Universität Potsdam
Mathematisch-Naturwisssenschaftliche Fakultät
Institut für Physik und Astronomie

Computational Physics

Star Trek Dreikörperproblem

Autor: Marcus Dahlenburg
Matrikel-Nr.: 761708
Autor: Christian Gößl
Matrikel-Nr.:762627

Korrektor: Prof. Dr. Arkady Pikovsky

24.01.2017

Contents

Part I.
Einleitung

Unendliche Weiten, wir schreiben das Jahr 2017 und unser Raumschiff fliegt seine Bahnen um unser Erde-Mond System. Wir werden in unserer Arbeit das Dreikörperproblem genauer untersuchen. Wir betrachten das Gravitationspotential zweier Himmelskörper wie die Erde und den Mond und lassen einen Probekörper - Raumschiff durch dieses System hindurch fliegen. Wie bereits bekannt ist das Dreikörperproblem nicht allgemein exakt analytisch lösbar. Es gibt viele Möglichkeiten das Problem näherungsweise zu lösen. Hier betrachten wir die zwei Körper Erde-Mond als massereiche Körper und ein Raumschiff, was nur eine verschwindend geringe Masse besitzt, was die Dynamik des Erde-Mond Systems kaum stört. Demnach können wir den Hamiltonoperator und daraus die Bewegungsgleichungen des Raumschiffs angeben. Es ist ein gekoppeltes Differentialgleichzugsystem, wodurch die Trajektorie des Raumschiffs stark von der Wahl der Anfangsbedingungen abhängt. Die Erde und der Mond befinden sich in einer Ebene und rotieren gemeinsam als ein starrer Körper entgegengesetzt dem Uhrzeigersinn. Wir gehen über ins mitrotierende System, wo sich die Erde bei $(-\mu,0)$ und der Mond bei $(1-\mu,0)$ befindet. Demnach befindet sich der Schwerpunkt im Ursprung des Koordinatensystems. In unserer Arbeit werden die folgenden Themen näher untersuchen: Hills Region, Lagrange-Punkte, Symplektisches Intergrationsverfahren und einige interessante Flugbahnen des Raumschiff finden und darstellen.

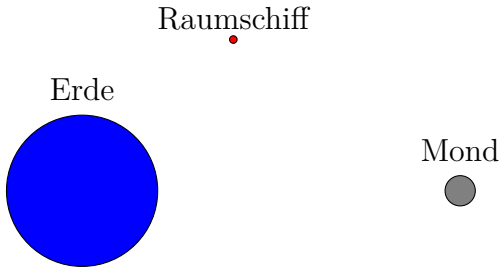

Figure 1: Das Erde-Mond System mit Raumschiff im mitrotierenden System.

Part II.
Grundlagen

Die Dynamik dieses Dreikörperproblems wird von folgenden Hamiltonian beschrieben.

$$H = \frac{(p_x + y)^2}{2} + \frac{(p_y - x)^2}{2} - \Omega(x,y) \tag{1}$$
$$\text{mit } \Omega(x,y) = \frac{x^2 + y^2}{2} + \frac{1-\mu}{r_1} + \frac{\mu}{r_2} + \frac{\mu(1-\mu)}{2}$$
$$\text{wobei } r_1 = \sqrt{(x+\mu)^2 + y^2}, \quad r_2 = \sqrt{(x-1+\mu)^2 + y^2}$$

In Gleichung (1) ist $\mu = m_M$ und $m_E = 1 - \mu$. Die Gesamtmasse des Erd-Mond-Systems ist somit normiert. Die Hamiltonschen Bewegungsgleichungen lauten:

$$\begin{cases} \dot{x} = \frac{dx}{dt} = \frac{\partial H}{\partial p_x} = p_x + y \\ \dot{y} = \frac{dy}{dt} = \frac{\partial H}{\partial p_y} = p_y - x \end{cases} \tag{2}$$

$$\begin{cases} \dot{p}_x = \frac{dp_x}{dt} = -\frac{\partial H}{\partial x} = p_y - x + \Omega_x \\ \dot{p}_y = \frac{dp_y}{dt} = -\frac{\partial H}{\partial y} = -p_x - y + \Omega_y \end{cases} \tag{3}$$

$$\text{mit } \Omega_x = \frac{\partial \Omega}{\partial x}, \quad \Omega_y = \frac{\partial \Omega}{\partial y}$$

Leitet man die Glg. (2) nach t ab und setzt Glg. (3) ein so erhält man:

$$\begin{cases} \ddot{x} = \dot{p}_x + \dot{y} = p_y - x + \Omega_x + \dot{y} \\ \ddot{y} = \dot{p}_y - \dot{x} = -p_x - y + \Omega_y - \dot{x} \end{cases} \tag{4}$$

Stellt man Glg. (2) nach $p = (p_x, p_y)^T$ um und setzt diese Ausdrücke in Glg. (4) ein, erhält man schließlich die grundlegenden Bewegungsgleichungen, die wir für unserer weiteren Untersuchungen numerisch analysieren werden.

$$\begin{cases} \ddot{x} = \dot{y} + x - x + \Omega_x + \dot{y} = 2\dot{y} + \Omega_x \\ \ddot{y} = -\dot{x} + y - y + \Omega_y - \dot{x} = -2\dot{x} + \Omega_y \end{cases} \tag{5}$$

Des Weiteren ist die Energie des Systems E wie folgt definiert.

$$E = \Omega(x,y) - \frac{\dot{x}^2 + \dot{y}^2}{2} \tag{6}$$

Bei näheren Untersuchungen stellt sich heraus, dass die Energie eine Erhaltungsgröße ist.

$$\dot{E} = \dot{\Omega}(x,y) - \ddot{x}\dot{x} - \ddot{y}\dot{y} = \Omega_x\dot{x} + \Omega_y\dot{y} - \ddot{x}\dot{x} - \ddot{y}\dot{y} \tag{7}$$

Aus Gleichung (5) erhält man $\Omega_y = \ddot{y} + 2\dot{x}$ und $\Omega_x = \ddot{x} - 2\dot{y}$. Setzt man dies nun in Glg. (7) ein erkennt man, dass die Energie erhalten ist.

$$\begin{aligned} \dot{E} &= (\ddot{x} - 2\dot{y})\dot{x} + (\ddot{y} + 2\dot{x})\dot{y} - \ddot{x}\dot{x} - \ddot{y}\dot{y} \\ &= \ddot{x}\dot{x} - 2\dot{y}\dot{x} + \ddot{y}\dot{y} + 2\dot{x}\dot{y} - \ddot{x}\dot{x} - \ddot{y}\dot{y} = 0 \end{aligned}$$

Für die numerische Analyse des Differentialgleichungssystem (5) transformieren wir dieses zu einem Differentialgleichungssystem 1. Ordnung mit 4 Variablen:

$$\begin{cases} \dot{z}_1 = z_3 \\ \dot{z}_2 = z_4 \\ \dot{z}_3 = 2z_4 + z_1 - \frac{(1-\mu)\cdot(z_1+\mu)}{r_1^3} - \frac{\mu\cdot(z_1-1+\mu)}{r_2^3} = 2z_4 + \Omega_{z_1}(z_1,z_2) \\ \dot{z}_4 = -2z_3 + z_2 - \frac{(1-\mu)\cdot z_2}{r_1^3} - \frac{\mu\cdot z_2}{r_2^3} = -2z_3 + \Omega_{z_2}(z_1,z_2) \end{cases} \tag{8}$$

$$\text{mit } z_1 = x \text{ und } z_2 = y$$

$$\text{wobei } r_1 = \sqrt{(z_1+\mu)^2 + z_2^2}, \quad r_2 = \sqrt{(z_1-1+\mu)^2 + z_2^2}$$

Um herauszufinden, wo die Lagrange-Punkte $L_1 - L_5$ (Gleichgewichtspunkte, siehe Abb. **2**) liegen, erhalten wir mithilfe des Euler-Verfahren folgende Iterationsvorschrift:

$$\begin{cases} z_1^{n+1} = z_1^n + z_3^n h \\ z_2^{n+1} = z_2^n + z_4^n h \\ z_3^{n+1} = z_3^n + \Omega_{z_1}(z_1^n, z_2^n) h \\ z_4^{n+1} = z_4^n + \Omega_{z_2}(z_1^n, z_2^n) h \end{cases}$$

$$\Leftrightarrow \vec{z}^{\,n+1} = \vec{z}^{\,n} + \vec{f}(z_1^n, z_2^n, z_3^n, z_4^n) \cdot h \tag{9}$$

$$\text{mit } h = t^{n+1} - t^n \text{ und } \vec{z}^{\,n} = (z_1^n, z_2^n, z_3^n, z_4^n)^T$$

Die Gleichgewichtsbedingung für Gleichung (9) lautet $\vec{z}^{\,n+1} = \vec{z}^{\,n}$ und ist erfüllt, wenn $\vec{f}(z_1^n, z_2^n, z_3^n, z_4^n) = \vec{0}$ gilt. Somit sind dort $z_3^n = 0$ und $z_4^n = 0 \; \forall \; n \in \mathbb{Z}^+$. Für L_1, L_2 und L_3 ist $z_2 = 0$ und somit $\Omega_{z_2}(z_1, z_2 = 0) = 0$. Damit vereinfacht sich die Findung dieser Lagrangepukte zur einfachen Bestimmung der Nullstelle von $\Omega_{z_1}(z_1, z_2 = 0) = 0$. Diese bestimmen wir mithilfe des Bisektionsverfahren (Glg. (18), S. 15).

Part III.
Auswertung

1. Lagrangepunkte und Hill's Region

Die Werte für die Koordinaten und die Potentiale an den Lagrangepunken sind in der Tabelle 1 aufgeführt.

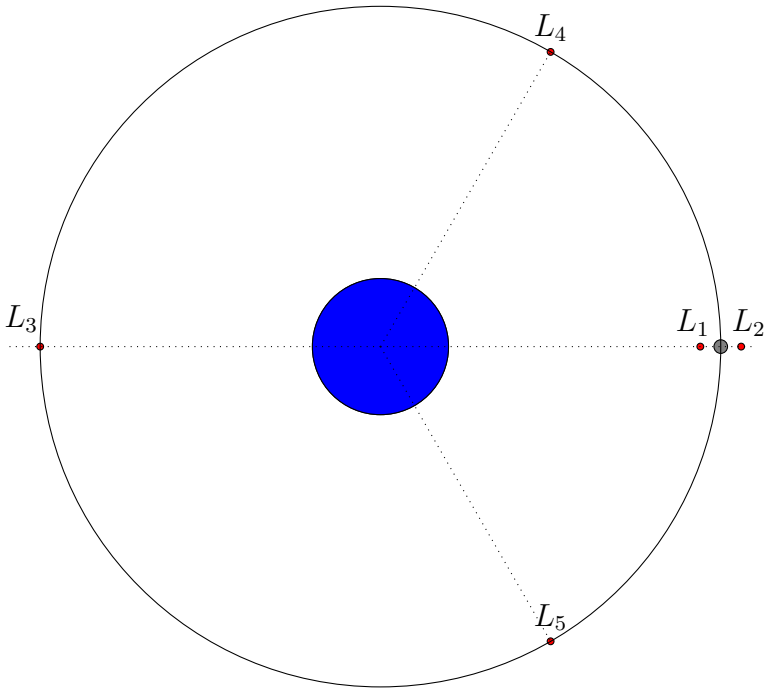

Figure 2: Die Lagrange Punkte im Gravitationsfeld des Erde-Mond Systems.

i	$L_i = (x_i,y_i)$	$\Omega_i = \Omega(x_i,y_i)$
1	(0.716,0)	1.734
2	(1.228,0)	1.701
3	(-1.021,0)	1.549

Table 1: Tabelle mit Koordinaten der Lagrangepunkte L_i (Werte mit einer Genauigkeit von 0.001) und der dort existenten Potentiale $\Omega(L_i)$.

Die Menge aller Punkte (x,y) für die $\Omega(x,y) \geq E$ definiert ein Gebiet namens Hill's Region, welches für ein Raumfahrzeug mit der Energie E erreichbar ist. In der Abbildung **3** sind die Begrenzungen von 6 solcher Regionen dargestellt.

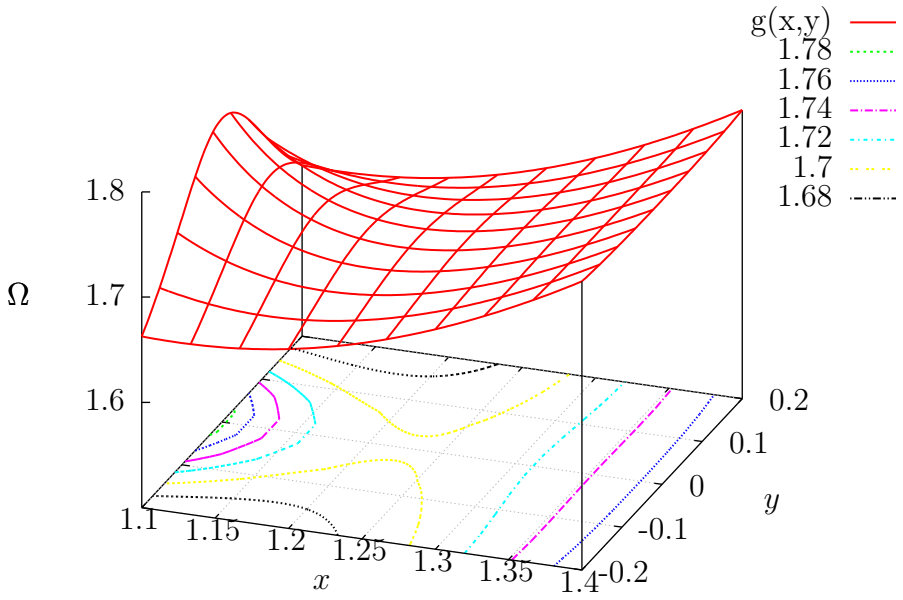

Figure **3**: Potential Ω um L_2 mit 6 Begrenzungen der jeweiligen Hill's Region.

Anhand der Abb. **3** ist ersichtlich, dass sich für Energien $E \leq \Omega_2$ ein „Hals"ausbildet, welcher es einem Körper ermöglicht von außerhalb des Mondes ($x > x_2$) zur inneren Region ($|x| < 1 - \mu - R_2$, wobei R_1 =Radius der Erde, R_2 =Radius des Mondes) zu gelangen und umgekehrt. Ein umkreisen der Erde ist aufgrund der Energiebilanz $\Omega_2 < \Omega_1$ nie ausgeschlossen. Diskutiert man jedoch die Frage: Existiert ein E für die ein Raumfahrzeug von der Erde aus startend den Mond umkreisen kann, ohne das System Erde-Mond verlassen zu können?, so ist dies durchaus der Fall. Die Bedingung dafür lautet $\Omega_1 < E < \Omega_2$.
Trotz dieser Betrachtungen ist anzumerken, dass energetische Bedingungen nicht genügen, um die Trajektorie des Raumfahrzeugs oder lediglich Charakteristika, wie z.B. Anzahl der Umdrehungen des zu betrachtenden Körpers bzw. das Verhalten des Körpers für $t \to \infty$ vorherzusagen. Die Dynamik des Systems ist durch die Anfangsbedingungen determiniert, wobei sie bei diesem Problem sensitiv von ihnen beeinflusst wird.

2. Iterationsverfahren

2.1. Runge-Kutta-Verfahren

In Kapitel II ergab sich die Differenzialgleichung (8). Aus Gleichung (9) ist $\vec{f}(z_1,z_2,z_3,z_4)$ bekannt. Folglich betrachten wir das Problem:

$$\dot{\vec{z}} = \vec{f}(\vec{z}) \tag{10}$$

Aufgrund der niedrigen Konvergenzordnung $q = 1$ verwenden wir das Iterationsverfahren (9) nicht. Stattdessen nutzen wir das folgende Verfahren mit Konvergenzordnung 3:

$$\vec{z}^{\,n+1} = \vec{z}^{\,n} + h\left(\frac{1}{6}\vec{k}_1 + \frac{4}{6}\vec{k}_2 + \frac{1}{6}\vec{k}_3\right) \tag{11}$$

mit

$$\begin{cases} \vec{k}_1 = \vec{f}(\vec{z}^{\,n}) \\ \vec{k}_2 = \vec{f}\left(\vec{z}^{\,n} + \frac{h}{2}\vec{k}_1\right) \\ \vec{k}_3 = \vec{f}\left(\vec{z}^{\,n} - h\vec{k}_1 + 2h\vec{k}_2\right) \end{cases}$$

2.2. Symplektisches Integrationsverfahren

In der Hamiltonischen Mechanik bietet es sich an das Symplektische Integrationsverfahren anzuwenden. Dabei wird der Gesamthamiltonoperator in zwei Operator aufgespalten. Speziell für unser Problem ergibt sich:

$$H = H_1 + H_2 \quad H_1 = \frac{(p_x + y)^2}{2} + \frac{(p_y - x)^2}{2} \quad H_2 = -\Omega \tag{12}$$

Aus diesen beiden Operatoren H_1, H_2 bekommen wir die Evolutionsoperatoren unter Anwendung der Vorschrift zur Berechnung der Bewegungsgleichungen.

$$\Psi_1 = \begin{pmatrix} \vec{x} + h\nabla_{\vec{p}} H_1 \\ \vec{p} \end{pmatrix}$$
$$\Psi_2 = \begin{pmatrix} \vec{x} \\ \vec{p} + h\nabla_{\vec{x}}\Omega \end{pmatrix}$$

Hiermit gelangen wir zu der vollständigen Iterationsvorschrift für dieses Verfahren, dabei werden Halbschritte bei der Ausführung von Ψ_2 gemacht.

$$\Psi_2\left(\frac{h}{2}\right)\Psi_1(h)\Psi_2\left(\frac{h}{2}\right) \Leftrightarrow \begin{array}{l} \vec{p}^{\,n+\frac{1}{2}} = \vec{p}^{\,n} + \frac{h}{2}\nabla_{\vec{x}}\,\Omega(\vec{x}^{\,n}) \\ \vec{x}^{\,n+1} = \vec{x}^{\,n} + h\nabla_{\vec{p}}\,H_1(\vec{p}^{\,n},\vec{x}^{\,n}) \\ \vec{p}^{\,n+1} = \vec{p}^{\,n+\frac{1}{2}} + \frac{h}{2}\nabla_{\vec{x}}\,\Omega(\vec{x}^{\,n}) \end{array}$$

2.2.1. Energieerhaltung

Für den Test der Energieerhaltung haben wir beide numerische Verfahren benutzt, um ein Vergleich zu haben. Zur Berechnung der Energie im Symplektischen Verfahren müssen die berechneten Impulse wieder in die entsprechenden Geschwindigkeiten transformiert werden. Die Transformationsgleichungen lauten:

$$\begin{cases} p_x &= \dot{x} - y \\ p_y &= \dot{y} + x \end{cases} \tag{13}$$

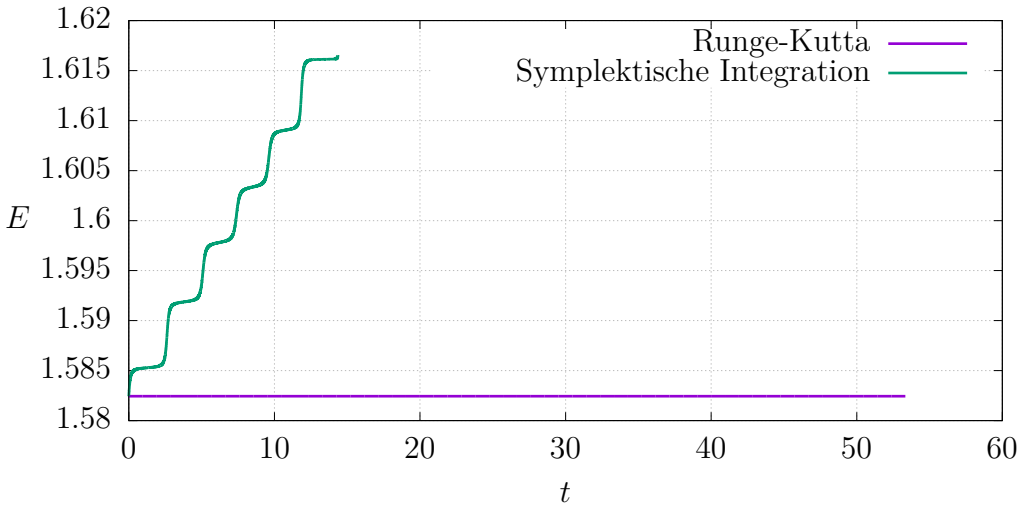

Figure 4: Energie-Zeit Diagramm zur Überprüfung der Energieerhaltung mit den Werten $v = 2.55541$ und $\alpha = 1.44513$.

Das Symplektische Verfahren zeigt bei allen gewählten Zeitschritten einen Anstieg der Gesamtenergie an. Ein maximalen Anstiege scheinen periodisch aufzutreten. Beim Runge-Kutta-Verfahren hingegen bleibt die Energie erhalten. Eine mögliche Erklärung dafür ist, dass das Runge- Kutta-Verfahren aufgrund der höheren Konvergenzordnung genauer arbeitet. Die Periodizität der großen Energieverluste könnte ein Indiz dafür sein, dass die Trajektorie zu diesen Zeitpunkten die stärkste Richtungsänderung aufweist, wodurch sich somit auch der größte Rechenfehler ergibt. Daraus sieht man wie stark die Wahl des Iterationsverfahren die Flugbahn beeinflussen kann, zumal die Dynamik sensitiv ist.

3. Interessante Trajektorien

Allgemein ist das Ende der Flugbahn erreicht, sobald eine der folgenden 4 Bedingungen zutreffen:

$$\text{if}(|x| > 2)$$
$$\text{if}(|y| > 2)$$
$$\text{if}\left((x+\mu)^2 + y^2 < R_1^2\right)$$
$$\text{if}\left(((x-(1-\mu))^2 + y^2 < R_2^2\right)$$

wobei $R_1 = 0.2 =$ Radius der Erde und $R_2 = 0.01 =$ Radius des Mondes

Weiterhin ist aus Gleichung (13) (S. 8) bekannt, wie sich die initialen Impulse $\vec{p} = (p_x, p_y)$ aus den initialen Geschwindigkeiten $\vec{v} = (v_x, v_y)$ berechnen. Die Vorgabe der jeweiligen $\vec{v}$ erfolgt in den folgenden jeweiligen Abschnitten.

3.1. Erde-Mond-Erde

Als erstes untersuchen wir hier Trajektorien denen es, wie im Abschnitt 1 beschrieben, energetisch möglich ist die Erde zu verlassen und den Mond zu umkreisen, jedoch nicht das Erde-Mond-System zu verlassen. Die Flugbahnen enden somit auf dem Mond oder auf der Erde, wobei wir unsere Betrachtung auf die Bahnen beschränken, welche auf der Erdoberfläche enden. Weiterhin ist j die Anzahl der Mondumdrehungen. Diese wird

berechnet, indem man annimmt, dass zu Beginn der Flugbahn $j = 0$. Weiterhin wird j gemäß der Bedingung (14) bis zum Ende der Trajektorie erhöht:

$$\begin{aligned}&\text{if}(y(t_n) > 0)\{\\ &\quad \text{if}(\text{sgn}(x(t_n) - (1-\mu)) \neq \text{sgn}(x(t_{n+1}) - (1-\mu))) \quad j = j + 1;\\ &\}\end{aligned} \tag{14}$$

wobei sgn(x) im Abschnitt 4 (S. 15) definiert wurde

Aus der Gleichung (6) ergibt sich der Bereich für die initialen Geschwindigkeitsbeträge $|\vec{v}|$, die diese Trajektorien begünstigen. Ein Korrekturwert von 0.1 wird aufgrund der Dissipativität des Iterationsverfahrens hinzugefügt.

$$2.60861 \approx \sqrt{2 \cdot (\Omega_E - \Omega_2)} + 0.1 > |\vec{v}| \geq \sqrt{2 \cdot (\Omega_E - \Omega_1)} \approx 2.49541 \tag{15}$$

Hierbei ist $\Omega_E = \Omega(x_E,y_E)$, wobei (x_E,y_E) ein Punkt auf der Erdoberfläche ist. Hierfür wählten wir $(x_E,y_E) = (0.15,0)$. Weiterhin wird die Orientierung der initialen Geschwindigkeit durch den Winkel $\alpha = \arctan\left(\frac{\dot{y}}{\dot{x}}\right)$ bestimmt, also ist $v_x = |\vec{v}|\cos(\alpha)$ und $v_y = |\vec{v}|\sin(\alpha)$. Somit haben wir zwei Werte, welche wir auf der Suche nach der maximalen Umdrehungszahl um den Mond variieren können α und $|\vec{v}|$. Dabei lassen wir $|\vec{v}|$ konstant und suchen den Winkel $\alpha_{\max}$ mit dem wir die bis dahin meisten Mondumrundungen erzeugt haben. Danach speichern wir den Wert für diesen Winkel und setzen ihn zurück und variieren $|\vec{v}|$. Im Anschluss starten wir wieder den Suchlauf nach $\alpha_{\max}$. Das Tupel $(|\vec{v}|,\alpha)$ mit den größtem j und dem von uns gewünschten Ende (Weltall oder Erdoberfläche) wird abgespeichert. Die jeweiligen Variationsalgorithmen sind gleich und im Abschnitt 5 (S. 15) beschrieben. Hierbei sei m die Dimension des Vektors, welcher für den Variationsalgorithmus benötigt wird. Für den Winkel gilt $\alpha \in [0,\frac{\pi}{2})$.

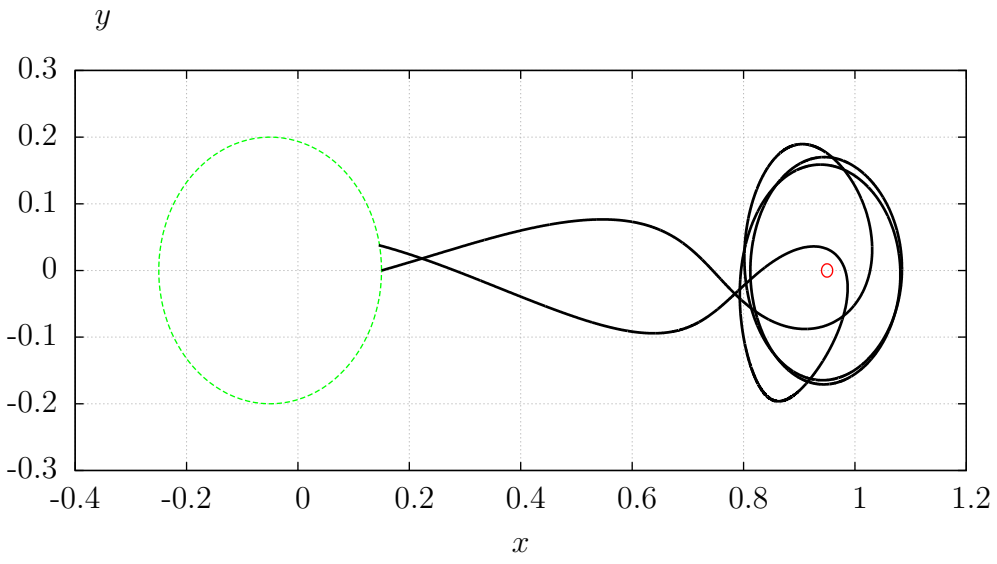

Figure 5: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 20$,
$\epsilon = 10^{-5}$, $|\vec{v}| = 2.52937$ und $\alpha = 0.229572$.

In Abbildung 5 ist die vierfache Umdrehung des Mondes mit anschließendem Zurückkehren zur Erde dargestellt. Neben der Abbruchbedingung (20) (Seite 16), welche hier lediglich für die Variation des Winkels benötigt wurde, musste der Abbruch der Variation beider Variablen $|\vec{v}|$ und α erfolgen sobald $j = 4$.

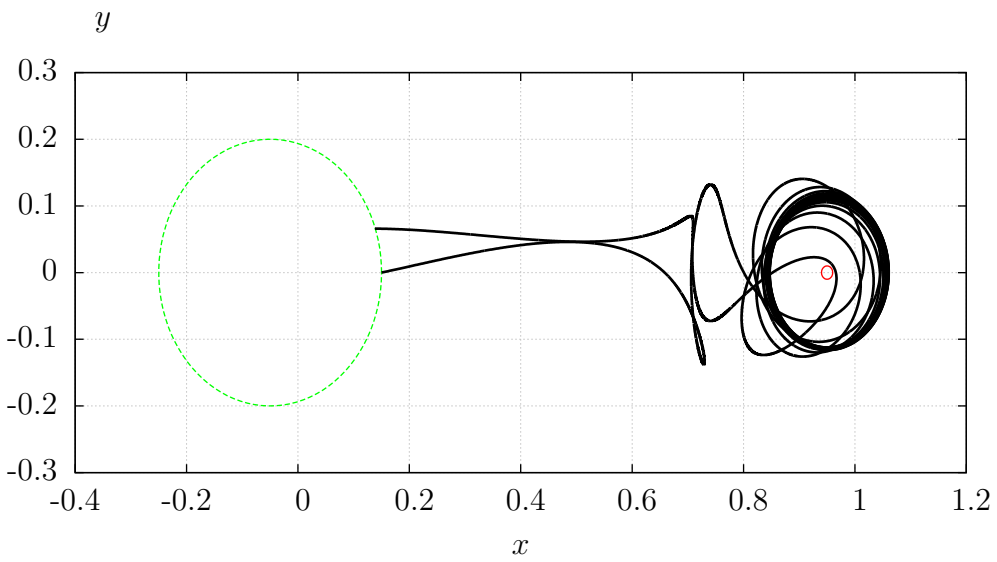

Figure 6: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 120$,
$\epsilon = 10^{-8}$, $|\vec{v}| = 2.51215$ und $\alpha = 0.177274$.

In Abbildung 6 ist die achtzehnfache Umdrehung des Mondes mit anschließendem Zurückkehren zur Erde dargestellt. Dies ist die maximale Anzahl an Mondumrundungen mit anschließender Erdrückkehr, welche wir mit dem Symplektischen Integrationsverfahren gefunden haben.

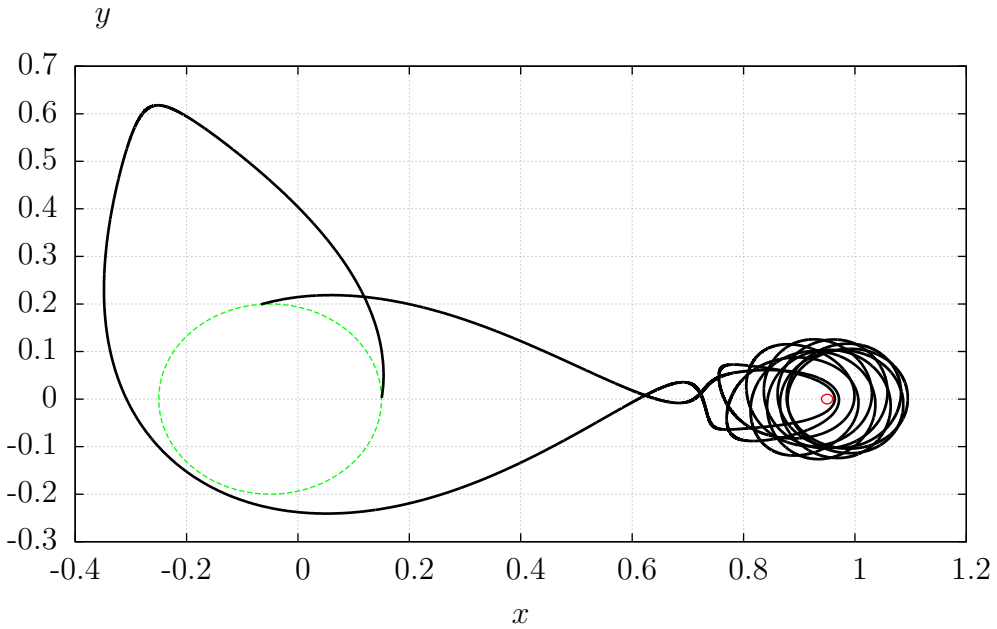

Figure 7: Simuliert mit dem Runge-Kutta-Verfahren mit den Werten $m = 10$,
$\epsilon = 10^{-6}$, $|\vec{v}| = 2.4983$ und $\alpha = 1.42628$.

In Abbildung 7 ist die 18-fache Umdrehung des Mondes mit anschließendem Zurückkehren zur Erde dargestellt. Dies ist die maximale Anzahl an Mondumrundungen mit anschließender Erdrückkehr, welche wir mit dem Runge-Kutta-Verfahren gefunden haben.

Vergleicht man die Trajektorien Abb. 7 und Abb.6, so erkennt man, dass die mit dem Runge-Kutta-Verfahren simulierten Trajektorien erst eine Erdumdrehung verrichten bevor sie den Mond umkreisen, was bei der Symplektischen Integration nicht der Fall ist. Außer, dass also sich die α stark unterscheiden, benötigt das Runge-Kutta-Verfahren für maximale Mondumdrehungen kleinere Anfangsgeschwindigkeiten als das Symplektische Integrationsverfahren. Unter Betrachtung der Abbildungen 12 und 13 (Abschnitt 6, S. 17) wird ein weiterer Unterschied ersichtlich. Untersucht wurden die Trajektorien die auf der Mondoberfläche enden mit $|\vec{v}|$, wie bereits definiert, als Anfangsgeschwindigkeit. In Abb. 12 ist mit 17 die maximale Anzahl von Mondumdrehungen mithilfe des Symplektische Integrationsverfahren gefunden worden. In Abb. 13 wurde mit 1378 eine vergleichsweise große Anzahl an Mondumdrehungen mit dem Runge-Kutta-Verfahren simuliert, welche nicht die maximale Anzahl ist. Das Problem hierbei war die Rechenzeit, welche für eine so große Anzahl an Umdrehungen sehr groß wird.

3.2. Erde-Mond-Weltall

Die nächsten für uns interessanten Trajektorien sind diejenigen, welche nach ihren Mondumrundungen das Erde-Mond-System verlassen. Für die initiale Geschwindigkeit $\vec{v}$ gilt:

$$0.697826 = \sqrt{2 \cdot (\Omega(x_2 + 0.1{,}0) - \Omega_3)} + 0.1 > |\vec{v}| \geq \sqrt{2 \cdot (\Omega_E - \Omega_2)} = 0.250861 \tag{16}$$

Die Bedingung (16) ermöglicht den Trajektorien das Erde-Mond-System um den Lagrangepunkt L_2 zu verlassen, wobei es widerum unmöglich ist das System um L_3 bzw. an anderen Orten (x,y) zu verlassen. Trotzdessen ist ein Ende der Trajektorie auf der Erd- bzw. Mondoberfläche nicht ausgeschlossen.

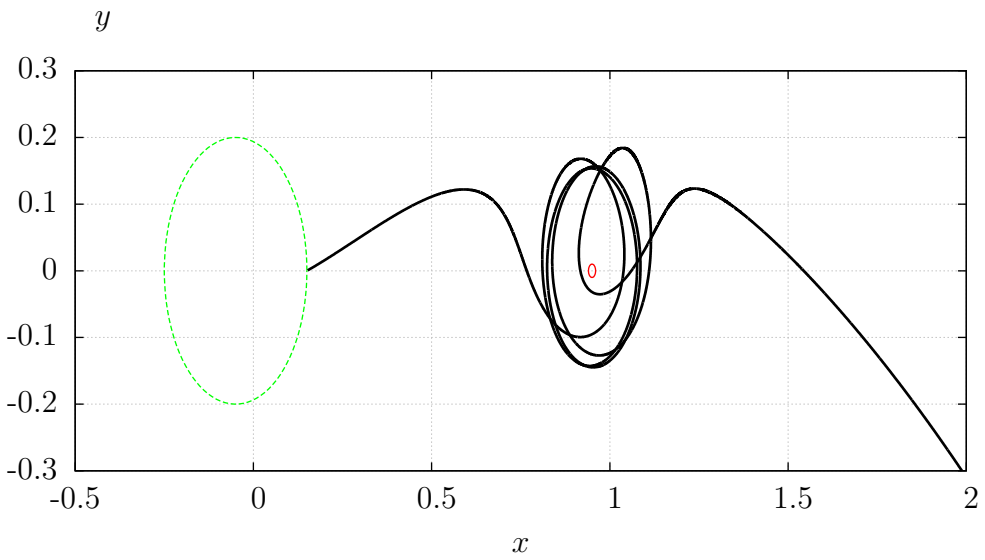

Figure 8: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 20$,
$\epsilon = 10^{-5}$, $|\vec{v}| = 2.5246$ und $\alpha = 0.275675$.

In Abbildung 8 ist die vierfache Umdrehung des Mondes mit anschließendem Zurückkehren zur Erde dargestellt. Die Abbruchbedingung wurde auch hier, wie im Unterabschnitt 3.1 beschrieben, modifiziert.

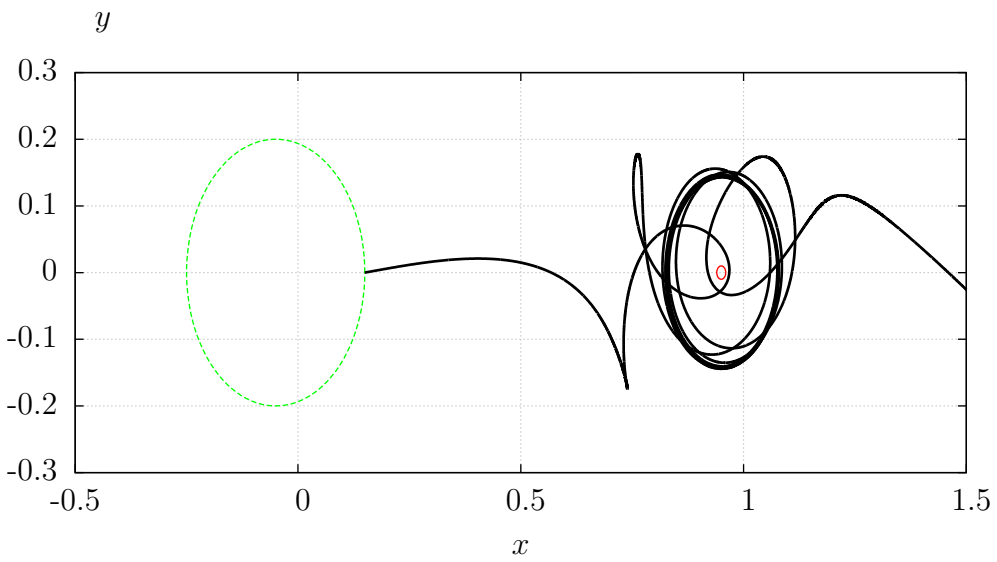

Figure 9: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 120$,
$\epsilon = 10^{-8}$, $|\vec{v}| = 2.5206$ und $\alpha = 0.119901$.

In Abbildung 9 ist die neunfache Umdrehung des Mondes mit anschließendem Verlassen des Erde-Mond-Systems dargestellt. Dies ist die maximale Anzahl an Mondumrundungen ohne zwischenzeitlicher Rückkehr zum Erdorbit, welche wir mit dem Symplektischen Integrationsverfahren gefunden haben.

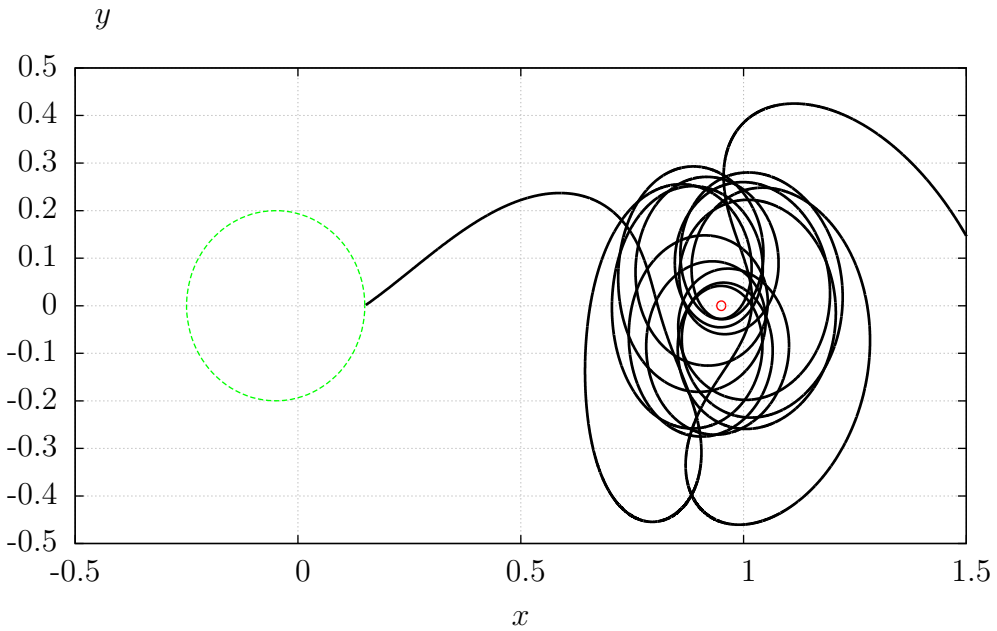

Figure 10: Simuliert mit dem Runge-Kutta-Verfahren mit den Werten $m = 30$,
$\epsilon = 10^{-6}$, $|\vec{v}| = 2.58559$ und $\alpha = 0.578022$.

In Abbildung 10 ist die 12-fache Umdrehung des Mondes mit anschließendem Verlassen des Erde-Mond-Systems dargestellt. Dies ist die maximale Anzahl an Mondumrundungen ohne zwischenzeitlicher Rückkehr zum Erdorbit, welche wir mit dem Runge-Kutta-Verfahren gefunden haben.

3.3. Weltall-Mond-Erde-Weltall

Hierbei soll die im Weltall beginnende Trajektorie mehrere Mondumdrehungen aufweisen, bevor sie nach mehreren Erdumrundungen im Weltall verschwindet. Als Startpunkt wählen wir $(x_2 + 0.1{,}0)$, wobei x_2 die x-Koordinate des Lagrangepunktes L_2 ist. Für die Suche passen wir hier den Variationsalgorithmus an, indem wir ein j_E definieren, was die Anzahl der Erdumrundungen angibt. Das für j_E verwendete Zählverfahren gleicht dem für j (14). Damit erweitern wir den Variationsalgorithmus um folgende Abbruchbedingung:

$$\begin{aligned} &\text{if}(j \geq 2)\{ \\ &\quad \text{if}(j_E \geq 2)\ \text{Abbruch;} \\ &\} \end{aligned} \tag{17}$$

Der nun durch die Bedingung (17) erweiterte Variationsalgorithmus findet eine Trajektorie, welche mindestens 2 Mond- bzw. 2 Erdumdrehungen aufweist, bevor die Trajektorie im Weltall endet. Weiterhin gestattet dieses Suchverfahren auch die Vorgabe von mehr als jeweils 2 Umrundungen.Für den Winkel gilt $\alpha \in [\pi, \frac{3\pi}{2})$. Für die Geschwindigkeit $\vec{v}$ gilt weiterhin Bedingung (16):

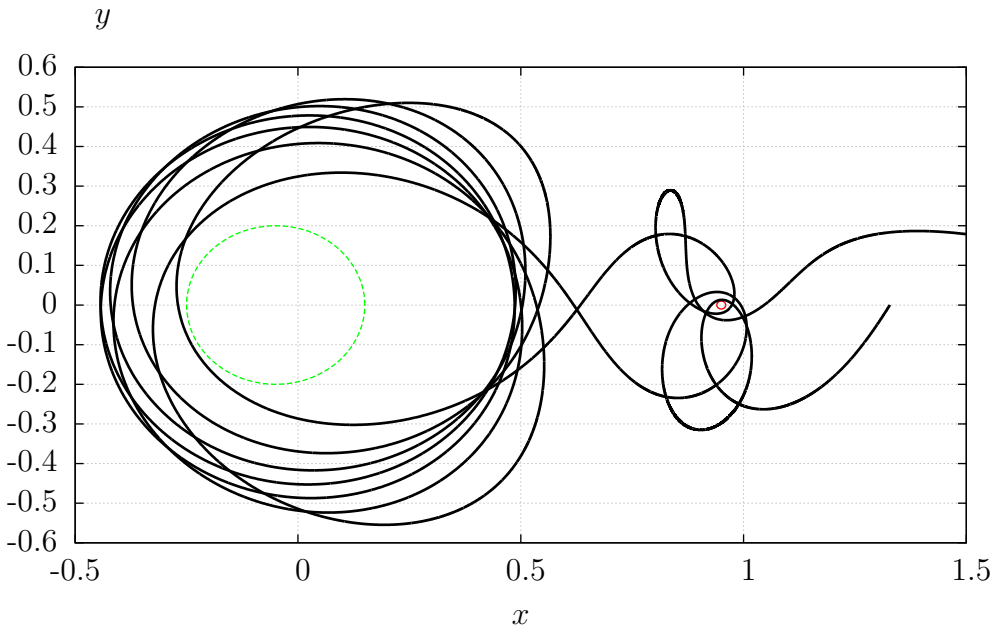

Figure 11: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 100$,
$\epsilon = 10^{-6}$, $|\vec{v}| = 0.557495$ und $\alpha = 4.20377$.

Die in Abb. 11 dargestellte Trajektorie weist 7 Erdumrundungen und 3 Mondumrundungen auf, wobei eine Mondumrundung erst nach den Erdumkreisungen stattfindet, woraufhin der Körper ins Weltall verschwindet. Bei Vorgabe höherer Umkreisungen sind häufigere Erd-Mond-Übergänge nicht ausgeschlossen, weswegen wir uns bei der Vorgabe der Minimalumrundungen mit jeweils 2 zufrieden gegeben haben. Sofern mehrere Umrundungen ohne Erhöhung der Erd-Mond-Transitionen gefordert wären, müsste man den Suchalgorithmus weiterhin modifizieren.
Abb. 14 (S. 18) zeigt eine Trajektorie mit mehreren Übergängen und jeweils 4 Umrundungen.

Part IV.
Mathematische Grundlagen und Algorithmen

4. Bisektionsverfahren

Sei f eine stetige Funktion, wobei $f(x_0)$ und $f(y_0)$ verschiedene Vorzeichen haben, so existiert laut des Zwischenwertsatzes eine Nullstelle $z \in [x_0; y_0]$. Durch folgende Iteration kann man z approximativ bestimmen.

$$\begin{cases} z_k = \frac{x_{k-1}-y_{k-1}}{2}, & \forall\, k \in \mathbb{Z}^+ \\ x_k = z_k, y_k = y_{k-1}, & \text{falls} \quad \operatorname{sgn}(f(z_k)) = \operatorname{sgn}(f(x_{k-1})) \\ y_k = z_k, x_k = x_{k-1}, & \text{sonst} \end{cases} \tag{18}$$

In der Gleichung (18) ist $\operatorname{sgn}(x)$ die Signumfunktion für die folgendes gilt:

$$\operatorname{sgn}(x) = \begin{cases} 1, & \forall\, x > 0 \\ 0, & \text{falls} \quad x = 0 \\ -1, & \text{sonst} \end{cases}$$

5. Variationsalgorithmus

Sei $\vec{x}(t)$ eine Trajektorie um den Mond, wobei die Anzahl der Mondumdrehung j maximiert werden soll, indem der Anfangswert w variiert wird. Weiterhin soll gelten $w \in [a,b)$. Man definiert $\vec{W} \in \mathbb{R}^m$ und $\vec{J} \in \mathbb{N}^m$, wobei $\vec{W} = (w_0, w_1, .., w_{m-1})$ und $\vec{J} = (j_0, j_1, .., j_{m-1})$. Dabei ist $w_i = a + i \cdot w_{\text{step}}\ \forall\, i \in \{0,1,2,..,m-1\}$ mit $w_{\text{step}} = \frac{b-a}{m}$ und j_i gibt die Anzahl an Umdrehungen an, welche mit dem Anfangswert w_i erreicht wurden. Definiert man nun:

$$\begin{aligned} j_{\max} &= \max\{j_0, j_1, ..., j_{m-1}\} \\ l &= \text{Anzahl der Maximalelemente in } \vec{J} \\ k &= \text{kleinster Index aller Maximalelemente in } \vec{J} \end{aligned}$$

Das heißt, sofern es $l > 1$ Maximalelemente in $\vec{J}$ gibt, ist k das Minimum der Menge der Indizes aller $j_{\max}$ in $\vec{J}$.

Mithilfe dieser Grundlagen lautet der Variationsalgorithmus:

$$
\begin{aligned}
&\text{if}(l == m)\\
&\quad w_{\text{step,neu}} = \epsilon;\\
&\text{else}\{\\
&\quad \text{if}(l == 1)\{\\
&\qquad \text{if}(k == 0)\{\\
&\qquad\quad w_0 = w_k;\\
&\qquad\quad w_{\text{step,neu}} = \frac{w_{\text{step,alt}}}{m};\\
&\qquad \}\\
&\qquad \text{if}(k == m-1)\{\\
&\qquad\quad w_0 = w_{k-1};\\
&\qquad\quad w_{\text{step,neu}} = \frac{w_{\text{step,alt}}}{m};\\
&\qquad \}\\
&\qquad \text{else}\{\\
&\qquad\quad w_0 = w_{k-1};\\
&\qquad\quad w_{\text{step,neu}} = 2 \cdot \frac{w_{\text{step,alt}}}{m};\\
&\qquad \}\\
&\quad \}\\
&\quad \text{else}\{\\
&\qquad w_0 = w_k;\\
&\qquad w_{\text{step,neu}} = (l-1) \cdot \frac{w_{\text{step,alt}}}{m};\\
&\quad \}\\
&\}
\end{aligned}
\tag{19}
$$

Wobei dieser Algorithmus endet, sofern die Abbruchbedingung (20) erfüllt ist.

$$
\text{if}(w_{\text{step,neu}} \leq \epsilon) \tag{20}
$$

Sofern alle in $\vec{W}$ enthaltenen Anfangswerte die gleiche Anzahl an Mondumrundungen zur Folge haben, erfolgt der Abbruch. Existieren mehrere aber weniger als m Maximalwerte, so beginnt der Algorithmus erneut ab dem Anfangswert $j_{\max}$ mit dem kleinsten Index k und endet bei $j_{\max}$ mit dem größten Index. Es wird also nur zwischen diesen $j_{\max}$ geschaut, wobei wir davon ausgegangen sind, dass diese Maximalwerte hintereinanderliegen, also dass, wenn l die Anzahl der $j_{\max}$ ist, der größter Index der Maximalwerte $k + l - 1$ ist. Folglich ist $w_{\text{step,neu}} = (l-1) \cdot \frac{w_{\text{step,alt}}}{m}$. Tritt der Fall auf, dass es genau ein $j_{\max}$ gibt und k dessen Index sei, so wird um diesen Maximalwert geschaut, also zwischen w_{k-1} und w_{k+1}. Folglich ist hier $w_{\text{step,neu}} = 2 \cdot \frac{w_{\text{step,alt}}}{m}$. Da dies für die Fälle, dass $k = 0$ bzw. $k = m - 1$ nicht möglich ist, wird hier zwischen w_0 und w_1 bzw. w_{m-2} und w_{m-1} geschaut. Somit ist die neue Schrittweite $w_{\text{step,neu}} = \frac{w_{\text{step,alt}}}{m}$.

Part V.
Anhang

6. Andere interessante Trajektorien

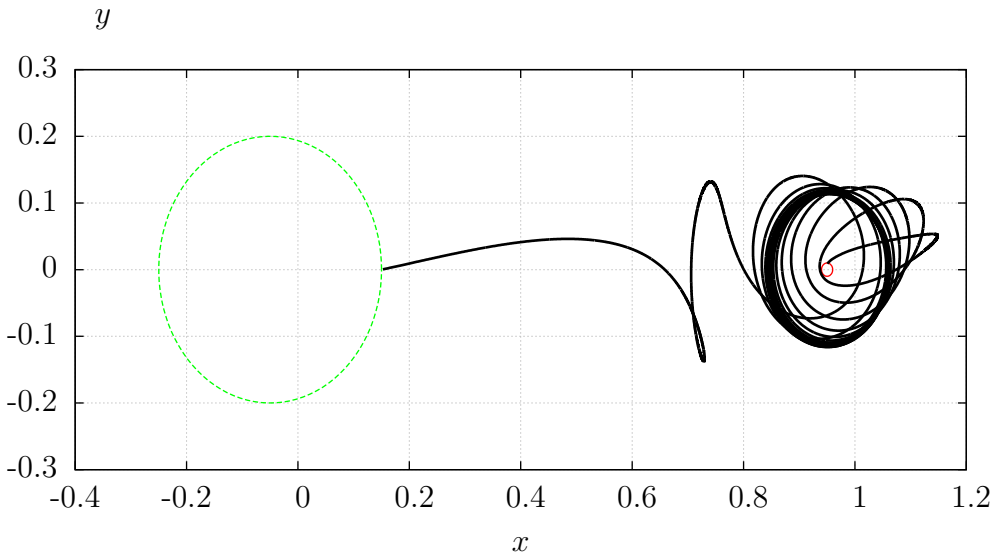

Figure 12: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 120$,
$\epsilon = 10^{-8}$, $|\vec{v}| = 2.54291$ und $\alpha = 0.235626$.

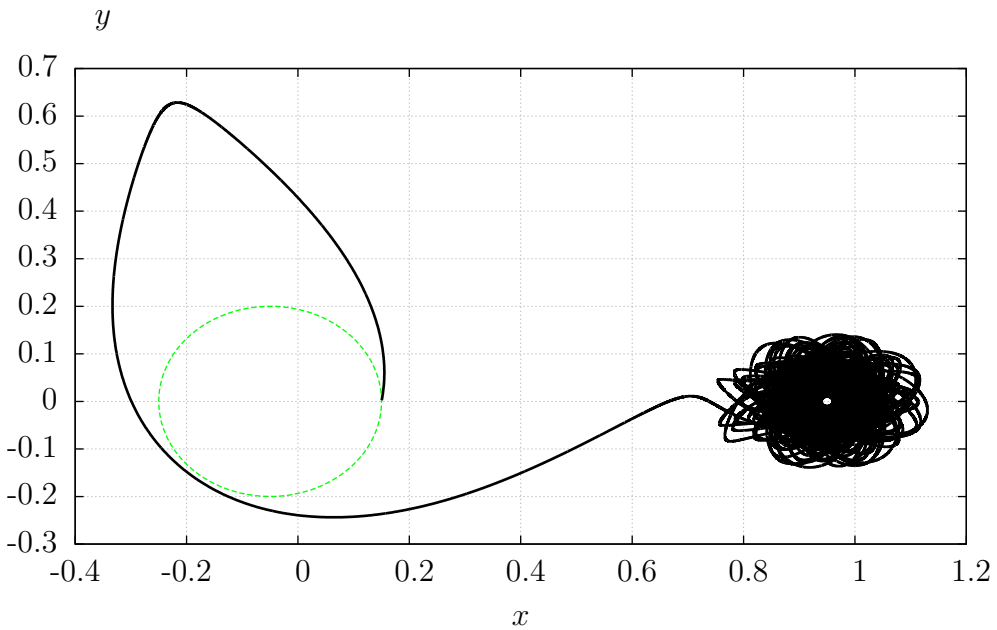

Figure 13: Simuliert mit dem Runge-Kutta-Verfahren mit den Werten $m = 10$,
$\epsilon = 10^{-6}$, $|\vec{v}| = 2.4983$ und $\alpha = 1.40115$.

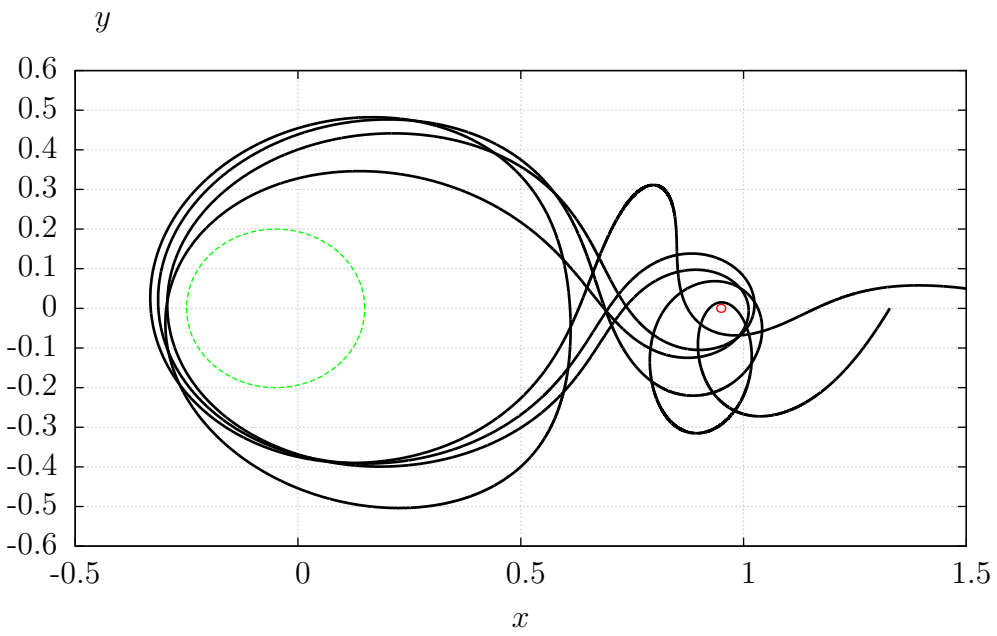

Figure 14: Simuliert mit dem Symplektischen Integrationsverfahren mit den Werten $m = 100$,
$\epsilon = 10^{-6}$, $|\vec{v}| = 0.56685$ und $\alpha = 4.21256$.

7. Programmiercode

```
1  /****************** planet-system ******************/
2
3  using namespace std;
4  #include <stdlib.h>
5  #include <iostream>
6  #include <fstream>
7  #include <math.h>
8
9
10 /*************  important parameters  **************/
11 const int neqn = 4;            //Anzahl der Gleichungen
12 bool schalter = true;          //Bedingung fue das jeweilige Verfahren
13 bool abb=false;                //fuer Abbruchbedingung
14 bool abb4=false;               //Bool fuer gewuenschtes Ende der
    ↪ Trajektorie
15 const double tstep=0.001;      //Zeitschritt
16 const double gen=0.001;        //Genauigkeit der Lagrangepunkte
17 const double eps=0.00001;      //Genauigkeit des Geschw. bzw
    ↪ Winkelschritts fuer Abb.
18 const double mu=0.05;          //Mondmasse
19 const double mu2=1-mu;         //Erdmasse
20 const double r1=0.2            //Erdradius
```

```
21  const double r2=0.01;          //Mondradius
22  double omeg1,omeg2,omeg3;      //double fuer jeweilige Potentiale
23  double z1[3], z2;              //fuer das Bisektionsverfahren
24  const double pi=4.*atan(1.);
25  //const double alpha=0.0995*pi;
26
27  /*******************************************************/
28  int sgn(double h){//Signumfunktion
29    if (h==0) return 0;
30    else  return (h>0) ? 1 : -1;
31  }
32  /******************** Runge-Kutta Verfahren (Konvergenzordnung 3)
    ↪  ************/
33  void rk(double y[],int n,double x,double h,
34       void (derivs)(double, double[], double[]))
35  {
36    int j,i;
37    double h2,h6,xh, xhh,y1[n],k1[n],k2[n],k3[n];
38    h2=h*0.5;     h6=h/6.;
39    //Simpsonregel
40    xh=x+h2;          xhh=x+h;
41    (derivs)(x,y,k1);
42    for(i=0;i<n;i++) y1[i]=y[i]+h2*k1[i];
43    (derivs)(xh,y1,k2);
44    for(i=0;i<n;i++) y1[i]=y[i]-h*k1[i]+2*h*k2[i];
45    (derivs)(xhh,y1,k3);
46    x+=h;
47    for(i=0;i<n;i++) y[i]+=h6*(k1[i]+4.*k2[i]+k3[i]);
48  }
49
50  void planet(double t,double x[],double
    ↪  dxdt[])//Diffenentialgleichungen
51  {
52    double l1, l2, k1, k2;//x1=y[0], x2=y[1], x3=y[2], x4=y[3]
53    k1=x[0]+mu;
54    k2=k1-1.;
55    l1=k1*k1+x[1]*x[1];
56    l2=k2*k2+x[1]*x[1];
57    dxdt[0] =x[2];
58    dxdt[1] =x[3];
59    dxdt[2] =2*x[3]+x[0]-(1-mu)*k1/pow(l1, 1.5)-mu*k2/pow(l2, 1.5);
60    dxdt[3] =-2*x[2]+x[1]-(1-mu)*x[1]/pow(l1, 1.5)-mu*x[1]/pow(l2, 1.5);
```

```
61  }
62
63  /*******************************************************/
64
65  double f(double y_0){//Funktion zur Bestimmung der Nullstelle
66    double k1, k2,g;
67    k1=y_0+mu;
68    k2=k1-1.;
69    g=y_0-(1-mu)*sgn(k1)/(k1*k1)-mu*sgn(k2)/(k2*k2);
70    return g;
71  }
72
73  /*******************************************************/
74
75  double Omega(double a, double b){//Fkt. zur bestimmung des Potentials
76    double u;
77    u=(a*a+b*b)/2.+(1-mu)/sqrt((a+mu)*(a+mu)+b*b)+mu/sqrt((a-1.+mu)*(a-1
      ↪  .+mu)+b*b)+mu*(1-mu)/2.;
78    return u;
79  }
80  /******************Bisektions-Methode*******************/
81  /*Nullstelle bestimmen*/
82  void bisec(int a, int m, double b, int i){
83    if(a==m)z1[i]=b;
84    else          z2=b;
85  }
86
87  double lag(int j){
88    double w,c;
89    int m, g;
90    w=fabs(z2-z1[j]);
91    m=sgn(f(z1[j]));
92    while(w>gen){
93      c=(z2+z1[j])/2.;
94      g=sgn(f(c));
95      bisec(g,m, c,j);
96      w=fabs(z2-z1[j]);
97    }
98    return c;
99  }
100 /******************* Symplektisches Integrationsverfahren
    ↪  *******************/
```

```
101 /*Differentialgleichungen*/
102 void Hamilton(double t, double x[],double F[], bool Imp){
103   double l1, l2, k1, k2;//x1=y[0], x2=y[1], x3=y[2], x4=y[3]
104   if(Imp){
105     k1=x[0]+mu;
106     k2=k1-1.;
107     l1=k1*k1+x[1]*x[1];
108     l2=k2*k2+x[1]*x[1];
109     F[2]=-1*(x[0]-(1-mu)*k1/pow(l1, 1.5)-mu*k2/pow(l2, 1.5));
110     F[3]=-1*(x[1]-(1-mu)*x[1]/pow(l1, 1.5)-mu*x[1]/pow(l2, 1.5));
111   }
112   else{
113     F[0]=x[2]+x[1];
114     F[1]=x[3]-x[0];
115   }
116 }
117
118 void Sympl_Integr(double x[],int n,double &t,double h,
119      void (Ham_deriv)(double, double[], double[], bool))
120 {
121   int i,k=int(n/2);
122   double L[n], h2=h/2.;
123   (Ham_deriv)(t,x,L,true);
124   for(i=k;i<n;i++){//Halbschritt der Impulse
125     x[i]+=-h2*L[i];
126   }
127   (Ham_deriv)(t,x,L,false);
128   for(i=0;i<k;i++){//Schritt der Orte
129     x[i]+=h*L[i];
130   }
131   (Ham_deriv)(t,x,L,true);
132   for(i=k;i<n;i++){//Halbschritt der Impulse
133     x[i]+=-h2*L[i];
134   }
135   t+=h;
136 }
137
138 /*********************Abbruchbedingungen***************/
139
140 double cond_kreis(double x,double a, double r){
141   double abs_quad;
142   abs_quad=r*r-(x-a)*(x-a);
```

```
143   return abs_quad;
144 }
145
146 void cond(double x[]){
147   double g;
148   if (fabs(x[0])>2){//Kasten mit Kantenlaenge 4 in x-Richtung
149     abb=true;
150     abb4=true;
151   }
152   if (fabs(x[1])>2){//Kasten mit Kantenlaenge 4 in y-Richtung
153     abb=true;
154     abb4=true;
155   }
156   if(fabs(x[0]+mu)<r1){//Erdoberflaeche
157     g=x[1]*x[1];
158     if(g<cond_kreis(x[0],-1*mu,r1)){
159       abb=true;
160       //abb4=true;
161     }
162   }
163   if(fabs(x[0]-mu2)<r2){//Mondoberflaeche
164     g=x[1]*x[1];
165     if(g<cond_kreis(x[0],mu2,r2))abb=true;
166   }
167 }
168
169 int  count(double x[],double l, int cnt){//Zaehler der Mondumrundungen
170   if(x[1]>0){
171     if(sgn(l-mu2)!=sgn(x[0]-mu2))        cnt+=1;
172   }
173   return cnt;
174 }
175
176 int  count1(double x[],double l, int cnt){//Zaehler der Erdumrundungen
177   if(x[1]<0){
178     if(sgn(l+mu)!=sgn(x[0]+mu))        cnt+=1;
179   }
180   return cnt;
181 }
182
183 /*********************************Suchalgorithmus*********************
    **********/
```

```
184
185 void opt_wnk(int j, double &wnk_step, double &wnk, double wnk_vec[],
    ↪  int i, int m){
186   if(j==m)        wnk_step=eps;
187   else{
188     if(j==1){
189       if (i==0){
190         wnk=wnk_vec[i-1];
191         wnk_step=wnk_step/(double(m));
192       }
193       if(i==m-1){
194         wnk=wnk_vec[i-1];
195         wnk_step=wnk_step/(double(m));
196       }
197       else{
198         wnk=wnk_vec[i-1];
199         wnk_step=2*wnk_step/(double(m));
200       }
201     }
202     else{
203       wnk=wnk_vec[i];
204       wnk_step=double(j-1)*wnk_step/(double(m));
205     }
206   }
207 }
208
209 int list_max(int list1[], double list2[], int m){//Maximum des Vekors
    ↪  bestimmen
210   int y=0;
211   for(int i=0;i<m;i++)        if(list1[i]>y)        y=list1[i];
212   return y;
213 }
214
215 int list_koord(int list1[], double list2[], int m, int y){//minimale
    ↪  Koordinate von j_max
216   for(int i=0;i<m;i++){
217     if(list1[i]==y)        return i;
218   }
219 }
220
221 int list_len(int list1[], double list2[], int m, int y){//Anzahl von
    ↪  j_max
```

```
222   int j=0;
223   for(int i=0;i<m;i++){
224     if(list1[i]==y)         j+=1;
225   }
226   return j;
227 }
228
229 double energy(double x[]){//Berechnung der Energie
230   if(schalter) return Omega(x[0],x[1])-((x[2]+x[1])*(x[2]+x[1])+(x[3]-
      ↪  x[0])*(x[3]-x[0]))/2.;
231   else return Omega(x[0],x[1])-(x[2]*x[2]+x[3]*x[3])/2.;
232 }
233
234 main(){
235   int cnt=0, cnt1=0,k=0,max=20,m,m1=0, m2,len,len2, koord,koord2;
236   int cnt_vec[max],cnt_vec2[max];
237   bool abb1=false,abb2=false;
238   double  t=0, wnk_step=(pi/2.)/double(max), wnk=0,w,w1,l, wnkwnk, ww;
239   double x[neqn] ,wnk_vec[max],w_vec[max],lag_punkt[3];
240   const char* Dateiname;
241   z1[0]=-1.9;z1[1]=mu*(-1)+gen;z1[2]=1-mu+gen;
242   if (schalter)Dateiname="DaT_Dateien//planet4.dat";//Datei fuer Sympl.
      ↪  Verf.
243   else Dateiname="DaT_Dateien//planet5.dat";//Datei fuer
      ↪  Runge-Kutta-Verf.
244   //x1=x[0], x2=x[1], x1^punkt=x[2], x2^punkt=x[3] Anfangsbed.
245   x[1]=0.; x[2]=0.; x[3]=0.; t=0.;
246   for(int i=0;i<3;i++){//Speichern der Lagrangepunkte
247     z2=z1[i]+mu2;
248     lag_punkt[i]=lag(i);
249   }
250   omeg1=Omega(lag_punkt[2],0);
251   omeg2=Omega(lag_punkt[0],0);
252   x[0]=0.15;//lag_punkt[2]+gen;         //x-Startpunkt fuer den Trek
253   omeg3=Omega(x[0],x[1]);
254   w=(sqrt(2*(omeg3-omeg1)));
255   w1=(sqrt(2*(omeg3-omeg2))+0.1);
256   double w_step=(w1-w)/double(max);
257   while(abb2==false){//Suchalgorithmus fuer Geschw.
258     for(int n=0; n<max;n++){
259       while(abb1==false){//Suchalgorithmus fuer Winkel
260         for(int i=0; i<max;i++){
```

```
261             x[2]=cos(wnk)*w;//-x[1];//Startgeschwindigkeit/impuls
                ↪  (x-Richtung)
262             x[3]=sin(wnk)*w;//+x[0];//Startgeschwindigkeit/impuls
                ↪  (y-Richtung)
263             if (schalter){//Symplectic Integration
264               while(abb==false){
265                 l=x[0];
266                 Sympl_Integr(x,neqn,t,tstep,Hamilton);
267                 cond(x);
268                 cnt=count(x,l,cnt);     //Zaehler fuer Mondumdrehungen
269                 //cnt1=count1(x,l,cnt1);//Zaehler fuer Erdumdrehungen
270               }
271             }
272             else{//Runge-Kutta Verfahren
273               while(abb==false){
274                 l=x[0];
275                 rk(x,neqn,t,tstep,planet);
276                 cond(x);
277                 cnt=count(x,l,cnt);
278               }
279             }
280             cnt_vec[i]=cnt;
281             wnk_vec[i]=wnk;
282             if(abb4==true){
283               if (cnt==4){//>cnt1){//Bedingung fuer 4 bzw. Maximalbahn
284                 //if(cnt1=2){
285                 ww=w;
286                 wnkwnk=wnk;
287                 n=max;
288                 abb2=true;
289                 abb1=true;
290                 i=max;
291                 //cnt1=cnt;//Bei Maximalbahnbestimmung
292                 //}
293                 //else abb4=false;
294               }
295               else        abb4=false;
296             }
297             wnk+=wnk_step;
298             x[0]=0.15; x[1]=0.;abb=false;cnt=0;t=0;//cnt1=0;
299             if(wnk_step<=eps){//Abbruchbed. fuer Winkel
300               abb1=true;
```

```
301             i=max;
302           }
303         }
304         m=list_max(cnt_vec,wnk_vec, max);
305         koord=list_koord(cnt_vec,wnk_vec, max, m);
306         len=list_len(cnt_vec,wnk_vec, max,m);
307         opt_wnk(len,wnk_step,wnk,wnk_vec, koord, max);
308       }
309       cnt_vec2[n]=m;
310       w_vec[n]=w;
311       w+=w_step;
312       wnk_step=(pi/2.)/double(max); wnk=0;abb1=false;
313     }
314     m2=list_max(cnt_vec2,w_vec, max);
315     koord2=list_koord(cnt_vec2,w_vec, max, m2);
316     len2=list_len(cnt_vec2,w_vec, max,m2);
317     opt_wnk(len2,w_step,w,w_vec, koord2, max);
318     m1=0;
319     if(w_step<=eps){//Abbruchbed. fuer Geschwindigkeit
320       abb2=true;
321     }
322   }
323   ofstream fout(Dateiname,ios::out);
324   fout.setf(ios::scientific);fout.precision(6);
325   w=(sqrt(2*(Omega(0.15,0)-Omega(lag_punkt[1],0) )))+0.06;//ww;
326   wnk=0.46*pi;//wnkwnk;
327   x[0]=0.15;
328   x[1]=0.;
329   x[2]=cos(wnk)*w;//Startgeschwindigkeit (x-Richtung) fuer den Trek
330   x[3]=sin(wnk)*w;//Startgeschwindigkeit (y-Richtung) fuer den Trek
331   if (schalter){//Symplectic Integration
332     x[2]=x[2]-x[1];
333     x[3]=x[3]+x[0];
334     fout<<x[0]<<" "<<x[1]<<" "<< energy(x)<<" "<< t <<endl;
335     while(abb==false){
336       l=x[0];
337       Sympl_Integr(x,neqn,t,tstep,Hamilton);
338       fout<<x[0]<<" "<<x[1]<<" "<< energy(x)<<" "<< t <<endl;
339       cond(x);
340       cnt=count(x,l,cnt);
341     }
342   }
```

```
343     else{//Runge-Kutta Verfahren
344       fout<<x[0]<<" "<<x[1]<<" "<< energy(x)<<" "<< t <<endl;
345       while(abb==false){
346         l=x[0];
347         rk(x,neqn,t,tstep,planet);
348         fout<<x[0]<<" "<<x[1]<<" "<< energy(x)<<" "<< t <<endl;
349         cond(x);
350         cnt=count(x,l,cnt);
351       }
352     }
353     cout<<cnt<<endl;//Ausgabe der Mondumrundungen
354     fout.close();
355   }
```