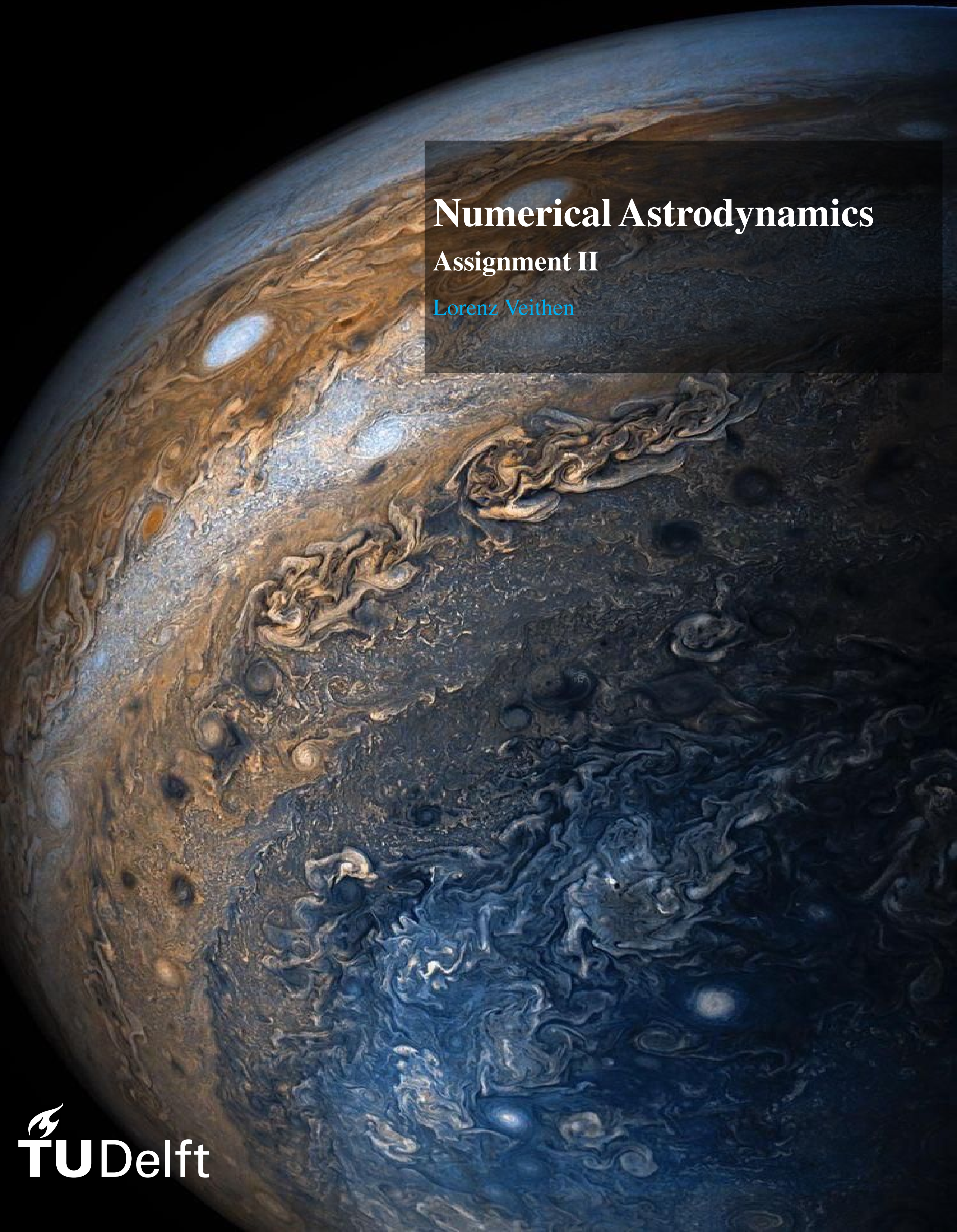

Numerical Astrodynamics

Assignment II

Lorenz Veithen

DELFT UNIVERSITY OF TECHNOLOGY

FACULTY OF AEROSPACE ENGINEERING
AE4868-1
NUMERICAL ASTRODYNAMICS

Numerical Propagation of JUICE

Lecturer: Dr. D. Dirkx and Dr. Ir. K.J. Cowan
Github repository link:
https://github.com/LorenzVeithen/NumericalAstrodynamics2023_Veithen_Lorenz
April 2, 2023

Q1	Q2	Q3	Q4	Q5
1h	6h	6h	12h	15h

Lorenz Veithen 5075211

Cooperating Students: Srujan Vaidya (5072034) and Oliver Ross (5008042)

Delft University of Technology

Problem 1

1.1 Numerical Model

The numerical solution shown in Figure 1.1 is computed from an unperturbed P-M (point-mass) Sun gravity field acceleration model (Keplerian orbit). The mathematical model solved is an exact representation of the Lambert arc model, as it follows the two main assumptions behind it.

1. **Central body is located at the origin (non-moving)**: the Sun is located at the origin of the reference frame, and therefore does not move, which verifies the assumption.
2. **Only acceleration on the spacecraft is the central body P-M**: the acceleration model is an unperturbed P-M central body gravity field, which verifies the assumption.

Table 1.1: Magnitude of Cartesian difference between Mars, or Earth and the spacecraft at start or end epoch. S/C: spacecraft. This shows that the spacecraft indeed starts from Earth and reaches Mars centers within the specified time of flight, which is expected from the Lambert model.

	$\Delta t = 3600$s, [km]
$\|\|\vec{r}_{Earth} - \vec{r}_{S/C}\|\|$ start epoch	3.432e-7
$\|\|\vec{r}_{Mars} - \vec{r}_{S/C}\|\|$ end epoch	3.487e-06

1.2 Source of Errors

Numerical errors give rise to the differences shown in Figure 1.2, this is seen from their very small magnitude (1e-6 km, while the position values are of the order of 1e8 km). Numerical errors also typically give rise to the spiky oscillations in Figure 1.2. ΔZ is smaller than ΔX and ΔY as its absolute coordinate value is smaller.

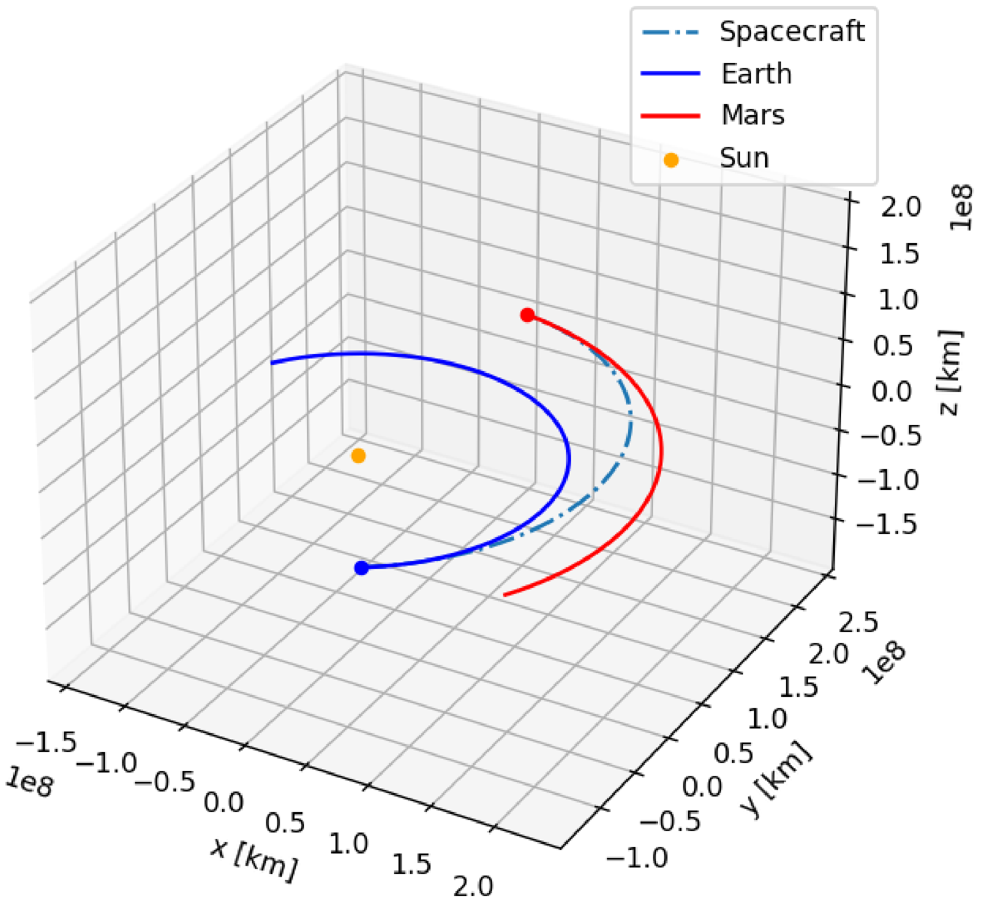

Figure 1.1: Three-dimensional Cartesian position of Earth, Mars and the spacecraft during the interplanetary transfer. The initial position of the Earth (and the spacecraft), and the final position of Mars (and the spacecraft) are also shown by blue and red spheres.

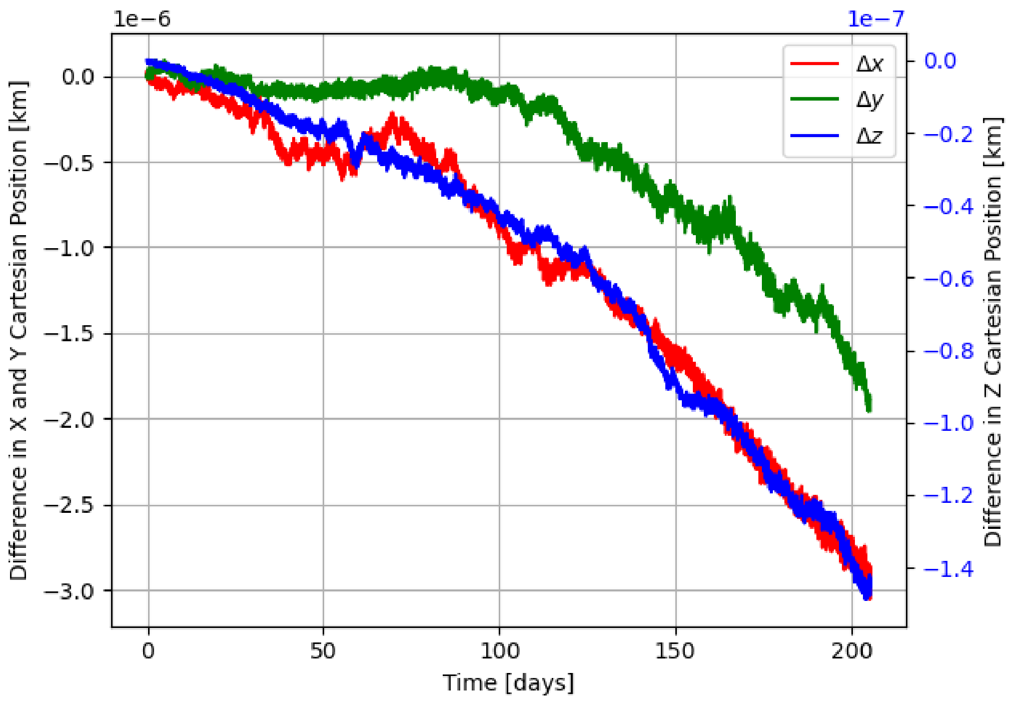

Figure 1.2: Cartesian difference between the numerical and the lambert targeter results. Note that the image-axis relative to the X and Y coordinates is on the left while the Z coordinate is plotted against the right axis. The difference is taken as numerical minus lambert arc value.

Problem 2

2.1 Differences with Lambert Model

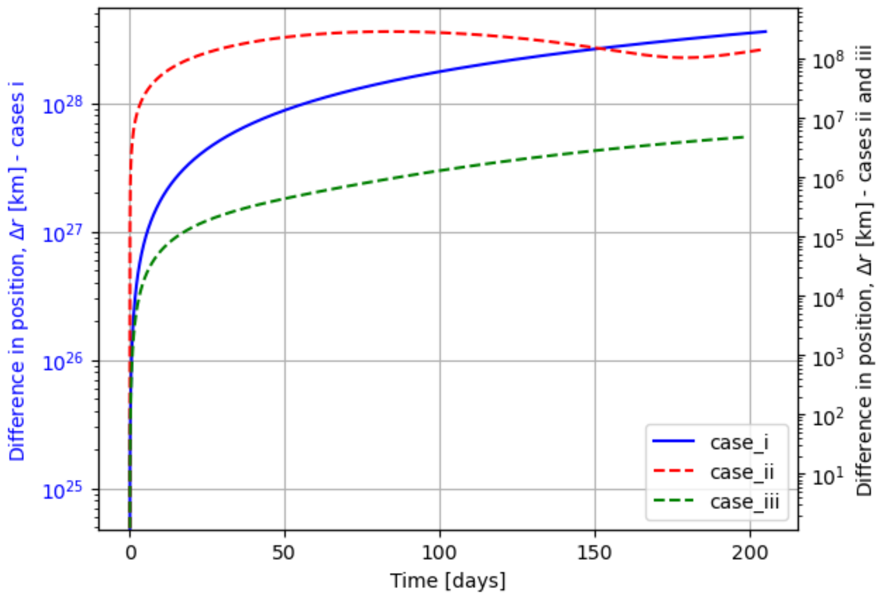

Figure 2.1: Difference in position $\Delta r = ||\vec{r}(t) - \vec{\bar{r}}(t)||$ for cases i, ii, and iii with a start to end propagation and time step $\Delta t = 3600$s. Note that case i is plotted against the left y-axis while case ii and iii are plotted on the right.

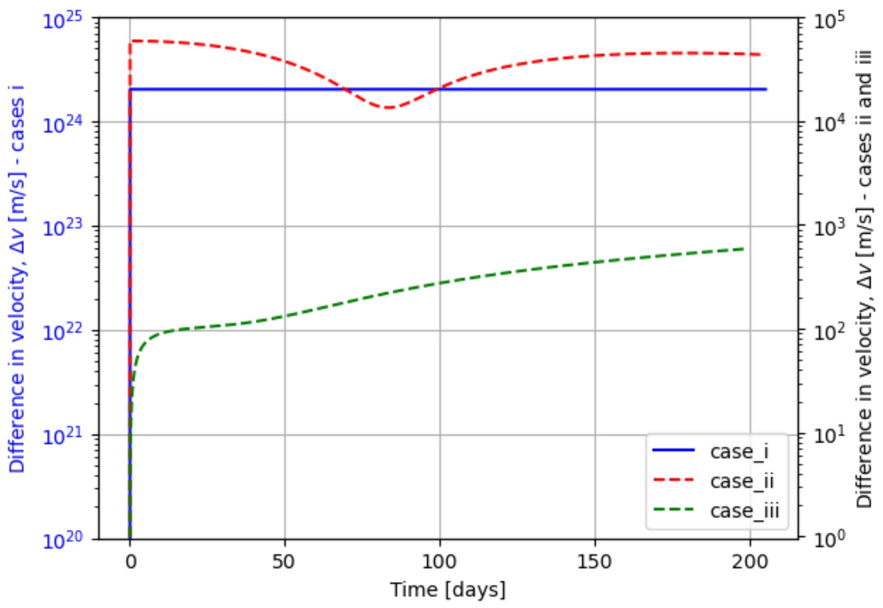

Figure 2.2: Difference in position $\Delta v = ||\vec{V}(t) - \vec{\bar{V}}(t)||$ for cases i, ii, and iii with a start to end propagation and time step $\Delta t = 3600$s. Note that case i is plotted against the left y-axis while case ii and iii are plotted on the right.

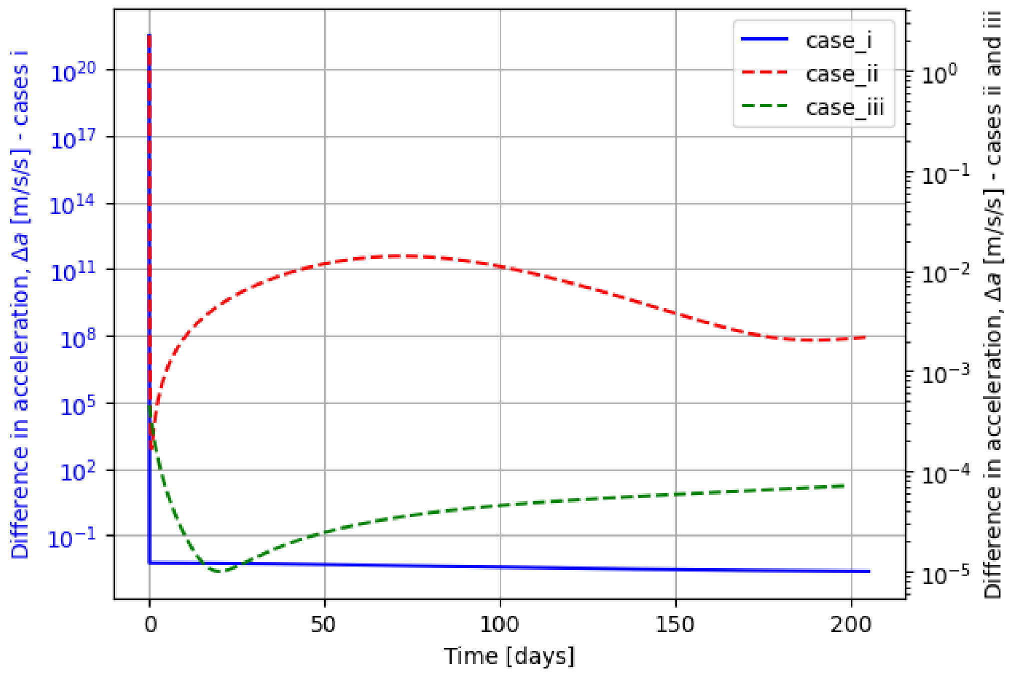

Figure 2.3: Difference in position $\Delta a = ||\vec{a}(t) - \vec{\bar{a}}(t)||$ for cases i, ii, and iii with a start to end propagation and time step $\Delta t = 3600$s. Note that case i is plotted against the left y-axis while case ii and iii are plotted on the right.

2.2 Case ii Mid-Point Propagation

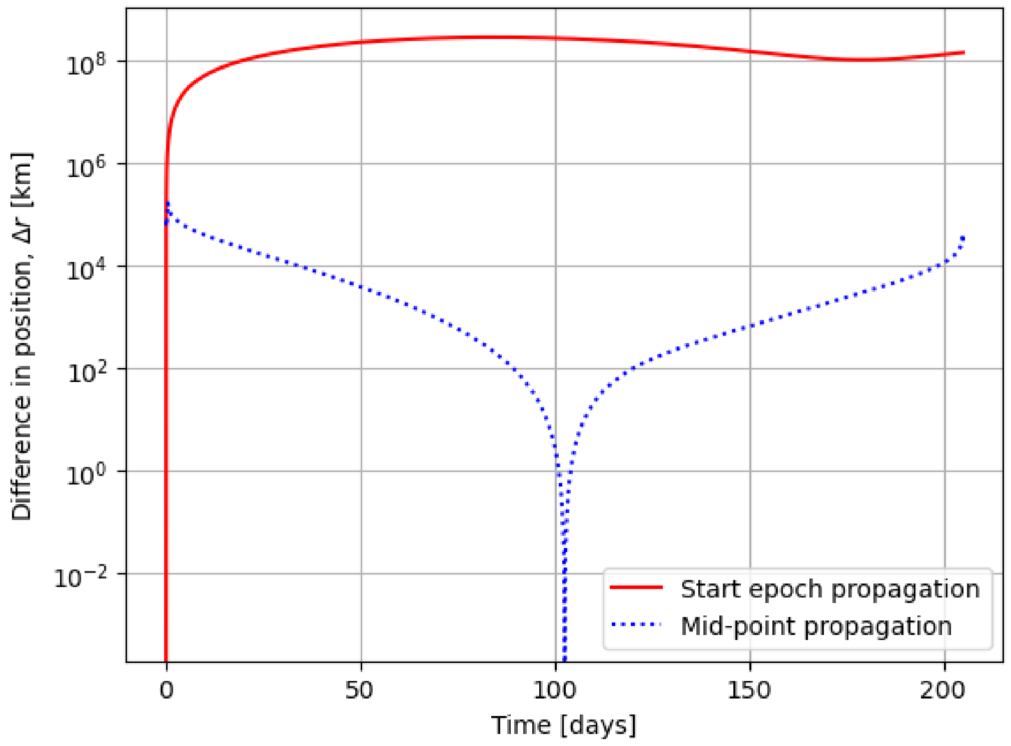

Figure 2.4: Difference in position $\Delta r = ||\vec{r}(t) - \vec{\bar{r}}(t)||$ for case ii using the mid-point propagation approach and time step $\Delta t = 3600$s.

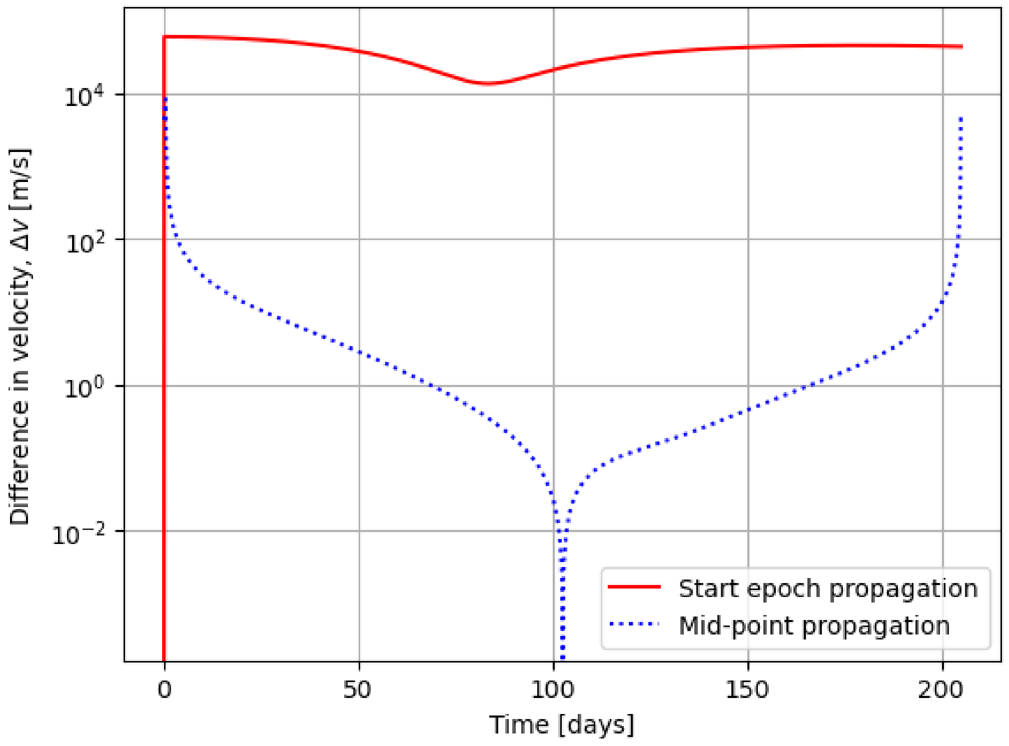

Figure 2.5: Difference in position $\Delta v = ||\vec{V}(t) - \vec{\bar{V}}(t)||$ for case ii using the mid-point propagation approach and time step $\Delta t = 3600$s.

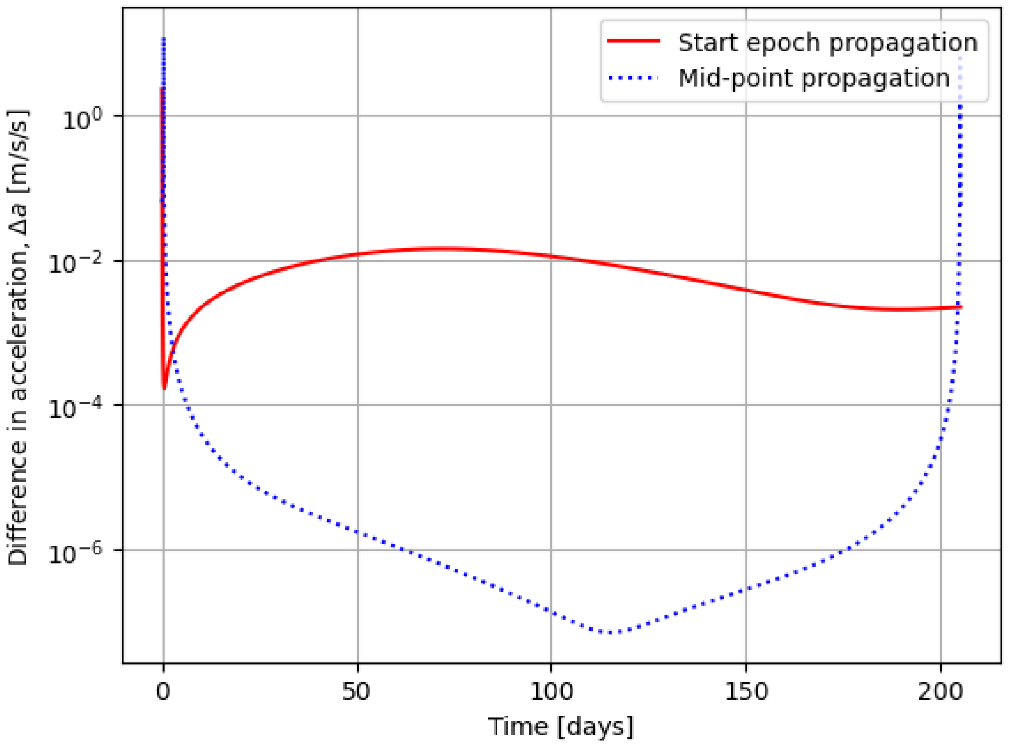

Figure 2.6: Difference in position $\Delta a = ||\vec{a}(t) - \vec{\bar{a}}(t)||$ for case ii using the mid-point propagation approach and time step $\Delta t = 3600$s.

2.3 Derivation of $\frac{d\Delta r}{dt}$ and $\frac{d\Delta v}{dt}$

$$\frac{d}{dt}(||\mathbf{b}||) = \frac{d}{dt}\left(\sqrt{\mathbf{b}\cdot\mathbf{b}}\right) = \frac{1}{2\sqrt{\mathbf{b}\cdot\mathbf{b}}}\frac{d(\mathbf{b}\cdot\mathbf{b})}{dt} = \frac{1}{2\sqrt{\mathbf{b}\cdot\mathbf{b}}}\left(\mathbf{b}\cdot\frac{d(\mathbf{b})}{dt} + \frac{d(\mathbf{b})}{dt}\cdot\mathbf{b}\right) = \frac{\mathbf{b}}{||\mathbf{b}||}\cdot\frac{d(\mathbf{b})}{dt}$$

$$\begin{aligned}\frac{d}{dt}(\Delta r) = \frac{d}{dt}(||\mathbf{r}-\bar{\mathbf{r}}||) &= \frac{(\mathbf{r}-\bar{\mathbf{r}})}{||\mathbf{r}-\bar{\mathbf{r}}||}\cdot\frac{d(\mathbf{r}-\bar{\mathbf{r}})}{dt} \\ &= \frac{(\mathbf{r}-\bar{\mathbf{r}})}{||\mathbf{r}-\bar{\mathbf{r}}||}\cdot(\mathbf{v}-\bar{\mathbf{v}}) = \frac{\Delta\mathbf{r}\cdot\Delta\mathbf{v}}{||\Delta\mathbf{r}||} = \frac{\Delta\mathbf{r}\cdot\Delta\mathbf{v}}{\Delta r} \end{aligned} \tag{2.1}$$

$$\begin{aligned}\frac{d}{dt}(\Delta v) = \frac{d}{dt}(||\mathbf{v}-\bar{\mathbf{v}}||) &= \frac{(\mathbf{v}-\bar{\mathbf{v}})}{||\mathbf{v}-\bar{\mathbf{v}}||}\cdot\frac{d(\mathbf{v}-\bar{\mathbf{v}})}{dt} \\ &= \frac{(\mathbf{v}-\bar{\mathbf{v}})}{||\mathbf{v}-\bar{\mathbf{v}}||}\cdot((\mathbf{a}_p)_S - \bar{\mathbf{a}}) \\ &= \frac{(\mathbf{v}-\bar{\mathbf{v}})}{||\mathbf{v}-\bar{\mathbf{v}}||}\cdot(\mathbf{a}_{Sun} + \mathbf{a}_{pert} - \bar{\mathbf{a}}) \qquad \mathbf{a}_{Sun} = \bar{\mathbf{a}} \\ &= \frac{(\mathbf{v}-\bar{\mathbf{v}})}{||\mathbf{v}-\bar{\mathbf{v}}||}\cdot(\mathbf{a}_{pert}) = \frac{\Delta\mathbf{v}\cdot\mathbf{a}_{pert}}{||\Delta\mathbf{v}||} = \frac{\Delta\mathbf{v}\cdot\mathbf{a}_{pert}}{\Delta v}\end{aligned} \tag{2.2}$$

2.4 Different Δr Behaviour for Difference Cases

Eq. (2.2) describes that Δv grows with time due to the component of $\mathbf{a}_{pert}$ along the difference of the velocity vector directions ($\frac{(\mathbf{v}-\bar{\mathbf{v}})}{||\mathbf{v}-\bar{\mathbf{v}}||}$), yielding a growth in Δr. Once $\Delta v \neq 0$, Δr grows over the entire trajectory (buildup).

$$(\vec{a}_{E,p})_S = \mu_E\left(\frac{\hat{r}_{E,S}}{||\vec{r}_{E,S}||^2} - \frac{\hat{r}_{E,p}}{||\vec{r}_{E,p}||^2}\right) \tag{2.3}$$

$$(\vec{a}_{M,p})_S = \mu_M\left(\frac{\hat{r}_{M,S}}{||\vec{r}_{M,S}||^2} - \frac{\hat{r}_{M,p}}{||\vec{r}_{E,p}||^2}\right) \tag{2.4}$$

Close to either Mars or Earth, their 3rd-body accelerations form the major component of $\mathbf{a}_{pert}$ (maxima of $\mathbf{a}_{pert}$ is reached close to these bodies), and are larger than $\mathbf{a}_{Sun}$ while within their SOIs. Increasing the buffer time permits to start and end the propagation at positions further away from the bodies' centres, hence $\mathbf{a}_{pert}$ does not become as large as for propagations without buffer time, and the growth in Δr and Δv is slower according to Eqs. (2.1) (2.2). For propagations starting within Earth's SOI, the dominant acceleration ($(\vec{a}_{E,p})_S$) acting on the spacecraft is not modelled by the Lambert arc, meaning that the same initial conditions will yield very different trajectories (cases i and ii). For propagations starting outside the SOI, the same accelerations are less significant (closer to the Lambert assumptions), and the trajectory diverges less from the Lambert arc (case iii). For midpoint propagations, the initial $\mathbf{a}_{pert}$ is much smaller, and Δr and Δv grow very slowly (Eq. (2.2)) over the interplanetary transfer. At arrival at either Earth (back-propagation) or Mars, Δr and Δv become larger (larger $\mathbf{a}_{pert}$), but the propagation is stopped before the spacecraft slingshots or crashes around/on the body (as would be expected without decelerating burn), and Δr does not grow further, yielding a lower maximum. Δr is then smaller because the trajectory does not diverge completely from the Lambert arc at its start.

Table 2.1: Key differences between cases, to link each case to the discussion above.

case	**Particularity**
case i	No buffer time: $\vec{r}_{E,p} \approx \vec{0} \Rightarrow (\vec{a}_{E,p})_S \to +\infty$ at the start epoch, and the spacecraft is sent into interstellar space
case ii	Small buffer: spacecraft in SOI at start epoch
case iii	Larger buffer: spacecraft outside SOI at start epoch
case ii mid-point	Propagation from the mid-point

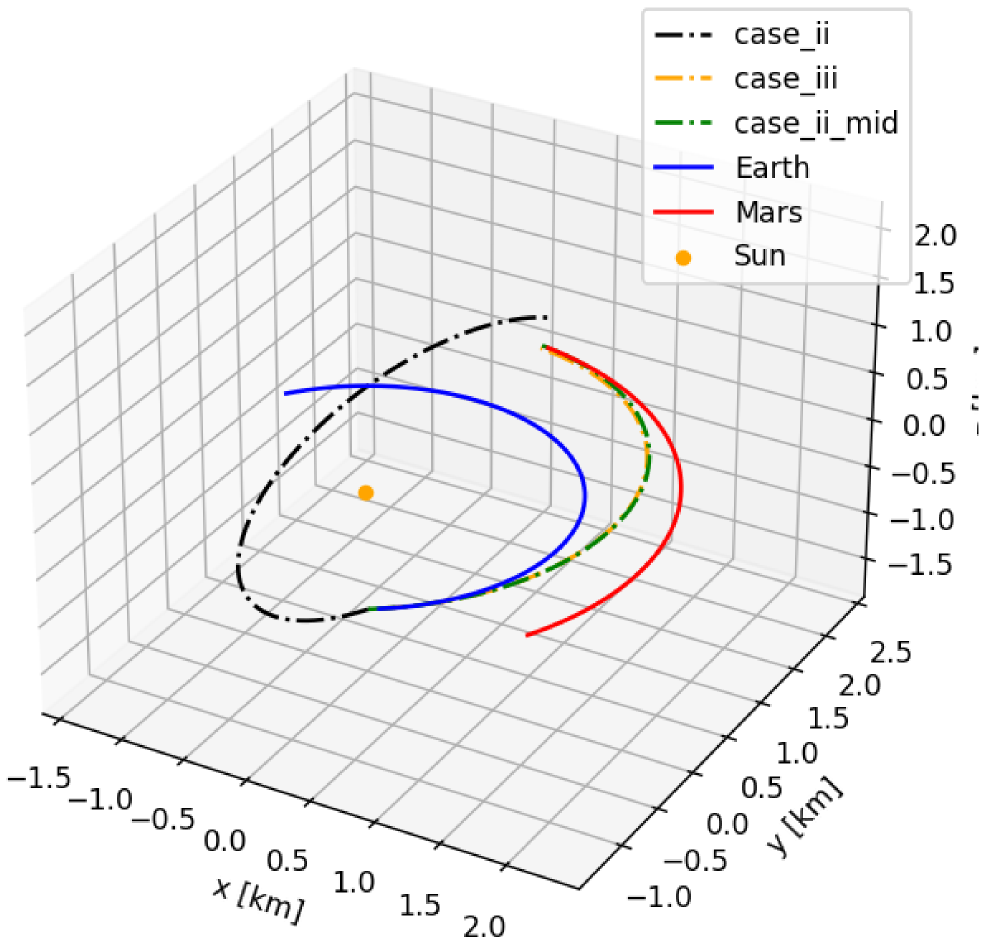

Figure 2.7: 3D plot of the trajectory for cases ii, iii, and ii using the midpoint propagation. This clearly shows that case ii does not follow a that will lead to Mars, due to the initial disturbing acceleration of Earth being dominant. The mid-point case ii and case iii yield a logical trajectory to Mars.

Problem 3

3.1 Propagation without trajectory correction

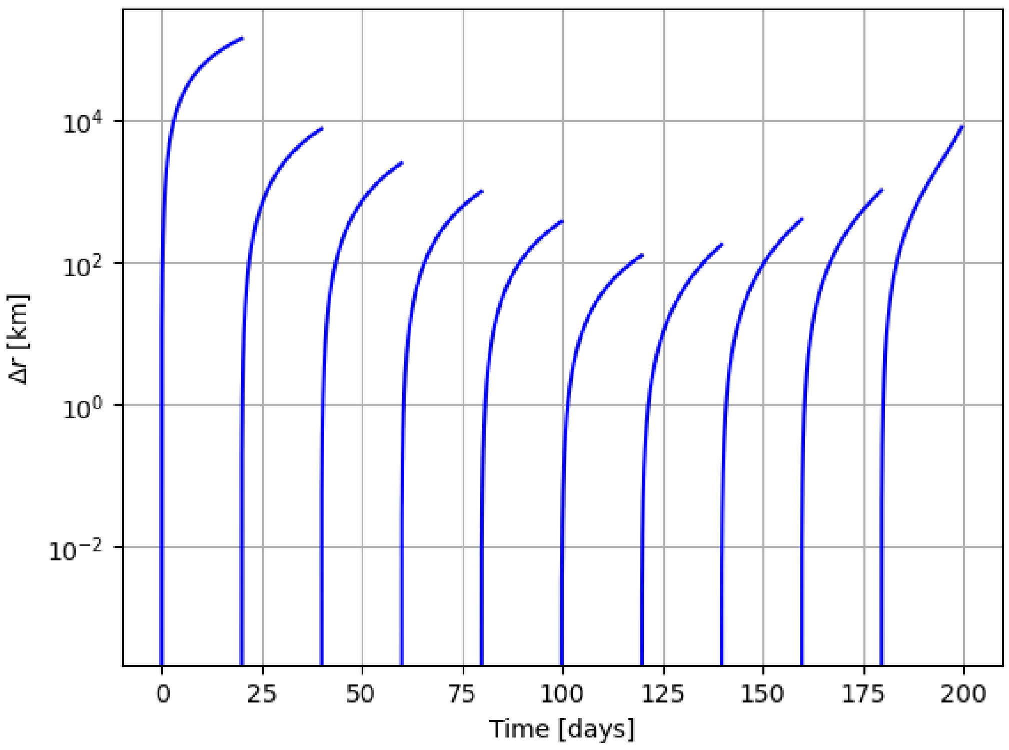

Figure 3.1: $\Delta r = ||\Delta \mathbf{r}||$ as a function of time without any corrections. The end of each arc is clearly seen from the non-zero end of the cane curves.

3.2 Correction Algorithm Formulation

From Eq.(3.1), with $\mathbf{p} = 0$ as only the initial state can be changed through the velocity, the relation is considered for an arc i, expanded and simplified in Eq. (3.2). The initial position cannot be changed to obtain as a continuous trajectory in position is desired (vertical strike-through lines), and the velocity at the end of the arc is free of constraints (horizontal strike-through lines). This yields Eq. (3.3), which can be solved by computing $\boldsymbol{\Phi}_i$ and determining the necessary change in state $-\Delta\tilde{\mathbf{x}}(t_{i+1}) = -(\mathbf{x}(t_{i+1}) - \mathbf{x}_{lamb}(t_{i+1}))$ for each arc. Finally, this $\Delta\vec{V}$ is applied at the start of the arc, yielding a decrease in Δr at the end of the arc compared to Fig. 3.1.

$$\Delta\tilde{\mathbf{x}}(t) = \boldsymbol{\Phi}(t, t_0)\Delta\mathbf{x}_0 + \mathbf{S}(t)\mathbf{p} = \boldsymbol{\Phi}(t, t_0)\Delta\mathbf{x}_0 \tag{3.1}$$

$$-\Delta\tilde{\mathbf{x}}(t_{i+1}) = \mathbf{\Phi}_i(t_{i+1}, t_{0_i})\Delta\mathbf{x}_{0_i} \Leftrightarrow -\underbrace{\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\\ \cancel{\Delta\dot{x}}\\ \cancel{\Delta\dot{y}}\\ \cancel{\Delta\dot{z}}\end{bmatrix}}_{\Delta\tilde{\mathbf{x}}(t_{i+1})} = \underbrace{\begin{bmatrix}\Phi_{xx} & \Phi_{xy} & \Phi_{xz} & \Phi_{x\dot{x}} & \Phi_{x\dot{y}} & \Phi_{x\dot{z}}\\ \Phi_{yx} & \Phi_{yy} & \Phi_{yz} & \Phi_{y\dot{x}} & \Phi_{y\dot{y}} & \Phi_{y\dot{z}}\\ \Phi_{zx} & \Phi_{zy} & \Phi_{zz} & \Phi_{z\dot{x}} & \Phi_{z\dot{y}} & \Phi_{z\dot{z}}\\ \cancel{\Phi_{\dot{x}x}} & \cancel{\Phi_{\dot{x}y}} & \cancel{\Phi_{\dot{x}z}} & \cancel{\Phi_{\dot{x}\dot{x}}} & \cancel{\Phi_{\dot{x}\dot{y}}} & \cancel{\Phi_{\dot{x}\dot{z}}}\\ \cancel{\Phi_{\dot{y}x}} & \cancel{\Phi_{\dot{y}y}} & \cancel{\Phi_{\dot{y}z}} & \cancel{\Phi_{\dot{y}\dot{x}}} & \cancel{\Phi_{\dot{y}\dot{y}}} & \cancel{\Phi_{\dot{y}\dot{z}}}\\ \cancel{\Phi_{\dot{z}x}} & \cancel{\Phi_{\dot{z}y}} & \cancel{\Phi_{\dot{z}z}} & \cancel{\Phi_{\dot{z}\dot{x}}} & \cancel{\Phi_{\dot{z}\dot{y}}} & \cancel{\Phi_{\dot{z}\dot{z}}}\end{bmatrix}}_{\mathbf{\Phi}_i(t_{i+1},t_{0_i})} \underbrace{\begin{bmatrix}\Delta x_0 = 0\\ \Delta y_0 = 0\\ \Delta z_0 = 0\\ \Delta\dot{x}_0\\ \Delta\dot{y}_0\\ \Delta\dot{z}_0\end{bmatrix}}_{\Delta\mathbf{x}_{0_i}} \tag{3.2}$$

$$\Delta\vec{V} = \begin{bmatrix}\Delta\dot{x}_0\\ \Delta\dot{y}_0\\ \Delta\dot{z}_0\end{bmatrix} = \begin{bmatrix}\Phi_{x\dot{x}} & \Phi_{x\dot{y}} & \Phi_{x\dot{z}}\\ \Phi_{y\dot{x}} & \Phi_{y\dot{y}} & \Phi_{y\dot{z}}\\ \Phi_{z\dot{x}} & \Phi_{z\dot{y}} & \Phi_{z\dot{z}}\end{bmatrix}^{-1} \begin{bmatrix}-\Delta x\\ -\Delta y\\ -\Delta z\end{bmatrix} \Leftrightarrow \Delta\vec{V} = \mathbf{\Phi}_{i,rv}(t_{i+1}, t_{0_i})^{-1}(-\Delta\vec{r}) \tag{3.3}$$

3.3 Propagation with single trajectory correction

The linearised model is helpful (applicable) in this case, as the use of the algorithm described above permitted to reduce $||\Delta\vec{r}(t_{i+1})||$ by several orders of magnitude for each arc. The non-linearity is the largest in the 1st arc, where the correction has the least impact, although still reducing Δr significantly (from 10000km to 38km).

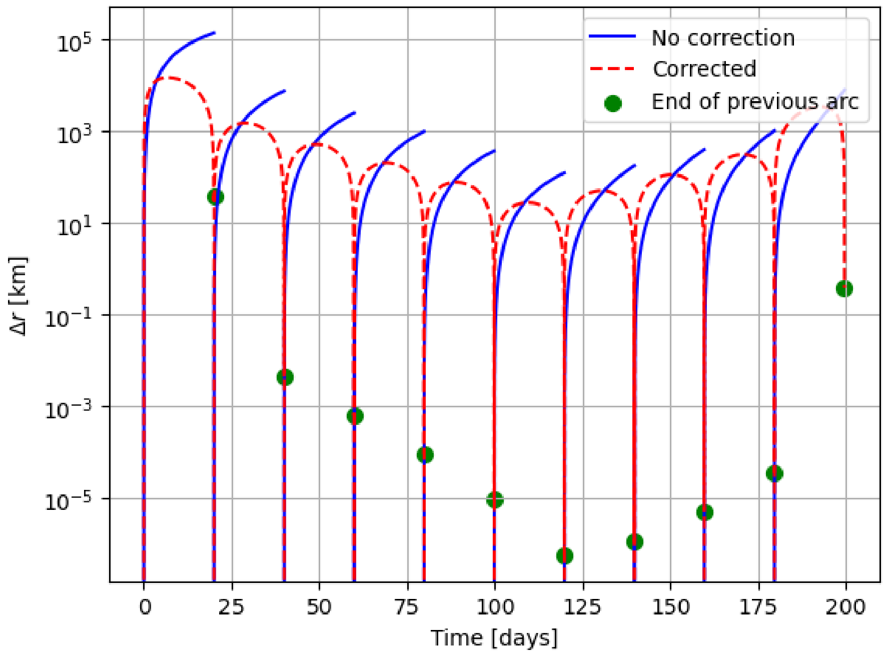

Figure 3.2: $\Delta r = ||\Delta\mathbf{r}||$ as a function of time with correction according to the model presented.

3.4 Total ΔV

Table 3.1: Break-down of the delta-V of the complete trajectory. Note that in addition to the delta-V computed from the model above directly, the deviation in velocity at the end epoch of each arc with respect to the lambert trajectory is taken into account.

Arc Number	ΔV [m/s]
0	80.501
1	24.754
2	4.251
3	1.582
4	0.617
5	0.213
6	0.154
7	0.362
8	0.890
9	5.484
Total	118.80853

Algorithm 1 Determining total ΔV

```
1: procedure TRAJECTORY CORRECTION
2: Initialise variables → $\Delta V_{total} = 0$, $\Delta\vec{V}_{dev,-1} = \vec{0}$
3: loop: for arc number i=0 → 9
4:     Solve variational equations → $\mathbf{\Phi}_i(t, t_i), \mathbf{x}_i(t), \mathbf{x}_{lamb,i}(t)$
5:     Extract lambert and numerical position at end epoch from states → $\mathbf{r}_i(t_{i+1}), \mathbf{r}_{lamb,i}(t_{i+1})$
6:     Obtain position deviation at arc end epoch → $\Delta\tilde{\mathbf{r}}(t_{i+1}) = \mathbf{r}_i(t_{i+1}) - \mathbf{r}_{lamb,i}(t_{i+1})$
7:     Compute trajectory correction → $\Delta\vec{V}_i = \mathbf{\Phi}_{i,rv}(t_{i+1}, t_i)^{-1}(-\Delta\tilde{\mathbf{r}}(t_{i+1}))$
8:     Add to total delta-V → $\Delta V_{total} = \Delta V_{total} + ||\Delta\vec{V}_i + \Delta\vec{V}_{dev,i-1}||$
9:     Propagate arc with trajectory correction → $\mathbf{x}_{corr_i}(t), \mathbf{x}_{corr_{lamb,i}}(t)$
10:    Compute velocity deviations at the end of arc epoch → $\Delta\vec{V}_{dev,i} = (\vec{V}_{corr_{lamb,i}}(t_{i+1}) - \vec{V}_{corr_i}(t_{i+1}))$
11: close loop;
12: print($\Delta V_{total}$)
```

3.5 Propagation with iterated trajectory corrections

Table 3.2: Break-down of the delta-V of the complete trajectory with iterations on the corrections. The difference with the previous section is smaller than 3 decimals for most arcs.

Arc Number	ΔV [m/s]
0	80.524
1	24.732
2	4.251
3	1.582
4	0.617
5	0.213
6	0.154
7	0.362
8	0.890
9	5.485
Total	118.80876

Table 3.3: Number of iterations per arc. Note that 1 iteration means that the result from the previous section already met the required 1m accuracy, and no additional iteration was performed.

Arc Number	**Number of Iterations**
0	2
1	2
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	2

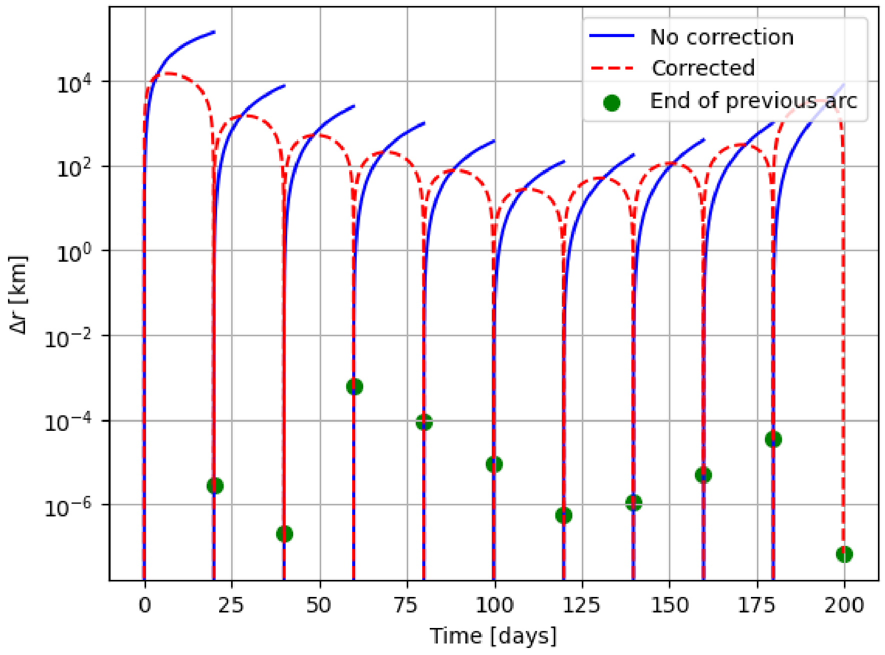

Figure 3.3: Δr as a function of time with corrections obtained by iterating on the model.

Problem 4

4.1 Single Arc Low Thrust Model

$$\Delta\tilde{\mathbf{x}}(t) = \mathbf{\Phi}(t,t_0)\Delta\mathbf{x}_0 + \mathbf{S}(t)\mathbf{p} = \mathbf{S}(t)\mathbf{p} \qquad\qquad \Delta\mathbf{x}_0 = 0$$

$-\Delta\tilde{\mathbf{x}}(t_E) = -(\mathbf{x}(t_E) - \mathbf{x}_{lamb}(t_E))$

$$-\Delta\tilde{\mathbf{x}}(t_E) = \mathbf{S}(t_E)\mathbf{p} \Leftrightarrow -\underbrace{\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\\ \cancel{\Delta\dot{x}}\\ \cancel{\Delta\dot{y}}\\ \cancel{\Delta\dot{z}}\end{bmatrix}}_{\Delta\tilde{\mathbf{x}}(t_E)} = \underbrace{\begin{bmatrix}S_{xR} & S_{xS} & S_{xW}\\ S_{yR} & S_{yS} & S_{yW}\\ S_{zR} & S_{zS} & S_{zW}\\ \cancel{S_{\dot{x}R}} & \cancel{S_{\dot{x}S}} & \cancel{S_{\dot{x}W}}\\ \cancel{S_{\dot{y}R}} & \cancel{S_{\dot{y}S}} & \cancel{S_{\dot{y}W}}\\ \cancel{S_{\dot{z}R}} & \cancel{S_{\dot{z}S}} & \cancel{S_{\dot{z}W}}\end{bmatrix}}_{\mathbf{S}(t_E)} \underbrace{\begin{bmatrix}p_R\\ p_S\\ p_W\end{bmatrix}}_{\mathbf{p}} \tag{4.1}$$

$$\Leftrightarrow \begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix} = \begin{bmatrix}S_{xR} & S_{xS} & S_{xW}\\ S_{yR} & S_{yS} & S_{yW}\\ S_{zR} & S_{zS} & S_{zW}\end{bmatrix}^{-1} \begin{bmatrix}-\Delta x\\ -\Delta y\\ -\Delta z\end{bmatrix} \Leftrightarrow \mathbf{p} = \left(\left(\frac{\partial \mathbf{r}}{\partial \mathbf{p}}\right)(t_E)\right)^{-1}(-\Delta\vec{r}) \tag{4.2}$$

4.2 Single Arc Low Thrust Results

The model shown above was iterated upon to obtain Figures 4.1 and 4.2, which show that, due to the (RSW constant) thrust, the position and velocity are continuous throughout the trajectory, $\bar{\mathbf{r}}(t_E)$ is reached, but $\bar{\mathbf{v}}(t_E)$ is not matched (as expected from a single arc model). The model therefore works as expected !

$$\mathbf{p} = 2.30329288\cdot 10^{-5}\hat{R} + 2.63750516\cdot 10^{-6}\hat{S} + 1.80984874\cdot 10^{-6}\hat{W}\ \text{m/s/s} \tag{4.3}$$

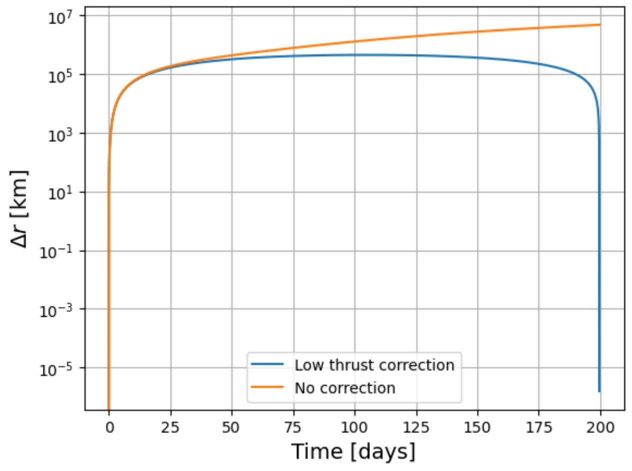

Figure 4.1: Cartesian distance $\Delta r = ||\Delta\mathbf{r}||$ between the Lambert and the numerical trajectories as function of time, with a single arc low thrust correction. Iterations were performed to meet a tolerance of 1 m.

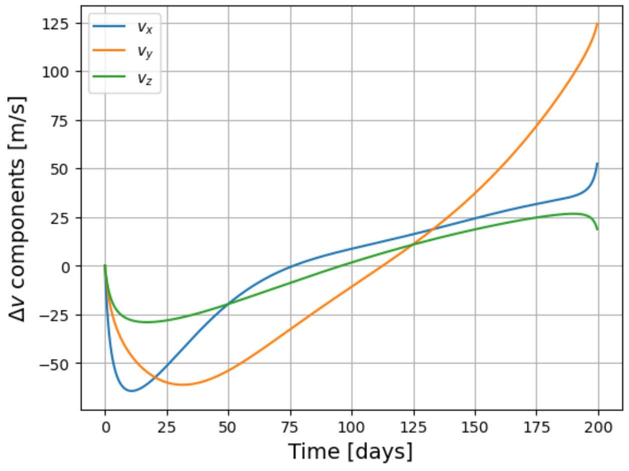

Figure 4.2: Difference in Cartesian velocity components between the numerical and Lambert trajectories, as function of time, with a single arc low thrust correction. Iterations were performed to meet a tolerance of 1 m in position.

4.3 Double Arc Low Thrust Model

Using two arcs, two separate propagations are performed, enforcing continuity in position and velocity at their junction. The deviation at the arrival epoch is given by Eq. (4.4), with $\Delta\mathbf{x}(t_{mid})$ given by Eq. (4.5). Note that all matrices are obtained from a propagation with initial position on the lambert arc.

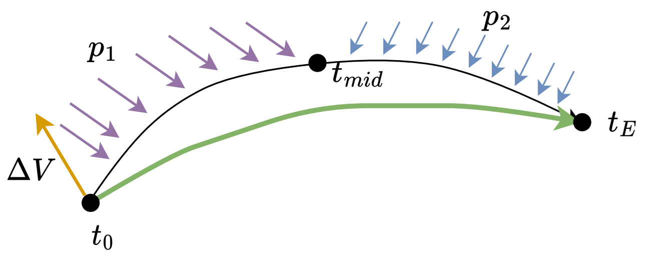

Figure 4.3: Diagram defining important times and variables. Note $\Delta V = \Delta v_i(t_0)$, and the green curve describes the Lambert arc.

$$\Delta\tilde{\mathbf{x}}(t_E) = \mathbf{\Phi}_2(t_E, t_{mid})\Delta\tilde{\mathbf{x}}(t_{mid}) + \mathbf{S}_2(t_E)\mathbf{p}_2 \tag{4.4}$$

$$\Delta\tilde{\mathbf{x}}(t_{mid}) = \mathbf{\Phi}_1(t_{mid}, t_0)\Delta\mathbf{x}(t_0) + \mathbf{S}_1(t_{mid})\mathbf{p}_1 \tag{4.5}$$

$$\begin{aligned}\Leftrightarrow \Delta\tilde{\mathbf{x}}(t_E) &= \mathbf{\Phi}_2(t_E, t_{mid})(\mathbf{\Phi}_1(t_{mid}, t_0)\Delta\mathbf{x}(t_0) + \mathbf{S}_1(t_{mid})\mathbf{p}_1) + \mathbf{S}_2(t_E)\mathbf{p}_2 \\ &= \mathbf{\Phi}_2(t_E, t_{mid})\mathbf{S}_1(t_{mid})\mathbf{p}_1 + \mathbf{S}_2(t_E)\mathbf{p}_2 + \mathbf{\Phi}_2(t_E, t_{mid})\mathbf{\Phi}_1(t_{mid}, t_0)\Delta\mathbf{x}(t_0)\end{aligned}$$

$$\Leftrightarrow \Delta\tilde{\mathbf{x}}(t_E) = \begin{bmatrix} \underbrace{\mathbf{\Phi}_2(t_E, t_{mid})\mathbf{S}_1(t_{mid})}_{=\mathbf{B}(\text{size } 6\times3)} & \underbrace{\mathbf{S}_2(t_E)}_{=\mathbf{C}(\text{size } 6\times3)} & \underbrace{\mathbf{\Phi}_2(t_E, t_{mid})\mathbf{\Phi}_1(t_{mid}, t_0)}_{=\mathbf{D}(\text{size } 6\times6)} \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \Delta\mathbf{x}(t_0) \end{bmatrix} \tag{4.6}$$

Defining $\mathbf{B} = \mathbf{\Phi}_2(t_E, t_{mid})\mathbf{S}_1(t_{mid})$, $\mathbf{C} = \mathbf{S}_2(t_E)$, and $\mathbf{D} = \mathbf{\Phi}_2(t_E, t_{mid})\mathbf{\Phi}_1(t_{mid}, t_0)$ and expanding,

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \cancel{\Delta\dot{x}} \\ \cancel{\Delta\dot{y}} \\ \cancel{\Delta\dot{z}} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & B_{13} & C_{11} & C_{12} & C_{13} & \cancel{D_{11}} & \cancel{D_{12}} & \cancel{D_{13}} & D_{14} & D_{15} & D_{16} \\ B_{21} & B_{22} & B_{23} & C_{21} & C_{22} & C_{23} & \cancel{D_{21}} & \cancel{D_{22}} & \cancel{D_{23}} & D_{24} & D_{25} & D_{26} \\ B_{31} & B_{32} & B_{33} & C_{31} & C_{32} & C_{33} & \cancel{D_{31}} & \cancel{D_{32}} & \cancel{D_{33}} & D_{34} & D_{35} & D_{36} \\ \cancel{B_{41}} & \cancel{B_{42}} & \cancel{B_{43}} & \cancel{C_{41}} & \cancel{C_{42}} & \cancel{C_{43}} & \cancel{D_{41}} & \cancel{D_{42}} & \cancel{D_{43}} & \cancel{D_{44}} & \cancel{D_{45}} & \cancel{D_{46}} \\ \cancel{B_{51}} & \cancel{B_{52}} & \cancel{B_{53}} & \cancel{C_{51}} & \cancel{C_{52}} & \cancel{C_{53}} & \cancel{D_{51}} & \cancel{D_{52}} & \cancel{D_{53}} & \cancel{D_{54}} & \cancel{D_{55}} & \cancel{D_{56}} \\ \cancel{B_{61}} & \cancel{B_{62}} & \cancel{B_{63}} & \cancel{C_{61}} & \cancel{C_{62}} & \cancel{C_{63}} & \cancel{D_{61}} & \cancel{D_{62}} & \cancel{D_{63}} & \cancel{D_{64}} & \cancel{D_{65}} & \cancel{D_{66}} \end{bmatrix} \begin{bmatrix} p_{1,x} \\ p_{1,y} \\ p_{1,z} \\ p_{2,x} \\ p_{2,y} \\ p_{2,z} \\ \Delta x_0 = 0 \\ \Delta y_0 = 0 \\ \Delta z_0 = 0 \\ \Delta\dot{x}_0 \\ \Delta\dot{y}_0 \\ \Delta\dot{z}_0 \end{bmatrix}$$

$$\Leftrightarrow \Delta\mathbf{r}(t_{E,2}) = \Delta\mathbf{r}(t_E) = \underbrace{\begin{bmatrix} B_{11} & B_{12} & B_{13} & C_{11} & C_{12} & C_{13} & D_{14} & D_{15} & D_{16} \\ B_{21} & B_{22} & B_{23} & C_{21} & C_{22} & C_{23} & D_{24} & D_{25} & D_{26} \\ B_{31} & B_{32} & B_{33} & C_{31} & C_{32} & C_{33} & D_{34} & D_{35} & D_{36} \end{bmatrix}}_{=\mathbf{A}} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \Delta\mathbf{v}_i(t_0) \end{bmatrix} \tag{4.7}$$

4.4 Double Arc Low Thrust Results

$$\Delta \mathbf{v}_i(t_0) = \vec{0} \qquad \mathbf{p}_1 = \mathbf{p}_{\text{previous section}} \qquad \Delta \mathbf{r}(t_E) = -(\mathbf{r}(t_E) - \mathbf{r}_{lamb}(t_E))$$

$$\mathbf{p}_2 = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix}^{-1} \left(\Delta \mathbf{r}(t_E) - \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} \mathbf{p}_1 \right) \tag{4.8}$$

Realising that the two matrices in Eq. (4.8), are the first three rows of $\mathbf{C} = \mathbf{S}_2(t_E)$ and $\mathbf{B} = \mathbf{\Phi}_2(t_E, t_{mid})\mathbf{S}_1(t_{mid})$ (subscript r indicating that only the first three rows considered), and that $\mathbf{S}_1(t_{mid})\mathbf{p}_1$ is an approximate for the state deviation at the end of the first arc (which can be computed as $\Delta\tilde{\mathbf{x}}(t_{mid})$). Furthermore, $(\mathbf{\Phi}_2(t_E, t_{mid})\Delta\tilde{\mathbf{x}}(t_{mid}))_r$ gives the variation in position at the end of the 2nd arc due to the variation in state at its start.

$$\mathbf{p}_2 = \mathbf{S}_{2,r}(t_E)^{-1} \left[\Delta \mathbf{r}(t_E) - (\mathbf{\Phi}_2(t_E, t_{mid})\mathbf{S}_1(t_{mid})\mathbf{p}_1)_r\right]$$
$$\Leftrightarrow \mathbf{p}_2 = \mathbf{S}_{2,r}(t_E)^{-1} \left[\Delta \mathbf{r}(t_E) - (\mathbf{\Phi}_2(t_E, t_{mid})\Delta\tilde{\mathbf{x}}(t_{mid}))_r\right]$$

The model can be further simplified by instead solving the variational equations for the 2nd arc from the (known) end state of the first arc, then the effect of $(\mathbf{\Phi}_2(t_E, t_{mid})\Delta\tilde{\mathbf{x}}(t_{mid}))_r$ is naturally present in the model, but $\mathbf{S}_{2,r}$ is different (becomes $\mathbf{S}_{2,r_n}$). This model is iterated upon to obtain Figures 4.4 and 4.5

$$\mathbf{p}_2 = \mathbf{S}_{2,r_n}(t_E)^{-1}\Delta\tilde{\mathbf{r}}(t_E) \tag{4.9}$$

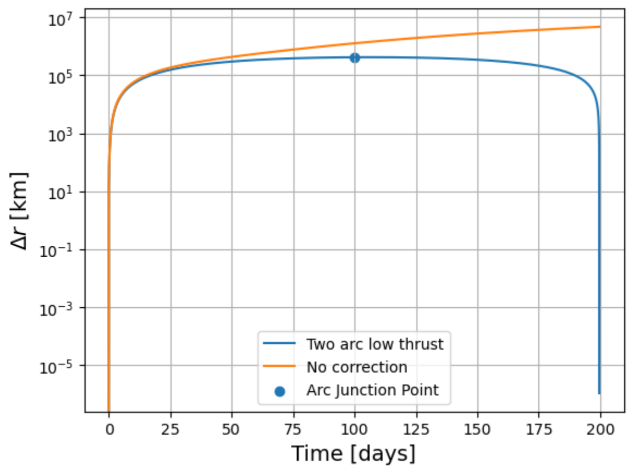

Figure 4.4: Cartesian distance $\Delta r = ||\Delta \mathbf{r}||$ between the Lambert and the numerical trajectories as function of time, with a single arc low thrust correction. Iterations were performed to meet a tolerance of 1 m.

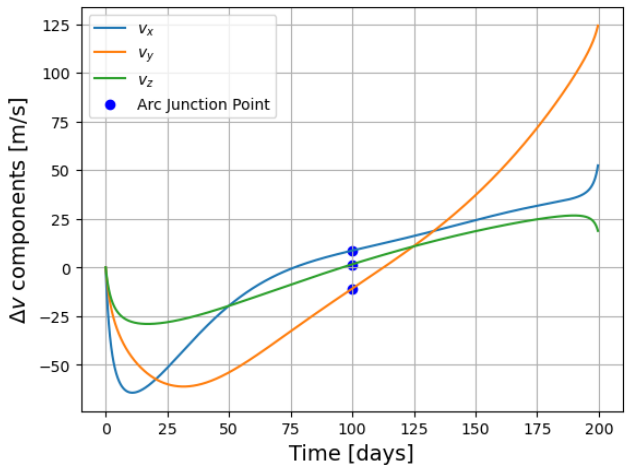

Figure 4.5: Difference in Cartesian velocity components between the numerical and Lambert trajectories, as function of time, with a two arc low thrust correction. Iterations were performed to meet a tolerance of 1 m in position.

4.5 Discussion

The model results in a final position close to the final Lambert position, but using two arcs did not result in the design freedom expected. The thrust in the 1st arc was fixed to a value determined to reach the target position over the full trajectory, yielding a start state of the 2nd arc which can only reach the target position if the same thrust is applied. A better solution is obtained by solving Eq. (4.6) for both $\mathbf{p}_1$ and $\mathbf{p}_2$ with $\Delta\mathbf{x}(t_0) = 0$ and specifying that the both the position and velocity match with the Lambert model at t_E. This has the added advantage of reaching Mars with the Lambert velocity, removing the need for a larger ΔV at arrival. See Fig.4.6 and 4.7

Q4d: $\mathbf{p}_1 = \mathbf{p}_2 = 2.30329288 \cdot 10^{-5}\hat{R} + 2.63750516 \cdot 10^{-6}\hat{S} + 1.80984874 \cdot 10^{-6}\hat{W}$ m/s/s

Q4e:

$$\mathbf{p}_1 = 2.34307729 \cdot 10^{-5}\hat{R} + 9.75540651 \cdot 10^{-6}\hat{S} + 2.92456749 \cdot 10^{-6}\hat{W} \text{ m/s/s}$$
$$\mathbf{p}_2 = -8.15156999 \cdot 10^{-5}\hat{R} + 3.05592342 \cdot 10^{-6}\hat{S} - 1.66032577 \cdot 10^{-7}\hat{W} \text{ m/s/s}$$

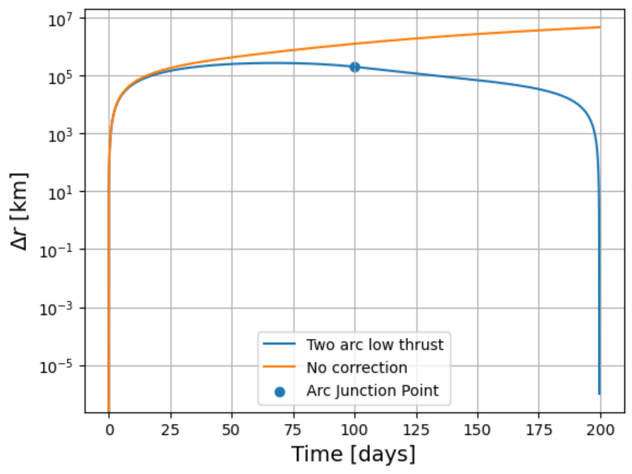

Figure 4.6: Cartesian distance $\Delta r = ||\Delta \mathbf{r}||$ between the Lambert and the numerical trajectories as function of time, with a two arc low thrust correction with a variable $\mathbf{p}_1$. Iterations were performed to meet a tolerance of 1 m.

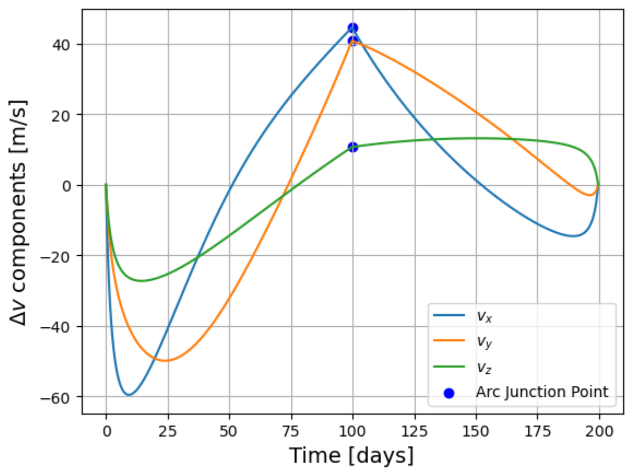

Figure 4.7: Difference in Cartesian velocity components between the numerical and Lambert trajectories, as function of time, with a two arc low thrust correction with a variable $\mathbf{p}_1$. Iterations were performed to meet a tolerance of 1 m in position.

Problem 5

5.1 Maximum Cartesian Correction

The algorithm aims to find the smallest correction value, per entry, which minimises one of the objective functions from Eq. (5.1) (5.2). With $\boldsymbol{\epsilon}_{r,max}$ and $\boldsymbol{\epsilon}_{v,max}$ from (with a perturbation p),

$$G_r = ||\boldsymbol{\epsilon}_{r,max}|| - 100000 \text{ m} \quad (5.1) \qquad G_v = ||\boldsymbol{\epsilon}_{v,max}|| - 1 \text{ m/s} \quad (5.2)$$

$$\Delta\mathbf{x}_0(j) = \begin{cases} p & \text{if } j = \text{current entry} \\ 0 & \text{otherwise} \end{cases} \quad (5.3)$$

$$\Delta\tilde{\mathbf{x}}(t) = \mathbf{S}(t)\Delta\mathbf{x}_0 \quad (5.4) \qquad \Delta\mathbf{x}(t) = \mathbf{x}_{\Delta\mathbf{x}_0 > \vec{0}}(t) - \mathbf{x}_{\Delta\mathbf{x}_0 = \vec{0}}(t) \quad (5.5)$$

$$\boldsymbol{\epsilon}_x(t) = \Delta\tilde{\mathbf{x}}(t) - \Delta\mathbf{x} \quad (5.6)$$

$$||\boldsymbol{\epsilon}_{r,max}|| = \max_t ||\boldsymbol{\epsilon}_r(t)|| \quad (5.7) \qquad ||\boldsymbol{\epsilon}_{v,max}|| = \max_t ||\boldsymbol{\epsilon}_v(t)|| \quad (5.8)$$

For each entry of each arc, the same optimisation procedure is followed twice, once for each objective function from Eq. (5.1) and (5.2), and the minimum result of the two is selected. The algorithm goes as follows.

1. For each arc, the non-corrected state and transition matrix histories, and initial state, are retrieved.
2. For each entry, an initial guess p is defined, yielding initial values of G_r and G_v prior to optimisation.
3. For each objective function $G = G_{r,v}$, an optimisation similar to a 1D version of the steepest descent is used (Eq. (5.9), $\alpha > 0$). In 1D, the sign of G permits to choose the direction of the iteration fully: if G is negative, a larger perturbation is necessary, hence p is increased (and inversely). The value of α is chosen such that p does not become negative and that G does not change sign (to improve convergence). This is done by starting with a large value of $\alpha = 10^9$ and halving it until the sign of $G(p_{new} - \alpha \cdot \text{sign}(G))$ is the same as the sign of G (and $p_{new} > 0$). Once the value of α fits both requirements, p_{new} is computed and the termination conditions are evaluated. The optimisation is stopped when $\frac{\alpha}{p} < ToL$ ($ToL < 1$).

$$p_{new} = p - \alpha \cdot \text{sign}(G) \quad (5.9)$$

The implementation runs within about 2.5 minutes, which is reasonable. The iteration stop condition can be set to $ToL < 25\%$ (eg. 10%), which ensures that the value obtained is within 25% of the true one. Then, p is changed by less than 25% than its previous value and is guaranteed to change by even less in the next iterations (due to the condition to not change the sign of G in finding α).

Table 5.1: Maximum for each entry Cartesian for each arc, each entry was computed individually. The tolerance was set to $ToL = 10\%$ to be on the safer side.

Arc i	Δx [km]	Δy [km]	Δz [km]	$\Delta \dot{x}$ [m/s]	$\Delta \dot{y}$ [m/s]	$\Delta \dot{z}$ [m/s]
0	24414.0725	32226.5725	32226.5725	147.84	196.26	173.91
1	343750.01	578125.01	406250.01	494.29	740.16	635.85
2	343750.01	503906.26	468750.01	501.74	732.71	665.65
3	390625.01	564453.135	531250.01	576.24	784.86	784.86
4	468750.01	656250.01	593750.01	665.65	971.12	904.07
5	593750.01	687500.01	687500.01	904.07	993.48	993.48
6	765625.01	687500.01	765625.01	1142.49	971.12	1112.69
7	890625.01	656250.01	812500.01	1261.70	904.07	1216.99
8	812500.01	632812.51	882812.51	1112.66	874.27	1231.90
9	117187.51	144531.26	164062.51	77.17	95.68	114.31

Algorithm 2 Compute $\boldsymbol{\epsilon}_x$

```
1: procedure EPS_X(original_initial_state, bodies, propagator_settings, state_transition_result, $\mathbf{x}(t)$, $\Delta\mathbf{x}_0$)
2:     # Obtain numerical difference
3:     Compute corrected initial state $\rightarrow \mathbf{x}_0$
4:     Propagate corrected state $\rightarrow \mathbf{x}_{corr}(t)$
5:     Compute deviation from uncorrected propagation ($\mathbf{x}(t)$) $\rightarrow \Delta\mathbf{x}(t) = \mathbf{x}_{corr}(t) - \mathbf{x}(t)$
6:     # Obtain linear difference
7:     Define linear deviation matrix $\rightarrow M$
8:     for $t_i$ in epochs do
9:         Obtain transition matrix at epoch $\rightarrow \Phi(t_i)$
10:        Compute linear deviation at epoch $\rightarrow \Delta\tilde{\mathbf{x}}(t_i)$
11:        Store linear deviation at epoch in linear deviation matrix, $M$
12:    $\Delta\tilde{\mathbf{x}}(t) = M$
13:    # Compute epsilon
14:    Compute difference between linear and numerical deviations $\rightarrow \boldsymbol{\epsilon}_x(t) = \Delta\tilde{\mathbf{x}}(t) - \Delta\mathbf{x}(t)$
15:    Distinguish $\boldsymbol{\epsilon}_r(t)$ (first three entries) and $\boldsymbol{\epsilon}_v(t)$ (last three entries) from $\boldsymbol{\epsilon}_x(t) \rightarrow \boldsymbol{\epsilon}_r(t), \boldsymbol{\epsilon}_v(t)$
16:    Determine $||\max_t \boldsymbol{\epsilon}_r(t)||$ and $||\max_t \boldsymbol{\epsilon}_v(t)|| \rightarrow \boldsymbol{\epsilon}_{r,max}, \boldsymbol{\epsilon}_{v,max}$
17:    return $\boldsymbol{\epsilon}_{r,max}, \boldsymbol{\epsilon}_{v,max}$
```

Algorithm 3 Compute perturbation vector

```
1: procedure PERT_V(perturbation, entry)
2:     Define identity matrix $\rightarrow I$
3:     Multiply $I$ and the perturbation $\rightarrow P = (\text{perturbation})I$
4:     return P[entry, :]
```

Algorithm 4 α step selection - note that this is a snippet of the optimisation code, and not a function. It was separated to make the pseudocode more readable. Note that the α passed to the procedure is from the previous iteration

```
1: procedure GET_ALPHA($p$, $G$, $\alpha$)
2:     $\alpha = 1e9$
3:     Obtain current objective function value $\rightarrow G$
4: while True
5:     Find next perturbation with current value of $\alpha \rightarrow p_{next} = p - \alpha sign(G)$
6:     Determine perturbation vector for $p_{next} \rightarrow v_{p_{next}} = PERT\_V(p_{next}, entry)$
7:     Evaluate $\boldsymbol{\epsilon}_x$ (EPS_X) for $v_{p_{next}} \rightarrow \boldsymbol{\epsilon}_{r,max}, \boldsymbol{\epsilon}_{v,max}$
8:     Evaluate relevant (based on optimisation case) objective function $\rightarrow G_{next}$
9:     if (sign($G$) == sign($G_{next}$)) and $p_{next} > 0$ then
10:        break
11:    else
12:        $\alpha = \alpha/2$
```

Algorithm 5 Optimisation of objective function - note that the tolerance on the objective function is set as much more constraining than the method explained above, it is only a safety net in case the main stop condition was to fail because of an error in the computation of α (eg.)

```
procedure OPTIMISATION ALGORITHM
    $r\_th = 100000$; $v\_th = 1$
    $ToL = 1e - 4$
    Define $M$, an array to store the perturbations at each entry
    for arc_index: i=0 $\rightarrow$ 9 do
        Determine arc start and end times $\rightarrow t_{0,i}, t_{E,i}$
        Solve variational equations $\rightarrow \mathbf{\Phi}_i(t), \mathbf{x}_i(t)$
        Obtain original initial condition $\rightarrow \mathbf{x}_0$
        for entry=0 $\rightarrow$ 5 do
            Define start guess perturbation $\rightarrow p$
            Compute perturbation vector $\rightarrow v_p$
            Compute $\boldsymbol{\epsilon}_{r,max}$ and $\boldsymbol{\epsilon}_{v,max}$ from EPS_X and $p \rightarrow \boldsymbol{\epsilon}_{r,max}, \boldsymbol{\epsilon}_{v,max}$
            Compute objective functions for both $r$ and $v \rightarrow G_r, G_v$
            for OptimisationCase=0,1 do # Optimisation for $G_r$ or $G_v$
                stop = False
                $\alpha = 1e9$
                if OptimisationCase==0 then
                    $G = G_r$; $th = r\_th$
                else
                    $G = G_v$; $th = v\_th$
                while $G/th > ToL$ & stop == False do
                    Compute $\boldsymbol{\epsilon}_{r,max}$ and $\boldsymbol{\epsilon}_{v,max}$ from EPS_X and current $p$
                    Compute $G_r$ and $G_v$ from $\boldsymbol{\epsilon}_{r,max}$ or $\boldsymbol{\epsilon}_{v,max}$
                    if OptimisationCase==0 then
                        $G = G_r$
                    else
                        $G = G_v$
                    $\alpha$ = GET_ALPHA($p$, $G$, $\alpha$)
                    if $\alpha/p < 0.1$ then
                        stop = True
                    $p = p - \alpha sign(G)$
                    Compute new perturbation vector $\rightarrow v_p$
                    Store perturbation value in $M$ array
                    print($M$)
```

5.2 Norm of Positional Maximum Corrections

Two cases can be distinguished: (1) in interplanetary space, where the accelerations of Earth and Mars are small, the norm increases with distance from the Sun because its gravity field gets weaker. The weaker the gravity field, the weaker the non-linearity in position is (in absence of gravity field, the case becomes linear). (2) close to either Mars or Earth, the gravity field is especially non-linear in position due to the interaction of the body and Sun's gravity, meaning that the norm decreases. This can be particularly seen at the SOI boundary, where slight changes in position make either the Sun or the body dominant in acceleration. Additionally, the norm is larger at Mars than Earth because its gravity field is smaller and the Sun gravity at Mars is larger.

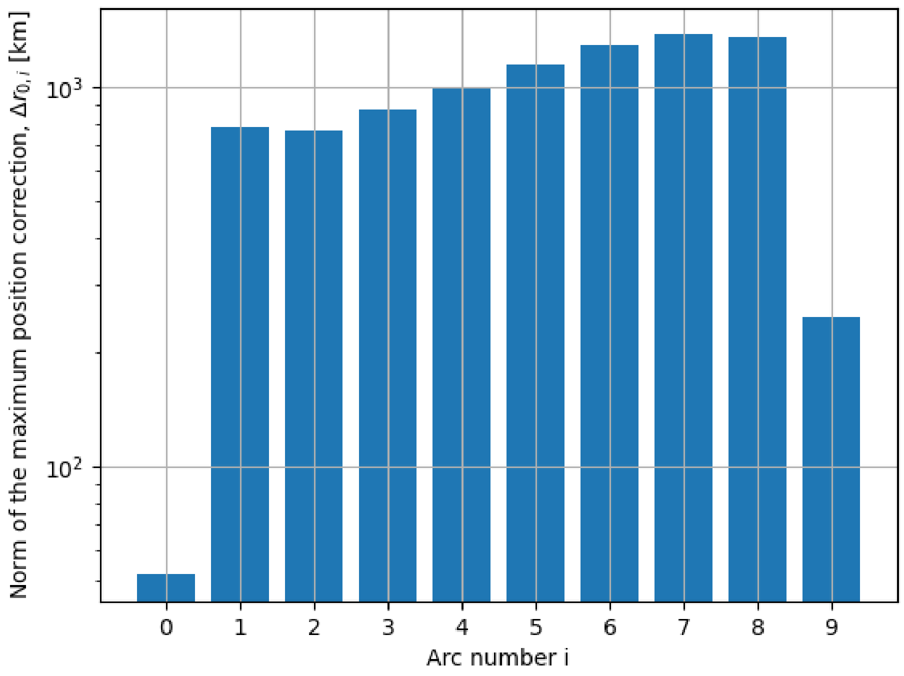

Figure 5.1: Norm of vector of maximum position corrections as a function of the arc number.

5.3 Maximum Corrections in RSW Frame

The only difference with question a) is in the formulation of the perturbation vector. The RSW axes can be determined from the original initial state: $\vec{r}$ and $\vec{V}$, as shown below and can be implemented right after line 8 of the OPTIMISATION ALGORITHM pseudocode. The pseudocode for the perturbation vector is also different,

$$\hat{R} = \frac{\vec{r}}{||\vec{r}||} \quad (5.10) \qquad \hat{W} = \frac{\vec{r} \times \vec{V}}{||\vec{r} \times \vec{V}||} \quad (5.11) \qquad \hat{S} = \frac{\hat{W} \times \hat{R}}{||\hat{W} \times \hat{R}||} \quad (5.12)$$

Algorithm 6 Perturbation vector definition for the RSW corrections

```
1: procedure PERT_V_RSW(perturbation, entry, R, S, W)
2: if entry < 3 # Position entry
3:     Determine the direction vector based on the entry: entry==0: R; entry==1: S; entry==2: W. $\rightarrow$ d
4:     Multiply the direction vector by the perturbation magnitude $\rightarrow \mathbf{v} = (\text{perturbation})\mathbf{d}$
5:     return [v, 0] $\rightarrow 1 \times 6$ sized array
6: else # Velocity entry
7:     Determine the direction vector based on the entry: entry==3: R; entry==4: S; entry==5: W. $\rightarrow$ d
8:     Multiply the direction vector by the perturbation magnitude $\rightarrow \mathbf{v} = (\text{perturbation})\mathbf{d}$
9:     return [0, v] $\rightarrow 1 \times 6$ sized array
```

The norms are only very slightly different from a), and follow the same trends for the same reasons. While the relative magnitude of each component will be different, the norm rests the same as this represents a distance from the original initial position, which is independent of the reference frame.

Table 5.2: Maximum correction for each direction in the RSW frame for both position and velocity, each entry was computed individually. The tolerance was set to $ToL = 10\%$ to be on the safer side.

Arc i	ΔR [km]	ΔS [km]	ΔW [km]	$\Delta \dot{R}$ [m/s]	$\Delta \dot{S}$ [m/s]	$\Delta \dot{W}$ [m/s]
0	31372.08	21484.39	32226.57	196.26	136.66	173.91
1	328125.01	531250.01	406250.01	457.03	732.71	635.85
2	343750.01	503906.26	468750.01	501.74	740.16	665.65
3	382812.51	531250.01	531250.01	576.24	784.86	784.86
4	437988.29	628906.26	593750.01	635.85	904.07	904.07
5	468750.01	687500.01	687500.01	665.65	1023.28	993.48
6	531250.01	765625.01	765625.01	784.86	1112.69	1112.69
7	578125.01	812500.01	812500.01	859.37	1216.99	1216.99
8	593750.01	812500.01	882812.51	846.33	1142.49	1231.89
9	144531.26	148437.51	164062.51	95.68	99.41	112.45

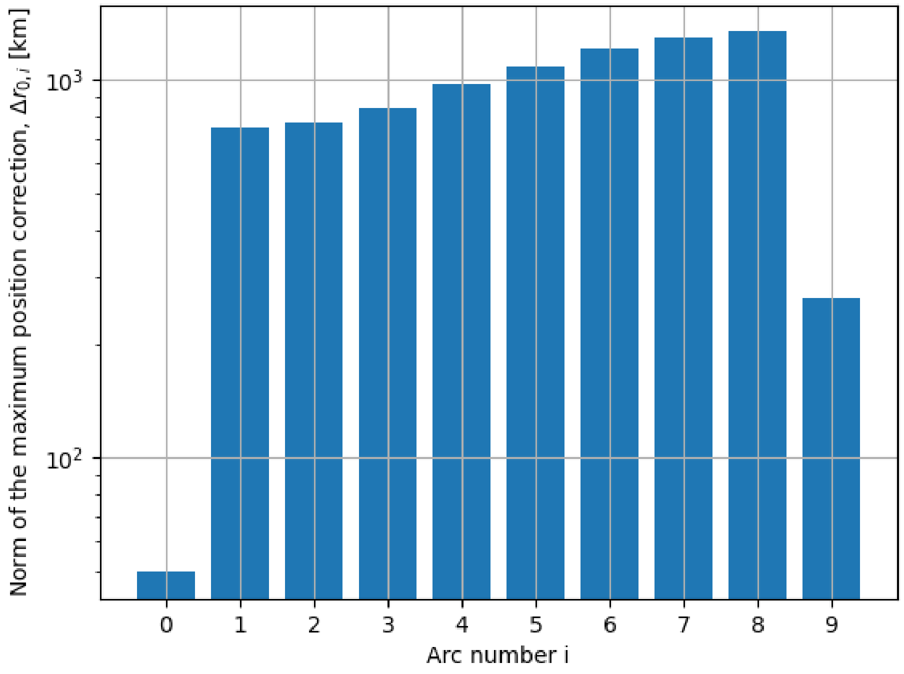

Figure 5.2: Norm of vector of maximum position corrections in the RSW frame as a function of the arc number.