, RMS error is defined as:

RMS Error $\triangleq \sqrt{\dfrac{\sum_{j=1}^{N}(z_j - \overline{z_j})^2}{N}}$, where N is number of turns and $\overline{z}_j$ is estimated value of z_j.

In addition, mean and variance of RMS error is defined below:

Mean of RMS Error $\triangleq \dfrac{1}{N}\sum_{j=1}^{N} E_{RMS_j}$

Variance of RMS Error $\triangleq \dfrac{1}{N}\sum_{j=1}^{N}\left(E_{RMS_j} - \dfrac{1}{N}\sum_{i=1}^{N} E_{RMS_i}\right)^2$

where E_{RMS_k} is RMS Error at k^{th} run.

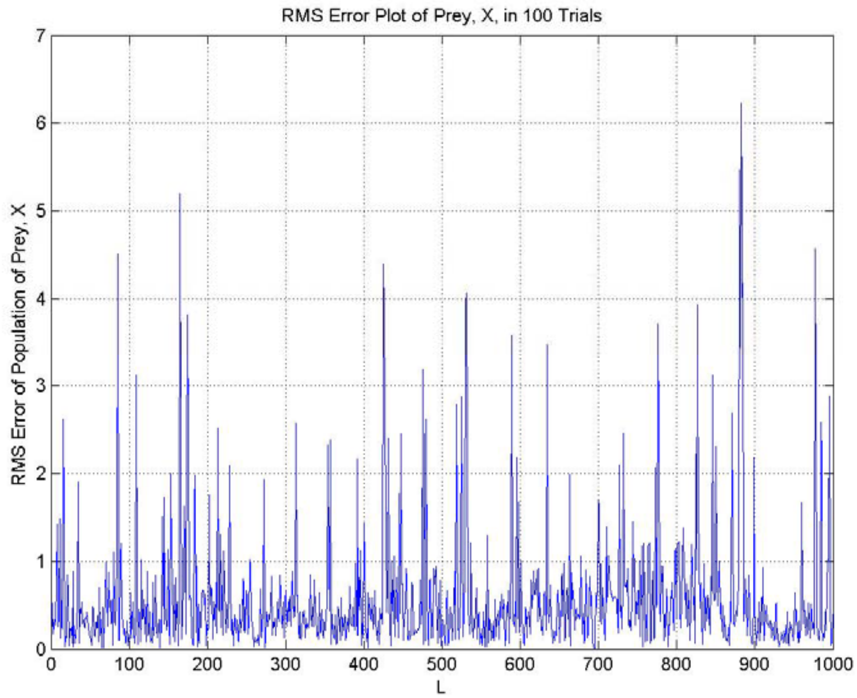

Figure 9. *Plot of RMS Error of X in 100 turns*

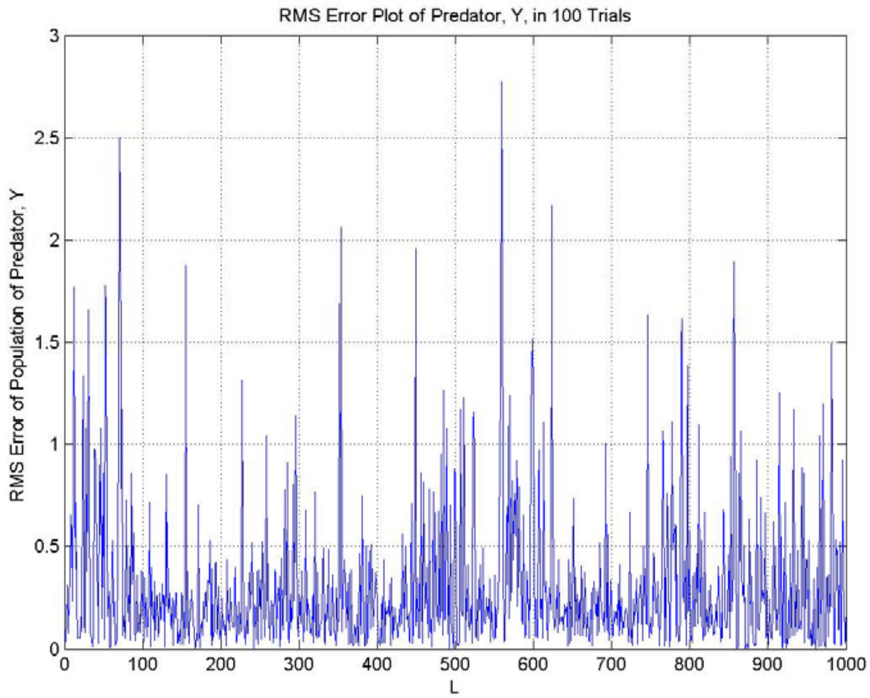

Figure 10. *The Plot of RMS Error of Y in 100 Turns*

Some numerical data of the graphics above are the following:

Table 3.*The numerical data of the RMS error plot of ODSA*

Mean RMS Error of X	0.5734
Mean RMS Error of Y	0.3002
Variance of RMS Error of X	0.5234
Variance of RMS Error of Y	0.1251

6.2 The Effects of Model Parameters

The algorithm output depends on both model and algorithm parameters. The model parameters are due to the model itself. These parameters are usually the variances of random terms in the model equation (6.1 and 6.2), namely, initial state variance, disturbance noise variance, death-birth rate variances and the observation noise variance.

6.2.1 The Effect of Variance of Initial States X_0 and Y_0 (var_{X0} & var_{Y0})

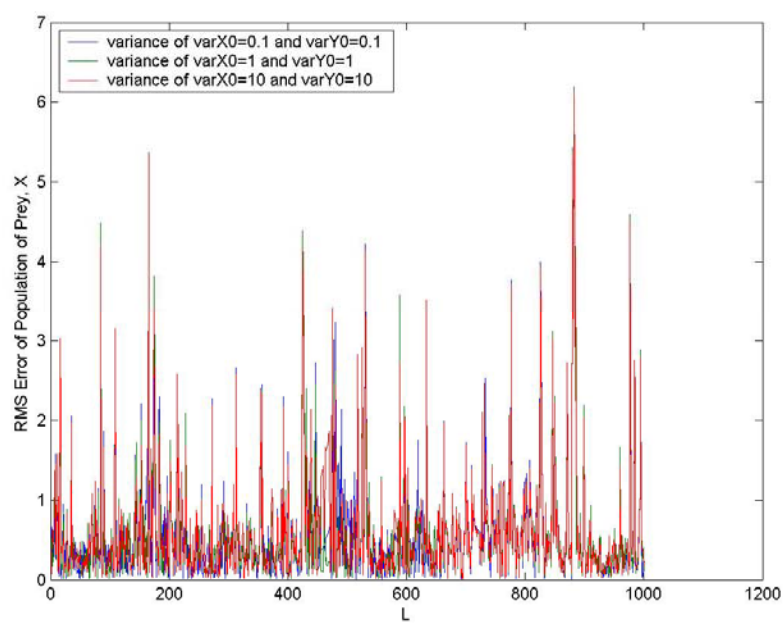

Figure 11 *The Plot of RMS Error of X vs. L, for Various Initial State Variances*

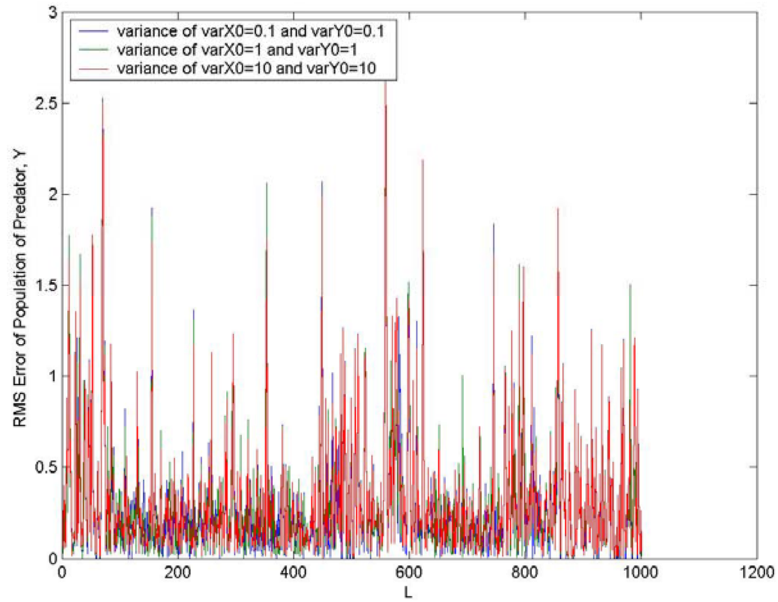

Figure 12. *The Plot of RMS Error of Y vs. L, for Various Initial State Variances*

Some numerical data of the graphics above are the given in the following table:

Table 4.*The numerical data of the RMS error for various initial state variances*

	VarX0= 0.1 VarY0= 0.1	VarX0= 1 VarY0= 1	VarX0= 10 VarY0= 10
Mean RMS Error of X	0.5767	0.5802	0.5975
Mean RMS Error of Y	0.2968	0.3017	0.3074
Variance of RMS Error of X	0.5158	0.5182	0.5198
Variance of RMS Error of Y	0.1232	0.1248	0.1295

6.2.2 The Effect of Variances of Disturbance Noises w_1 and w_2 (var_{w1} & var_{w2})

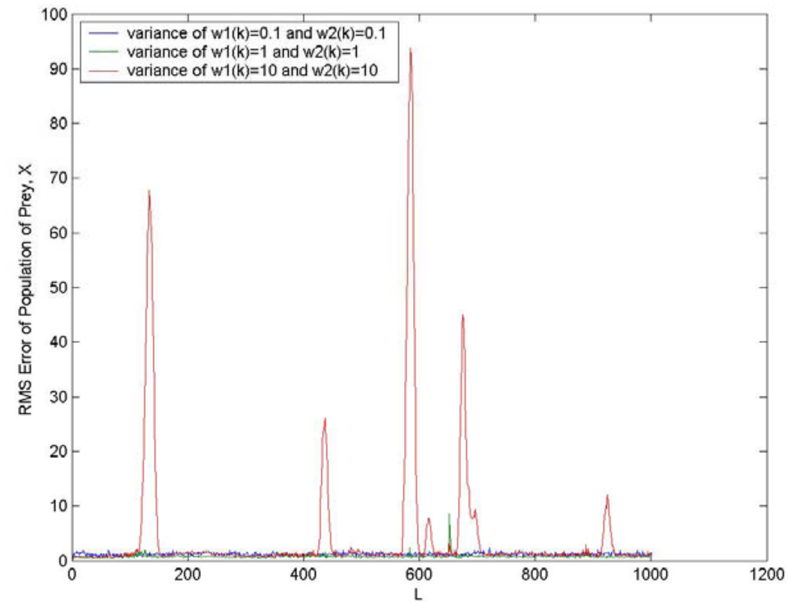

Figure 13. *The Plot of RMS Error of Y vs. L, for various Disturbance Noise Variances*

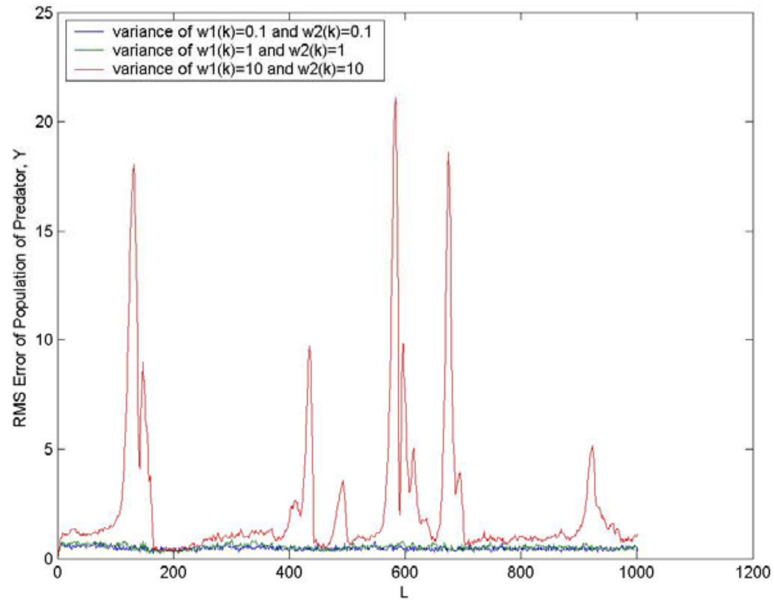

Figure 14. *The Plot of RMS Error of X vs. L, for various Disturbance Noise Variances*

Some numerical data of the graphics above are the given in the following table:

Table 5.*The numerical data of the RMS error for various disturbance noise variances*

	varw1= 0.1 varw2= 0.1	varw1= 1 varw2= 1	varw1= 10 varw2= 10
Mean RMS Error of X	0.7718	1.1895	4.4490
Mean RMS Error of Y	0.4658	0.5244	2.2649
Variance of RMS Error of X	0.0617	0.1531	142.36
Variance of RMS Error of Y	0.0082	0.0136	10.5760

6.2.3 The Effect of Variances of Birth Rate and Death Rate (*var_{br}* & *var_{dr}*)

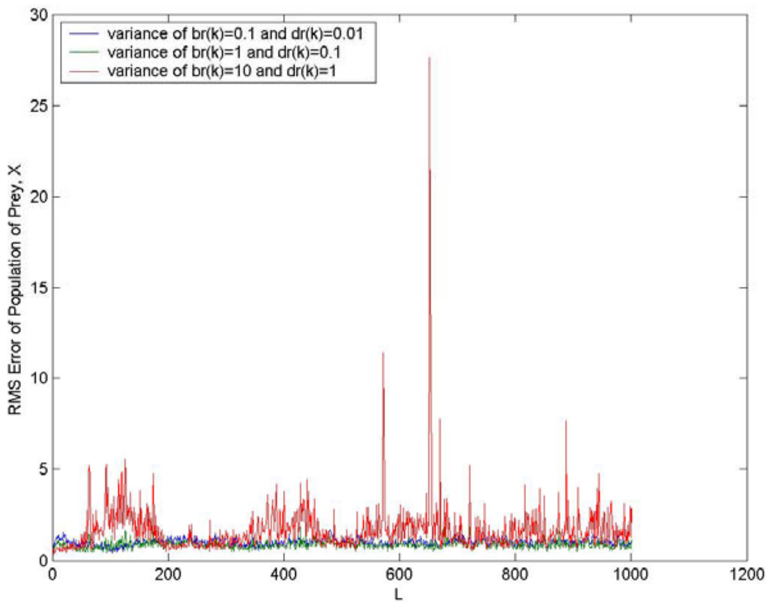

Figure 15 *The Plot of RMS Error of X vs. L, for Various Birth and Death Rate Variances*

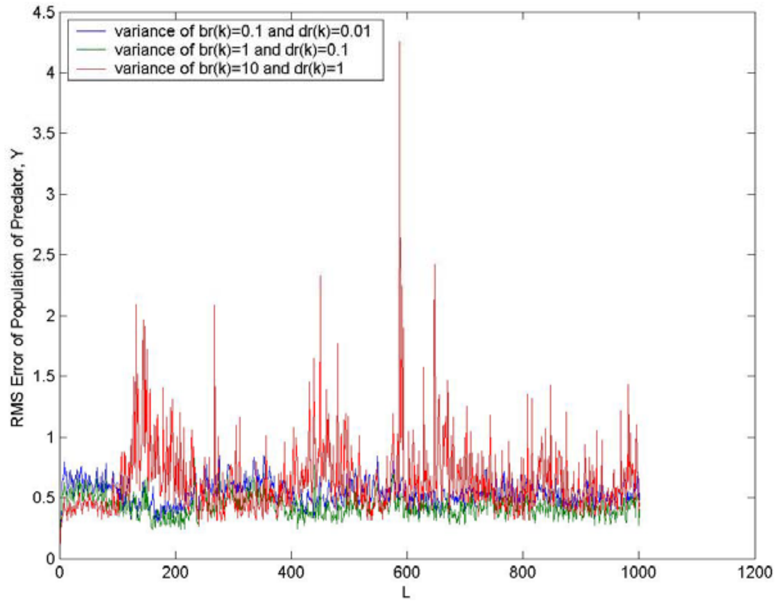

Figure 16 *The Plot of RMS Error of Y vs. L, for Various Birth and Death Rate Variances*

Some numerical data of the graphics above are the given in the following table:

Table 6.*The numerical data of the RMS error for death and birth rate variances*

	var_{br}= 0.01 var_{dr}= 0.01	var_{br}= 0.1 var_{dr}= 0.1	var_{br}= 1 var_{dr}= 1
Mean RMS Error of X	0.8591	1.0107	1.6626
Mean RMS Error of Y	0.4311	0.5378	0.6309
Variance of RMS Error of X	0.0388	0.0389	2.1911
Variance of RMS Error of Y	0.0081	0.0097	0.0897

6.2.4 The Effect of Observation Noise Variances ($\sigma_{v_1}{}^2$ & $\sigma_{v_2}{}^2$)

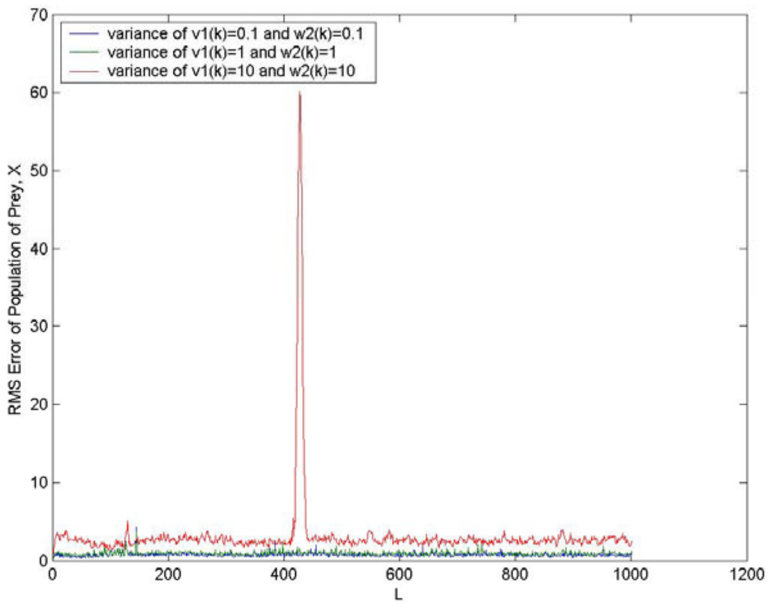

Figure 17 *The Plot of RMS Error of Y vs. L, for Various Observation Noise Variances*

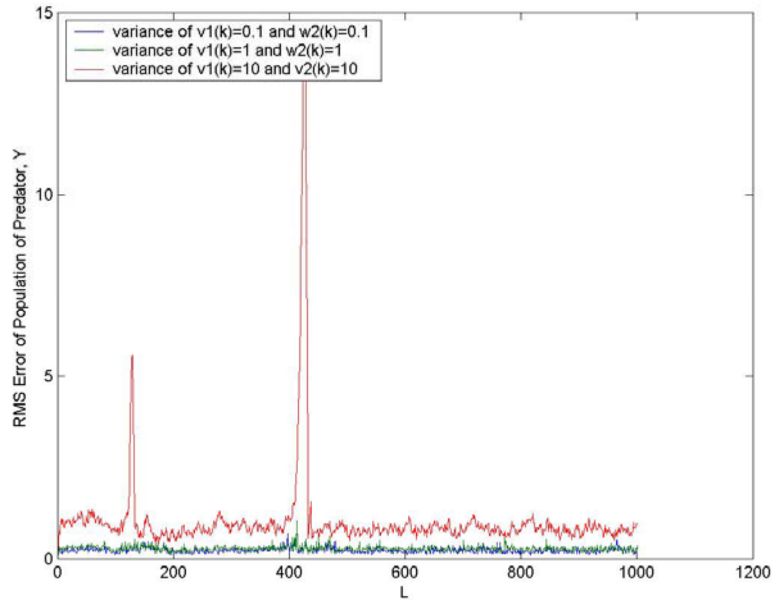

Figure 18 *The Plot of RMS Error of Y vs. L, for Various Observation Noise Variances*

Some numerical data of the graphics above are the given in the following table:

Table 7.*The numerical data of the RMS error for various observation noise variances*

	var_{v1}= 0.1 var_{v2}= 0.1	var_{v1}= 1 var_{v2}= 1	var_{v1}= 10 var_{v2}= 10
Mean RMS Error of X	0.7488	0.9078	3.1334
Mean RMS Error of Y	0.2281	0.2751	1.0223
Variance of RMS Error of X	0.0607	0.0687	23.0983
Variance of RMS Error of Y	0.6220	1.1324	1.6241

6.3 The Effects of Algorithm Parameters

6.3.1 The Effect of Maximum Number of States (*MaxState*)

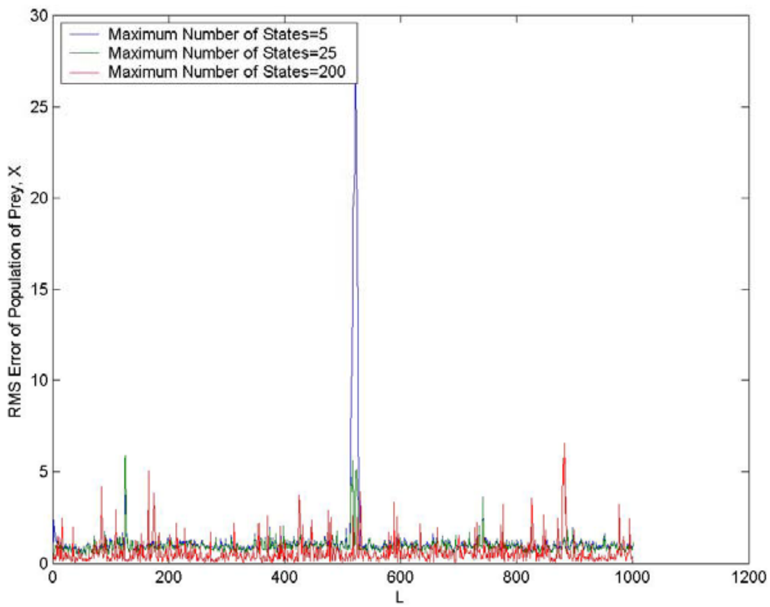

Figure 19 *The Plot of RMS Error of X vs. L, for The Effect Maximum Number of States*

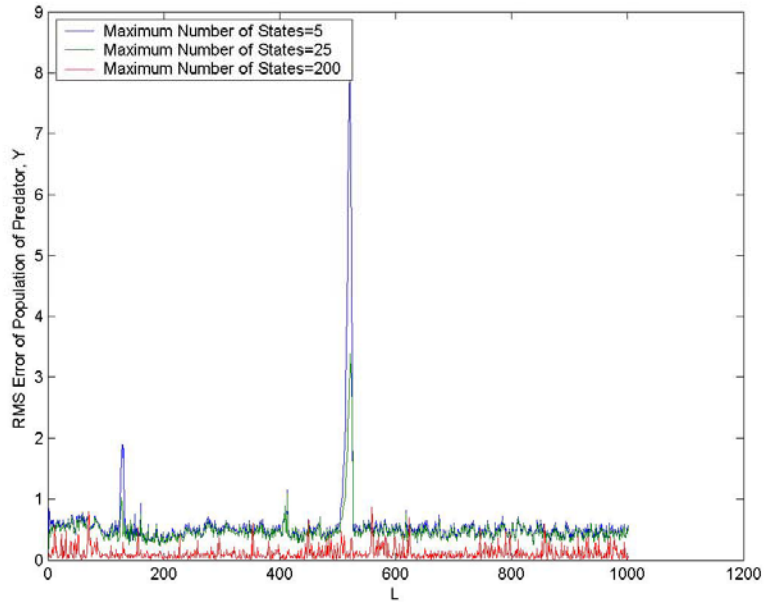

Figure 20 *The Plot of RMS Error of Y vs. L, for The Effect Maximum Number of States*

Some numerical data of the graphics above are the given in the following table:

Table 8.*The numerical data of the RMS error for various maximum number of states*

	MaxState=5	MaxState=25	MaxState=200
Mean RMS Error of X	1.2209	0.96147	0.59829
Mean RMS Error of Y	0.56783	0.48194	0.11175
Variance of RMS Error of X	4.1873	0.22863	0.45924
Variance of RMS Error of Y	0.36472	0.061338	0.011297

6.3.2 The Effect of Gate Size (*GS*)

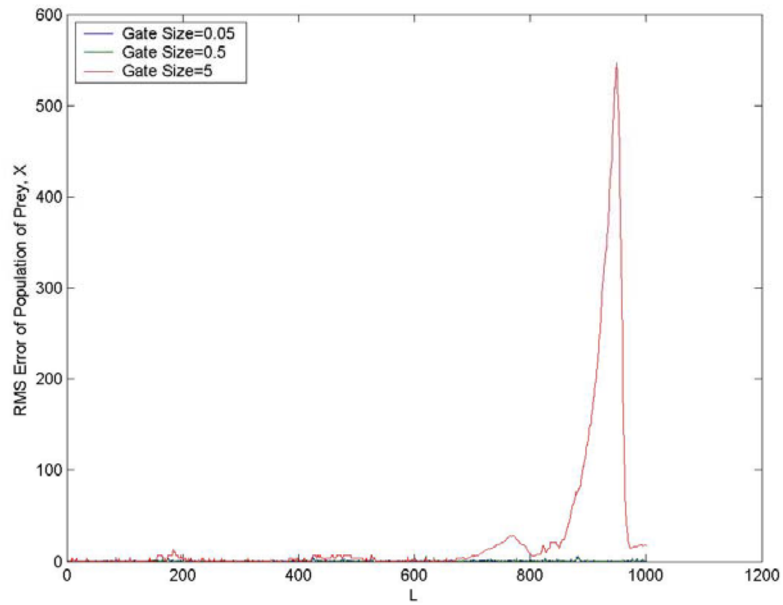

Figure 21 *The Plot of RMS Error of X vs. L, for The Effect of Gate Size*

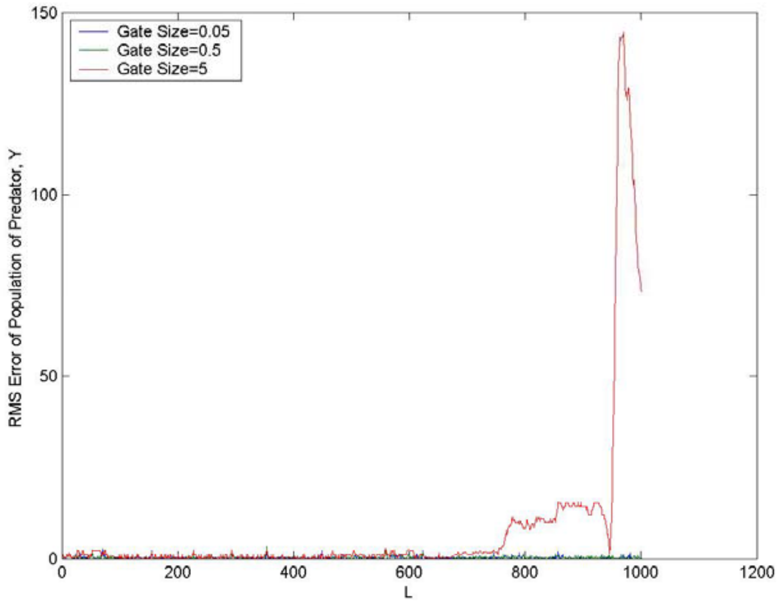

Figure 22 *The Plot of RMS Error of X vs. L, for The Effect of Gate Size*

Some numerical data of the graphics above are the given in the following table:

Table 9.*The numerical data of the RMS error for various gate sizes*

	GateSize= 0.05	GateSize= 0.5	GateSize= 5
Mean RMS Error of X	0.57638	0.64751	27.542
Mean RMS Error of Y	0.30093	0.3242	8.1364
Variance of RMS Error of X	0.49875	0.47615	6902.6
Variance of RMS Error of Y	0.11394	0.096012	584.08

6.3.3 The Effect of the Quantization Number of the Initial States (n_{X0} and n_{Y0})

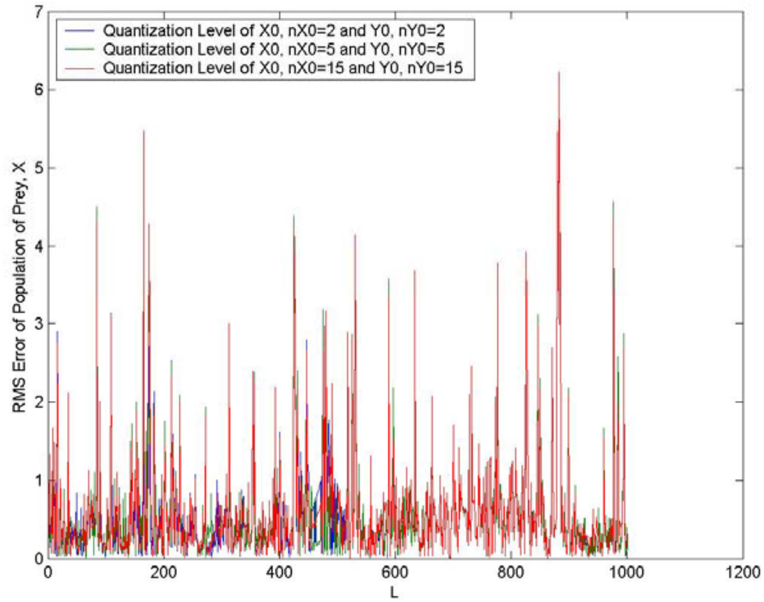

Figure 23. *The Plot of RMS Error of X vs. L, for number of Quantization Level Initial States X0 and Y0*

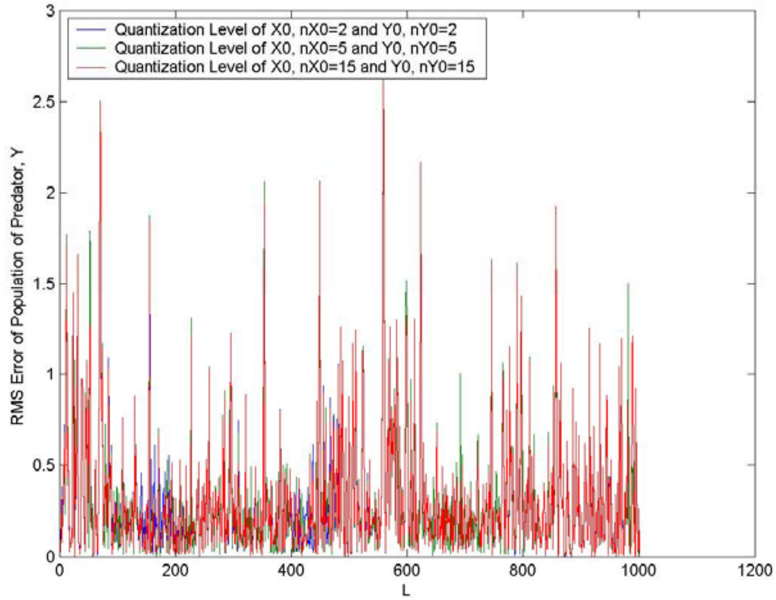

Figure 24 *The Plot of RMS Error of Y vs. L, for number of Quantization Level Initial States X0 and Y0*

Some numerical data of the graphics above are the given in the following table:

Table 10.*The numerical data of the RMS error for various quantization number of the initial states*

	n_{X0}= 2 n_{Y0}= 2	n_{X0}= 5 n_{Y0}= 5	n_{X0}= 15 n_{Y0}= 15
Mean RMS Error of X	0.57843	0.57337	0.58835
Mean RMS Error of Y	0.30218	0.3002	0.29871
Variance of RMS Error of X	0.48990	0.5234	0.53181
Variance of RMS Error of Y	0.12412	0.12509	0.12167

6.3.4 The Effect of the Quantization Number of the Disturbance Noises (n_{w1} and n_{w2})

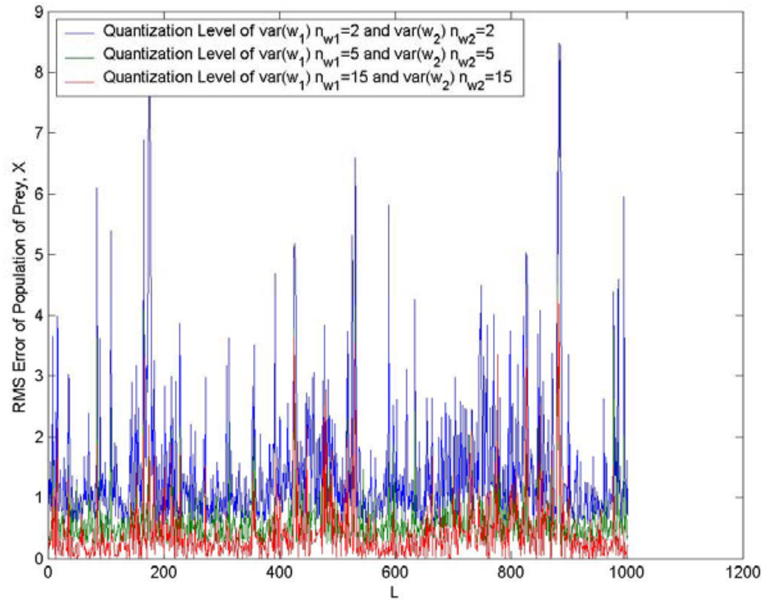

Figure 25 *The Plot of RMS Error of X vs. L, for Number of Quantization Level of Disturbance Noises* w_1 *and* w_2

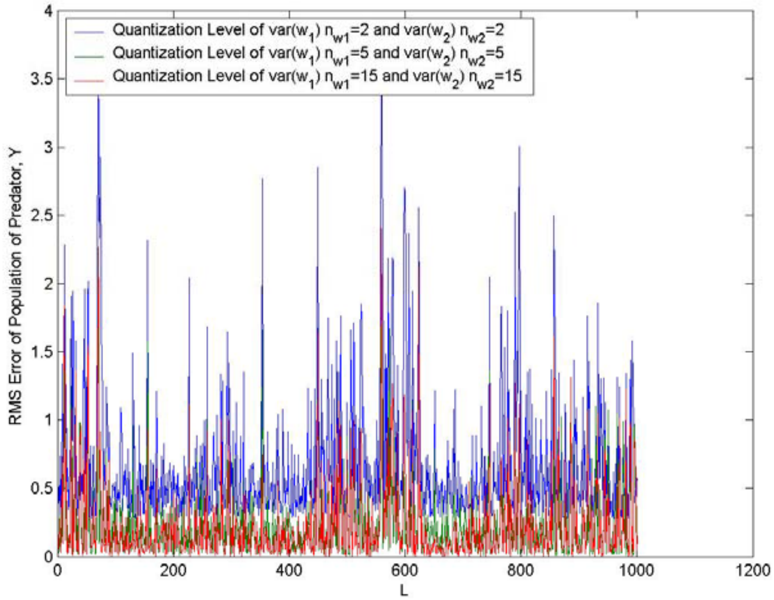

Figure 26 *The Plot of RMS Error of Y vs. L, for Number of Quantization Level of Disturbance Noises w_1 and w_2*

Some numerical data of the graphics above are the given in the following table:

Table 11.*The numerical data of the RMS error for various quantization number of the disturbance noise*

	n_{w1}= 2 n_{w2}= 2	n_{w1}= 5 n_{w2}= 5	n_{w1}= 15 n_{w2}= 15
Mean RMS Error of X	1.4375	0.62056	0.41171
Mean RMS Error of Y	0.69046	0.24363	0.23521
Variance of RMS Error of X	1.1863	0.29344	0.26515
Variance of RMS Error of Y	0.24057	0.073297	0.097041

6.3.5 The Effect of the Quantization Number of the Death Rate and Birth Rate (n_{br} and n_{dr})

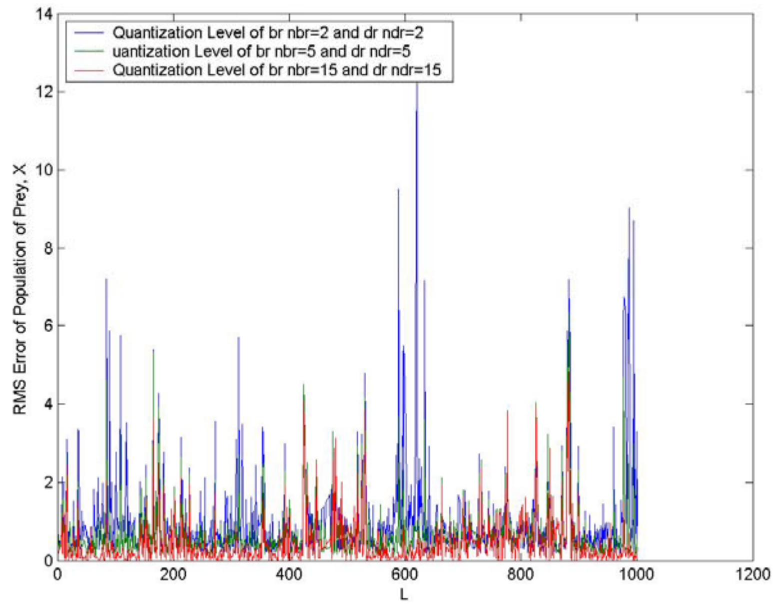

Figure 27 *The Plot of RMS Error of X vs. L, for Number of Quantization Level of Birth and Death Rates*

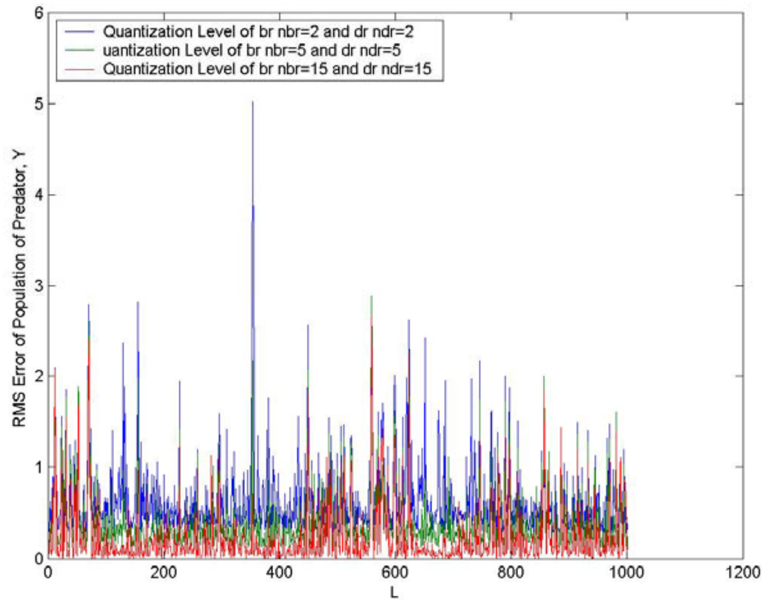

Figure 28 *The Plot of RMS Error of Y vs. L, for Number of Quantization Level of Birth and Death Rates*

Some numerical data of the graphics above are the given in the following table:

Table 12.*The numerical data of the RMS error for various quantization number of the birth and death rates*

	n_{br}= 2 n_{dr}= 2	n_{br}= 5 n_{dr}= 5	n_{br}= 15 n_{dr}= 15
Mean RMS Error of X	1.0244	0.69216	0.43035
Mean RMS Error of Y	0.64057	0.41093	0.23504
Variance of RMS Error of X	1.5092	0.5234	0.35239
Variance of RMS Error of Y	0.18524	0.12509	0.11773

6.4 Comments on Simulation Results

As it can be seen from the simulation results the variance of the RMS error is smaller than both the disturbance and observation noise variances.

For the effects of the model parameters, first of all, it can be seen from the Figure 11 and Figure 12 that the error does not change with respect to the initial state variance; therefore, we can say that the system is not sensitive to initial states.

Secondly, it can be seen from the figures 13 and 14 that, both the error mean and variance changes significantly by the change in disturbance noise variance which is expected. If the system is not disturbed much, the resultant error performance would be better.

Thirdly, when the birth and death rate variance increase, the error variance increases, on the other hand, when the birth and date rate variance decrease, the error variance also decreases.

Fourthly, it can be seen from the figures 17 and 18 that when the observation noise variances $\sigma_{v_1}^2$ and $\sigma_{v_2}^2$ decrease, this improves the estimation performance both in the mean of the RMS error and the variance of the RMS error. The theoretical expectations satisfied via the simulation results.

The effect of ODSA parameters on the estimation performance is as follows. While observing the effects of the ODSA parameters, one would encounter a compromise between the error performance and the time consumption. In section 6.3 five ODSA parameters are investigated via simulations, namely maximum number of states (MaxState), gate size (GS), number of quantization levels for initial states (nX0 and nY0), number of quantization levels for disturbance noise (nw1 and nw2) and number of quantization levels for birth and death rates (nbr and ndr).

First of all, as it can be seen from the figures 19, 20 and the Table 8, when MaxState increases, both RMS error variance and mean decreases. Note that the error decrease between MaxState=5 and MaxState=25 is more than the one between MaxState=25 and MaxState=200. This is because after a certain point, increase in the number of states does not affect the error performance significantly; however, increase in the number of states yields very high increase in the computation time. That is, one sample run takes about 2 minutes when MaxState=25 however it reaches 45 minutes for MaxState=200. Therefore, maximum number of states can be adjusted according to the allowed error mean and variance.

Secondly, Figure 21 and Figure 22 show the RMS error performance of the algorithm for various gate sizes. The error performance decreases with the increase in the gate size however, the enhancement in the performance is not too large. On the other hand, if gate size (GS) is further increased, the probability of estimate diverging increases. For instance, when the GS is larger than 3, e.g., 5, the algorithm diverges for the parameters chosen in pages 45-46 of Chapter 6. Besides, increase in the GS yields decrease in the computation time, which is expected.

Thirdly, the increase in the number of quantization level for initial states X0 and Y0 do not affect the performance of the algorithm, since number of quantization points for initial state X0, *nX0*, and of quantization points for initial state Y0, *nY0*, do not affect the estimation performance for large values of L. Since in Stochastic Discrete Lotka Volterra Model, L is in the order of thousands, resultant error performances do not change with the number of quantization levels for initial states X0 and Y0.

Fourthly, as it is seen in the Figure 25 and Figure 26, the increase in the number of quantization levels for disturbance noises w_1 and w_2 produces better performance in the RMS error mean and variance.

Finally, the number of quantization levels for birth and death rate affects the error performance as seen in the figures 27, 28 and the Table 12.

Briefly, ODSA parameters affect the error performance of the algorithm. The ODSA parameters are usually related with the "resolution" of the algorithm. Therefore, increase in the resolution yields increase in the performance in worth of the computation time. Hence there is a compromise between the error and the computation time.

To sum up, the theoretical facts are verified via the simulation results.

CHAPTER 7

COMPARISON of ODSA with PARTICLE FILTER

7.1 Simulation Results for Particle Filter

Simulations of Particle Filter applied to Discrete Lotka-Volterra Model are done by using MATLAB®. In order to simulate the Discrete Lotka-Volterra Model, state and observation vectors should be generated. The noises of the model are generated at MATLAB® using *randn (.)* function of MATLAB® to generate Gaussian distributed random vectors. Starting point u_1 is drawn by using *rand (.)* function of MATLAB® to generate uniform distributed random number.

The state space expression of the model is given in the following:

$$X[k+1]=X[k]+brX[k]-a_1X[k]Y[k]+w_1[k] \tag{6.1}$$

$$Y[k+1]=Y[k]-drY[k]-a_2Y[k]X[k]+w_2[k] \tag{6.2}$$

The observation equation is,

$$Z[k]=\begin{bmatrix} X[k] \\ Y[k] \end{bmatrix}+\begin{bmatrix} v_1 k] \\ v_2[k] \end{bmatrix} \tag{6.3}$$

Unless otherwise stated, the model parameters are chosen as in the following:

$a_1=0.007;$

a_2=0.006;

$mean_{BR}$=0.7;

$mean_{DR}$=0.5;

$mean_{X0}$= 20;

$mean_{Y0}$= 25;

var_{BR}=0.1;

var_{DR}=0.1;

var_{w1}=4;

var_{w2}=4;

var_{v1}= 1;

var_{v2}= 4;

var_{X0}= 1;

var_{Y0}= 1;

The simulations were performed by the SIR Particle Filter with 500 runs and ODSA with 100 runs throughout this section. In next section, the performance of SIR Particle Filter is shown for various number of particles, N_s.

7.1.1 Simulation of Particle Filter with Ns=2

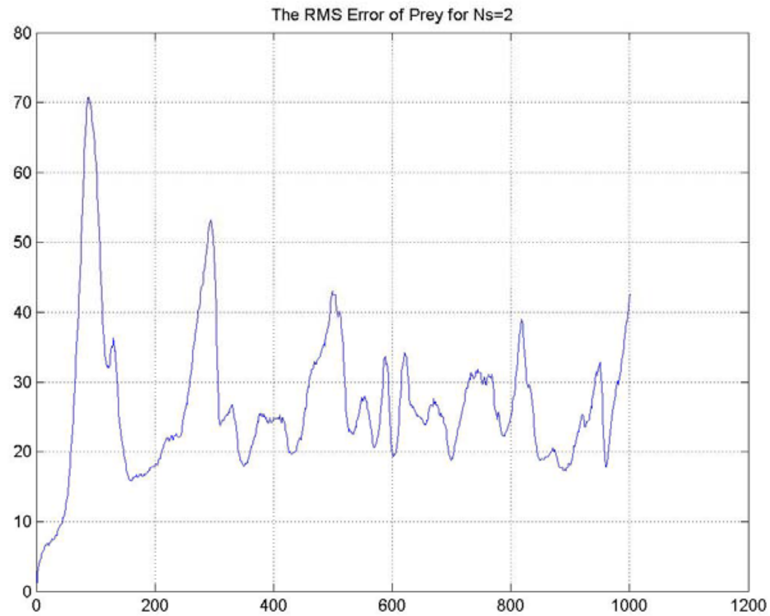

Figure 29. *The Plot of RMS Error of Prey, X, for Ns=2 using Particle Filter*

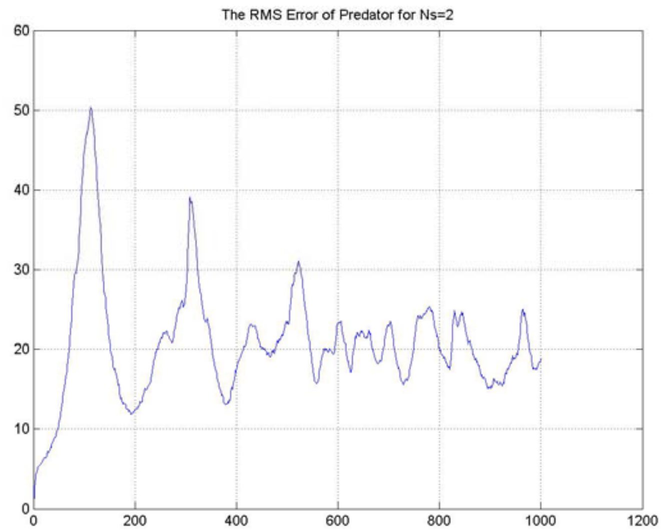

Figure 30 *The Plot of RMS Error of Predator, Y for Ns=2 using Particle Filter*

Mean of the RMS Error of Prey for Ns=2 is Mean_RMS=26.7345

Variance of the RMS Error of Prey for Ns=2 is Var_RMS=111.6645

Mean of the RMS Error of Predator for Ns=2 is Mean_RMS=20.779

Variance of the RMS Error of Predator for Ns=2 is Var_RMS=53.6546

7.1.2 Simulation of Particle Filter with Ns=5

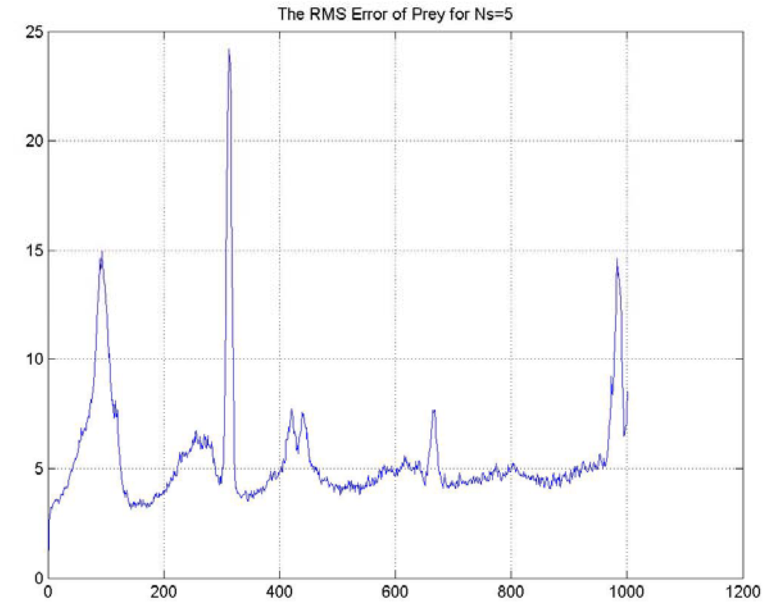

Figure 31. *The Plot of RMS Error of Prey, X, for Ns=5 using Particle Filter*

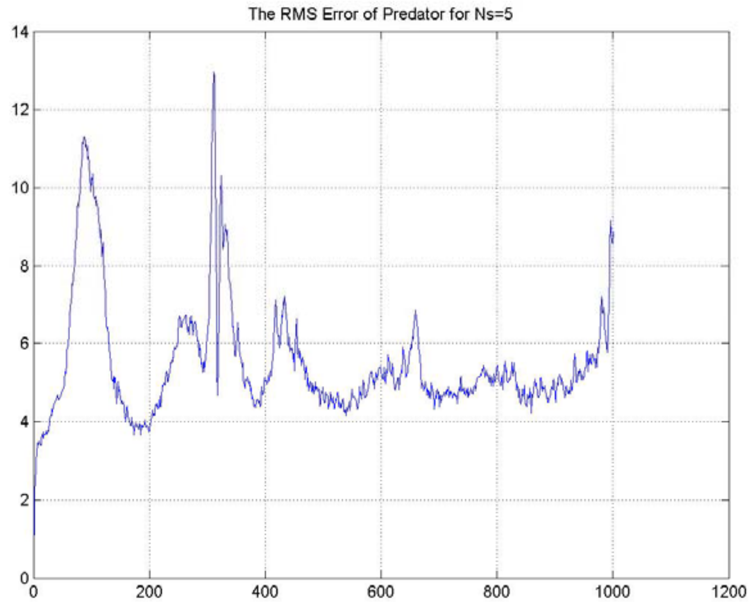

Figure 32. *The Plot of RMS Error of Predator, Y for Ns=5 using Particle Filter*

Mean of the RMS Error of Prey for Ns=5 is Mean_RMS=5.4072

Variance of the RMS Error of Prey for Ns=5 is Var_RMS=6.2831

Mean of the RMS Error of Predator for Ns=5 is Mean_RMS=5.5128

Variance of the RMS Error of Predator for Ns=5 is Var_RMS=2.3893

7.1.3 Simulation of Particle Filter with Ns=25

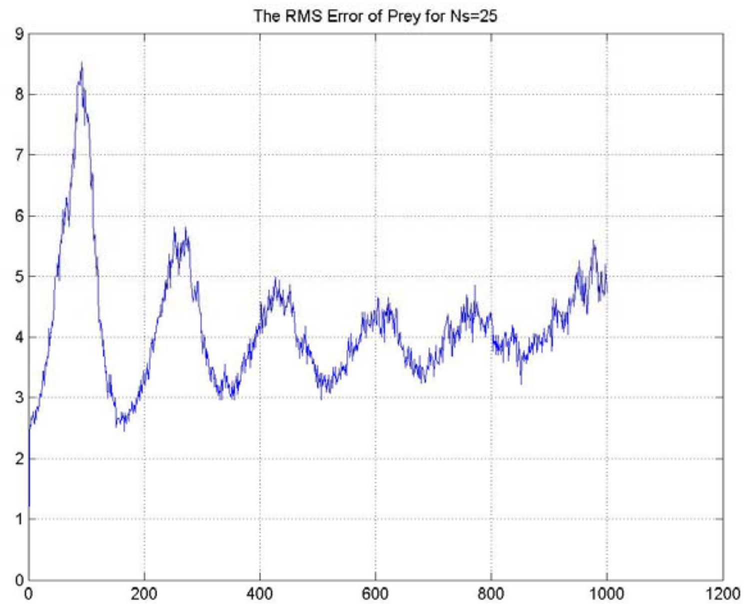

Figure 33. *The Plot of RMS Error of Prey, X, for Ns=25 using Particle Filter*

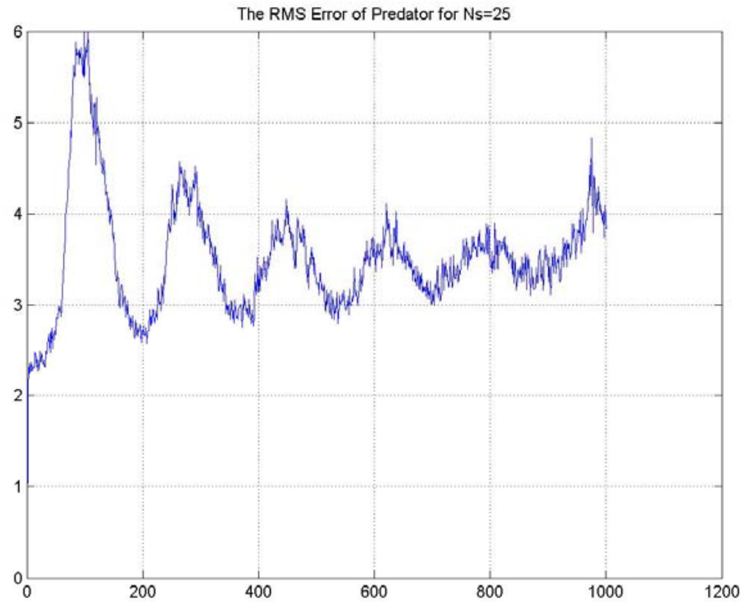

Figure 34. *Plot of RMS Error of Predator, Y for Ns=25 using Particle Filter*

Mean of the RMS Error of Prey for Ns=25 is Mean_RMS=4.1538

Variance of the RMS Error of Prey for Ns=25 is Var_RMS=0.98947

Mean of the RMS Error of Predator for Ns=25 is Mean_RMS=3.5199

Variance of the RMS Error of Predator for Ns=25 is Var_RMS=0.42396

7.1.4 Simulation of Particle Filter with Ns=100

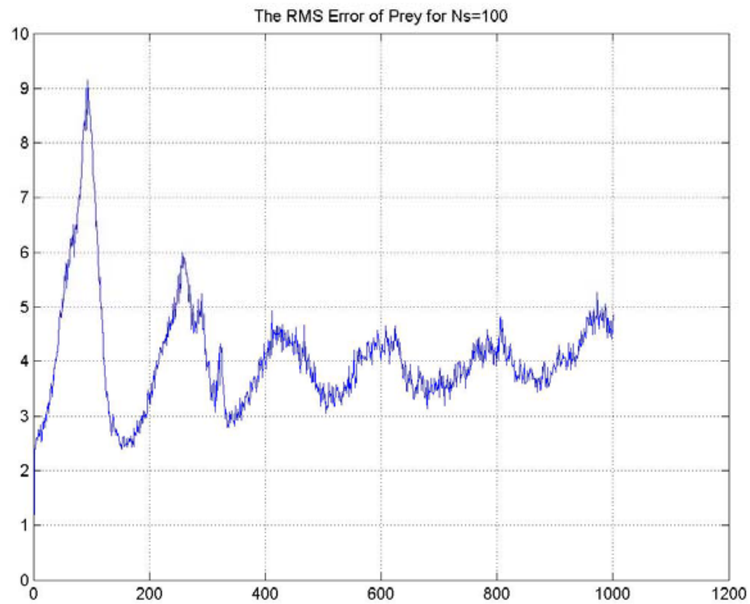

Figure 35. *The Plot of RMS Error of Prey, X, for Ns=100 using Particle Filter*

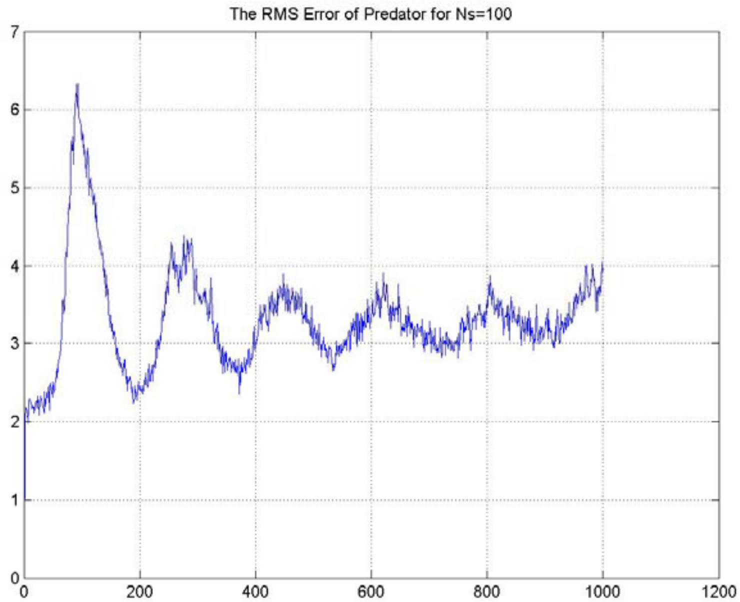

Figure 36. *Plot of RMS Error of Predator, Y for Ns=100 using Particle Filter*

Mean of the RMS Error of Prey for Ns=100 is Mean_RMS=4.072

Variance of the RMS Error of Prey for Ns=100 is Var_RMS=1.0435

Mean of the RMS Error of Predator for Ns=100 is Mean_RMS=3.3129

Variance of the RMS Error of Predator for Ns=100 is Var_RMS=0.45157

7.1.5 Simulation of Particle Filter with Ns=500

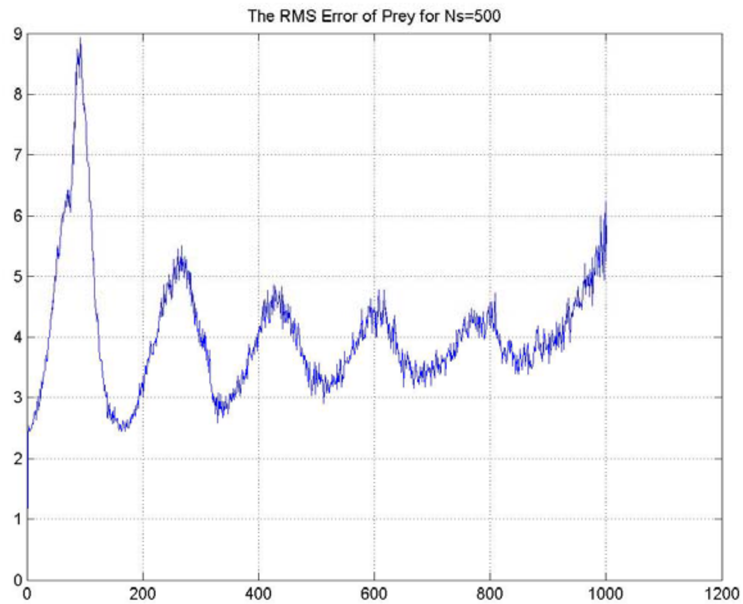

Figure 37. *The Plot of RMS Error of Prey, X, for Ns=500 using Particle Filter*

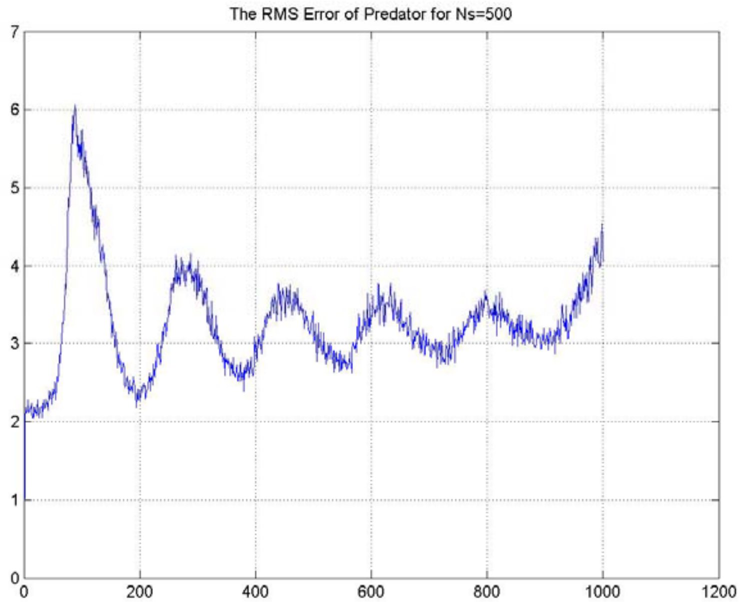

Figure 38. *Plot of RMS Error of Predator, Y for Ns=500 using Particle Filter*

Mean of the RMS Error of Prey for Ns=500 is Mean_RMS=4.0353

Variance of the RMS Error of Prey for Ns=500 is Var_RMS=1.0569

Mean of the RMS Error of Predator for Ns=500 is Mean_RMS=3.2591

Variance of the RMS Error of Predator for Ns=500 is Var_RMS=0.45044

To clarify the simulation results, it will be better to see in a single table. In Table 13, RMS Error between the real values of the states and the estimates are listed for varying N_s values.

Table 13.*Particle Filter Simulation Results, RMS Errors between the real values and the state varibles X and Y*

N_s	RMS Error Type	Error Value
2	Mean_RMS of X	26,7345
5	Mean_RMS of X	5,4072
25	Mean_RMS of X	4,1538
100	Mean_RMS of X	4,072
500	Mean_RMS of X	4,0353

N_s	RMS Error Type	Error Value
2	Mean_RMS of Y	20,779
5	Mean_RMS of Y	5,5128
25	Mean_RMS of Y	3,5199
100	Mean_RMS of Y	3,3129
500	Mean_RMS of Y	3,2591

2	Var_RMS of X	111,6645
5	Var_RMS of X	6,2831
25	Var_RMS of X	1,08947
100	Var_RMS of X	1,0435
500	Var_RMS of X	1,0569

2	Var_RMS of Y	53,6546
5	Var_RMS of Y	2,3893
25	Var_RMS of Y	0,45396
100	Var_RMS of Y	0,45157
500	Var_RMS of Y	0,45044

It can be seen from the Table 13 that, as N_s increases, the mean and the variance of the error decreases which is expected. On the other hand, the decrease between N_s=2 and N_s=5 is very large with respect to the others. As N_s gets larger, the amount of decrease diminishes. Therefore, optimum number of particles that should be used in Stochastic Discrete Lotka Volterra Model is N_s=25. Although for larger values of N_s, the error decreases, the computation time increases gradually. Since the error enhancement between N_s=25 and N_s=500 is not more than 1%, to avoid large computation time, N_s=25 will be best choice of Number of Particles.

7.2 Comparison of ODSA with Particle Filters

7.2.1 Comparison for MaxState=25 and Ns=25

In Figure 39, RMS errors of Prey, X, is seen both using ODSA and Particle Filter (PF) algorithm. It can be clearly seen from the figure that the ODSA algorithm gives better error performance with respect to the PF algorithm. In Figure 40, RMS error of Prey, Y, is seen. Again error performance of ODSA is better than the error performance of PF Algorithm.

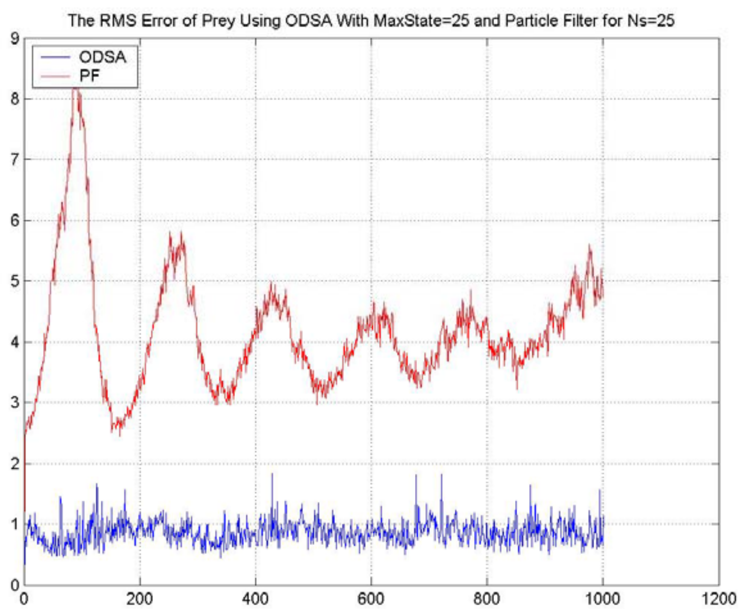

Figure 39 *RMS Error of Prey, X, Using ODSA with MaxState=25 and Particle Filter for Ns=25*

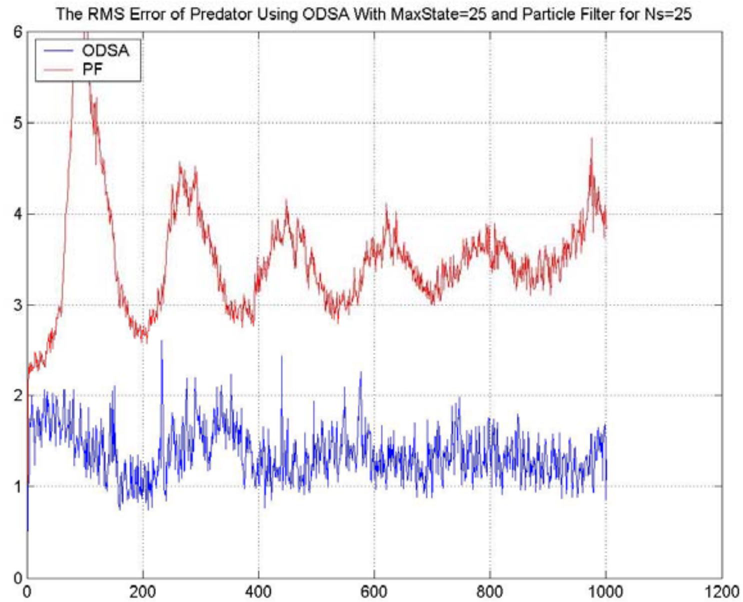

Figure 40. *RMS Error of Predator, Y, Using ODSA with MaxState=25 and Particle Filter for Ns=25*

To see the error difference clearly, to make a closer view to the Figure 39, the iteration numbers between 250 and 265 is chosen. Than the resultant plot is as follows:

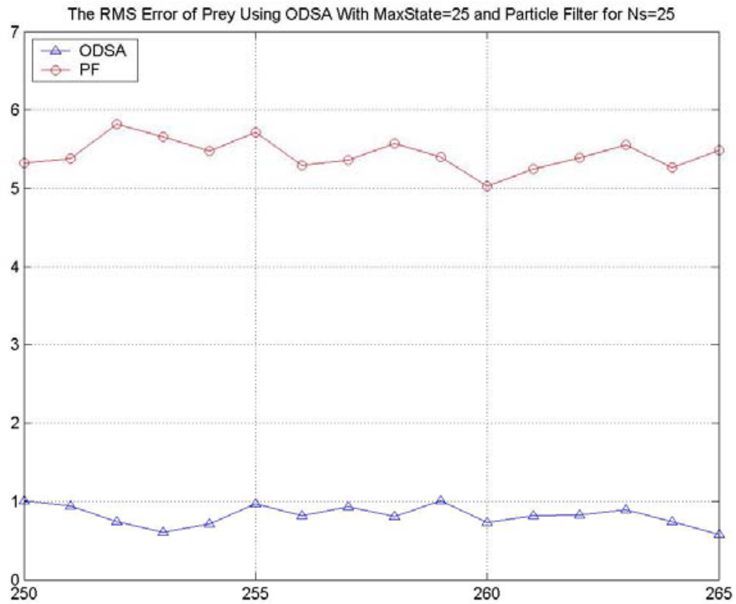

Figure 41. *The RMS Error of Prey Using ODSA with MaxState=25 and Particle Filter for Ns=25 Closer View for iteration number [250-265]*

The closer view to the Figure40 is I the following:

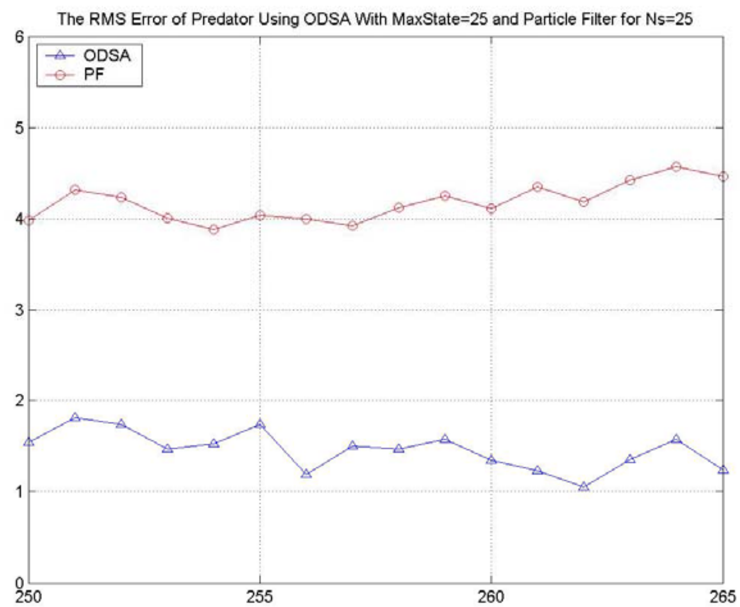

Figure 42. *The RMS Error of Predator Using ODSA with MaxState=25 and Particle Filter for Ns=25 Closer View for iteration number [250-265]*

The following table summarizes the comparison between ODSA and Particle Filter Algorithm for error and time performance.

Table 14.*The Comparison between ODSA and PArticle Filter Algorithm for Error and Time Performance*

Ns=25	ODSA	Particle Filter
Mean of RMS Error of X	0.8591	4.1538
Variance of RMS Error of X	0.0389	0.98947
Mean of RMS Error of Y	1.3640	3.5199
Variance of RMS Error of Y	0.0815	0.42396
Total Program Runtime	74.8535 secs	0.9832 secs
Runtime of single step	0.0749 secs	0.0009832 secs

7.2.2 Comparison for MaxState=5 and Ns=5

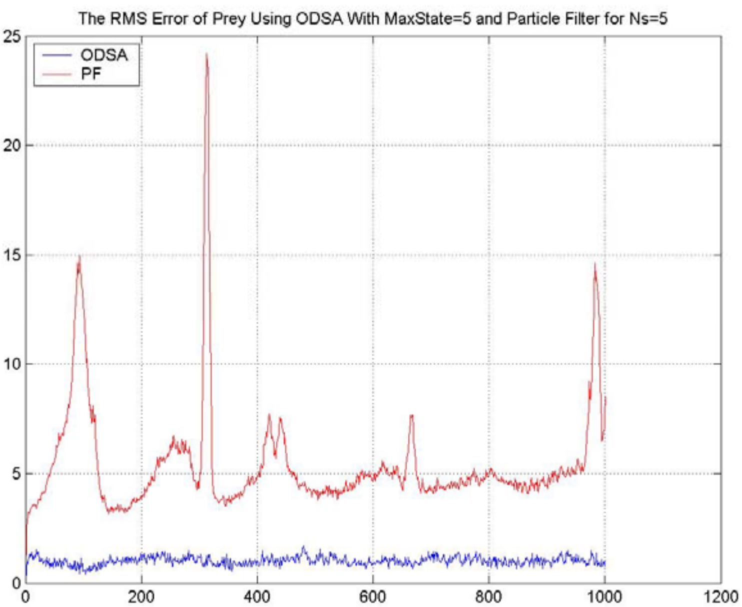

Figure 43. *The RMS Error of Prey Using ODSA with MaxState=5 and Particle Filter for Ns=5*

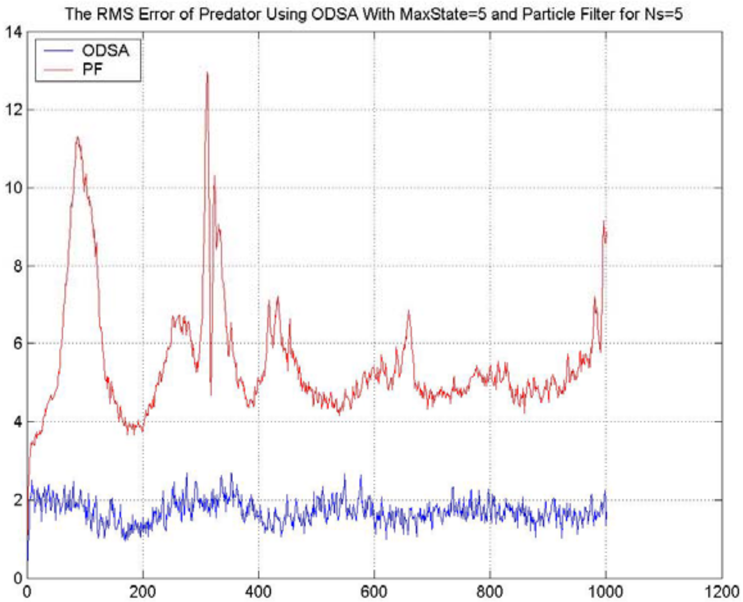

Figure 44. *The RMS Error of Predator Using ODSA with MaxState=5 and Particle Filter for Ns=5*

From the figures 43 and 44, it is seen that if MaxState and Ns are chosen 5 for ODSA and PF, respectively; again the error performance of ODSA is better than the PF algorithm. On the other hand, regarding the computation time, PF is faster than ODSA.

The following table summarizes the comparison between ODSA and Particle Filter Algorithm for error and time performance.

Table 15.*Performance Comparison between ODSA and Particle Filter Algorithm*

MaxState =5 and Ns=5	ODSA	Particle Filter
Mean of RMS Error of X	1.9432	5.4072
Variance of RMS Error of X	0.0234	6.2831
Mean of RMS Error of Y	2.03212	5.5128
Variance of RMS Error of Y	0.04645	2.3893
Total Program Runtime	14.18 secs	0.3318 secs
Runtime of single step	0.0142 secs	0.00033183 secs

CHAPTER 8

CONCLUSIONS

In this thesis, ODSA[1] and Particle Filter [32] algorithms are compared. The methodologies of both algorithms explained and some simulations are performed to see the performance of these estimation algorithms. The effects of the model parameters and algorithm parameters are investigated via simulations.

ODSA is based on Viterbi algorithm, a trellis diagram is obtained by reducing the population size of two species to a finite state model. Finite state model is obtained by using quantized state vector. The state vector is estimated using the observation model by finding the most probable path along the trellis diagram.

The performance of the ODSA depends on both model and algorithm parameters. Firstly, regarding the model parameters, noise variances negatively affect the performance; that is, larger noise variance gives worse error performance. Secondly, regarding the algorithm parameters, increase in quantization numbers of noise vectors and maximum number of states enhances the performance in worth of runtime performance. Briefly, using ODSA the error performance can be adjusted by tuning the algorithm parameters.

Particle Filter is sequential Monte Carlo methods that represent the required posterior density function by a set of random samples with associated weights and to compute the estimates based on these samples and weights. SIR Algorithm is presented to increase the performance of Particle Filter algorithm. SIR uses resampling to take 'better' samples while approximating the pdf.

The performances of the both algorithms are investigated via simulations. The investigation is done mainly on error and runtime performances. The simulation results showed that, ODSA is better on error performance on the contrary Particle Filter algorithm is better on runtime performance.

At the end, since there are very powerful computers developed, the computation time will not be a problem. Moreover while parallel processing considered, the computation time can be decreased further.

REFERENCES

1. K. Demirbaş, *"Target tracking in the presence of interference,"* Ph.D. Thesis, University of California, Los Angeles, 1981.

2. Lotka, A. J., Elements of Physical Biology, Williams & Wilkins Co., Baltimore, 1925;Volterra, V. Variazioni e fluttuazioni del numero d'individui in specie animali con-viventi. Mem. R. Accad. Naz. dei Lincei. Ser. VI, Vol. 21926; presented in many books, cf. pages 409-448 in Chapman, R. N., Animal Ecology, McGraw-Hill, New York, 1931.

3. G. D. Forney, JR., Inf. Control 25:222-247, 1974

4. A. J. Viterbi and J. K. Omura, "Principles of Digital Communication and Coding," McGraw-Hill, New York, 1979

5. E. Gazioğlu, Target Tracking With Input Estimation, M.S. Thesis, Middle East Technical University, Ankara, 2005

6. A.A.Berryman, The Origins and Evolution of Predator-Prey Theory, The Ecological Society of America, Ecology 73(5). 1992. pp. 1530-1535

7. Dubois Daniel M., Computing Anticipatory Systems with Incursion and Hyperincursion, Computing Anticipatory Systems: CASYS - First International Conference, Published by The American Institute of Physics, AIP Conference Proceedings 437, pp. 3-29, 1998.

8. R. Murray, Lecture Notes On Principals of Feedback and Control, Fall 2002, California Institute of Technology, Pasadena

9. Krebs, C.J., 1994. Ecology—The Experimental Analysis of Distribution and Abundance, 4th ed. Harper Collins College Publishers, New York, p. 801.

10. Alocock, J., 1989. Animal Behavior, 4th ed. Sinauer Associates INC, Sunderland, Massachusetts, p. 596.

11. Paine, R.T., 1966. Food web complexity and species diversity. Am. Naturalist 100, 65–75.

12. May, R.M., 1973. Stability and Complexity in Model Ecosystem. Princeton University Press, Princeton, NJ.

13. Renshaw, E., 1991. Modeling Biological Populations in Space and Time. Cambridge University Press, Cambridge. pp. 137–139.

14. Zhang, D.Y., Hanski, I., 1998. Sexual reproduction and stable and stable coexistence of identical competitors. J. Theor. Biol. 193, 465–473.

15. Seong-Joon Lee, D.J. Lee and H.S. Oh, Technological Forecasting At The Korean Stock Market: A Dynamic Competition Analysis Using Lotka–Volterra Model, Technological Forecasting and Social Change, Volume 72, Issue 8, October 2005, Pages 1044-1057

16. Yoram Louzoun, Sorin Solomon; Volatility driven market in a generalized Lotka–Voltera formalism,Physica A 302 (2001) pp220–233

17. Ronald A. Francisco, American Journal of Political Science > Vol. 40, No. 4 (Nov., 1996), pp. 1179-1204

18. Edward A. McGehee and Enrique Peacock-López, Turing patterns in a modified Lotka–Volterra model, Physics Letters A, Volume 342, Issues 1-2, 4 July 2005, Pages 90-98

19. Fang Liu, Yi-zhi Guo and Yun-fei Li, Interactions of microorganisms during natural spoilage of pork at 5 °C, Journal of Food Engineering, Volume 72, Issue 1, January 2006, Pages 24-29

20. Yinan Tang and J.W. Zhang, A competition model for two CPU vendors, Physica A: Statistical and Theoretical Physics, Volume 348, 15 March 2005, Pages 465-480

21. P.H. Leslie, A stochastic model for studying the properties of certain biological systems by numerical methods, Biometrika 45 (1957) 16–31.

22. Johnson Holt and John Colvin; A differential equation model of the interaction between the migration of the Senegalese grasshopper, Oedaleus senegalensis, its predators, and a seasonal habitat, Ecological Modelling, Volume 101, Issues 2-3, 15 August 1997, Pages 185-193

23. P.A. Piana, L.C. Gomes and A.A. Agostin; Comparison of predator-prey interaction models for fish assemblages from the neotropical region Ecological Modelling, Volume 192, Issues 1-2, 15 February 2006, Pages 259-270

24. David Schley, C. Patrick Doncaster and Tim Sluckin; opulation models of sperm-dependent parthenogenesis, Journal of Theoretical Biology, Volume 229, Issue 4, 21 August 2004, Pages 559-572

25. P. AugerR. Bravo de la ParraE. ; Hawk-dove game and competition dynamics, Mathematical and Computer Modelling, Volume 27, Issue 4, February 1998, Pages 89-98

26. Lawrence J. De Chant and Chris J. De Chant; Development of an elementary quantitative competing species model: potential guidelines for exotic/weed plant species control and ecosystem restoration programs, Ecological Engineering, Volume 22, Issue 2, 1 April 2004, Pages 67-75

27. Mark Emmerson, T. Martijn Bezemer, MarkD. Hunter, T. Hefin Jones, GregoryJ. Masters and Nicole M. Van Dam; How does global change affect the strength of trophic interactions?, Basic and Applied Ecology, Volume 5, Issue 6, 13 December 2004, Pages 505-514

28. Claudia Neuhauser and Joseph E. Fargione; A mutualism-parasitism continuum model and its application to plant-mycorrhizae interactions, Ecological Modelling, Volume 177, Issues 3-4, 1 October 2004, Pages 337-352

29. J.M. Hammersley and K.W. Morton, “Poor man’s Monte Carlo,” J. Roy.Stat. Soc., Series B, vol. 16, no. 1, pp. 23-38, 1954.

30. M.N. Rosenbluth and A.W. Rosenbluth, “Monte Carlo calculation of the average extension of molecular chains,” J. Chem. Phys., vol. 23, no. 2, pp. 356-359, 1956.

31. N.J. Gordon, D.J. Salmond, and A.F.M. Smith, “Novel approach to nonlinear/ non-Gaussian Bayesian state estimation,” IEE Proc.-F, vol. 140, no. 2, pp. 107-113, 1993.

32. M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” IEEE Trans. Signal Processing, vol. 50, pp. 174-188, Feb. 2002.

33. Petar M. Djuric, Jayesh H. Kotecha, Jianqiu Zhang, Yufei Huang, Tadesse Ghirmai, Mónica F. Bugallo, Joaquín Míguez, IEEE SIGNAL PROCESSING MAGAZINE pp. 19-38, September, 2003

34. Özdemir, Salih Zeki, Proposal of two system modeling practices : "fuzzy discrete event dynamical systems and fuzzy differential game theory".Thesis : For the degree of master of science in the Department of Electrical and Electronics Engineering., METU, 2001

APPENDIX A

+APPROXIMATION OF A CONTINUOUS RANDOM VARIABLE WITH A DISCRETE RANDOM VARIABLE UP TO 50 POSSIBLE VALUES

Possible values of the discrete random variable approximating the Gaussian random variable with zero mean and unity variance (y values) [5]*:*

N	*y value*									
1	0									
2	-0.675	0.675								
3	-1.0052	0	1.0052							
4	-1.2177	-0.3546	0.3546	1.2177						
5	-1.3767	-0.592	0	0.592	1.3767					
6	-1.4992	-0.7678	-0.2419	0.2419	0.7678	1.4992				
7	-1.6027	-0.9077	-0.4242	0	0.4242	0.9077	1.6027			
8	-1.6897	-1.0226	-0.5694	-0.1839	0.1839	0.5694	1.0226	1.6897		
9	-1.7644	-1.1198	-0.6896	-0.3315	0	0.3315	0.6896	1.1198	1.7644	
10	-1.8178	-1.1985	-0.7888	-0.4527	-0.1479	0.1479	0.4527	0.7888	1.1985	1.8178
11	-1.8799	-1.2737	-0.8779	-0.5575	-0.2716	0	0.2716	0.5575	0.8779	1.2737
	1.8799									
12	-1.9282	-1.3373	-0.9545	-0.6476	-0.377	-0.1239	0.1239	0.377	0.6476	0.9545
	1.3373	1.9282								
13	-1.9714	-1.3942	-1.0226	-0.727	-0.4688	-0.2301	0	0.2301	0.4688	0.727
	1.0226	1.3942	1.9714							
14	-2.0218	-1.4507	-1.0868	-0.7997	-0.5511	-0.3235	-0.1067	0.1067	0.3235	0.5511
	0.7997	1.0868	1.4507	2.0218						
15	-2.0449	-1.4918	-1.1387	-0.8611	-0.622	-0.4047	-0.1996	0	0.1996	0.4047
	0.622	0.8611	1.1387	1.4918	2.0449					
16	-2.0966	-1.5435	-1.1948	-0.9227	-0.6899	-0.4798	-0.2831	-0.0936	0.0936	0.2831
	0.4798	0.6899	0.9227	1.1948	1.5435	2.0966				
17	-2.1372	-1.5879	-1.2443	-0.9777	-0.7509	-0.5474	-0.3581	-0.1771	0	0.1771
	0.3581	0.5474	0.7509	0.9777	1.2443	1.5879	2.1372			
18	-2.1569	-1.6206	-1.2846	-1.0245	-0.804	-0.607	-0.4247	-0.2515	-0.0833	0.0833
	0.2515	0.4247	0.607	0.804	1.0245	1.2846	1.6206	2.1569		
19	-2.196	-1.6609	-1.3285	-1.0725	-0.8564	-0.6642	-0.4872	-0.3199	-0.1585	0
	0.1585	0.3199	0.4872	0.6642	0.8564	1.0725	1.3285	1.6609	2.196	
20	-2.2125	-1.6894	-1.3636	-1.113	-0.902	-0.7149	-0.5432	-0.3816	-0.2265	-0.0751
	0.0751	0.2265	0.3816	0.5432	0.7149	0.902	1.113	1.3636	1.6894	2.2125
21	-2.2538	-1.7277	-1.4036	-1.1557	-0.9479	-0.7644	-0.5967	-0.4396	-0.2895	-0.1437
	0	0.1437	0.2895	0.4396	0.5967	0.7644	0.9479	1.1557	1.4036	1.7277
	2.2538									

22	-2.2838	-1.7598	-1.4388	-1.1941	-0.9896	-0.8096	-0.6457	-0.4927	-0.3471	-0.2064	
	-0.0685	0.0685	0.2064	0.3471	0.4927	0.6457	0.8096	0.9896	1.1941	1.4388	
	1.7598	2.2838									
23	-2.2808	-1.7756	-1.4627	-1.2237	-1.024	-0.8484	-0.6888	-0.5402	-0.3992	-0.2634	
	-0.1309	0	0.1309	0.2634	0.3992	0.5402	0.6888	0.8484	1.024	1.2237	
	1.4627	1.7756	2.2808								
24	-2.2909	-1.7968	-1.4894	-1.2545	-1.0584	-0.8863	-0.7302	-0.5852	-0.4481	-0.3165	
	-0.1885	-0.0626	0.0626	0.1885	0.3165	0.4481	0.5852	0.7302	0.8863	1.0584	
	1.2545	1.4894	1.7968	2.2909							
25	-2.3389	-1.8343	-1.5255	-1.2912	-1.0965	-0.9262	-0.7722	-0.6296	-0.4952	-0.3667	
	-0.2423	-0.1205	0	0.1205	0.2423	0.3667	0.4952	0.6296	0.7722	0.9262	
	1.0965	1.2912	1.5255	1.8343	2.3389						
26	-2.3622	-1.8601	-1.5539	-1.3221	-1.1299	-0.9622	-0.8109	-0.6711	-0.5396	-0.4141	
	-0.2929	-0.1746	-0.058	0.058	0.1746	0.2929	0.4141	0.5396	0.6711	0.8109	
	0.9622	1.1299	1.3221	1.5539	1.8601	2.3622					
27	-2.3997	-1.8914	-1.585	-1.3542	-1.1635	-0.9976	-0.8483	-0.7107	-0.5816	-0.4587	
	-0.3403	-0.2251	-0.112	0	0.112	0.2251	0.3403	0.4587	0.5816	0.7107	
	0.8483	0.9976	1.1635	1.3542	1.585	1.8914	2.3997				
28	-2.3823	-1.8963	-1.5983	-1.3728	-1.1862	-1.0238	-0.8777	-0.7432	-0.6172	-0.4975	
	-0.3825	-0.2709	-0.1616	-0.0537	0.0537	0.1616	0.2709	0.3825	0.4975	0.6172	
	0.7432	0.8777	1.0238	1.1862	1.3728	1.5983	1.8963	2.3823			
29	-2.3899	-1.9123	-1.6183	-1.3957	-1.2116	-1.0515	-0.9077	-0.7756	-0.6521	-0.535	
	-0.4227	-0.314	-0.2079	-0.1035	0	0.1035	0.2079	0.314	0.4227	0.535	
	0.6521	0.7756	0.9077	1.0515	1.2116	1.3957	1.6183	1.9123	2.3899		
30	-2.3969	-1.928	-1.6382	-1.4187	-1.2372	-1.0795	-0.938	-0.8081	-0.6868	-0.572	
	-0.4621	-0.3559	-0.2524	-0.1507	-0.0501	0.0501	0.1507	0.2524	0.3559	0.4621	
	0.572	0.6868	0.8081	0.938	1.0795	1.2372	1.4187	1.6382	1.928	2.3969	
31	-2.4309	-1.9555	-1.6649	-1.4458	-1.2652	-1.1087	-0.9686	-0.8403	-0.7208	-0.6079	
	-0.5	-0.3959	-0.2946	-0.1953	-0.0973	0	0.0973	0.1953	0.2946	0.3959	
	0.5	0.6079	0.7208	0.8403	0.9686	1.1087	1.2652	1.4458	1.6649	1.9555	
	2.4309										
32	-2.4455	-1.9739	-1.6857	-1.4686	-1.2898	-1.135	-0.9966	-0.87	-0.7522	-0.6411	
	-0.5351	-0.433	-0.3339	-0.237	-0.1416	-0.0471	0.0471	0.1416	0.237	0.3339	
	0.433	0.5351	0.6411	0.7522	0.87	0.9966	1.135	1.2898	1.4686	1.6857	
	1.9739	2.4455									
33	-2.4934	-2.0071	-1.7156	-1.4976	-1.3188	-1.1644	-1.0266	-0.9008	-0.784	-0.6741	
	-0.5695	-0.469	-0.3717	-0.2768	-0.1836	-0.0915	0	0.0915	0.1836	0.2768	
	0.3717	0.469	0.5695	0.6741	0.784	0.9008	1.0266	1.1644	1.3188	1.4976	
	1.7156	2.0071	2.4934								
34	-2.5056	-2.0233	-1.7342	-1.5182	-1.3412	-1.1885	-1.0524	-0.9283	-0.8132	-0.705	
	-0.6021	-0.5033	-0.4077	-0.3146	-0.2234	-0.1335	-0.0444	0.0444	0.1335	0.2234	
	0.3146	0.4077	0.5033	0.6021	0.705	0.8132	0.9283	1.0524	1.1885	1.3412	
	1.5182	1.7342	2.0233	2.5056							
35	-2.4801	-2.0215	-1.7407	-1.5295	-1.356	-1.2062	-1.0726	-0.9507	-0.8376	-0.705	
	-0.6303	-0.5335	-0.44	-0.3491	-0.2602	-0.1727	-0.0861	0	0.0861	0.1727	
	0.2602	0.3491	0.44	0.5335	0.6303	0.7313	0.8376	0.9507	1.0726	1.2062	
	1.356	1.5295	1.7407	2.0215	2.4801						
36	-2.5117	-2.0453	-1.763	-1.5516	-1.3784	-1.2292	-1.0964	-0.9755	-0.8636	-0.705	
	-0.659	-0.5637	-0.4718	-0.3826	-0.2955	-0.21	-0.1256	-0.0418	0.0418	0.1256	0.21
	0.2955	0.3826	0.4718	0.5637	0.659	0.7586	0.8636	0.9755	1.0964	1.2292	
	1.3784	1.5516	1.763	2.0453	2.5117						
37	-2.4969	-2.0475	-1.7715	-1.5639	-1.3935	-1.2466	-1.1158	-0.9967	-0.8865	-0.705	
	-0.6853	-0.5917	-0.5015	-0.414	-0.3287	-0.2451	-0.1627	-0.0811	0	0.0811	
	0.1627	0.2451	0.3287	0.414	0.5015	0.5917	0.6853	0.7832	0.8865	0.9967	
	1.1158	1.2466	1.3935	1.5639	1.7715	2.0475	2.4969				
38	-2.5497	-2.0816	-1.8007	-1.5912	-1.42	-1.2728	-1.1421	-1.0234	-0.9137	-0.705	
	-0.7138	-0.621	-0.5318	-0.4455	-0.3615	-0.2793	-0.1985	-0.1187	-0.0395	0.0395	
	0.1187	0.1985	0.2793	0.3615	0.4455	0.5318	0.621	0.7138	0.811	0.9137	
	1.0234	1.1421	1.2728	1.42	1.5912	1.8007	2.0816	2.5497			

39	-2.5824	-2.1058	-1.8233	-1.6136	-1.4427	-1.2961	-1.1661	-1.0481	-0.9392	-0.705
	-0.7412	-0.6495	-0.5614	-0.4762	-0.3934	-0.3125	-0.2331	-0.1548	-0.0772	0
	0.0772	0.1548	0.2331	0.3125	0.3934	0.4762	0.5614	0.6495	0.7412	0.8374
	0.9392	1.0481	1.1661	1.2961	1.4427	1.6136	1.8233	2.1058	2.5824	
40	-2.534	-2.0914	-1.8202	-1.6167	-1.45	-1.3066	-1.1793	-1.0637	-0.957	-0.705
	-0.7628	-0.6728	-0.5864	-0.5029	-0.4218	-0.3426	-0.2649	-0.1884	-0.1127	-0.0375
	0.0375	0.1127	0.1884	0.2649	0.3426	0.4218	0.5029	0.5864	0.6728	0.7628
	0.8572	0.957	1.0637	1.1793	1.3066	1.45	1.6167	1.8202	2.0914	2.534
41	-2.5632	-2.1134	-1.8407	-1.6369	-1.4704	-1.3274	-1.2006	-1.0856	-0.9795	-0.705
	-0.7869	-0.6979	-0.6126	-0.5303	-0.4504	-0.3725	-0.2962	-0.2211	-0.1469	-0.0733
	0	0.0733	0.1469	0.2211	0.2962	0.3725	0.4504	0.5303	0.6126	0.6979
	0.7869	0.8804	0.9795	1.0856	1.2006	1.3274	1.4704	1.6369	1.8407	2.1134
	2.5632									
42	-2.6205	-2.1483	-1.8696	-1.6634	-1.4959	-1.3526	-1.2259	-1.1112	-1.0056	-0.705
-0.8142	-0.7258	-0.6411	-0.5595	-0.4804	-0.4033	-0.3278	-0.2536	-0.1804	-0.1079	
-0.0359	0.0359	0.1079	0.1804	0.2536	0.3278	0.4033	0.4804	0.5595	0.6411	0.7258
	0.8142	0.9071	1.0056	1.1112	1.2259	1.3526	1.4959	1.6634	1.8696	2.1483
	2.6205									
43	-2.654	-2.1715	-1.8907	-1.684	-1.5165	-1.3734	-1.247	-1.1327	-1.0276	-0.705
-0.8375	-0.7499	-0.6661	-0.5854	-0.5072	-0.4311	-0.3567	-0.2837	-0.2118	-0.1407	
-0.0702	0	0.0702	0.1407	0.2118	0.2837	0.3567	0.4311	0.5072	0.5854	
0.6661	0.7499	0.8375	0.9297	1.0276	1.1327	1.247	1.3734	1.5165	1.684	
1.8907	2.1715	2.654								
44	-2.5842	-2.1461	-1.8793	-1.6799	-1.5171	-1.3774	-1.2537	-1.1417	-1.0386	-0.705
-0.852	-0.766	-0.6837	-0.6045	-0.5279	-0.4535	-0.3809	-0.3097	-0.2396	-0.1704	
-0.1019	-0.0339	0.0339	0.1019	0.1704	0.2396	0.3097	0.3809	0.4535	0.5279	0.6045
	0.6837	0.766	0.852	0.9425	1.0386	1.1417	1.2537	1.3774	1.5171	1.6799
	1.8793	2.1461	2.5842							
45	-2.591	-2.1561	-1.8909	-1.6927	-1.531	-1.3924	-1.2697	-1.1586	-1.0564	-0.705
	-0.8716	-0.7866	-0.7054	-0.6273	-0.5518	-0.4785	-0.407	-0.337	-0.2682	-0.2003
	-0.1331	-0.0664	0	0.0664	0.1331	0.2003	0.2682	0.337	0.407	0.4785
	0.5518	0.6273	0.7054	0.7866	0.8716	0.9612	1.0564	1.1586	1.2697	1.3924
	1.531	1.6927	1.8909	2.1561	2.591					
46	-2.5938	-2.1646	-1.9019	-1.7054	-1.5451	-1.4077	-1.2861	-1.1761	-1.075	-0.705
	-0.8924	-0.8084	-0.7281	-0.6509	-0.5763	-0.5039	-0.4333	-0.3642	-0.2964	-0.2296
	-0.1635	-0.0979	-0.0326	0.0326	0.0979	0.1635	0.2296	0.2964	0.3642	0.4333
	0.5039	0.5763	0.6509	0.7281	0.8084	0.8924	0.9809	1.075	1.1761	1.2861
	1.4077	1.5451	1.7054	1.9019	2.1646	2.5938				
47	-2.632	-2.1915	-1.926	-1.7285	-1.5678	-1.4303	-1.3088	-1.199	-1.0981	-0.705
	-0.9159	-0.8322	-0.7523	-0.6755	-0.6013	-0.5293	-0.4592	-0.3907	-0.3235	-0.2574
	-0.1922	-0.1277	-0.0637	0	0.0637	0.1277	0.1922	0.2574	0.3235	0.3907
	0.4592	0.5293	0.6013	0.6755	0.7523	0.8322	0.9159	1.0042	1.0981	1.199
	1.3088	1.4303	1.5678	1.7285	1.926	2.1915	2.632			
48	-2.643	-2.2031	-1.9384	-1.7417	-1.5818	-1.4451	-1.3244	-1.2154	-1.1153	-0.705
	-0.9347	-0.8518	-0.7727	-0.6968	-0.6236	-0.5526	-0.4835	-0.416	-0.3498	-0.2847
	-0.2205	-0.157	-0.094	-0.0313	0.0313	0.094	0.157	0.2205	0.2847	0.3498
	0.416	0.4835	0.5526	0.6236	0.6968	0.7727	0.8518	0.9347	1.0222	1.1153
	1.2154	1.3244	1.4451	1.5818	1.7417	1.9384	2.2031	2.643		
49	-2.6541	-2.2145	-1.9505	-1.7545	-1.5953	-1.4593	-1.3393	-1.231	-1.1316	-0.705
	-0.9524	-0.8702	-0.7918	-0.7166	-0.6441	-0.5739	-0.5057	-0.4392	-0.3741	-0.3101
	-0.247	-0.1846	-0.1227	-0.0612	0	0.0612	0.1227	0.1846	0.247	0.3101
	0.3741	0.4392	0.5057	0.5739	0.6441	0.7166	0.7918	0.8702	0.9524	1.0392
	1.1316	1.231	1.3393	1.4593	1.5953	1.7545	1.9505	2.2145	2.6541	
50	-2.6676	-2.2268	-1.9629	-1.7673	-1.6086	-1.4731	-1.3536	-1.2458	-1.1469	-0.705
	-0.9688	-0.8873	-0.8097	-0.7354	-0.6638	-0.5945	-0.5272	-0.4616	-0.3974	-0.3343
	-0.2721	-0.2107	-0.15	-0.0898	-0.0299	0.0299	0.0898	0.15	0.2107	0.2721
	0.3343	0.3974	0.4616	0.5272	0.5945	0.6638	0.7354	0.8097	0.8873	0.9688
	1.055	1.1469	1.2458	1.3536	1.4731	1.6086	1.7673	1.9629	2.2268	2.6676

Probabilities of the corresponding y values (p values):[1]

N	*p value*									
1	1									
2	0.5003	0.5003								
3	0.3148	0.3704	0.3148							
4	0.2225	0.2771	0.2771	0.2225						
5	0.1686	0.2167	0.2295	0.2167	0.1686					
6	0.1337	0.1751	0.1911	0.1911	0.1751	0.1337				
7	0.1095	0.1455	0.1619	0.1667	0.1619	0.1455	0.1095			
8	0.0919	0.1235	0.1391	0.1459	0.1459	0.1391	0.1235	0.0919		
9	0.0786	0.1065	0.1211	0.1287	0.131	0.1287	0.1211	0.1065	0.0786	
10	0.0686	0.0931	0.1065	0.1141	0.1176	0.1176	0.1141	0.1065	0.0931	0.0686
11	0.0603	0.0824	0.0949	0.1023	0.1064	0.1077	0.1064	0.1023	0.0949	0.0824
	0.0603									
12	0.0537	0.0736	0.0851	0.0923	0.0966	0.0986	0.0986	0.0966	0.0923	0.0851
	0.0736	0.0537								
13	0.0483	0.0663	0.0769	0.0838	0.0882	0.0906	0.0914	0.0906	0.0882	0.0838
	0.0769	0.0663	0.0483							
14	0.0435	0.0602	0.0701	0.0767	0.081	0.0837	0.085	0.085	0.0837	0.081
	0.0767	0.0701	0.0602	0.0435						
15	0.0399	0.055	0.0641	0.0703	0.0745	0.0773	0.0788	0.0794	0.0788	0.0773
	0.0745	0.0703	0.0641	0.055	0.0399					
16	0.0363	0.0504	0.059	0.065	0.0691	0.072	0.0738	0.0746	0.0746	0.0738
	0.072	0.0691	0.065	0.059	0.0504	0.0363				
17	0.0333	0.0465	0.0546	0.0602	0.0643	0.0671	0.069	0.0701	0.0704	0.0701
	0.069	0.0671	0.0643	0.0602	0.0546	0.0465	0.0333			
18	0.031	0.0431	0.0507	0.056	0.0598	0.0626	0.0646	0.0658	0.0664	0.0664
	0.0658	0.0646	0.0626	0.0598	0.056	0.0507	0.0431	0.031		
19	0.0286	0.0401	0.0472	0.0523	0.056	0.0588	0.0608	0.0621	0.0629	0.0631
	0.0629	0.0621	0.0608	0.0588	0.056	0.0523	0.0472	0.0401	0.0286	
20	0.0268	0.0374	0.0441	0.0489	0.0525	0.0552	0.0572	0.0586	0.0595	0.0599
	0.0599	0.0595	0.0586	0.0572	0.0552	0.0525	0.0489	0.0441	0.0374	0.0268
21	0.0248	0.035	0.0414	0.046	0.0494	0.052	0.054	0.0555	0.0565	0.057
	0.0572	0.057	0.0565	0.0555	0.054	0.052	0.0494	0.046	0.0414	0.035
	0.0248									
22	0.0232	0.0328	0.0389	0.0433	0.0466	0.0492	0.0511	0.0526	0.0537	0.0543
	0.0546	0.0546	0.0543	0.0537	0.0526	0.0511	0.0492	0.0466	0.0433	0.0389
	0.0328	0.0232								
23	0.0222	0.031	0.0367	0.0408	0.044	0.0464	0.0483	0.0498	0.0509	0.0516
	0.052	0.0521	0.052	0.0516	0.0509	0.0498	0.0483	0.0464	0.044	0.0408
	0.0367	0.031	0.0222							
24	0.0211	0.0293	0.0347	0.0386	0.0416	0.044	0.0458	0.0473	0.0484	0.0492
	0.0497	0.0499	0.0499	0.0497	0.0492	0.0484	0.0473	0.0458	0.044	0.0416
	0.0386	0.0347	0.0293	0.0211						
25	0.0196	0.0277	0.0329	0.0366	0.0396	0.0419	0.0438	0.0452	0.0463	0.0471
	0.0476	0.0479	0.048	0.0479	0.0476	0.0471	0.0463	0.0452	0.0438	0.0419
	0.0396	0.0366	0.0329	0.0277	0.0196					
26	0.0185	0.0262	0.0311	0.0348	0.0376	0.0398	0.0416	0.0431	0.0442	0.0451
	0.0457	0.0461	0.0463	0.0463	0.0461	0.0457	0.0451	0.0442	0.0431	0.0416
	0.0398	0.0376	0.0348	0.0311	0.0262	0.0185				

[1] The maximum error of the sum of the *p* values is less than 0.001.

27	0.0174	0.0248	0.0296	0.0331	0.0358	0.038	0.0398	0.0412	0.0424	0.0433	
	0.0439	0.0444	0.0446	0.0446	0.0446	0.0444	0.0439	0.0433	0.0424	0.0412	
	0.0398	0.038	0.0358	0.0331	0.0296	0.0248	0.0174				
28	0.0169	0.0238	0.0283	0.0316	0.0342	0.0362	0.0379	0.0393	0.0404	0.0413	
	0.042	0.0424	0.0427	0.0428	0.0428	0.0427	0.0424	0.042	0.0413	0.0404	
	0.0393	0.0379	0.0362	0.0342	0.0316	0.0283	0.0238	0.0169			
29	0.0162	0.0227	0.027	0.0302	0.0327	0.0347	0.0363	0.0376	0.0387	0.0396	
	0.0403	0.0407	0.041	0.0412	0.0412	0.0412	0.041	0.0407	0.0403	0.0396	
	0.0387	0.0376	0.0363	0.0347	0.0327	0.0302	0.027	0.0227	0.0162		
30	0.0156	0.0217	0.0258	0.0288	0.0312	0.0331	0.0347	0.0361	0.0371	0.038	
	0.0387	0.0392	0.0396	0.0399	0.04	0.04	0.0399	0.0396	0.0392	0.0387	
	0.038	0.0371	0.0361	0.0347	0.0331	0.0312	0.0288	0.0258	0.0217	0.0156	
31	0.0147	0.0207	0.0247	0.0276	0.0299	0.0318	0.0334	0.0346	0.0357	0.0366	
	0.0373	0.0378	0.0383	0.0386	0.0388	0.0388	0.0388	0.0386	0.0383	0.0378	
	0.0373	0.0366	0.0357	0.0346	0.0334	0.0318	0.0299	0.0276	0.0247	0.0207	
	0.0147										
32	0.0141	0.0198	0.0236	0.0265	0.0287	0.0305	0.032	0.0333	0.0343	0.0352	
	0.0359	0.0365	0.0369	0.0373	0.0375	0.0376	0.0376	0.0375	0.0373	0.0369	
	0.0365	0.0359	0.0352	0.0343	0.0333	0.032	0.0305	0.0287	0.0265	0.0236	
	0.0198	0.0141									
33	0.0132	0.0189	0.0226	0.0254	0.0276	0.0294	0.0309	0.0322	0.0332	0.034	
	0.0347	0.0353	0.0357	0.0361	0.0363	0.0365	0.0365	0.0365	0.0363	0.0361	
	0.0357	0.0353	0.0347	0.034	0.0332	0.0322	0.0309	0.0294	0.0276	0.0254	
	0.0226	0.0189	0.0132								
34	0.0127	0.0181	0.0217	0.0244	0.0265	0.0283	0.0297	0.0309	0.0319	0.0328	
	0.0335	0.0341	0.0346	0.035	0.0352	0.0354	0.0354	0.0354	0.0354	0.0352	
	0.035	0.0346	0.0341	0.0335	0.0328	0.0319	0.0309	0.0297	0.0283	0.0265	
	0.0244	0.0217	0.0181	0.0127							
35	0.0125	0.0176	0.0209	0.0235	0.0255	0.0272	0.0286	0.0297	0.0308	0.0316	
	0.0323	0.0329	0.0334	0.0337	0.034	0.0342	0.0343	0.0343	0.0343	0.0342	
	0.034	0.0337	0.0334	0.0329	0.0323	0.0316	0.0308	0.0297	0.0286	0.0272	
	0.0255	0.0235	0.0209	0.0176	0.0125						
36	0.0119	0.0169	0.0202	0.0227	0.0247	0.0263	0.0276	0.0288	0.0297	0.0306	
	0.0312	0.0318	0.0323	0.0327	0.0329	0.0331	0.0333	0.0333	0.0333	0.0333	
	0.0331	0.0329	0.0327	0.0323	0.0318	0.0312	0.0306	0.0297	0.0288	0.0276	
	0.0263	0.0247	0.0227	0.0202	0.0169	0.0119					
37	0.0117	0.0164	0.0195	0.0219	0.0238	0.0253	0.0266	0.0278	0.0287	0.0295	
	0.0302	0.0307	0.0312	0.0316	0.0319	0.0321	0.0323	0.0323	0.0323	0.0323	
	0.0323	0.0321	0.0319	0.0316	0.0312	0.0307	0.0302	0.0295	0.0287	0.0278	
	0.0266	0.0253	0.0238	0.0219	0.0195	0.0164	0.0117				
38	0.011	0.0156	0.0187	0.0211	0.023	0.0245	0.0258	0.0269	0.0279	0.0286	
	0.0293	0.0299	0.0304	0.0307	0.031	0.0313	0.0314	0.0315	0.0315	0.0315	
	0.0315	0.0314	0.0313	0.031	0.0307	0.0304	0.0299	0.0293	0.0286	0.0279	
	0.0269	0.0258	0.0245	0.023	0.0211	0.0187	0.0156	0.011			
39	0.0104	0.015	0.018	0.0203	0.0222	0.0237	0.025	0.0261	0.027	0.0278	
	0.0284	0.029	0.0295	0.0299	0.0302	0.0304	0.0306	0.0307	0.0308	0.0308	
	0.0308	0.0307	0.0306	0.0304	0.0302	0.0299	0.0295	0.029	0.0284	0.0278	
	0.027	0.0261	0.025	0.0237	0.0222	0.0203	0.018	0.015	0.0104		
40	0.0105	0.0147	0.0175	0.0197	0.0214	0.0229	0.0241	0.0251	0.026	0.0268	
	0.0275	0.028	0.0285	0.0289	0.0292	0.0295	0.0297	0.0298	0.0299	0.0299	
	0.0299	0.0299	0.0298	0.0297	0.0295	0.0292	0.0289	0.0285	0.028	0.0275	
	0.0268	0.026	0.0251	0.0241	0.0229	0.0214	0.0197	0.0175	0.0147	0.0105	
41	0.01	0.0142	0.0169	0.0191	0.0208	0.0222	0.0234	0.0244	0.0253	0.026	
	0.0267	0.0272	0.0277	0.0281	0.0284	0.0287	0.0289	0.029	0.0291	0.0292	
	0.0292	0.0292	0.0291	0.029	0.0289	0.0287	0.0284	0.0281	0.0277	0.0272	
	0.0267	0.026	0.0253	0.0244	0.0234	0.0222	0.0208	0.0191	0.0169	0.0142	0.01

42	0.0094	0.0135	0.0163	0.0184	0.0201	0.0214	0.0226	0.0236	0.0245	0.0253
	0.0259	0.0265	0.027	0.0274	0.0277	0.028	0.0283	0.0284	0.0286	0.0286
	0.0286	0.0286	0.0286	0.0286	0.0284	0.0283	0.028	0.0277	0.0274	0.027
	0.0265	0.0259	0.0253	0.0245	0.0236	0.0226	0.0214	0.0201	0.0184	0.0163
	0.0135	0.0094								
43	0.0089	0.013	0.0157	0.0178	0.0194	0.0208	0.022	0.023	0.0238	0.0246
	0.0252	0.0258	0.0262	0.0267	0.0271	0.0273	0.0276	0.0277	0.0279	0.028
	0.028	0.028	0.028	0.028	0.0279	0.0277	0.0276	0.0273	0.0271	0.0267
	0.0262	0.0258	0.0252	0.0246	0.0238	0.023	0.022	0.0208	0.0194	0.0178
	0.0157	0.013	0.0089							
44	0.0092	0.0129	0.0154	0.0174	0.0189	0.0202	0.0213	0.0223	0.0231	0.0238
	0.0244	0.025	0.0255	0.0259	0.0262	0.0264	0.0267	0.0269	0.027	0.0271
	0.0271	0.027	0.027	0.0271	0.0271	0.027	0.0269	0.0267	0.0264	0.0262
	0.0259	0.0255	0.025	0.0244	0.0238	0.0231	0.0223	0.0213	0.0202	0.0189
	0.0174	0.0154	0.0129	0.0092						
45	0.0089	0.0126	0.015	0.0169	0.0184	0.0196	0.0207	0.0217	0.0225	0.0232
	0.0238	0.0243	0.0247	0.0252	0.0255	0.0257	0.026	0.0261	0.0263	0.0264
	0.0265	0.0265	0.0265	0.0265	0.0265	0.0264	0.0263	0.0261	0.026	0.0257
	0.0255	0.0252	0.0247	0.0243	0.0238	0.0232	0.0225	0.0217	0.0207	0.0196
	0.0184	0.0169	0.015	0.0126	0.0089					
46	0.0087	0.0122	0.0146	0.0164	0.0178	0.0191	0.0201	0.021	0.0218	0.0225
	0.0231	0.0236	0.0241	0.0245	0.0248	0.0251	0.0254	0.0255	0.0257	0.0258
	0.0259	0.026	0.026	0.026	0.026	0.0259	0.0258	0.0257	0.0255	0.0254
	0.0251	0.0248	0.0245	0.0241	0.0236	0.0231	0.0225	0.0218	0.021	0.0201
	0.0191	0.0178	0.0164	0.0146	0.0122	0.0087				
47	0.0082	0.0117	0.014	0.0158	0.0172	0.0185	0.0195	0.0204	0.0212	0.0219
	0.0225	0.023	0.0235	0.024	0.0243	0.0246	0.0249	0.0251	0.0252	0.0253
	0.0254	0.0254	0.0254	0.0254	0.0254	0.0254	0.0254	0.0253	0.0252	0.0251
	0.0249	0.0246	0.0243	0.024	0.0235	0.023	0.0225	0.0219	0.0212	0.0204
	0.0195	0.0185	0.0172	0.0158	0.014	0.0117	0.0082			
48	0.008	0.0114	0.0136	0.0154	0.0168	0.018	0.019	0.0199	0.0206	0.0213
	0.0219	0.0224	0.0229	0.0233	0.0237	0.024	0.0242	0.0245	0.0246	0.0248
	0.0249	0.0249	0.025	0.025	0.025	0.025	0.0249	0.0249	0.0248	0.0246
	0.0245	0.0242	0.024	0.0237	0.0233	0.0229	0.0224	0.0219	0.0213	0.0206
	0.0199	0.019	0.018	0.0168	0.0154	0.0136	0.0114	0.008		
49	0.0078	0.0111	0.0133	0.015	0.0163	0.0175	0.0185	0.0194	0.0201	0.0208
	0.0214	0.0219	0.0224	0.0228	0.0231	0.0234	0.0236	0.0238	0.024	0.0242
	0.0243	0.0244	0.0244	0.0244	0.0244	0.0244	0.0244	0.0244	0.0243	0.0242
	0.024	0.0238	0.0236	0.0234	0.0231	0.0228	0.0224	0.0219	0.0214	0.0208
	0.0201	0.0194	0.0185	0.0175	0.0163	0.015	0.0133	0.0111	0.0078	
50	0.0076	0.0108	0.0129	0.0146	0.0159	0.0171	0.0181	0.0189	0.0197	0.0203
	0.0209	0.0214	0.0218	0.0222	0.0225	0.0228	0.0231	0.0232	0.0235	0.0236
	0.0238	0.0238	0.0238	0.0238	0.0239	0.0239	0.0238	0.0238	0.0238	0.0238
	0.0236	0.0235	0.0232	0.0231	0.0228	0.0225	0.0222	0.0218	0.0214	0.0209
	0.0203	0.0197	0.0189	0.0181	0.0171	0.0159	0.0146	0.0129	0.0108	0.0076