Multi-objective Optimization of LARP Parameters using Weighted Sum DE Method

Behnam Seyedi

Submitted to the
Institute of Graduate Studies and Research
in partial fulfilment of the requirements for the Degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
October 2014
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We 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 in Computer Engineering.

Asst. Prof. Dr. Mehmet Bodur
Supervisor

Examining Committee

1. Asst. Prof. Dr. Adnan Acan _____

2. Asst. Prof. Dr. Mehmet Bodur _____

3. Asst. Prof. Dr.Ahmet Ünveren _____

ABSTRACT

This thesis employed the single-objective differential evolution (DE) algorithm to search the multi-objective solutions to obtain lateral controller (LARP) settings for an auto-steered tractor by combining two fitness functions, lateral peak and RMS errors, to a single objective using the weighted-sum-method. Compared to the multi-objective differential algorithm, weighted-sum DE algorithm covered a larger range of the Pareto-front. After modifying DE to set the search space adaptively, the modified method finds better non-dominated solutions than MODE by less number of fitness evaluations. Weighted Sum DE algorithm obtained better non-dominated solutions than MODE algorithm although weighted sum DE uses 20 000 fitness evaluation while MODE used 500 000 evaluations.

Keywords: Weighted sum optimization, LARP control, Automatic steering agricultural vehicle

ÖZ

Bu tezde otomatik sürüşlü bir traktörün tepe ve RMS hata olmak üzere iki objektifli en iyi LARP denetleci ayarlarını bulmak için tek-objektifli farksal evrim algoritması, iki objektifi ağırlıklı toplam yöntemiyle tek objektife indirerek kullanıldı. Çok objektifli farksal evrim algoritması MODE ile karşılaştırıldığında ağırlıklı toplamlı DE algoritmasının Pareto-önde daha geniş bir aralığı doldurduğu görülmüştür. Ağırlıklı toplamlı DE yönteminde arama uzayını adaptif olarak ayarlayacak değişikliklerden sonra oluşturulan yöntem MODE nin bulduğu çözümlerden daha iyi baskın olmayan çözümleri daha az uyum değerlendirmesi ile bulabilmektedir. Ağırlıklı toplamlı DE'nin 2000 uyum değerlendirmesiyle bulduğu çözümlerin MODE'nin 500000 uyum değerlendirmesiyle bulduklarından daha iyi olduğu gözlenmiştir.

Anahtar Kelimeler: Ağırlıklı toplamlı optimizasyon, LARP kontrol, Otomatik sürüşlü tarımsal araç

To My Family

ACKNOWLEDGMENT

I would like to express my special appreciation and thanks to my advisor Asst. Prof. Dr. Mehmet Bodur, who has been a tremendous mentor for me. I would like to thank him for encouraging my research and for promoting me to become a researcher. His priceless pieces of advice in both my research and my career have always been my guidelines. I would also like to thank my committee members, Asst. Prof. Dr. Adnan Acan, and Asst. Prof. Dr. Ahmet Ünveren for spending their precious time on my manuscript and offering their nourishing comments. I also want to thank them for their brilliant comments and suggestions in my defence.

My special thanks go to my family. Words cannot express how grateful I am to my mother father and brother for all of the sacrifices they have made on my behalf. Your prayer for me was what sustained me thus far. I would also like to thank all my friends who supported me in writing, and incented me to strive towards my goal.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

A function $f(x)$ from a real vector space to a single real scalar value is called a real valued function. Real valued functions have significant importance in modelling the input-output behaviour of many multi-input single-output engineering systems, including robotics mechanisms and their control algorithms [1] [2] [3].

An agricultural tractor is required to move mostly along the pre-specified trajectories in an agricultural field, with a sufficient precision to process the soil and planted products. The motion of a tractor along a desired trajectory consists of an important control problem that affects the success and the efficiency of the agricultural automation. A researcher may develop a mathematical model of the tractors acceleration, velocity, and the location for a set of inputs such as rear-tyre force, steering-wheel angle, gravitational force, and the ground-surface using parameters such as mass, inertia, and frictional forces involved in the mechanism. Mathematical modelling the motion of a tractor provides to a researcher very valuable and precise calculation of the tractors position, and gives them a chance to try many control strategies and control laws to keep the tractor on a desired trajectory for almost zero cost compared to the real implementation of that engineering system. Such models are already available in literature as a valuable tool of research.

Double Look-Ahead Reference Point (DLARP) simulation test-bed is a simulation of a tractor with an already proven control law to keep the modelled tractor on a desired trajectory. The simulation program needs a reference trajectory that shall be tracked by the simulated tractor calculating the deviation from the trajectory at specified time steps. Once the test drive is completed, the performance of keeping the tractor on the trajectory is easily determined from the lateral deviations of the tractor at each time steps [4].

One method to measure the performance is to find the maximum deviation along the trajectory, which means the maximum of all absolute valued deviations. Another method may be the integral of the absolute valued deviations. Mostly, small lateral deviations are tolerable, but larger deviations create much higher loss of product efficiency along an agricultural trajectory tracking. Square of the deviations provide better measure of performance considering the nonlinear dependence of the product efficiency on the lateral displacement. The root mean square (RMS) of the deviations provides both correction of the scale, and higher punishment of the larger displacements. In conclusion, different measures of performance provide development of better efficiency in different perspectives. In many cases, best efficiency depends on the optimization of critical control parameters to minimize more than one performance measure.

The test-bed numerically calculates a real valued function $f_m(x)$, where m is method of evaluation for the performance measure, and x stands for the vector of decision variables, which are involved in the optimization process of the control law plus some of the critical design parameter.

1.1 Single-Objective Optimization

In the optimization, a maximization problem is convertible to a minimization without loss of generalization [5]. This section will define and discuss the single objective optimization by only minimization of the functions. However, all definitions and methods described here are also valid for maximization of the functions.

A general single-objective optimization problem is defined as minimizing a real valued objective function $f(x)$ subject to constraints $g_i(x) \leq 0$, i = {1,...,m}, and $h_j(x) =0$, j = {1,...,p} $x \in \Omega$.

A solution $x= (x_1 ,..., x_n) \in \Omega$ is a n-dimensional decision variable vector that minimizes the scalar function $f(x)$. $g_i(x) \leq 0$ and $h_j(x)=0$ are the constraints to be satisfied in minimizing $f(x)$. Ω is the feasible region of x, in other words, it is the set of all possible x vectors that can be used to satisfy an evaluation of $f(x)$ and its constraints. Both decision vector x and the function f can be continuous or discrete.

A solution $x^* \in \Omega$, $\Omega \neq \emptyset$ provides $f^* = f(x^*) > -\infty$, the global minimum of a single objective function f: $\Omega \subseteq R^n \rightarrow R$ if and only if $\forall x \in \Omega : f(x^*) \leq f(x)$.

f is the objective function, x^* is the global minimum solution, and Ω is the feasible region of x. A single-objective optimization problem may have a unique optimal solution.

In this thesis, the objective function $f(x)$ is the measure of performance of the tractor simulation along a test trajectory. The objective function is evaluated by running DLARP simulation for a test trajectory that contains a ϕ=10m U-turn between two

100m linear paths, similar to the studies by Kiani [4], and Mahdizadeh [6], which provides a chance to the comparison of the results.

1.2 Multi-Objective Optimization

Depending on the type of agricultural tasks, there may be more than one objective in tracking a trajectory. Even for a particular task, different perspectives of success may require more than one objective to optimize for that task. For example, for the safety of the operation along the path, equipment and obstacles close to the tracking path such as irrigation systems in plantation area imposes minimum peak for the lateral displacement error, which requires damping the control action. Using control settings for minimum peak makes the motion of the tractor sluggish, and results in an undesired increase of the average of deviation. It means that the two objectives, the peak and the RMS error along a test path compete to each other. For a successful and efficient agricultural process, the lateral control law must be set to optimize both objectives at a required extent, which shall be determined depending on the conditions of the process.

Mathematically, a multi-objective optimization problem is a search of all optimum points that satisfies all competing objectives of the problem. The concept of Pareto front provides an understanding of the competing optimum solutions [5].

Let $f(x)=[\ f_1(x)\ \ f_2(x)\ \dots\ f_k(x)]'$ be the vector of k objective functions, where $x \in R^n$ is the decision vector, and all objective functions are well defined over the feasible region Ω, which is described by the constraints $g_i(x) \leq 0$, i = $\{1,...,m\}$, and $h_j(x) = 0$, $j = \{1,...,p\}$ $x \in \Omega$. The vector $f(x^{\wedge})$ is defined to dominate any other vector $f(x^{o})$, only if $f_i(x^{\wedge}) \leq f_i(x^{o})$ for all $i=1,\dots,k$. Domination of $f(x^{\wedge})$ over $f(x^{o})$ is denoted by $f(x^{\wedge}) \prec f(x^{o})$.

A solution x^* in the feasible region Ω is called Pareto-optimal solution if and only if there is no solution $x^o \in \Omega$ that satisfies $f(x^*) \prec f(x^o)$. All Pareto optimal solutions form the Pareto-front surface in the objective space.

There are plenty of multi-objective optimization algorithms to search the Pareto-front of objectives, and the corresponding Pareto-optimal solutions. NSGA-II and MODE were applied on the same problem by H. Mahdizadeh [6].

Pareto-front approach is not the only solution for the multi objective problems. Indeed, if someone needs only an arbitrary point on the Pareto-front surface, there is no need to search it by multi-objective optimization algorithms. E. Kiani obtained a satisfactory solution using iterative sub-optimal search algorithms to get minimum RMS error while keeping the peak error lower than a predetermined level [4].

1.3 Conversion of Multi-Objective Problem to Single-Objective

Gass and Saaty [8] converted two-objective problems to single objective (Equation 1.1):

$$J = w_1 f_1(x) + w_2 f_2(x) \qquad \text{(Eq. 1.1)}$$

where w1 and w2 are strictly non-negative weights which shall changed as parameter to get solutions corresponding to the different locations of the Pareto-front.

Zadeh [9] extended the same idea for a three-objective multi-objective optimization problem. Converting multi-objective problem to single objective does not reduce the computational effort in determination of the Pareto-front since it gives only one of the Pareto-solutions while a similar Multi-Objective algorithm may get a large

number of Pareto-solutions for almost the same number evaluations of the objective function.

1.4 Approach of This Thesis to Find an Optimal Controller Setting

The aim of this thesis is to determine the multi-objective optimal solutions of the lateral control problem of an agricultural tractor using a different approach, by converting the multi-objective nature of the problem to a single objective using the weighted sum method. The benefit of the selected method is mainly due to the adaptability of the weighted sum method for a low computational cost incremental iterative algorithm to shift the solution to a near location on the Pareto-front just continuing the iterations after a small change of weights.

Although the opportunity of shifting the solution along the Pareto-front is mentioned above, this thesis is limited only to the first step of the above mentioned plan, which is restricted to the implementation of a standard single-objective optimization algorithm.

Among the single-objective optimization algorithms, the differential evolution (DE) algorithm by Kenneth Price and Rainer Storn [10], [11], [12] gives better results in continuous decision spaces such as tuning of the controller parameters of robotic manipulators [1]. The success of DE is also observed in tuning the fuzzy modelling parameters for highly uncertain data modelling [13].

The remaining chapters of this thesis contain the following items. The second Chapter is devoted to explain the lateral control of an agricultural tractor and the performance test-bed to evaluate objective functions. The third Chapter explains the Differential Evolutionary algorithm to search the optimum solution of a real valued

function. The fourth Chapter explains the experimental details of obtaining a Pareto-front by using single objective optimization with simple weighted sum of objective functions. The fifth Chapter concludes the thesis.

Chapter 2

AUTO STEER CONTROL OF A TRACTOR

2.1 Tractor in Industrialized Agriculture

An agricultural tractor is an off-road vehicle to carry out agricultural processes such as ploughing, plantation, applying pesticides and fertilizers, and harvesting for the industrialized production of agricultural products. With the development of higher technologies, there is a heavy trend for the automation of the tractors for mainly to increase the production in quality and quantity with less cost including the qualified human work power as well as unqualified work power [4].

Conventionally, most of the agricultural processes, which are listed above, are carried while a tractor moves along a path, since agricultural plants are grown in lanes. Manual or auto steered, in most cases an agricultural tractor is steered along predefined reference routes at a constant speed. While the tractor tracks the reference trajectory, it makes two kinds of deflection from its reference path as seen in Figure 2.1.

a) A translational deviation along the reference trajectory (local x direction), either by delaying, or by moving advanced with respect to the point it shall be. This error is tolerable for the agricultural production efficiency since it only affects the time of processing.

b) A lateral deviation from the reference trajectory, to the right (local y direction) or to the left side (local -y direction). This error is critical, and the inefficiency of the process (i.e., sprayed fertiliser) decreases by the square of the error at the point of agricultural application.

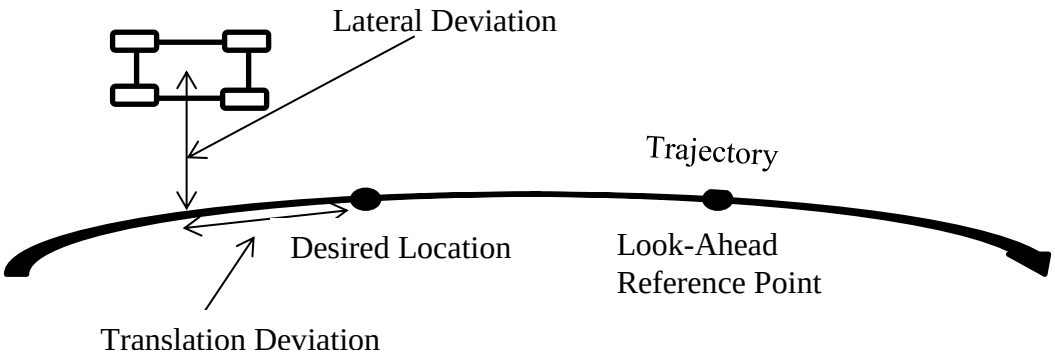

Figure 2.1: Desired location, lateral and translational deviation, and look-ahead reference point on a trajectory

At a typical 8 m/s speed, a manually driven tractor may have a lateral error about 0.2 m in average while following a line, reducing the efficiency of the operations 20 percent. This is because driving a tractor needs attention of the driver at a much higher degree than an average driver effort during all the work hours of driving. Worst of all, at the bending of the trajectories the manual driver makes up to 1.2 m peak lateral displacement errors, which may even put the life of driver in danger [4].

The auto steering of the tractor mimics the driving of an expert driver typically by looking to an advance point on the reference trajectory while correcting the lateral error by a closed loop PD control. The method is mostly called as “follow the carrot” by the driving school teachers, who suggests the young drivers to look almost twenty to fifty meters ahead depending on the driving velocity. At an auto-steered tractor, the closed loop PD control with a single carrot point reduces the lateral error at linear

trajectories down to sub centimetres. However, the reduction of the peak lateral error requires another independent measurement point on the reference trajectory. Double Look-ahead Reference Point (DLARP) completes this second feed forward information at a very low cost, and provides reduction of both peak and RMS errors below one centimetre [7].

2.2 Non-Holonomic Lateral Motion and Side Slip Motion

On a solid ground with high friction coefficient, a four-wheel vehicle with Ackerman front wheel navigation moves tangential to the back and front tyres, since the lateral friction on the tyres stop sidewise movement but the tyres roll on the translational direction. This kind of tangential restricted movement is called non-holonomic motion in kinematics. Ackerman mechanism converts the angular position of the driving wheel to steering angles of the right and left front tyres to develop a circular non-holonomic motion around the centre that passes through the rear axis of the vehicle with a radius *r*. Without loss of any degree of freedom, the dynamics of the tractor converges to a bicycle as seen in Figure 2.2.

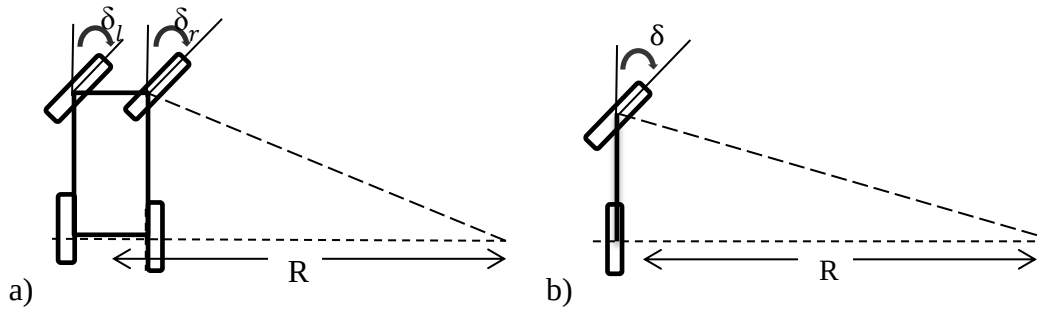

Figure 2.2: 4-wheel Tractor with Ackermann Steering Mechanism.
a) Non-holonomic turns with radius R, b) Bicycle representation of the mechanism

The agricultural soil does not conform to the properties of a solid ground, and movement contains extremely large slip (lateral frictional motion) and skid (translational frictional motion).

A tractor on a solid ground is expected to move tangential to the direction of its front and rear tyres, and make a non-holonomic circular motion when the front tyre directed to δ. Because of this tangential motion restriction, on the soil, together with its non-holonomic motion, it moves significantly sidewise, making a slip motion, and in forward-backward direction, making a skid motion. That means, a tractor floats on the tyres sidewise, because of lateral forces while it moves forward tangential to the tyres [4].

2.3 Mechanical Laws for Dynamic Simulation

After simplifying the tractor to a bicycle, the following mechanical laws are necessary to model the dynamic motion of the mechanism on the loose soil surface [4].

a) The acceleration of a body depends on the total applied force and the mass of the body by Newton's law of acceleration
b) The angular acceleration of a body depends on the total torque and the angular inertia by Euler's law.
c) Holonomic motion restricts the direction of motion in the limits of tyre frictions. As a result the body develops slip motion depending on side slip forces.
d) There is a random disturbance on the side slip forces due to randomly distributed soil clods on the trajectory of the tractor.

Once the dynamic motion of equations for the vehicle is modelled mathematically, it is discretized to calculate the new position of the vehicle for a small time step T=50ms, which is 100 times smaller than the expected time constants and oscillation modes in the system [7].

2.4 Reference Trajectory of Simulation

The simulation needs some other important items such as a list of points that describe the reference trajectory of the test path, sampled from a 100m line, U-turn of 8m diameter, and another 100m line with regular 10ms time intervals at 8 m/s speed. The code of the Matlab program in the Appendix 1, which is written by M. Bodur, calculates the points and writes them to a text file to be used by the DLARP simulation program [7]. Figure 2.3 shows the shape and dimensions of the reference trajectory.

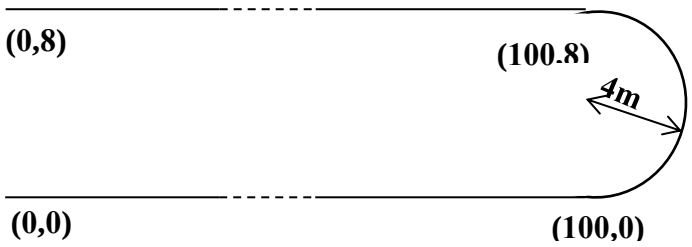

Figure 2.3: The shape of the reference trajectory

2.5 Units of the Auto Steering Simulation Test Bed

The overall block diagram of the auto-steering test-bed is shown in Figure 2.4. Main unit of the simulation test bed is the dynamic model of the tractor in discrete time steps. It calculates the differential amount of movement of the tractor to determine its acceleration, velocity, and location of the tractor from the side slip force

corresponding to the steering angle. Additional to the dynamic model unit, there are the model of the electro-hydraulic actuator that drives the steering angle of the front tyres, and the lateral controller that sends an electric signal to control the steering angle of the tyres to reduce the lateral displacement while the tractor tracks the reference path at constant 8m/s speed [7].

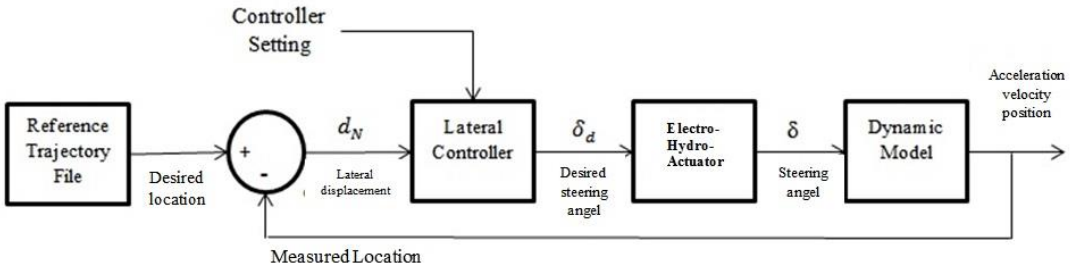

Figure 2.4: Block diagram of the auto-steering test-bed

The control unit calculates the lateral displacement by comparing the desired location to the measured location of the tractor. It uses DLARP control-law, which consists of six control settings. The desired and measured location of the tractor, its lateral displacement, desired and actual steering angles, and finally the acceleration and velocity of the vehicle are logged on an observation vector for later calculation of the average and peak errors of the tractor.

After processing all points of the reference trajectory, calculated lateral displacements are processed to compute the peak and the RMS error along the reference trajectory.

2.6 Peak and RMS Errors

The peak lateral displacement occurs especially at the curvature transitions, such as the start or the end of a U-turn. In the literature, the most advanced adaptive control methods can trim the controller parameters to reduce the average or RMS error to a

minimum level for the cost of extra peak errors at the start and end of the curvatures. The increased peak error of adaptive control system is explained by the adaptation time of the adaptive control to the new curvature condition [7].

The peak error, E_{peak}, of the tractor along the test trajectory is the maximum absolute lateral displacement on the test path (Equation 2.1).

$$E_{\text{peak}} = \max_{t=0}^{T_e} |d_N(t)| \qquad \text{(Eq. 2.1)}$$

Peak error is a major measure of performance because it is the main criteria to guarantee collusion free navigation if there are many critical obstacles in the work area. Agricultural fields may contains several kind of sensors and equipment for irrigation.

If the vehicle moves stable under the correctional effects of the lateral controller any deflection from a perfect linear reference trajectory is expected to converge asymptotically to zero. The efficiency of many agricultural processes are directly related to the root mean square error, shortly RMS error, shown by the symbol E_{RMS}.

$$E_{\text{RMS}} = \sqrt{\frac{1}{T_e}\int_{t=0}^{T_e} (d_N(t))^2 dt}\,. \qquad \text{(Eq. 2.2)}$$

Peak and RMS errors can be calculated after the computer simulation delivers N samples of lateral deviation, d_{N}, with time steps $T=T_e/N$.

$$E_{\text{peak}} = \max_{n=1}^{N} (|d_N(n)|) \qquad \text{(Eq. 2.3)}$$

$$E_{\text{RMS}} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} (d_N(n))^2} \qquad \text{(Eq. 2.4)}$$

Both E_{peak} and E_{RMS} are non-negative real numbers determined by the calculation of the lateral deviations d_N of the tractor along a desired reference test path [7].

2.7 Double Look-Ahead Reference Point Control Law

DLARP control law is a lateral control law, which calculates the desired steering angle δ_{des} by four terms based on the nearest (normal) point P_N on the reference path, and two look-ahead reference points P_{L1} and P_{L2}.

$$\delta_{des} = K_d d_N + K_N \theta_N + K_1 \theta_1 + K_2 \theta_2 \qquad \text{(Eq. 2.5)}$$

where d_N is the lateral deviation of the tractor to the path which is measured by the distance from P_N to the centre-of-gravity (CoG) of the tractor; θ_N, θ_1 and θ_2 are angular deviations between the heading angles of the tractor and the path points P_N, P_{L1} and P_{L2}; $\{K_d, K_N, K_1, K_2\}$ are controller coefficients. They are the control settings to be searched to reduce lateral error to a reasonably low level along the desired test path. Additionally, the points P_{L1} and P_{L2} are at the distances L_1 and L_2 from the point P_N as seen in Figure 2.5. The four controller coefficients together with the two look-ahead distances form the controller settings of the lateral control [7].

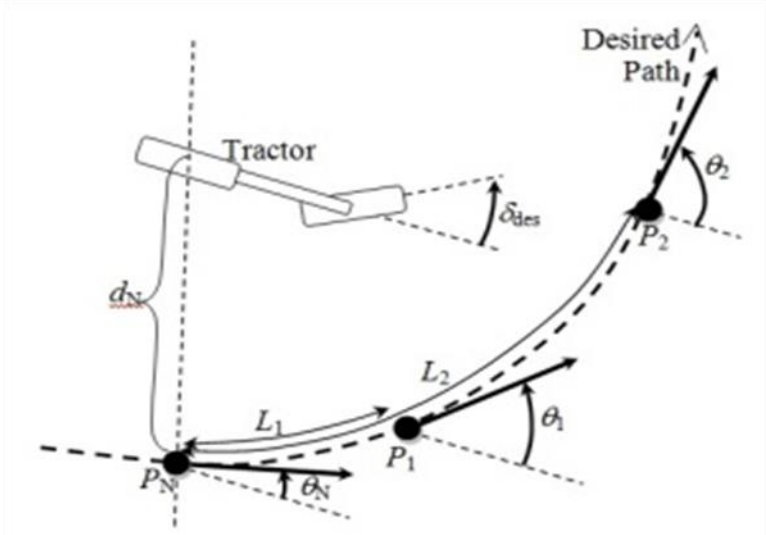

Figure 2.5: Variables related to DLARP control law

2.8 Evaluation of the Fitness Functions

Appendix 1 contains the MatLab source code "*fitness2.m*" of the fitness function, which is developed by M. Bodur [4]. The coded function has six arguments: four of them corresponding to the coefficients $\{K_N\ K_D.\ K_1,\ K_2\}$, plus, two arguments for the look ahead distances $\{L_1$, and $L_2\}$. It returns the errors E_{peak}, and E_{RMS}. Lines 154 to 170 of code cover the settings code for path, switching of three kinds of plots and initialize the coefficients for the simulation of the motion of the tractor, which is defined by Bevly and Derrick as described in [4]. Lines 177 to 187 of codes initialize some of the simulation variables of the test to set the tractor to the initial point. The simulation loop starts at line 188 of code. Lines 189 to 195 increments the time by the time step of simulation, and calculates several Cartesian and angular coordinates related to the motion of vehicle. Lines 196 to 207 correspond to lateral deviation measurement that search the normal point P_N on the desired path. Lines 208 to 220 of codes calculate the position of look-ahead points and then find angular deviation for each point. Control law determines the desired steering angle at line 223 once at every 50 ms period. Lines 231 to 234 calculate the action of the electro-hydraulic

servo-actuator giving a second order delay to the steering angle. The dynamics of the tractor is calculated to get local longitudinal and lateral acceleration and speeds of the tractor at lines 236 to 252. Lines 253 to 255 convert the local speed and direction to absolute coordinate frame. E_{peak} and E_{RMS} are obtained at lines 264 and 265 [6].

Chapter 3

DIFFERENTIAL EVOLUTION

3.1 Evolutionary Optimization Methods

Evolutionary optimization methods are population based search methods where populations of solution candidates are evolved over generations using the Darwinian principle of survival of the fittest [13]. The common structure of evolutionary optimization methods that minimize a real function $f(x)$ are summarized in the following manner.

An initial set of solutions, $X_0=\{x_1, x_2, \ldots, x_{np}\}$, which are mostly initialized randomly to increase the probability of converging to the global optimum of the problem. This set of candidate solutions is called the initial chromosome population. Each candidate solution is called a chromosome, and its components are called genes, i.e., any $x_i=(x_{i,1}, x_{i,2}, \ldots, x_{i,n}) \in \Omega$ is a chromosome and individually its components $x_{i,1}$, $x_{i,2}, \ldots, x_{i,n}$ are called genes. The function $f(x)$ is also called the fitness function of evolution. If the goal is to minimize $f(x)$, then, $f(x_i)<f(x_j)$ means the probability to of getting the best solution by modifying x_i is higher than the probability of achieving the best solution by modifying x_j. At the initialization step, all chromosomes are evaluated by their fitness values, and the population is ranked accordingly.

In the evolution loop, the population is processed by following operations:

i) **Reproduction**: to determine which chromosomes shall be selected for further processing and transfer their genes to the future generation.

ii) **Mutation**: to apply random changes to some of the genes of the selected chromosomes with a certain probability called *mutation rate*.

iii) **Crossover**: to swap the randomly determined genes of two selected chromosomes with a certain probability that is called the *crossover rate*.

iv) **Selection** of successful chromosomes to the existing population to get the next population.

iv) **Termination**: The evolution loop is stopped either when a certain number of generations without any better solution, or after a pre-specified number of evaluation of the fitness function.

At the end of the evolution, the solution that gives the lowest f(x) is declared to be the best solution of the problem.

Genetic Algorithm, GA, is the first introduced evolutionary algorithm, and it will be used here to describe the structure of the evolutionary algorithms. Genetic algorithm is general-purpose, population-based evolutionary algorithm to optimize objective functions that are non-differentiable, non-continuous, non-linear, noisy, flat, multidimensional, or has many local minima or constraints [14]. For a given minimization problem described by a parameter space $x \in \Omega$ and an objective function $f(x)$ defined on this space, a randomly initialized initial population is constructed, and individuals are evaluated by $f(x)$. The value $f(x)$ of an individual x is referred to its fitness and it is a relative measure of performance of x to satisfy the optimization criteria.

In a loop of evolution, the population is evolved over a number of generations, by a sequence of genetic operators: i) reproduction, ii) crossover, iii) mutation, and iv) selection as seen in Figure 3.1, which applies also for Differential Evolution algorithm.

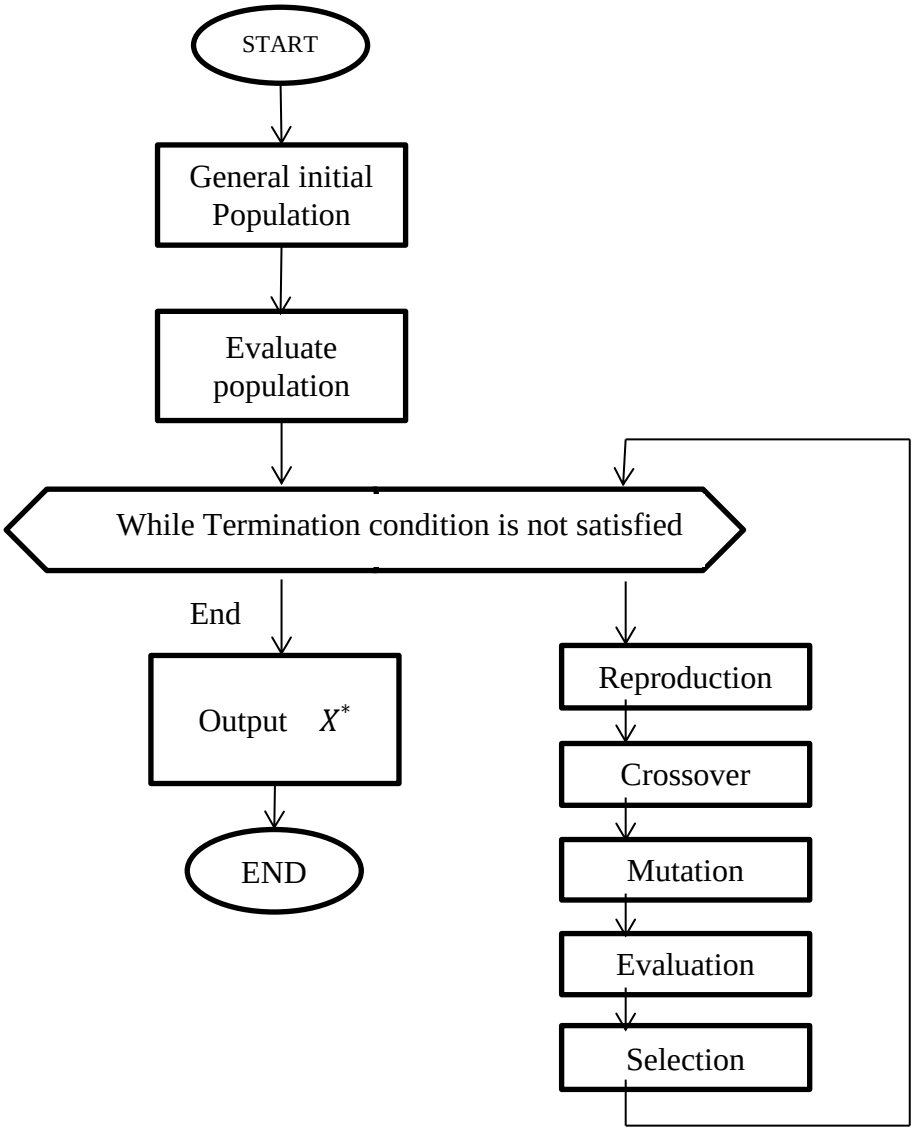

Figure 3.1: Simplified flowchart of Genetic and Differential Evolutionary optimization algorithms

3.2 Differential Evolution Algorithm

Mutation and greedy selection operators are the main search mechanism that directs the differential evolution population towards the regions closer to the optimum solution point in the search space Ω [13]. In contrast to GA, DE uses mutations of the differences of the parameter vector, while GA relies directly on random mutation values of the genes.

Let n_p be the population size, and g=0 ,… , n_g denote the generation number of the chromosome ${}^{g}x_i$ in the population ${}^{g}X$, and let ${}^{g}x_{i,k}$ be the k^{th} gene of ${}^{g}x_i$. A new mutant chromosome candidate ${}^{g,v}x_i$ is composed from the uniformly randomly chosen chromosomes ${}^{g}x_{r1}$, ${}^{g}x_{r2}$ and ${}^{g}x_{r3}$, such that the indices i, $r1$, $r2$ and $r3$ are mutually distinct numbers. The weighted difference of ${}^{g}x_{r2}$ and ${}^{g}x_{r3}$ is added on ${}^{g}x_{r3}$ to get a donor vector:

$$ {}^{g,v}x_i = {}^{g}x_{r1} + F\,({}^{g}x_{r2} - {}^{g}x_{r3}) $$

where $F \in (0,2]$ is called the mutation constant, for all i=1,…, n_p. One of the genes ${}^{g,v}x_{i,k}$, or ${}^{g,v}x_{i,k}$ is selected as the i$^{\text{th}}$ trial vector ${}^{g,t}x_{i,k}$:

$$ {}^{g,t}x_{i,k} = {}^{g}s_{i,k}\ {}^{g,v}x_{i,k} + (1 - {}^{g}s_{i.k})\ {}^{g}x_{i,k} $$

where ${}^{g}s_{i,k}$ =1 only if ${}^{g,t}x_{i,k} \in \Omega$ with probability CR; and ${}^{g}s_{i,k}$ =0 otherwise. (i.e., if ${}^{g,t}x_{i,k} \in \Omega$, and a random number $r_k \in [0,1]$ satisfies r_k<CR then ${}^{g}s_{i,k}$ =1, else ${}^{g}s_{i,k}$ =0). The constant CR is called the crossover rate.

The process of generating the trial chromosomes may be considered a kind of crossover because the randomly selected genes of i^{th} chromosome is exchanged with

the mutant donor chromosome, which is composed from three randomly selected chromosomes. The trial vectors ${}^{g,t}x_i$ (i=1,…, n_p) are accepted to the next generation only by a greedy selection process:

$$ {}^{g+1}x_i = {}^{g}c_i \, {}^{g,t}x_i + (1 - {}^{g}c_i) \, {}^{g}x_i $$

where ${}^{g}c_i$=0 if $f({}^{g,v}x_{i,k}) < f({}^{g,v}x_{i,k})$, and ${}^{g}c_i$=1 otherwise.

The greedy selection of the better chromosomes for next generations guarantees an always-decreasing objective function for all chromosomes in the population. In parallel to this advantage, the weighted combination of the three randomly selected chromosomes spans the entire search space, increasing the probability of catching the global solution. There are many variants of the DE algorithm to trim the evolutionary parameters *F* and *CR* from generations to generations, however, the basic algorithm with constant *F* and *CR* are conveniently used since the variants does not have an extremely significant advantage over the basic algorithm.

3.3 Weighted Sum Method to convert problem to single-objective

Single-objective DE is not suitable to solve multi-objective optimization problems unless the objectives F_i, i=1…k, are combined into a single objective. The weighted sum method (WSM) is the best known and simplest method to combine multiple objectives to a single objective [5].

The weighted sum method converts the multi objective problem of minimizing the vector of criteria functions, into a scalar problem by constructing a weighted sum $F(x)$ of all the objectives.

$$ F(x) = \sum_{i=1}^{k} w_i \, f_i(x) \qquad \text{(Eq. 3.1)} $$

where and k is number of criteria functions, x is objectives vector, and w_i is weight of i^{th} objective in the weighted sum.

For the simplicity of the formulation of the lateral controller settings, the weighted sum parameters w_i are selected as normalized factors of the two objectives, $f_{peak}(x) = E_{peak}$, and $f_{RMS}(x) = E_{RMS}$. In both $f_{peak}(x)$ and $f_{RMS}(x)$, $x=(k_N, k_D, k_1, k_2, L_1, L_2)$ is the vector of lateral controller settings, and the normalized weights are obtained by using a parameter, α, in the range $0 \leq \alpha \leq 1$, i.e.,

$$f_w(\alpha) = \alpha f_{peak} + (1-\alpha) f_{RMS} \ . \qquad \text{(Eq. 3.2)}$$

By these normalized weights, $w_{peak}=\alpha$, and $w_{RMS}=1-\alpha$ satisfies

$$\sum_{i=1}^{k} w_i = w_{peak} + w_{RMS} = 1 \ . \qquad \text{(Eq. 3.3)}$$

The normalized weights provide the weighted sum to be of the same units and in the same range with the fitness functions $f_{peak}(x) = E_{peak}$, and $f_{RMS}(x) = E_{RMS}$.

Chapter 4

SIMULATION RESULT OF OPTIMIZATION

4.1 Modifications of DE Algorithm for Weighted Sum Method

At the beginning of the simulation runs, the original DE algorithm is tested for an extensive number of fitness evaluations to search the optimum lateral control parameters of the modelled tractor to minimize only peak error, and only RMS error, which correspond to the cases of alpha=1 and alpha=0. The results, and the In these runs, the bounds of the problem search space has shown its importance, and later an automatic update mechanism is constructed into the DE search algorithm as an important contribution of this thesis.

Another issue is observed when the DE algorithm is started to run for alpha values in between zero and one. Typically, original DE starts with a randomly initialized population, which means completely forgetting the previous solutions. The initialization of the algorithm is modified to include the best solution of the previous run into the population of the next run, which has a small change in alpha value, and the best solution is expected at the near vicinity of the old solution.

4.2 Simulation Run for Minimum Peak Error (For $\alpha = 1$)

The best solution for minimum peak error is searched by setting $\alpha=1$, since $f_w(\alpha)= \alpha\, f_{peak}+(1-\alpha)\, f_{RMS}$ with $\alpha=0$ gives $f_w(1)=f_{peak}$ at the optimization run with ID 14_9_24_19_28.

DE parameters were set to number of generations *MaxIt*=20, population size *npop* = 5000. The lower and upper bounds of the mutation parameter *F*=(0.01, 0.6), and the crossover parameter *CR*=0.6, which are default values and usually good for almost all problems.

The bounds of the search space are dynamically set to 10% above and below the previous optimum solution dynamically, initializing them at the beginning to:

$-9.57<k_N<-8.32$, $0.86<k_D<0.98$, $7.84<k_1<9.02$, $8.06<k_2<9.29$, $0.52<L_1<0.59$, $-0.30<L_2<-0.26$

The convergence curve of the optimization simulation 14_9_24_19_28 is shown in Figure 4.1. The best solution obtained by this simulation after 500 000 evaluation of fitness has RMS error E_{RMS}= 0.00052m, and peak error around E_{peak}=0.00158m. In the solution space, the corresponding control parameters of this run are:

(k_N, k_D, k_1, k_2, L_1, L_2)= (-8.92216 0.88061 8.36509 8.59987 0.55795 -0.27465)

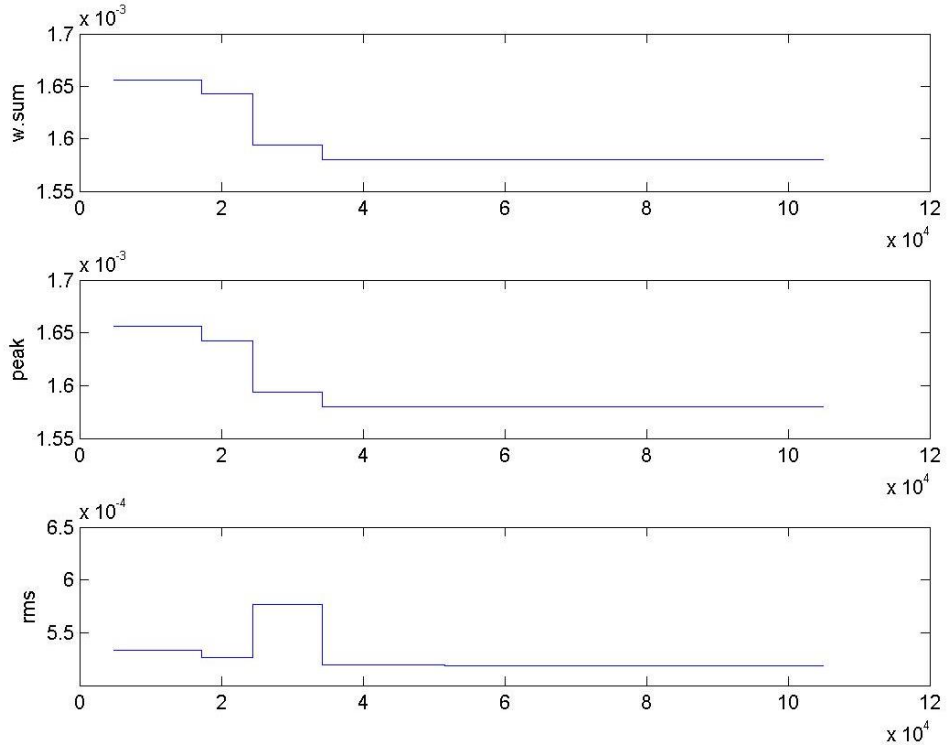

Figure 4.1: Convergence curve for α=1, x-axis is number of fitness evaluations

4.3 Simulation Run for Minimum Peak Error (For $\alpha = 0$)

Minimum peak error is searched by setting $\alpha = 0$, which gives the weighted fitness $f_w(0) = f_{\text{RMS}}$ in the simulation run 14_9_20_12_55, the DE parameters were set to number of generations *MaxIt*=20, population size *npop* = 5000. The lower and upper bounds of the mutation parameter F=(0.01, 0.6), and the crossover parameter CR=0.6, as used previously for α=1.

The bounds of the search space are dynamically set to 10% above and below the previous optimum solution dynamically, initializing them at the beginning to:

$-11.14<k_N<-10.08,\ 1.42<k_D<1.57,\ 7.41<k_1<8.19,\ 6.74<k_2<7.45,\ 0.54<L_1<0.60,\ -0.32<L_2<-0.29$

The convergence curve of the optimization simulation 14_9_20_12_55 is shown in Figure 4.2. The best solution obtained by this simulation gave the RMS error E_{RMS}= 0.00070m, and the peak error around E_{peak}=0.00362m. The control parameters obtained by this run is:

$$(k_N, k_D, k_1, k_2, L_1, L_2) = (-11.169\ 1.4267\ 7.4514\ 6.778\ 0.57945\ -0.30613)$$

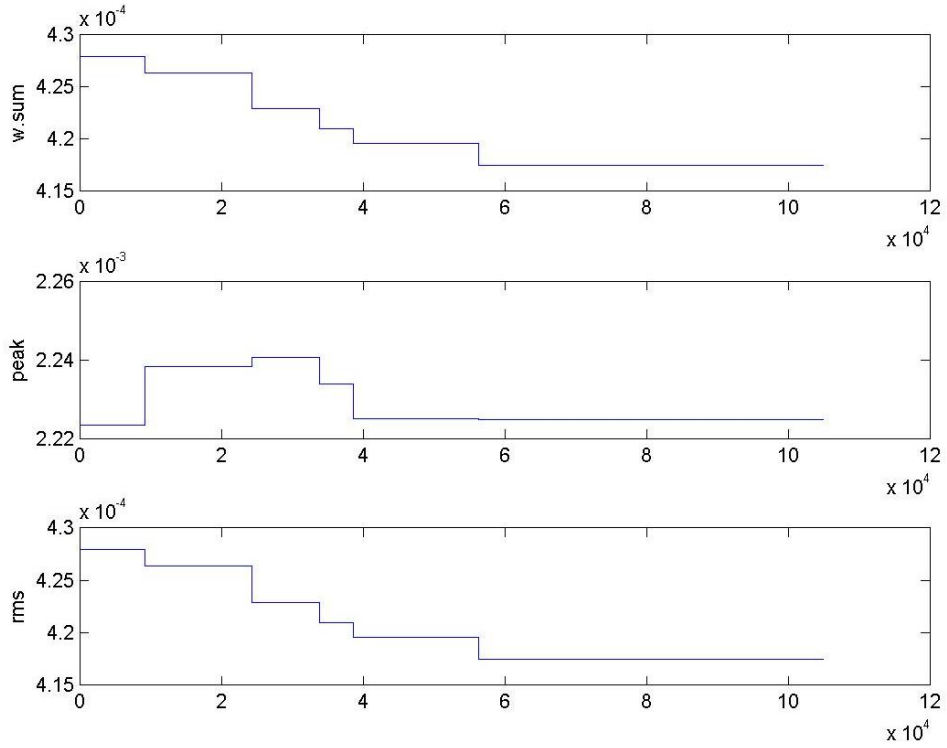

Figure 4.2: Convergence curve for α=0, x-axis is number of fitness evaluations

4.4 Simulation Runs for Intermediate Alpha Values

For simulation runs of the intermediate alpha values, alpha is started from 1.000, and decremented at each run (50 generations and 2000 population) by 0.025 until it reaches down to 0.6. From α=0.6 on, it is decremented by 0.05 at each step (20 generations and 5000 population), because by trial and error these settings provided good convergence plots. The best solution of the initial population is set to the best result of the previous step. The bounds of the solution space are set to 10% higher, and 10% lower values of the best solution that is obtained from the previous step. Moreover, at each generation, the bounds of the solution space are recalculated using the best solution of the previous generation. The dynamic adjustment of solution space bounds resulted in shift of the solution space bounds from step to step, and improved the optimization search significantly. The lists of the results of all runs are

shown in Appendix A. The fitness of the solutions by these runs are shown in Figure 4.3.

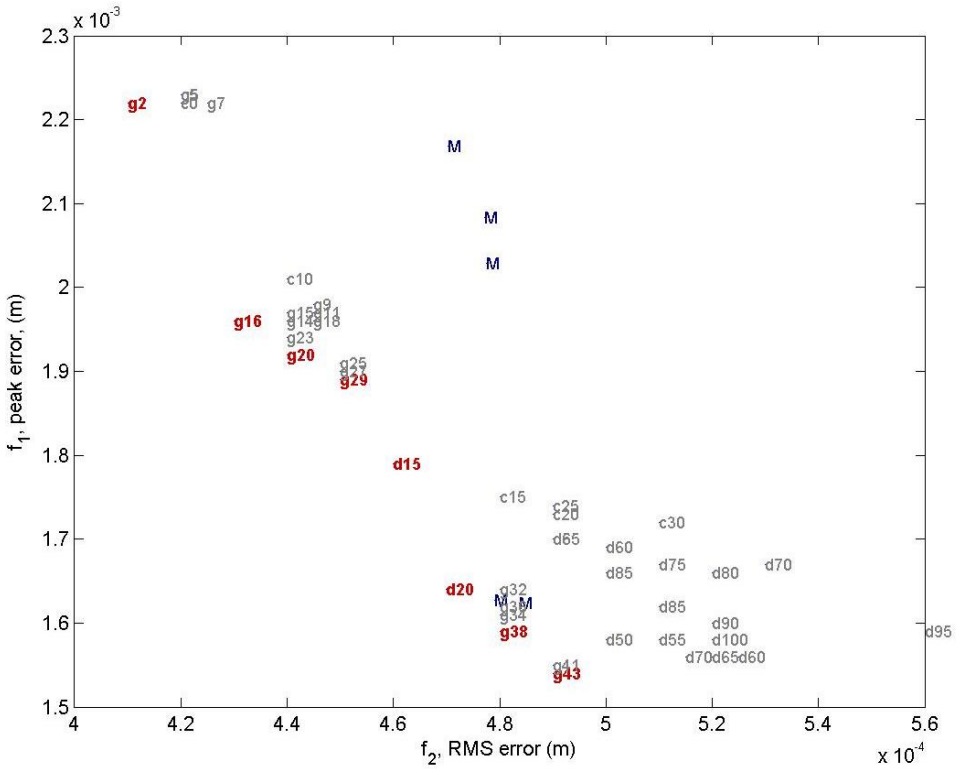

Figure 4.3: Fitness plane obtained by weighted sum DE method where each solution is marked by 100α.

In the fitness plane, the letter in the marker identifies the series of solution, and the number shows 100 times α, so that alpha value of each point appears on the objective plane with a marker. The c-series solutions c0, c10, c15, c20, c25, c30 are obtained starting from the point c35 with α=0.35. The points d50, d55, d60, d65, d70, d75, d80, d85, d90, d95 and d100 form the second set that was obtained by incrementing α from 0.50 to 1.00 by steps of 0.05. The g series of solutions were obtained by searching 15 solutions from α=0.43 to α=0.0225 at equal steps. The M series points indicate the Pareto-front obtained by H. Mehdizadeh, as listed in Table 4.1. [6].

Table 4.1: Controller Parameters for Pareto Optimal Points by MODE

ID	k_N	k_D	k_1	k_2	L_1	L_2	E_{peak}	E_{RMS}
Mehdizadeh 1	-9.9821	0.9018	8.1422	7.3861	0.56041	-0.30910	0.00162	0.000478
Mehdizadeh 2	-9.9283	0.9018	8.14229	7.3861	0.56041	-0.30910	0.00162	0.000478
Mehdizadeh 3	-8.37385	1.0092	7.4550	6.5188	0.58636	-0.32082	0.0020289	0.000477
Mehdizadeh 4	-8.37153	1.2401	7.4551	6.5164	0.58038	-0.31410	0.0020837	0.000477
Mehdizadeh 5	-8.54626	1.6540	7.5156	6.6306	0.58610	-0.32048	0.0021680	0.000470

4.5 Comparison of Pareto-Fronts by Weighted Sum against MODE

The Pareto-front obtained in this study is shown in Figure 4.4. The solutions corresponding to the Pareto-front fitness points are listed in Table 4.2. Mehdizadeh (2014) has obtained four Pareto-front points by multi-objective differential evolution algorithm for the same problem after reducing the six control parameters to four by the minimum offset condition on centrifugal and linear trajectories. Compared to the MODE, the weighted sum DE method provides (a) larger number of non-dominated solutions, which are almost evenly distributed along the Pareto-front curve; (b) the non-dominated solutions of MODE were fully dominated by the weighted sum DE solutions. In the plot obviously the five Pareto-front points obtained by MODE method has been largely dominated by Pareto-front points of weighted sum DE.

Table 4.2: Controller Parameters for Pareto Optimal Points by weighted DE

ID	α	*x*						*f*(*x*)	fitness	
		k_N	k_D	k_1	k_2	L_1	L_2		E_{peak}	E_{RMS}
14_9_25_9_23	0.15	-10.776	1.4753	8.1359	7.3654	0.5532	-0.30702	0.00066	0.00179	0.00046
14_10_2_14_2	0.20	-10.218	0.8834	8.1253	7.3509	0.5625	-0.30824	0.00071	0.00164	0.00047
14_10_7_19_55	0.427	-10.142	0.6738	8.1532	7.4057	0.55977	-0.30864	0.00094	0.00154	0.00049
14_10_7_20_36	0.3825	-10.050	0.7940	8.1267	7.3554	0.56152	-0.30788	0.00091	0.00159	0.00048
14_10_7_21_57	0.2925	-10.405	0.8761	7.7581	7.0261	0.56441	-0.30732	0.00087	0.00189	0.00045
14_10_7_23_18	0.2025	-10.540	0.9893	7.7483	7.0054	0.57148	-0.30681	0.00074	0.00192	0.00044
14_10_7_23_59	0.1575	-10.923	1.0967	7.7345	6.9744	0.57173	-0.30688	0.00068	0.00196	0.00043
14_10_8_2_0	0.0225	-11.346	1.2784	7.4382	6.7495	0.57881	-0.30659	0.00046	0.00222	0.00041

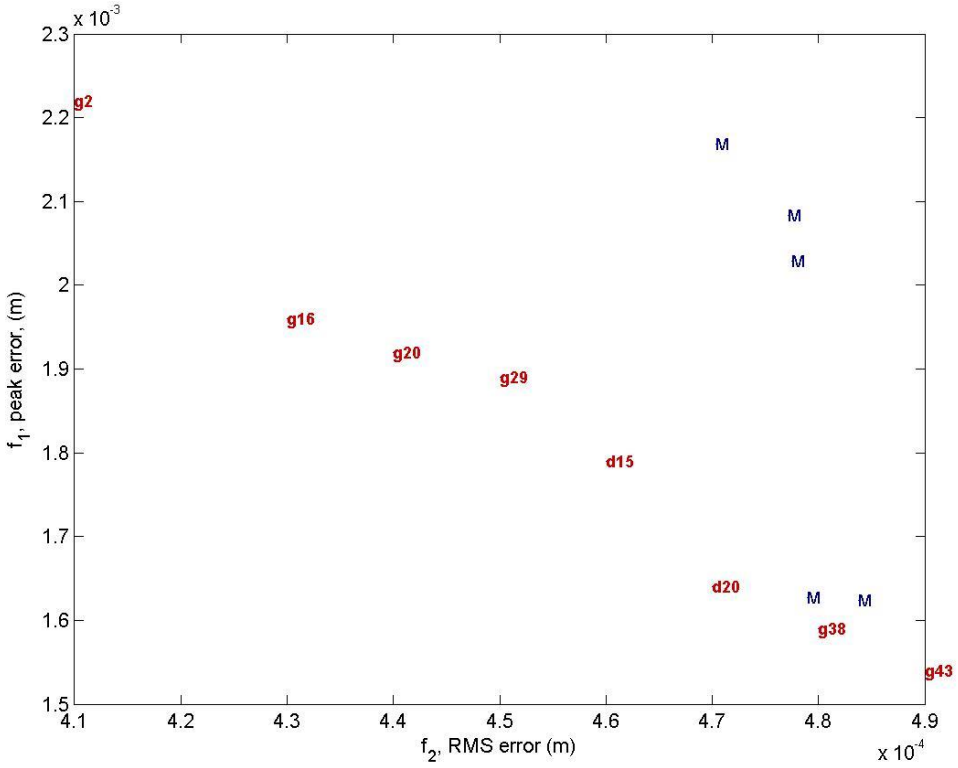

Figure 4.4: Pareto-front obtained by weighted sum DE method where each solution is marked by 100α

Chapter 5

CONCLUSION

This thesis targeted to observe possible advantages and disadvantages of the weighted sum method applied with DE in solving the lateral controller parameters that minimize both the peak and the RMS errors along the test trajectory of an auto-steered tractor. The single-objective differential evolution algorithm was employed to search the multi-objective solutions to obtain lateral controller (LARP) settings for an auto-steered tractor by combining two fitness functions, lateral peak and RMS errors, to a single objective using the weighted-sum-method with two small modifications. The first modification provides adaptive selection of the search space, and the second modification provides using the best solution of the previous run in the initial population. The obtained solutions of weighted-sum DE algorithm with only 10000 evaluations are superior compared to MODE results obtained with 500000 fitness evaluations. Weighted sum DE also provides a larger section of the Pareto-front.

As a result, weighted sum DE provides fast and more accurate solutions for the lateral controller settings of an agricultural vehicle. However, as a future work, the researcher recommends application of this method to a more suitable problem rather than the lateral controller because MODE already provides fine enough solutions in the sub millimetre range.

REFERENCES

[1] M. Bodur, “Real-time Population Based Optimization For Adaptive Motion Control Of Robot Manipulators," Engineering Letters, vol. 16, pp. 27-35, Mar 2008.

[2] M. Bodur, "An adaptive cross-entropy tuning of the pid control for robot manipulators" in Proceedings of the 2007 International Conference of Computational Intelligence and Intelligent Systems (ICCIIS-07), pp. 93-98, July 2007

[3] M. Bodur and M. E. Sezer, "Adaptive control of flexible multilink manipulators," International Journal of Control, vol. 58, no. 3, pp. 519-536, 1993

[4] Ehsan Kiani, *Design and Development of an Auto-Steering System Control for Off-Road Vehicles*, PhD Thesis, MEng Dept. Eastern Mediterranean Univ, Famagusta, TRNC, Cyprus. Sep 2012

[5] Carlos A. Coello Coello, Gary B. Lamont and David A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Second Edition, Springer, 2002

[6] H. Mahdizadeh, *Multi Objective Optimization of Control Parameters for Auto-Steering of Off-Road Vehicles*, MS Thesis, CMPE Dept., Eastern Mediterranean Univ. Famagusta, TRNC, Cyprus, Mar. 2014

[7] M.Bodur E. Kiani. H. Hacisevki., "Double look-ahead reference point control for autonomous agricultural vehicles," SCI Journal, Biosystems Engineering, Volume 113 , Issue 2, pp 173-186 , 2012

[8] S. Gass and T. L. Saaty. “The computational algorithm for the parametric objective function. Naval Research Logistics Quarterly,” 2,pp 39–45, 1955

[9] L. A. Zadeh. “Optimality and Nonscalar-Valued Performance Criteria”. IEEE Transactions on Automatic Control, AC-8(1),pp 59–60, 1963

[10] R. Storn and K. Price. “*Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces*”. Technical Report TR-95-012, International Computer Science Institute, Berkeley, California, March 1995.

[11] R. Storn and K. Price. “Differential Evolution - A Fast and Efficient Heuristic for Global Optimization over Continuous Spaces”. Journal of Global Optimization, 11,pp 341–359, 1997.

[12] K. V. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution. A Practical Approach to Global Optimization*. Springer, Berlin, 2005. ISBN 3-540-20950-6.

[13] M. Bodur, A. Acan, and T. Akyol, "Fuzzy system modeling with the genetic and differential evolutionary optimization," in Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on

Intelligent Agents, Web Technologies and Internet Commerce, vol. 1, pp. 432 - 438, Nov 2005

[14] D.E. Goldberg, *Genetic Algorithms in search, Optimization, and Machine Learning*, Addison Wesley, Massachusetts, USA, 1989

APPENDICES

APPENDIX A: MatLab Source Code of DE and Fitness Function

```
%% DE rewritten by M.Bodur 2014
function deM3
clc;
clear;
close all; format compact; format short;

nVar=6;     % Number of Decision Variables
fd ='opt4.txt'; fdx='xxx4.txt';% results file names
c =[-9.352 1.478 8.0008 6.53723 0.55241 -0.33589 0.00098 0.00180 0.00053 12 64414 0 ];
% Make constraints +/-10% around cx
cx = c(1:6); cxd=0.1*abs(cx);  xbound=[cx-cxd ; cx+cxd];
for ial=1:7
    recoverflag=0; StartIt=1; % recover=1 recovers the run from StartIt
    alpha=0.35-ial*0.05;gamma=0.1; delta=0.05;disp(alpha);
    MaxIt=20;     % Maximum Number of Iterations
    nPop=5000;    % Population Size
    if recoverflag==0
        clk=clock; fp(fd,'t = [ %i %i %i %i %i ];\r', clk(1:5) );
        pfs=int2str(clk(1));pfs=pfs(3:4);
        pfs= [pfs '_' int2str(clk(2)) '_' int2str(clk(3))];
        pfs= [pfs '_' int2str(clk(4)) '_' int2str(clk(5))];
        fh=fopen(fd,'a+'); fprintf(fh,'%% Start %s\r',pfs); fclose(fh);
        fhx=fopen(fdx,'w'); fprintf(fhx,'%% Start %s\r',pfs); fclose(fhx);
        Fmin=0.01;  % Lower Bound of Scaling Factor
        Fmax=0.5;   % Upper Bound of Scaling Factor
        pCR=0.4;    % Crossover Probability
        fp(fd,'a = [ %f %f %f %f %f %f ];\r', ...
            [ alpha MaxIt nPop Fmin Fmax pCR ]);
        fp(fd,'b = [ %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f \r',xbound(1,:) );
        fp(fd,'      %5.2f %5.2f %5.2f %5.2f5 %5.2f %5.2f ];\r',xbound(2,:));

        %% Initialization
        fcount=0;   % fitness count
        evold={zeros(MaxIt,1)}; evoli=0; % evolution convergence data
        empty_individual.x=[]; empty_individual.f=[];
        pop=repmat(empty_individual,nPop,1);

        % start the best as the first random item xn
        xn=cx; pop(1).x=xn; fcount=fcount+1;
        [fpx frx]=fitness2(xn(1),xn(2),xn(3),xn(4),xn(5),xn(6));
        pop(1).f= [alpha*fpx+(1-alpha)*frx   fpx  frx 1 fcount 0];
        % initialize remaining items
        for i=2:nPop
            xn=xbound(1,:)...
               +(xbound(2,:)-xbound(1,:)).*rand(1,size(xbound,2));
            [fpx frx]=fitness2(xn(1),xn(2),xn(3),xn(4),xn(5),xn(6));
            fcount=fcount+1; pop(i).x=xn;
            pop(i).f= [alpha*fpx + (1-alpha)*frx  fpx frx 1 fcount 0];
            if pop(i).f(1)< pop(1).f(1),
                tmp=pop(i); pop(i)=pop(1); pop(1)=tmp; end
        end
        cx= 0.9*cx+0.1*pop(1).x; % update the best for dynamic bounds
        % initialize convergence history of bestr
        evoli=evoli+1; evold{evoli}=pop(1);
        disp( evold{evoli});  fpc(fd,evold{1});
    end

%% DE Main Loop % do not use for loop it locks it and MaxIt.
it=StartIt; while(it<=MaxIt),
    if recoverflag==0, save('iter.mat');pause(0.1); end
    if recoverflag==1, load('iter.mat');pause(0.1); end
    %% dynamic constraints +/-10% around pop(1).x
    cxd=delta*abs(cx); xbound= [ cx-cxd ; cx+cxd ];
    % skip 1'th chromosome since it is best
    for i=2:nPop
        ssns=1; % search space not satisfied.
        while ssns,
           r1=randi(nPop);% r1, r2, r3 are mutually extinct
           r2=randi(nPop-1);r2=r2+(r2>=r1);
           r3=randi(nPop-2);r3=r3+(r3>=r1)+(r3>=r2);

           F=unifrnd(Fmin,Fmax,1,nVar); % Mutation scale parameter
           xv=pop(r1).x+F.*(pop(r2).x-pop(r3).x); % mutant donor xv

           xt=pop(i).x;       % Crossover initially x(i) trial xt
           j0=randi([1 nVar]);% at least x(j0) will be replaced
           ssns=0;
           for j=1:nVar,
               if j==j0 || rand<=pCR, xt(j)=xv(j); end,
               if( xt(j) > xbound(2,j) ), ssns=1;end
```

```
                if( xt(j) < xbound(1,j) ), ssns=1;end;
            end
        end

        [fpx frx]=fitness2(xt(1),xt(2),xt(3),xt(4),xt(5),xt(6));
        fcount=fcount+1;
        ft= [alpha*fpx+(1-alpha)*frx fpx frx it fcount 0];
        if ft(1)<pop(i).f(1),
            pop(i).x =xt; pop(i).f=ft; end % selected
        if ft(1)<pop(1).f(1),
            pop(i)=pop(1);
            pop(1).x =xt; pop(1).f=ft;  % swapped to best
            fpc(fd,pop(1));
            cx= gamma*cx+(1-gamma)*pop(1).x; %update dynamic bounds.
            gamma=min(1,gamma+0.2); delta=max(0.05,delta-0.01);
            end
        pause(0.001);
    end
    pop(1).f(6)=fcount; evoli=evoli+1; evold{evoli}=pop(1);
           fpc(fdx,evold{evoli}); disp( evold{evoli}); disp(it);
           plotconv(evold,evoli,fcount);
           saveas(gcf,pfs,'jpg');% save the plot

           it=it+1; StartIt=it;
       end
       disp( ['alpha ' num2str(alpha) ' finished.']);
       end
       disp('all finished.');
       return;
       end

       function fpc(fd,ss)
       fp(fd,'c = [ %8.5f %8.5f %8.5f %8.5f %8.5f %8.5f ',ss.x );
       fp(fd,'%8.5f %8.5f %8.5f %8.0f %8.0f %8.0f ];\r', ss.f );
       return
       end

       function fp(fd,s,d) % write to file 'opt.txt'
       fh=fopen(fd,'a+');
       fprintf(fh, s,d);
       pause(0.01);
       fclose(fh);
       return
       end

       function plotconv(evold,evoli,fcount)
       pdi=1;
       pd=zeros(2*evoli,6);
       pd(pdi,:)=evold{1}.f;
       if evoli>1,
         for i=2:evoli,
           pdi=pdi+1;
           pd(pdi,1:3)=pd(pdi-1,1:3); pd(pdi,4:6)=evold{i}.f(4:6);
           pdi=pdi+1; pd(pdi,:)=evold{i}.f;
         end
       end
       pdi=pdi+1; pd(pdi,:)=pd(pdi-1,:);pd(pdi,5)=fcount;
       hf=figure(1); set(hf,'Position',[30 60 750 750]);
       subplot(3,1,1); plot(pd(1:pdi,5),pd(1:pdi,1),'-');
       ylabel('w.sum');
       pause(0.1);
       subplot(3,1,2); plot(pd(1:pdi,5),pd(1:pdi,2),'-');
       ylabel('peak');
       pause(0.1);
       subplot(3,1,3); plot(pd(1:pdi,5),pd(1:pdi,3),'-');
       ylabel('rms');
       pause(0.1);
       return
       end

       %% fitness by M.Bodur (c) 2011
       function [peakd rmsd]=fitness2(kN,kD,k1,k2,L1,L2)
       %[a,b]=fitness(0,3,0,4,-0.8,0.8)
       % Path and Plot
       path=pathdef();
       len_path=size(path,1);
       startstep=0.001;
       plotgrA=0; plotgrB=0; plotgrM=0;
       %plotgrA=1; plotgrB=0; plotgrM=1;
       % Vehicle Coefficients
       x=1; y=2; w=3;       % Positional terms
       s=0; b=0; v=2.0; smax=0.5585; smin=-smax; % Steering parameter
       R=abs(path(1,2));     % Path parameters L2>L1
       % Bevly & Derrick (2008) coefficients
       m=11340; I=18500; Lf=1; Lr=2.0; L=Lf+Lr; Cr=286479; Cf=137510; Fr=0;
       dt=0.01; dtc=0.05; dts=0.001;
       ds=path(2,1); equit=1; cquit=0;
```

```
ibest=1; vbest=[]; iter=0; tcalc=0;

LrmseS=1050;LrmseE=1500; LpeakS=100; LpeakE=2100;
Lcancel=2400; Lpoff=900; % for circular and overall test
% Controller Coefficients
% kN=0; kD=3; k1=0; k2= 4.7;L1=-0.7;L2= 0.73;
% KNLine=5.6; KLCirc=2.28;
% k1=(KLCirc-k2*L2)/L1; kN=KNLine-k1-k2;

%--initialization----------------------------------------
t=0; j=0; tf=25;
ts=t; s1=0; s2=0; s3=0; % actuator time and states
tc=t+dt; % controller time
% Initial position and heading pv=[R -Lr+25 -pi/2];
pv=[-R-startstep Lr-10 pi/2];
vv=[0  v   0];    % Initial velocities
av=[0  0   0];    % Initial acceleration
xr=pv(x)-Lr*cos(pv(w)); yr=pv(y)-Lr*sin(pv(w));
clear ov; % observation vector is cleared
idN=2; dsign=1;
while( t<tf)
    t=t+dt; j=j+1; %time & iteration
    %simplification of program notation
    Cw= cos(pv(w)); Sw= sin(pv(w)); Cs= cos(s); Cb= cos(b);
    % CoG (x,y) --> (xr,yr) rear wheels point
    xrp=xr; yrp=yr; xr=pv(x) - Lr*Cw; yr=pv(y) - Lr*Sw;
    if(t==dt), xrp=xr; yrp=yr; end % debug
    if tc>=t, tc=tc+dtc; % control part
% control starts with finding pN
    % inputs idN, xr, yr,thetar, path  outputs idN, dN, d
    idNs=idN; idNend=idN+200;
    xk=path(idN-1,2); yk= path(idN-1,3);
    ddNp= (xk-xr)*(xk-xr)+(yk-yr)*(yk-yr); ddNpp=ddNp;
    for kdN=idNs:idNend,
        xk=path(kdN,2); yk= path(kdN,3);
        ddN=(xk-xr)*(xk-xr)+(yk-yr)*(yk-yr);
        idN=max(kdN-1,2);
        if(ddNp<ddN), break; end
        ddNpp=ddNp; ddNp=ddN;
    end
    dNbest=sqrt(ddNp); dNfw=sqrt(ddN); dNbw=sqrt(ddNpp);
    if(dNfw<dNbw), dNnext=dNfw; else dNnext=dNbw; end
    dN=nearestdist(dNbest,dNnext,ds); dN=abs(dN);
    thetaP=path(idN,4); thetaN=thetaP-pv(w);
    thetaP1=path(min(max(idN+round(L1/ds),1),len_path),4);
        theta1=thetaP1-pv(w);
    thetaP2=path(min(max(idN+round(L2/ds),1),len_path),4);
        theta2=thetaP2-pv(w);
    ca=path(idN,1); xn=path(idN,2); yn=path(idN,3);
    % determine sign of distance
    xNd=-xn+path(idN+1,2); yNd=-yn+path(idN+1,3);
    dsign= sign(xNd*(-yr+yn)-yNd*(-xr+xn));
    d= dsign*abs(dN);

    % Control Law for steer angle
    sd=kD*d + kN*thetaN + k1*theta1 + k2*theta2;
   end
  %--control part is over, vehicle simulation starts here--------

    % Quit the case if error exceeds a threshold
     eer=abs(d);  if eer>equit, cquit=1; t=tf;  end

    % Servo Actuator
    while(ts<t), ts=ts+dts;
       s=sd*0.000297 + 2.894*s1 - 2.858997*s2 +0.9647*s3 ;
       s3=s2; s2=s1;s1=s; end
    s=max(min(s,smax),smin);

  % Vehicle equations of motion
    if s>=0, Ff= -Cf*(s-b- vv(w)*Lf*Cb/v);  % Friction force is
    else Ff= Cf*(b + vv(w)*Lf*Cb/v-s); end  %   normal to tires
    Fc=m*(v*cos(b))^2*tan(s)*sign(b)/L; Fwf=27e3*sin(s);
    ddy=(-Fr-Ff*Cs-Fc+Fwf)/m+v*vv(w);
    av(w)= (-Ff*Cs*Lf + Fr*Lr+Fwf*Lf)/I; av(x)=-ddy*Sw; av(y)= ddy*Cw;
    % Absolute Velocities
    vv(x)=vv(x)+av(x)*dt; vv(y)=vv(y)+av(y)*dt; vv(w)=vv(w)+av(w)*dt;
    % Constant Forward Velocity
    vyy=-vv(x)*Sw+vv(y)*Cw; % Lateral y velocity
    vv(x)=-vyy*Sw + v*Cw; vv(y)= vyy*Cw + v*Sw;
    % Side slip angle
    b= atan2(vv(y),vv(x))-pv(w);
    Fr= Cr*(b - vv(w)*Lr*Cb/v); % Friction force is normal to tires
    % Vehicle positions and orientation
    pv(x)=pv(x)+vv(x)*dt; pv(y)=pv(y)+vv(y)*dt; pv(w)=pv(w)+vv(w)*dt;
  %=============================================================
    %        1 2 3  4  5  6  7   8   9     10    11 Observation Vector
    %        t s Xn Yn d  xr yr  dN thetaN dsign ca
```

```
255.            ov(j,:)=[t s xn yn d  xr yr  dN thetaN dsign ca ];
256.        end % case run completed
257.
258.        L_ov=size(ov,1);
259.        %peaka=mxerr; rmsd=mxerr;
260.        %LrmseS=1050;LrmseE=1500; LpeakS=100; LpeakE=2100;
261.        if cquit, peakd=equit; rmsd=equit;
262.        else
263.          [peaka,ipeak]=max(ov(100:2100,8));
264.          peakd = abs(ov(ipeak+100,5));
265.          rmsd  = sqrt(ov(100:1500,5)'*ov(100:1500,5)/(1500-100) ) ;
266.          poff=ov(900,5);
267.          dispvec=[ L1 L2 k1 k2 peakd*1000 rmsd*1000 poff*1000 ];
268.          % plot the testdrive
269.          if plotgrA,
270.            figure(2);
271.            plot(ov(:,6),ov(:,7),'LineWidth',1.5);hold on;
272.            plot(ov(:,3),ov(:,4),'r','LineWidth',0.5);
273.          end
274.          if plotgrM,
275.            figure(1);
276.            plot(ov(:,11),ov(:,5),'r','LineWidth',1.5);
277.          end
278.          %fit=alpha*peakd+(1-alpha)*rmsd;
279.        end
280.        return
281.        end
282.
283.        function d=nearestdist(d1,d2,d0)
284.        e=(d1*d1-d2*d2-d0*d0)/(2*d0); d=sqrt(abs(d2*d2-e*e)); a=-e;
285.        return
286.        end
287.
288.        function path=pathdef()
289.         path=[ 0.000000  -7.000000   -10.000000   1.570796   1
290.        0.013890  -7.000000   -9.986110   1.570796   2
291.        0.027780  -7.000000   -9.972220   1.570796   3
292.        0.041670  -7.000000   -9.958330   1.570796   4
293.        0.055560  -7.000000   -9.944440   1.570796   5
294.        0.069450  -7.000000   -9.930550   1.570796   6
295.        0.083340  -7.000000   -9.916660   1.570796   7
296.        0.097230  -7.000000   -9.902770   1.570796   8
297.        0.111120  -7.000000   -9.888880   1.570796   9
298.        0.125010  -7.000000   -9.874990   1.570796   10
299.        0.138900  -7.000000   -9.861100   1.570796   11
300.        0.152790  -7.000000   -9.847210   1.570796   12
301.        …
302.        …
303.        93.924180  7.000000   -61.933031   -1.570796   6763
304.        93.938070  7.000000   -61.946921   -1.570796   6764
305.        93.951960  7.000000   -61.960811   -1.570796   6765
306.        93.965850  7.000000   -61.974701   -1.570796   6766
307.        93.979740  7.000000   -61.988591   -1.570796   6767
308.        93.993630  7.000000   -62.002481   -1.570796   6768];
309.        return
310.        end
```

APPENDIX B: Simulation Output for Best Solutions

The following list of outputs is obtained from the optimization runs. It contains

1- t = [year, month, day, hour, minutes] of the start (ID number for the run).

2- a = [α, # of Generations, Population size, Fmin, Fmax, CR]

3- b = [$k_{N,min}$, $k_{D,min}$, $k_{1,min}$, $k_{2,min}$, $L_{1,min}$, $L_{2,min}$,
$k_{N,min}$, $k_{D,max}$, $k_{1,max}$, $k_{2,max}$, $L_{1,max}$, $L_{2,max}$]

the upper and lower bounds of solution space.

4- c = [k_N, k_D, k_1, k_2, L_1, L_2, f, fp, fr, count of gener., count of fitness eval, none]

of the best solutions in that run. The convergence plots of each run are obtained from these records. The last record gives the best solution for that run.

The The line that starts with % specifies the attributes of the marker on the fitness plane plot. The second character is "1" for the Pareto-front points, "3" for the non-Pareto-front cases, and "4" for the points found by H.Mahdizadeh. The third character appears as prefix on the marker.

```
%4MStart Hamed Mahdizadeh  1
a = [ 0.000000 0.000000 0.000000 0.0 0.0 0.0 ];
b = [ -1.00 -1.00  4.00  2.00  0.50 -0.80
      0.00  3.00  6.00  3.00  1.00  0.40 ];
c = [-9.98215  0.8648   8.16135   7.4208   0.559544  -0.3081386 0 0.001623511593118   0.000483730689492 0 0 0];
%4MStart Hamed Mahdizadeh  2
a = [ 0.000000 0.000000 0.000000 0.0 0.0 0.0 ];
b = [ -1.00 -1.00  4.00  2.00  0.50 -0.80
      0.00  3.00  6.00  3.00  1.00  0.40 ];
c = [  -9.92839  0.9018   8.14229   7.3861  0.560413  -0.3091005 0 0.001626986744380   0.000478981810351 0 0 0];
%4MStart Hamed Mahdizadeh  3
a = [ 0.000000 0.000000 0.000000 0.0 0.0 0.0 ];
b = [ -1.00 -1.00  4.00  2.00  0.50 -0.80
      0.00  3.00  6.00  3.00  1.00  0.40 ];
c = [-8.37385  1.0092   7.45505   6.5188   0.586364  -0.3208218 0 0.002028945785805   0.000477515211713 0 0 0];
%4MStart Hamed Mahdizadeh  4
a = [ 0.000000 0.000000 0.000000 0.0 0.0 0.0 ];
b = [ -1.00 -1.00  4.00  2.00  0.50 -0.80
      0.00  3.00  6.00  3.00  1.00  0.40 ];
c = [-8.37153   1.2401   7.4551   6.5164   0.580384  -0.314105 0 0.002083796721039   0.000477172698992 0 0 0];
%4MStart Hamed Mahdizadeh   5
a = [ 0.000000 0.000000 0.000000 0.0 0.0 0.0 ];
b = [ -1.00 -1.00  4.00  2.00  0.50 -0.80
      0.00  3.00  6.00  3.00  1.00  0.40 ];
c = [-8.54626   1.6540   7.5156   6.6306   0.586108  -0.3204827 0 0.002168091751488   0.000470349336846 0 0 0];
```

```
t = [ 2014 9 19 10 45 ];
%3cStart 14_9_19_10_45
a = [ 0.300000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -10.29  1.33  7.20  5.88  0.50 -0.37
     -8.42  1.63  8.80  7.195  0.61 -0.30 ];
c = [ -9.35235  1.47865  8.00088  6.53723  0.55241 -0.33589  0.00091  0.00180  0.00053       1       1       0 ];
c = [ -9.60914  1.45284  7.71328  6.64298  0.56950 -0.32623  0.00091  0.00194  0.00047       1    9581       0 ];
c = [ -9.76418  1.50604  8.06660  6.92770  0.55266 -0.32338  0.00090  0.00184  0.00050       2   10065       0 ];
c = [ -9.79152  1.45152  7.98090  6.49907  0.55039 -0.33857  0.00090  0.00175  0.00054       2   10136       0 ];
c = [ -9.69560  1.47565  8.05578  6.91740  0.55461 -0.32380  0.00087  0.00172  0.00051       7   35619       0 ];
c = [ -10.09929  1.48730  8.02353  6.84971  0.55942 -0.32399  0.00087  0.00175  0.00050      19   99873       0 ];
c = [ -9.65382  1.48012  8.05004  6.90501  0.56183 -0.32220  0.00087  0.00172  0.00051      20  100084       0 ];

t = [ 2014 9 19 14 10 ];
%3cStart 14_9_19_14_10
a = [ 0.250000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -10.26  1.40  7.60  6.42  0.53 -0.35
     -9.28  1.55  8.40  7.095  0.58 -0.31 ];
c = [ -9.33143  1.46847  8.08668  6.97889  0.55006 -0.32128  0.00083  0.00171  0.00053       1    4597       0 ];
c = [ -9.66929  1.50094  8.10080  6.99817  0.56147 -0.31305  0.00082  0.00177  0.00050       1    9165       0 ];
c = [ -10.08129  1.49503  8.03670  6.87962  0.55898 -0.32587  0.00082  0.00173  0.00051       2   12780       0 ];
c = [ -9.84103  1.48217  8.06744  6.93276  0.54985 -0.32514  0.00081  0.00176  0.00049       3   17408       0 ];
c = [ -10.00063  1.52335  8.03719  6.87879  0.55506 -0.31688  0.00081  0.00173  0.00050       4   21685       0 ];
c = [ -9.85682  1.42798  8.05779  6.91882  0.55802 -0.31487  0.00081  0.00172  0.00051       4   24205       0 ];
c = [ -10.29011  1.45327  8.07410  6.94176  0.55082 -0.31627  0.00080  0.00175  0.00049       6   30226       0 ];
c = [ -10.20902  1.43827  8.06716  6.92878  0.55600 -0.31746  0.00080  0.00174  0.00049      13   67690       0 ];

t = [ 2014 9 20 2 26 ];
%3cStart 14_9_20_2_26
a = [ 0.200000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -10.42  1.42  7.65  6.56  0.53 -0.34
     -9.43  1.57  8.46  7.255  0.58 -0.30 ];
c = [ -9.92299  1.49222  8.05383  6.90916  0.55512 -0.32104  0.00074  0.00173  0.00050       1       1       0 ];
c = [ -10.37997  1.43296  8.06100  6.91441  0.55688 -0.31759  0.00074  0.00177  0.00048      12   63446       0 ];
c = [ -10.32443  1.44038  8.06594  6.92595  0.55186 -0.32530  0.00074  0.00175  0.00049      13   67211       0 ];
c = [ -10.12375  1.43483  8.05628  6.90934  0.55885 -0.31775  0.00074  0.00173  0.00049      15   76199       0 ];

t = [ 2014 9 20 5 39 ];
%3cStart 14_9_20_5_39
a = [ 0.150000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -10.74  1.37  7.66  6.57  0.53 -0.34
     -9.71  1.51  8.46  7.265  0.58 -0.30 ];
c = [ -10.22556  1.43738  8.06026  6.91584  0.55608 -0.32042  0.00068  0.00174  0.00049       1       1       0 ];
c = [ -10.61251  1.46558  7.79271  7.08966  0.56658 -0.31186  0.00067  0.00200  0.00044      14   74722       0 ];
c = [ -10.46567  1.44903  8.05242  6.89827  0.55490 -0.31712  0.00067  0.00175  0.00048      16   84282       0 ];

t = [ 2014 9 20 8 57 ];
%3cStart 14_9_20_8_57
a = [ 0.100000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -11.02  1.38  7.58  6.60  0.53 -0.33
     -9.97  1.53  8.38  7.305  0.59 -0.30 ];
c = [ -10.76682  1.41024  7.78055  7.06177  0.57224 -0.30965  0.00060  0.00200  0.00044       1     113       0 ];
c = [ -10.29047  1.52060  7.77543  7.05932  0.57174 -0.30679  0.00060  0.00199  0.00045       1    6319       0 ];
c = [ -10.67012  1.51144  7.79865  7.09837  0.56916 -0.30592  0.00060  0.00201  0.00044       4   24671       0 ];
c = [ -10.63266  1.46465  7.79406  7.09382  0.57491 -0.30781  0.00060  0.00201  0.00044       6   30320       0 ];
c = [ -10.61571  1.50357  7.79729  7.09794  0.56440 -0.31087  0.00060  0.00201  0.00044       8   44760       0 ];
c = [ -10.65404  1.49421  7.79802  7.09949  0.57404 -0.30416  0.00060  0.00201  0.00044       9   46723       0 ];
c = [ -10.53104  1.45468  7.76314  7.03378  0.56505 -0.30573  0.00060  0.00198  0.00044      11   59808       0 ];
c = [ -10.90110  1.49609  7.78809  7.07807  0.56423 -0.30153  0.00059  0.00201  0.00044      15   78444       0 ];
c = [ -10.99275  1.48726  7.77866  7.05976  0.56910 -0.30181  0.00059  0.00201  0.00044      16   81288       0 ];

t = [ 2014 9 20 12 55 ];
%3cStart 14_9_20_12_55
a = [ 0.050000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -11.14  1.42  7.41  6.74  0.54 -0.32
     -10.08  1.57  8.19  7.455  0.60 -0.29 ];
```

```
c = [ -10.88526  1.51123  7.79826  7.44132  0.57434 -0.29767  0.00053  0.00201  0.00045      1    4440       0 ];
c = [ -10.78589  1.46323  7.45050  6.78158  0.58341 -0.29945  0.00051  0.00220  0.00043      1    6404       0 ];
c = [ -10.66690  1.53019  7.47364  6.82559  0.58078 -0.30837  0.00051  0.00222  0.00042      4   23403       0 ];
c = [ -10.74032  1.44863  7.48845  6.85443  0.58821 -0.30806  0.00051  0.00223  0.00042      7   35568       0 ];
c = [ -10.86809  1.49006  7.47364  6.82559  0.58078 -0.31000  0.00051  0.00222  0.00042      8   40567       0 ];
c = [ -10.97141  1.40714  7.47830  6.83227  0.58966 -0.30837  0.00051  0.00223  0.00042      8   41876       0 ];

t = [ 2014 9 20 16 13 ];
%3cStart 14_9_20_16_13
a = [ -0.000000 20.000000 5000.000000 0.010000 0.500000 0.400000 ];
b = [ -11.32  1.40  7.11  6.49  0.55 -0.32
     -10.25  1.55  7.85  7.185  0.61 -0.29 ];
c = [ -10.78463  1.47533  7.47985  6.83445  0.58414 -0.30794  0.00043  0.00222  0.00043      1      1       0 ];
c = [ -10.69546  1.49024  7.50541  6.89136  0.58412 -0.31092  0.00043  0.00224  0.00043      1    9268       0 ];
c = [ -10.91662  1.44351  7.49660  6.87005  0.58331 -0.31006  0.00042  0.00224  0.00042      4   24343       0 ];
c = [ -11.05423  1.44724  7.47569  6.82909  0.58804 -0.30683  0.00042  0.00223  0.00042      6   33910       0 ];
c = [ -11.00383  1.52112  7.46550  6.80872  0.58036 -0.30874  0.00042  0.00222  0.00042      7   38682       0 ];
c = [ -11.16894  1.42673  7.45140  6.77833  0.57945 -0.30613  0.00042  0.00222  0.00042     11   56371       0 ];

t = [ 2014 9 22 11 47 ];
%3dStart 14_9_22_19_47
a = [ 0.600000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.23  1.20  7.42  6.72  0.51 -0.33
     -8.54  1.44  8.88  8.045  0.61 -0.28 ];
c = [ -9.25047  1.43868  7.82410  7.50047  0.56830 -0.29672  0.00137  0.00193  0.00052      1     30       0 ];
c = [ -9.96917  1.21460  8.22529  7.54856  0.55379 -0.30705  0.00127  0.00177  0.00051      1   6919       0 ];
c = [ -9.11469  1.39125  8.17540  7.47017  0.55877 -0.30093  0.00125  0.00171  0.00056      1   8647       0 ];
c = [ -9.65486  1.34694  8.13627  7.38966  0.55902 -0.30200  0.00125  0.00172  0.00054      4  23412       0 ];
c = [ -9.41279  1.34580  8.15188  7.41332  0.56209 -0.30223  0.00122  0.00171  0.00049      4  23488       0 ];
c = [ -9.18836  1.28409  8.17005  7.44890  0.56182 -0.30237  0.00122  0.00170  0.00050     11  58934       0 ];
c = [ -9.19483  1.27229  8.16707  7.44508  0.55464 -0.31077  0.00121  0.00169  0.00050     15  75209       0 ];
c = [ -9.13416  1.30371  8.15916  7.42892  0.54926 -0.30953  0.00121  0.00169  0.00050     19  95116       0 ];

t = [ 2014 9 23 0 16 ];
%3dStart 14_9_23_0_16
a = [ 0.650000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.92  1.27  7.75  7.06  0.53 -0.32
     -8.98  1.41  8.57  7.805  0.59 -0.29 ];
c = [ -9.00894  1.28210  8.17903  7.47535  0.55104 -0.31237  0.00128  0.00168  0.00055      1   4521       0 ];
c = [ -9.71601  1.29100  8.13587  7.37465  0.55667 -0.31216  0.00128  0.00171  0.00048      8  40520       0 ];
c = [ -9.65706  1.31498  8.12947  7.36587  0.56247 -0.31013  0.00128  0.00171  0.00048      9  48941       0 ];
c = [ -8.96792  1.29020  8.18762  7.48961  0.56168 -0.30464  0.00128  0.00169  0.00053     15  75855       0 ];
c = [ -9.44884  1.31203  8.14779  7.40286  0.56054 -0.30481  0.00128  0.00170  0.00049     16  82084       0 ];
c = [ -9.61218  1.27986  8.12337  7.35195  0.56153 -0.30697  0.00128  0.00170  0.00049     18  94955       0 ];
c = [ -9.47012  1.25505  8.15355  7.41312  0.55442 -0.30241  0.00127  0.00170  0.00049     20 100406       0 ];

t = [ 2014 9 23 5 27 ];
%3dStart 14_9_23_5_27
a = [ 0.700000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.85  1.24  7.75  7.04  0.53 -0.32
     -8.91  1.37  8.56  7.795  0.59 -0.29 ];
c = [ -9.38190  1.30100  8.15297  7.41487  0.56108 -0.30676  0.00134  0.00170  0.00049      1      1       0 ];
c = [ -9.40671  1.24739  8.22550  7.91163  0.56170 -0.28738  0.00133  0.00170  0.00049      5  26429       0 ];
c = [ -9.24937  1.27523  8.16308  7.43487  0.55149 -0.30302  0.00133  0.00169  0.00049      6  34860       0 ];
c = [ -9.03985  1.20099  8.18561  7.48589  0.55068 -0.30832  0.00133  0.00167  0.00053     16  82634       0 ];

t = [ 2014 9 23 10 56 ];
%3dStart 14_9_23_10_56
a = [ 0.750000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.81  1.15  7.61  7.00  0.51 -0.32
     -8.53  1.32  8.76  8.055  0.59 -0.28 ];
c = [ -9.42312  1.30126  8.26319  7.97734  0.55032 -0.29754  0.00146  0.00178  0.00049      1   3527       0 ];
c = [ -9.09426  1.19057  8.22824  7.90984  0.56033 -0.29825  0.00144  0.00174  0.00053      1   7576       0 ];
c = [ -8.72873  1.28656  8.16946  7.45770  0.55904 -0.30409  0.00143  0.00173  0.00053      3  18135       0 ];
c = [ -8.69808  1.25926  8.32114  8.12259  0.56097 -0.28678  0.00143  0.00170  0.00062      4  20013       0 ];
c = [ -9.36668  1.25120  8.20907  7.87230  0.55460 -0.29642  0.00140  0.00170  0.00051      4  21431       0 ];
c = [ -8.93381  1.25266  8.16783  7.44730  0.55007 -0.30846  0.00140  0.00169  0.00050      4  21796       0 ];
```

```
c = [ -8.74074  1.26691  8.18428  7.48482  0.55777 -0.30883  0.00138  0.00167  0.00053       9    45101       0 ];
c = [ -9.06500  1.22815  8.17007  7.45162  0.55424 -0.30501  0.00138  0.00167  0.00050       9    45623       0 ];
c = [ -8.80820  1.20708  8.26179  7.99007  0.55845 -0.29721  0.00138  0.00167  0.00051      13    69300       0 ];

t = [ 2014 9 23 16 53 ];
%3dStart 14_9_23_16_53
a = [ 0.800000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.39  1.19  7.82  7.43  0.53 -0.31
     -8.50  1.32  8.64  8.215  0.59 -0.28 ];
c = [ -9.16365  1.24541  8.19869  7.50821  0.54906 -0.30540  0.00146  0.00169  0.00052       1     4328       0 ];
c = [ -8.66566  1.24197  8.27612  8.02124  0.55696 -0.28499  0.00145  0.00169  0.00052       1     6790       0 ];
c = [ -8.52047  1.21216  8.28514  8.04488  0.56235 -0.28630  0.00144  0.00166  0.00054       4    22913       0 ];
c = [ -8.80737  1.23606  8.17774  7.47165  0.55388 -0.30532  0.00143  0.00166  0.00052      17    86746       0 ];

t = [ 2014 9 23 23 11 ];
%3dStart 14_9_23_23_11
a = [ 0.850000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.30  1.14  7.65  7.21  0.52 -0.32
     -8.08  1.31  8.81  8.295  0.60 -0.28 ];
c = [ -8.81247  1.26186  8.57559  7.81971  0.53735 -0.30091  0.00162  0.00180  0.00059       1      663       0 ];
c = [ -9.06421  1.15614  8.19192  7.50191  0.55973 -0.30327  0.00150  0.00166  0.00058       1     8261       0 ];
c = [ -9.21332  1.17473  8.24510  7.95351  0.55159 -0.29026  0.00150  0.00167  0.00050       1     9260       0 ];
c = [ -9.08115  1.18026  8.24475  7.96001  0.55035 -0.29585  0.00150  0.00167  0.00054      10    52544       0 ];
c = [ -8.78444  1.14180  8.30639  8.08533  0.55847 -0.29826  0.00149  0.00166  0.00055      11    55677       0 ];
c = [ -8.89578  1.15011  8.19351  7.50605  0.55321 -0.30137  0.00149  0.00165  0.00058      13    66629       0 ];
c = [ -9.21332  1.12612  8.24510  7.95351  0.55159 -0.29318  0.00149  0.00166  0.00050      14    72540       0 ];

t = [ 2014 9 22 15 19 ];
%3dStart 14_9_22_15_19
a = [ 0.550000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.06  1.27  7.23  6.18  0.50 -0.35
     -9.05  1.56  8.84  7.565  0.61 -0.28 ];
c = [ -10.05873  1.41552  8.03300  6.86868  0.55830 -0.31401  0.00117  0.00172  0.00050       1       1       0 ];
c = [ -9.31051  1.31184  8.16421  7.43744  0.55763 -0.30377  0.00116  0.00170  0.00049       3    18391       0 ];

t = [ 2014 9 22 19 47 ];
%3dStart 14_9_22_19_47
a = [ 0.600000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.23  1.20  7.42  6.72  0.51 -0.33
     -8.54  1.44  8.88  8.045  0.61 -0.28 ];
c = [ -9.25047  1.43868  7.82410  7.50047  0.56830 -0.29672  0.00137  0.00193  0.00052       1      30       0 ];
c = [ -9.96917  1.21460  8.22529  7.54856  0.55379 -0.30705  0.00127  0.00177  0.00051       1     6919       0 ];
c = [ -9.11469  1.39125  8.17540  7.47017  0.55877 -0.30093  0.00125  0.00171  0.00056       1     8647       0 ];
c = [ -9.65486  1.34694  8.13627  7.38966  0.55902 -0.30200  0.00125  0.00172  0.00054       4    23412       0 ];
c = [ -9.41279  1.34580  8.15188  7.41332  0.56209 -0.30223  0.00122  0.00171  0.00049       4    23488       0 ];
c = [ -9.18836  1.28409  8.17005  7.44890  0.56182 -0.30237  0.00122  0.00170  0.00050      11    58934       0 ];
c = [ -9.19483  1.27229  8.16707  7.44508  0.55464 -0.31077  0.00121  0.00169  0.00050      15    75209       0 ];
c = [ -9.13416  1.30371  8.15916  7.42892  0.54926 -0.30953  0.00121  0.00169  0.00050      19    95116       0 ];

t = [ 2014 9 23 0 16 ];
%3dStart 14_9_23_0_16
a = [ 0.650000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.92  1.27  7.75  7.06  0.53 -0.32
     -8.98  1.41  8.57  7.805  0.59 -0.29 ];
c = [ -9.00894  1.28210  8.17903  7.47535  0.55104 -0.31237  0.00128  0.00168  0.00055       1     4521       0 ];
c = [ -9.71601  1.29100  8.13587  7.37465  0.55667 -0.31216  0.00128  0.00171  0.00048       8    40520       0 ];
c = [ -9.65706  1.31498  8.12947  7.36587  0.56247 -0.31013  0.00128  0.00171  0.00048       9    48941       0 ];
c = [ -8.96792  1.29020  8.18762  7.48961  0.56168 -0.30464  0.00128  0.00169  0.00053      15    75855       0 ];
c = [ -9.44884  1.31203  8.14779  7.40286  0.56054 -0.30481  0.00128  0.00170  0.00049      16    82084       0 ];
c = [ -9.61218  1.27986  8.12337  7.35195  0.56153 -0.30697  0.00128  0.00170  0.00049      18    94955       0 ];
c = [ -9.47012  1.25505  8.15355  7.41312  0.55442 -0.30241  0.00127  0.00170  0.00049      20   100406       0 ];

t = [ 2014 9 23 5 27 ];
%3dStart 14_9_23_5_27
a = [ 0.700000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.85  1.24  7.75  7.04  0.53 -0.32
     -8.91  1.37  8.56  7.795  0.59 -0.29 ];
```

```
c = [ -9.38190  1.30100  8.15297  7.41487  0.56108 -0.30676  0.00134  0.00170  0.00049      1      1      0 ];
c = [ -9.40671  1.24739  8.22550  7.91163  0.56170 -0.28738  0.00133  0.00170  0.00049      5  26429      0 ];
c = [ -9.24937  1.27523  8.16308  7.43487  0.55149 -0.30302  0.00133  0.00169  0.00049      6  34860      0 ];
c = [ -9.03985  1.20099  8.18561  7.48589  0.55068 -0.30832  0.00133  0.00167  0.00053     16  82634      0 ];

t = [ 2014 9 23 10 56 ];
%3dStart 14_9_23_10_56
a = [ 0.750000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.81  1.15  7.61  7.00  0.51 -0.32
     -8.53  1.32  8.76  8.055  0.59 -0.28 ];
c = [ -9.42312  1.30126  8.26319  7.97734  0.55032 -0.29754  0.00146  0.00178  0.00049      1   3527      0 ];
c = [ -9.09426  1.19057  8.22824  7.90984  0.56033 -0.29825  0.00144  0.00174  0.00053      1   7576      0 ];
c = [ -8.72873  1.28656  8.16946  7.45770  0.55904 -0.30409  0.00143  0.00173  0.00053      3  18135      0 ];
c = [ -8.69808  1.25926  8.32114  8.12259  0.56097 -0.28678  0.00143  0.00170  0.00062      4  20013      0 ];
c = [ -9.36668  1.25120  8.20907  7.87230  0.55460 -0.29642  0.00140  0.00170  0.00051      4  21431      0 ];
c = [ -8.93381  1.25266  8.16783  7.44730  0.55007 -0.30846  0.00140  0.00169  0.00050      4  21796      0 ];
c = [ -8.74074  1.26691  8.18428  7.48482  0.55777 -0.30883  0.00138  0.00167  0.00053      9  45101      0 ];
c = [ -9.06500  1.22815  8.17007  7.45162  0.55424 -0.30501  0.00138  0.00167  0.00050      9  45623      0 ];
c = [ -8.80820  1.20708  8.26179  7.99007  0.55845 -0.29721  0.00138  0.00167  0.00051     13  69300      0 ];

t = [ 2014 9 23 16 53 ];
%3dStart 14_9_23_16_53
a = [ 0.800000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.39  1.19  7.82  7.43  0.53 -0.31
     -8.50  1.32  8.64  8.215  0.59 -0.28 ];
c = [ -9.16365  1.24541  8.19869  7.50821  0.54906 -0.30540  0.00146  0.00169  0.00052      1   4328      0 ];
c = [ -8.66566  1.24197  8.27612  8.02124  0.55696 -0.28499  0.00145  0.00169  0.00052      1   6790      0 ];
c = [ -8.52047  1.21216  8.28514  8.04488  0.56235 -0.28630  0.00144  0.00166  0.00054      4  22913      0 ];
c = [ -8.80737  1.23606  8.17774  7.47165  0.55388 -0.30532  0.00143  0.00166  0.00052     17  86746      0 ];

t = [ 2014 9 23 23 11 ];
%3dStart 14_9_23_23_11
a = [ 0.850000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.30  1.14  7.65  7.21  0.52 -0.32
     -8.08  1.31  8.81  8.295  0.60 -0.28 ];
c = [ -8.81247  1.26186  8.57559  7.81971  0.53735 -0.30091  0.00162  0.00180  0.00059      1    663      0 ];
c = [ -9.06421  1.15614  8.19192  7.50191  0.55973 -0.30327  0.00150  0.00166  0.00058      1   8261      0 ];
c = [ -9.21332  1.17473  8.24510  7.95351  0.55159 -0.29026  0.00150  0.00167  0.00050      1   9260      0 ];
c = [ -9.08115  1.18026  8.24475  7.96001  0.55035 -0.29585  0.00150  0.00167  0.00054     10  52544      0 ];
c = [ -8.78444  1.14180  8.30639  8.08533  0.55847 -0.29826  0.00149  0.00166  0.00055     11  55677      0 ];
c = [ -8.89578  1.15011  8.19351  7.50605  0.55321 -0.30137  0.00149  0.00165  0.00058     13  66629      0 ];
c = [ -9.21332  1.12612  8.24510  7.95351  0.55159 -0.29318  0.00149  0.00166  0.00050     14  72540      0 ];

t = [ 2014 9 24 9 38 ];
%3dStart 14_9_24_9_38
a = [ 0.850000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.13  1.01  7.42  7.16  0.50 -0.32
     -8.29  1.24  9.07  8.755  0.61 -0.26 ];
c = [ -9.21332  1.12612  8.24510  7.95351  0.55159 -0.29318  0.00149  0.00166  0.00050      1      1      0 ];
c = [ -8.90359  1.05378  8.39952  8.68503  0.56218 -0.27509  0.00149  0.00165  0.00057      4  20529      0 ];
c = [ -9.33688  1.06453  8.24656  7.96013  0.55682 -0.28804  0.00148  0.00165  0.00053      5  27429      0 ];
c = [ -8.93711  1.03219  8.38233  8.63607  0.55124 -0.28271  0.00147  0.00164  0.00051      6  34762      0 ];
c = [ -8.98293  1.03809  8.28362  8.03295  0.55813 -0.29042  0.00146  0.00163  0.00051      9  49983      0 ];
c = [ -8.99438  1.00502  8.36001  8.58849  0.55522 -0.27906  0.00146  0.00162  0.00051     10  53917      0 ];
c = [ -9.19503  0.98899  8.34226  8.55084  0.55920 -0.27451  0.00146  0.00162  0.00051     19  97789      0 ];

t = [ 2014 9 24 13 3 ];
%3dStart 14_9_24_13_3
a = [ 0.900000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.50  0.99  7.91  7.90  0.53 -0.30
     -8.59  1.09  8.74  8.735  0.58 -0.27 ];
c = [ -8.59926  1.02351  8.27687  8.02649  0.55882 -0.29012  0.00154  0.00165  0.00054      1   1869      0 ];
c = [ -8.70471  0.99060  8.28590  8.03752  0.55108 -0.29642  0.00153  0.00164  0.00051      2  12725      0 ];
c = [ -8.88297  1.02388  8.39259  8.66656  0.55344 -0.27756  0.00153  0.00164  0.00054      3  16273      0 ];
c = [ -8.76309  0.95523  8.37678  8.62691  0.55782 -0.27277  0.00150  0.00160  0.00052      4  24804      0 ];

t = [ 2014 9 24 16 17 ];
```

```
%3dStart 14_9_24_16_17
a = [ 0.950000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.42  0.92  7.78  7.96  0.52 -0.30
     -8.18  1.05  8.96  9.165  0.59 -0.26 ];
c = [ -8.56931  0.94196  8.81548  9.09256  0.53618 -0.27779  0.00159  0.00164  0.00065       1     3952       0 ];
c = [ -9.06891  0.91578  8.77141  8.99288  0.53556 -0.28095  0.00156  0.00162  0.00058       4    21017       0 ];
c = [ -9.26102  0.89809  8.36006  8.58189  0.55821 -0.27438  0.00155  0.00160  0.00051      10    51795       0 ];
c = [ -8.68772  0.93568  8.38034  8.64318  0.55970 -0.28176  0.00154  0.00159  0.00056      13    65677       0 ];

t = [ 2014 9 24 19 28 ];
%3dStart 14_9_24_19_28
a = [ 1.000000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.57  0.86  7.84  8.06  0.52 -0.30
     -8.32  0.98  9.02  9.285  0.59 -0.26 ];
c = [ -8.53841  0.88719  8.40297  8.67975  0.55395 -0.27681  0.00166  0.00166  0.00053       1     4814       0 ];
c = [ -8.68736  0.90411  8.39162  8.65502  0.55834 -0.27742  0.00164  0.00164  0.00053       3    17179       0 ];
c = [ -9.14067  0.88074  8.50456  9.35189  0.55565 -0.26753  0.00164  0.00164  0.00053       4    21694       0 ];
c = [ -9.22528  0.96100  8.34422  8.56184  0.55904 -0.27552  0.00162  0.00162  0.00053       4    22672       0 ];
c = [ -8.77956  0.94633  8.37677  8.63719  0.54921 -0.28021  0.00159  0.00159  0.00058       4    24435       0 ];
c = [ -8.92009  0.88035  8.36651  8.60524  0.55632 -0.27301  0.00158  0.00158  0.00052       6    34220       0 ];
c = [ -8.92216  0.88061  8.36509  8.59987  0.55795 -0.27465  0.00158  0.00158  0.00052      10    51507       0 ];

t = [ 2014 9 24 9 23 ];
%1dStart 14_9_25_9_23
a = [ 0.150000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.51  1.30  7.25  6.21  0.50 -0.35
     -9.42  1.59  8.86  7.595  0.61 -0.29 ];
c = [ -10.46567  1.44903  8.05242  6.89827  0.55490 -0.31712  0.00067  0.00175  0.00048       1       1       0 ];
c = [ -11.04108  1.48120  8.09330  7.28303  0.54947 -0.30900  0.00066  0.00178  0.00046       4    21559       0 ];
c = [ -10.77658  1.47530  8.13592  7.36540  0.55321 -0.30702  0.00066  0.00179  0.00046      20   102772       0 ];

t = [ 2014 10 2 14 2 ];
%1dStart 14_10_2_14_2
a = [ 0.200000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.98  0.78  7.35  6.68  0.50 -0.34
     -8.98  0.95  8.98  8.165  0.62 -0.28 ];
c = [ -9.98220  0.86480  8.16140  7.42080  0.55950 -0.30810  0.00071  0.00162  0.00048       1       1       0 ];
c = [ -10.47653  0.86461  8.13473  7.36780  0.55277 -0.31139  0.00071  0.00165  0.00048       2    11259       0 ];
c = [ -10.07339  0.86845  8.11900  7.34047  0.55518 -0.30757  0.00071  0.00162  0.00048       2    13118       0 ];
c = [ -9.89304  0.85193  8.15726  7.41479  0.55050 -0.30818  0.00071  0.00161  0.00048       6    33853       0 ];
c = [ -10.21804  0.88343  8.12531  7.35091  0.56254 -0.30824  0.00071  0.00164  0.00047      14    74564       0 ];

t = [ 2014 10 3 7 5 ];
%3dStart 14_10_3_7_5
a = [ 0.500000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.98  0.78  7.35  6.68  0.50 -0.34
     -8.98  0.95  8.98  8.165  0.62 -0.28 ];
c = [ -9.98220  0.86480  8.16140  7.42080  0.55950 -0.30810  0.00105  0.00162  0.00048       1       1       0 ];
c = [ -9.57051  0.90485  8.16225  7.42891  0.55372 -0.30625  0.00105  0.00161  0.00049       4    23420       0 ];
c = [ -9.95101  0.88038  8.12981  7.36145  0.54874 -0.30328  0.00105  0.00162  0.00048       6    32292       0 ];
c = [ -9.60872  0.85520  8.18374  7.47091  0.56186 -0.30954  0.00105  0.00160  0.00050       8    42282       0 ];
c = [ -9.80468  0.86632  8.13635  7.37820  0.56128 -0.30486  0.00105  0.00160  0.00049       8    43597       0 ];
c = [ -9.90949  0.84964  8.13425  7.37232  0.56090 -0.30272  0.00104  0.00160  0.00049       9    48354       0 ];
c = [ -9.47636  0.89833  8.15754  7.41967  0.55150 -0.29930  0.00104  0.00160  0.00049      14    72165       0 ];
c = [ -9.36243  0.87500  8.16808  7.44229  0.55361 -0.30188  0.00104  0.00158  0.00050      20   102273       0 ];

t = [ 2014 10 3 10 21 ];
%3dStart 14_10_3_10_21
a = [ 0.550000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.26  0.82  7.74  7.04  0.53 -0.32
     -9.28  0.91  8.56  7.785  0.59 -0.29 ];
c = [ -9.76746  0.86751  8.15206  7.40784  0.55800 -0.30592  0.00110  0.00160  0.00049       1       1       0 ];
c = [ -9.78226  0.88044  8.13496  7.37244  0.56241 -0.30478  0.00110  0.00161  0.00049       8    42482       0 ];
c = [ -9.35919  0.86820  8.17841  7.46338  0.54963 -0.30556  0.00110  0.00158  0.00051      13    65649       0 ];

t = [ 2014 10 3 13 39 ];
%3dStart 14_10_3_13_39
```

```
a = [ 0.600000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.96  0.83  7.76  7.07  0.53 -0.32
     -9.01  0.92  8.57  7.815  0.58 -0.29 ];
c = [ -9.48567  0.87149  8.16589  7.43716  0.55333 -0.30536  0.00115  0.00159  0.00050       1       1       0 ];
c = [ -9.60663  0.85928  8.14884  7.40100  0.55692 -0.30500  0.00115  0.00159  0.00049       8   44722       0 ];
c = [ -9.24099  0.84123  8.18513  7.47787  0.55195 -0.30347  0.00114  0.00156  0.000525      20  103563       0 ];

t = [ 2014 10 3 19 14 ];
%3dStart 14_10_3_19_14
a = [ 0.650000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.07  0.82  7.74  7.03  0.53 -0.32
     -9.11  0.90  8.56  7.775  0.58 -0.29 ];
c = [ -9.34705  0.84702  8.17476  7.45589  0.55334 -0.30394  0.00120  0.00157  0.00051       1       1       0 ];
c = [ -9.37413  0.82071  8.17340  7.45505  0.55095 -0.30844  0.00120  0.00156  0.00052      14   71760       0 ];

t = [ 2014 10 4 10 39 ];
%3dStart 14_10_4_10_39
a = [ 0.700000 20.000000 5000.000000 0.010000 0.600000 0.600000 ];
b = [ -9.84  0.78  7.76  7.08  0.52 -0.32
     -8.90  0.86  8.58  7.835  0.58 -0.29 ];
c = [ -9.37143  0.82334  8.17354  7.45514  0.55119 -0.30799  0.00125  0.00156  0.00051       1       1       0 ];
c = [ -9.35001  0.81172  8.17952  7.46601  0.55602 -0.30684  0.00124  0.00156  0.00052      12   62550       0 ];
c = [ -9.39735  0.81406  8.17420  7.45459  0.55314 -0.30107  0.00124  0.00156  0.000515     16   83334       0 ];

t = [ 2014 10 7 14 23 ];
%3gStart 14_10_7_14_23
a = [ 0.150000 5.000000 20000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.23  1.22  7.43  6.76  0.57 -0.31
     -10.51  1.45  7.71  7.025  0.58 -0.30 ];
c = [ -11.19675  1.30942  7.70918  6.92425  0.57476 -0.30494  0.00068  0.00197  0.00045       1   18415      33 ];
c = [ -11.31246  1.27346  7.71412  6.93362  0.57437 -0.30566  0.00067  0.00198  0.00044       1   20627      37 ];
c = [ -10.92839  1.36773  7.73124  6.96943  0.57640 -0.30591  0.00067  0.00198  0.00044       1   27609      50 ];
c = [ -10.79928  1.24523  7.72624  6.96092  0.57119 -0.30560  0.00067  0.00196  0.00045       1   30486      55 ];
c = [ -11.15794  1.25868  7.71975  6.94504  0.57378 -0.30587  0.00067  0.00198  0.00044       1   35776      65 ];
c = [ -11.02624  1.20181  7.73047  6.96585  0.57239 -0.30566  0.00067  0.00197  0.00044       4   96455     181 ];

t = [ 2014 10 7 19 55 ];
%1gStart 14_10_7_19_55
a = [ 0.427500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.28  0.78  7.98  7.26  0.56 -0.31
     -9.80  1.01  8.27  7.525  0.56 -0.31 ];
c = [ -10.13013  0.81633  8.12736  7.35603  0.56062 -0.30826  0.00096  0.00160  0.00048       1     179       0 ];
c = [ -10.03267  0.77157  8.16343  7.42559  0.55987 -0.30818  0.00096  0.00159  0.00049       2    2352       4 ];
c = [ -10.07994  0.75290  8.13772  7.37903  0.55935 -0.30840  0.00096  0.00158  0.00050       3    3118       6 ];
c = [ -10.02338  0.79308  8.13437  7.36973  0.56027 -0.30850  0.00096  0.00159  0.00048       4    4289       8 ];
c = [ -9.88999  0.74385  8.16968  7.43845  0.55794 -0.30830  0.00095  0.00158  0.00049       5    5402      10 ];
c = [ -10.09478  0.71958  8.16676  7.43186  0.56054 -0.30858  0.00095  0.00157  0.00049       5    5661      10 ];
c = [ -9.99320  0.75160  8.13858  7.37999  0.56106 -0.30864  0.00095  0.00157  0.00049       5    5806      11 ];
c = [ -9.98756  0.76109  8.15601  7.41157  0.55766 -0.30815  0.00095  0.00158  0.00049       6    6315      12 ];
c = [ -9.80559  0.73218  8.15873  7.42108  0.56184 -0.30813  0.00095  0.00155  0.00051       6    6412      12 ];
c = [ -10.03398  0.70327  8.16737  7.43294  0.56091 -0.30865  0.00095  0.00156  0.00049       6    6875      13 ];
c = [ -10.10036  0.70296  8.14322  7.38752  0.56108 -0.30853  0.00095  0.00155  0.00049       7    7393      14 ];
c = [ -10.14266  0.67381  8.15323  7.40577  0.55977 -0.30864  0.00094  0.00154  0.00049       8    8207      15 ];

t = [ 2014 10 7 20 16 ];
%3gStart 14_10_7_20_16
a = [ 0.405000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.34  0.81  7.95  7.23  0.56 -0.31
     -9.86  1.03  8.23  7.495  0.57 -0.31 ];
c = [ -10.01224  0.81761  8.15171  7.40463  0.55950 -0.30830  0.00094  0.00160  0.00048       1     815       1 ];
c = [ -10.15764  0.79803  8.14226  7.38465  0.55988 -0.30801  0.00094  0.00160  0.00048       3    3952       7 ];
c = [ -10.16490  0.75733  8.14468  7.38877  0.56005 -0.30801  0.00093  0.00159  0.00048       4    4799       9 ];
c = [ -10.12871  0.70927  8.15029  7.39863  0.55803 -0.30814  0.00092  0.00156  0.00049       5    5844      11 ];
c = [ -10.00683  0.71525  8.14004  7.38001  0.55751 -0.30815  0.00092  0.00155  0.00049      10   10332      19 ];

t = [ 2014 10 7 20 36 ];
%1gStart 14_10_7_20_36
```

```
a = [ 0.382500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.40  0.84  7.91  7.20  0.56 -0.31
     -9.92  1.06  8.20  7.455  0.57 -0.31 ];
c = [ -10.38412  0.93765  8.11626  7.33010  0.56218 -0.30809  0.00093  0.00167  0.00048      1     226       0 ];
c = [ -10.08738  0.87788  8.11978  7.34226  0.56188 -0.30831  0.00092  0.00162  0.00048      1    1603       3 ];
c = [ -9.94520  0.81976  8.16430  7.42677  0.56129 -0.30813  0.00092  0.00161  0.00049      2    2514       5 ];
c = [ -10.12585  0.83564  8.13238  7.36323  0.56252 -0.30812  0.00092  0.00161  0.00048      2    2723       5 ];
c = [ -9.92557  0.81876  8.13664  7.37191  0.56145 -0.30818  0.00092  0.00159  0.00050      4    4166       8 ];
c = [ -10.12585  0.81708  8.13238  7.36323  0.55995 -0.30817  0.00091  0.00161  0.00049      5    5165       9 ];
c = [ -10.18704  0.75984  8.14975  7.39643  0.56088 -0.30796  0.00091  0.00159  0.00049      6    6128      11 ];
c = [ -9.96715  0.81019  8.13615  7.37416  0.56208 -0.30804  0.00091  0.00159  0.00048      6    6198      11 ];
c = [ -10.05098  0.79405  8.12678  7.35547  0.56152 -0.30788  0.00091  0.00159  0.00048      8    8951      16 ];

t = [ 2014 10 7 20 56 ];
%3gStart 14_10_7_20_56
a = [ 0.360000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.46  0.86  7.88  7.16  0.56 -0.31
     -9.98  1.09  8.16  7.425  0.57 -0.31 ];
c = [ -10.13347  0.90550  8.11710  7.34168  0.56078 -0.30780  0.00091  0.00164  0.00050      1     958       2 ];
c = [ -10.29141  0.99551  8.10331  7.30855  0.56112 -0.30790  0.00090  0.00167  0.00047      1    1716       3 ];
c = [ -10.11854  0.98709  8.11680  7.33560  0.55968 -0.30771  0.00090  0.00166  0.00048      3    3589       7 ];
c = [ -10.13412  0.90424  8.15046  7.39907  0.55948 -0.30796  0.00090  0.00165  0.00048      4    4438       8 ];
c = [ -10.35120  0.88236  8.13124  7.36092  0.56194 -0.30765  0.00090  0.00165  0.00047      5    5930      11 ];
c = [ -10.23293  0.91913  8.11029  7.32216  0.56162 -0.30787  0.00090  0.00165  0.00048      7    7365      13 ];
c = [ -10.17554  0.87044  8.15232  7.40371  0.56215 -0.30807  0.00090  0.00164  0.00048      7    7679      14 ];
c = [ -10.17524  0.86015  8.10933  7.32005  0.56214 -0.30767  0.00089  0.00162  0.00048      7    7832      14 ];
c = [ -10.14181  0.85817  8.12999  7.36024  0.56146 -0.30756  0.00089  0.00162  0.00048      9    9180      17 ];

t = [ 2014 10 7 21 16 ];
%3gStart 14_10_7_21_16
a = [ 0.337500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.52  0.89  7.84  7.13  0.56 -0.31
     -10.04  1.12  8.13  7.395  0.57 -0.31 ];
c = [ -10.32208  0.96524  8.09269  7.28843  0.56220 -0.30760  0.00088  0.00166  0.00048      1     412       1 ];
c = [ -10.21977  0.93362  8.10297  7.31135  0.56214 -0.30755  0.00087  0.00165  0.00048      2    2801       5 ];
c = [ -10.23602  0.95036  8.10383  7.30975  0.56153 -0.30768  0.00087  0.00165  0.00048      3    3302       6 ];
c = [ -10.06551  0.92026  8.14655  7.39276  0.56159 -0.30745  0.00087  0.00165  0.00048      4    4522       8 ];
c = [ -10.10663  0.82994  8.13227  7.36263  0.56072 -0.30732  0.00087  0.00161  0.00049      7    7107      13 ];
c = [ -10.24051  0.84918  8.13653  7.37332  0.56130 -0.30761  0.00087  0.00163  0.00048      9    9699      18 ];
c = [ -9.99205   0.85644  8.12566  7.35318  0.55909 -0.30782  0.00086  0.00161  0.00048     10   10795      20 ];

t = [ 2014 10 7 21 36 ];
%3gStart 14_10_7_21_36
a = [ 0.315000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.58  0.92  7.81  7.10  0.56 -0.31
     -10.10  1.15  8.09  7.365  0.57 -0.31 ];
c = [ -10.38514  1.03839  8.08276  7.26882  0.56170 -0.30742  0.00086  0.00170  0.00048      1     435       1 ];
c = [ -10.37601  1.01353  8.12062  7.34238  0.55979 -0.30773  0.00086  0.00169  0.00047      1    1064       2 ];
c = [ -10.55169  0.96567  8.08164  7.26573  0.55998 -0.30780  0.00085  0.00168  0.00047      2    2432       4 ];
c = [ -10.40668  0.92911  8.11084  7.32192  0.55968 -0.30765  0.00085  0.00166  0.00047      3    3446       6 ];
c = [ -10.23086  0.91333  8.12930  7.35934  0.56213 -0.30734  0.00084  0.00165  0.00047      5    5287      10 ];
c = [ -10.29206  0.88314  8.10042  7.30298  0.56070 -0.30732  0.00084  0.00164  0.00048      6    6057      11 ];
c = [ -10.41562  0.85725  8.10471  7.31151  0.56010 -0.30790  0.00084  0.00164  0.00048      8    8674      16 ];

t = [ 2014 10 7 21 57 ];
%1gStart 14_10_7_21_57
a = [ 0.292500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.63  0.95  7.77  7.07  0.56 -0.31
     -10.16  1.17  8.05  7.325  0.57 -0.31 ];
c = [ -10.34893  0.96424  7.78451  7.08030  0.56521 -0.30751  0.00089  0.00192  0.00046      1     378       1 ];
c = [ -10.43553  0.94225  7.77969  7.06774  0.56550 -0.30739  0.00088  0.00192  0.00045      1    1925       4 ];
c = [ -10.41640  0.95059  7.75955  7.02693  0.56540 -0.30774  0.00088  0.00191  0.00045      2    2455       5 ];
c = [ -10.42502  0.93167  7.73021  6.97306  0.56477 -0.30776  0.00088  0.00189  0.00045      3    3862       7 ];
c = [ -10.19264  0.97205  7.77784  7.06646  0.56387 -0.30762  0.00088  0.00191  0.00045      4    4253       8 ];
c = [ -10.28581  0.93920  7.77177  7.05319  0.56567 -0.30752  0.00087  0.00190  0.00045      6    6432      12 ];
c = [ -10.12001  0.91687  7.77210  7.05634  0.56765 -0.30769  0.00087  0.00188  0.00045      7    7015      13 ];
c = [ -10.14030  0.91396  7.77428  7.05958  0.56555 -0.30775  0.00087  0.00188  0.00045      9    9905      18 ];
```

```
c = [ -10.40584  0.87617  7.75816  7.02613  0.56441 -0.30732  0.00087  0.00189  0.00045      10    10450       20 ];

t = [ 2014 10 7 22 17 ];
%3gStart 14_10_7_22_17
a = [ 0.270000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.69  0.98  7.74  7.04  0.56 -0.31
     -10.22  1.20  8.02  7.295  0.57 -0.31 ];
c = [ -10.41583  1.13293  7.80728  7.12003  0.56729 -0.30735  0.00086  0.00197  0.00045       1     185       0 ];
c = [ -10.52895  1.16816  7.77512  7.05883  0.56701 -0.30740  0.00086  0.00197  0.00045       1    1127       2 ];
c = [ -10.27617  1.01311  7.75684  7.02503  0.56984 -0.30720  0.00084  0.00191  0.00045       1    1133       2 ];
c = [ -10.31071  0.97851  7.77191  7.05468  0.56673 -0.30759  0.00084  0.00192  0.00045       3    3126       6 ];
c = [ -10.33053  1.00779  7.76110  7.03220  0.57048 -0.30772  0.00084  0.00192  0.00045       4    4925       9 ];
c = [ -10.50011  0.98858  7.75775  7.02434  0.57199 -0.30735  0.00084  0.00193  0.00044       5    5837      11 ];
c = [ -10.24255  0.93645  7.78192  7.07307  0.57198 -0.30739  0.00084  0.00190  0.00045       7    7302      13 ];
c = [ -10.43651  0.92685  7.75442  7.01885  0.56533 -0.30763  0.00084  0.00190  0.00045       8    8128      15 ];
c = [ -10.25066  0.92314  7.75007  7.01145  0.56879 -0.30761  0.00084  0.00189  0.00045       8    8192      15 ];
c = [ -10.40941  0.90456  7.75445  7.01875  0.56548 -0.30754  0.00084  0.00190  0.00045       8    8981      16 ];

t = [ 2014 10 7 22 37 ];
%3gStart 14_10_7_22_37
a = [ 0.247500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.75  1.01  7.70  7.00  0.56 -0.31
     -10.28  1.23  7.98  7.265  0.57 -0.31 ];
c = [ -10.32696  1.04275  7.76561  7.04410  0.56656 -0.30729  0.00082  0.00193  0.00045       1     783       1 ];
c = [ -10.32925  1.03984  7.76757  7.04201  0.56635 -0.30740  0.00082  0.00193  0.00045       1    1193       2 ];
c = [ -10.47452  1.13318  7.75391  7.01626  0.56518 -0.30745  0.00081  0.00195  0.00044       2    2269       4 ];
c = [ -10.39941  1.01355  7.77148  7.05025  0.56992 -0.30758  0.00081  0.00193  0.00044       2    2429       4 ];
c = [ -10.26416  1.04141  7.77895  7.06718  0.57084 -0.30742  0.00081  0.00193  0.00044       5    5372      10 ];
c = [ -10.60575  1.07227  7.75107  7.00970  0.57065 -0.30739  0.00081  0.00195  0.00044       5    5674      10 ];
c = [ -10.48669  1.02967  7.75556  7.01912  0.56458 -0.30758  0.00081  0.00193  0.00044       5    5792      11 ];
c = [ -10.26470  0.97194  7.76759  7.04669  0.56637 -0.30749  0.00081  0.00191  0.00045       6    6180      11 ];
c = [ -10.34968  0.98559  7.76352  7.03743  0.56822 -0.30736  0.00081  0.00192  0.00045       7    7956      15 ];
c = [ -10.38256  0.95305  7.75551  7.02003  0.56824 -0.30721  0.00081  0.00191  0.00045      10   10188      19 ];

t = [ 2014 10 7 22 58 ];
%3gStart 14_10_7_22_58
a = [ 0.225000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.81  1.03  7.66  6.97  0.57 -0.31
     -10.34  1.26  7.95  7.235  0.57 -0.31 ];
c = [ -10.73721  1.08172  7.76167  7.03045  0.57178 -0.30745  0.00078  0.00196  0.00044       1     213       0 ];
c = [ -10.58387  1.07091  7.74868  7.00771  0.56852 -0.30703  0.00078  0.00194  0.00045       1    1927       3 ];
c = [ -10.55564  1.02511  7.76771  7.04290  0.56621 -0.30737  0.00078  0.00194  0.00045       2    2041       4 ];
c = [ -10.64314  1.06504  7.74995  7.00941  0.56776 -0.30710  0.00078  0.00195  0.00044       2    2167       4 ];
c = [ -10.69393  1.09615  7.74085  6.99005  0.56962 -0.30724  0.00078  0.00195  0.00044       2    2276       4 ];
c = [ -10.44405  1.10760  7.76578  7.03992  0.57017 -0.30682  0.00078  0.00195  0.00044       2    2874       5 ];
c = [ -10.67321  1.03451  7.75355  7.01613  0.56925 -0.30729  0.00078  0.00195  0.00044       3    3121       6 ];
c = [ -10.44786  1.03816  7.76268  7.03386  0.56726 -0.30740  0.00078  0.00193  0.00044       3    3971       7 ];
c = [ -10.74657  1.02839  7.74788  7.00358  0.56719 -0.30699  0.00078  0.00195  0.00044       5    5271      10 ];
c = [ -10.78078  1.00865  7.74118  6.98831  0.56934 -0.30726  0.00078  0.00194  0.00044       7    7643      14 ];
c = [ -10.65552  0.98825  7.75543  7.01787  0.56924 -0.30691  0.00078  0.00194  0.00044      10   10025      18 ];

t = [ 2014 10 7 23 18 ];
%1gStart 14_10_7_23_18
a = [ 0.202500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.87  1.06  7.63  6.94  0.57 -0.31
     -10.40  1.29  7.91  7.205  0.57 -0.31 ];
c = [ -10.63004  1.10146  7.72596  6.96179  0.56930 -0.30708  0.00075  0.00194  0.00045       1     196       0 ];
c = [ -10.43273  1.08080  7.74694  7.00465  0.56913 -0.30689  0.00075  0.00193  0.00045       1    1372       2 ];
c = [ -10.47115  1.07284  7.75470  7.01731  0.56966 -0.30717  0.00075  0.00194  0.00044       2    2550       5 ];
c = [ -10.39664  1.08803  7.75656  7.02294  0.57290 -0.30672  0.00075  0.00194  0.00044       3    3426       6 ];
c = [ -10.49755  1.02538  7.74619  7.00134  0.56884 -0.30707  0.00074  0.00193  0.00044       3    3446       7 ];
c = [ -10.54071  0.98932  7.74837  7.00548  0.57148 -0.30681  0.00074  0.00192  0.00044       7    7783      15 ];

t = [ 2014 10 7 23 38 ];
%3gStart 14_10_7_23_38
a = [ 0.180000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.93  1.09  7.59  6.91  0.57 -0.31
```

```
    -10.46  1.31  7.88  7.165  0.58 -0.31 ];
c = [ -10.69418  1.15165  7.77456  7.05182  0.56877 -0.30692  0.00072  0.00197  0.00045      1     65      0 ];
c = [ -10.50333  1.09136  7.75041  7.01243  0.57296 -0.30693  0.00072  0.00194  0.00045      1   1030      2 ];
c = [ -10.86223  1.14776  7.73337  6.97444  0.57125 -0.30719  0.00071  0.00196  0.00044      1   1316      2 ];
c = [ -10.69194  1.11726  7.74773  7.00105  0.56789 -0.30739  0.00071  0.00196  0.00044      3   3137      6 ];
c = [ -10.87757  1.09425  7.72832  6.96334  0.57125 -0.30720  0.00071  0.00196  0.00044      3   3353      6 ];
c = [ -10.90138  1.10520  7.73548  6.97709  0.57326 -0.30715  0.00071  0.00196  0.00044      3   3852      7 ];
c = [ -10.89740  1.09819  7.73594  6.97759  0.56842 -0.30700  0.00071  0.00196  0.000445     6   6918     13 ];

t = [ 2014 10 7 23 59 ];
%1gStart 14_10_7_23_59
a = [ 0.157500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -10.99  1.12  7.56  6.87  0.57 -0.31
    -10.52  1.34  7.84  7.135  0.58 -0.31 ];
c = [ -10.97574  1.21875  7.72914  6.96524  0.57405 -0.30669  0.00068  0.00197  0.00044      1    119      0 ];
c = [ -10.87580  1.15638  7.72614  6.95940  0.57060 -0.30670  0.00068  0.00196  0.00044      1   1130      2 ];
c = [ -10.86994  1.22930  7.73864  6.98225  0.56815 -0.30681  0.00068  0.00197  0.00044      2   2608      5 ];
c = [ -10.73285  1.08332  7.73936  6.98552  0.57364 -0.30665  0.00068  0.00195  0.00044      2   2856      5 ];
c = [ -10.92365  1.09673  7.73457  6.97444  0.57173 -0.30688  0.00068  0.00196  0.00043      2   2984      5 ];

t = [ 2014 10 8 0 19 ];
%3gStart 14_10_8_0_19
a = [ 0.135000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.05  1.15  7.52  6.84  0.57 -0.31
    -10.58  1.37  7.81  7.105  0.58 -0.31 ];
c = [ -10.61557  1.17295  7.76082  7.02784  0.57439 -0.30702  0.00065  0.00196  0.00044      1    243      0 ];
c = [ -10.56068  1.20306  7.74363  6.99631  0.57130 -0.30680  0.00065  0.00196  0.00044      1   1158      2 ];
c = [ -10.68148  1.18138  7.74784  7.00307  0.57301 -0.30684  0.00064  0.00196  0.00044      1   1940      3 ];
c = [ -10.73003  1.14487  7.74819  7.00365  0.57154 -0.30691  0.00064  0.00196  0.00044      2   2412      4 ];
c = [ -10.74441  1.11176  7.74167  6.99020  0.57055 -0.30672  0.00064  0.00196  0.00044      6   6630     12 ];
c = [ -10.68011  1.12556  7.75027  7.00794  0.57448 -0.30720  0.00064  0.00196  0.00044      7   7898     14 ];

t = [ 2014 10 8 0 39 ];
%3gStart 14_10_8_0_39
a = [ 0.112500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.11  1.17  7.49  6.81  0.57 -0.31
    -10.63  1.40  7.77  7.075  0.58 -0.31 ];
c = [ -10.87219  1.20225  7.76031  7.02608  0.57405 -0.30691  0.00061  0.00198  0.00044      1    212      0 ];
c = [ -11.04429  1.15845  7.75538  7.01509  0.57498 -0.30654  0.00061  0.00198  0.00044      3   3347      6 ];
c = [ -11.03409  1.21459  7.72521  6.95647  0.57146 -0.30647  0.00061  0.00197  0.00044      3   3478      6 ];
c = [ -10.93178  1.15845  7.75538  7.01509  0.57155 -0.30663  0.00061  0.00198  0.00044      4   4477      8 ];
c = [ -10.90670  1.29782  7.74050  6.98668  0.57122 -0.30639  0.00061  0.00198  0.00044      4   4699      9 ];
c = [ -11.02234  1.16740  7.72621  6.95708  0.57205 -0.30650  0.00061  0.00197  0.00044      5   5482     10 ];
c = [ -10.94005  1.18107  7.73532  6.97605  0.57490 -0.30649  0.00061  0.00197  0.00044      7   7764     14 ];
c = [ -11.06969  1.14552  7.72621  6.95708  0.57249 -0.30661  0.00061  0.00197  0.000445     9   9762     18 ];

t = [ 2014 10 8 0 59 ];
%3gStart 14_10_8_0_59
a = [ 0.090000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.17  1.20  7.45  6.78  0.57 -0.31
    -10.69  1.43  7.74  7.045  0.58 -0.31 ];
c = [ -11.01166  1.22181  7.72999  6.96643  0.57448 -0.30642  0.00058  0.00197  0.00044      1    649      1 ];
c = [ -10.90444  1.30891  7.76619  7.03566  0.57359 -0.30673  0.00058  0.00199  0.00044      7   7138     13 ];
c = [ -10.97294  1.29411  7.73564  6.97668  0.57242 -0.30643  0.00058  0.00198  0.00044      8   8232     15 ];
c = [ -10.88611  1.22961  7.73471  6.97516  0.57298 -0.30656  0.00058  0.00198  0.000445     9   9434     17 ];

t = [ 2014 10 8 1 19 ];
%3gStart 14_10_8_1_19
a = [ 0.067500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.23  1.23  7.42  6.75  0.57 -0.31
    -10.75  1.45  7.70  7.005  0.58 -0.31 ];
c = [ -10.98877  1.45305  7.45873  6.79425  0.57855 -0.30679  0.00054  0.00222  0.00042      1      5      0 ];
c = [ -10.99205  1.35658  7.45170  6.78050  0.57930 -0.30658  0.00054  0.00221  0.00042      1   1497      3 ];
c = [ -11.08368  1.44725  7.46296  6.80164  0.57716 -0.30621  0.00054  0.00223  0.00042      3   3242      6 ];
c = [ -11.00899  1.36733  7.45629  6.78972  0.57872 -0.30660  0.00054  0.00222  0.00042      4   4181      8 ];
c = [ -11.05912  1.35596  7.44732  6.77103  0.57919 -0.30636  0.00054  0.00221  0.00042      5   5931     11 ];
c = [ -11.12051  1.34373  7.45438  6.78471  0.57699 -0.30646  0.00054  0.00222  0.00042      8   8196     15 ];
```

```
c = [ -11.24648  1.33261  7.44127  6.75740  0.57839 -0.30628  0.00054  0.00222  0.000425      10    10236       19 ];

t = [ 2014 10 8 1 40 ];
%3gStart 14_10_8_1_40
a = [ 0.045000 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.29  1.26  7.38  6.71  0.57 -0.31
     -10.81  1.48  7.66  6.975  0.58 -0.31 ];
c = [ -11.14386  1.34707  7.43722  6.75111  0.57830 -0.30658  0.00050  0.00221  0.00042       1      39       0 ];
c = [ -10.99636  1.40385  7.46767  6.81193  0.58137 -0.30679  0.00050  0.00223  0.00042       2    2152       4 ];
c = [ -11.09381  1.36165  7.44849  6.77262  0.57912 -0.30641  0.00050  0.00222  0.00042       4    4020       7 ];
c = [ -11.14296  1.32583  7.45079  6.77618  0.57980 -0.30679  0.00050  0.00222  0.00042       5    5496      10 ];
c = [ -11.20915  1.41041  7.45432  6.78300  0.58128 -0.30660  0.00050  0.00223  0.00042       6    6254      11 ];
c = [ -11.24754  1.29072  7.43928  6.75277  0.57906 -0.30626  0.00050  0.00223  0.00042       6    6437      12 ];

t = [ 2014 10 8 2 0 ];
%1gStart 14_10_8_2_0
a = [ 0.022500 10.000000 1000.000000 0.010000 0.600000 0.600000 ];
b = [ -11.35  1.29  7.34  6.68  0.57 -0.31
     -10.87  1.51  7.63  6.945  0.58 -0.31 ];
c = [ -11.29881  1.37184  7.45750  6.78906  0.58183 -0.30647  0.00046  0.00223  0.00042       1     409       1 ];
c = [ -11.27247  1.40942  7.45507  6.78404  0.57673 -0.30619  0.00046  0.00223  0.00042       5    5049       9 ];
c = [ -11.32583  1.29975  7.44095  6.75631  0.57886 -0.30631  0.00046  0.00222  0.00042       8    8739      16 ];
c = [ -11.37670  1.36653  7.44861  6.76992  0.57910 -0.30624  0.00046  0.00223  0.00041       9    9291      17 ];
c = [ -11.34649  1.27841  7.43822  6.74952  0.57881 -0.30659  0.00046  0.00222  0.00041       9    9862      18 ];
```