(Preprint) AAS 15-582

GLOBAL OPTIMIZATION OF INTERPLANETARY TRAJECTORIES IN THE PRESENCE OF REALISTIC MISSION CONSTRAINTS

David Hinckley Jr.,* Jacob Englander,† and Darren Hitt‡

Interplanetary missions are often subject to difficult constraints, like solar phase angle upon arrival at the destination, velocity at arrival, and altitudes for flybys. Preliminary design of such missions is often conducted by solving the unconstrained problem and then filtering away solutions which do not naturally satisfy the constraints. However this can bias the search into non-advantageous regions of the solution space, so it can be better to conduct preliminary design with the full set of constraints imposed. In this work two stochastic global search methods are developed which are well suited to the constrained global interplanetary trajectory optimization problem.

INTRODUCTION

Interplanetary missions are often subject to difficult constraints, like solar phase angle upon arrival at the destination, velocity at arrival, and altitudes for flybys. Preliminary design of such missions is often conducted by solving the unconstrained problem and then filtering away solutions which do not naturally satisfy the constraints. However this can bias the search into non-advantageous regions of the solution space, so it can be better to conduct preliminary design with the full set of constraints imposed. In this work a stochastic global search method is developed which is well suited to the constrained global interplanetary trajectory optimization problem.

In the interplanetary frame, the patched conic approach is used to simplify the planetary arcs. It is often useful in preliminary design to model an interplanetary trajectory using the Multiple Gravity Assist with one Deep-Space Maneuver (MGA-1DSM)[1,2] problem transcription. This allows for trajectories in which the spacecraft makes a single maneuver between each pair of planetary flybys. In 2007 Vasile, Minisci, and Locatelli[3] used this problem transcription with a robust variant of Differential-Evolution (DE) called Inflationary DE Algorithm. In 2008 Vinkó and Izzo[2] explored solving problems of this type using cooperative combinations of DE, Genetic Algorithm (GA), and Particle Swarm Optimization. They found that combinations of different methods had the potential to surpass component methods on certain problems. This is meaningful to this work as the algorithm presented here draws inspiration from elements of existing evolutionary algorithms. In 2013 Cassioli et al.[4] applied Monotonic Basin Hopping (MBH) and Simulated Annealing to different trajectory design problems. Most recently, Englander[5] solved trajectory optimization problems of the type discussed here using MBH and explicit handling of nonlinear constraints as opposed to a penalty method approach.

*Mechanical Engineering Graduate Student, University of Vermont
†Aerospace Engineer, Navigation and Mission Design Branch, NASA Goddard Space Flight Center
‡Professor of Mechanical Engineering, University of Vermont

This work is focused on chemical propulsion mission design; as such, the impulsive thrust model is used. The algorithm presented in this work is implemented in the Evolutionary Mission Trajectory Generator (EMTG), NASA Goddard's open-source interplanetary trajectory optimization tool.[6,7] EMTG provides all of the necessary modeling for the example problems in this work.

MGA-1DSM

The MGA-1DSM problem transcription was first developed by Vasile and De Pascale.[1] Later, Vinkó and Izzo[2] further developed the model. In this model, interplanetary trajectories are described by, after launch, an alternating sequence of impulsive DSMs and unpowered flybys. Each full three-dimensional DSM, the times of flight for the planetary arcs, parameters pertaining to the flybys, as well as launch and arrival condition parameters are all decision variables. Figure 1 illustrates a MGA-1DSM trajectory where T_i refers to the time of flight for the i^{th} arc and η_i is the fraction of the arc at which the DSM, shown as a red star, occurs.

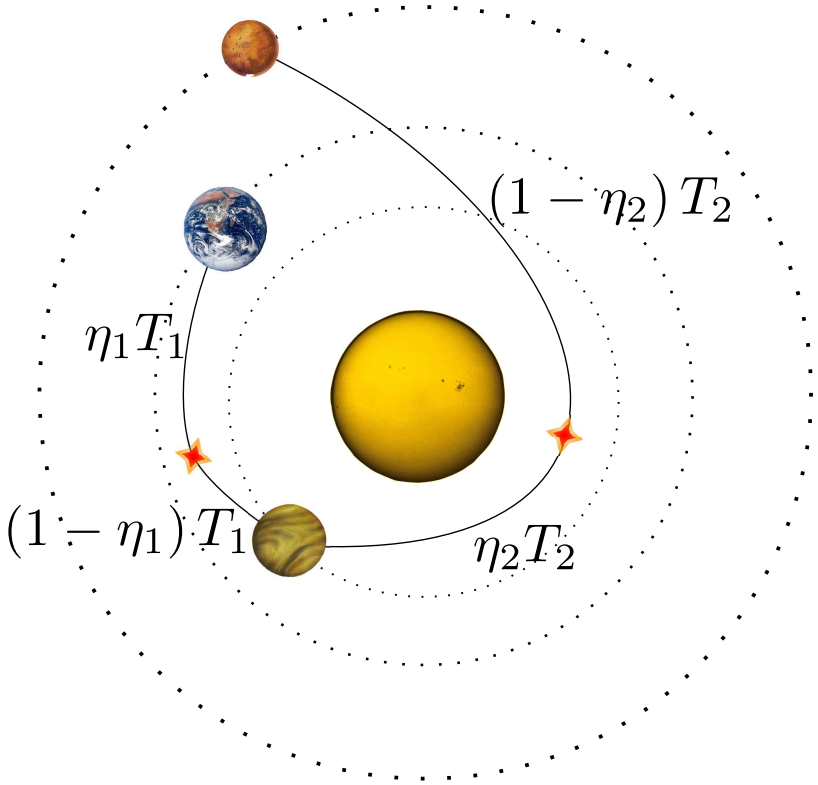

Figure 1. Diagram of a Two-Phase MGA-1DSM Mission

For a more complete explanation of this model see Vinkó and Izzo.[2]

ALGORITHM: PHASE GENETIC SOLVER

The algorithm presented in this work is a Phase-Genetic Solver (PGS). Like all evolutionary algorithms, the PGS requires a population of solutions from which to draw genetic material. However unlike other evolutionary algorithms, PGS evaluates only a single candidate solution per

generation.PGS takes advantage of problem-specific gene groupings common to all MGA-1DSM trajectory optimization problems.

PGS maintains a set of rank-sorted archives, or "bins," to provide the required genetic material. The bins are filled with a random assortment of initial genetic material during an initialization period that occurs only once per solver run. Since storing duplicate solutions lessens the total diversity of genetic material stored, a non-exclusionary "kill" distance is employed. If a solution sorted into the archive is within a user-controlled distance of its archive neighbors the less fit (or closest in case the new solution is sufficiently close to both neighbors) is removed. In the MGA-1DSM transcription, groups of decision variables completely describe the planet-to-planet arcs. The fact that when grouped together these decision variables have meaning beyond their independent value was the inspiration for the implementation of a GA-style recombination where these phase gene-blocks are taken from different existing solutions and combined to form a new potential solution.

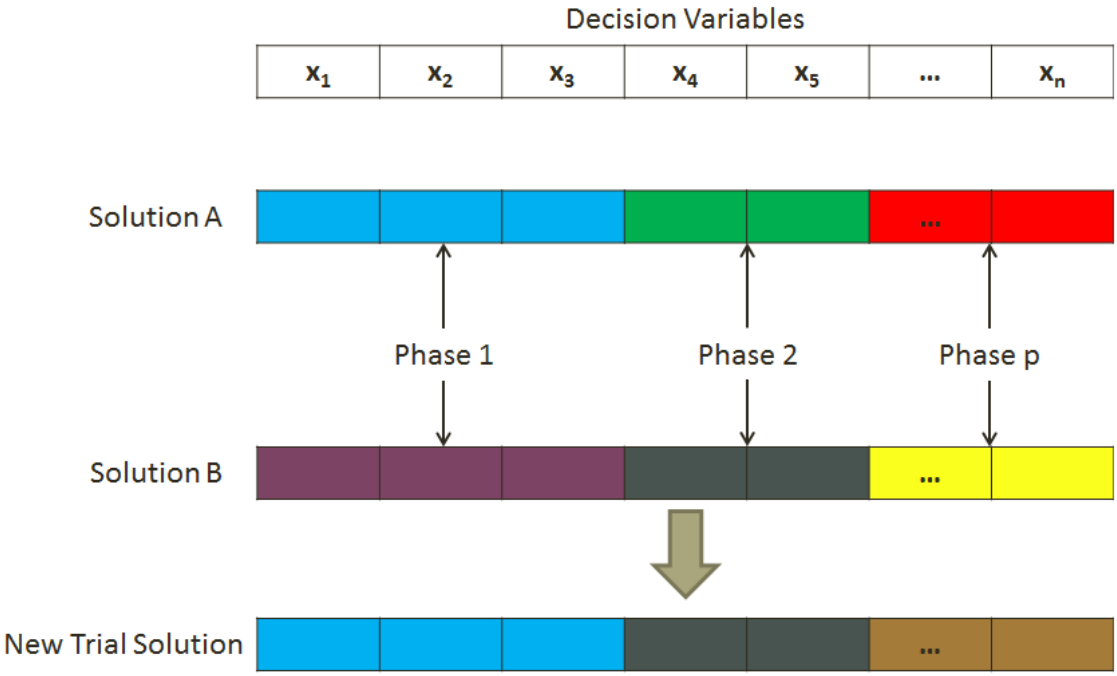

Figure 2. GA-inspired recombination

Figure 2 illustrates the phase-wise gene recombination. Since such a recombination cannot create new values for any decision variable, this is paired with a DE-inspired recombination. This operation uses the weighted decision vector combination part of the primary DE creation scheme as an alternative creation method to the GA-style recombination. The switch between the two is a user-controlled parameter set to a value on [0,1] where a uniform random draw falling below the set value triggers solution creation by the DE scheme. This is illustrated by Equation 1.

$$\mathbf{y} = \mathbf{x_1} + F_w(\mathbf{x_2} - \mathbf{x_3}) \tag{1}$$

where $\mathbf{y}$ is the new candidate solution, F_w is the DE scaling parameter, and $\mathbf{x_i}$ are unique solutions drawn from the bin. The DE recombination excels at exploration while the GA recombination is well suited to exploitation once the archive fills with fit feasible solutions. To further aid exploration two

additional perturbations are added. First, a gene-wise chance of adding a two-tailed Pareto random value was added. Second, a user-controlled likelihood of adding orbital periods to time decision variables[4] was added. These stochastic additions increase the method's ability to search beyond the limitations of the finite genetic information from which both creation schemes draw.

Step-by-step

The algorithm begins with an initialization period during which trial solutions are drawn from a uniform random distribution over the bounded decision space. The number of trials evaluated during this initialization, i.e. the size of the initial gene pool, is user-controlled but should be set to a number greater than the bin size yet not too close to that number so that to ensure that the bin is filled in the case of duplicates being removed through the "kill" procedure. After the initialization period the main loop begins and continues until one of a set of stopping criteria is met. The main loop begins with creating a new solution by either the GA or DE recombination schemes. Then the element-wise Pareto mutation and the orbit period time hop mutation are applied with the user selected probability. The trial is then locally optimized. For the results shown in this paper local optimization was done using a variation on the Nelder-Mead algorithm[9] that addresses the scale issue in the basins of minima in the MGA-1DSM transcription. The Nelder-Mead local optimizer is run multiple times where each successive run shrinks the initial step distance; this decrease is not dependent on the previous run but the starting position is taken from where the previous run terminated. This is done since very small steps are needed toward the end of the local optimization to narrow in on the minimum, yet starting at those step sizes would greatly increase the time taken for this operation. After local optimization, the solution is sorted into the bin if it is sufficiently fit. This continues until user-specified conditions are met. This work used elapsed time as the primary termination condition. The algorithm is shown in pseudo-code form in Algorithm 1.

TEST PROBLEMS

Testing of the algorithm was conducted of four test problems composed of two types. The first type was European Space Agency Global Trajectory Problem (ESA-GTOP) database[10] inspired problems. The second was derived from real mission objectives and constraints. The purpose of the first set of problems was to provide a means of assessing the method on an understood problem. While the versions of the problems used here do not exactly follow the GTOP specifications, the problems are close enough that answers found should be close to the GTOP posted optimum. The second set of problems demonstrates the ability of the algorithm to address real-world problems that are constrained in meaningful ways. These problems are variations on the OSIRIS-REx mission.[11] For all problems the objective function is minimization of total Δv.

Problem 1: Cassini 2

The Cassini 2 problem is an academic problem based on a simplified version of the Cassini mission.[12] The version of Cassini 2 here employed closely mirrors the problem as described in the GTOP database. The planetary sequence is identical: launch from Earth with a fly-by sequence of Venus, Venus, Earth, Jupiter ending with a rendezvous maneuver at Saturn. This arrival condition is as specified by the GTOP database. A rendezvous maneuver consumes more fuel than the real Cassini mission's orbital insertion upon arrival. In this implementation only the total time of flight was bounded as opposed to the bounding of the time-of-flight for planet-to-planet arcs as is the GTOP problem description. The mission's time of flight was given an upper bound of 8 years. This

Algorithm 1 Phase Genetic Solver

for $genSw$ times **do**
  generate random point $\mathbf{x}$
  run NLP solver (here Nelder-Mead) to find point $\mathbf{x}^*$ from $\mathbf{x}$
  if bin is not full **or** $F(\mathbf{x}^*) < F(x_w)$ where x_w the worst solution currently in the bin as determined by a weighted sum of objective value and constraint violation **then**
    $Sort(\mathbf{x}^*)$, Algorithm 2
  end if
  if $\mathbf{x}^*$ is a feasible point **then**
    save $\mathbf{x}^*$ to archive
  end if
end for
while not hit stop criterion **do**
  get random value [0,1] $= varType$
  if $varType < mutCap = 0.5$ by default **then**
    create trial $\mathbf{x}$ through DE-recombination
    $\mathbf{x} = \mathbf{x_1} + 0.85 * (\mathbf{x_2} - \mathbf{x_3})$ for 3 unique $\mathbf{x_i}$
  else
    create trial $\mathbf{x}$ through GA-recombination
    separate the genome into phase-specific blocks such that:
    $\mathbf{y} = [g_1, g_2, ...g_p]$ where g_i are vectors of phase-specific gene blocks and p is the number of phases
    then: $\mathbf{x} = [g_1^{v_n}, g_2^{v_m}, ...g_p^{v_q}]$ where $g_i^{v_w}$ means take gene-block i from population member v_w
  end if
  with probability $mutUse = 2\%$ by default, apply Pareto noise to every element of the trial solution
  for each time of flight variable t_i in $\mathbf{x}'$ **do**
    if $rand\,(0,1) < \rho_{time-hop}$ **then**
      shift t_i forward or backward one synodic period
    end if
  end for
  run NLP solver to find point $\mathbf{x}^*$ from $\mathbf{x}$
  if $F(\mathbf{x}^*) < F(x_w)$ **then**
    $Sort(\mathbf{x}^*)$, Algorithm 2
  end if
  if $\mathbf{x}^*$ is a feasible point **then**
    save $\mathbf{x}^*$ to archive
  end if
end while
return best $\mathbf{x}^*$ in archive

Algorithm 2 $Sort(\mathbf{x})$

```
recursively find the best value beaten
if x beats anyone then
    check to see if neighbors are within kill distance
    if yes then
        kill by proximity then rank, self termination is possible
    end if
    if not self-terminated then
        insert into population
    end if
end if
```

was selected as the total time bounds as it seemed to be a reasonable amount of time relative to the real mission's design. The first allowed launch days was set to April 1^{st} 1997.

The spacecraft in the Cassini 2 problem launches on an Atlas V 551 and performs DSMs with a thruster I_{sp} of 320 s. Note that the actual Cassini spacecraft launched on a Titan IVb, but since a Titan IVb performance model was not available Atlas V 551 was used instead. The problem description is summarized in Table 1.

Table 1. Summary of specifications for Problem 1

Description	Value
Launch date open	4-1-1997
Launch date close	1-1-2000
Flight time upper bound	8 years
Launch vehicle	Atlas V 551
Spacecraft I_{sp}	320
Planet sequence	E-V-V-E-J-S
Arrival condition	rendezvous
Objective function	minimize Δv
Run-time bound	2 hours

Problem 2: MESSENGER Reduced

The second test problem is based on the "MESSENGER Reduced" problem from the GTOP database and uses the same flyby sequence. The mission launches from Earth with a fly-by of Earth and two fly-bys of Venus ending with a rendezvous maneuver at Mercury. The total time of flight was bounded to the sum of the bounded legs as given in the GTOP description, 1600 days. This problem was set with the same launch window open date, maximum initial mass, and I_{sp} as Problem 1. The launch vehicle was changed to an Atlas V 401 which is more similar to the Delta II used for the real MESSENGER mission.[13] This problem's specifications are summarized in Table 2.

Problem 3 and 4: OSIRIS-REx with Sun-target-angle constraints

Problem 3 and 4 are based on the OSIRIS-REx mission. This mission consists of two journeys, first from Earth to rendezvous with the asteroid Bennu by way of an Earth flyby, then returning to

Earth with an intercept condition. To ensure visibility of the asteroid upon arrival a constraint is placed on the optimization that requires approach within some angular range of the Sun-Bennu line. For Problem 3 the angle is $90°$, and for Problem 4 the angle is $45°$. Since the problem is handed in a patched-conic framework the position at arrival is the center of the asteroid. To account for this the Sun-target-angle is computed using the incoming velocity vector as an approximation to the direction of approach. Figure 3 illustrates the angle defined by this constraint. The yellow circle represents the Sun and the grey circle is Bennu. From this construction it is clear that the angle shown represents a measure of visibility at arrival.

Figure 3. Sun-Target-Angle

In addition, the arrival velocity at Earth was constrained to be less than $6.28\frac{km}{s}$. This constraint prevents the sample return capsule from burning up on re-entry. Launch and vehicle settings for this problem were set inline with their actual values. The first allowed launch date was September 1^{st} 2016 with a total flight bounds of 10000 days so as to not enforce assumptions on mission time. This is also sensible as the return to Earth is bounded between January 1^{st} 2019 and November 1^{st} 2023. Since this is the only problem where the craft returns from the target, the new decision variable of the stay time at Bennu is introduced. That stay time was given an upper bound of 1500 days. The problem was set to use an Atlas V 411 with an initial mass of 1955kg and an I_{sp} of 230.7s as per the real OSIRIS-REx mission.

Table 2. Summary of specifications for Problem 2

Description	Value
Launch date open	4-1-1997
Launch date close	1-1-2000
Flight time upper bound	1600 days
Launch vehicle	Atlas V 401
Spacecraft I_{sp}	320
Planet sequence	E-E-V-V-Y
Arrival condition	rendezvous
Objective function	minimize Δv
Run-time bound	2 hours

RESULTS

Table 4 shows the default settings for PGS as applied to this testing. Deviations from this set will be outlined for specific tests. A setting preceded by "NM" denotes a setting that affects only the Nelder-Mead local optimizer. The settings "stagnation limit" and "repeat" refer to the alterations to the base Nelder-Mead algorithm where "stagnation limit" refers to the number of iterations allowed without improvement and "repeat" refers to the amount of restart and shrinks performed. The "narrowing factor" is the amount by which the Nelder-Mead step size is multiplied upon restart to address the problem of scale inherent in the basins of this problem. Kill distance and initial step here are expressed in the scaled decision space where all decision variables assume values between 0 and 1 which directly map to the entire bounded space. This normalization of the search space is done to improve performance, and has been shown effective with other solvers.[2]

Problem 1: Cassini 2

The total Δv was 8.4878 km/s launching on May 7^{th} 1997. Figure 4 shows the best found solution. The tuning that found this solution did not use the "kill-distance" duplicate reduction measure. This tuning is more likely to get locally trapped since the GA recombination procedure will increasingly make duplicates once they appear. In 10 trials, the average for this tuning was poor relative to other tunings but the allowed persistence of solutions that are close in the decision space allows for the DE recombination to improve best fitness through more of a local search. It bears mentioning that the amount by which this tuning's best solution beat tunings with a better average is negligible considering the error inherent in patched conics. For this problem, the best average results were obtained through an increase in the percent likelihood of choosing the DE recombination over the GA recombination. Of the values tested, changing the default 50% trade-off value to a 70% value favoring DE had all of a set of trials tested within 0.04 of the best solution which had a total Δv of 8.517 km/s.

These solutions are reasonably close the the GTOP record holding solution whose total Δv is 8.383 km/s. Comparing the presented solutions directly with the current record is improper as the problems were not run using the same ephemerides and some details regarding the problem construction are different. The solutions being as close as they are simply shows that the new algorithm presented in this work recovered a sufficiently good solution.

Table 5 shows the various tunings that were explored for this problem.

Table 3. Summary of specifications for Problems 3 & 4

Description	Value
Journey	Earth to Bennu
Launch date open	4-1-1997
Body encounter sequence	E-E-Bennu
Arrival condition	rendezvous
Journey	Bennu to Earth
Wait time lower bound	500 days
Wait time upper bound	1500 days
Arrival date bounds	[1-1-2019,11-1-2023]
Body encounter sequence	Bennu-E
Arrival condition	intercept
Arrival velocity upper bound	$6.28\frac{km}{s}$
Global	
Total flight time upper bound	1600 days
Launch vehicle	Atlas V 401
Spacecraft I_{sp}	320
Objective function	minimize Δv
Total run-time bound	2 hours

Table 4. PGS default settings

Description	Value
Bin size	100
Number of initialization generations	500
Enabled proximity kill	true
Kill distance	0.9
Fitness penalty weight	100
DE creation percentage	50
DE scaling parameter	0.85
Pareto mutation percentage	2%
NM max iterations	500
NM stagnation limit	30
NM initial step	0.05
NM repeat	5
NM narrowing factor	0.01

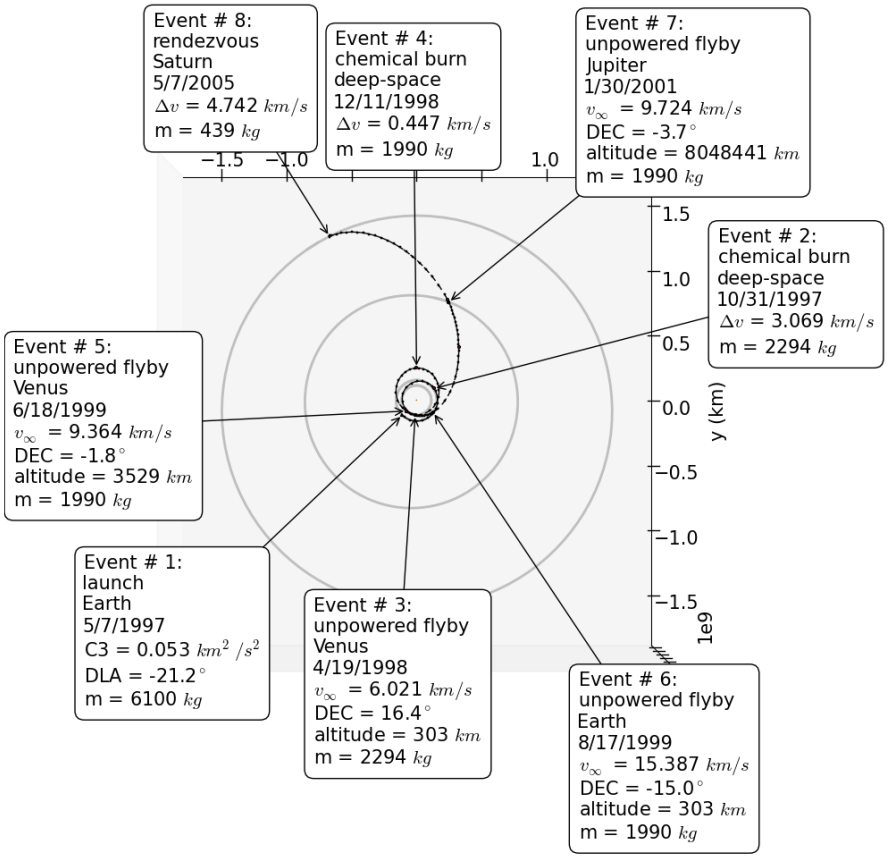

Figure 4. Cassini 2 result trajectory

Test Name	Minimum Fitness	Maximum Fitness	Mean Fitness	Standard Deviation of Fitness
NM max iterations = 1000	8.507702203	13.80813145	10.19337892	2.381004889
NM max iterations = 250 & NM stagnation limit = 30	8.495142971	13.80811605	9.129719809	1.583863494
Disabled proximity kill	8.487794692	15.4574581	9.517611548	2.019381623
NM initial step = 0.1	8.519776442	13.80824377	12.40553514	2.155671733
NM repeat = 3	8.491886681	13.80811288	9.534945467	2.055517186
NM initial step = 0.01	8.529650524	13.80819269	9.742230666	1.56202128
Kill distance = 1.25	8.548351326	13.80855295	9.618825493	2.094259801
NM expansion parameter = 5	8.493211225	13.80810881	9.213243008	1.572387353
Kill distance = 0.45	8.489449107	15.64071313	10.84552859	2.550076981
Kill distance = 1.5	8.548151998	8.6373963	8.588254452	0.028344059
DE creation percentage = 60	8.510255988	13.80812768	10.37414168	2.37686051
NM max iterations = 250	8.502856018	13.80823687	10.28526561	2.330492865
Fitness penalty weight = 50	8.510825618	9.445452757	8.696852204	0.354515305
DE creation percentage = 30	8.943567626	16.04331072	13.08227751	2.115409215
DE creation percentage = 80	8.518332019	8.578794109	8.541124301	0.015297718
NM stagnation limit = 50	8.500648814	13.80810493	9.037603714	1.590176354
Fitness penalty weight = 10	8.50422969	44.25559316	17.57454411	11.51069538
Default	8.501509161	15.64214162	11.20484013	2.95955532
DE creation percentage = 70	8.517078919	8.556319048	8.534003909	0.009798648
Kill distance = 2.0	8.638286021	13.87964676	10.2763394	2.346441835
Pareto mutation percentage = 100	8.501850121	11.16184864	8.972233049	0.816102361

Table 5. Tunings tested for Problem 1

Problem 2: MESSENGER Reduced

For this problem the total Δv was 7.848 km/s launching on April 25^{th} 1998. The delivered mass was 326 kg with a flight time of 4.19 years. Figure 5 shows the trajectory. This solution was found by altering the Nelder-Mead local optimizer settings to increase the maximum number of iterations allowed and increasing the number of iterations without improvement before returning a value. This was not the best solution found in terms of Δv. Solutions to this academically posed problem were found with lower total Δv but they, due to how this problem is posed, perform maneuvers that are not applicable in real mission flight. Figure 6 shows the best solution found to this problem. The total Δv was 7.700 km/s which is considerably better than the previous solution. The launch date was May 12^{th} 1998, the delivered mass was 360 kg, and the total flight time was 4.36 years. The problem with this solution is that it essentially performs most of the rendezvous as part of a deep-space maneuver and then follows Mercury through most of an orbit. This is due to the rendezvous condition which is not a realistic arrival condition. Real missions would either perform an intercept, for direct landing, or an orbital insertion upon arrival. Neither of these conditions allow for this matching of Mercury for most of an orbit to be locally advantageous. In this case, the academic formulation of the MESSENGER-Reduced problem is not an accurate representation of the true preliminary design problem. However it is used here for the sake of comparison to other published work.

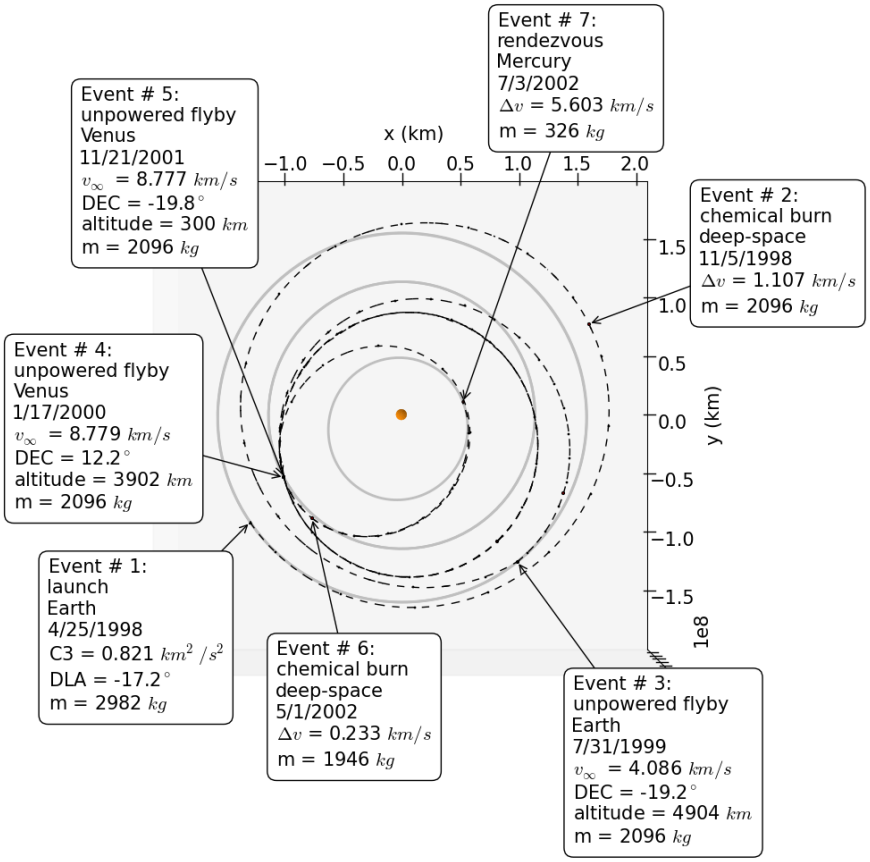

Figure 5. MESSENGER Reduced result trajectory

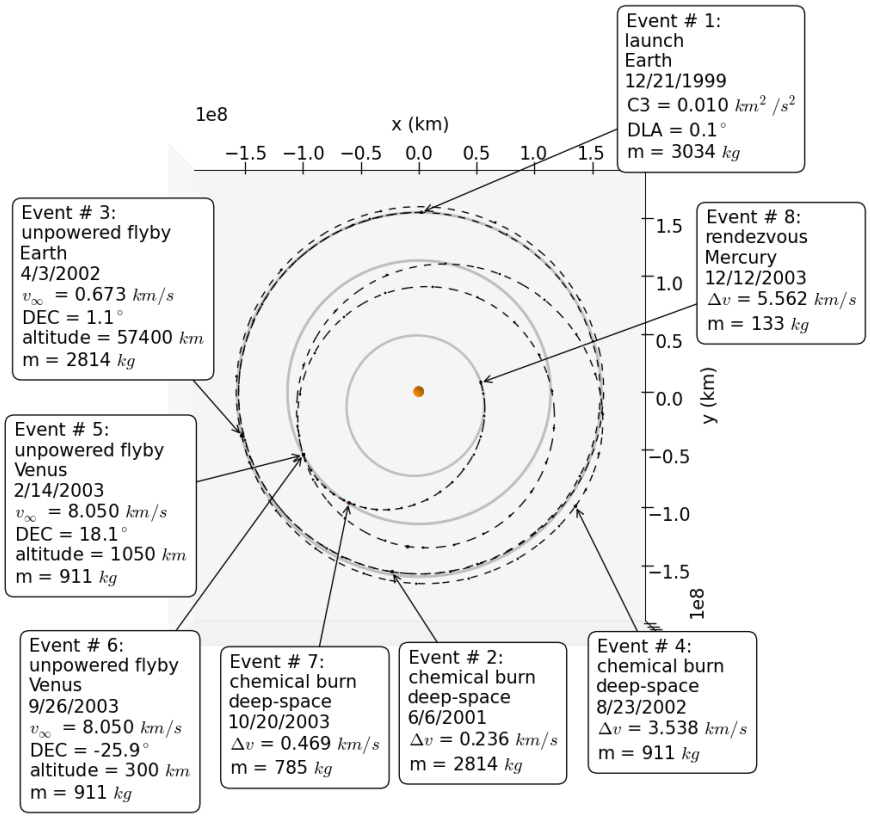

Figure 6. MESSENGER Reduced result trajectory – Not flight viable trajectory

For this problem, although comparison is improper, because our bounds on the flight times between bodies are slightly different than those found in the GTOP version of MESSENGER-reduced, the best solution found beats the GTOP record whose Δv is 8.630 km/s. As was the case with Problem 1, the only conclusion that can be drawn from this comparison is that the new optimizer presented in this work is capable of addressing this class of problem given some consideration to tuning.

Table 6 shows the tunings tested for this problem. Different versions of the MESSENGER problem were tested before testing the version shown here. Since the other problem variations had more success with this set of tunings than with others, including default, only these were tested.

Test Name	Minimum Fitness	Maximum Fitness	Mean Fitness	Standard Deviation of Fitness
DE creation percentage = 30	7.700553833	10.077949	8.69836204	0.957901076
NM max iterations = 1000 &				
NM stagnation limit = 100 &				
Pareto mutation percentage = 100	7.885601368	9.838366048	8.135282451	0.580954014
NM max iterations = 1000 &				
NM stagnation limit = 100	7.848294059	9.196548589	8.092378624	0.424963406
Disabled proximity kill	7.883441903	11.46388641	9.370414423	1.286407838
NM max iterations = 1000	7.885493493	11.46414506	8.830137907	1.18791056
NM repeat = 3	7.884693973	10.5763399	8.818799601	0.85652511
DE creation percentage = 70	7.891120342	9.968710382	8.988314296	0.931023514
NM stagnation limit = 50	7.884667061	9.950609605	8.665714852	0.95438583
NM stagnation limit = 100	7.884242668	9.951119188	8.163303062	0.632826718
NM initial step = 0.01	7.883914168	11.46634384	9.96979681	1.206551117

Table 6. Tunings tested for Problem 2

Problem 3: OSIRIS-REx: 90° Sun-Target-Angle

The total Δv for the best solution found was 0.7238 km/s launching on September 23^{rd} 2016. The Δv for the journey to Bennu was 0.4624 km/s arriving on December 22^{nd} 2018. The delivered mass to Bennu was 1594 kg. The departure date from Bennu was May 14^{th} 2021. The return date to Earth was September 25^{th} 2023 with a final mass of 1420 kg. Figure 7 shows the trajectory of this solution.

The tuning that produced this result, like that of Problem 1, was the disabling of the duplicate control measure. This led to premature convergence in many of the trials but also allowed for both recombination types to act locally. For this problem the local solver, whose default settings suited the Problem 1, required more adjustment than the algorithm. The best average was obtained through increasing the number of iterations of the Nelder-Mead algorithm and increasing the number of number of generations without improvement allowed before the solution was considered locally optimal. The best found solution of this setting was only 0.0004 km/s worse than the result shown here.

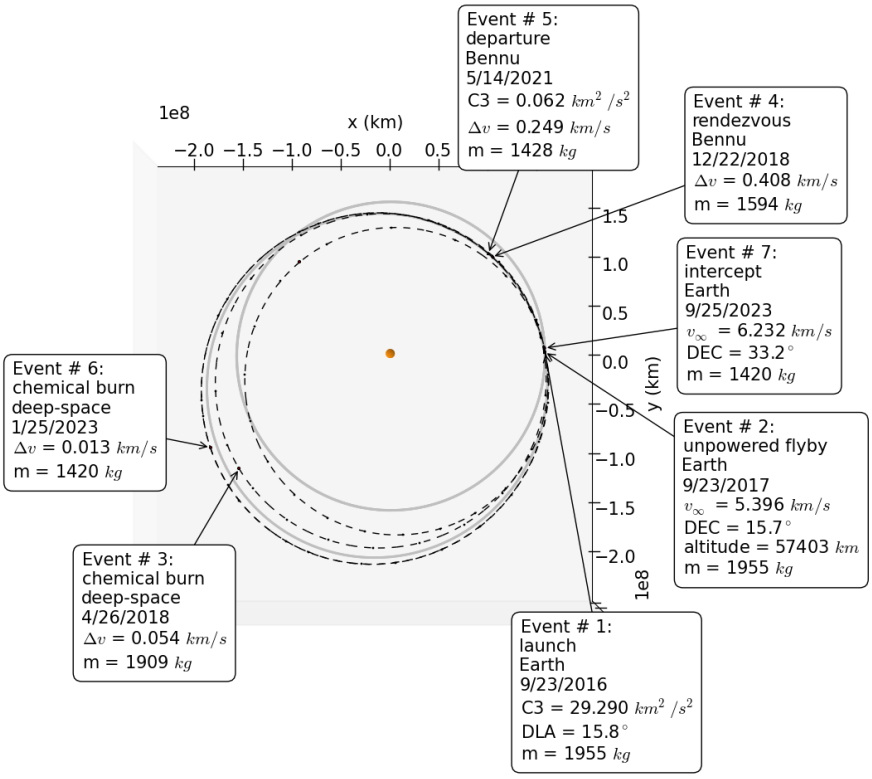

Figure 7. OSIRIS-REx 90° constraint result trajectory

Table 7 lists the tunings tested. Initial testing on an unconstrained version of this problem revealed much of the deviation required from default tuning was in the local optimizer. This makes sense as the default tunings for both the method and local optimization are based on testing for Problem 1. As of yet, a tuning law for all problems has not been established.

Test Name	Minimum Fitness	Maximum Fitness	Mean Fitness	Standard Deviation of Fitness
Disabled proximity kill	0.723849797	2.932235497	1.173036182	0.68831895
NM initial step = 0.1	0.725279214	1.796980154	0.835174524	0.320672698
NM stagnation limit = 50	0.724487126	1.233349354	0.810434537	0.154609675
NM max iterations = 1000	0.724595703	1.608428143	0.954556275	0.291337706
NM stagnation limit = 100	0.724276918	1.607709109	0.852585628	0.260491888
Kill distance = 1.2	0.725769679	1.288751624	0.802798213	0.16791975
Kill distance = 0.45	0.724017532	1.607853071	0.898080663	0.258426999
DE creation percentage = 30	0.72518409	17.29093509	3.433378663	4.746951776
NM max iterations = 1000 & NM stagnation limit = 100	0.724384127	873.0522079	122.7224839	257.1555035
DE creation percentage = 70	0.725050606	1.609679952	0.831758675	0.26446865
Pareto mutation percentage = 100	0.724466525	1.609754007	0.869859487	0.256750254
Default	0.725657145	3.738515419	1.107804668	0.881684002

Table 7. Tunings tested for Problem 3

Problem 4: OSIRIS-REx: 45° Sun-Target-Angle

For this more constrained version of the previous problem the best total Δv found was 0.7241 km/s launching on the same day as the previously reported result. The Δv for the journey to Bennu was 0.4625 km/s arriving on the same day as the solution presented for Problem 3. The delivered mass to Bennu was also the same. The departure date from Bennu was May 15^{th} 2021. The return date to Earth and the final mass were also the same as the result for Problem 3. The solution to Problem 4 here is just an adjustment of Problem 3's solution that achieves the new Sun-target-angle constraint.

The tuning that produced this result was also the tuning that best solved the problem in the most trials. The only adjustment relative to the default tuning was increasing the Nelder-Mead local solver's number of allowed generations without improvement.

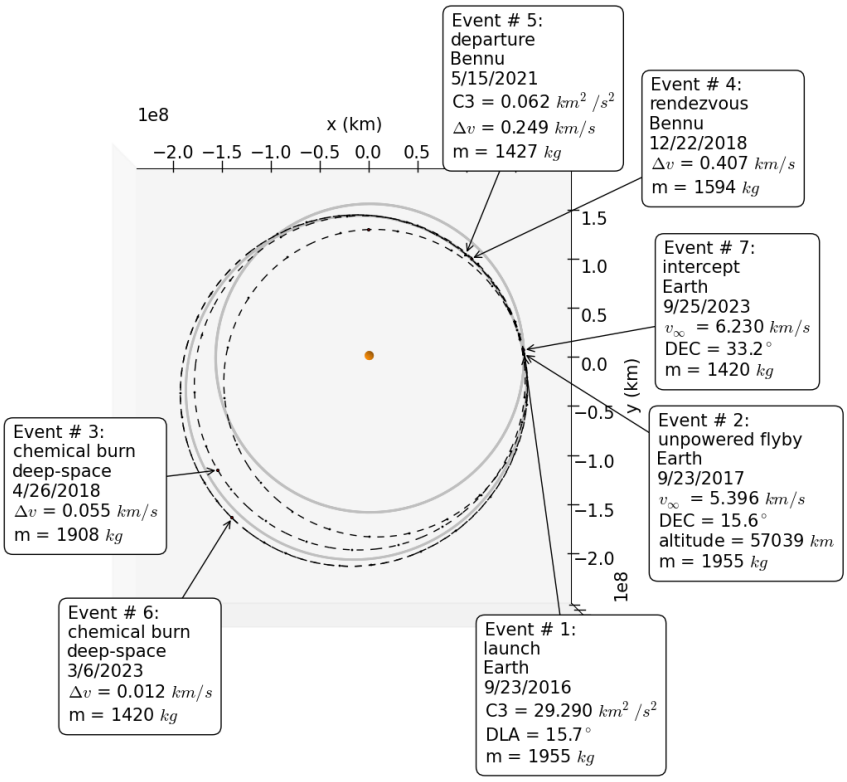

Figure 8. OSIRIS-REx 45° constraint result trajectory

Table 8 shows the tunings tested for Problem 4. These tests are identical to those for Problem 3 since the difference between the two problems is the adjustment of a single constraint. While this problem is harder due to this different constraint it was not thought too different to affect the general pattern of tunings explored.

Test Name	Minimum Fitness	Maximum Fitness	Mean Fitness	Standard Deviation of Fitness
Disabled proximity kill	0.794681232	3.797895363	1.64365942	1.101750074
Pareto mutation percentage = 100	0.799880971	6.160661388	1.468606482	1.583540108
NM max iterations = 1000	0.724436189	4.045370978	1.159974362	0.987205563
NM stagnation limit = 100	0.724134498	1.068555671	0.801919046	0.124221252
Default	0.762818765	1.64313778	0.957427469	0.248590476
Kill distance = 1.2	0.800867037	1.834345561	0.943784705	0.309631387
Kill distance = 0.45	0.754246276	5.67378172	1.451568898	1.443897955
NM initial step = 0.1	0.797778251	2.361178032	0.993787507	0.461541481
DE creation percentage = 70	0.798337374	1.084989391	0.879957468	0.105577135
NM max iterations = 1000 & NM stagnation limit = 100	0.795939001	0.96675381	0.836301535	0.065568903
DE creation percentage = 30	0.810500627	13.98726773	2.472446933	3.860479861
NM stagnation limit = 50	0.80221576	2.052751678	0.966982707	0.368824685

Table 8. Tunings tested for Problem 4

CONCLUSION

In summary, a new global stochastic optimizer is presented that is shown capable of solving interplanetary trajectory optimization problems well within reasonable time-frames. The varied tunings tested for these problems are given so as to show the need for proper tuning for a given problem. While no universal tuning law exists at the moment, problem length in terms of journey and flyby sequence is currently thought to be the biggest factor in how the algorithm must be tuned. This is beneficial as the problem's length is known before optimization, thus tunings that hinge upon this can be settled without a trial and error approach. Many of the tunings explored in this work were focused on refining the local optimizer which is wholly separate from the actual algorithm.

Future Work

As of this writing the algorithm has only been tested when coupled with a Nelder-Mead penalty-based local optimizer but the method is independent of local optimizer choice and can easily interface with others, such as the Sparse Nonlinear OPTimizer (SNOPT). Knowledge of the behavior of the method as generations progress is based on the assumption that unfeasible solutions are allowed to be returned with a penalty definition of fitness. The effect of discarding infeasible solutions may reduce the genetic diversity of the bin in a deleterious manner but the extent of such potential effects is unknown.

This algorithms approach of GA-style recombination is not limited to MGA-1DSM problems. This method can be applied to any optimization problem where disjoint groupings of decision variables have distinct meaning. An appropriate renaming of the optimizer for such an extension would the Codon Genetic Solver, borrowing terminology from biology.

The most important work to be done is to determine a tuning law that is applicable to a wide range of problems. This would allow PGS to be useful in the context of a hybrid optimal control tool,[5,6] where the individual trajectory optimization problems are generated in real-time by a discrete "outer-loop" solver. In such an environment there is no opportunity for a human analyst to study each individual trajectory design problem and assign appropriate tunings. Future work on this project will be focused on determining such a generic tuning law.

ACKNOWLEDGMENT

This work was partially supported by the Vermont Space Grant under NASA Cooperative Agreement # NNX10AK674.

REFERENCES

[1] M. Vasile and P. De Pascale, "Preliminary Design of Multiple Gravity-Assist Trajectories," *Journal of Spacecraft and Rockets*, Vol. 43, No. 4, 2006, pp. 794–805.

[2] T. Vinkó and D. Izzo, "Global Optimisation Heuristics and Test Problems for Preliminary Spacecraft Trajectory Design," Tech. Rep. GOHTPPSTD, European Space Agency, the Advanced Concepts Team, 2008.

[3] M. Vasile, E. Minisci, and M. Locatelli, "An Inflationary Differential Evolution Algorithm for Space Trajectory Optimization," *Evolutionary Computation, IEEE Transactions on*, Vol. 15, April 2011, pp. 267–281, 10.1109/TEVC.2010.2087026.

[4] A. Cassioli, D. Izzo, D. Di Lorenzo, M. Locatelli, and F. Schoen, "Global Optimization Approaches for Optimal Trajectory Planning," *Modeling and Optimization in Space Engineering* (G. Fasano and J. D. Pintr, eds.), Vol. 73 of *Springer Optimization and Its Applications*, pp. 111–140, Springer New York, 2013.

[5] J. Englander, "Multi-Objective Hybrid Optimal Control For Multiple-Flyby Interplanetary Mission Design Using Chemical Propulsion," *AAS/AIAA Astrodynamics Conference, Vail, Co*, August 9-13 2015.

[6] "EMTG (Evolutionary Mission Trajectory Generator)," https://sourceforge.net/projects/emtg/.

[7] J. Englander, B. Conway, and T. Williams, "Automated Mission Planning via Evolutionary Algorithms," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 6, 2012, pp. 1878–1887.

[8] A. Cassioli, D. Izzo, D. Di Lorenzo, M. Locatelli, and F. Schoen, "Global Optimization Approaches for Optimal Trajectory Planning," *Modeling and Optimization in Space Engineering* (G. Fasano and J. D. Pintr, eds.), Vol. 73 of *Springer Optimization and Its Applications*, pp. 111–140, Springer New York, 2013.

[9] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, Vol. 7, No. 4, 1965, pp. 308–313, 10.1093/comjnl/7.4.308.

[10] "Global Trajectory Optimization Problem Database," October 2012. http://www.esa.int/gsp/ACT/inf/op/globopt.htm.

[11] "OSIRIS-REx," http://www.asteroidmission.org/.

[12] "Cassini," http://saturn.jpl.nasa.gov.

[13] "MESSENGER," http://messenger.jhuapl.edu/.