

2.1 Il problema di Newton con due corpi

Affrontiamo adesso un altro caso molto interessante. Il moto di due corpi soggetti all'attrazione gravitazionale. Come nel caso precedente sul proiettile con attrito il primo passo è esprimere correttamente la componenti dell'accelerazione convertendo le formule del libro di testo che esprimono il modulo della forza di Newton in formule per componenti.

Trattiamo un caso semplice, l'orbita di una massa m (Terra ad esempio) intorno ad una massa M (Sole) con $m \ll M$. Grazie all'ultima condizione possiamo supporre che la massa M sia ferma nell'origine mentre la massa m le orbita intorno. Questa condizione si può modificare in un secondo momento senza complicare troppo la trattazione. Il vettore posizione della massa m (d'ora in poi lo chiameremo Terra) sarà

$$\vec{r} = (x, y, z).$$

Sui libri di testo la forza di attrazione gravitazionale è indicata dalla famosa formula

$$F_G = -\frac{GMm}{r^2}.$$

A mio modesto parere questa formula espressa in questo modo è responsabile della confusione di intere generazioni di studenti. La prima domanda di uno studente che ha davvero voglia di capire sarà *ma cosa è F_G ?* Così espresso non può essere il vettore forza poiché è chiaramente uno scalare. Potrebbe essere il modulo della forza, ma come fa ad essere negativo? Normalmente sui libri di testo si spiega che il segno meno indica il fatto che la forza è attrattiva, ma questo vale solo se F_G è la componente della forza lungo l'asse che congiunge di due corpi orientata dal corpo che esercita la forza a quello che la subisce.

Ammesso e non concesso che uno studente riesca a capire tutto ciò senza aiuto, questo trattamento non dà alcuna speranza di “spiegare”

al computer come affrontare questo problema.

Intanto scriviamo il modulo della forza gravitazionale

$$F_G = \frac{GMm}{r^2}.$$

Il lettore attento avrà subito notato la scomparsa del segno meno! Poi, come nel caso del proiettile, serve il versore della forza. La forza di attrazione gravitazionale agisce lungo la congiungente di due corpi. Il vettore \vec{r}_{12} che congiunge due punti nello spazio \vec{r}_1 e \vec{r}_2 si ottiene con una semplice operazione

$$\vec{r}_{12} = \vec{r}_2 - \vec{r}_1. \quad (2.1)$$

Per mia esperienza questo problema risulta molto ostico per gli studenti. Spesso cercano complessi ragionamenti utilizzando funzioni goniometriche o processi simili.

Però è abbastanza semplice rendersi conto di come la semplice differenza tra vettori possa aiutare a risolvere questo problema. Per capire meglio l'ordine dei vettori nell'operazione di differenza si può ragionare anche così, riscriviamo la (2.1) così:

$$\vec{r}_2 = \vec{r}_1 + \vec{r}_{12}.$$

In questo maniera dovrebbe essere chiaro che il vettore \vec{r}_2 si ottiene da \vec{r}_1 aggiungendogli \vec{r}_{12} che quindi, se uno pensa alla regola del parallelogramma, proprio il vettore che parte dalla punta di \vec{r}_1 e finisce sulla punta di \vec{r}_2 .

Ora che abbiamo il vettore che va dal Sole alla Terra dobbiamo trovarne il versore usando la stessa formula del capitolo precedente, ovvero

$$\hat{r}_{12} = \frac{\vec{r}_{12}}{|\vec{r}_{12}|}.$$

Ora abbiamo modulo, versore, e considerando adesso il segno meno perché la forza deve essere attrattiva otteniamo la formula della forza che il corpo 2 sente a causa della presenza del corpo 1 come

$$\vec{F}_G = -\frac{GMm}{|\vec{r}_{12}|^2} \hat{r}_{12}. \quad (2.2)$$

Adesso sì che la formula può essere *spiegata* ad un PC!

Iniziamo però con un caso semplice, supponendo, come detto prima, il Sole fermo nell'origine e il pianeta Terra che gli orbita intorno. In questo caso abbiamo ovviamente

$$\vec{r}_{12} = \vec{r}_2; \quad \hat{r}_{12} = \hat{r}_2;$$

inoltre l'accelerazione ovviamente si ottiene dividendo la forza che subisce la Terra per la sua massa m che quindi si semplifica nella formula finale. Otteniamo quindi

$$\vec{a}_2 = -\frac{GM}{|\vec{r}_2|^2} \hat{r}_2.$$

Come sappiamo

$$\hat{r}_2 = \frac{\vec{r}_2}{|\vec{r}_2|}$$

ottenendo una formulazione alternativa che permette di evitare il calcolo del versore

$$\vec{a}_2 = -\frac{GM}{|\vec{r}_2|^3} \vec{r}_2.$$

Chiamando le componenti $\vec{r}_2 = (x, y, z)$ otteniamo per componenti

$$\vec{a}_2 = \left(-\frac{GM}{|\vec{r}_2|^3} x, -\frac{GM}{|\vec{r}_2|^3} y, -\frac{GM}{|\vec{r}_2|^3} z \right), \quad (2.3)$$

dove ovviamente

$$|\vec{r}_2|^3 = (x^2 + y^2 + z^2)^{3/2}.$$

Siamo pronti a scrivere la routine in python che calcola l'accelerazione. Dobbiamo solo calcolare il fattore GM . Ovviamente il calcolo andrebbe effettuato nel sistema internazionale, tuttavia in questo caso si può adottare un trucco (detto di riscaldamento) utilizzando unità di misura “furbe”. Se vogliamo simulare il moto della Terra sappiamo che il periodo è di un anno, l'orbita quasi circolare, con raggio $R = 1.5 \cdot 10^{11} \text{ m} = 1 \text{ u.a.}$ ovvero un'unità astronomica. Se noi prendiamo le formule per la velocità orbitale in un'orbita circolare (che per la Terra è un'ottima approssimazione almeno per il livello di trattazione che utilizziamo in questo testo)

$$\frac{GM}{R^2} = \frac{V^2}{R} \Rightarrow \omega = \frac{V}{R} = \sqrt{\frac{GM}{R^3}}$$

da cui

$$T = \frac{2\pi}{\omega} = 2\pi \sqrt{\frac{R^3}{GM}}.$$

Ora, nelle nostre unità di misura se $T = 1 \text{ anno}$ e $R = 1 \text{ u.a.}$ otteniamo che

$$GM = 4\pi^2 \text{ u.a.}^3/\text{anni}^2.$$

In questa maniera tutti i nostri calcoli numerici avranno il tempo misurato in anni e le distanze in u.a. e soprattutto permetterà la gestione di numeri ragionevolmente “piccoli” senza costringere il programma a calcoli in notazione scientifica. Tutti i linguaggi di programmazione gestiscono le potenze di 10 però diventa delicato poi evitare rischi di under- e over-flow, ovvero di numeri troppo piccoli o troppo grandi per essere gestiti da un calcolatore. ¹

Adesso siamo pronti a scrivere il codice che calcola l'accelerazione. Implementando la formula (2.3) scriveremo

```
#Definiamo l'Accelerazione

def acc(x, y, z):
    GM = 4 * math.pi * math.pi # math.pi e' pigreco
```

¹qui ci sarebbe un discorso lungo, però un esempio....

```

r = math.sqrt(x*x+y*y+z*z)
r3= r*r*r
# la funzione elevamento a potenza andrebbe
# importata ma possiamo risolvere rapidamente ricorrendo
# alla definizione stessa di potenza

return -GM*x/r3, -GM*y/r3, -GM*z/r3

# notare come i vettori riescano a compattare tutti
# i calcoli necessari

```

Adesso il programma completo sarà sulla falsariga di quelli del capitolo precedente, ovvero

```

#Sistema Sole-Terra

import math

def acc(x, y, z):
    GM = 4 * math.pi * math.pi # math.pi e' pigreco

    r = math.sqrt(x*x+y*y+z*z)
    r3= r*r*r
    # la funzione elevamento a potenza andrebbe
    # importata ma possiamo risolvere rapidamente ricorrendo
    # alla definizione stessa di potenza

    return -GM*x/r3, -GM*y/r3, -GM*z/r3

    # notare come i vettori riescano a compattare tutti
    # i calcoli necessari

# condizioni iniziali
t = 0
Tmax = 200
DT = 0.0002

x = 1
y = 0.
z = 0.

Vx = 0.
Vy = 2 * math.pi
Vz = 0.

#####

# Loop sul tempo

```

```
while t < Tmax:

    x = x + Vx*DT
    y = y + Vy*DT
    z = z + Vz*DT

    ax, ay, az = acc(x, y, z)

    Vx = Vx + ax*DT
    Vy = Vy + ay*DT
    Vz = Vz + az*DT

    t = t + DT
```

Una piccola nota meritano le condizioni iniziali. Abbiamo posto la Terra in posizione $(1, 0, 0)$ con velocità $V = (0, 2\pi, 0)$. Questa è esattamente la velocità richiesta per l'orbita circolare. In effetti per l'orbita circolare si ha che la Terra deve compiere una circonferenza di raggio 1 u.a. in un tempo $T = 1$ anno, quindi si avrà

$$V = 2\pi R/T = 2\pi \text{ u.a./anno}.$$

Abbiamo quindi lanciato la Terra affinché compisse un'orbita perfettamente circolare.

Prima di iniziare a divertirci con il nostro codice dobbiamo però verificare se stia funzionando bene. Sappiamo dalle leggi della gravitazione universale e da quelle di Keplero che le orbite sono coniche. In particolare le orbite con energia totale negativa si “chiudono”, ovvero ripassano esattamente dal punto di partenza (per essere più precisi ogni punto dell'orbita è un punto di partenza da cui la Terra deve ripassare esattamente dopo un periodo T). Inoltre sappiamo che il moto di un pianeta intorno al Sole conserva l'energia totale e il momento angolare rispetto al Sole. Noi abbiamo una simulazione numerica del moto dei pianeti e non possiamo aspettarci una conservazione *esatta*, tuttavia piccole oscillazioni numeriche ci renderebbero abbastanza sicuri che la simulazione stia funzionando ragionevolmente bene.

Intanto vediamo come calcolare l'energia totale. In realtà, poiché la massa m della Terra si semplifica nel calcolo dell'accelerazione possiamo calcolare per semplicità l'energia per unità di massa ottenendo

$$\frac{E}{m} = \frac{1}{2}V^2 - \frac{GM}{r},$$

dove (e non è banale ripeterlo)

$$V^2 = V_x^2 + V_y^2 + V_z^2$$

mentre

$$r = (x^2 + y^2 + z^2)^{1/2}.$$

Quindi due funzioni che calcolino l'energia cinetica e potenziale lungo la traiettoria saranno

```
def Ekin(Vx, Vy, Vz):
    return 0.5*(Vx*Vx + Vy*Vy + Vz*Vz)

def Epot(x, y, z):
    GM = 4 * math.pi * math.pi
    r = math.sqrt(x*x+y*y+z*z)
    return -GM/r
```

mentre nel corpo principale del programma sommeremo le due grandezze per ottenere l'energia totale (in questo modo potremmo poi stampare a schermo o su file i due termini separati). La parte principale del programma diventerà quindi

```
# Loop sul tempo

while t < Tmax:

    x = x + Vx*DT
    y = y + Vy*DT
    z = z + Vz*DT

    ax,ay,az = acc(x,y,z)

    Vx = Vx + ax*DT
    Vy = Vy + ay*DT
    Vz = Vz + az*DT

    Ep = Epot(x,y,z)
```

```
Ek = Ekin(Vx,Vy,Vz)
Etot = Ep + Ek

print(t,Ek,Ep,Ek+Ep)

t = t + DT
```

Se lanciato così come presentato il risultato è soddisfacente, l'energia fluttua ma molto poco e soprattutto non presenta tendenze costanti di crescita o decrescita.

Per far capire quanto può essere delicato fare simulazioni numeriche di buon livello vediamo ora cosa succede se facciamo una piccola modifica al codice

```
while t < Tmax:

    # stavolta calcolo l'accelerazione PRIMA di
    # aggiornare la posizione

    ax,ay,az = acc(x,y,z)

    x = x + Vx*DT
    y = y + Vy*DT
    z = z + Vz*DT

    Vx = Vx + ax*DT
    Vy = Vy + ay*DT
    Vz = Vz + az*DT

    Ep = Epot(x,y,z)
    Ek = Ekin(Vx,Vy,Vz)
    Etot = Ep + Ek

    print(t,Ek,Ep,Ek+Ep)

    t = t + DT
```

La modifica sembra innocua. Alla fine il time-step è molto piccolo e non dovrebbe cambiare molto se calcolo l'accelerazione sulla posizione attuale o su quella già aggiornata. Invece non è così! Se si fa la prova si nota come l'orbita non si chiuda perfettamente mentre l'energia lentamente aumenta.

Spiegare il perché avvenga questo è al di fuori dello scopo di queste lezioni. Solo per dare un suggerimento di lettura, il primo schema richiama il metodo di integrazione di Verlet ² mentre il secondo è il più semplice metodo di Eulero. Lo schema di Verlet viene usato proprio nelle simulazioni di sistema Hamiltoniani, ovvero di sistema conservativi come quello in esame.

D'ora in poi utilizzeremo il metodo a la Verlet per le nostre simulazioni, quindi aggiorneremo prima la posizioni e poi calcoleremo l'accelerazione per aggiornare la velocità.

Controlliamo ora la conservazione del momento angolare. Dal punto di vista numerico questa legge di conservazione è molto più facile da ottenere, tuttavia il calcolo del momento angolare è istruttivo e quindi procederemo alla verifica. Poiché abbiamo posto la posizione iniziale e la velocità nel piano $x-y$ l'unica componente del momento angolare \vec{L} diversa da zero sarà L_z . Usiamo la scrittura con i versori per calcolarla ricordando che

$$\hat{x} \times \hat{y} = \hat{z}, \quad \hat{y} \times \hat{x} = -\hat{z}$$

$$\hat{x} \times \hat{x} = \hat{y} \times \hat{y} = 0$$

otteniamo

$$\begin{aligned} \frac{1}{m} \vec{L} &= \vec{r} \times \vec{V} = (\hat{x}x + \hat{y}y) \times (\hat{x}V_x + \hat{y}V_y) \\ &= \hat{z}(xV_y - yV_x). \end{aligned}$$

Ed ecco di conseguenza il codice per calcolare L_z

```
# calcolo della componente z del momento angolare

def L_zt(x, y, Vx, Vy):
    return x*Vy - y*Vx
```

E' facile notare lanciando il programma come questa grandezza sia conservata con un'ottima precisione.

²https://en.wikipedia.org/wiki/Verlet_integration

La serie di esperimenti che si possono fare già con questo codice è lunghissima. Per ora mostriamo cosa succede se incrementiamo la velocità iniziale dal valore $V_y = 2\pi$ al valore 4π lasciando tutto il resto invariato.

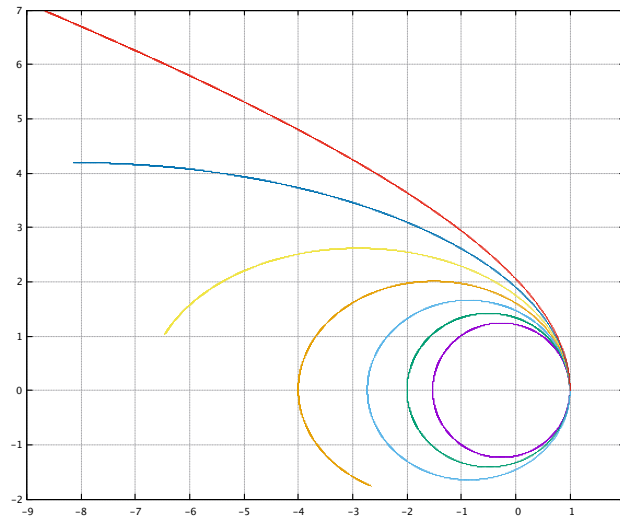


Figura 2.1: Orbite ottenute modificando la velocità iniziale da 2π a 3π .

Per ottenere in automatico le differenti orbite si può procedere in molti modi. Per esempio basta circondare la parte del codice contenente le condizioni iniziali e il ciclo sul tempo con

```
# ciclo for con le diverse velocita- iniziali

for kv in range(20,28):

    # condizioni iniziali

    T = 0
    Tmax = 3
    DT = 0.001

    x = 1
    y = 0.
    z = 0.

    VV = 2 * math.pi * kv /20.

    .... # qui va tutto il loop sul tempo
```

Un risultato interessante è sicuramente il calcolo della velocità di fuga. Come sappiamo essa si calcola ponendo uguale a zero l'energia totale (sempre per unità di massa), quindi

$$\frac{1}{2}V^2 - \frac{GM}{R} = 0 \Rightarrow V = \sqrt{2\frac{GM}{R}} = \sqrt{2}V^*$$

dove V^* è la velocità dell'orbita circolare. Dovremmo quindi osservare che quando la velocità nelle nostre unità supera il valore soglia di $2\sqrt{2}\pi$ il pianeta non torna più indietro seguendo un'orbita parabolica.

Una proprietà più difficile da verificare sarebbe la verifica della terza legge di Keplero nel caso di orbite ellittiche. Ricordiamo che la terza legge di Keplero dice che

$$T^2 \propto a^3$$

dove a è il semiasse maggiore dell'orbita. Nel caso di orbite circolari la dimostrazione è alquanto facile. Diverso è il caso di orbita ellittica. Si possono seguire due vie, integrare l'equazione completamente come fece Newton dopo essersi inventato l'analisi infinitesimale oppure ragionare sulla costanza della velocità areale.

E numericamente come possiamo fare? Il periodo e il semiasse maggiore non sono di immediata determinazione, però possiamo fare un paio di considerazioni preliminari. Prima di tutto se partiamo con la velocità perpendicolare al vettore Sole-Terra con velocità iniziale maggiore di V^* avremo che l'istante $t = 0$ corrisponde al perielio, istante di minima distanza. L'afelio sarà il primo punto incontrato lungo l'orbita in cui avremo di nuovo che \vec{r} e \vec{V} sono perpendicolari.

Eccoci di nuovo davanti ad un altro problema che spesso spiazza gli studenti non familiari con i vettori. Come si fa a capire se due vettori sono perpendicolari? Normalmente uno studente inizia a cercare il modo di calcolare l'angolo fra i due vettori con i soliti metodi

complicati basati sulle funzioni goniometriche inverse quando invece c'è una scorciatoia facilissima: il prodotto scalare. Sappiamo che il prodotto scalare fra due vettori è nullo quando i due vettori sono perpendicolari. Ripetendo lo stesso ragionamento fatto con il momento angolare e il versore ricordando che

$$\hat{x} \cdot \hat{x} = \hat{y} \cdot \hat{y} = 1.$$

$$\hat{x} \cdot \hat{y} = 0$$

abbiamo che

$$\vec{r} \cdot \vec{V} = xV_x + yV_y + zV_z.$$

Ragionando sul significato geometrico del prodotto scalare possiamo notare che

$$\vec{r} \cdot \vec{V} > 0$$

quando la Terra si allontana dal Sole, mentre

$$\vec{r} \cdot \vec{V} < 0$$

la Terra si sta avvicinando al Sole. Numericamente è impossibile trovare l'istante in cui $\vec{r} \cdot \vec{V} = 0$ però possiamo registrare lungo l'orbita quando il prodotto scalare cambia segno e memorizzare il tempo a cui questo avviene. Questo tempo sarà con ottima approssimazione la metà del periodo dell'orbita. Per trovare il semiasse a basta notare come la somma di perielio e afelio sia pari a due volte il semiasse maggiore a .

Dobbiamo quindi aggiungere nel ciclo sul tempo le seguenti istruzioni

```
# calcolo di periodo e semiasse maggiore
# individuando perielio o afelio

# calcolo il prodotto scalare fra r e V
ps = Vx * x + Vy * y + Vz * z

# se il prodotto fra ps e il vecchio prodotto
# scalare psold e' negativo vuol dire che sono
# discordi e il prodotto scalare nell'intervallo
# Dt e' stato zero.
```

V_0 (u.a./anno)	Afelio (u.a.)	T (anni)
6.28	1.53	1.42
6.44	1.74	1.60
6.60	2.00	1.84
6.75	2.32	2.14
6.91	2.73	2.55
7.07	3.27	3.11

Tabella 2.1: Velocità iniziale V_0 , Afelio (il Perielio è 1 u.a. per come è effettuato il calcolo numerico), periodo T per la verifica della terza legge di Keplero.

```

if ps*psold <= 0:
    print("r = ", math.sqrt(x*x+y*y+z*z), " t = ", T)
psold = ps

```

il trucco e' cercare l'istante in cui il prodotto scalare moltiplicato per il prodotto scalare al tempo precedente è negativo o nullo. Questo vuol dire che il prodotto scalare è zero o è stato zero nel time-step Dt . Alla variabile *psold* viene poi assegnato, dopo la condizione *if*, il valore *ps* appena calcolato. *psold* va inizializzato a zero prima del loop sul tempo.

Nella tabella 2.1 riportiamo come esempio i risultati ottenuti utilizzando il nostro codice:

Riportando i risultati in figura si vede come l'accordo con la legge di Keplero

$$T \propto a^{2/3}$$

sia ottimo. La costante di proporzionalità ricavato da un best fit sui dati.

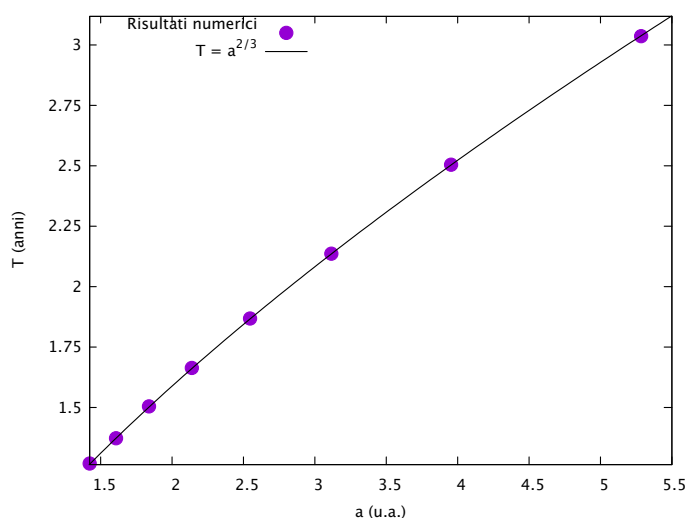


Figura 2.2: I dati della Tabella 2.1 riportati in grafico con il best fit (linea continua) che mostra l’ottimo accordo con la legge terza legge di Keplero.

La lista di tutti i possibili “esperimenti” effettuabili con questo codice è davvero lunga, e probabilmente molti ne verranno in mente al lettore curioso. Mi limito qui solamente a proporne uno a mio avviso molto interessante.

La prima legge di Keplero è forse la più sottovalutata. Dice che le orbite sono coniche, quindi quelle chiuse sono ellissi. Sembra un risultato banale ma non lo è affatto. Dire che le orbite sono delle ellissi vuol dire che le orbite si chiudono, ovvero passano sempre sugli stessi punti senza riempire tutta la regione del piano che sarebbe invece compatibile per la conservazione dell’energia e del momento angolare. Ma perché questo succede?

In realtà la spiegazione reale sta nel fatto che la forza di Newton ha un’altra grandezza conservata, il vettore di Lenz³. Discutere questo aspetto è ovviamente al di là delle nostre intenzioni, tuttavia è molto semplice rendersi conto di quale sia l’ingrediente fondamentale per la chiusura delle orbite. E’ l’esponente 2 al denominatore della forza

³https://it.wikipedia.org/wiki/Vettore_di_Lenz

di Newton. Se invece di avere

$$F \propto \frac{1}{r^2}$$

avessimo

$$F \propto \frac{1}{r^\alpha}$$

con $\alpha \neq 2$ osserveremmo un comportamento oltremodo differente. Credo che ormai a questo punto il lettore abbia già abbandonato queste pagine e stia già provando sul codice.

Basta porre nella routine dell'accelerazione qualcosa di questo tipo

```
def acc(x, y, z):  
    #Definiamo l'Accelerazione  
    r = math.sqrt(x*x+y*y+z*z)  
  
    r3 = math.pow(r,3.2) # una potenza diversa!  
  
    return -gm*x/r3, -gm*y/r3, -gm*z/r3
```

e vedere cosa succede! Come si può ammirare, la traiettoria spiraleggia riempiendo uniformemente la regione di piano compatibile con le due leggi di conservazione. Il raggio minimo e massimo della corona circolare si possono calcolare supponendo che la velocità radiale sia nulla e impostando l'equazione della conservazione dell'energia.

2.2 Il problema dei tre corpi

Il problema dei tre corpi apre un mondo completamente diverso. Non esiste soluzione analitica in questo caso, a parte casi molto molto particolari. L'unica soluzione anche per i fisici che hanno completato il corso di studi universitario è ricorrere al calcolo numerico.

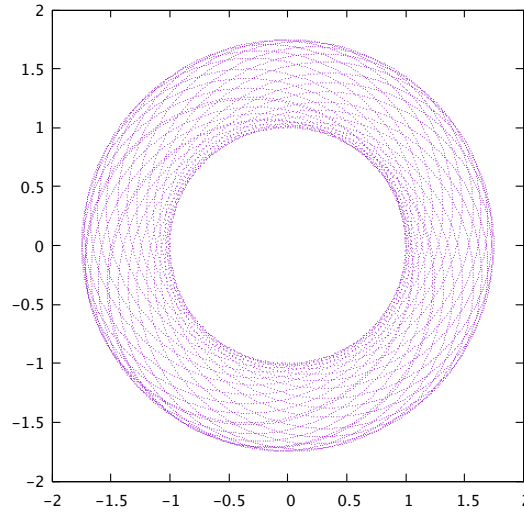


Figura 2.3: Traiettorie ottenute ponendo $\alpha = 2.2$. Notare come l'orbita non si chiude e riempie uniformemente una corona circolare.

Ovviamente in quel caso lo si farà con maggiore attenzione agli schemi di integrazione, tuttavia è interessante vedere cosa si può ottenere con i nostri mezzi!

Prima di tutto, e dal punto di vista didattico forse è la cosa più importante, bisogna scrivere il codice. Nello spirito di mantenere le cose abbastanza semplici possiamo supporre che una massa centrale molto grande si trovi ferma nell'origine mentre altre due masse più piccole si muovono nel piano subendo l'attrazione sia della massa centrale che della più piccola che si muove.

Il punto di partenza è l'equazione ricavata nella sezione precedente

$$\vec{F}_G = -\frac{GMm}{|\vec{r}_{12}|^2} \hat{r}_{12}.$$

Definiamo i vettori posizione delle due masse mobili m_1 e m_2

$$\vec{r}_1 = (x_1, y_1, z_1),$$

$$\vec{r}_2 = (x_2, y_2, z_2),$$

e scriviamo la procedura che calcola la forza (non l'accelerazione questa volta) che m_2 subisce a causa di m_1

```
gm = 4 * math.pi * math.pi
m1 = 0.05
m2 = 0.07

def F12(x1,y1,z1, x2,y2,z2,):

    # Calcolo il vettore differenza da 1 -> 2

    dx = x2 - x1
    dy = y2 - y1
    dz = z2 - z1

    r12 = math.sqrt( dx*dx + dy*dy + dz*dz )

    r12-cub = r12*r12*r12

    # cosi' ottengo la forza che m2 senta da parte di m1

    Gm1m2 = gm * m1 * m2

    return -Gm1m2 * dx / r12-cub, -Gm1m2 * dy / r12-cub,\
        -Gm1m2 * dz / r12-cub
```

Va notato come, nelle unità da noi scelte, m_1 e m_2 sono il rapporto fra la reale massa dei due “satelliti” e la massa centrale.

Ricordandoci che, grazie al terzo principio della dinamica si ha

$$\vec{F}_{12} = -\vec{F}_{21}$$

e che, invece, grazie al secondo principio

$$m_2 \vec{a}_{12} = -m_1 \vec{F}_{21}.$$

Quindi potremo scrivere semplicemente

$$\vec{a}_{12} = \frac{1}{m_2} \vec{F}_{12}$$

e

$$\vec{a}_{21} = -\frac{1}{m_1} \vec{F}_{12}.$$

La parte principale del codice diverrà quindi

```

# condizioni iniziali

T = 0
Tmax = 50
DT = 0.001

x1 = 1; y1 = 0.; z1 = 0.
x2 = -2; y2 = 0.; z2 = 0.

V1x = 0; V1y = 2 * math.pi; V1z = 0.
V2x = 0; V2y = -1.414 * math.pi; V2z = 0.

#####

# Loop sul tempo

k = 0

while T < Tmax:

    x1 = x1 + V1x*DT; y1 = y1 + V1y*DT; z1 = z1 + V1z*DT
    x2 = x2 + V2x*DT; y2 = y2 + V2y*DT; z2 = z2 + V2z*DT

    a1x,a1y,a1z = acc(x1,y1,z1)
    a2x,a2y,a2z = acc(x2,y2,z2)

    f12x, f12y, f12z = F12(x1,y1,z1, x2,y2,z2)

    a1x = a1x - f12x / m1; a1y = a1y - f12y / m1; a1z = a1z -
        f12z / m1
    a2x = a2x + f12x / m2; a2y = a2y + f12y / m2; a2z = a2z +
        f12z / m2

    V1x = V1x + a1x*DT; V1y = V1y + a1y*DT; V1z = V1z + a1z*DT
    V2x = V2x + a2x*DT; V2y = V2y + a2y*DT; V2z = V2z + a2z*DT

    T = T + DT

    if k%5 == 0:
        Win.plot(x1,y1,"white")
        Win.plot(x2,y2,"red")

    k = k + 1

```

Se già con due corpi gli esperimenti possibile erano tanti con tre non c'è praticamente modo di elencarli tutti. Qui ne proponiamo due supponendo che la fantasia del lettore potrà aggiungerne (e perché no, suggerire all'autore) tanti altri.

Il primo esempio proposto deriva dalla lettura di un bellissimo quaderno de Le Scienze dei primi anni '90 dedicato alle missioni Voyager. Le due sonde (che sono ancora funzionanti!) fotografarono per la prima volta gli anelli di Saturno da molto vicino notando una struttura a microsolco con zone degli anelli privi di micro-satelliti. Queste zone si scoprì che erano corrispondenti alle orbite di satelliti un po' più grandi degli altri che spazzavano via gli altri micro-satelliti. Questo effetto, ad un neofita quale io ero ai tempi, risultò alquanto sorprendente. Ad una prima analisi sembrerebbe che due satelliti in orbite circolari a causa dell'attrazione reciproca siano portati ad avvicinarsi. Ma questo è falso! L'interazione è più complessa e avviene con i due corpi a velocità diverse. Quello che succede è molto simile all'effetto fionda, meccanismo utilizzato proprio dalle sonde Voyager per raggiungere i confini del sistema solare.

Facciamo ora l'esperimento ponendo le seguenti condizioni iniziali

```
.....
gm = 4 * math.pi * math.pi
m1 = 0.001
m2 = 0.04

.....

# condizioni iniziali

T = 0
Tmax = 5
DT = 0.001

x1 = 1; y1 = 0.; z1 = 0.
x2 = -2; y2 = 0.; z2 = 0.

V1x = 0; V1y = 2 * math.pi; V1z = 0.
V2x = 0; V2y = -1.414 * math.pi; V2z = 0.
```

Nella seconda figura il ruolo delle due masse è scambiato semplicemente ponendo

```
m2 = 0.001
m1 = 0.04
```

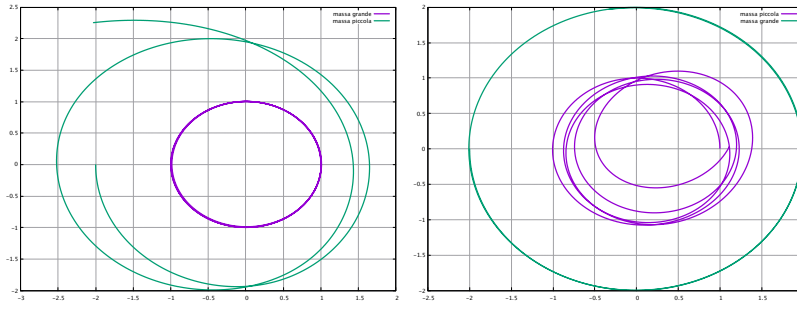


Figura 2.4: Traiettorie ottenute ponendo la massa più grande prima all'interno e poi all'esterno. In entrambi i casi si nota come la massa più piccola si allontana da quella più grande.

Passiamo ora al secondo esperimento. Vediamo di simulare il sistema Sole-Terra-Luna. Non è un esercizio così banale poiché dobbiamo inserire le condizioni iniziali corrette. Prima di tutto le masse. m_1 e m_2 saranno i rapporti fra le masse di Terra e Luna e quella del sole, ovvero

$$m_1 = \frac{M_\oplus}{M_\odot} \simeq 3.0 \cdot 10^{-6}$$

e

$$m_2 = \frac{M_\text{D}}{M_\odot} \simeq 3.7 \cdot 10^{-8}.$$

Per quanto riguarda le posizioni, la distanza media Terra-Luna R_2 è

$$R_2 = 2.574 \cdot 10^{-3} \text{ u.a.}$$

Per la velocità iniziale bisogna stare attenti. Supponiamo per semplicità che la Luna orbiti intorno alla Terra che sta ferma (il rapporto delle masse non è così sbilanciato come nel caso Terra-Sole, questo punto andrebbe discusso meglio per un calcolo più preciso). Il periodo della Luna è di 28 giorni, quindi in anni avremo che la velocità sarà

$$V_\text{D} = 2\pi \frac{R_2}{T} \simeq 2\pi R_2 \frac{28}{365} \simeq 0.21 \text{ u.a./anno.}$$

Questa però è la velocità rispetto alla Terra. Per ottenere quella rispetto al Sole dobbiamo aggiungere a questo valore la velocità orbitale della Terra rispetto al Sole. Il nostro codice quindi dovrà contenere le seguenti righe

```
.....
gm = 4 * math.pi * math.pi
m1 = 3.0e-6
m2 = 3.7e-8

.....

# condizioni iniziali

T = 0
Tmax = 5
DT = 0.0001

R2 = 2.574e-3 # distanza Terra-Luna

x1 = 1; y1 = 0.; z1 = 0.
x2 = 1+R2; y2 = 0.; z2 = 0.

V1x = 0; V1y = 2 * math.pi; V1z = 0.
V2x = 0; V2y = V1y + 0.21; V2z = 0.
```

Non è facile graficare in maniera chiara le due orbite a causa della notevole sproporzione fra i due raggi orbitali, mostriamo qui un ingrandimento delle orbite. Una cosa interessante che si può notare è che la Luna, rispetto al Sole, non effettua mai moto retrogrado ma si muove sempre nello stesso verso (antiorario visto dal polo Nord) della Terra.

Per il lettore più curioso si potrebbe poi pensare di simulare il comportamento di una piccola massa nei punti di Lagrange della Terra. I punti di Lagrange⁴ sono quei punti in cui due corpi dotati di grande massa, tramite l'interazione della rispettiva forza gravitazionale, consentono a un terzo corpo, dotato di massa molto inferiore, di mantenere una posizione stabile relativamente ad essi. Attraverso il nostro codice è pure possibile osservare quali punti sono stabili e quali no semplicemente facendo partire la massa piccola in una posizione leggermente diversa dai punti di Lagrange.

⁴https://it.wikipedia.org/wiki/Punti_di_Lagrange

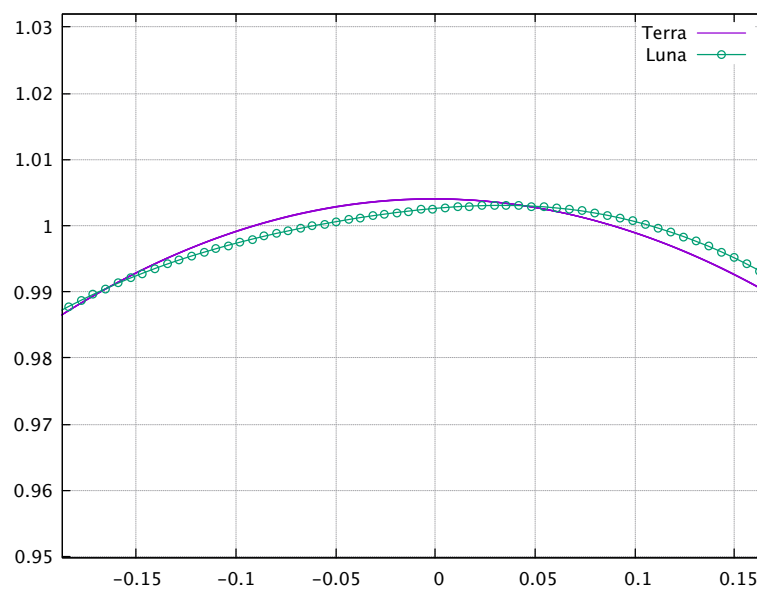


Figura 2.5: Un particolare delle orbite di Terra e Luna.