Differential Evolution and Particle Swarm Optimization for the Multi Gravity Assist Problem

Martin Papanek
martin.papanek@gmail.com
Vienna University of Technology
Overview

- Introduction to interplanetary travel
  - Δv budget, Hohmann transfer models, Lambert problem
- Multi gravity assist problem
  - Definition, swing by model, pruning
- Differential evolution
  - Principle, algorithm, parameters
- Particle swarm optimization
  - Principles, neighborhood structure, parameters, PSO types
- Cassini mission case study
Introduction

- Interplanetary travel requires significant amounts of costly propellant mass
- Minimize fuel mass
  - Minimize number of impulses
  - Sun's orbit vs continuous impulses
  - Energy from gravitational slingshots (gravity assists)
Introduction - $\Delta v$ budget

- Engine impulse $\Rightarrow \Delta v$

- Impulse during launch, during powered gravity assists, braking impulse during orbital injection

- $\Delta v$ budget
  - Enumeration of velocity changes per maneuvers
  - measure of efficiency in km/s

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>$\Delta v$ in km/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earth Escape velocity (C3)</td>
<td>Mars Transfer Orbit</td>
<td>0.6</td>
</tr>
<tr>
<td>Mars Transfer Orbit</td>
<td>Mars Capture Orbit</td>
<td>0.9</td>
</tr>
<tr>
<td>Mars Capture Orbit</td>
<td>Deimos Transfer Orbit</td>
<td>0.2</td>
</tr>
<tr>
<td>Deimos Transfer Orbit</td>
<td>Deimos surface</td>
<td>0.7</td>
</tr>
<tr>
<td>Deimos Transfer Orbit</td>
<td>Phobos Transfer Orbit</td>
<td>0.3</td>
</tr>
<tr>
<td>Phobos Transfer Orbit</td>
<td>Phobos surface</td>
<td>0.5</td>
</tr>
<tr>
<td>Mars Capture Orbit</td>
<td>Low Mars Orbit</td>
<td>1.4</td>
</tr>
</tbody>
</table>
Interplanetary Travel

- Two impulse orbit-to-orbit transfer (Hohmann transfer orbit)
  - Generalization => two impulse planet-to-planet
  - Example Earth-to-Mars transfer
- Low thrust planet-to-planet transfer
- Multi gravity assist and gravity assist planet-to-planet transfer
  - Special case Multi gravity assist with deep space maneuvers
  - Example Earth-Saturn transfer
Hohmann transfer orbit

- Basic low Δv maneuver
- Altering the spacecrafts orbit around the sun through accelerating/decelerating
Lambert Problem

- A Keplerian orbit about a given gravitational center of force is to be found connecting two points P1 and P2 in a given time $\Delta t$

- Input:
  - position vector at P1 and P2 (ephemeris)
  - transfer time $\Delta t$ (constraint)

- Output:
  - velocity at departure
  - velocity at arrival

- Goal: pick the solution with best $\Delta v$
Gravity Assist

- Gravity assist
  - using a part of a celestial objects angular momentum
  - spacecraft flies close enough to the swing by planet to get caught by its gravity
  - impulse at the appropriate point to gain even more momentum
  - Enables the spacecraft to accelerate/decelerate
- Direction of approach
  - Front => decelerate
  - Back => accelerate
- Energy is lost by the planet and gained by the spacecraft
  => conservation of energy
GA – Swing by model

- Parameters to swing by model
  - Incoming and outgoing velocity
  - Minimum pericenter radius
  - Gravitational constant of the celestial object
- Returns required impulse $\Delta v$
- Powered swing-by vs unpowered
Gravity Assist

- The before and after velocities are equal in magnitude with respect to the planet.
- With respect to the sun the magnitude of the after velocity is greater.
- Spacecraft accelerates with respect to the sun.
Multi Gravity Assist Problem

- MGA problem formalization
  - Sequence of N+2 planets
  - N planets to exploit in a gravity assist maneuver
  - Vector $x = [t_0, T_1, T_2, \ldots, T_{N+1}]$
  - Departure epoch $t_0$
  - $T_i$ durations to travel along arcs joining two planets

- Patched conic approach
  - Trajectory constructed from planet-to-planet arcs
  - Lambert problems
Multi Gravity Assist Problem

- EVVEJS trajectory
- Best solution
- EVVE close-up
Multi Gravity Assist problem

- Challenges:
  - Sequential solving of arcs
    - => no analytical representation
    - => no gradients
  - Continuous search space
  - Deep space maneuvers
    - => increases search space
  - Establishing launch window
  - Search space pruning?
    - GASP method
European Space Agency framework

- ESA framework for the multi gravity problem
  - Encapsulates multi gravity assist missions
  - Calculates ephemeris
  - Solves Lambert problems
- Goal: Optimization heuristic for MGA missions
- Global optimization heuristics for continuous problems
  - Particle swarm optimization
  - Differential evolution
Differential Evolution

- Created by Kenneth Price and Rainer Storm 1995
- Population-based evolution
  - Generating perturbations (mutants) from random candidates
  - Differential mutation
  - Trail vector through recombination between mutant vector and a vector from the current population
  - Trial vector competes against the current vector
- Vector difference well suited for continuous search spaces
  - No need for bit string to real number mapping like in GA
- Simple, efficient and fast
DE - Principles

- Differential Mutation:

\[ v_{i,g} = x_{r1,g} + F(x_{r2,g} - x_{r3,g}) \]

- Uniform Crossover (discrete recombination):

\[ u_{j,i,g} = \begin{cases} 
  v_{j,i,g} & \text{if } (\text{rand}_j(0,1) \leq Cr \lor j = \text{random index}) \\
  x_{j,i,g} & \text{otherwise}
\end{cases} \]

  Crossover probability Cr => approximation of the true probability distribution

- Selection:

\[ x_{i,g+1} = \begin{cases} 
  u_{i,g} & \text{if } f(u_{i,g}) \leq f(x_{i,g}) \\
  x_{i,g} & \text{otherwise}
\end{cases} \]
DE - Algorithm

A vector population is generated such that the allowed parameter region is entirely covered.

All vectors get a unique index for bookkeeping because each of them has to enter a competition.

$u_0 = x_r3 + F \cdot (x_{r1} - x_{r2})$

$x_{r3}$ is another randomly selected vector which, together with the weighted difference vector, yields the trial vector $u_0$.

$x_{r1}$ and $x_{r2}$ are two randomly selected vectors from the vector population.

$u_0$ competes against the vector no. 0 of the population.

The vector with the lower objective function value gets marked as vector no. 0 of the next population.
DE - Parameters

- Crossover parameter $Cr$
  - Responsible for the diversity
  - When $Cr = 1$ loss of diversity
  - When $Cr = 0$ only one component of the trial vector is taken over

- Amplification(Scale) parameter $F$
  - May be constant since the step size is also affected by the probability distribution of the vector differences
  - $F$ from the interval $(0,1)$
  - Influences convergence
  - $F$ as random variable useful when population small
Particle Swarm Optimization

- Kennedy and Ebenhart in 1995
- Inspiration
  - Evolutionary algorithms
  - Bird flocking
  - Fish schooling
  - Artificial intelligence
- Numerical (continuous) as well as combinatorial problems
- Applications:
  - Neural network training, telecommunications, data mining etc.
PSO – Principles 1

- Candidate solutions are represented as particles in a swarm (population)
- Particle
  - Velocity
  - Position
  - Fitness of a position
  - Memory of particle's best position
  - Neighborhood consisting of other particles
- Fitness function evaluates positions
- Particles attempt to move to the optimal position
PSO – Principles 2

- Velocity and position updated each iteration

\[
\begin{align*}
\vec{v}_i(t+1) & \leftarrow \vec{v}_i(t) + \vec{U}(0, c_1) \times (\vec{p}_i - \vec{x}_i(t)) + \vec{U}(0, c_2) \times (\vec{g}_i - \vec{x}_i(t)) \\
\vec{x}_i(t+1) & \leftarrow \vec{v}_i(t+1) + \vec{x}_i(t)
\end{align*}
\]

- New position determined by a combination of particle's own best position and the best position in the neighborhood

- Problems:
  - How to pick the neighborhood structure?
  - What values should the parameters have?
PSO – Neighborhood structure

- Neighbors determined by the Euclidian proximity of particles
  - Real-life approach
  - Costly computation
- Neighbors determined by the connections in a communication graph
  - Fully connected (global best topology)
  - Random graph topology
  - Star topology
  - Ring topology
PSO - Parameters

- **Momentum**
  - Tendency to continue its current direction

- **Cognitive component**
  - Tendency to return to particle's own best solution

- **Social component**
  - Governs the influence by the best solution in its neighborhood

- **Low vs high social and cognitive component**
  - Social > cognitive => better for unimodal functions
  - Cognitive > social => better for multimodal functions
PSO – types 1

- Inertia based PSO
  - inertia parameter c0 to constrict momentum
  - Inertia parameter as a balance between exploration and exploitation

\[ \vec{v}_i(t+1) \leftarrow c_0 \vec{v}_i(t) + \vec{U}(0, c_1) \times (\vec{p}_i - \vec{x}_i(t)) + \vec{U}(0, c_2) \times (\vec{g}_i - \vec{x}_i(t)) \]

- Constricted coefficients PSO
  - introduces constriction factor
  - Constriction makes incorrect parameter choice less severe

\[ \vec{v}_i(t+1) \leftarrow X(\vec{v}_i(t) + \vec{U}(0, c_1) \times (\vec{p}_i - \vec{x}_i(t)) + \vec{U}(0, c_2) \times (\vec{g}_i - \vec{x}_i(t))) \]

\[ X = \frac{2}{2 - c - \sqrt{c^2 - 4 * c}} \]
PSO – types 2

- Fully informed PSO
  - incorporates the local best of every particle

\[
\vec{v}_i(t+1) \leftarrow X[ \vec{v}_i(t) + \frac{1}{|N_i|} \sum_{b_j \in N_i} \tilde{U}(0, c_1) \times (\vec{b}_j - \vec{x}_i(t))] 
\]

- Bare bones PSO
  - Positions are obtained by sampling a normal distribution
  - Particles do not have velocities

- Multiple PSO (MPSO)
  - Several swarms coexist
  - Every nth iteration, swarms trade several particles
Cassini mission

- Planet sequence:
  - Earth, Venus, Venus, Earth, Jupiter, Saturn (EVVEJS)
- Destination is Saturn's orbit with an eccentricity of 0.98 and pericenter radius of 108950km
- The function to be minimized is the \( \Delta v \) budget of the mission
- Real life mission Cassini/Huygens
  - Cost => 3.26 billion US$
  - With an overall \( \Delta v \) of approximately 7.1 km/s and mass 5600kg
  - Included deep space maneuver between Venus swing-bys
Cassini mission - constraints

- Each segment of the trajectory has an upper and lower bound

<table>
<thead>
<tr>
<th>Arc</th>
<th>Variable</th>
<th>LB</th>
<th>UB</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Takeoff</td>
<td>t0</td>
<td>-1000</td>
<td>0</td>
<td>MJD2000</td>
</tr>
<tr>
<td>Earth – Venus</td>
<td>T1</td>
<td>30</td>
<td>400</td>
<td>days</td>
</tr>
<tr>
<td>Venus – Venus</td>
<td>T2</td>
<td>100</td>
<td>470</td>
<td>days</td>
</tr>
<tr>
<td>Venus – Earth</td>
<td>T3</td>
<td>30</td>
<td>400</td>
<td>days</td>
</tr>
<tr>
<td>Earth – Jupiter</td>
<td>T4</td>
<td>400</td>
<td>2000</td>
<td>days</td>
</tr>
<tr>
<td>Jupiter – Saturn</td>
<td>T5</td>
<td>1000</td>
<td>6000</td>
<td>days</td>
</tr>
</tbody>
</table>

- Minimal pericenter radii at each planet:

\[ rp_1 > 6351.8 \text{ km} \]
\[ rp_2 > 6351.8 \text{ km} \]
\[ rp_3 > 6778.1 \text{ km} \]
\[ rp_4 > 671492 \text{ km} \]
Search space analysis

Search space analysis suggests a combination with local optimization
Nelder-Mead Downhill simplex

- Local optimization algorithm published in 1965
- Multidimensional unconstrained optimization without derivatives
  - Only requires function evaluations no gradients
- Simplex-based search
  - Simplex a convex hull of n+1 vertexes
- Principle:
  - reduce the function values at the vertexes through geometric transformation
Nelder-Mead - Algorithm

- Step 1: Initialize simplex
  - Right angled or regular simplex
- Step 2: order indices
  - Determine worst, second worst and best
- Step 3: calculate centroid of the best side
- Step 4: transformation
  - Reflect
  - Expand
  - Contract
  - Shrink
- Step 5: termination test
  - If test fails goto Step 2
Nelder-Mead Downhill simplex
Convergence of PSO
- Slower convergence
- Convergence depends on parameter choice

Convergence of Differential evolution
- Quick convergence
- Monotonically decreasing
Cassini - Comparison of heuristics

Results:

<table>
<thead>
<tr>
<th>Best solution</th>
<th>Algorithm</th>
<th>Author</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0088 km/s</td>
<td>Monotonic basin hopping method</td>
<td>Bernardetta Addis</td>
<td>Florence University</td>
</tr>
<tr>
<td>4.934 km/s</td>
<td>Differential Evolution</td>
<td>Fabio Pinna and Dario Izzo</td>
<td>European Space Agency</td>
</tr>
<tr>
<td>4.9307 km/s</td>
<td>modified version of Particle Swarm Optimisation</td>
<td>Manfred Stickel</td>
<td>Max-Planck-Institut fuer Astronomie</td>
</tr>
</tbody>
</table>

DE with local optimization vs PSO with local optimization
- Population n=1000, maximum of 3000 iterations, 20 samplings

<table>
<thead>
<tr>
<th>Best solution</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Algorithm</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.93241 km/s</td>
<td>4.9831</td>
<td>0.0584</td>
<td>Constrained coefficient PSO</td>
<td>Phi1=2, Phi2= 2, RING</td>
</tr>
<tr>
<td>4.93187 km/s</td>
<td>5.2649</td>
<td>1.0731</td>
<td>Constrained coefficient PSO</td>
<td>Phi1=2, Phi2= 2, STAR</td>
</tr>
<tr>
<td>4.93071 km/s</td>
<td>4.9867</td>
<td>0.1338</td>
<td>DE</td>
<td>F=0.4, Cr=0.4</td>
</tr>
<tr>
<td>4.94792 km/s</td>
<td>5.2858</td>
<td>0.0769</td>
<td>DE</td>
<td>F=0.2, Cr=0.7</td>
</tr>
</tbody>
</table>
Case Study - Parameters

- **Parameters for PSO**
  - Inertia based PSO
  - Population n=1000
  - Best 4.95223 for Phi1 = 0.5 Phi2 = 1

<table>
<thead>
<tr>
<th>Phi1</th>
<th>Phi2</th>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>6.5346</td>
<td>0.33840</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>5.7084</td>
<td>0.28137</td>
</tr>
<tr>
<td>0.5</td>
<td>1.5</td>
<td>5.7890</td>
<td>0.21128</td>
</tr>
<tr>
<td>0.5</td>
<td>2</td>
<td>5.5307</td>
<td>0.04856</td>
</tr>
<tr>
<td>1</td>
<td>0.5</td>
<td>8.1600</td>
<td>5.31780</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>5.8747</td>
<td>0.14590</td>
</tr>
<tr>
<td>1</td>
<td>1.5</td>
<td>5.8696</td>
<td>0.55520</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>5.7799</td>
<td>0.08906</td>
</tr>
<tr>
<td>1.5</td>
<td>0.5</td>
<td>7.2315</td>
<td>1.66621</td>
</tr>
<tr>
<td>1.5</td>
<td>1</td>
<td>5.7826</td>
<td>0.16324</td>
</tr>
<tr>
<td>1.5</td>
<td>1.5</td>
<td>6.0026</td>
<td>0.25869</td>
</tr>
<tr>
<td>1.5</td>
<td>2</td>
<td>29.9183</td>
<td>54.97590</td>
</tr>
<tr>
<td>2</td>
<td>0.5</td>
<td>8.9165</td>
<td>5.85123</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5.6898</td>
<td>0.27528</td>
</tr>
<tr>
<td>2</td>
<td>1.5</td>
<td>31.9386</td>
<td>45.47970</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>28.3266</td>
<td>47.75490</td>
</tr>
</tbody>
</table>

- **Parameters for DE**
  - Population n=1000
  - Best 4.93071 for F = 0.4 Cr = 0.4

<table>
<thead>
<tr>
<th>Cr</th>
<th>F</th>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>4.9938</td>
<td>0.002962</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2</td>
<td>4.9354</td>
<td>0.000527</td>
</tr>
<tr>
<td>0.6</td>
<td>0.2</td>
<td>4.9319</td>
<td>0.000162</td>
</tr>
<tr>
<td>0.8</td>
<td>0.2</td>
<td>4.9310</td>
<td>0.000039</td>
</tr>
<tr>
<td>1</td>
<td>0.2</td>
<td>4.9308</td>
<td>0.000611</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4</td>
<td>4.9770</td>
<td>0.001771</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.6</td>
<td>0.4</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.8</td>
<td>0.4</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>1</td>
<td>0.4</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.6</td>
<td>5.1548</td>
<td>0.033637</td>
</tr>
<tr>
<td>0.4</td>
<td>0.6</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.8</td>
<td>0.6</td>
<td>4.9307</td>
<td>0.000000</td>
</tr>
<tr>
<td>1</td>
<td>0.6</td>
<td>4.9307</td>
<td>0.000098</td>
</tr>
<tr>
<td>0.2</td>
<td>0.8</td>
<td>4.9677</td>
<td>0.001857</td>
</tr>
</tbody>
</table>
References


[6] LIACS Natural Computing Group Leiden University, “Particle Swarm Optimization”
Thank you for your attention!

Any questions?