

## VARIABLE STATE SIZE OPTIMIZATION PROBLEMS IN ASTRODYNAMICS: N-IMPULSE ORBITAL MANEUVERS

Troy A. Henderson\* and Dario Izzo†

This paper details recent results in variable state size optimization problems in astrodynamics. The application presented in this paper is fueloptimal  $N$ -impulse orbital maneuvers. Previous work required the user to choose the number of impulses,  $N$ , while the current work considers  $N$  as a variable to be optimized, making the state size a variable throughout the optimization process. It is well known that for  $N > 2$ , a numerical optimization method is required for a general solution to the orbit transfer problem. Two algorithmic approaches are presented such that they may be used with a variety of numerical optimization techniques. The first structure runs an outer problem which optimizes the number of impulses, thus determining the length of the state vector, while an inner problem optimizes the orbital maneuver for the given number of impulses. The second structure incorporates the variable  $N$  into the state vector to be optimized, leading to a dynamic state vector length. This structure causes some properties of any given numerical optimization method to necessarily be modified. For example, the evolutionary operators must be able to deal with states of different lengths. Examples are provided that show new scenarios under each structure and show agreement with examples from the literature.

### INTRODUCTION

Significant attention has been given to solving the minimum  $\Delta v$  orbit transfer problem for  $N$ -impulses.<sup>1-4</sup> When  $N > 2$ , an analytical solution is difficult (if not impossible) to find, except in very special cases. Evolutionary algorithms, such as the Genetic Algorithm<sup>5</sup> (GA) Particle Swarm Optimization algorithm<sup>6</sup> (PSO), are well suited to numerically solve the minimum  $\Delta v$ ,  $N$ -impulse orbit transfer problem. The PSO, among other evolutionary algorithms, has been thoroughly studied for optimizing orbit transfers.<sup>7,8</sup> However, no known publications have allowed the number of impulses,  $N$ , to be a variable to be optimized. A publication to appear attempts to re-write the GA for multiple-gravity assist trajectory.<sup>1</sup>

The contribution of this article is twofold. First, the optimization problem is constructed in a way that nests one optimization problem within another, which we call *problem encoding*. Essentially, it provides an inner and outer problem which are dependent on each other.

\*Assistant Professor, Aerospace and Ocean Engineering, Virginia Tech, 215 Randolph Hall, Blacksburg, VA 24061, henderson@vt.edu.

†Advanced Concepts Team, European Space Technology and Research Center, Keplerlaan, 2201 AZ, Noordwijk, The Netherlands, dario.izzo@esa.int.

Second, the widely used PSO will be re-written in a fashion that allows the number of variables to be a parameter to be optimized on, which we will call *variable length encoding*. In other words, the PSO state vector becomes dynamic in size according to the choice of  $N$ . This paper will show details of both methods and compare results for the application of the  $N$ -impulse orbit transfer problem.

## ORBIT TRANSFER PROBLEM

A major limitation of (published) previous work on the  $N$ -impulse orbit transfer problem has been that for  $N > 3$  the orbit transfer problem became computationally intractable.<sup>3</sup> In addition, resolution of the bit-encoded genetic algorithm (GA) caused noticeable error or convergence tolerances were necessarily large to make the problem solveable.<sup>2</sup> More importantly, only up to a three impulse transfers were given in the two aforementioned references. The infeasible computation problem was due to the random selection of  $\Delta v$  values inside of the Genetic Algorithm. Analytical bounds were derived in previous work that solved most of these issues.<sup>4</sup> These bounds will be used in this paper to ensure that the problem has a feasible solution.

Bounds were derived on the choice of generic impulsive velocity such that the new trajectory (i.e., after the  $\Delta v$  is applied) does not intersect a sphere simulating the Earth plus an altitude constraint input by the user. The  $\Delta v$  was decomposed into the following three (generally non-orthogonal) components:

$$\Delta \mathbf{v} = \Delta v_v \hat{\mathbf{v}} + \Delta v_r \hat{\mathbf{r}} + \Delta v_h \hat{\mathbf{h}} \quad (1)$$

Note that  $\Delta v_{v_{\max}}$  and  $\Delta v_h$  cannot be bounded.  $\Delta v_{v_{\min}}$  is bounded by Eqn. (2).

$$\Delta v_{v_{\min}} = -v + v \sqrt{\frac{2\mu r_p (r - r_p)}{r(h^2 - r_p^2 v^2)}} \quad (2)$$

The value of  $\Delta v_r$  can be bounded by

$$\Delta v_r = -\sqrt{\frac{\mu}{p}} e \sin(\nu) \pm \sqrt{\left(\frac{1}{r} - \frac{1}{r_p}\right) \left(2\mu - h^2 \left(\frac{1}{r} + \frac{1}{r_p}\right)\right)} \quad (3)$$

It should be noted that the “+” sign corresponds with the maximum allowed  $\Delta v_r$  value while the “-” sign corresponds with the minimum allowed  $\Delta v_r$  value that guarantee that the resulting orbit will not have a perigee less than the prescribed value  $r_p$ .

These bounds limit the infeasible trajectories that intersect the Earth, and therefore make the problem much more tractable. Thus, the bounds are used when solving the  $N$ -impulse problem for this paper.

The initial and final orbit must be given. The time and change in velocity vector are parameters to be solved for. Given a known initial orbit and both time and new velocity (just after the  $\Delta v$  is applied), the new orbit may be determined. Thus, the specified orbit transfer is achieved.

## OPTIMIZATION PROBLEM DEFINITION

The variable state-size optimization problem seeks the optimal data set  $\mathcal{Z} = \{z_1, z_2, \dots, z_N\}$  to minimize a cost function  $J : \mathfrak{R}^N \rightarrow \mathfrak{R}$ . For the application of  $N$ -impulse orbit maneuvers, the set  $\mathcal{Z}$  is the time or impulse and the three impulse vector components.

As an attempt at the variable state length GA has been made,<sup>1</sup> the PSO algorithm will be the focus of this paper. The PSO algorithm pseudocode is shown in Algorithm 1. Let  $\mathbf{x} \in \mathfrak{R}^N$  and  $\mathbf{v} \in \mathfrak{R}^N$ . Define  $\hat{\mathbf{x}}_i$  as the best position for the  $i^{\text{th}}$  particle and  $\hat{\mathbf{g}}$  be the global best position (with *best position* defined as the position that gives the minimum cost function value). Finally, let  $m$  be the number of particles in the swarm. In the algorithm,

---

### Algorithm 1 Particle Swarm Optimization Algorithm Pseudocode

---

```

1: Initialize  $\mathbf{x}_i$  and  $\mathbf{v}_i$  for  $i = 1, \dots, m$ 
2:  $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$  and  $\hat{\mathbf{g}} = \min_{\mathbf{x}_i} J(\mathbf{x}_i)$  for  $i = 1, \dots, m$ 
3: while Not Terminate do
4:   for  $i = 1$  to  $m$  do
5:     Randomly generate  $r_1, r_2 \in U[0, 1]$  for  $j = 1, \dots, m$ 
6:     Update particle velocities:  $\mathbf{v}_i \leftarrow \omega \mathbf{v}_i + c_1 r_1 (\hat{\mathbf{x}}_i - \mathbf{x}_i) + c_2 r_2 (\hat{\mathbf{g}} - \mathbf{x}_i)$ 
7:     Update particle positions:  $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$ 
8:     Update particle best positions:
9:     if  $J(\mathbf{x}_i) < J(\hat{\mathbf{x}}_i)$  then
10:        $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$ 
11:     end if
12:     Update global best position:
13:     if  $J(\mathbf{x}_i) < J(\hat{\mathbf{g}})$  then
14:        $\hat{\mathbf{g}} \leftarrow \mathbf{x}_i$ 
15:     end if
16:   end for
17: end while

```

---

the value  $\omega$  is the *inertial constant*. Generally, this value is slightly less than 1, but it can be a random value for each particle. The random values  $c_1$  and  $c_2$  are the *cognitive* and *social* components, respectively. These values determine the influence on the movement of particles based on the particle's personal best position,  $c_1$ , and the global best position,  $c_2$ . Typical values of  $c_1$  and  $c_2$  are very near 2.

### Problem Encoding

The first solution method is a nested optimization problem approach which will be called Problem Encoding (PE). For the case of the  $N$ -impulse orbit transfer problem, the outer problem selects the integer,  $N$ , defining the state vector length to be optimized in the inner problem. In this application, the inner problem then optimizes the  $\Delta v$  based on a given number of impulses. This solution method is simple to implement and allows extremely simple modularity and combinatorial application of algorithms. For example, a GA could

be used to optimize on  $N$  and a PSO could be used to optimize the  $\Delta v$  of the orbit transfer. Examples of this method are given in a following section.

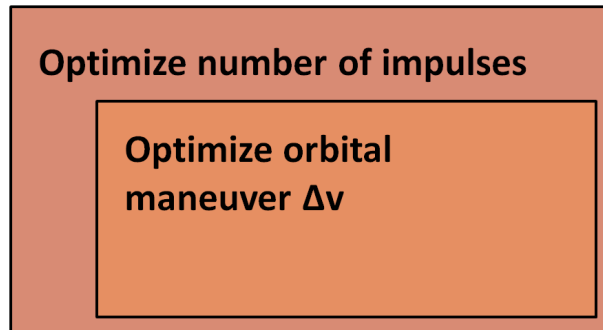


Figure 1. Problem Encoding

### Variable Length Encoding

A search of the current literature will show many algorithms which allow for a variable number of *population* during the optimization process. The most current research in evolutionary programming is variable length encoding, or developing algorithms that are able to handle variable state vector lengths during the optimization process. In pursuing this research, algorithms are modified out of necessity. For example, the GA requires the genetic operators (crossover and mutation) to be able to deal with chromosomes of different lengths.

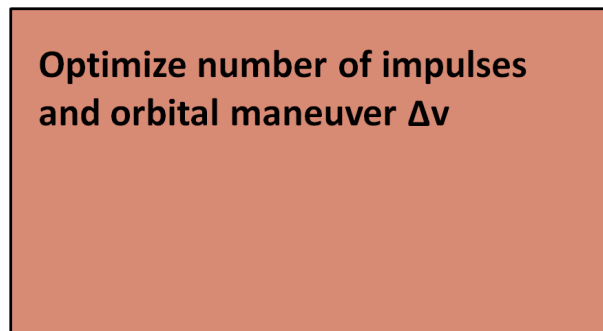


Figure 2. Variable Length Encoding

### IMPLEMENTATION

MATLAB was used as the tool to test the algorithms. The orbit maneuver was completely described and implemented previously.<sup>4</sup> The orbit maneuver formulation requires the user to input initial and final orbit parameters (not true anomaly), a maximum time of flight, minimum allowable altitude, and maximum (scalar) thrust per maneuver. The problem

encoding algorithm was easily implemented by wrapping an outer loop around the existing method. Results will be shown in the next section.

The variable length encoding algorithm was limited to the PSO in MATLAB, and basically became a problem of memory management. For the examples presented here, the variable length encoding was achieved by maintaining the maximum set of variables in memory and only accessing the number of states needed per iteration. In other words, the user must define the lower and upper bounds on the number of allowed states (making it "unconstrained" by choosing zero and infinity, respectively). Thus, if a user chose a maximum of 5 state variables, over 10 iterations, with a population of 25, then a total of 1,250 storage locations would be required, or 125 storage locations per iteration. When less than the maximum number of state variables were used in a given iteration, the first  $k$  state variables were used and the last  $N - k$  state variables were copied over from the previous iteration (and all were initialized to a given value of zero in this paper).

## EXAMPLES

Both solution methods (problem encoding and variable length encoding) were applied to a set of orbit transfer problems. Namely, the Hohmann transfer problem was used as the first example as it has a known solution.

For the results shown here, for the problem encoding, both the GA and PSO were used on the outer loop and the PSO was used on the inner loop. For the variable length encoding method, only the PSO was used as described above.

A statistical analysis was performed as the algorithms being used are heuristic and do not guarantee that a minimum value will be found. For each problem, 100,000 cost function calls were allowed per trial and 100 independent trials were run. As an additional metric, the `cputime` command in MATLAB was used to time each trial.

### Co-Planar, Long Duration Transfer

As a first example, a co-planar transfer from 7,000 km to 42,164 km was investigated with a maximum transfer time of 3 days. The value of  $N$  was arbitrarily bounded between 2 and 5.

Given the sufficiently long transfer time, the PE found the required  $\Delta v = 3.7707$  km/s using 2 impulses (94 trials) when either the GA or PSO was used in the outer loop. (Note that the other 6 trials gave a result of 3 impulses, but the second impulse was at least two orders of magnitude smaller than the others, suggesting this was effectively zero.) This is in fact the same  $\Delta v$  required by the Hohmann transfer, and the transfer times matched to sub-second accuracy (5 hours, 19 minutes, 38.16 seconds). The mean CPU time measured was 148 seconds.

In addition, the variable length encoding method found the same  $\Delta v$  to less than mm/s level in each of the 100 trials and the transfer time matched to sub-second accuracy in each trial. This method gave a result of two impulses 99 trials and the other 1 trial gave a result of 3 impulses, but again, the second impulse in this scenario was on the order of

a few mm/s. The interesting thing about these methods is that the variable length encode took about one-half of the computation time on average over the 100 trials, averaging 72 seconds.

### **Co-Planar, Short Duration Transfer**

The second example was again a co-planar transfer from 7,000 km to 42,164 km. However, the maximum transfer time was changed to 2 hours, making a Hohmann transfer impossible. The value of  $N$  was again arbitrarily bounded between 2 and 5.

The problem encoding method found the required  $\Delta v = 5.3028$  km/s using 4 impulses. It is interesting however, that impulses 2 and 3 were very small (less than 0.4 km/s, or less than one-fifth of the first and fourth impulses). The standard deviation on the  $\Delta v$  result was 0.2001 km/s. The statistical analysis of this method using both the GA and PSO in the outer loop resulted in 93 trials using 4 impulses, two trials using 3 impulses, and five trials using 5 impulses. The mean time of flight was 1.9864 hours, and the mean CPU time was 252 seconds.

The variable length encoding method also preferred 4 impulses for 91 of the 100 trials (using 3 impulses on all other 9 trials). However, the  $\Delta v$  was slightly improved in both average and standard deviation to a mean of 5.2861 km/s and standard deviation of 0.0314 km/s. The most impressive feat in this example was that the computation time required was on average one-fourth of that taken by the problem encoding method, averaging just 68 seconds. It is assumed that the problem encoding method was basically performing a search through all of the allowed number of impulses, thus taking significantly longer.

### **Plane Change**

The final example involved both a plane and semimajor axis change. The plane was changed from 28.5 deg to equatorial (0 deg) and the semimajor axis from 6,700 km to 26,500 km. Both orbits were considered circular. The maximum transfer time was 2 days and the value of  $N$  was arbitrarily bounded between 3 and 20.

The problem encoding method found the required  $\Delta v = 4.1877$  km/s using 3 impulses (for 89 of the 100 trials). The standard deviation was 0.3579 km/s. As expected, the first impulse took a couple of degrees of inclination, but the majority of plane change was performed after the altitude raising. In addition, the final impulse was generally around 0.2 km/s, indicating a small change to the orbit. For this scenario, 8 trials used 4 impulses and 3 trials used 5 impulses. The mean time of flight was 21.2941 hours, and the mean CPU time was 489 seconds.

The variable length encoding method also preferred 3 impulses for 94 of the 100 trials (using 4 impulses on all other 6 trials). The mean  $\Delta v$  was 4.0913 km/s and standard deviation of 0.2037 km/s. Again, the final impulse was on the order of 0.2 km/s, suggesting minor changes to the orbit. The time of flight found exactly matched the problem encoding solution at 21.2941 hours, but the CPU time was drastically improved to 131 seconds on average.

## CONCLUSIONS AND FUTURE WORK

A comparison of solution methods was made based on the simple examples. The utility of the problem encoding method lies in the fact that it can easily be made into a hybrid of two evolutionary algorithms. However, it consistently showed to take three to four times longer in computation time using MATLAB. A variable state size PSO was easily implemented and shown to work faster than the problem encoding method.

Future work will include using a numerical optimizer that requires a starting point, such as simulated annealing, using a Lambert solution as the starting point. In addition, modifications will be made to allow for both variable state size and a variable number of particles within the same optimizer.

## REFERENCES

- [1] A. Gad and O. Abdelkhalik, "Hidden Genes Genetic Algorithm for Multi-Gravity-Assist Trajectories Optimization," *Journal of Spacecraft and Rockets (to appear)*, 2011.
- [2] Y. H. Kim and D. B. Spencer, "Optimal Spacecraft Rendezvous Using Genetic Algorithms," *Journal of Spacecraft and Rockets*, Vol. 39, November–December 2002, pp. 859–865.
- [3] O. O. Abdelkhalik and D. Mortari, "*N*-Impulse Orbit Transfer Using Genetic Algorithms," *Journal of Spacecraft and Rockets*, Vol. 44, March–April 2007, pp. 456–459.
- [4] T. A. Henderson, D. Mortari, and M. E. Avendaño, "Admissible *N*-impulse Orbit Transfer and Rendezvous Solved Using a Learning Optimization Algorithm," *AAS 10–252 of the 20th AAS/AIAA Space Flight Mechanics Meeting Conference*, San Diego, CA, February 8–12 2010.
- [5] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
- [6] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942–1948.
- [7] P. J. Gage, R. D. Braun, and I. M. Kroo, "Interplanetary Trajectory Optimization Using a Genetic Algorithm," *Journal of the Astronautical Sciences*, Vol. 43, No. 1, 1995, pp. 59–75.
- [8] G. A. Rauwolf and V. L. Coverstone-Carroll, "Near-Optimal Low-Thrust Orbit Transfers Generated by a Genetic Algorithm," *Journal of Spacecraft and Rockets*, Vol. 33, November–December 1996, pp. 859–862.