Optimal Low-Thrust Rendezvous using a Hybrid Evolutionary Strategy

Christopher J. Scott **Denise L. Brown** **Peter M. Cipollo**

Dept. of Aerospace Engineering
Pennsylvania State University
University Park, PA 16802

Abstract

This paper develops a hybrid evolutionary algorithm combining the Covariance Matrix Adaptation evolutionary strategy (CMA-ES) with Matlab's *fsolve* local gradient search algorithm to robustly solve the low-thrust rendezvous problem. The nonlinear equations of relative motion govern spacecraft trajectory instead of the more common, linearized Clohessey-Wiltshire equations. The hybrid algorithm solved the unconstrained problem with a reliability of 99% and demonstrated the ability to converge to a solution faster than using either the evolutionary strategy or the local gradient search method alone.

1 INTRODUCTION

Traditional maneuver design using impulsive burns is not applicable to low-thrust applications, which utilize low-power, continuously thrusting engines. Low thrust propulsion systems are characterized by variable exhaust velocity and limited power, imposing additional constraints upon guidance maneuvers. One of the most common guidance maneuvers is rendezvous, where one object in space approaches another object until the final position and velocity of one relative to the other is zero.

Lembeck and Prussing (1993) studied unbounded, low-thrust rendezvous to return to an initial point after an impulsive burn carried the spacecraft away from its initial orbit. The low-thrust return minimized fuel consumption. The bounded rendezvous problem, with upper and lower bounds on thrust acceleration magnitude, was analyzed by Carter and Pardis (1996), who used Newton's method to numerically solve the nonlinear equations. Guelman and Aleshin (2001) further constrained the problem by specifying a final approach direction.

Standard numerical approaches have been applied to solving the nonlinear equations of motion describing the rendezvous problem. Igarashi (2004) applied both an evolutionary strategy and a simple genetic algorithm to the optimal continuous thrust orbit transfer problem, but did not assess reliability of the algorithms.

This research proposes the use of a hybrid evolutionary strategy to solve both the unconstrained and the constrained minimum fuel, continuous thrust rendezvous maneuver.

2 PROBLEM FORMULATION

2.1 LINEARIZED EQUATIONS OF RELATIVE MOTION

Generally, rendezvous is performed from a state close to the final state and the familiar Clohessey-Wiltshire (CW) equations of relative motion are used to describe the motion of one spacecraft relative to another. These are simplified, linearized equations of motion formulated by assuming circular reference orbits and small separations between the 'chase' vehicle and the reference spacecraft (Schaub and Junkins, 2003). The linearized equations of relative motion for a chaser spacecraft approaching a target spacecraft are shown below

$$\begin{aligned} \ddot{x} - 2n\dot{y} - 3n^2 x + \Gamma_x &= 0 \\ \ddot{y} + 2n\dot{x} + \Gamma_y &= 0 \\ \ddot{z} + n^2 z + \Gamma_z &= 0 \end{aligned} \qquad (2.1.1)$$

where x is the radial component of the chaser spacecraft position relative to the reference craft, z is the out-of-plane component, and y is the along-track component which completes the right handed coordinate system. The thrust acceleration vector is represented by Γ and n is the mean motion of the chief satellite. The performance index for minimizing fuel is given by

$$J = \frac{1}{2}\int_{t_0}^{t_f} \Gamma^T \Gamma \, dt \qquad (2.1.2)$$

where t_0 and t_f are fixed initial and final times.

Using classical optimal control theory, it can be shown that for unbounded thrust acceleration, the state and adjoint equations are linear and the adjoint equations are uncoupled from the state equations. Solving these equations yields an optimal thrust control vector of

$$\Gamma^* = -\Phi_{\lambda v}\vec{\lambda}_0 \tag{2.1.3}$$

where $\vec{\lambda}_0$ is the adjoint initial condition vector and $\Phi_{\lambda v}$ is a submatrix of the adjoint transition matrix. The solution for the state vector resulting from the optimal thrust acceleration vector is given by

$$\vec{x}(t) = \Phi(t - t_0)\vec{x}(t_0) + \Psi(t - t_0)\vec{\lambda}_0 \tag{2.14}$$

where Φ is the state transition matrix and Ψ is the convolution integral for the state vector, $\vec{x}$, due to the optimal thrust acceleration vector. The initial adjoint vector is defined such that the boundary conditions are fulfilled:

$$\vec{\lambda}_0 = \Psi^{-1}(t_f - t_0)\left[\vec{x}(t_f) - \Phi(t_f - t_0)\vec{x}(t_0)\right] \tag{2.1.5}$$

Note that the final state $\vec{x}(t_f)$ is equal to the zero vector for docking type rendezvous, but it may also be some other user specified final relative state. For more details, refer to Guelman and Aleshin (2001).

These linear equations provide theoretical insight, yet are of limited use in real-world applications. Therefore, one of the goals of this study is to find a reliable way to solve for the costates in the nonlinear problem.

2.2 NONLINEAR EQUATIONS OF RELATIVE MOTION

The linearized equations of motion are only accurate for small separation distances between the reference and chaser satellites and circular reference orbits. Most real world applications require the greater accuracy of the nonlinear equations of relative motion from which the linearized equations are derived.

Assuming there are no disturbances acting on the satellites and a circular reference orbit, the deputy satellite orbital equations in the rotating relative reference frame used in Section 2.1 are

$$\begin{aligned}
\ddot{x} - 2n\dot{y} - n^2 x - \frac{\mu}{r_c^2} &= -\frac{\mu}{r_d^3}(r_c + x) + \Gamma_x \\
\ddot{y} + 2n\dot{x} - n^2 y &= -\frac{\mu}{r_d^3} y + \Gamma_y \\
\ddot{z} &= -\frac{\mu}{r_d^3} z + \Gamma_z
\end{aligned} \tag{2.2.1}$$

where r_c is the radius of the reference orbit. The radius of the chaser satellite orbit is designated r_d and is equal to:

$$r_d = \sqrt{(r_c + x)^2 + y^2 + z^2}$$

in the relative frame. These equations can be simplified by working in canonical units. Normalizing the distances so that $r_c = 1$ and defining the gravitational constant such that $\mu = 1$ yields the following nonlinear equations of motion

$$\begin{aligned}
\ddot{x} - 2\dot{y} - x - 1 &= -\frac{(1 + x)}{r_d^3} + \Gamma_x \\
\ddot{y} + 2\dot{x} - y &= -\frac{y}{r_d^3} + \Gamma_y \\
\ddot{z} &= -\frac{z}{r_d^3} + \Gamma_z
\end{aligned} \tag{2.2.2}$$

where

$$r_d = \sqrt{(1 + x)^2 + y^2 + z^2}$$

These equations are not subject to the constraint of small separations between the reference and chaser satellite. If the same cost function as that in Section 1.1 is still used, the Hamiltonian for the nonlinear system is

$$\begin{aligned}
H = {} & \frac{1}{2}(\Gamma \cdot \Gamma) + \lambda_1\dot{x} + \lambda_2\dot{y} + \lambda_3\dot{z} + \\
& \left(1 + \Gamma_x + x + 2\dot{y} - \frac{1 + x}{r_d^3}\right)\lambda_4 + \\
& \left(\Gamma_y - 2\dot{x} + y - \frac{y}{r_d^3}\right)\lambda_5 + \left(\Gamma_z - \frac{z}{r_d^3}\right)\lambda_6
\end{aligned} \tag{2.2.3}$$

where Γ is the thrust vector. The six costate equations are therefore

$$\begin{aligned}
\dot{\lambda}_1 &= -\frac{\left(r_d^5 - r_d + 3(1 + x)^2\lambda_4 - 3(1 + x)(y\lambda_5 + z\lambda_6)\right)}{r_d^5} \\
\dot{\lambda}_2 &= \frac{r_d\lambda_5 - r_d^5\lambda_5 - 3y(\lambda_4 + x\lambda_4 + y\lambda_5 + z\lambda_6)}{r_d^5} \\
\dot{\lambda}_3 &= \frac{r_d\lambda_6 - 3z(\lambda_4 + x\lambda_4 + y\lambda_5 + z\lambda_6)}{r_d^5} \\
\dot{\lambda}_4 &= -\lambda_1 \\
\dot{\lambda}_5 &= -\lambda_2 \\
\dot{\lambda}_6 &= -\lambda_3
\end{aligned} \tag{2.2.4}$$

It can be easily seen that, for the nonlinear equations of relative motion, the state and costate (adjoint) equations

are coupled and must be solved simultaneously. The optimal thrust vector is then given by:

$$\Gamma^* = \begin{bmatrix} \Gamma_x & \Gamma_y & \Gamma_z \end{bmatrix}^T = \begin{bmatrix} -\lambda_4 & -\lambda_5 & -\lambda_6 \end{bmatrix}^T \quad (2.2.5)$$

Thus, given an initial state for the chaser spacecraft

$$\vec{x}(t_0) = \begin{bmatrix} x_0 & y_0 & z_0 & \dot{x}_0 & \dot{y}_0 & \dot{z}_0 \end{bmatrix}^T$$

and the final conditions for rendezvous (i.e. relative position and velocity of the chase vehicle are zero), it is possible to calculate the thrust vector profile by solving the two point boundary value problem, where the initial costate is unknown.

Due to the highly nonlinear nature of the equations, it is hypothesized that traditional solvers will be quite unreliable for this problem.

2.3 CONSTRAINTS

The unconstrained problems formulated in the previous sections assume the thrusters on the chaser spacecraft are capable of outputting an infinite amount of thrust. In reality, maximum (and sometimes minimum) limits for thrust exist, with the limits set by the specifications of the spacecraft. This paper will consider only a maximum bound on thrust acceleration, i.e. thrust acceleration vector magnitude is constrained to be below some maximum allowable level.

$$\left\| \Gamma \right\| \le \Gamma_{\max} \quad (2.3.1)$$

For both the linear and nonlinear constrained problem, then relationship of equation 2.2.5 holds if the magnitude of the thrust vector is less than Γ_{max}. However, an additional condition on thrust is needed for the constrained problem, and the optimal thrust can be expressed as

$$\Gamma^*(t) = \begin{cases} -\vec{\lambda}_v(t) & \left| -\vec{\lambda}_v(t) \right| \le \Gamma_{\max} \\ -\dfrac{\vec{\lambda}_v(t)}{\left| \vec{\lambda}_v(t) \right|} \Gamma_{\max} & \left| -\vec{\lambda}_v(t) \right| > \Gamma_{\max} \end{cases} \quad (2.3.2)$$

where

$$\vec{\lambda}_v = \begin{bmatrix} \lambda_4 & \lambda_5 & \lambda_6 \end{bmatrix}^T.$$

The equations of motion for the constrained linear and nonlinear problem must use equation 2.3.2 to calculate the value of the thrust acceleration vector at each time step.

In general, the constrained rendezvous problem is difficult to solve and there is no guarantee a solution exists. Once a maximum bound is placed upon the thrust acceleration magnitude, the chaser spacecraft may not be able to reach the target spacecraft within the given amount of time.

One goal of this paper is to ascertain whether it is possible to solve the constrained rendezvous problem more reliably using heuristic algorithms.

3 ALGORITHMS

The standard approach for finding the thrust vector profile during rendezvous maneuvers is to use the analytical solution to the linear problem. If more accuracy is desired, the nonlinear problem can be solved numerically using numerical integrators and local nonlinear optimization techniques. This paper evaluates the use of an evolutionary strategy (ES) to solve for the six initial costates. The objective function to be minimized is the norm of the final state of the chaser spacecraft.

$$f = \left\| \vec{x}(t_f) \right\|$$

This is a simple, accurate measure of whether or not the chaser spacecraft achieves rendezvous with the target spacecraft for a given initial state since the desired final state is the zero vector.

The final step of the analysis is to use a hybrid ES incorporating both the chosen ES and local optimization techniques available in Matlab to reliably and quickly solve for the costate.

3.1 LOCAL OPTIMIZATION METHODS

The dogleg method is an example of a trust-region method for a nonlinear minimization problem. If the goal is to minimize some function F, the algorithm locally approximates it with another function p. The function that approximates the local neighborhood, known as the trust region, is tested by taking some trial step q. Therefore, the local problem can be written as follows

$$\min_g \left(p(q) \;\; q \in T \right) \quad (3.1.1)$$

where T defines the subspace that makes up the trust region. If $F(x+q) < F(x)$, the current point is updated to $x+g$. If not, the current point is retained and the size of the trust region is decreased for the next iteration.

The *fsolve* algorithm used for this study uses a quadratic approximation of the local neighborhood.

$$p(q) = \left\{ \frac{1}{2} q^T H q + q^T g \,|\, \left\| Dq \le \Delta \right\| \right\} \quad (3.1.2)$$

Here H is the Hessian matrix, g is the gradient, D is a diagonal scaling matrix, and Δ is some positive

constant. The algorithm used in *fsolve* uses a method that reduces this problem into a 2-dimensional subspace.

Both the Gauss-Newton and Levenberg-Marquardt methods use least squares to solve for search directions. In the Gauss-Newton method, the search direction d_k is obtained by solving

$$\min_{x\in\Re^n}\left\|J(x_k)d_k - F(x_k)\right\|^2 \qquad (3.1.3)$$

where J is the Jacobian. The Levenberg-Marquardt method uses a variation of Gauss-Newton method that increases the robustness of the algorithm. The search direction is determined by solving a linear set of equations

$$\left(J(x_k)^T J(x_k) + \lambda_k I\right)d_k = -J(x_k)^T F(x_k) \qquad (3.1.4)$$

where λ is a controlled parameter. (See Matlab help files for further information and references.)

Fsolve will not optimize a scalar value, so the desired final state minus the actual final state achieved was used as the objective function within fsolve.

3.2 COVARIANCE MATRIX ADAPTATION EVOLUTIONARY STRATEGY (CMA-ES)

The CMA-ES is a robust algorithm for solving non-linear optimization problems when traditional methods fail due to badly scaled, highly non-separable objective functions. It requires relatively small population sizes and utilizes step size control to prevent preconvergence, although it does not guarantee the search will not result in a local optimum (Hanson, 2001).

The CMA-ES is based upon updating a covariance matrix at each generation. The covariance matrix, *C*, describes the correlations between the *n* state variables. Geometrically, it can be defined as a hyper-ellipsoid whose surface defines an equal density of the population. The eigenvalues of *C* are the squared lengths of the principle axes of the hyper-ellipsoid, and the eigenvectors of *C* correspond to the principle axes. The purpose of adapting the covariance matrix is to approximate the inverse Hessian matrix, thus rotating and rescaling the hyper-ellipsoid so that the search distribution fits the contour lines of the objective function.

A population of λ new search points is generated by sampling a multivariate normal distribution given an overall mean, standard deviation, and the covariance matrix defining the correlations between variables

$$x_k^{(g+1)} \sim N\left(m^{(g)}, \left(\sigma^{(g)}\right)^2, C^{(g)}\right) \qquad (3.2.1)$$

for $k = 1,\ldots,\lambda$. The new population is denoted $x_k^{(g+1)}$, $m^{(g)}$ is the mean of the current population, $\sigma^{(g)}$ is the overall standard deviation of the population, and $C^{(g)}$ is the covariance matrix. The mean of the next generation is a weighted average of the μ selected points from the randomly generated sample

$$m^{(g+1)} = \sum_{i=1}^{\mu} w_i x_{i:\lambda}^{(g+1)} \qquad (3.2.2)$$

where

$\sum_{i=1}^{\mu} w_i = 1$, $w_1 > w_2 > \ldots > w_\lambda$, $w_i > 0$, and

$x_{1:\lambda}^{(g+1)}$ indicates the *i*-th best individual out of $x^{(g+1)}$.

Thus selection is implemented by choosing $\mu < \lambda$ and applying the weighting factors w_i so that the best solution of the current generation is weighted most heavily when generating the next generation, the second-best solution next heavily, and so on. Recombination is implemented through using $\mu > 1$ 'parents' to produce the next generation according to equation (3.2.1).

The covariance matrix, $C^{(g)}$, is updated by combining rank-μ updating and rank-one updating, the latter of which uses an evolution path to exploit correlations between successive steps. In the CMA-ES, a step is the normalized distance between the best individual in the next population and the mean of the current population. With each successive step, the covariance matrix evolves, thus the hyper-ellipsoid defined by the matrix rotates and the lengths of the principle axes change. Rank-μ updating efficiently uses the information in the current population, while the rank-one updating uses the information of correlations between steps to update the covariance matrix. Rank-μ updating is important for large populations, while rank-one updating is especially pertinent in small populations.

CMA-ES also applies step-size control through $\sigma^{(g)}$. Step-size refers to the distance in objective space the strategy moves between successive generations. Step size control is beneficial since the overall step length cannot be well approximated by the formula for updating the covariance matrix, and because the largest reliable learning rate for the covariance matrix update is too slow to achieve competitive change rates for the overall step length. Again, an evolution path, which is the sum of successive steps, is utilized. If selection biases the evolution path to be longer than expected, then the standard deviation is increased; if the path is biased by selection to be shorter than expected, σ is decreased. The expected length of the evolution path is the expected value of the multivariate normal distribution *N(0,I)*.

To use the CMA-ES, several parameters affecting the update of the mean, overall standard deviation, and covariance matrix must be set. The default strategy parameters suggested by the authors of CMA-ES provide robust performance and work well for most objective functions.

For a detailed description of the algorithm and explicit derivation of the update equations, refer to Hanson (2001, 2005).

3.3 HYBRIDIZATION

The hybrid approach to solving this problem combines the Covariance Matrix Adaptation evolutionary strategy for global search with the traditional numerical methods integrated into the *fsolve* function in Matlab as a local search tool. This method builds on the strengths of each algorithm to robustly converge to an acceptable solution for the unconstrained two-point boundary value problem corresponding to spacecraft rendezvous.

As has been previously discussed, the CMA-ES algorithm is capable of searching a large parameter space without extreme effects on its ability to find regions of interest for multimodal topographies. Although the ability of the evolutionary strategy to treat all of the parameter space as a potential solution allows for a robust algorithm, this aspect of its design can also lead to long computation times in order to obtain high levels of precision in the solution. To circumvent this negative aspect of CMA-ES, a local search algorithm is used in conjunction with the evolutionary strategy to maximize the search space and minimize the time required for convergence to the specified tolerances.

One of the most convenient local search tools available is the *fsolve* function built into the Matlab optimization toolbox. As previously discussed, this function is capable of using several different algorithms to minimize a problem. Although *fsolve* is much faster computationally than CMA-ES, its algorithms are highly dependent on the initial guess passed to them for convergence. It is predicted that allowing the CMA-ES algorithm to pre-condition this initial guess to a reasonable degree of precision will increase the convergence rate for *fsolve*.

The pseudocode for the hybrid algorithm is as follows:

```
Compute the solution to the linearized problem;
Initialize CMA-ES with the linearized solution;
while generations < max generations
  Run CMA-ES for one generation;
  if Error of Best Member (CMA-ES) < Loop-
    Break Tolerance (LBT)
    run fsolve with best solution from CMA-ES;
    if fsolve converges
      save solution;
      break loop;
    else
      LBT = gap factor*LBT
    end;
  end;
end;
```

The hybrid algorithm performed robustly when either the CMA-ES or *fsolve* algorithms failed individually. It was decided not to inject the best value determined by *fsolve* back into CMA-ES when *fsolve* failed to converge. Injection of the best value obtained by the local search could skew the data in favor of this value and reduce the benefit of the evolutionary strategy – mainly a large search space.

Throughout experimentation it was not possible to achieve 100% reliability for 100 random seeds using the hybrid algorithm. In an attempt to increase reliability, an additional loop of code was added to the end of the original hybrid algorithm. This loop attempts to solve the divergent seeds by relaxing the transfer time parameter. Shorter transfer times are more accurately predicted by the linearized equations and are more readily solved than problems with longer transfer times. By keeping the initial state variable constant and reducing the transfer time, a solution can be found for the more easily solved problem. The transfer time can be incremented to be closer to the desired length of time, and the previous transfer time solution can then be used to initialize the search space for the new problem. This approach improves the performance of the hybrid algorithm because for small changes in transfer time there are only small changes in the trajectory. The pseudocode for this new loop is as follows:

```
dT = transfer time/number of loops;
transfer time = loop number*dT;
for i = 1:number of loops
    Run Hybrid Algorithm;
    if Hybrid converges
      save solution;
      transfer time = transfer time + dT;
    else
      break;
    end;
end;
```

The final loop corresponds to the total desired transfer time, but the initial search space is now much different than it was for the standard hybrid loop. Implementing this method increased the reliability during random seed analysis at the cost of longer iteration time. Using this technique for most seeds is unnecessary, but the authors feel that the increase in reliability far outweighs the additional function evaluations required to implement this extra code.

4 UNCONSTRAINED RENDEZVOUS PROBLEM RESULTS

4.1 *FSOLVE*

It is of considerable advantage to use a traditional solver whenever possible to avoid unnecessary computation. Therefore, some tests were performed in order to gauge the efficacy of the solvers used by the

fsolve function in Matlab. A random seed test for 100 seeds was performed for transfer times of $\pi/2$ time units (TU) and initial state vectors drawn from a uniform distribution from -.001 to .001. The final state vector was chosen to correspond to rendezvous with the chief satellite. The initial hypercube used in the search was based on the magnitude of the costate found using the linear equations of motion. The tolerance was set to 10^{-10}.

Using the trust-region dogleg method for the same initial and final conditions and similar stopping criteria yielded 100% reliability with a mean of 21 function evaluations. Here, an initial guess for the costate vector was taken from the linear approximation (Eqn. 2.1.5). Similarly, the Gauss-Newton and Levenberg-Marquardt based solvers also gave 100% reliability with an average of 28 function evaluations for each algorithm.

For short transfer times it appears that a traditional solver will handle the problem efficiently and reliably. To gauge when the nonlinear effects cause the solver to fail, a range of transfer times was selected to study reliability, shown in Tables 4.1.1, 4.1.2, and 4.1.3. Here a failure means that either the maximum number of function evaluations was reached (in this case 1,000 function evaluations) or the code converged to local minimum. The choice of 1,000 function evaluations is reasonable since, in cases where the traditional methods do converge, they are much less than this number.

Table 4.1.1: Mean Function Evaluations and Reliability for Dogleg method

Transfer time (TU)	Mean func. Evals. (when successful)	% Reliability
$\pi/2$	21	100
π	28	100
$3\pi/2$	70	100
2π	NA	0

Table 4.1.2: Mean Function Evaluations and Reliability for Gauss-Newton method

Transfer Time (TU)	Mean func. Evals. (when successful)	% Reliability
$\pi/2$	28	100
π	39	100
$3\pi/2$	151	100
2π	646	5

Table 4.1.3: Mean Function Evaluations and Reliability for Levenberg-Marquardt method

Transfer Time (TU)	Mean func. Evals. (when successful)	% Reliability
$\pi/2$	28	100
π	40	100
$3\pi/2$	138	100
2π	NA	0

All three methods consistently failed to converge for transfer times of one orbital period, 2π TU. It should be noted that in most cases the dogleg method did come close, but due to internal parameters the algorithm stopped before meeting the convergence criteria. Most often the algorithm converged to a local minimum, which provides evidence of the multimodality of the problem.

4.2 CMA-ES

During random seed analysis, the CMA-ES algorithm performed with 90% reliability using a transfer time of one orbital period, a tolerance of 1e-10, and a maximum allowance of 14,400 function evaluations. Table 4.2.1 summarizes the statistical data from a random seed analysis using only the CMA-ES algorithm to solve the problem.

Table 4.2.1: Function Evaluations for 100 Pseudorandom Seeds using the CMA-ES Algorithm

	Avg. Function Evaluations	Standard Deviation
CMA-ES	8428.4	2712.4

It was found during experimentation that the integrator could become trapped in an infinite loop; the ODE87 Matlab function was modified to prevent this problem by exiting the integrator after a specified number of iterations. The CMA-ES code was able to handle these 'bad' data points by assigning a large fitness value to parameter strings that showed this instability in the integrator. The ability of the algorithm to discount these points allowed the code to remain stable and reliable over a large number of poorly conditioned parameter strings. This robust quality is extremely important because it allowed the authors to continue to use their integrator without further modifications.

4.3 HYBRID ALGORITHM

To improve the reliability of the ES beyond 90%, the hybrid approach was designed. Implementing the hybrid algorithm for 100 random seeds provided

promising results for robustness and quality of solutions. Table 4.3.1 shows the statistical information from random seed analysis using the hybrid algorithm.

Table 4.3.1: Function Evaluations for 100 Pseudorandom Seeds using the Hybrid Algorithm

	Avg. Function Evaluations	Standard Deviation
CMA-ES	7530.2	2922.7
Fsolve	1993.3	434.75
Total	9379.5	3034.2

The hybrid algorithm converged with 94% reliability using a transfer time of one period, a tolerance of 1e-10, a maximum allowance of 14,400 function evaluations for CMA-ES, and a maximum allowance of 4,000 function evaluations[1] for *fsolve*. The precision required to remain within the CMA-ES loop (before transferring to the local solver) is referred to as the loop-break tolerance (LBT), and was set intentionally low at 1e-3 for this random seed analysis. Using a low value for the loop break tolerance provided high reliability, but also increased the total number of function evaluations from the hybrid method because it was unlikely that *fsolve* would be able to converge at these low-precision initial guesses.

As a point of comparison, using *fsolve* by itself resulted in 5% reliability with these same problem settings, even when it was allowed to use 1,000 function evaluations per seed. As discussed in section 3.1, increasing the number of function calls past 1,000 for the *fsolve* algorithm did not increase its reliability.

When comparing the hybrid results to the CMA-ES results shown in Table 3.2.1, it can be seen that the hybrid algorithm has a larger total number of function evaluations, but the CMA-ES section of the hybrid approach has a lower number of function evaluations than using the ES alone. This is an important characteristic because the time required to compute a function call using CMA-ES averaged 14.4 *ms*, while the time required to compute a function call for *fsolve* averaged 8.2 *ms*. This adds credence to the concept that faster solution times may be achieved with the hybrid approach, although they were not achieved during the preliminary random seed analysis.

These preliminary results suggested that decreasing the initial CMA-ES loop-break tolerance value to 1e-5 would reduce the overall number of function evaluations but could also reduce reliability. This conclusion was drawn from noting that the local searcher was called an average of 4.91 times for each seed with a standard deviation of 2.37, showing that an increase in the precision for the loop-break tolerance of 2 to 3 orders of magnitude would solve this problem more effectively.

A parametric study was performed using a variety of LBT values to determine any trends in reliability based on this parameter setting. A complete random seed analysis of 100 seeds was performed for each loop-break tolerance. The best results were found when the loop-break tolerance was set to 1e-5, and the results of this run are shown in Table 4.3.2.

Table 4.3.2: Function Evaluations for 100 Pseudorandom Seeds, LBT = 1e-5

	LBT = 1.00E-05	
	Mean	Standard Deviation
CMA-ES	7896.50	2420.40
fsolve	709.68	181.92
Total	8606.20	2345.40
Subloops	13.80	3.61
Reliability	95/100	

This parametric study was performed with a transfer time of one orbital period, a tolerance of 1e-10, a maximum allowance of 14,400 function evaluations for CMA-ES, and a maximum allowance of 850 function evaluations for *fsolve* over 17 subloops[2]. The small increase in reliability to 95% over the previous hybrid settings is important because reliability was the primary goal of this study.

Adding in the secondary loop described in Section 3.3 showed significant improvement concerning reliability. The results of the entire parametric study and the effect of this additional code can be seen graphically in Figure 4.3.1.

[1] 500 function evaluations per local search, with a maximum of 8 local searches allowed

[2] With a gap-factor of 0.5 and a LBT of 1e-5, the local searcher can be called a maximum of 17 times before CMA-ES must converge past the problem tolerance or fail. Each *fsolve* call is allowed to run for 50 iterations. See Section 3.3 for the pseudocode explanation of this process.

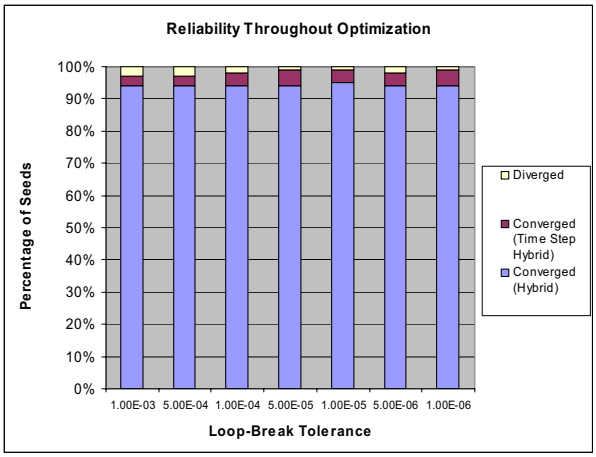

Figure 4.3.1: Algorithm Reliability for Varying LBT

It is plain that the transfer time iteration loop is able to converge for several of the worst seeds used in this study. Throughout the parametric study only seed 72 was unable to achieve convergence. The authors feel that decreasing the size of the transfer time increment will increase the reliability of this code at the cost of increased function evaluations. It is entirely possible to allow the code to adjust this increment automatically rather than hard coding it; this modification will be considered for future work.

5 CONSTRAINED PROBLEM RESULTS

Once the hybrid algorithm demonstrated reliability in solving the unconstrained nonlinear problem, it was applied to the constrained nonlinear problem.

Although none of the seeds converged when thrust was constrained to be less than or equal to 95% of its maximum value for the unconstrained rendezvous problem, initial results indicate that allowing more function calls and longer computation time could result in convergence. This topic will be pursued in future research.

6 CONCLUSIONS

The studies performed for this paper have shown that the reliability of the *fsolve* algorithms is not acceptable for the nonlinear continuous-thrust rendezvous problem for the range of transfer times that are of interest. Similarly, the CMA-ES algorithm demonstrated a robust operation for this problem at all transfer times considered, but the operational overhead of the algorithm led to long times for convergence and an unacceptable failure rate when used alone.

The hybrid approach used the CMA-ES algorithm to reduce the search space and approximate the final solution, which was then passed into the *fsolve* function for final convergence. This method demonstrated the highest reliability of all the tested methods, and showed that it was capable of decreasing the overall convergence time and number of function evaluations for this problem for longer transfer times.

The addition of the incrementally increasing transfer time loop demonstrated its ability to solve initial conditions that the hybrid was unable to solve for this problem. Reducing the transfer time increment in this loop is likely to show extremely high reliability when convergence time is not the paramount constraint.

The success of the hybrid algorithm in solving the unconstrained nonlinear rendezvous problem provides ample motivation for application of the algorithm to the constrained problem. Further work on this topic is in progress. Analysis of the initial data sets should provide insight into the reasons the algorithm failed to converge and future work will focus on successfully applying the hybrid algorithm to the constrained problem.

References

T. Carter and C. Pardis. (1996). Optimal power-limited rendezvous with upper and lower bounds on thrust. *Journal of Guidance, Control, and Dynamics.* **19**(5):1124-1133.

M. Guelman and M. Aleshin. (2001). Optimal bounded low-thrust rendezvous with fixed terminal-approach direction. *Journal of Guidance, Control, and Dynamics*. **24**(2):378-385.

N. Hanson and A. Ostermeier. (2001). Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation*. **9**(2):159-195.

N. Hanson. (2005). The CMA Evolution Strategy: A Tutorial. http://www.bionik.tu-berlin.de/user/niko/cmatutorial.pdf.

J. Igarashi. (2004). Optimal continuous thrust orbit transfer using evolutionary algorithms. Master's Thesis. The Pennsylvania State University. University Park, PA.

C. Lembeck and J. Prussing. (1993). Optimal impulsive intercept with low-thrust rendezvous return. *Journal of Guidance, Control, and Dynamics.* **16**(3):426-433.

H. Schaub and J. Junkins. (2003). *Analytical Mechanics of Space Systems*, 593-628. AIAA. Reston, VA.