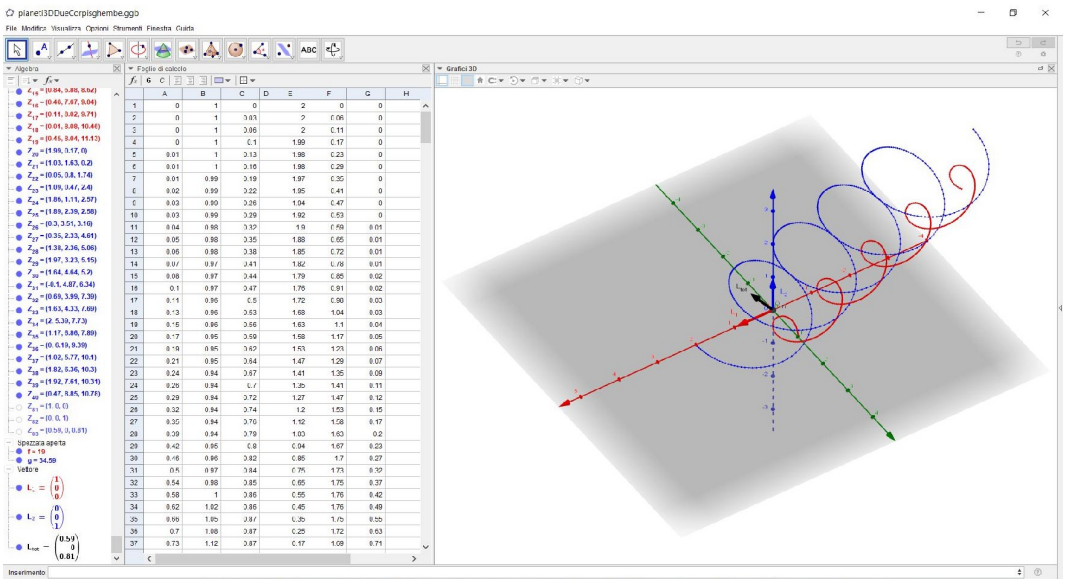
Liceo Cantonale di Bellinzona

Corso di Aggiornamento - 5 maggio 2026

Percorsi di Fisica FAM (Monte Ore 2024-25)

Idee sul problema dei 2-3 corpi

T. Corridoni F. De Vito



Versione 2026-05-05

Indice

Introduzione

0.1 Storia e motivo di questo lavoro

La scoperta, la ricerca e lo studio degli esopianeti hanno riportato il problema delle orbite di un sistema di N corpi all'attenzione di un pubblico di studenti e studentesse più vasto di quanto si pensasse. In effetti, è difficile non rimanere affascinati da Fig. 1, dove si riporta la recente scoperta di un esopianeta che ruota in un piano ortogonale a quello in cui si svolge l'orbita delle sue due stelle, a loro volta rotanti una attorno all'altra [1, 2]. *Ma una volta acceso il fascino, cosa e come far costruire ad una persona inter-essata alla fisica del problema, affinché non si spenga* ?

Questo testo, basato su un singolo caso studio [3], nasce come risposta parziale a questa domanda. Ma per capirne la motivazione più che la storia (e poi il metodo e la struttura) si parta da un *gioco* proprio su Fig. 1. La si osservi per un po' e ci si chieda *cosa rappresenta*, rispondendo a se stessi/e con disegni, schemi, parole, equazioni.... Si passi poi a raccogliere ed analizzare le risposte che danno altre persone, colleghi/e, allieve/i, familiari, amici... che trovino la cosa interessante (ma solo per avere dati significativi). Ci si accorgerà allora che chiunque abbia studiato fisica, astronomia, matematica, ingegneria, chimica... finirà per dire qualcosa di riconducibile al *problema degli N corpi*, siano essi pianeti o particelle, mentre gli *altri* costruiranno spiegazioni organizzate in modi non sempre evidenti, perché evocative, ricche di analogie ed immagini mentali raccolte da stereotipi di diversa origine, anagrafica e culturale: la scuola o i libri per le persone "'anziane"', serie cinematografiche, siti internet, videogiochi, AI e social per i "'giovani"'.

Forse un/a fisico/a, un/a matematico/a... di professione troverebbe tutto ciò interessante ma fine a se stesso. Un/a *docente di* fisica ecc. dovrebbe invece riconoscere nel *gioco* una questione molto importante nella didattica delle materie (non solo) scientifiche: *chi cerca di apprendere qualcosa non lo vede come chi lo ha già appreso; a causa di ciò e dell'evolvere delle sorgenti e dei processi di organizzazione della cultura, problemi risolti anche da tempo da una comunità di esperti/e sono continuamente riscoperti e rilanciati dai non esperti/e in vesti sempre nuove.* Si tratta di *un'evoluzione* del problema individuato da Hanson attorno al 1950 [4], riassunto nella domanda giocosa:

> *Tycho Brahe e Johannes Kepler (che ha letto Copernico) osservano il sorgere del Sole: vedono la stessa cosa ?*

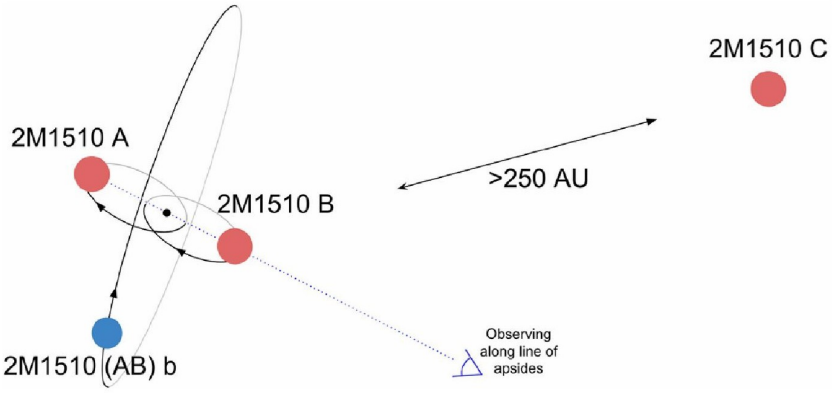


Fig. 5. Configuration of 2M1510 and naming convention for the various bodies. Brown dwarfs are in red, and the planet is in blue. Direction to Earth relative to the binary is shown.

Figura 1: L'esopianeta 2M1510(AB)b ruota in un'orbita ortogonale al piano su cui si muovono le due *brown dwarfs* 2M1510 A e B. È il primo sistema di questo tipo mai osservato [1, 2].

Hanson partiva da ciò per arrivare a quanto sarebbe stato poi approfondito da Kuhn: l'evoluzione di un sapere scientifico deve accompagnarsi ad un *cambiamento di paradigma* nella comunità degli esperti [5]. Ma qui non si tratta del solo problema "'per addetti ai lavori"', bensì *del problema didattico*, nel quale c'è appunto un'*evoluzione*: l'*apprendimento significativo, almeno in campo scientifico, richiede un orientamento all'interno di una pluralità di paradigmi, perché generazione dopo generazione i problemi ritornano in forme apparentemente diverse che richiedono trattazioni molteplici e con strumenti diversi.* In breve: le materie evolvono, gli studenti cambiano ma i problemi tornano.

A causa di ciò è da almeno 30 anni che un/a docente non può più essere qualcuno che, volendo esagerare, *trasmette il sapere* e per verificarne la *ricezione* (radiofonica) fa per tutta la sua vita lavorativa le stesse domande di cui sa già la risposta. Se non vuole essere sostituito da AI, influencer e siti "'dedicati"' ai problemi che ritornano (si pensi al terrapiattismo) [6], non può utilizzare solo *lezioni*, ossia *letture* in cui *s-piega* (come un foglio prima piegato) una *teoria*, ossia una *fila* di passaggi logici. Questa è *una delle forme*, adatta forse per *in-segnare* un singolo filo ma non l'intera *trama concettuale* di una materia [7], che non è infatti solo *complicata*, ossia *piegata* come il suddetto foglio, ma *complessa*, ossia *intrecciata*, formata da più fili [8]. Tale analogia da tessitore, oggi poi non basta neanche più, perché *si è visto che oltre un certo grado di complessità i saperi non possono più essere linearizzati, districati in "'parti più piccole"', come appunto fili di una trama*, e sono pertanto appresi secondo *livelli di competenza* più o meno discontinui, insiemi irriducibili di *sapere, saper essere e saper fare* (come si dice in pedagogia) difficilmente costruiti da una persona che ascolta solamente un qualche trasmettitore, umano o elettronico. Non a caso la pratica essenziale del *laboratorio*, che immerge *operativamente* una persona in una certa trama concettuale, mediante esperienza diretta, nelle scuole è stata estesa dalle scienze sperimentali alla matematica, alle lingue, al teatro.

In breve e senza retorica, occorre insomma riconoscere che il problema del come un/a docente possa rispondere al *ritorno dei problemi* non richiede una riflessione ideologica ma epistemologica, perché a ben vedere... c'è sempre stato. Chi insegna sa infatti che ad un/a docente sono sempre più richieste competenze molto particolari:

1. *deve (avere piacere di)*[1] *sapere la materia, e saperla in più modi.* Esempi più o meno ingenui: il teorema di Pitagora è dimostrato in centinaia di modi; il concetto di massa in fisica subisce vere e proprie metamorfosi; i principi della dinamica risultano essere la forma differenziale dei principi di conservazione...;

2. *deve (avere piacere di) saper ricostruire e far ricostruire le trame concettuali (o anche i singoli "'fili"') di una o più materie (in ottica interdisciplinare), in uno o più paradigmi*, dove a questo punto occorre definire brevemente cosa sia un paradigma (per dettagli e referenze, [10]). In base a studi avviati nel 1900, *l'apprendimento delle scienze avviene facendo evolvere delle "'concezioni"' in "'modelli"' mediante "'ridescrizione rappresentazionale"'*. Per un "'tecnico"' della scienza può definirsi *modello* la *rappresentazione in un linguaggio logico della struttura delle relazioni fra un insieme di parti, siano esse oggetti concreti o astratti* [11]. La maggior parte dei fenomeni può non a caso raccontarsi mediante una lingua che definisca cose ed eventi, con un livello di astrazione che aumenta con la precisione e la coerenza logica del linguaggio: il disegno di un oggetto attaccato ad una molla è passibile di interpretazione, l'equazione di un *oscillatore armonico semplice* no. Proprio questa evoluzione, solitamente data per scontata da uno scienziato, è invece accuratamente studiata da psicologi e pedagogisti, alcuni dei quali preferiscono definirla *ridescrizione rappresentazionale* perché i modelli costruiti da un bambino sono emotivi, personali, meno organizzati, condivisibili e coerenti di quelli di un adulto (che infatti è meno incline a ridefinirli), e vengono pertanto detti *concezioni o rappresentazioni*. La ridescrizione di queste rappresentazioni è appunto *l'insieme dei processi di riorganizzazione delle concezioni che la persona che apprende svolge per arrivare a poterle condividere, diventati modelli, con una comunità di esperti*. La persona parlerà ad esempio di *modello degli N corpi* solo una volta arrivata a formalizzare in tal modo riconoscibile e riconosciuto le sue concezioni. Ora, la domanda è: *si può arrivare a più modelli di uno stesso fenomeno ?* La risposta è sì: in fisica è ben noto come a seconda di quali elementi, relazioni e linguaggi si scelgano, la luce possa descriversi come *un'onda* o come un *gas di fotoni*, due modelli completamente diversi che per essere compresi richiedono processi di riorganizzazione rappresentazionale, e quindi di insegnamento, completamente diversi. Ebbene, *possiamo definire paradigma l'insieme di elementi, relazioni e linguaggi (nonché, volendo, dei processi di apprendimento) che generano un tipo di modello di un fenomeno.* Proprio grazie al ritorno dei problemi, con l'evoluzione del sapere diversi paradigmi possono diventare casi particolari di teorie unificate;

[1] La parentesi sottolinea come il "'dovere"', implicito nei profili di competenza della figura professionale del docente [9], sia per il/la professionista in realtà "'piacere di fare"', cosa che rende l'insegnamento un *mestiere*, più che un *impiego*.

3. *deve (avere piacere di) saper comunicare quanto sa non per trasmettere, ma per innescare l'apprendimento*, con nuovi mezzi, nuovi linguaggi, estendendo in questo stesso modo quanto sa, che non è statico, ma si evolve anche solo nel linguaggio;

4. *deve (avere piacere di) saper sostenere percorsi di apprendimento collettivi ed individuali definiti nel tempo in cui vive*. Il teorema di Pitagora non è "'di Pitagora"' ma di chi lo sa usare per far evolvere una sua linea di pensiero. Tuttavia, *un apprendimento individuale senza cambiamento di paradigma collettivo produce concezioni magari raffinatissime ma non modelli condivisi*. Si pensi alla relatività di Einstein: prima che il calcolo differenziale assoluto di Ricci-Curbastro (in sostanza la definizione di cosa sia una metrica) diventasse uno strumento di pensiero condiviso in grado di innescare un cambiamento di paradigma, le critiche alle *inutili complicazioni* matematiche della relatività sconfinavano quasi sistematicamente nell'attacco personale [12];

5. *deve (avere piacere di) fare dell'apprendimento altrui (e proprio) l'oggetto del suo studio scientifico*, così come ha studiato fenomeni e teorie proprie della materia che insegna, dovendo ovviamente aggiungervi lo studio degli aspetti emotivi e psicologici. Tale pratica è fondamentale sia per *far evolvere le proprie competenze di insegnamentento*, sia per *costituire un'etica professionale riconosciuta in un ruolo sociale*, che sappia rispondere alle domande del consesso civile, spesso riconducibili alla differenza fra *processo* e *prodotto* dell'insegnamento. Quando viene sollevato ad esempio il problema di *come valutare* le competenze, chi tutti i giorni studia i fenomeni di apprendimento sa benissimo che *il problema non è valutarle in uscita*, alla fine di un percorso, cosa immediata quando va tutto bene, ma semmai *innescarne lo sviluppo in entrata*, all'inizio di unità didattiche ramificate in laboratorii, lezioni, esercitazioni (si pensi al tema della *pressione*, o a quello delle *equazioni differenziali*);

6. *deve (avere piacere di) riconoscere l'insieme di tutto quanto detto su di sé, preparandosi ad imparare per tutta la vita in continuità con scienziati/e, colleghi/e, studenti e studentesse*, condividendo e costruendo con loro paradigmi, trovandosi ad essere spesso Brahe quando pensava di essere Kepler. Solo superando lo smarrimento che ciò può comportare avrà la fortuna di scoprire che il ritorno dei problemi non è altro che la fortunatissima modalità mediante la quale quanto ha costruito per anni viene riorganizzato da chi apprende, che lo porterà avanti assieme a lui/lei e si troverà nei suoi stessi panni nel giro di una generazione (o forse molto prima, data la velocità con cui oggi evolve l'informazione). Perché, appunto, il sapere scientifico evolve in un vasto processo di selezione condivisa collettivamente, e reinterpretare sotto un'altra luce qualcosa che si pensava compreso è fonte spesso di entusiasmanti scoperte, sulla materia, sull'apprendimento in generale, su di sé.

Nel Liceo ticinese, il contesto didattico nel quale il/la docente può fare al meglio quanto elencato è il Lavoro di Maturità (LAM). Dopo tutto quanto è stato detto si ripeta allora il *gioco* di Hanson mettendo un docente di fisica al posto di Brahe e uno studente di Fisica e applicazioni della matematica (FAM) al posto di Kepler:

T.C. e F.d.V. (che nel suo LAM vuole simulare il Sistema Solare al PC) osservano Fig. 1: vedono la stessa cosa ?

Ovviamente no, e le tante parole spese finora non servono certo per rispondere in tal modo né alla domanda di Hanson né a quella di adesso, ma a riconoscere che tale differenza nel cosa e come vedere non è un problema, ma un'opportunità di ricerca.

0.2 Sviluppo e struttura di questo lavoro

Il nucleo iniziale di questo testo è un insieme di trattazioni didattiche sviluppate fra il 2019 ed il 2023 con classi di seconda FAM (quando i docenti di fisica sono stati chiamati ad insegnare informatica), quarta FAM o lavori LAM [3], nelle quali si affrontava il problema dei due corpi in ambito gravitazionale, elettrostatico (l'esperimento di Rutherford) fino ad incursioni nel campo dell'elettrolocazione. L'idea era di riprendere, riorganizzare ed estendere quanto sviluppato per andare verso il problema degli N corpi: il *sapere minimo necessario* ad un docente che *progetti una trasposizione didattica* di parte dei materiali, a sua scelta e nel modo che riterrà migliore. Così com'è, l'intero testo non è assolutamente adatto per un corso base, e andrebbe certamente cambiato anche per un corso FAM. Potrebbe invece già essere utile nello sviluppo di un altro LAM di studenti e studentesse particolarmente affascinati dai problemi emersi. Infatti, l'incursione del punto di vista della *programmazione per oggetti* nell'organizzazione logica del problema di Kepler ha stravolto del tutto il percorso classico, evidenziandone nodi concettuali generalmente trascurati che hanno suggerito la successione degli argomenti nel Cap. 1:

1. già nel problema dei due corpi, l'utilizzo della *grafica 3D*, accessibile e soddisfacente, ha spinto verso *sviluppi matematici più geometrici che algebrici*. Al classico problema differenziale in coordinate polari, pur introdotto (Par. 1.1), è stata preferita una *trattazione prevalentemente vettoriale* a partire dagli invarianti del moto (Par. 1.1.2). C'è stata quindi fin da subito una *intrinseca coesistenza fra paradigmi*;

2. ciò ha portato ad utilizzare subito *il vettore di Runge-Lenz, che si è quindi dovuto "'riscoprire"' nella trattazione differenziale*, dove non viene evidenziato in quanto è consuetudine impostare il problema di Kepler in un riferimento in cui le orbite coniche vengono alla fine automaticamente riferite agli assi. *La simulazione al calcolatore, al contrario, spinge verso la scelta di condizioni iniziali e riferimenti del tutto arbitrarii*, rendendo l'approccio differenziale più difficile, ma estendendo significativamente il valore dei suoi risultati;

3. dalla *sovrapposizione* fra il paradigma in cui il problema dei due corpi è *di tipo differenziale, basato sui principi della dinamica* (Par. 1.1), e quello in cui è *di tipo integrale, basato sui principi di conservazione* (Par. 1.1.2), è emerso un aspetto che diventa evidente solo dalla teoria di Bohr-Sommerfeld: *i problemi differenziali portano ad equazioni orarie*, soluzioni parametriche nel tempo che i ragazzi conoscono fin

dai moti rettilinei; *l'approccio basato sugli invarianti porta invece a luoghi geometrici*, traiettorie dove il tempo scompare (come in meccanica quantistica), risultato nuovo per uno studente;

4. il punto precedente apre possibilità interdisciplinari di storia della fisica e della matematica: lo stesso Kepler, che partì dalla forma delle orbite, determinata sperimentalmente, si trovò a dover ricostruire l'equazione oraria del moto dei pianeti partendo dalle proprietà dell'ellisse (Par. 1.1.3; per una recente trattazione didattica, [13]), risolta numericamente con l'aiuto del matematico-orologiaio svizzero Jost Bürgi;

5. l'insieme degli ostacoli finora elencati ha suggerito la strategia per il resto del lavoro: *cambiare riferimento per riuscire a vedere quanto previsto dai modelli* (Par. 1.1.4). Ciò è avvenuto praticamente in tre fasi:

   - inizialmente si è avuta la sorpresa di scoprire che passando dal riferimento in cui uno dei due corpi è fermo, in genere non inerziale, ad un qualunque sistema inerziale, i due corpi possono inseguirsi su spirali ellittiche conservando gli invarianti previsti. Ciò ha fornito la ragione per introdurre il sistema inerziale del Centro di Massa (CdM), come si fa solitamente in gravitazione (Par. 1.1.4);
   - la seconda fase è nata da una domanda molto più interessante: *la caratterizzazione della forma delle orbite al variare di energia e momento angolare è stata trovata nel riferimento non inerziale in cui uno dei due corpi è fermo. Come si estende al sistema del CdM, in cui nel caso generale entrambi i corpi sono in moto ?* Tale problema viene sempre "'liquidato"' introducendo la distanza relativa e la massa ridotta. Tuttavia didatticamente ciò non funziona, perché *non viene riconosciuto il trasferimento del modello già costruito, che vuole un punto fermo ed uno in moto.* Quello che occorre chiedersi è: *quale massa efficace devo assegnare al CdM dei due corpi, affinché ciascuno di essi veda in esso solo un "'sole"' fermo, ed abbia quindi un'orbita descritta dal modello già visto ?* La risposta è stata dimostrata, evidenziando un aspetto della geometria delle orbite mai (forse) menzionato: sono omotetiche rispetto al CdM (Par. 1.1.4);
   - la terza fase è nata nello studio del problema dei 3 corpi: prima si è scoperto che il riferimento sinodale (Par. 1.1.4), data la geometria del problema a due soli corpi, è definibile sia nel caso ideale, sia in quello reale, poi che è comunque raggiungibile mediante una doppia rotazione nello spazio (Par. 1.1.5), complessa in termini differenziali, ma molto diretta ed efficace nella grafica 3D;

6. raggiunto il problema dei 3 corpi (Cap. 3), le difficoltà algebriche anche nel solo problema circolare ridotto, dovute all'introduzione della costante di Jacobi, delle zone di Hill e soprattutto dei punti di Lagrange (Par. 3.3.1), hanno nascosto i dubbi di apprendimento più interessanti per la didattica, riemersi tuttavia nella necessità di costruire unità di misura ragionevoli per le simulazioni o nella lenta e graduale presa di coscienza che una simulazione è sempre e comunque un falso, sopratutto per problemi dei quali, come oramai noto, non esiste soluzione esplicita.

Ciascuno di questi punti apre molteplici direzioni di aggiornamento professionale, sia disciplinare, sia didattico. Come si è cominciato già a delineare, il testo è organizzato quindi in modo da fornire sviluppi teorici e simulazioni numeriche *nella forma assunta a seguito delle richieste dell'evoluzione ora descritta*:

- nel Cap. 1 è discusso il problema dei due corpi. Prima si mostra il sistema differenziale per ricavare la forma delle orbite (nel piano ortogonale al momento angolare totale) nel riferimento in cui uno dei due corpi sia fermo, discutendo proprio tale forma in funzione dell'energia e del momento angolare del corpo in moto. Successivamente si presenta l'equazione di Kepler, ossia la legge oraria del corpo mobile, solo recentemente proposta nella didattica [13]. Infine, in una sorta di rivoluzione copernicana nella stessa rivoluzione copernicana, si rileggono i risultati cambiando sistema di riferimento, introducendo in particolare i riferimenti *siderale* e *sinodale*, che serviranno poi nel problema dei tre corpi, e le trasformazioni di coordinate mediante cui vi si arriva;

- nel Cap. 2 si sviluppa la simulazione numerica del problema dei due corpi mediante il PC. Si utilizzano due tecniche molto differenti, sia disciplinarmente, sia didatticamente. Prima si introduce (partendo da un esempio di meccanica elementare) la *modellizzazione dinamica* mediante `Insightmaker`, un'interfaccia grafica *on-line* che permette di rappresentare il codice che simula un problema di fisica mediante un linguaggio che esprime simbolicamente le parti delle equazioni differenziali (ordinarie) alla base di esso, poi integrate nel tempo. I modelli sviluppati sono tutti disponibili online sull'account insightmaker di T. Corridoni. Successivamente si presenta un approccio abbastanza classico, basato su programmazione di codici in `C++`, l'ausilio di file *Excel* e *Geogebra* per la rappresentazione grafica dei risultati. Il fine è mostrare come la simulazione, con tutte le sue potenzialità ed i suoi limiti, possa costituire sia una sorta di laboratorio di esperimenti gravitazionali altrimenti difficilmente realizzabili (per i quali è pertanto richiesta grande consapevolezza fisica di cosa si sta facendo), sia un'analoga palestra matematica, dato che scegliere un riferimento o saper valutare l'effetto delle approssimazioni numeriche non sono solo procedure operative o di programmazione, ma cambiamenti di punto di vista che richiedono nuove strategie di pensiero. **Mancano modelli nel sistema sinodale**;

- nel Cap. 3 si discute come gli obiettivi didattici più interessanti e accessibili del problema degli N corpi possano trovarsi anche solo nel caso N=3. A riprova di ciò, si propone una rielaborazione della trattazione nel caso ideale circolare dei 3 corpi, basata su calcoli e files geogebra prodotti da F. De Vito nel suo LAM, seguito da T. Corridoni e I. Arini;

- nel Cap. 4, analogamente a quanto fatto nel Cap. 2, si mostrano soluzioni numeriche del problema dei 3 corpi. **Mancano modelli IM in generale (comunque nel caso piano), e C++ nel sistema sinodale o nel caso circolare**;

- nell'Appendice si riportano i listati dei codici `C++`, commentati nei Cap. 2 e 4, ma anche il codice Python sviluppato da F. De Vito con la supervisione di I. Arini [3].

La bibliografia, sia sul tema fisico, sia sulle tecniche informatiche, è abbastanza limitata: andrebbe aggiornata ed estesa. Mancano poi opere didattiche sul tema.

Quanto prodotto sarà aggiornato nel tempo (si veda la data sulla copertina): possono esserci errori di varia natura e gravità, soprattutto nelle trattazioni matematiche o nelle simulazioni numeriche. Si ringrazia in anticipo chi dovesse segnalarli (anche perché dimostrerà di aver studiato un testo come minimo impegnativo!). Per informazioni e segnalazioni: T. Corridoni, `ukk922@edu.ti.ch`

Capitolo 1

Il problema dei 2 corpi

1.1 Traiettorie in un campo centrale

1.1.1 Problema differenziale e sua soluzione

Si vuole trovare la traiettoria di un corpo puntiforme p di massa m (un *pianeta*) posto nel campo generato da un altro corpo puntiforme S di massa M (un *Sole*), in un sistema di riferimento in cui quest'ultimo è fermo (Fig. 1.1). La forza gravitazionale fra i due[1] è sempre parallela al braccio con cui agisce, quindi ha momento nullo ed è *conservato il vettore momento angolare del pianeta* $\vec{L} = \vec{r} \times \vec{P}$, con $\vec{r}$ posizione di p rispetto a S e $\vec{P} = m\vec{v}_{tan}$ sua quantità di moto [14]. Ne segue che:

- la traiettoria è bidimensionale, *sul piano ortogonale al vettore* $\vec{L}$;
- la velocità areica del corpo su tale piano, ossia l'area infinitesima dA spazzata in un dt dal vettore r che collega p ed S (al limite, quella di un triangolo di altezza r e base $r \cdot d\theta$, Fig. 1.1), è costante (*II legge di Kepler*):

$$L = I\omega = mr^2\dot{\theta} \qquad dA = \frac{1}{2}r \cdot r d\theta \Rightarrow \frac{dA}{dt} = \frac{1}{2}r^2\dot{\theta} = \frac{L}{2m} \tag{1.1}$$

Conviene quindi esprimere il 2. Principio della dinamica, componente per componente, rispetto ad un sistema di coordinate sul piano ortogonale ad L, centrato nel Sole. In coordinate cartesiane (Fig. 1.1) si ottiene così il sistema differenziale:

$$\vec{F} = -G\frac{Mm}{r^2}\hat{r} \Rightarrow \begin{cases} F_x = -G\frac{Mm}{r^2}\cos\theta = m\ddot{x} \\ F_y = -G\frac{Mm}{r^2}\sin\theta = m\ddot{y} \end{cases} \Rightarrow \begin{cases} \ddot{x} = -\frac{GMx}{(x^2+y^2)^{\frac{3}{2}}} \\ \ddot{y} = -\frac{GMy}{(x^2+y^2)^{\frac{3}{2}}} \end{cases} \tag{1.2}$$

Passando a coordinate polari (Fig. 1.1), le derivate diventano:

$$\begin{cases} x = r\cos\theta \\ y = r\sin\theta \end{cases} \quad \begin{cases} \dot{x} = \dot{r}\cos\theta - r\dot{\theta}\sin\theta \\ \dot{y} = \dot{r}\sin\theta + r\dot{\theta}\cos\theta \end{cases} \quad \begin{cases} \ddot{x} = \ddot{r}\cos\theta - 2\dot{r}\dot{\theta}\sin\theta - r\ddot{\theta}\sin\theta - r\dot{\theta}^2\cos\theta \\ \ddot{y} = \ddot{r}\sin\theta + 2\dot{r}\dot{\theta}\cos\theta + r\ddot{\theta}\cos\theta - r\dot{\theta}^2\sin\theta \end{cases} \tag{1.3}$$

[1] Si considereranno sempre e solo corpi puntiformi.

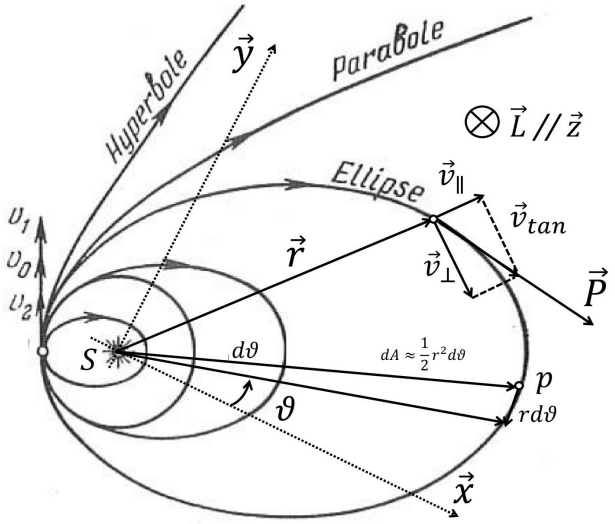


Figura 1.1: Traiettorie del pianeta p attorno al Sole S al variare del valore iniziale v_i della sua velocità tangenziale $\vec{v}_{tan}$, rappresentate in un riferimento cartesiano (x, y) e polare (r, θ), centrati su S, posti nel piano ortogonale al momento angolare $\vec{L}$ di p, parallelo all'asse $\vec{z}$ (chiralità destrorsa). Il verso di percorrenza è orario: $\vec{L}$ entra nel piano ($L < 0$) e z ne esce. Sono indicate anche le velocità radiale $\vec{v}_{//}$ e ortogonale $\vec{v}_{\perp}$ (componenti di $\vec{v}_{tan}$), la quantità di moto $\vec{P} = mv_{tan}$ di p e l'area dA spazzata dal vettore $\vec{r}$ nella rotazione di angolo infinitesimo $d\theta$. Le traiettorie risultano sezioni coniche, riferite agli assi scegliendo $\theta = 0$ all'afelio o al perielio (fig. modificata da [15]).

ma *essendo L conservato*, deve essere nulla la sua derivata rispetto al tempo:

$$L = I\dot{\theta} = mr^2\dot{\theta} \Rightarrow \dot{L} = 0 = m(r^2\ddot{\theta} + 2r\dot{r}\dot{\theta}) = mr(r\ddot{\theta} + 2\dot{r}\dot{\theta}) \tag{1.4}$$

Dunque il sistema Eq. 1.2 si semplifica, sebbene resti di 2. ordine e non lineare:

$$\begin{cases} \ddot{x} = -\frac{GM}{r^2}\cos\theta = (\ddot{r} - r\dot{\theta}^2)\cos\theta - (2\dot{r}\dot{\theta} + r\ddot{\theta})\sin\theta \\ \ddot{y} = -\frac{GM}{r^2}\sin\theta = (\ddot{r} - r\dot{\theta}^2)\sin\theta + (2\dot{r}\dot{\theta} + r\ddot{\theta})\cos\theta \end{cases} \Rightarrow \begin{cases} \ddot{r} - r\dot{\theta}^2 = -\frac{C}{mr^2} \\ \dot{\theta} = \frac{L}{mr^2} \end{cases} \tag{1.5}$$

dove $C = GMm$. Ora, visto che l'equazione in θ esprime la conservazione di L, si esprima anche l'energia totale E (cinetica più potenziale) in coordinate polari:

$$E = \frac{1}{2}m(\dot{x}^2 + \dot{y}^2) - \frac{C}{r} = \frac{1}{2}m(\dot{r}^2 + r^2\dot{\theta}^2) - \frac{C}{r} = \frac{1}{2}m\left(\dot{r}^2 + \frac{L^2}{m^2r^2}\right) - \frac{C}{r} \tag{1.6}$$

e si noti come, annullando la derivata di E rispetto al tempo, essendo conservata anch'essa, si ritrova l'equazione in r delle 1.5:

$$\dot{E} = 0 = \frac{1}{2}\left(2\dot{r}\ddot{r} - 2\frac{L^2}{m^2}r^{-3}\dot{r}\right) + \frac{C}{mr^2}\dot{r} \Rightarrow \ddot{r} = \frac{L^2}{m^2r^3} - \frac{C}{mr^2} \tag{1.7}$$

Ciò suggerisce di non integrare nel tempo il sistema Eq. 1.5, di 2. ordine, bensì l'espressione di E, come equazione del 1. ordine solo in r ed $\dot{r}$:

$$E + \frac{C}{r} = \frac{1}{2}m\left(\dot{r}^2 + \frac{L^2}{m^2r^2}\right) = \frac{1}{2}m\left(\left(\frac{dr}{d\theta}\frac{\theta}{dt}\right)^2 + \frac{L^2}{m^2r^2}\right) = \frac{1}{2}m\left(\left(\frac{dr}{d\theta}\frac{L}{mr^2}\right)^2 + \frac{L^2}{m^2r^2}\right) \tag{1.8}$$

dove si è passati dalla derivata $\dot{r}$ rispetto al tempo a quella $\frac{dr}{d\theta}$ rispetto all'angolo θ.

Isolando proprio $\frac{dr}{d\theta}$, il problema si riduce così ad una equazione differenziale a variabili separabili:

$$\frac{dr}{d\theta} = \pm r\sqrt{hr^2 + kr - 1} \qquad h = \frac{2mE}{L^2} \qquad k = \frac{2mC}{L^2} \tag{1.9}$$

L'integrale che ne segue è noto [16], e porta a:

$$\theta(r) - \theta_0 = \pm \int_{r_0}^{r} \frac{dr}{r\sqrt{hr^2 + kr - 1}} = \tan^{-1} \frac{kr - 2}{2\sqrt{hr^2 + kr - 1}} - \tan^{-1} \frac{kr_0 - 2}{2\sqrt{hr_0^2 + kr_0 - 1}} \tag{1.10}$$

con (r_0, θ_0) coordinate iniziali del pianeta nel riferimento polare.

Calcolando la tangente di entrambi i membri ($\tan(\alpha - \beta) = \frac{\tan(\alpha) - \tan(\beta)}{1 + \tan(\alpha)\tan(\beta)}$) ed esplicitando h e k, gli stessi passaggi algebrici suggeriscono di definire:

$$p = \frac{L^2}{mC} \quad \epsilon = \sqrt{1 + 2\frac{EL^2}{mC^2}} \quad a_0 = \frac{\rho_0 - 1}{\sqrt{(\epsilon^2 - 1)\rho_0^2 + 2\rho_0 - 1}} \quad f(\theta) = \frac{1 - a_0 \tan(\theta - \theta_0)}{\tan(\theta - \theta_0) + a_0} \tag{1.11}$$

dove p è una lunghezza, $\rho = \frac{r}{p}$, $\rho_0 = \frac{r_0}{p}$ ed ϵ è adimensionale. Tutto ciò riduce infatti l'Eq. 1.10 ad un'equazione di secondo grado in ρ, parametrica in ρ_0, θ_0:

$$\left(1 - \frac{\epsilon^2}{1 + f^2(\theta)}\right)\rho^2 - 2\rho + 1 = 0 \quad \Rightarrow \quad \rho(\theta) = \frac{1}{1 \pm \epsilon\left(\frac{\sin(\theta - \theta_0)}{\sqrt{1 + a_0^2}} + \frac{a_0 \cos(\theta - \theta_0)}{\sqrt{1 + a_0^2}}\right)} \tag{1.12}$$

L'Eq. 1.12 è una *conica ruotata di un angolo* δ, in quanto definite opportune funzioni trigonometriche, ed utilizzando $\cos(\alpha - \beta) = \cos\alpha\cos\beta + \sin\beta\sin\alpha$:

$$\frac{a_0}{\sqrt{1 + a_0^2}} = \frac{\rho_0 - 1}{\epsilon\rho_0} = \cos\delta \quad \frac{1}{\sqrt{1 + a_0^2}} = \sin\delta \quad \Rightarrow \quad r(\theta) = \frac{p}{1 \pm \epsilon\cos(\theta - \theta_0 - \delta)} \tag{1.13}$$

Ruotando quindi il riferimento polare di $\delta = \arccos\frac{\rho_0 - 1}{\epsilon\rho_0}$, *la conica viene riferita agli assi, con la posizione* $r = p$ *corrispondente all'angolo* $\theta = 0$ (Fig.1.2).

Scegliendo pertanto già $\theta_0 = -\delta$, o effettuando la rotazione di δ che porta il semiasse maggiore della traiettoria lungo l'asse x, le traiettorie sono [14]:

$$r(\theta) = \frac{p}{1 - \epsilon\cos\theta} \qquad p = \frac{L^2}{mC} \qquad \epsilon = +\sqrt{1 + 2\frac{EL^2}{mC^2}} \tag{1.14}$$

ossia una famiglia di rami di coniche *riferite agli assi*, con un fuoco nell'origine (quello a sinistra/destra del centro per un'ellisse/iperbole), ampiezza p ed eccentricità ϵ, percorse in senso antiorario (al crescere di θ) se $L > 0$. La loro *forma* dipende dall'energia E e dal momento angolare L (in modulo), grandezze conservate fissate dalle condizioni iniziali (posizione, velocità tangenziale o rotazionale):

$$E = \frac{1}{2}m(\dot{x}_0^2 + \dot{y}_0^2) - \frac{C}{r_0} = \frac{1}{2}m(v_{0,x}^2 + v_{0,y}^2) - \frac{C}{\sqrt{x_0^2 + y_0^2}} \qquad L = I\omega_0 = mr_0^2\dot{\theta}_0 \tag{1.15}$$

Per studiare quali valori di E ed L determinino quale conica nel riferimento scelto, si torni in coordinate cartesiane:

$$r = \frac{p}{1 - \epsilon\cos\theta} = \frac{p}{1 - \epsilon\frac{x}{r}} = \frac{rp}{r - \epsilon x} \quad \Rightarrow \quad r - \epsilon x = p \quad \Rightarrow \quad \sqrt{x^2 + y^2} - \epsilon x = p$$

$$x^2 + y^2 = (p + \epsilon x)^2 = p^2 + 2\epsilon p x + \epsilon^2 x^2 \quad \Rightarrow \quad \epsilon \neq 1 : x^2 - 2\frac{\epsilon p}{1 - \epsilon^2}x + \frac{y^2}{1 - \epsilon^2} = \frac{p^2}{1 - \epsilon^2}$$

$$\left(x - \frac{\epsilon p}{1 - \epsilon^2}\right)^2 + \frac{y^2}{1 - \epsilon^2} = x^2 - 2\frac{\epsilon p}{1 - \epsilon^2}x + \left(\frac{\epsilon p}{1 - \epsilon^2}\right)^2 + \frac{y^2}{1 - \epsilon^2} = \frac{p^2}{1 - \epsilon^2} + \left(\frac{\epsilon p}{1 - \epsilon^2}\right)^2 = \left(\frac{p}{1 - \epsilon^2}\right)^2$$

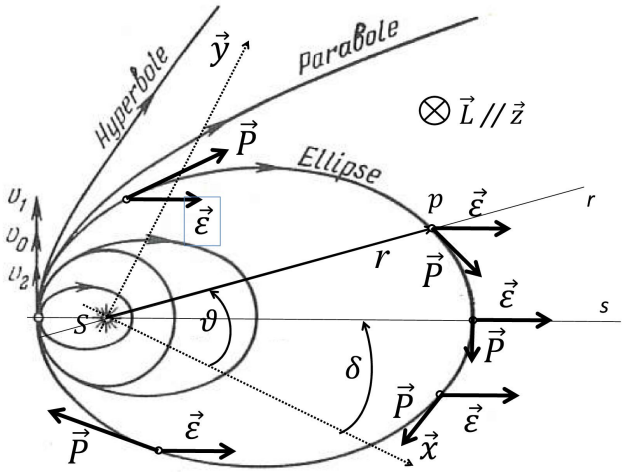


Figura 1.2: Fig. 1.1 in cui si è aggiunto l'asse di simmetria delle orbite s, parallelo al semiasse maggiore, ruotato di un angolo δ rispetto all'asse x. Il vettore $\vec{\epsilon}$ indicato qualitativamente in vari punti della traiettoria ellittica assieme al vettore quantità di moto $\vec{P}$, è il vettore di Runge-Lenz: quando la traiettoria conica è riferita agli assi, giace sull'asse x (fig. modificata da [15]).

Per $\epsilon \neq 1$ si ha quindi l'equazione di una conica in coordinate cartesiane:

$$\frac{(x-x_0)^2}{a^2} - \mathtt{sgn}(E)\frac{y^2}{b^2} = 1 \quad a = \frac{p}{|1-\epsilon^2|} = \frac{C}{2|E|} \quad x_0 = \epsilon a \quad b^2 = pa = \frac{L^2}{2m|E|} \tag{1.16}$$

dove $|E|$ e $\mathtt{sgn}(E)$ derivano dal dover avere $a = \mp\frac{C}{2E} > 0$ e $b^2 = \frac{p^2}{1-\epsilon^2} = -\frac{L^2}{2mE} > 0$.

In particolare, ricordando che è $\epsilon \geq 0$ (Eq. 1.14):

- **se $\epsilon = 0$, l'orbita è un cerchio di raggio $R = p$ e l'energia è negativa**. Per $\epsilon = 0$, infatti:

$$\frac{(x-x_0)^2}{a^2} + \frac{y^2}{b^2} = 1 \qquad a = p = \frac{C}{2|E|} \qquad x_0 = \epsilon p = 0 \qquad b^2 = p^2 = a^2 \tag{1.17}$$

  e $E_C = -\frac{mC^2}{2L^2}$ è l'energia di un moto circolare uniforme, dove $L = I\omega = R^2 m\omega$:

$$0 = \epsilon \Rightarrow -E_C = \frac{mC^2}{2L^2} = \frac{mC^2}{(R^2m)^2\omega^2} = \frac{F^2}{2m\omega^2} = \frac{m^2v^4}{2\omega^2R^2m} = \frac{1}{2}mv^2 = \frac{1}{2}mR^2\omega^2 = \frac{1}{2}I\omega^2$$

  Ad ulteriore conferma, se la velocità angolare ω è costante ed il raggio è $r = p$:

$$\dot{\theta} = \frac{L}{mr^2} = \frac{CL^2}{LAmr^2} = \frac{Cp}{Lr^2} = \omega \quad \Rightarrow \quad p\omega = v = \frac{C}{L} = \frac{C}{mr^2\omega} = \frac{|F|}{m\omega} = \frac{a}{\omega} = \frac{\omega^2 r}{\omega} = r\omega$$

- **se $\epsilon < 1$, l'orbita è ellittica e $E_C < E < 0$**:

$$1 > \epsilon = \sqrt{1 + 2\frac{EL^2}{mC^2}} = \sqrt{1 - \frac{E}{E_C}} > 0 \quad \Rightarrow \quad E_C = -\frac{mC^2}{2L^2} < E < 0 \tag{1.18}$$

  pertanto, l'orbita circolare per $\epsilon = 0$ ha energia E_C minima. Semiassi a, b, e semidistanza focale c sono:

$$a = \frac{C}{2|E|} \qquad b^2 = \frac{L^2}{2m|E|} \qquad c^2 = a^2 - b^2 = \frac{C^2}{4E^2} - \frac{L^2}{2m|E|} = a^2\epsilon^2 \tag{1.19}$$

Si trovano così afelio (+) e perielio (-):

$$a \pm c = \frac{C}{2|E|} \pm \sqrt{\frac{C^2}{4E^2} - \frac{L^2}{2m|E|}} = a(1 \pm \epsilon) \tag{1.20}$$

mentre il rapporto fra l'area πab e la velocità areolare, costante, dà il periodo T del moto (III legge di Kepler):

$$T = \frac{\pi ab}{\frac{dA}{dt}} = \frac{\pi}{\frac{L}{2m}} \frac{C}{2|E|} \frac{L}{\sqrt{2m|E|}} = \pi C \sqrt{\frac{m}{2|E|^3}} = 2\pi C \sqrt{\frac{ma^3}{C^3}}$$
$$T^2 = \frac{4\pi^2 m}{C} a^3 = \frac{4\pi^2}{GM} a^3 = \gamma_{Sole} a^3 \tag{1.21}$$

- **Se $\epsilon = 1$, l'orbita è parabolica e l'energia totale è nulla**:

$$1 = \epsilon = \sqrt{1 + 2\frac{EL^2}{mC^2}} \Rightarrow E = 0 \tag{1.22}$$

Trattandosi del caso prima escluso, va ricalcolata l'espressione di x in funzione di y^2, determinando anche l'ascissa x_V del vertice:

$$x^2 + y^2 = (p + \epsilon x)^2 = p^2 + 2\epsilon px + \epsilon^2 x^2 = p^2 - 2px + x^2 \quad \Rightarrow \quad x^2 + y^2 = (p + x)^2$$
$$y^2 = \pm(p^2 + 2px) \quad \Rightarrow \quad x(y) = -\frac{p^2 \pm y^2}{2p} \quad p = \frac{L^2}{mC} \quad x_V = x(0) = -\frac{p}{2} \tag{1.23}$$

- **Se $\epsilon > 1$, l'orbita è iperbolica, e $E > 0$.** Semiassi a, b e semidistanza focale c sono calcolati da Eq. 1.19. Gli asintoti sono rette di equazione $y = q(x - x_0)$, dove la pendenza q è il limite del rapporto $\frac{y}{x-x0}$ per x tendente all'infinito:

$$\frac{(x - x_0)^2}{a^2} - \frac{y^2}{b^2} = 1 \Rightarrow \frac{y^2}{(x - x_0)^2} = \frac{b^2}{a^2} - \frac{b^2}{(x - x_0)^2} \quad \Rightarrow \quad q = \lim_{x \to \infty} \frac{y}{x - x0} = \pm\sqrt{\frac{b^2}{a^2}} = \pm\sqrt{\epsilon^2 - 1}$$

Una classificazione grafica delle orbite si ottiene isolando la parte di energia cinetica dovuta al solo spostamento *radiale*:

$$E = \frac{1}{2}m\dot{r}^2 + \frac{1}{2}\frac{L^2}{mr^2} - \frac{C}{r} = \frac{1}{2}m\dot{r}^2 + \frac{1}{2}\frac{CL^2}{Cmr^2} - \frac{C}{r} = \frac{1}{2}m\dot{r}^2 + C\left(\frac{1}{2}\frac{p}{r^2} - \frac{1}{r}\right) \tag{1.24}$$

Fissato quindi il valore di E, i valori di r che soddisfano l'equazione:

$$E = f(r) = C\left(\frac{1}{2}\frac{p}{r^2} - \frac{1}{r}\right) \Longleftrightarrow \dot{r} = 0 \tag{1.25}$$

sono le distanze r in cui non c'è spostamento radiale, ossia il pianeta smette di allonta-/avvici-narsi da-/a-l sole perché la sua traiettoria curva, cambiando segno $\dot{r}$. Studiando f in funzione di r, inteso come positivo (Fig. 1.3):

- se $r \to 0^+, E = f(r) > 0$ solo nel perielio di una traiettoria iperbolica;

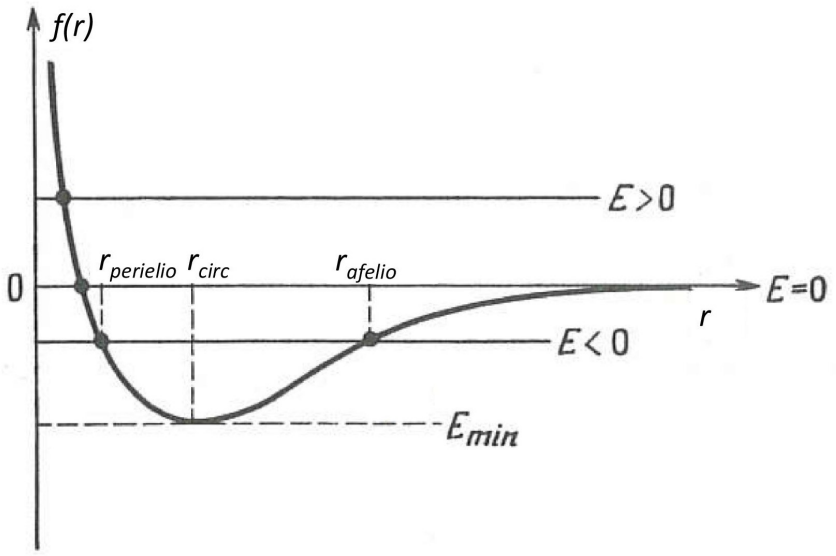


Figura 1.3: Soluzione grafica dell'Eq. 1.25: le intersezioni forniscono le distanze r di perielio ed afelio dell'orbita [15].

- se $r = \frac{p}{2}, f(r) = 0$, $E = f = 0$ nel vertice dell'orbita parabolica (Eq. 1.23);
- se $\frac{p}{2} < r < +\infty$, visto che per $r \to +\infty, f(r) < 0$ e $f(r) \to 0^-$, l'equazione $E = f(r)$ ha sempre due radici, perielio ed afelio di un'orbita ellittica. Risolvendo l'equazione di 2. grado in r e ricordando che $E < 0$, si trova $r = a \pm c$.

  Fra tutte le traiettorie ellittiche:

  - per $\frac{d}{dr}\left(\frac{1}{2}\frac{p}{r^2} - \frac{1}{r}\right) = 0 \Rightarrow r = p, f = f_{min}$: l'orbita è circolare, di raggio p. Si tratta del minimo di f, quindi c'è solo la soluzione $E = f = E_C$, unico caso in cui r non aumenta e non diminuisce;
  - per $\frac{d^2}{dr^2}\left(\frac{1}{2}\frac{p}{r^2} - \frac{1}{r}\right) = 0 \Rightarrow r = \frac{3}{2}p, f = -\frac{4C}{9p}$ dove c'è un flesso. Inoltre, $E = \frac{8}{9}E_C, \epsilon = \frac{1}{3}$, quindi l'afelio è a distanza doppia del perielio, che è in $r = \frac{3}{4}p$. È la prima delle infinite orbite ellittiche con afelio oltre $r = \frac{3}{2}p$, sempre più eccentriche (E tende a 0^-), caratteristiche delle comete.

1.1.2 Trovare le orbite senza equazioni differenziali

La simmetria delle traiettorie rispetto all'orientazione δ definita in Eq. 1.13, che indica ad es. la direzione del semiasse maggiore nelle traiettorie ellittiche, suggerisce di cercare un *invariante* oltre all'energia ed al momento angolare. Essendo δ l'inclinazione del vettore $\vec{P} \times \vec{L}$ quando il pianeta è nell'afelio/perielio, si studia come cambia questo vettore nelle altre posizioni. Dalle Eq. 1.3 si trova:

$$\vec{P}(t) \times \vec{L} = m(\dot{x}\hat{i} + \dot{y}\hat{j}) \times mr^2\dot{\theta}\hat{k} = m^2r^2\dot{\theta}((r\dot{\theta}\sin\theta - \dot{r}\cos\theta)\hat{j} + (\dot{r}\sin\theta + r\dot{\theta}\cos\theta)\hat{i}) \quad (1.26)$$

Ma sostituendo $r(\theta)$ e $\dot{r}(\theta) = -\frac{r^2}{p}\epsilon\sin(\theta - \theta_0 - \delta)\dot{\theta}$ (da Eq. 1.13), e $\dot{\theta} = \frac{L}{mr^2}$, si ha:

$$\vec{P}(t) \times \vec{L} = mC(\cos(\theta - \theta_0 - \delta)\hat{i} + \sin(\theta - \theta_0 - \delta)\hat{j} - \epsilon\hat{i}) \quad \Rightarrow \quad \epsilon\hat{i} = \hat{r}(t) - \frac{\vec{P}(t) \times \vec{L}}{mC} \quad (1.27)$$

dove $\hat{r}(t) = \cos(\theta - \theta_0 - \delta)\hat{i} + \sin(\theta - \theta_0 - \delta)\hat{j}$ è il versore della direzione radiale nel riferimento (x, y) (Fig. 1.1). Essendo ϵ una costante del moto, se ne deduce che *esiste un vettore invariante che istante per istante, puntato sul pianeta, ha modulo* ϵ *ed è orientato costantemente come il semiasse principale della traiettoria, ossia secondo l'angolo* δ (Fig. 1.2). È detto *vettore di Runge-Lenz*.

La sua definizione a *priori* permette di *determinare le orbite con il calcolo vettoriale invece che con le equazioni differenziali*, suggerendo un interessante percorso didattico alternativo [17]. L'idea parte dal definire un vettore molto simile, posto nel piano dell'orbita, dunque ortogonale al momento angolare:

$$\vec{\eta}(t) = -mC\epsilon\hat{i} = \vec{P}(t) \times \vec{L} - mC\hat{r}(t) \qquad \vec{\eta}(t) \cdot \vec{L} = 0 \tag{1.28}$$

Il suo modulo quadro sarà:

$$|\vec{\eta}(t)|^2 = |\vec{P}(t) \times \vec{L}|^2 + m^2C^2|\hat{r}(t)|^2 - 2mC(\vec{P}(t) \times \vec{L}) \cdot \hat{r}(t)$$

Ma dato che

$$(\vec{a} \times \vec{b}) \cdot \vec{c} = (\vec{b} \times \vec{c}) \cdot \vec{a} = (\vec{c} \times \vec{a}) \cdot \vec{b} \qquad |\vec{a} \times \vec{b}|^2 = |\vec{a}|^2|\vec{b}|^2 - |\vec{a} \cdot \vec{b}|^2$$

si ha che:

$$|\vec{\eta}(t)|^2 = |\vec{P}(t)|^2|\vec{L}|^2 - |\vec{P}(t) \cdot \vec{L}|^2 + m^2C^2 \cdot 1 - \frac{2mC}{r(t)}(\vec{r}(t) \times \vec{P}(t)) \cdot \vec{L}$$

Ma essendo $\vec{P}(t) \cdot \vec{L} = 0$ e $\vec{r}(t) \times \vec{P}(t) = \vec{L}$, resta:

$$|\vec{\eta}(t)|^2 = P^2L^2 + m^2C^2 - \frac{2mC}{r(t)}L^2 \qquad E = \frac{P^2(t)}{2m} - \frac{C}{r(t)}$$

Dall'espressione dell'energia totale E si trova d'altra parte $P^2(t)$, da cui:

$$\eta = |\vec{\eta}(t)| = \sqrt{2mEL^2 + m^2C^2} \tag{1.29}$$

ossia, il vettore $\vec{\eta}(t)$ *ha modulo costante nel tempo*, in quanto funzione di sole grandezze conservate.

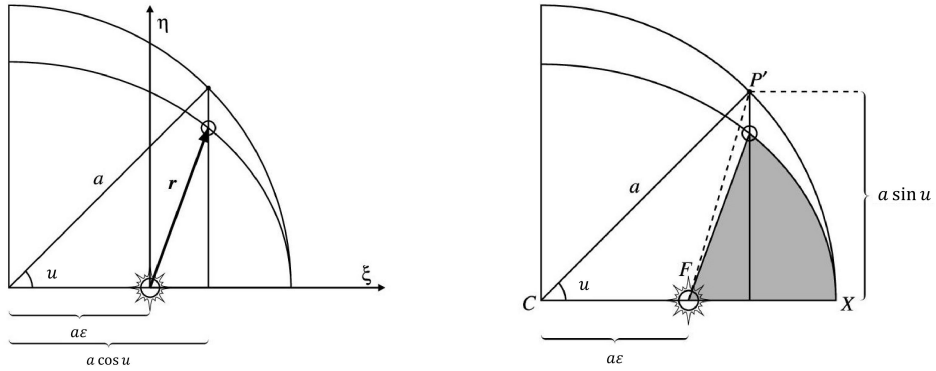
Ora, ricordando come già visto che $(\vec{P}(t) \times \vec{L}) \cdot \vec{r}(t) = L^2$, è facile verificare che:

$$\vec{\eta}(t) \cdot \vec{r}(t) + mCr(t) = L^2 \quad \Rightarrow \quad \eta r(t)\cos\theta(t) + mCr(t) = L^2$$

dove $\theta(t)$ è l'angolo fra $\vec{\eta}(t)$ e $\vec{r}(t)$, *in cui può già essere compreso* δ. Ciò porta alla nota traiettoria:

$$r(t) = \frac{p}{1 - \epsilon\cos\theta(t)} \qquad p = \frac{L^2}{mC} \qquad \epsilon = -\frac{\eta}{mC} = -\sqrt{1 + \frac{2EL^2}{mC^2}} \tag{1.30}$$

così che ripetendo quanto fatto nelle Eqq. 1.26, 1.27, si giunge anche alla stessa conclusione lì trovata: esiste un vettore conservato istante per istante, di modulo ϵ ed orientato secondo il semiasse principale della conica. Il vettore di Runge-Lenz.



(a) (b)

Figura 1.4: Costruzione geometrica per la definizione degli angoli *anomalia vera* θ, *eccentrica* u *e media* Φ: l'orbita ellittica del corpo in moto è inscritta in un cerchio di raggio a, avente lo stesso centro dell'ellisse. La costruzione è riferita a due diversi riferimenti: con l'origine nel Sole (quindi in un fuoco) 1.4(a); nel centro dell'ellisse 1.4(b) (leggermente modificato da [18]).

1.1.3 L'equazione di Kepler come *legge oraria*

Dal Par. 1.1, risulta che la traiettoria di un corpo di massa m rappresentato da un punto P, attratto da un Sole di massa M posto in un fuoco F, è piana e calcolabile esattamente in coordinate polari (Eq. 1.14):

$$r(\theta) = \frac{p}{1 - \epsilon \cos\theta} \quad p = \frac{L^2}{GMm^2} = a(1-\epsilon^2) \quad a = \frac{GMm}{2|E|} \quad \epsilon = \sqrt{1 + 2Ep} \tag{1.31}$$

con L momento angolare, E energia, a semiasse maggiore, ϵ eccentricità, tutti parametri stabiliti da posizione e velocità iniziali del corpo. La forma della traiettoria, parametrizzata nelle grandezze conservate L, E, è un risultato essenziale, sia in sé, sia come modello generale della fisica, anche moderna (si pensi al modello di Bohr-Sommerfeld), *ma costituisce proprio per questo qualcosa di diverso dalla rassicurante legge oraria esplicitata nei modelli elementari del moto.* È quindi utile mostrare come si possa ricavare una legge oraria anche nel moto dei due corpi, per semplicità solo nel caso di orbite chiuse.

Si parta dalle Figg. 1.4(a),1.4(b). Iscrivendo l'orbita ellittica del corpo in moto in una circonferenza c di raggio a e stesso centro C dell'ellisse, ad ogni angolo θ corrisponde un angolo u, detto *anomalia eccentrica*, perché anticamente θ era chiamato *anomalia vera* [18]. La ragione della costruzione si basa sul fatto che per la conservazione del momento angolare, la velocità areolare A è costante in entrambi i casi (Eq. 1.1). Si definisca pertanto *anomalia media* Φ l'angolo fra il perielio e il raggio r posizione di P, come se in un tempo $t - t_0$ il pianeta di massa m ruotasse lungo c di moto circolare uniforme con velocità angolare ω costante nel tempo:

$$\Phi = 2\pi \frac{t - t_0}{T} \qquad T = \pi GMm \sqrt{\frac{m}{2|E|^3}} \qquad T^2 = \frac{4\pi^2}{GM} a^3 \tag{1.32}$$

con T periodo del moto (Eq. 1.21). Nota l'eccentricità ϵ, l'area S in grigio in Fig. 1.4(b), legata al tempo ed al periodo, può allora esprimersi in base alla

proporzionalità fra le aree di circonferenza πa^2 ed ellisse πab:

$$S = \frac{\pi ab}{T}(t - t_0) = \frac{b}{a} \cdot Area(FP'X) = \frac{b}{a}[Area(CP'X) - Area(CP'F)] = \\ \frac{b}{a}\left(\frac{1}{2}a(au) - \frac{1}{2}(a\epsilon)a\sin u\right) = \frac{1}{2}ab(u - \epsilon\sin u) = \frac{1}{2}ab\Phi \qquad (1.33)$$

legando anomalia media Φ ed eccentrica u nell'*equazione di Kepler* [18]:

$$\Phi = u - \epsilon\sin u \qquad (1.34)$$

Espressa Φ mediante t, e u mediante Φ, si esprime ora θ mediante u. Essendo $CF = a\epsilon$, con un po' di trigonometria sui triangoli $P'CQ$ e PFQ in Fig. 1.4(a) si trovano le formule che legano le funzioni $\sin, \cos$ di u e θ:

$$\cos\theta = \frac{\cos u - \epsilon}{1 - \epsilon\cos u} \quad \cos u = \frac{\cos\theta + \epsilon}{1 + \epsilon\cos\theta} \\ \sin\theta = \sqrt{1-\epsilon^2}\frac{\sin u}{1 - \epsilon\cos u} \quad \sin u = \sqrt{1-\epsilon^2}\frac{\sin\theta}{1 + \epsilon\cos\theta} \qquad (1.35)$$

Le prime due dalle possibili espressioni di FQ:

$$r(\theta)\cos\theta = a\frac{1-\epsilon^2}{1+\epsilon\cos\theta}\cos\theta = |FQ| = a\cos u - a\epsilon$$

Le seconde, esprimendo in due modi QP: dalle proprietà dell'ellisse o da r:

$$|QP| = \frac{b}{a}|QP'| = \frac{b}{a}a\sin u = a\sqrt{1-\epsilon^2}\sin u = r(\theta)\sin\theta = a\frac{1-\epsilon^2}{1+\epsilon\cos\theta}\sin\theta \Rightarrow \sin u = \sqrt{1-\epsilon^2}1 + \epsilon\cos\theta\sin\theta$$

E sostituendo l'espressione del $\cos\theta$:

$$\sin u = \frac{1-\epsilon^2}{1+\epsilon\cos\theta}\sin\theta = \frac{\sqrt{1-\epsilon^2}}{1+\epsilon\frac{\cos u-\epsilon}{1-\epsilon\cos u}}\sin\theta \Rightarrow \sin\theta = \frac{\sin u}{1-\epsilon^2}\left(1 + \epsilon\frac{\cos u - \epsilon}{1 - \epsilon\cos u}\right) = \sqrt{1-\epsilon^2}\frac{\sin u}{1-\epsilon\cos u}$$

Trovate le Eq. 1.35, con un poco di trigonometria:

$$\cos^2\frac{\theta}{2} = \frac{1+\cos\theta}{2} = \frac{1-\epsilon}{1-\epsilon\cos u}\frac{1+\cos u}{2} = \frac{1-\epsilon}{1-\epsilon\cos u}\cos^2\frac{u}{2} \\ \sin^2\frac{\theta}{2} = \frac{1-\cos\theta}{2} = \frac{1+\epsilon}{1-\epsilon\cos u}\frac{1-\cos u}{2} = \frac{1+\epsilon}{1-\epsilon\cos u}\sin^2\frac{u}{2} \\ \tan\frac{\theta}{2} = \sqrt{\frac{1+\epsilon}{1-\epsilon}}\tan\frac{u}{2} \qquad (1.36)$$

In sostanza, noti il semiasse maggiore a, l'eccentricità ϵ ed il periodo T, calcolati in base alle condizioni iniziali del pianeta (posizione e velocità) nel sistema in cui il Sole è nell'origine, invece di ottenere un'equazione oraria esplicita:

- noto T, si calcola $\Phi(t) = 2\pi\frac{t-t_0}{T}$ (Eq. 1.32);
- nota ϵ, si inverte l'equazione di Kepler Eq. 1.34, trovando $u(\Phi)$;
- da u si trova $\theta(u) = 2\arctan\left(\sqrt{\frac{1+\epsilon}{1-\epsilon}}\tan\frac{u}{2}\right)$, Eq.1.36;
- nota a, da θ si trova $r(\theta) = a\frac{1-\epsilon^2}{1+\epsilon\cos\theta}$ (Eq.1.31), e volendo (x, y).

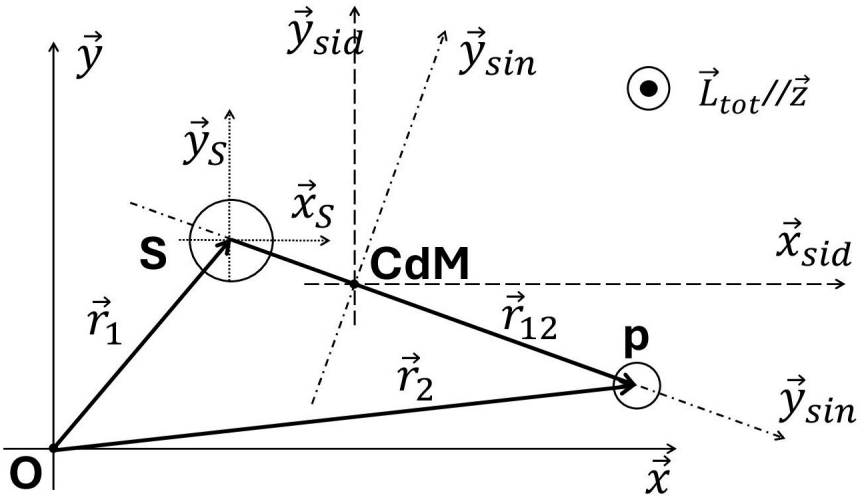


Figura 1.5: Riferimenti per lo studio del problema dei 2 corpi: **riferimento inerziale** (x, y) (assi a linee continue), fermo e centrato in O, con il Sole in r_1 ed il $p$ianeta in r_2; **riferimento in cui S è fermo**, (x_S, y_S) (linee puntate), in genere non inerziale; **riferimento siderale** (x_{sid}, y_{sid}) (tratti), centrato nel Centro di Massa (CdM), inerziale; **riferimento sinodale** (x_{sin}, y_{sin}) (tratto-punto), non inerziale, dove il CdM G è fermo ma gli assi ruotano nel tempo in modo che l'asse x_{sin} passi sempre per le due parti del sistema. Tutti sono nel piano su cui si trovano le orbite, ortogonale al vettore momento angolare totale $\vec{L}_{tot}$, pensato lungo l'asse z, in chiralità destrorsa.

Invertire l'equazione di Kepler Eq. 1.34 richiede l'utilizzo di un metodo numerico [2]. In alternativa, la si può derivare rispetto al tempo, ricordando Eq. 1.32:

$$\frac{du}{dt} = \frac{d\Phi}{dt} + \epsilon \cos u \frac{du}{dt} \qquad \Rightarrow \qquad \frac{du}{dt} = \frac{d\Phi/dt}{1 - \epsilon \cos u} = \frac{2\pi}{T} \frac{1}{1 - \epsilon \cos u} \tag{1.37}$$

da cui quindi un'equazione differenziale per la cui soluzione servono T, ϵ e la condizione iniziale $u(0)$, trovata dalla posizione iniziale θ_0 invertendo la relazione $\theta(u_0) = \theta_0$, Eq. 1.36.

1.1.4 Le leggi di Kepler in più riferimenti

Nei Parr. 1.1.1-1.1.3 si è sempre considerato il Sole *fermo* nell'origine di un riferimento, cartesiano o polare (Fig. 1.1), il cui piano (x, y) fosse quello dell'orbita del pianeta, il solo in moto. Tale riferimento sarebbe inerziale solo qualora il Sole fosse *realmente fermo*, o al limite se la sua massa fosse di molti ordini di grandezza superiore a quella del pianeta. In caso contrario è necessario studiare il moto *di entrambi i corpi*, siano essi due soli o due pianeti. Risulta a tal fine conveniente *cambiare riferimento*, osservando come ogni sua possibile scelta (Fig. 1.5) costituisca una sorta di *rivoluzione copernicana a sé*, in quanto porta a rappresentazioni geometriche diverse, ma tutte coerenti con il modello finora costruito.

[2] Ad es. il metodo di Newton. Storicamente sembra sia stato proprio questo problema a far sviluppare a Jost Bürgi (1552-1632), esperto di meccanica e matematico svizzero che collaborava con Kepler, uno dei primi e più efficienti algoritmi di calcolo della funzione seno [19], [20].

Riferimento inerziale e Riferimenti con una delle parti ferma

In un riferimento inerziale centrato in O, che veda il Sole in $\vec{r}_1$, il pianeta in $\vec{r}_2$ e la loro distanza indicata da $\vec{r}_{12} = \vec{r}_2 - \vec{r}_1$ (assi (x, y) in Fig. 1.5), in accordo con il 3. principio le forze fra le due parti sono:

$$\vec{F}^O_{2\to1} = m_1\ddot{\vec{r}}_1 = +m_1 G\frac{m_2}{r_{12}^2}\hat{r}_{12} \qquad \vec{F}^O_{1\to2} = m_2\ddot{\vec{r}}_2 = -m_2 G\frac{m_1}{r_{12}^2}\hat{r}_{12} \tag{1.38}$$

Escludendo che i due corpi abbiamo massa nulla, ricavando dal 2. principio le loro accelerazioni e *riferendo quella del pianeta a quella del Sole*, si ottiene:

$$\ddot{\vec{r}}_{12} = \ddot{\vec{r}}_2 - \ddot{\vec{r}}_1 = \frac{\vec{F}^O_{1\to2}}{m_2} - \frac{\vec{F}^O_{2\to1}}{m_1} = -\left(\frac{1}{m_1} + \frac{1}{m_2}\right) G\frac{m_1 m_2}{r_{12}^2}\hat{r}_{12} \tag{1.39}$$

Definendo la *massa ridotta* μ come $\frac{1}{\mu} = \frac{1}{m_1} + \frac{1}{m_2}$, ossia $\mu = \frac{m_1 m_2}{m_1+m_2}$, ciò porta a:

$$\mu\ddot{\vec{r}}_{12} = -G\frac{m_1 m_2}{r_{12}^2}\hat{r}_{12} \quad \Longleftrightarrow \quad \ddot{\vec{r}}_{12} = -G\frac{m_1 + m_2}{r_{12}^2}\hat{r}_{12} \tag{1.40}$$

Questa equazione va letta in due riferimenti. In quello inerziale centrato in O, esprime la variazione temporale di $\vec{r}_{12}$. In un riferimento in cui uno (qualunque) dei due corpi sia fermo, esprime invece l'accelerazione $\ddot{\vec{r}}_{12} = \ddot{\vec{r}}_i$ del corpo i in moto *riferita a quello j fermo*, permettendo di studiare *le orbite di ciascuna parte nel riferimento non inerziale in cui l'altra le appaia ferma.* Isolando $\ddot{\vec{r}}_{12}$ dall'Eq. 1.40 ed esplicitando μ, dal 2. principio per la parte i in moto si ha infatti:

$$\vec{F}^{(j)}_{j\to i} = m_i\ddot{\vec{r}}_{12} = -Gm_i\frac{m_1 + m_2}{r_{12}^2}\hat{r}_{12} \tag{1.41}$$

Quindi *nel riferimento centrato sulla parte j, tutto va come se la parte in moto i, di massa m_i, vedesse la parte j ferma, a distanza $\vec{r}_{12}$ e di massa $m_j = m_1+m_2$.* Qualora la parte j avesse massa praticamente infinita rispetto alla i, ossia $m_j >>> m_i$, si avrebbe $m_i + m_j \approx m_j$ (oppure $\mu \approx m_i$) e l'Eq. 1.40 descriverebbe il moto della sola parte i nel riferimento *inerziale* in cui la j è *ferma rispetto ad O.*

Le caratteristiche dell'orbita della parte i in moto si esprimono *riferendo posizioni e velocità a quelle di j* e sostituendo $m_i, m_j = m_1 + m_2, E_i, L_i \ldots$ nelle Eqq. 1.14, 1.16, 1.21, in quanto il problema è stato ricondotto al caso di un "'Sole"' di massa "'efficace"' $m_1 + m_2$ fermo nell'origine.

Considerando tuttavia che la posizione $\vec{r}_{12}$ e la velocità $\dot{\vec{r}}_{12}$ sono *in ogni caso relative*, quindi comuni, si possono definire le grandezze:

$$\varsigma = \frac{E_i}{m_i} = \frac{1}{2}|\dot{\vec{r}}_{12}|^2 - G\frac{m_1 + m_2}{r_{12}} \quad \lambda = \frac{L_i}{m_i} = |\vec{r}_{12}\times\dot{\vec{r}}_{12}| \quad \chi = \frac{C_i}{m_i} = G(m_1+m_2) \tag{1.42}$$

Ne segue che *le caratteristiche dell'orbita di i nel riferimento in cui vede j ferma, sono le stesse dell'orbita di j nel riferimento in cui vede i ferma*:

$$\epsilon = \sqrt{1 + 2\frac{E_i L_i^2}{m_i C_i^2}} = \sqrt{1 + 2\frac{\varsigma\lambda^2}{\chi^2}} \quad p = \frac{L_i^2}{m_i C_i} = \frac{\lambda^2}{\chi} \quad a = \frac{p}{1-\epsilon^2} \quad T^2 = \frac{4\pi^2}{\chi}a^3 \tag{1.43}$$

Ciò deriva dall'Eq. 1.40, che non specifica quale corpo sia in moto. *Ma leggerne il senso in più riferimenti esplicitandone le conseguenze geometriche dai risultati del Par. 1.1.1, ha maggiore valenza didattica.* Essendo infatti esperienza comune vedere dalla Terra il Sole sorgere e tramontare, dimostrare che le orbite sono identiche toglie ogni dubbio sul fatto che dal Sole si vedrebbe la Terra fare lo stesso moto, pur dovendo considerare l'influenza della sua rotazione su quel che si vede[3].

Un problema ancora più delicato concettualmente riguarda l'energia. In effetti *si è ricondotto il problema dei due corpi a due casi in cui ciascuno è in moto rispetto ad un punto fermo cui viene assegnata una massa efficace.* Ciò permette di usare le espressioni di energia e momento angolare *del solo corpo in moto* per calcolare le caratteristiche della sua orbita, come visto nel Par. 1.1.1. *Ma l'energia o il momento angolare totali del sistema non sono la somma delle energie e dei momenti angolari così calcolati per ciascuna parte*: nel sistema inerziale centrato in O, ed in quello in cui il corpo j sia *effettivamente* fermo, l'energia del sistema vale rispettivamente:

$$E_O = \frac{1}{2}m_1 v_{1,O}^2 + \frac{1}{2}m_2 v_{2,O}^2 - G\frac{m_1 m_2}{r_{12}} \qquad E_j = \frac{1}{2}m_i \dot{r}_{12}^2 - G\frac{m_1 + m_2}{r_{12}}m_i \quad (1.44)$$

Il senso di queste due espressioni deve essere ben chiarito: E_O è l'energia del sistema dei due corpi, conservata nel riferimento inerziale, mentre E_j è l'energia "'efficace"' di i, definita e conservata solo nel riferimento in cui j è fermo, *dove determina infatti le caratteristiche dell'orbita di* i. Lo stesso vale per il momento angolare L, del sistema e della parte in moto, nei due sistemi di riferimento.

Riferimento siderale

Il riferimento siderale è centrato nel Centro di Massa (CdM)[4] dei due corpi (assi (x_{sid}, y_{sid}) in Fig. 1.5). Se le loro masse sono m_1, m_2, le sue coordinate sono:

$$x_{CdM} = \frac{m_1 x_1(t) + m_2 x_2(t)}{m_1 + m_2} \qquad y_{CdM} = \frac{m_1 y_1(t) + m_2 y_2(t)}{m_1 + m_2} \quad (1.45)$$

mentre le componenti della sua velocità sono:

$$v_{xCdM} = \frac{m_1 v_{x1}(t) + m_2 v_{x2}(t)}{m_1 + m_2} \qquad v_{yCdM} = \frac{m_1 v_{y1}(t) + m_2 v_{y2}(t)}{m_1 + m_2}. \quad (1.46)$$

La sua scelta è dettata dal fatto che sommando e poi integrando due volte le forze in Eq. 1.38, definite nel sistema inerziale centrato in O, si ha:

$$m_1\ddot{\vec{r}}_1 + m_2\ddot{\vec{r}}_2 = 0 \Rightarrow m_1\vec{r}_1 + m_2\vec{r}_2 = at + b \Rightarrow x_{CdM}(t) = a_x t + b_x, y_{CdM}(t) = a_y t + b_y$$

[3] La *rivoluzione copernicana* ha rappresentato storicamente il sapere ciò *a priori* rispetto al riconoscerlo *a posteriori* [5]. Per Hanson, di fronte al sorgere del Sole, J. Kepler e T. Brahe *non vedevano neanche la stessa cosa* [4].

[4] Il CdM è indicato solitamente con G, ma è meglio limitare questo simbolo alla costante di gravitazione universale.

ossia, *avendo fra loro solo forze/momenti interne/i, il CdM dei due corpi si muove di moto rettilineo uniforme, su una retta ortogonale al vettore momento angolare totale e posta quindi nel piano delle orbite dei due pianeti* (3. principio).

In un riferimento che "'rincorra"' quindi il CdM vedendolo fermo nella sua origine (assi (x_{sid}, y_{sid}) in Fig. 1.5), riferendo appunto al CdM i vettori posizione $\vec{r}_1, \vec{r}_2$ dei due corpi (che dovrebbero indicarsi come $\vec{r}_i^{CdM}$), si avrebbe il sistema di equazioni:

$$\begin{cases} m_1\vec{r}_1 + m_2\vec{r}_2 = 0 \\ \vec{r}_{12} = \vec{r}_2 - \vec{r}_1 \end{cases} \Rightarrow \begin{cases} \vec{r}_1 = -\frac{m_2}{m_1+m_2}\vec{r}_{12} \\ \vec{r}_2 = \frac{m_1}{m_1+m_2}\vec{r}_{12} \end{cases} \tag{1.47}$$

Ora, essendo il riferimento siderale un particolare riferimento inerziale, in esso varrà l'Eq. 1.40, trovata nel riferimento inerziale centrato in O. Sostituendo quindi in essa $\vec{r}_{12}$ e $\ddot{\vec{r}}_{12}$ con le loro espressioni in r_1, o separatamente in r_2, ricavate dalle Eq. 1.47, si ha, ad esempio:

$$\ddot{\vec{r}}_2 = \frac{m_1}{m_1+m_2}\ddot{\vec{r}}_{12} = -G\frac{m_1}{r_{12}^3}\vec{r}_{12} = -G\frac{m_1^3}{(m_1+m_2)^2}\frac{\vec{r}_2}{r_2^3} = -G\frac{m_1}{\left(1+\frac{m_2}{m_1}\right)^2}\frac{\vec{r}_2}{r_2^3} \tag{1.48}$$

Moltiplicando quindi $\ddot{\vec{r}}_2$ o $\ddot{\vec{r}}_1$ per le rispettive masse m_i, le forze diventano:

$$\vec{F}_{2\to1}^{CdM} = m_1\ddot{\vec{r}}_1 = -m_1G\frac{m_2}{\left(1+\frac{m_1}{m_2}\right)^2}\frac{\hat{r}_1}{r_1^2} \qquad \vec{F}_{1\to2}^{CdM} = m_2\ddot{\vec{r}}_2 = -m_2G\frac{m_1}{\left(1+\frac{m_2}{m_1}\right)^2}\frac{\hat{r}_2}{r_2^2} \tag{1.49}$$

Confrontando con le Eq.1.38, si vede che *nel riferimento siderale, tutto va come se ciascuna parte i di massa m_i, vedesse nel CdM comune, da cui dista $\vec{r}_i$, un "'Sole"' di massa $m_j\left(1+\frac{m_i}{m_j}\right)^{-2}$, fermo*[5]. Calcolando quindi per ciascuna delle due parti:

- le componenti di posizione x_{iG}, y_{iG} e velocità v_{xiG}, v_{yiG} rispetto al CdM;
- e da queste la distanza r_i dal CdM, l'energia E_i ed il momento angolare L_i;

le caratteristiche dell'orbita di ciascuna parte possono esprimersi sostituendo le opportune grandezze $m_i, m_j \to m_j\left(1+\frac{m_i}{m_j}\right)^{-2}, E_i, L_i \ldots$ nelle Eqq. 1.14, 1.16, 1.21, in quanto il problema è stato ricondotto al caso di un Sole di massa "'efficace"' $m_j\left(1+\frac{m_i}{m_j}\right)^{-2}$ fermo nel CdM. Nei calcoli serve attenzione: oltre a cambiare i valori delle masse, velocità, posizioni e distanze, in particolare nell'energia potenziale, vanno riferite al CdM[6].

Altrettanta attenzione serve sul piano concettuale: come già scritto, *l'aver ricondotto il problema dei due corpi al caso in cui ciascuno è in moto rispetto ad un*

[5] In un contesto didattico, è *necessario uno schema geometrico dei vettori posizione e forza* che spieghi il senso dei segni negativi. La sua assenza qui è... *un invito a costruirlo!*

[6] È istruttivo programmare un foglio elettronico per verifiche numeriche: richiede consapevolezza in quanto "'nasconde"' le formule e, dati gli ordini di grandezza, obbliga a fare attenzione sia alle unità di misura sia alla valutazione di eventuali approssimazioni numeriche.

punto fermo cui viene assegnata una massa efficace, fa sì che possa crearsi confusione fra le espressioni di energia e momento angolare del solo corpo considerato in moto, necessarie a calcolare le caratteristiche geometriche della sua orbita in quel riferimento, *ed i valori di energia e momento angolare totale dell'intero sistema in altri riferimenti.*

Aiutandosi con la costruzione in Fig. 1.6, è ora molto interessante dimostrare quali corollari alle leggi di Kepler si abbiano nel riferimento siderale, come detto una sorta di *rivoluzione copernicana nella rivoluzione copernicana*:

- *le orbite dei due pianeti sono omotetiche. Se chiuse, essi percorrono ellissi con fuochi allineati (di cui uno condiviso: il CdM), hanno semiassi maggiori tali che* $m_2 a_2 = m_1 a_1$, *stessa eccentricità* ϵ *e sono percorse con stesso periodo* T *e stesso verso (ossia,* L_i *ed* L_j *hanno stesso segno).*

In effetti, per costruzione il CdM è allineato istante per istante con i due pianeti (Fig. 1.6(a)). Mano a mano che essi si muovono lungo le proprie orbite, il vettore r_{12} che li unisce ruota quindi passando sempre per il CdM, con velocità angolare non necessariamente costante e cambiando in genere lunghezza. *Tale dinamica impone che verso e periodo* T *di percorrenza delle orbite siano comuni.* Dall'Eq. 1.21 per T, considerato che ciascun pianeta vede la massa efficace dell'altro, segue così la relazione fra masse e semiassi maggiori:

$$\frac{4\pi^2}{G\frac{m_2^3}{(m_1+m_2)^2}}a_1^3 = T^2 = \frac{4\pi^2}{G\frac{m_1^3}{(m_1+m_2)^2}}a_2^3 \quad \Rightarrow \quad m_1 a_1 = m_2 a_2 \tag{1.50}$$

Ma essendo i fuochi allineati, devono esserlo anche gli afelî ed i perielî, raggiunti contemporaneamente dalle due parti. Essendo $m_1 x_1 = -m_2 x_2$, perché il CdM è nell'origine, devono quindi esistere due istanti in cui:

$$x_1 = (a_1 \pm c_1) = a_1(1 \pm \epsilon_1) \quad x_2 = -a_2(1 \pm \epsilon_2) \Rightarrow m_1 a_1(1 \pm \epsilon_1) = m_2 a_2(1 \pm \epsilon_2) \tag{1.51}$$

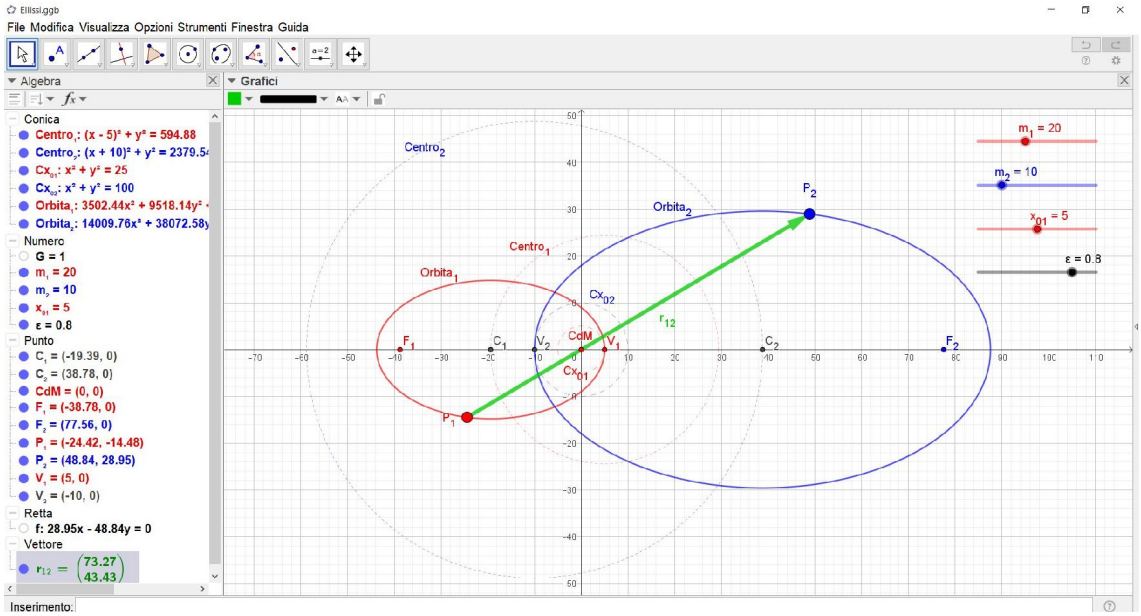
Ma se avere lo stesso periodo implica già $m_1 a_1 = m_2 a_2$, allora questo risultato è valido solo se $\epsilon_1 = \epsilon_2$, da cui la tesi.

- *Nel caso di orbite chiuse, deve valere* $m_1^3 L_1^2 E_1 = m_2^3 L_2^2 E_2$.

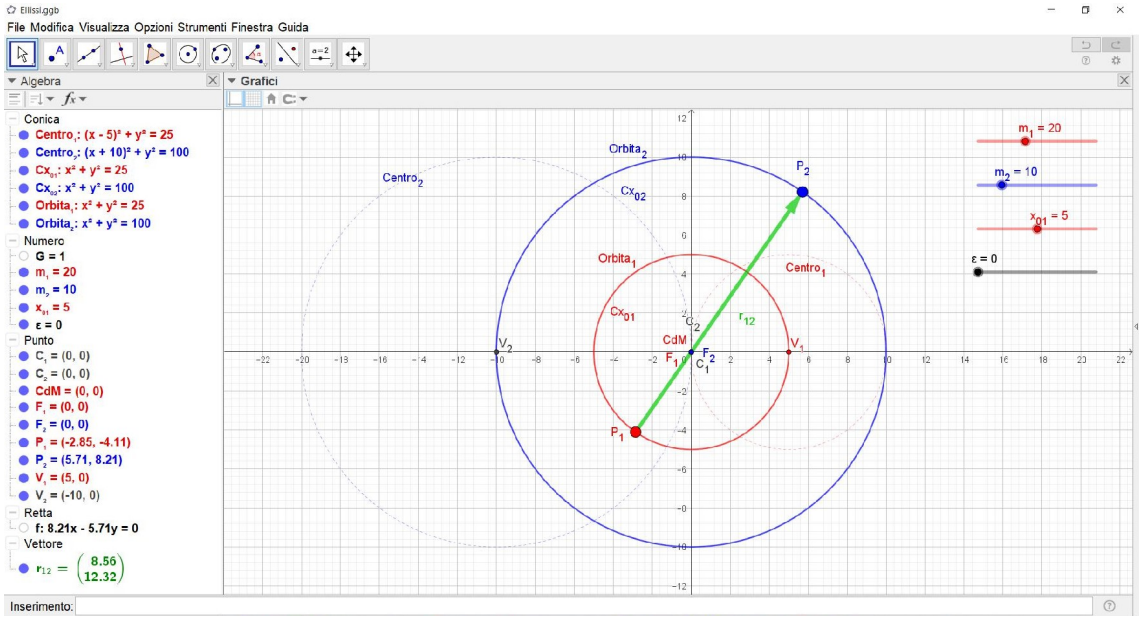
Si osserva innanzi tutto che E_i *è l'energia del corpo* i *quando il problema è ricondotto al caso in cui* i *è in moto rispetto ad un punto fermo, qui il CdM, cui viene assegnata una massa efficace* $\frac{m_j^3}{(m_1+m_2)^2}$*, come se il corpo* j *non ci fosse.* Le E_i permettono pertanto di calcolare le *eccentricità delle orbite*, ma *l'energia totale del sistema dei due corpi nel sistema siderale non è la somma di* E_1, E_2, bensì un'espressione analoga ad E_O nell'Eq. 1.44. Chiarito ciò, per l'Eq. 1.18, avere la stessa eccentricità implica:

$$\sqrt{1 - \frac{E_1}{E_{C1}}} = \epsilon = \sqrt{1 - \frac{E_2}{E_{C2}}} \Rightarrow \frac{E_1}{E_2} = \frac{E_{C1}}{E_{C2}} = \frac{m_1 C_1^2}{2L_1^2}\frac{2L_2^2}{m_2 C_2^2} \tag{1.52}$$

Con un poco di algebra si arriva al risultato sostituendo $C_i = \frac{G m_i m_j^3}{(m_1+m_2)^2}$.



(a)



(b)

Figura 1.6: È particolarmente istruttivo costruire con `Geogebra` cosa si vede nel riferimento siderale. Fra i vari modi per farlo, in Fig. 1.6(a) si è scelto di parametrizzare con degli slider le masse m_1, m_2, la posizione iniziale x_{01} del corpo 1 lungo l'asse x (tutte in unità arbitrarie) e l'eccentricità comune ϵ. Nota la relazione $m_1 x_{01} = -m_2 x_{02}$ ed il fatto che ogni afelio dista $a_i(1-\epsilon)$ dall'origine, ossia dal CdM posto in uno dei fuochi, possono trovarsi i centri C_1, C_2 e gli altri due fuochi F_1, F_2 delle ellissi, poi costruite da `Geogebra`. In Fig. 1.6(b) si mostra il caso in cui $\epsilon = 0$ (si noti la scala). Si lascia a chi legge il divertimento di realizzare un file `Geogebra` migliore, in cui *procedendo per omotetia* si generalizzi la costruzione anche ad *orbite non chiuse*. Non solo: in 1.6(a) si è scelto il riferimento siderale in cui l'asse x_{sid} passa per i tre fuochi, cosa non vera per velocità iniziali arbitrarie nel piano delle orbite.

- *la velocità angolare $\Omega(t)$ della rotazione del segmento r_{12} che unisce i due corpi attorno al CdM (Fig. 1.6(a)), è $\Omega^2(t) = \frac{Gm_j^3}{(m_1+m_2)^2}\frac{1-\epsilon\cos\theta(t)}{r_i^3(t)}$, invariante per $i \leftrightarrow j$, massima/minima quando i, j sono (assieme) al perielio/afelio, costante solo per orbite circolari (Fig. 1.6(b)).*

Consideriamo per cominciare il moto del corpo di massa m_i. Allora, dalle Eqq. 1.1, 1.14:

$$\dot\theta = \frac{L_i}{m_i r_i^2} = \frac{L_i}{m_i p_i^2}(1-\epsilon\cos\theta)^2 = \frac{m_i^2 C^2 L_i}{m_i L_i^4}(\ldots)^2 = \frac{m_i^3 G^2 m_j^2}{L_i^3}(\ldots)^2 = \frac{m_i^3 G^2 m_j^2}{m_i^3 r_i^6 \dot\theta^3}(\ldots)^2$$

Interpretiamo ora il risultato generale $\Omega^2(t) = \dot\theta^2 = \frac{Gm_j}{r_i^3}(1-\epsilon\cos\theta)$ considerando *anche* il moto del corpo j:

- se è fermo nell'origine, allora $m_j \to m_1 + m_2$ e $\Omega^2(t) = \frac{G(m_1+m_2)}{r_i^3}(1-\epsilon\cos\theta)$;
- nel caso m_j sia fermo nell'origine ed il moto di i sia circolare, allora $\epsilon = 0$ e si ottiene un'espressione ben nota, $\Omega^2(t) = \frac{G(m_1+m_2)}{R_i^3}$ (risultato che si trova anche eguagliando l'attrazione gravitazionale alla forza centripeta, o da $|E_C| = \frac{m_i C_i^2}{2L_i^2} = \frac{1}{2} I_i \Omega^2$);
- se i, j sono nel sistema del centro di massa, allora

$$m_j \to m_j\left(1+\frac{m_i}{m_j}\right)^{-2} \quad \Rightarrow \quad \Omega^2(t) = \left(\frac{m_j}{r_i(t)}\right)^3 \frac{G}{(m_1+m_2)^2}(1-\epsilon\cos\theta(t)) \quad (1.53)$$

  Ma se il CdM è nell'origine, al di là dei segni $m_i r_i(t) = m_j r_j(t)$ e Ω è invariante per $i \leftrightarrow j$. Quindi $\Omega(t)$ è comune, e dipendendo dall'inverso del cubo della distanza dal CdM è minima negli afelî, massima nei perielî e costante solo per traiettorie circolari, dove $\forall t : r_i(t) = R_i, r_j(t) = R_j, R_i + R_j = r_{12}$.

Dato che dopo aver affermato che le due orbite sono omotetiche ci si è limitati al caso ellittico, si lascia a chi legge il divertimento di dimostrare se tutto quanto detto sia valido anche per orbite non chiuse[7].

Riferimento sinodale

Il riferimento sinodale è centrato nel CdM dei due corpi, fermo nel tempo, mentre gli assi (x_{sin}, y_{sin}) ruotano con velocità angolare $\Omega(t)$ (Eq. 1.53) in modo che l'asse $x_{sin} // \vec r_{12}$ passi sempre per entrambi (Figg. 1.5, 1.6). È il miglior riferimento *non inerziale* per osservare l'influenza del moto di due pianeti su quello di altri, soprattutto se di masse molto minori (*problema degli N corpi*). Come si deduce da Fig. 1.6, dato che il segmento r_{12} ruota con velocità angolare $\Omega(t)$ passando sempre per il CdM, in tale riferimento i due pianeti rimangono lungo l'asse $\vec x_{sin} // \vec r_{12}$ allonta/avvici-nandosi al CdM e restando fra $a_i - c_i = a_i(1-\epsilon)$ (perielio) e $a_i + c_i = a_i(1+\epsilon)$ (afelio) con $x_{i,sin} m_i = -x_{j,sin} m_j$. Se le orbite sono circolari ($\epsilon = 0$), appaiono fermi nelle posizioni $x_{i,sin} = R, x_{j,sin} = -R\frac{m_i}{m_j}$.

Per rappresentare i due corpi nel sistema sinodale:

[7] Basta riflettere che risolvendo l'equazione di Kepler per ciascuno dei due corpi nel sistema siderale, le due orbite condividono T, ϵ, quindi anche $\Phi(t), u(t), \theta(t)$. L'unica grandezza che le identifica è quindi a_i (e r_i), che sappiamo segue $a_i m_i = a_j m_j$, ossia l'omotetia, valida $\forall\epsilon$.

- si riportano sull'asse x_{sin} le distanze $x_{i,sin} = r_i = \sqrt{x_{i,sid}^2 + y_{i,sid}^2}, x_{j,sin} = r_j = -\frac{m_i}{m_j} r_i$ calcolate dalle coordinate nel riferimento siderale ($z_{i,sid} = 0$);
- noti ϵ, T, a_i nel riferimento siderale, si risolve l'Eq. 1.34 di Kepler: si calcola $u(t)$ dalla versione differenziale Eq. 1.37 del, poi $\theta(u(t))$ dalla Eq. 1.36, poi le $r_i(t) = \frac{a_i(1-\epsilon^2)}{1-\epsilon\cos\theta(t)} = -\frac{m_j}{m_i} r_j(t)$;
- sostituendo $r_i(t) = r(\theta(t))$ nell'Eq. 1.53, $\Omega(t)$ è funzione solo di θ. Noto θ_0 nel sistema siderale, si dovrebbe dunque poter trovare $\theta(t)$, e quindi poi $r_i(t)$, dall'equazione differenziale (invariante per $i \leftrightarrow j$):

$$\dot{\theta}(t) = \Omega(t) = \sqrt{\frac{G m_j^3}{a_i^3 (1-\epsilon^2)^3}} \frac{(1 - \epsilon\cos\theta(t))^2}{m_1 + m_2} \tag{1.54}$$

- si opera una doppia rotazione delle coordinate, istante per istante, come si mostra nel Par. 1.1.5.

1.1.5 Cambiamenti di riferimento e coordinate

Nel Par. 1.1.4 si è visto come bastino considerazioni geometriche a capire come le orbite di due corpi cambino con il riferimento. Le relative trasformazioni di coordinate vanno tuttavia approfondite per due ragioni:

- *tutti i riferimenti in Fig. 1.5 pongono gli assi* (x, y) *nel piano delle orbite, non immediato da trovare in tre dimensioni*[8];
- estendendo il problema da 2 ad N corpi (Cap. 3), occorre distinguere fra il caso *ideale*, nel quale due corpi vedono l'interazione gravitazionale fra loro ma non quella con altri (i quali invece li "'vedono"'), ed il caso *reale*, nel quale *le interazioni vanno tutte considerate*. Infatti, *solo nel caso ideale si può disaccoppiare fin da subito la dinamica dei due corpi principali da quella delle altre* $N-2$ *parti che devono seguirli mentre si muovono indisturbati, ad esempio nel riferimento sinodale (Par. 1.1.4).*

Se non si vuole o non si può ridurre il sistema al caso ideale, per cambiare riferimento *a tutte le N parti* occorre stabilire trasformazioni di coordinate, semplici geometricamente ma comunque rischiose in termini di precisione numerica.

Note le condizioni iniziali dei due pianeti in tre dimensioni, per passare nei riferimenti siderale e sinodale occorre porre prima di tutto il CdM nell'origine, con una traslazione:

$$x'(t) = x(t) - x_G(t) \qquad y'(t) = y(t) - y_G(t) \qquad z'(t) = z(t) - z_G(t) \tag{1.55}$$

[8]Il metodo più semplice è porsi nel riferimento in cui uno dei due corpi è fermo nell'origine: fra tutti i piani contenenti il vettore *velocità relativa iniziale* dell'altro corpo, il piano dell'orbita sarà l'unico passante per l'origine, nonché ortogonale al momento angolare del corpo che si muove. Poi, con trasformazioni di coordinate, si esprime il piano in un riferimento inerziale.

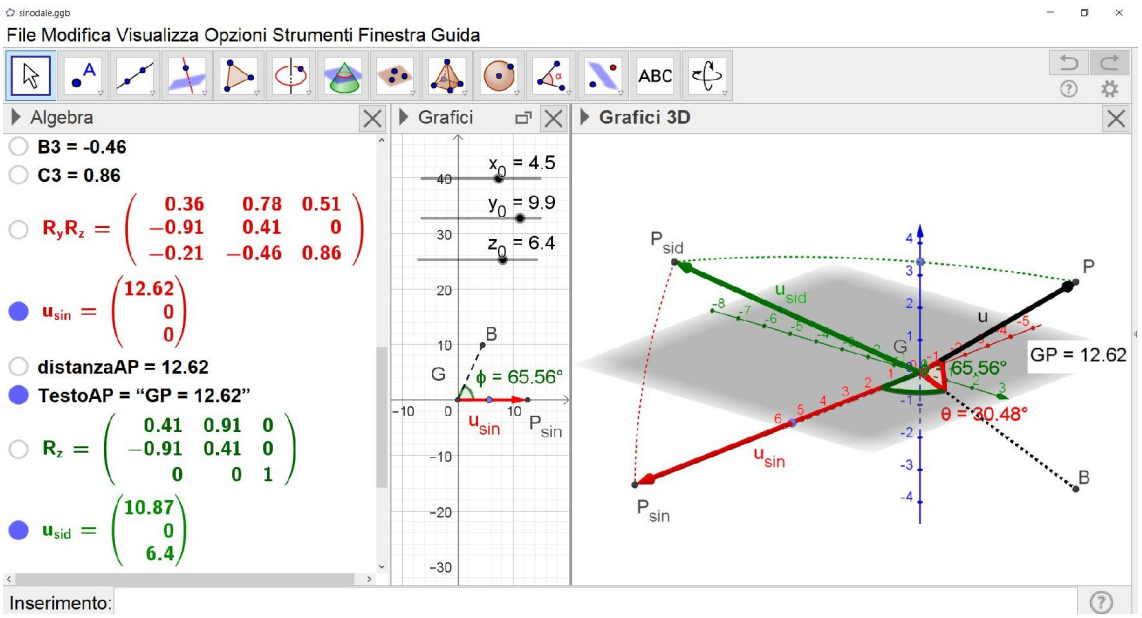


Figura 1.7: Portato il CdM nell'origine del riferimento, i sistemi siderale e sinodale si raggiungono mediante due rotazioni (indicate dagli archi di circonferenza tratteggiati) di angoli determinati dalle coordinate istantanee di uno solo dei due corpi. L'asse x è rosso, l'asse y verde, quello z blu.

Ricavando da queste nuove coordinate *istante per istante* la longitudine $\phi(t)$ e la latitudine $\vartheta(t)$ di uno dei due corpi (rispetto al CdM, Fig. 1.7)[9], con una rotazione di $-\phi(t)$ che lo porti sul piano (x, z) si raggiunge il riferimento siderale, e con una seconda rotazione di $\vartheta(t)$ che lo porti su un nuovo asse $x'' = x_{sin}$, si raggiunge quello sinodale. *Trovandosi infatti i due corpi ai capi del vettore* r_{12} *(Fig. 1.6(a)), ponendone uno su* x_{sin} *vi si pone automaticamente anche l'altro.* Ora, le matrici di rotazione antioraria di un angolo α rispetto a dei generici assi (x, y, z) sono [21]:

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \quad R_y(\alpha) = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{pmatrix} \quad R_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

con l'inversa di ciascuna pari alla sua trasposta, corrispondente ad una rotazione oraria, ossia di un angolo $-\alpha$.

Immaginiamo dunque un punto $P(x_0 > 0, y_0 > 0, z_0 > 0)$ in un riferimento *già centrato nel CdM* e destrorso, nel quale quindi se l'asse x ruota sull'asse y, l'asse z vede la rotazione antioraria (Fig. 1.7). Per comodità, dopo la traslazione di Eq. 1.55, togliamo gli apici alle coordinate e mettiamo un pedice 1, per specificare che P rappresenta il corpo 1. Per arrivare in un sistema siderale, occorre ruotare in senso orario, rispetto all'asse z, l'intero piano contenente P e passante per l'asse z, fino a sovrapporlo al piano individuato dagli assi (x, z) (il sistema siderale, appunto).

[9]Come in geografia terrestre, in Fig. 1.7 la longitudine ϕ è nulla sul piano (x, z) (il meridiano di Greenwich) e positiva in verso antiorario rispetto all'asse z. Il verso geografico della latitudine ϑ, nulla all'equatore e crescente verso il polo nord, è qui invece invertito, affinché la rotazione di $+\vartheta$ sia antioraria rispetto all'asse y.

L'angolo ϕ_1 di questa rotazione è caratterizzato da:

$$\sin\phi_1 = \frac{y_1}{\sqrt{x_1^2+y_1^2}} \qquad \cos\phi = \frac{x_1}{\sqrt{x_1^2+y_1^2}} \tag{1.56}$$

dove x_1, y_1, z_1 *sono funzioni del tempo* e si vedono subito i rischi qualora si utilizzi una simulazione numerica: i denominatori non devono essere nulli (o comunque troppo piccoli), e sul piano (z, y) va scelto $\phi_1 = \pm 90°$. Le nuove coordinate *di qualsiasi punto* i rispetto alle vecchie sono quindi:

$$\begin{pmatrix} x_{sid,i} \\ y_{sid,i} \\ z_{sid,i} \end{pmatrix} = R_z(-\phi_1) \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} \cos\phi_1 & \sin\phi_1 & 0 \\ -\sin\phi_1 & \cos\phi_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix}$$

In base alle Eq. 1.56, si vede che il corpo $i = 1$, di coordinate x_1, y_1, z_1, viene ad avere $y_{sid,1} = 0$.

Per raggiungere ora il sistema sinodale occorre ruotare in senso antiorario rispetto all'asse y l'intero piano contenente P e passante per l'asse y, fino a sovrapporlo al piano individuato dagli assi (x, y). L'angolo ϑ_1 di questa rotazione è caratterizzato da altre due espressioni che non devono essere singolari:

$$\sin\vartheta_1 = \frac{z_1}{\sqrt{x_1^2+y_1^2+z_1^2}} \qquad \cos\vartheta_1 = \frac{\sqrt{x_1^2+y_1^2}}{\sqrt{x_1^2+y_1^2+z_1^2}} \tag{1.57}$$

Ricordando che le rotazioni, come le matrici, non seguono la proprietà commutativa, le nuove coordinate *di un punto qualunque* i rispetto a quelle di partenza o a quelle siderali sono quindi:

$$\begin{pmatrix} x_{sin,i} \\ y_{sin,i} \\ z_{sin,i} \end{pmatrix} = R_y(\vartheta_1)R_z(-\phi_1) \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = R_y(\vartheta_1) \begin{pmatrix} x_{sid,i} \\ y_{sid,i} \\ z_{sid,i} \end{pmatrix} = \begin{pmatrix} \cos\vartheta_1 & 0 & \sin\vartheta_1 \\ 0 & 1 & 0 \\ -\sin\vartheta_1 & 0 & \cos\vartheta_1 \end{pmatrix} \begin{pmatrix} x_{sid,i} \\ y_{sid,i} \\ z_{sid,i} \end{pmatrix}$$

Pertanto, riassumendo:

$$\begin{cases} x_{sid,i} = x_i\cos\phi_1 + y_i\sin\phi_1 \\ y_{sid,i} = -x_i\sin\phi_1 + y_i\cos\phi_1 \\ z_{sid,i} = z_i \end{cases} \qquad \begin{cases} x_{sin,i} = x_{sid,i}\cos\vartheta_1 + z_{sid,i}\sin\vartheta_1 \\ y_{sin,i} = y_{sid,i} = -x_i\sin\phi_1 + y_i\cos\phi_1 \\ z_{sin,i} = -x_{sid,i}\sin\vartheta_1 + z_{sid,i}\cos\vartheta_1 \end{cases} \tag{1.58}$$

Dalle Eq. 1.57 tali relazioni implicano che per il corpo $i = 1$:

$$x_{sin,1}(t) = \sqrt{x_1^2(t)+y_1^2(t)+z_1^2(t)} \qquad y_{sin,1} = 0 \qquad z_{sin,1} = 0 \tag{1.59}$$

ossia, come atteso, esso si muove sull'asse x del sistema sinodale. Per il corpo 2, dato che la trasformazione ruota solamente gli assi rispetto al CdM, fermo nell'origine, calcolata $x_{sin,1}(t)$ dovrà valere al solito $m_1 x_{sin,1}(t) = -m_2 x_{sin,2}(t)$.

È molto più importante calcolare come la doppia rotazione studiata operi sulle coordinate di un qualunque altro corpo portato nel sistema sinodale dei soli due considerati, sia che li perturbi (caso reale), sia che non lo faccia (caso ideale). Si

lascia a chi legge il divertimento di dimostrare che un terzo corpo $i = 3$ avrebbe coordinate (funzioni delle $x_1(t), y_1(t), z_1(t)$ del corpo 1, in cui sono espressi ϕ_1 e ϑ_1):

$$\begin{cases} x_{sid,3} = x_3 \cos\phi_1 + y_3 \sin\phi_1 = \frac{x_3 x_1 + y_3 y_1}{\sqrt{x_1^2+y_1^2}} \\ y_{sid,3} = -x_3 \sin\phi_1 + y_3 \cos\phi_1 = \frac{-x_3 y_1 + y_3 x_1}{\sqrt{x_1^2+y_1^2}} \\ z_{sid,3} = z_3 \end{cases}$$

$$\begin{cases} x_{sin,3} = x_{sid,3} \cos\vartheta_1 + z_{sid,3} \sin\vartheta_1 = x_{sid,3} \sqrt{\frac{x_1^2+y_1^2}{x_1^2+y_1^2+z_1^2}} + \frac{z_3 z_1}{\sqrt{x_1^2+y_1^2+z_1^2}} = \frac{\vec{r}_1 \cdot \vec{r}_3}{r_1} \\ y_{sin,3} = y_{sid,3} = \frac{(\vec{r}_1 \times \vec{r}_3)\cdot \hat{k}}{|\vec{r}_1 \times \hat{k}|} \\ z_{sin,3} = -x_{sid,3} \sin\vartheta_1 + z_{sid,3} \cos\vartheta_1 = -\frac{x_{sid,3} z_1}{\sqrt{x_1^2+y_1^2+z_1^2}} + z_3 \sqrt{\frac{x_1^2+y_1^2}{x_1^2+y_1^2+z_1^2}} = \frac{(\vec{r}_1 \times \vec{r}_3)}{r_1} \cdot \frac{\vec{r}_1 \times \hat{k}}{|\vec{r}_1 \times \hat{k}|} \end{cases} \tag{1.60}$$

Nel Cap. 3 si tornerà sul solo caso $N = 3$.

Capitolo 2

Simulazione del moto dei 2 corpi

2.1 Introduzione al problema

Richiamando i principi della dinamica e quelli di conservazione, nel Cap. 1:

- sono state scritte e risolte le equazioni differenziali (Eq. 1.5) che portano alla forma della traiettoria di un corpo che si muove sotto l'azione di una forza gravitazionale;
- lo stesso problema è stato risolto con i soli principi di conservazione ed il calcolo vettoriale (Par. 1.1.2);
- si è definito un algoritmo per trovare la sua legge oraria dall'equazione di Kepler Eq. 1.34, non lineare;
- si è continuamente mostrato come cambiare coordinate o riferimento porti a *vedere* il problema in modi diversi.

Tutto ciò dimostra sia la potenza sia i limiti della fisica matematica, innescando domande *sul metodo* per costruire un *modello fisico*:

- quali principi richiamare, quali grandezze studiare, in funzione di quali altre ?
- come preservare aspetti non solo algebrici (ad esempio, schemi vettoriali) ?
- quale metodo scegliere per simulare numericamente i casi non lineari ?

Se si pensa a come Hestenes [11], ad esempio, definisce cosa sia un *modello* in fisica, ossia *la rappresentazione in un linguaggio logico della struttura delle relazioni fra un insieme di parti, siano esse oggetti concreti o astratti*, si comprende come queste domande abbiano implicazioni soprattutto didattiche, in quanto uno stesso modello può essere (e dovrebbe essere) rappresentato in modi diversi cambiando *paradigma*: già nel caso dei due corpi si sono utilizzati ad esempio sia i principi della dinamica, sia quelli di conservazione, nonché cambiamenti di riferimento, considerati alla base del *cambiamento di paradigma* per antonomasia, quello copernicano [5, 10].

Partendo da un esempio semplice (la caduta di un oggetto in aria) e passando successivamente al problema dei due corpi in più riferimenti, in questo capitolo introdurremo pertanto due metodi complementari: i *modelli di simulazione dinamica* e le *simulazioni da codice*. La simulazione dinamica è un modo di procedere molto utile quando le equazioni sono molte, abbastanza complesse da perdere facilmente il controllo del senso fisico del problema matematico e soprattutto *qualora non si voglia affrontare la scrittura di un codice*. Il metodo consente di impostare molti problemi di notevole complessità [22], *permettendo di controllarne la struttura logica mediante un linguaggio grafico* a patto di affidarsi, questo il prezzo, a molte *black box* (un metodo di approssimazione numerica, un linguaggio di programmazione, la potenza di calcolo di un computer...), accettando anche qualche manchevolezza (ad esempio, la rappresentazione grafica in sole 2D). Si presenterà quindi poi anche il metodo complementare e più noto, ossia la scrittura di un codice (in `C++`), anch'esso con pregi, come la velocità di calcolo, e difetti, primo fra tutti il fatto che il problema fisico sia ben nascosto da un linguaggio *da apprendere di per sé*.

2.2 Simulazione dinamica (dynamical modeling)

2.2.1 Da dynamo a Insightmaker

Se la storia del *pensare per modelli* è quella dell'intera scienza, quella della *simulazione dinamica* nasce al MIT di Boston attorno al 1970. L'idea era di richiamare *routines* scritte in un codice chiamato `dynamo`, disegnando *simboli grafici* usati nei *diagrammi di flusso* dei loro progetti. Nel 1972 il metodo fu richiesto dal Club di Roma per realizzare un grande modello finalizzato a stimare *i limiti dello sviluppo* dell'umanità sulla Terra, ancora oggi emblematico negli studi sulla sostenibilità [23].

Proprio in quegli anni, sulla base dei lavori di molti matematici, nacque anche la *teoria dei sistemi dinamici*. Semplificando, si scoprì il *caos deterministico*: anche se sappiamo scrivere equazioni differenziali esatte per un fenomeno, basta in genere che non siano lineari per far sì che neanche un computer possa trovarne soluzioni esplicite. Nei casi peggiori possiamo sperare di determinare solo approssimazioni numeriche su tempi brevi (problema della *sensibile dipendenza dalle condizioni iniziali*) o comportamenti asintotici (gli *attrattori* del sistema). Se si escludono gli studi in cui non fu riconosciuto [24], il caos deterministico è stato osservato per la prima volta nel *modello di Lorentz* (1963), finalizzato a descrivere l'evoluzione della temperatura nell'atmosfera [25]. Da quel momento in poi la *costruzione di modelli dinamici* mediante software specifici (`Basic, Fortran, Pascal, C, C++`...) divenne parte della fisica: da un lato necessaria per equazioni irresolubili, dall'altro problematica se non gestita rigorosamente.

Negli anni '90 del 1900, essendosi imposta negli anni '80, con l'avvento dei PC, l'abitudine di identificare con *icone* delle parti di codice, *dynamo* è stato tradotto in *oggetti grafici* da software come `Stella`, `Berkeley-Madonna`, `Vensim`, `StochSD`

[26]-[29]. Qui si è scelto `Insightmaker` [30], per tre ragioni: svolge la maggior parte delle funzioni di questi software, non richiede licenze di installazione, è on-line, quindi qualunque allieva/o connesso in rete da PC o Smartphone può lavorarvi autonomamente, decidendo quali risultati rendere pubblici. Chiaramente, data una classe reale, serve una riflessione sull'uso di un tale strumento, ma le sue potenzialità sono evidentemente positive.

2.2.2 La caduta verticale di un oggetto

Per cominciare un modello Insightmaker (IM), si va su `https://insightmaker.com` (Fig. 2.1(a)) e si crea un account con nome e password (Fig. 2.1(b)). All'accesso, compare una pagina con i propri modelli, detti *insights* (in Fig. 2.1(c) c'è ad es. il modello *Harmonic Oscillator*, che l'utente non ha reso pubblico). Per creare un modello si clicca su *Create new insight* e si chiude la pagina dei consigli (Fig. 2.1(d)).

Il metodo classico per modellizzare un oggetto che cada in aria è scegliere come grandezza di studio la sua posizione verticale z, applicare il 2. principio di Newton e trovare così un'equazione differenziale ordinaria del 2. ordine, la cui soluzione *numerica* richiede due *integrazioni*. Dato che ciò aumenta i calcoli e quindi gli errori numerici, si preferisce un'equazione del 1. ordine. Ricordando che il 2. principio può scriversi mediante la *quantità di moto* $\vec{p} = m\vec{v}$:

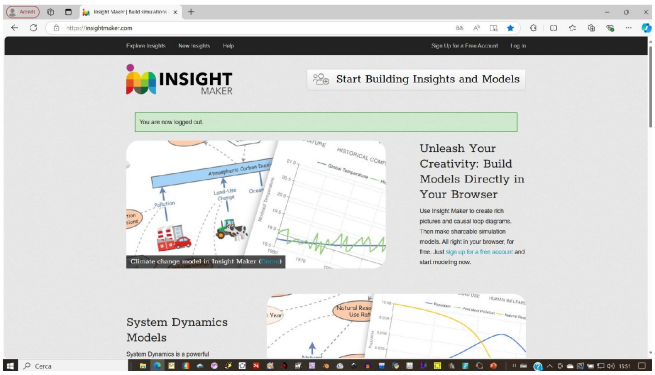
$$\vec{F} = m\vec{a} = m\ddot{\vec{z}} \qquad \vec{F} = m\ddot{\vec{z}} = \frac{d}{dt}m\dot{\vec{z}} = \frac{d}{dt}m\vec{v} = \frac{d\vec{p}}{dt} = \dot{\vec{p}} \tag{2.1}$$

ecco l'equazione del prim'ordine cercata, in $\vec{p}$. Infatti:

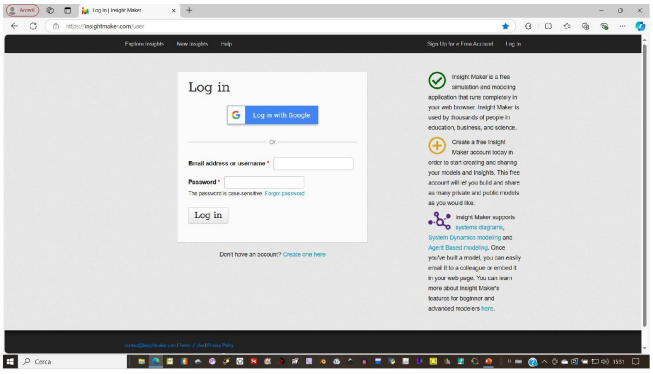
- $\vec{F}$ è il tasso di variazione di $\vec{p}$ nel tempo, componente per componente: $\vec{F}$ si misura in N, $\vec{p}$ in $kg\frac{m}{s} = N \cdot s$ ($\vec{p}$ è l'integrale di $\vec{F}$ in t: $\vec{p}(t) = \int_0^t \vec{F}(t)dt$);
- $\vec{p}$ è *estensiva*: *può vedersi come immagazzinata dal corpo*, per l'azione di $\vec{F}(t)$;
- $\vec{p}$ è *conservata*: non studiamo come Terra e aria si muovano scambiando $\vec{p}$ con il corpo, ma *se ci sono solo forze interne, la somma di tutte le quantità di moto deve restare pari al valore iniziale*. Altrimenti il modello, o il calcolo, è errato.

Vediamo quindi come si traduce l'equazione differenziale $\dot{\vec{p}} = \vec{F}$ in un modello. Da *Primitive*, selezioniamo *Add Stock* (Fig. 2.2(a)). Per *Stock* s'intende *il valore numerico della grandezza fisica che compare come derivata prima nell'equazione*: qui, la componente z della quantità di moto $\vec{p}$. Nominiamo quindi QdM lo stock[1], e specifichiamo nella finestra a destra (Fig. 2.2(b)):

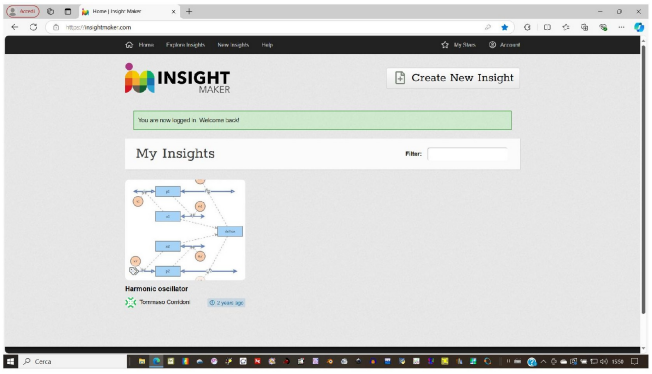
[1]Sarebbe decisamente più opportuno nominarlo pz! Si sta cercando tuttavia di simulare anche la crescita che vivrebbe una persona che apprende: lavorando autonomamente on-line, potrà rivedere molte volte il modello, fino a decidere che QdM non è la scelta migliore.



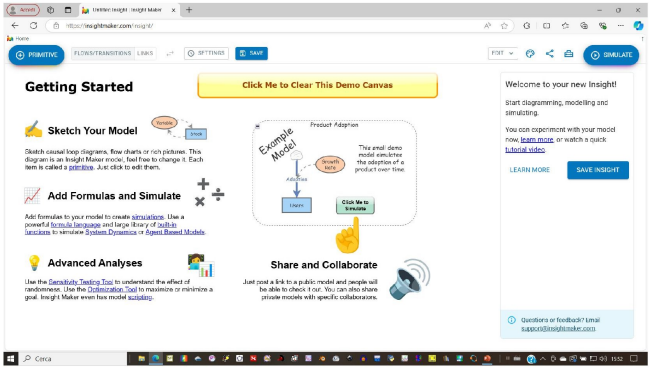
(a)



(b)

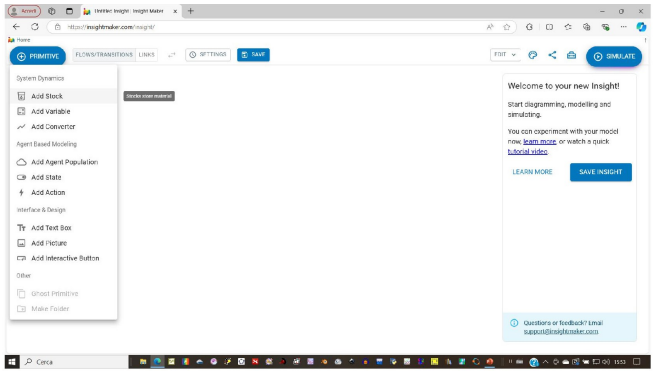


(c)

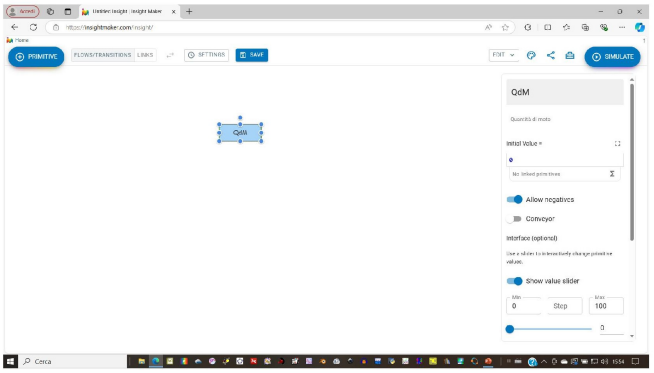


(d)

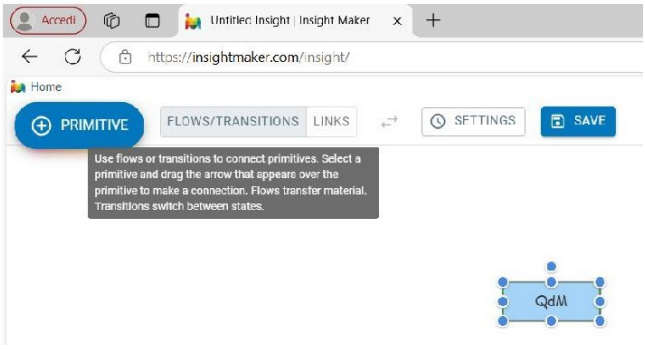
Figura 2.1: 2.1(a): Pagina iniziale di Insightmaker; 2.1(b): log in; 2.1(c): un account, con un modello; 2.1(d): pagina di consigli quando si crea un nuovo modello.



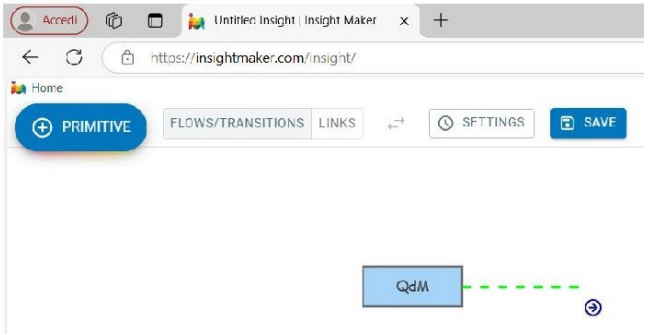
(a)



(b)



(c)



(d)

Figura 2.2: 2.2(a): Aggiungiamo uno Stock; 2.2(b): Diamo allo Stock un nome, una unità di misura, un valore iniziale; 2.2(c): Aggiungiamo un flow: distinzione dal link; 2.2(d): Cliccando sulla freccia e spostandosi tenendo premuto, esce il connettore verde....

- l'*unità di misura*, da selezionarsi se presente fra quelle *metriche*. Se assente, va definita perché c'é un *controllo di coerenza*. Nel caso, al posto di *unitless* occorre scrivere *Newton*Second* o *Newtons*Seconds*, usando solo i nomi suggeriti per le unità fondamentali;

- il *valore iniziale* della grandezza, inserito direttamente o da variabile esterna, meglio se attivando uno *slider* (*Show value slider*).

Rappresentata p_z come uno Stock, *caratterizziamo ora F_z come il suo tasso di variazione nel tempo, ossia la sua derivata prima, rappresentata mediante una freccia che entra (o esce, vedremo subito) nello Stock*, detta *flow*, ossia flusso. A tal fine:

- selezioniamo *flow/transitions* (Fig. 2.2(c)), clicchiamo sulla freccia nello Stock e "'tiriamo"' verso destra...;

- ...compare un segmento tratteggiato verde (Fig. 2.2(d)), che, rilasciando il mouse, diventa una freccia (Fig. 2.3(a));

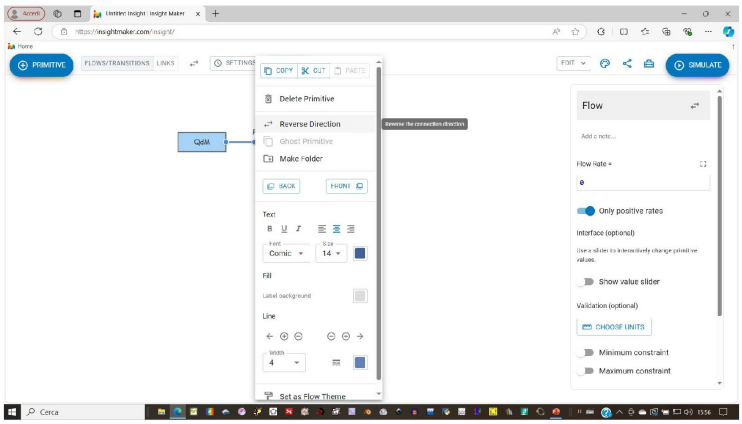
- caratterizziamo il flow con nome, unità di misura e soprattutto *verso* (Fig. 2.3(b), o $\rightleftharpoons$ a sinistra di *settings*). In questo caso la forza non cambia segno nel tempo, così il flusso può scegliersi unidirezionale. Per una forza ad es. elastica sarebbe *necessario* sceglierlo bidirezionale.

In Fig. 2.3(c) vediamo come l'equazione differenziale $\dot{p}_z = F_z$ sia stata tradotta in uno Stock p_z "'caricato"' da un flusso F_z. Manca tuttavia l'insieme dei *parametri* e delle altre variabili, che il programma chiama collettivamente *variables*, necessarî/e a specificare il valore di F_z: la massa m e l'intensità del campo gravitazionale g_T. A tal fine, selezioniamo *primitive* e poi *add variable* (Fig. 2.3(d)).

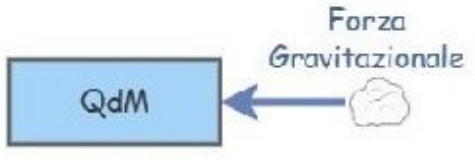
Le variabili si caratterizzano in nome, unità e valore come gli stocks, ma *mentre gli stocks sono sempre il risultato di integrazioni numeriche, le variabili no*: sono costanti o risultati di formule algebriche. Nel nostro caso sono una massa m, cui daremo un valore costante in *Kilograms* (volendo, da slider), e l'intensità del campo gravitazionale $g_T = G\frac{M_T}{R_T^2} \approx 9.81\frac{N}{kg}$, la cui unità di misura va scritta esplicitamente. Si noti che g_T è *scelta costante* proprio perché considerando la massa m del corpo molto minore di quella della Terra M_T, ed il suo spostamento molto minore del raggio della Terra R_T, non utilizziamo la legge di gravitazione universale. Per trovare orbite paraboliche, ellittiche o iperboliche dovremo farlo prossimamente, definendo g_T mediante i parametri G, M_T, R_T. Ma per ora l'obiettivo è mostrare che *si classificano come variabili sia i parametri del sistema, sia grandezze variabili nel tempo che non rappresenteremmo come stoks perché non estensive, o non derivate prime in un'equazione differenziale*.

Passando al software, nelle Figg. 2.4 vediamo come aggiungere e caratterizzare le due variabili, mentre nelle Figg. 2.5 vediamo che, una volta selezionato il tasto *link* (a lato di *flow*), si usa la freccia già vista nello stock, presente anche nelle variabili, per connetterle al flow della forza F (prima appare la linea tratteggiata verde).

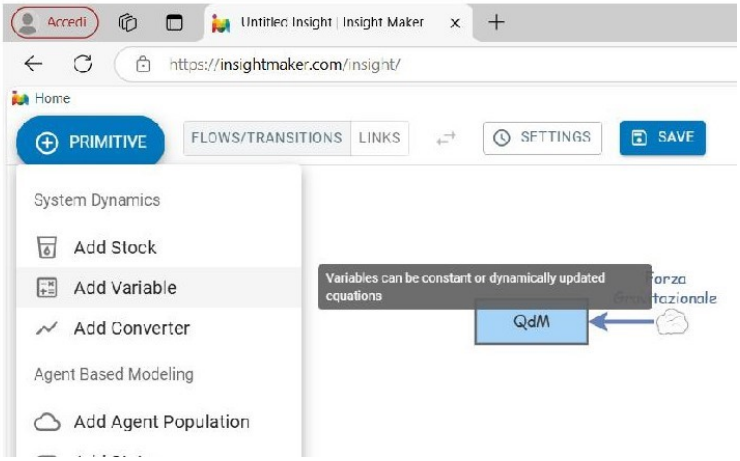
(a)



(b)

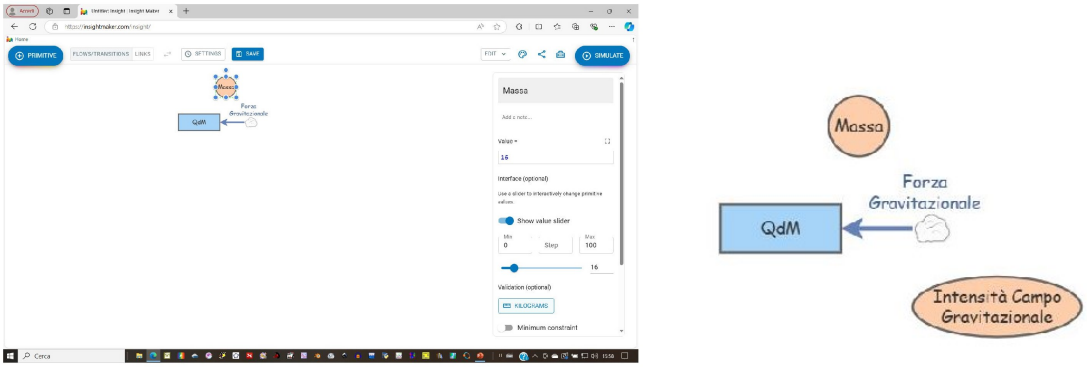


(c)



(d)

Figura 2.3: 2.3(a): ...rilasciato, il connettore verde diventa un flow; 2.3(b): caratterizziamo il flow: nome, unità di misura, verso, (cosa importantissima); 2.3(c): manca il valore del flow: $F = mg_T$; 2.3(d): Aggiungiamo delle Variables.



(a) (b)

Figura 2.4: 2.4(a): Caratterizziamo la variabile massa m: nome, unità, valore; 2.4(b): Aggiungiamo (e caratterizziamo: non mostrato) la variabile campo gravitazionale g_T.

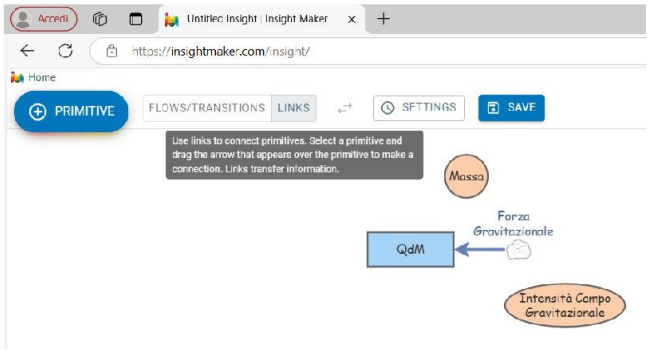
I link hanno verso ma non hanno nomi, perché restano connettori logici (Fig. 2.5(c)), da non confondere mai con i flow. Il loro ruolo è dire ad IM quali grandezze sono necessarie per scrivere algebricamente il valore numerico di variabili, stock o flow, nel caso F come prodotto di m e g_T (Fig. 2.5(d)).

Fatto questo, il modello finale di Fig. 2.5(e) è pronto a descrivere sotto forma di grafico o tabella come aumenta nel tempo la quantità di moto p_z di un oggetto di massa m soggetto all'azione del campo gravitazionale g_T.

Selezionando *simulate* (la prima volta) oppure *settings* (le successive), compare però prima una finestra importantissima (Fig. 2.6(a)). Come ben noto esistono molti *metodi numerici* per approssimare un integrale, ossia una somma al continuo, con una sommatoria discreta:

$$\dot{p}_z = F_z(t) \quad \Rightarrow \quad p_z(t) = \int_0^{t_{max}} F_z(t)dt = \sum_{i=0}^{N} F_z(t_i) + \epsilon(N) \tag{2.2}$$

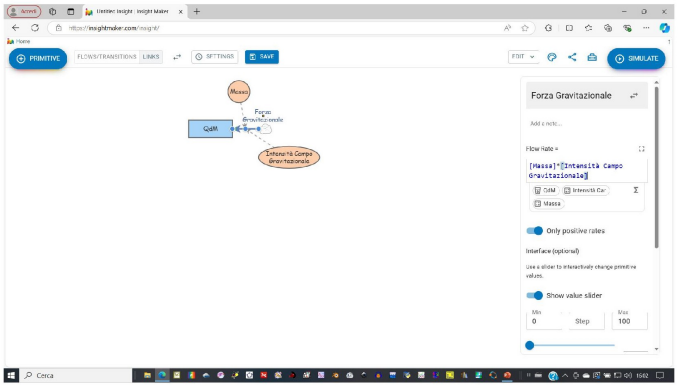
Tutti questi metodi implicano un certo errore ϵ che dipende in genere dai valori del *tempo totale di simulazione* t_{max}, dal *passo di simulazione* dt, ossia da come il tempo t_{max} è suddiviso in N intervalli più o meno "'istantanei"' (dt è considerato in genere costante, ma può non esserlo), e chiaramente *dall'algoritmo* con il quale sono approssimati gli elementi della somma. IM chiede pertanto t_{max}, dt (in unità da scegliere: qui Seconds) e propone solo due algoritmi: quello *di Euclide*, il più elementare ed impreciso ma molto rapido, e quello *di Runge e Kutta d'ordine 4*, il miglior compromesso fra velocità di esecuzione e minimizzazione degli errori [31]. In genere, a seconda del fenomeno, si sceglie inizialmente un t_{max} dell'ordine di 1.0-5.0s ed un dt abbastanza piccolo (0.01s, 0.001s), per una simulazione abbastanza dettagliata ma veloce. Si può selezionare prima l'algoritmo di Euclide solo per vedere se il modello è formalmente corretto (unità, link...), passando poi al Runge-Kutta4 se, come spesso accade, i risultati numerici non sono attendibili (ad esempio, ci sono divergenze esponenziali). Scelti i parametri di simulazione il modello va salvato da *edit info*, decidendo se è pubblico o meno (Fig. 2.6(b)).



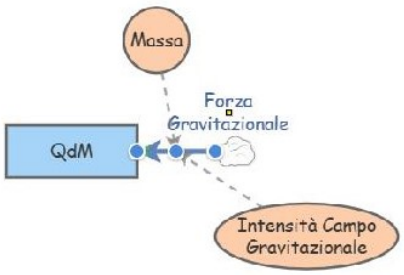
(a)

(b)

(c)

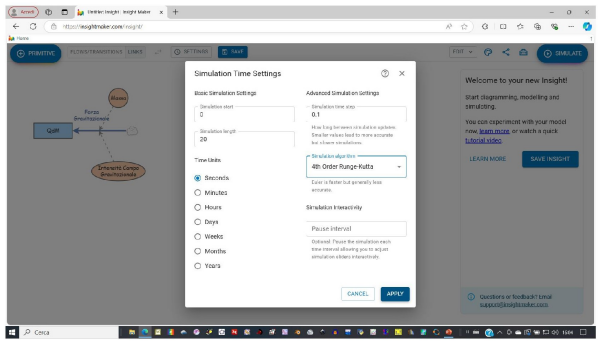


(d)

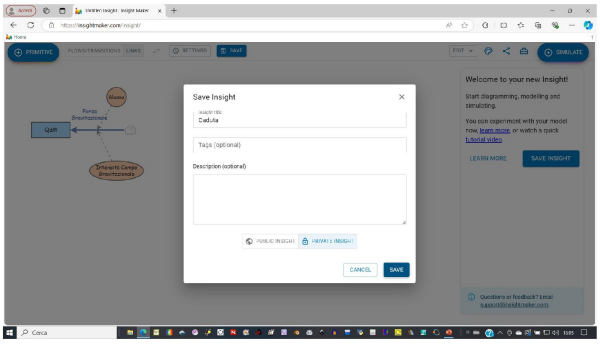


(e)

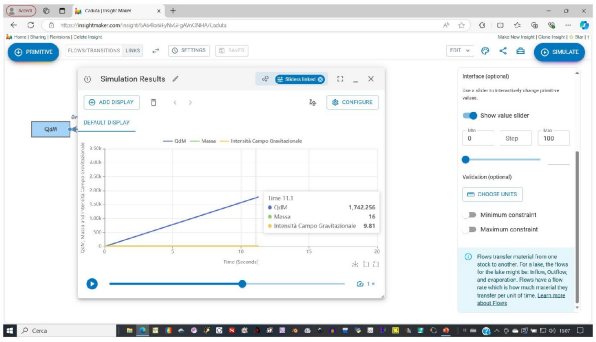
Figura 2.5: 2.5(a): Selezioniamo i link (diversi dai flow); 2.5(b): connettiamo la forza gravitazionale all'intensità del campo; 2.5(c): Caratterizziamo il link; 2.5(d): una volta che F è connessa a m e g_T, specifichiamo il valore $F = m \cdot g_T$. 2.5(e): Un modello per la caduta nel campo gravitazionale.



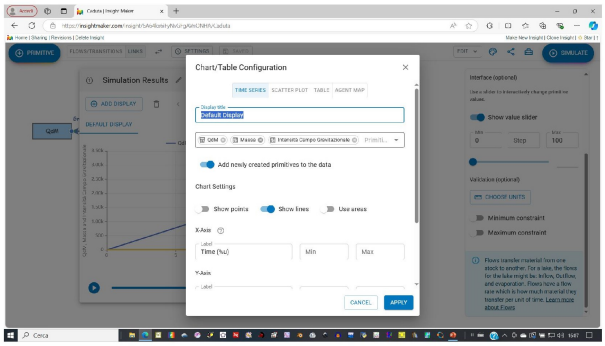
(a)



(b)



(c)



(d)

Figura 2.6: 2.6(a): Scelta dei parametri di simulazione: tempo totale, passo dt di integrazione, metodo numerico di integrazione; 2.6(b): SALVA SEMPRE IL MODELLO! 2.6(c): Risultato della simulazione: p_z cresce linearmente nel tempo, le variabili, costanti, no; 2.6(d): selezione delle grandezze da visualizzare sul grafico: non ci interessano m e g_T.

Fig. 2.6(c) mostra il risultato della simulazione: sotto l'effetto della forza costante $F_z = mg_T$, la componente $p_z = mv_z$ della quantità di moto del corpo, che parte da fermo, cresce linearmente come $p_z = mv_z = mg_T \cdot t$ (in basso a destra nel grafico, dov'è ∞, si seleziona la velocità di comparsa dei dati).

Sta ora a chi modellizza raffinare ed interpretare. Innanzi tutto IM propone su una stessa scala i grafici delle variabili e degli stocks, da selezionare e separare (Fig. 2.6(d)). Poi, è evidente che fin dall'inizio *è stato dato per scontato* che l'asse $\vec{z}$ sia orientato verso il basso, così da avere $p_z > 0$. Si tratta di una scelta convenzionale, ma per IM è il risultato della selezione del verso di F_z come flow... *Questi aspetti devono essere sempre considerati in una simulazione dinamica, in quanto spesso gli errori vengono dal fatto che un PC non può capire cose per noi sottintese.*

Per avere i risultati sia in forma grafica (come file .png), sia come tabella numerica (.csv), occorre scegliere queste opzioni in *configure*. Si possono avere anche grafici xy, in particolare lo *spazio delle fasi*, ponendo la quantità di moto p_z in funzione della posizione z, non ancora definita però come stock.

Introduciamo ora l'attrito viscoso in modo da far comprendere la ragione della seconda delle domande iniziali (come preservare aspetti non solo algebrici). La forza di attrito viscoso è

$$F_{a,z}(t) = -\lambda \cdot v_z(t) = -\lambda \frac{p_z(t)}{m} \tag{2.3}$$

da definirsi introducendo e connettendo con link due variabili: la componente z della velocità, $v_z(t)$ (Fig. 2.7(a)), definita come rapporto $\frac{p_z}{m}$ (Fig. 2.7(b)), ed il coefficiente di attrito viscoso, λ, costante (Fig. 2.7(c)). Dal punto di vista fisico è essenziale *considerare $F_{a,z}$ come un flow in uscita dallo stock p_z*, ottenuto cliccando su *flow/transitions* e poi sulla freccia nello stock QdM, da "'tirare"' in verso contrario a F_z (Fig. 2.7(a)). È infatti quanto afferma il 2. principio della dinamica:

$$\dot{p}_z = F_z - F_{a,z} = mg_T - \lambda \cdot v_z(t) \tag{2.4}$$

se la forza gravitazionale porta quantità di moto nel sistema mediante il campo, prendendola dalla Terra, la forza di attrito gliela sottrae cedendola all'aria. Non a caso, l'unità di misura di λ è $\frac{Ns}{m}$, ossia quantità di moto (in Ns) persa in un metro di percorso. Pertanto, *la rappresentazione delle due forze come flussi in entrata ed uscita è equivalente al diagramma vettoriale delle forze che sempre viene illustrato* (il particolare per discutere la velocità limite).

Si noti anche la velocità istantanea come *variabile*. Matematicamente si è soliti pensarla come stock, essendo $\dot{v}(t) = a$. Ma d'altra parte la grandezza che abbiamo preferito perché estensiva e conservata, la quantità di moto, è comunque legata ad essa, sia da una *legge capacitiva*, $p = mv$, che considera *come un corpo la immagazzina*, sia da una *legge conduttiva* (fra le possibili), che considera *come viene trasportata* in uscita (Fig. 2.7(c)):

$$p = mv \qquad \dot{p}_{persa} = -\lambda v_y(t) \tag{2.5}$$

Scopriamo così come la modellizzazione dinamica metta in evidenza l'organizzazione delle relazioni fisiche, in questo caso

- *caratterizzate come leggi capacitive o conduttive;*
- *espresse in forma differenziale (istantanea), come nei principi della dinamica, o integrale (su tempi finiti), come nei principi di conservazione.*

Fig. 2.8 mostra il risultato della simulazione: la quantità di moto tende ad un valore asintotico, in quanto il corpo ne acquista e ne perde fino a che il bilancio degli scambi è nullo, ed essa resta costante. Sullo stesso grafico è riportata anche la velocità, che ha un ordine di grandezza diverso: può riportarsi su un'altra scala.

A questo punto si vuole trattare anche l'*energia* dell'oggetto. Quella *totale*, estensiva e conservata, si rappresenta come stock. Si consideri inizialmente la sola energia cinetica. La sua variazione nel tempo, intesa come *potenza*, è composta da due "'flussi"': una *potenza disponibile*, data dal prodotto della forza gravitazionale per la velocità istantanea, che va ad incrementare l'energia cinetica del corpo in caduta, ed una *potenza dissipata*, data dal prodotto della forza d'attrito viscoso per la velocità istantanea, che va a diminuirla:

$$E_{disp} = \int_0^t P_{disp}(t)dt = mg_T \int_0^t v(t)dt \quad E_{diss} = \int_0^t P_{diss}(t)dt = \int_0^t F_a(t)v(t)dt \quad (2.6)$$

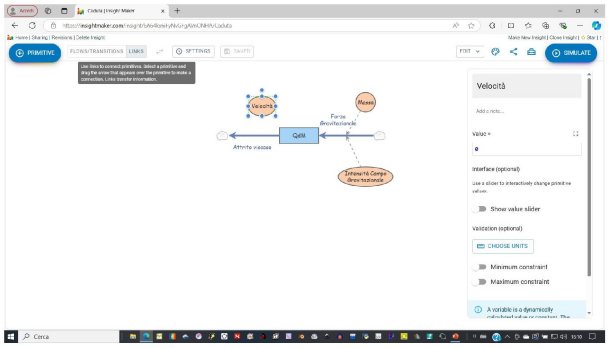
Fig. 2.9(a) mostra la modellizzazione dell'energia cinetica in questa rappresentazione: uno stock con la potenza disponibile in entrata e quella dissipata in uscita, definite come flow[2] e calcolate mediante link a forze e velocità. Nel suo grafico nel tempo c'è un asintoto (Fig. 2.9(b)) ma anche un flesso (Fig. 2.9(c)).

Fig. 2.9(d) mostra infine una versione migliorata del modello, disponibile a `https://insightmaker.com/insight/6A64kx6iFyNvGFgAVnONHA`. In essa la quantità di moto iniziale è introdotta come variabile p_0, scelto da slider, associata con link agli stock della *Qdm* (rinominato p_z) e dell'energia cinetica, il cui valore iniziale va definito come $p_0^2/2m$. Per non confondere troppo la rappresentazione, la variabile p_0 è *copiata graficamente* cliccando il tasto destro del mouse su di essa e scegliendo *ghost primitive*.

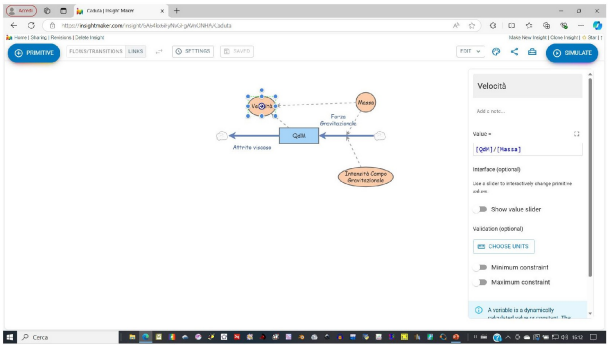
Le due piccole icone in alto a destra sui grafici permettono altri controlli di coerenza. Quando si attiva la prima, *explore the results on the model icons*, passando sulle parti del modello appaiono i relativi grafici nel tempo. Quando si attiva la seconda, *slider linked*, variando da slider i parametri m, p_0, λ (il cui ordine può variarsi con *reorder*), *il grafico è aggiornato in tempo reale*. Se partendo da zero si aumenta la quantità di moto iniziale p_0, si scopre così che se il corpo comincia a cadere con velocità superiore a quella limite, nel tempo rallenta fino a raggiungerla.

Sebbene ancora incompleto e da migliorare, il modello è già di per sé una *mappa concettuale* finalizzata a riconoscere nella sua stessa rappresentazione grafica la *struttura logica della fisica del sistema*, oltre a quella algebrica. Servono ovviamente

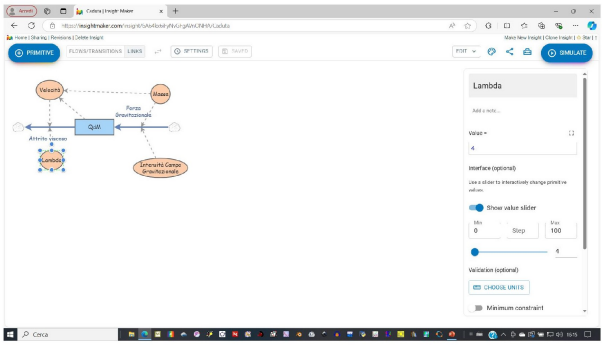
[2]In tre dimensioni, ogni flusso, scalare, sarebbe la componente di un vettore.



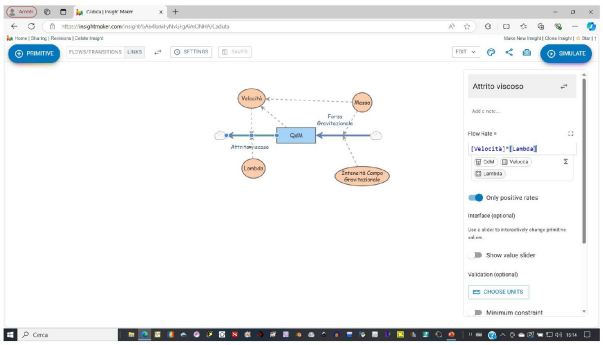
(a)



(b)

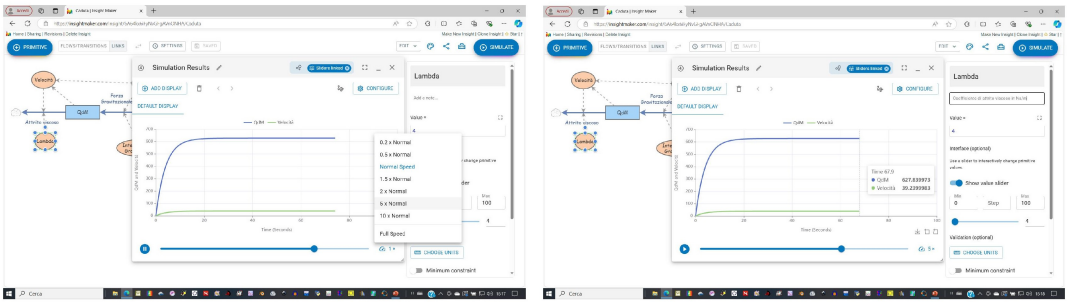


(c)



(d)

Figura 2.7: 2.7(a): Introduciamo la forza di attrito come flow in uscita e la velocità come variabile; 2.7(b): link, $v = \frac{p}{m}$; 2.7(c): introduciamo λ come variabile; 2.7(d): definiamo finalmente $F_a = \lambda v$; il segno è nel verso del flow.



(a) (b)

Figura 2.8: Quantità di moto e velocità in un corpo in caduta con attrito viscoso: 2.8(a): selezione velocità di calcolo; 2.8(b): controllo dei valori ad un certo istante.

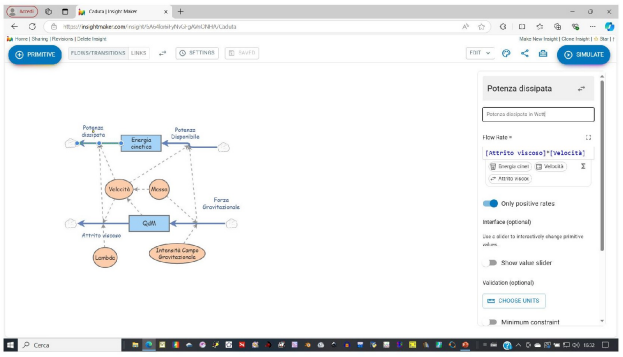
ancora molti altri controlli di coerenza: introducendo ad esempio la variabile $\frac{1}{2}mv^2$, essa risulta identica istante per istante all'energia cinetica, sebbene ciò non sia stato imposto algebricamente. Analogamente, si potrà introdurre la quota z dell'oggetto come stock, al fine di mostrare, introducendo anche l'energia potenziale gravitazionale, come l'energia totale sia conservata istante per istante, sebbene nessuna equazione algebrica lo imponga. Fatto ciò, *è infatti particolarmente istruttivo (cambiando da settings la durata della simulazione ed il passo di integrazione) imparare a valutare l'affidabilità della simulazione in base a se e quanto siano accettabili le approssimazioni numeriche sui valori delle grandezze conservate*. Una simulazione come questa, è affidabile solo se la somma di energia potenziale, cinetica e dissipata, mantiene istante per istante un valore molto prossimo a quello calcolabile dalle condizioni iniziali. In Fig. 2.10 si mostrano il modello completo e una rappresentazione del bilancio dell'energia, ossia gli andamenti nel tempo dell'energia totale e delle sue tre parti: energia cinetica, potenziale gravitazionale e dissipata. Come si vede, l'energia totale è conservata durante tutta la dinamica (Fig. 2.10(b)).

Da notare infine un ultimo ma ulteriore passo in avanti: impostando una quantità di moto iniziale del corpo p_{0z} come negativa (basta mettere un estremo inferiore negativo nello slider relativo), di fatto si è simulato un oggetto che *viene lasciato cadere dopo essere stato lanciato verso l'alto* (l'asse z è orientato positivamente verso il basso). In genere chi impara ha notevoli problemi ad accettare da subito che le equazioni che regolano il moto di un oggetto siano le stesse, sia che sia lanciato verso l'alto, sia che sia lanciato verso il basso. Il dynamical modeling lo dimostra invece immediatamente, in quanto i due casi sono simulati da un unico modello: per passare dall'uno all'altro, basta semplicemente cambiare il segno di p_{0z}.

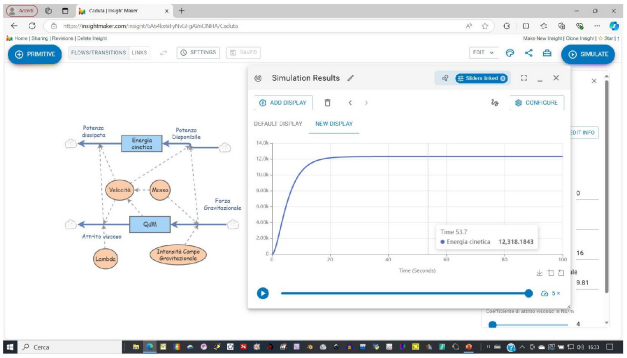
2.2.3 Il problema dei 2 corpi quando uno è fermo

Moto del pianeta nel riferimento in cui il Sole è fermo

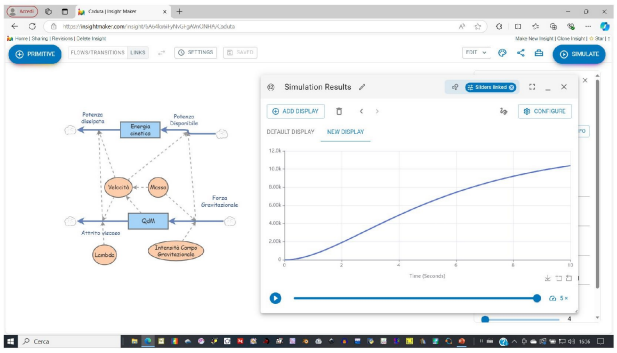
Introdotte nel Par. 2.2.2 le principali funzionalità di IM, lo si utilizza ora per i modelli del moto caratteristici del problema dei due corpi, soggetti ad interazioni



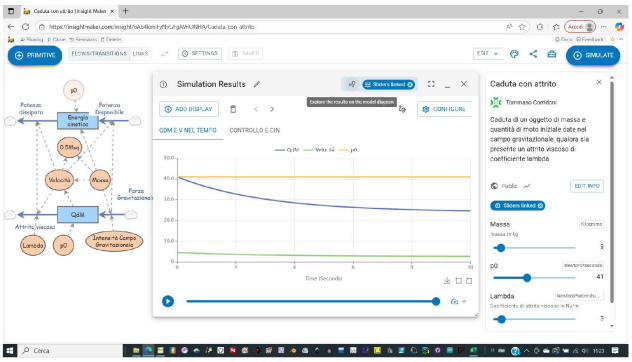
(a)



(b)

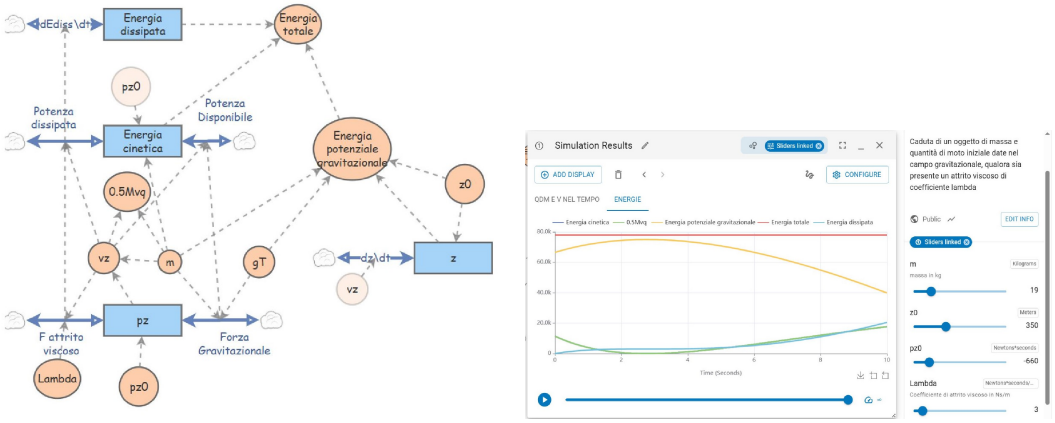


(c)



(d)

Figura 2.9: 2.9(a): caratterizzazione dei flussi di energia; 2.9(b): energia cinetica del corpo nel tempo: appaiono un asintoto...; 2.9(c):...ed un flesso; 2.9(d): modello finale con commenti, condizioni iniziali da slider e controllo che l'energia cinetica sia $\frac{1}{2}mv^2$ istante per istante (`https://insightmaker.com/insight/6A64kx6iFyNvGFgAVnONHA`).



(a) (b)

Figura 2.10: Modello completo della caduta in un mezzo viscoso: 2.10(a): modello con controllo dell conservazione dell'energia totale; 2.10(b): bilancio dell'energia. Modello disponibile a `https://insightmaker.com/insight/6A64kx6iFyNvGFgAVnONHA`.

solo gravitazionali. Prendendo spunto da esempi in `Stella` [22], nelle Figg. 2.11 si mostra il modello IM (Fig. 2.11(a)) e la situazione dei due pianeti al tempo iniziale: un pianeta di massa m_2 che, partendo da un punto $(x_0, 0)$ con quantità di moto iniziale $(p_x(0), p_y(0))$ si appresti a ruotare attorno ad un Sole di massa M_1 fermo nell'origine degli assi, *posti già nel piano dell'orbita*. In Fig. 2.11(b) si definiscono alcune delle grandezze utilizzate nel modello e in Fig. 2.11(c) le condizioni di integrazione: simulazione con algoritmo Runge-Kutta del 4. ordine su un tempo di 2.0s, con passo di $2.0{\cdot}10^{-3}$s.

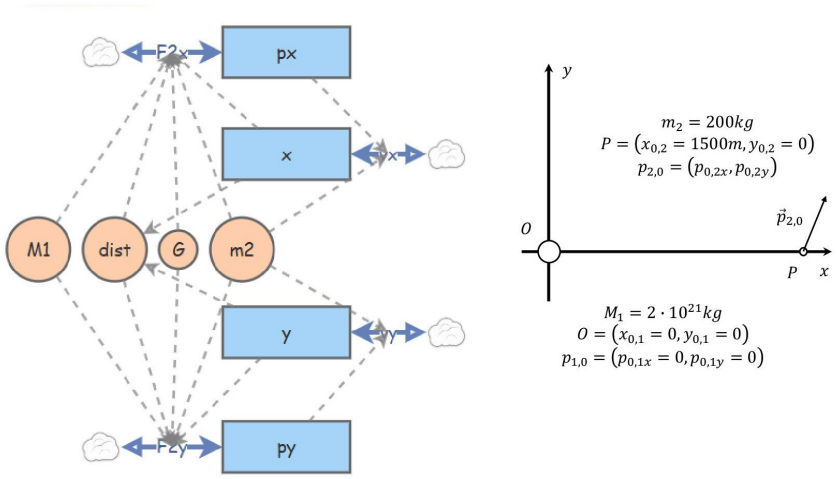
Nel modello, le componenti (p_x, p_y) del vettore quantità di moto del pianeta, i cui valori iniziali sono scelti da slider, sono indicate come Stocks. In essi entrano flows che rappresentano le componenti F_x, F_y della forza del Sole sul pianeta:

$$\frac{dp_x}{dt} = F_x = -G\frac{M_1 m_2 x}{(x^2+y^2)^{\frac{3}{2}}} \qquad \frac{dp_y}{dt} = F_y = -G\frac{M_1 m_2 y}{(x^2+y^2)^{\frac{3}{2}}} \tag{2.7}$$

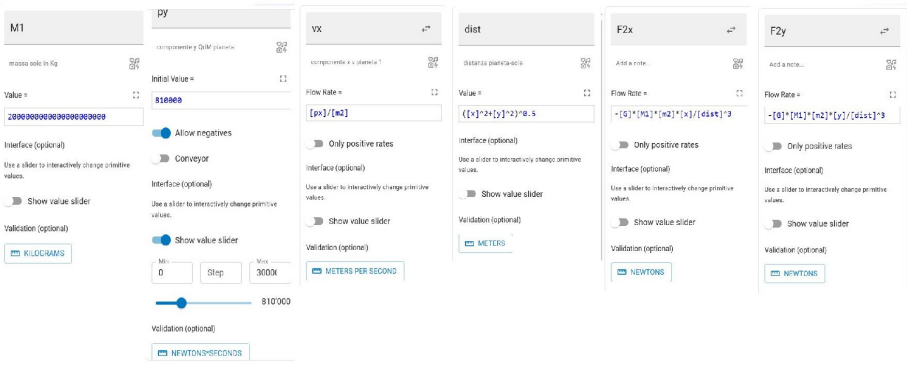
La costante di gravitazione universale $G = 6.67 \cdot 10^{-11}\frac{Nm^2}{kg^2}$ e le masse M_1, m_2 sono date, ed è introdotta la funzione *dist*anza $\sqrt{x^2+y^2}$.

Nella definizione delle grandezze è essenziale stabilire da *Allow negatives* se possono assumere valori negativi o meno (Fig. 2.11(b)). In questo caso, in cui tutto deve poter cambiare segno, imporre la positività porterebbe ad errori evidenti. Altro aspetto cui fare attenzione è *l'abilitazione del doppio verso nei flow*, in particolare nelle forze: quando lo si fa, in essi appare una *freccia bianca* ad indicare il verso dei flussi positivi. Nel caso gravitazionale tuttavia la forza attrattiva richiede un segno negativo, dunque la scelta dei versi va fatta come indicato in Fig. 2.11(a), da cui anche i segni delle velocità. In fase di ottimizzazione del modello, *per cambiare subito un verso su cui si hanno dubbi* è utile il comando *Reverse connection*.

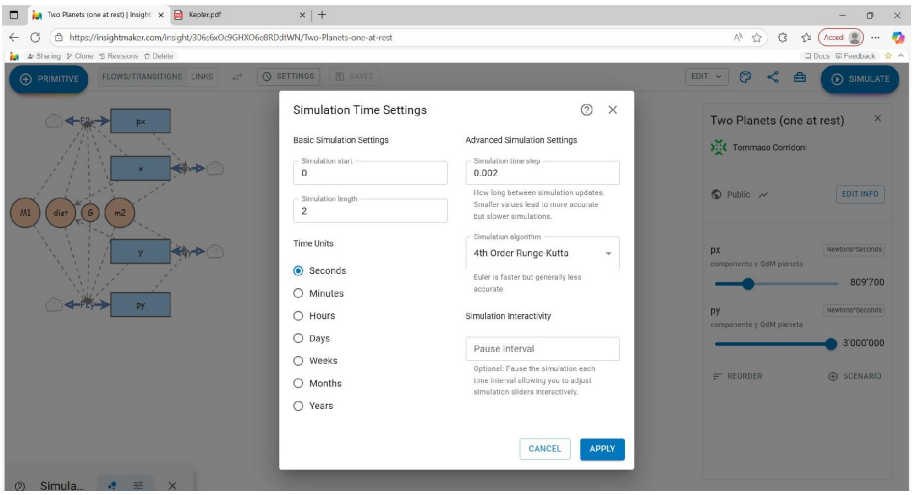
Dalle componenti della quantità di moto, mediante la massa m_2, si passa poi alle componenti (v_x, v_y) della velocità, a loro volta rappresentate da flussi in entrata



(a)



(b)



(c)

Figura 2.11: 2.11(a): Modello IM di un pianeta di massa m_2 che parte da un punto $(x_0, 0)$ con quantità di moto iniziale (p_{0x}, p_{0y}), e ruota attorno ad un Sole di massa M_1 fermo nell'origine degli assi; 2.11(b): Caratterizzazione di diversi parametri e slider; 2.11(c): *setting* dell'algoritmo di integrazione. Il modello è disponibile a `https://insightmaker.com/insight/306s6xOc9GHXO6e8RDdtWN`

negli Stocks delle coordinate (x, y):

$$\frac{dx}{dt} = v_x = \frac{p_x}{m_2} \qquad \frac{dy}{dt} = v_y = \frac{p_y}{m_2} \tag{2.8}$$

Le coordinate sono infine connesse a loro volta alle componenti della forza, calcolate mediante le Eq. 2.7. A questo punto tutto è pronto per simulare la traiettoria del corpo di massa m_2 nel piano (x, y), oppure gli andamenti nel tempo delle coordinate $x(t), y(t)$ o delle componenti della quantità di moto $p_x(t), p_y(t)$, da aggiornare automaticamente cambiando i parametri da slider (dopo aver attivato *slider linked*).

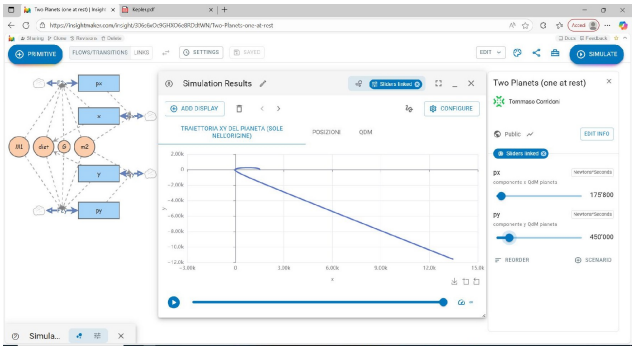
I risultati, qualitativamente molto buoni, vanno tuttavia interpretati da un punto di vista più rigoroso. Si considerino in effetti le Figg. 2.12, dove si mostrano le traiettorie al variare dei valori iniziali delle componenti $p_x(0), p_y(0)$, ottenuti con le *settings* in Fig. 2.11(c). In Fig. 2.12(a) il pianeta attratto dal Sole, fermo nell'origine, vi cade sopra. La traiettoria successiva all'urto è quasi certamente priva di senso: l'algoritmo di integrazione ha incontrato un denominatore quasi nullo nell'espressione delle forze, determinando approssimazioni incontrollabili dei numeri. Non solo: *cambiando le condizioni di integrazione, in particolare il tempo totale di simulazione ed il passo di integrazione, è facile osservare come cambino anche le forme delle orbite: una traiettoria che potrebbe benissimo esistere, per la simulazione può risultare impossibile.*

Un controllo attento di tale eventualità deve essere fatto calcolando l'energia E ed il (modulo del) momento angolare L del pianeta (conviene usare un foglio elettronico), ricordando che *esistono orbite solo se l'energia totale è maggiore o uguale a quella di un moto circolare* (Eq.1.18):

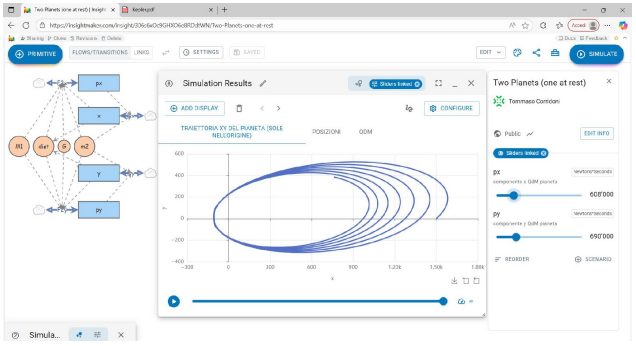
$$E = \frac{1}{2} m_2 \left(v_{0x}^2 + v_{0y}^2\right) - \frac{G M_1 m_2}{\sqrt{x_0^2 + y_0^2}} \geq E_{min} = -\frac{G^2 M_1^2 m_2^3}{2L^2} \quad |L| = |\vec{p} \times \vec{x}| = |p_y x - p_x y| \tag{2.9}$$

Particolarmente insidioso è quindi il caso di Fig. 2.12(b), ottenuto ancora nelle condizioni di Fig. 2.11(c): l'orbita sembra portare alla caduta a spirale del pianeta sul Sole, *ma calcolando il valore dell'energia risulta che l'orbita ellittica dovrebbe esistere*. Il problema è nella precisione dell'algoritmo: riducendo il tempo ed il passo di integrazione a parità di condizioni iniziali, si ottiene l'orbita cercata. In questi casi "'sospetti"', modificando il modello come in Fig. 2.13(a), si può verificare l'attendibilità della simulazione dal grafico di E ed L nel tempo, grandezze che, sebbene non sia stato imposto, *sono conservate* nella dinamica. La simulazione dà quindi risultati affidabili quando i moduli di E ed L hanno variazioni relative accettabili, dovute alle sole approssimazioni numeriche nel perielio (Fig. 2.13(a)).

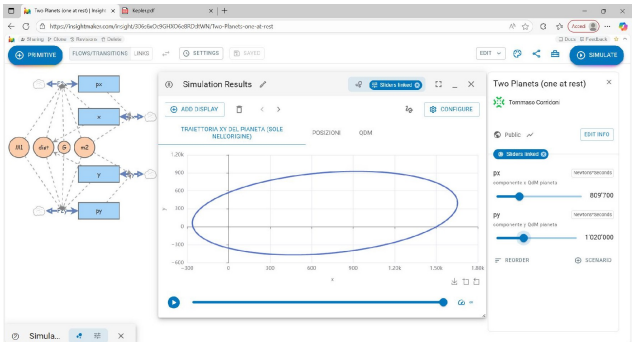
Stabilite le condizioni di integrazione ottimali, si osserva l'evoluzione della forma dell'orbita all'aumentare della quantità di moto iniziale del pianeta: inizialmente esso cade sul Sole, ha poi un'orbita circolare (difficile selezionarla, date le approssimazioni), poi ellittica (Fig. 2.12(c)), successivamente parabolica ed infine iperbolica (Fig. 2.12(d)). Occorre certamente discutere come questa evoluzione sia *qualitativamente corretta ma quantitativamente dipendente dalle scelte sull'integrazione*,



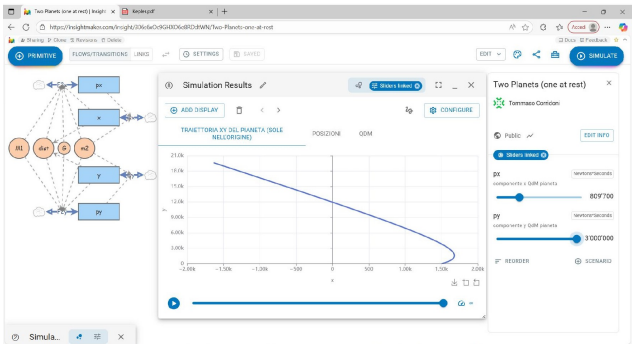
(a)



(b)

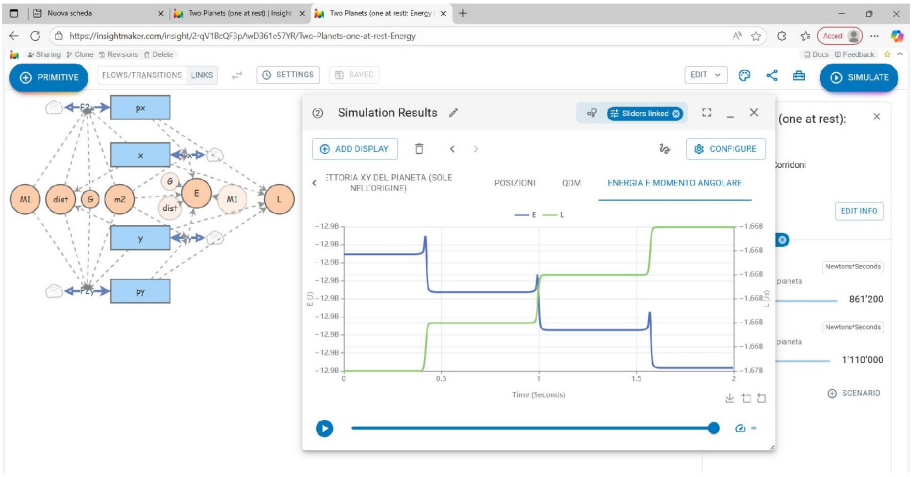


(c)

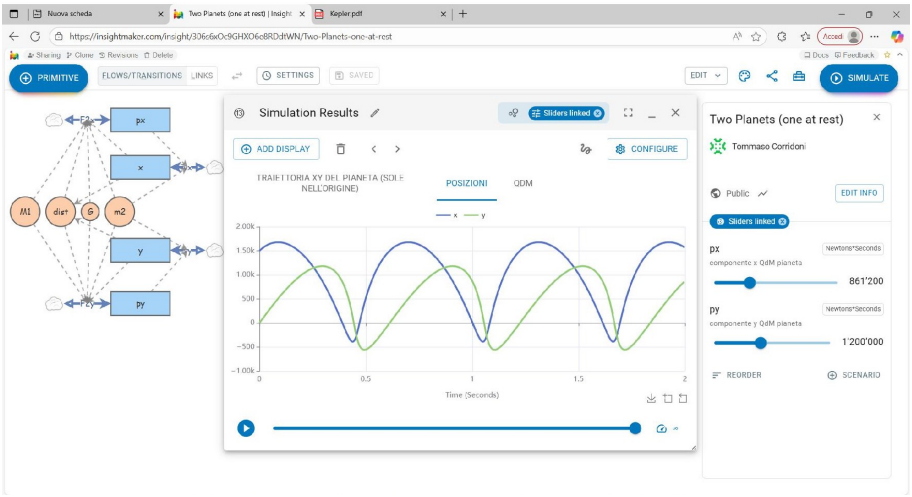


(d)

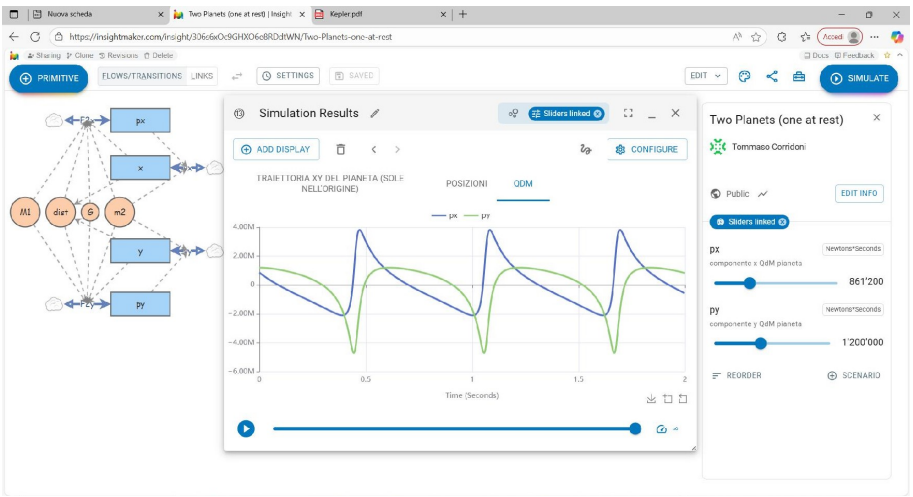
Figura 2.12: Orbite ottenute dal modello di Fig. 2.11(a) al variare della quantità di moto iniziale del pianeta: 2.12(a): il pianeta non riesce a scappare e cade sul Sole; 2.12(b): spirale ellittica di caduta del pianeta sul sole, che si rivela falsa al controllo dell'energia; 2.12(c): orbita ellittica coerente con la teoria; 2.12(d): orbita iperbolica. `https://insightmaker.com/insight/306s6xOc9GHXO6e8RDdtWN`



(a)



(b)



(c)

Figura 2.13: Fig. 2.13(a): modifica del modello per calcolare le grandezze conservate E ed $|L|$ nel tempo: la loro variazione, presente ma minima (si notino le due scale), è indice di affidabilità della simulazione. Figg. 2.13(b), 2.13(c): evoluzione nel tempo delle coordinate e delle velocità del pianeta nelle direzioni x, y, su un'orbita ellittica. `https://insightmaker.com/insight/2rqV1BcQF3pAwD361e57YR`

ma una volta ottenute orbite chiuse e ben definite la simulazione è in genere attendibile: i semiassi di ellissi come quella di Fig. 2.12(c) sono coerenti con le Eq. 1.16 (non essendo la velocità iniziale parallela ad uno degli assi, l'ellisse non è neanche riferita ad essi, come previsto), e semmai, al crescere dell'eccentricità, il problema è nella scelta automatica della scala del grafico, che distorcendo le traiettorie fa sì che l'origine possa non sembrare in un fuoco.

Ottenute orbite affidabili, è importante mostrare anche l'evoluzione, periodica (non sinusoidale) delle coordinate e delle velocità del pianeta nelle direzioni x, y (Fig. 2.13(b), Fig. 2.13(c)). Con minime variazioni del modello, è possibile anche separare nel tempo gli andamenti dell'energia potenziale e di quella cinetica.

Moto di un corpo visto dall'altro, fermo

Come discusso nel Par. 1.1.4, in un riferimento in cui uno dei due corpi sia fermo, l'Eq. 1.40 esprime l'accelerazione $\ddot{\vec{r}}_{12} = \ddot{\vec{r}}_i$ del corpo i in moto *riferita a quello j fermo*. Ciò permette di scrivere l'orbita di i come fatto nel Par. 1.1.1 anche nel riferimento (in genere) non inerziale in cui j appare fermo, *ma a patto di assegnare a j la massa efficace $m_1 + m_2$*.

Per una conferma numerica, in Fig. 2.14 si mostra lo schema di un pianeta di massa m_2 =10^8kg, con velocità e distanza iniziali $v_y = 1.0\frac{m}{s}, 25m$, in rotazione attorno ad un Sole di massa M_1 =10^{12}kg, fermo nell'origine (entrambi puntiformi). Sono riportati poi i calcoli *dell'energia e del momento angolare dell'intero sistema nel riferimento inerziale*:

$$E_{tot12} = \frac{1}{2}M_1(v_{x01}^2 + v_{y01}^2) + \frac{1}{2}m_2(v_{x02}^2 + v_{y02}^2) - \frac{C}{r_{12}} \qquad C = GM_1m_2 \tag{2.10}$$

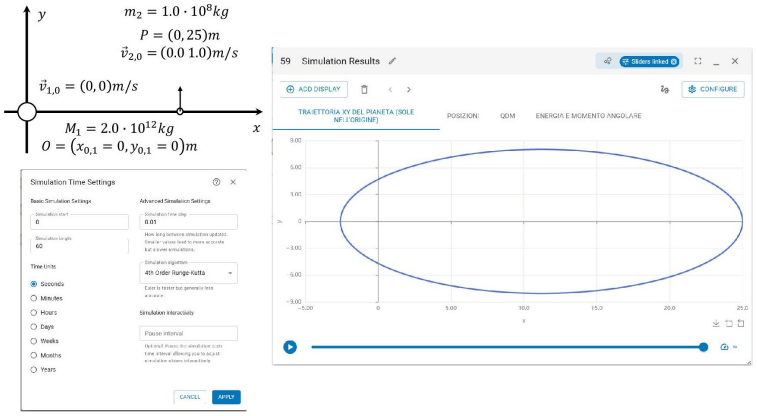
mentre $\vec{L}_i = m_i\vec{r}_i \times \vec{v}_i, \vec{L}_{tot12} = \vec{L}_1 + \vec{L}_2$. Per calcolare tuttavia le caratteristiche dell'orbita di m_2 *nel riferimento in cui M_1 è fermo*:

- si riferiscono posizioni e velocità di m_2 a quelle di M_1 (si sottraggono quelle di M_1 a quelle di m_2);
- si calcola l'energia di m_2 sostituendo M_1 con $M_1 + m_2$ in:

$$E_{21eff} = \frac{1}{2}m_2(v_{x021}^2+v_{y021}^2) - \frac{C_{21}}{r_{12}} \quad C_{21} = G(M_1+m_2)m_2 \quad E_{21eff} \neq E_{tot12}; \tag{2.11}$$

- si calcola anche il momento L_2 di m_2 con le nuove componenti della velocità, e dalle formule del Par. 1.1.1 si trovano a, b, ϵ, T per l'orbita di m_2 vista da M_1 fermo.

L'analisi dell'orbita ottenuta dal modello IM, *nel quale si assume il corpo 1 fermo e di massa M_1, non $M_1 + m_2$*, conferma tale algoritmo. Date le condizioni iniziali, l'orbita del corpo 2 è un'ellisse riferita agli assi: selezionando sul grafico *configure* e poi *table* si ottiene un file `.csv` con tutte le coordinate (t, x, y) della traiettoria, dalle quali si selezionano i dati nella tabella in basso a sinistra in Fig. 2.14(a): istanti e posizioni di afelio, perielio e massima/minima ordinata, da cui calcolare a, b, T. I valori di a, b, T così trovati (colonna IM) sono praticamente quelli calcolati a mano.



(a)

Riferimento inerziale						Riferimento M1 fermo			IM	Riferimento m2 fermo		
G	6.67E-11	Nm^2/kg^2	dist12	25	m	x011	0	m		x012	-25	m
M1	2E+12	kg	dist21	25	m	y011	0	m		y012	0	m
m2	1.00E+08	kg				vx011	0	m/s		vx012	0	m/s
C	1.33400E+10	Nm^2	Etot12	-4.83600E+08	J	vy011	0	m/s		vy012	-1	m/s
x01	0.0	m	L1	0	Nsm	x021	25	m		x022	0	m
y01	0.0	m	L2	2.50000E+09	Nsm	y021	0	m		y022	0	m
vx01	0.0	m/s	px01	0	Ns	vx021	0	m/s		vx022	0	m/s
vy01	0.0	m/s	py01	0	Ns	vy021	1	m/s		vy022	0	m/s
x02	25.0	m			Ns	C21	1.33E+10	Nm^2		C12	2.66813E+14	Nm^2
y02	0.0	m			Ns	L11	0.00E+00	Nsm		L12	5E+13	Nsm
vx02	0.00000E+00	m/s	px02	0.00000E+00	Ns	L21	2.50E+09	Nsm		L22	0.00E+00	Nsm
vy02	1.00000E+00	m/s	py02	1.00000E+08	Ns	E11eff	0.00	J		E12eff	-9.67253E+12	J
						E21eff	-4.83627E+08	J		E22eff	0.00	J
						p21	4.68	m		p12	4.68	m
						eps21	0.81			eps12	0.81	
						a21	13.79	m	13.79	a12	13.79	m
						b21	8.04	m	8.039	b12	8.04	m
						T21	27.86	s	27.85	T12	27.86	s

Simulazione IM (M1 fermo)			
	t(s)	x(m)	y(m)
y max	10.57	11.2077253	8.038627
perielio	13.93	-2.5847148	0.024868
	13.94	-2.5842299	-0.07185
y min	17.3	11.2228259	-8.03862
afelio	27.86	24.9999969	-0.00514
	27.87	24.9999972	0.004859
y max	38.44	11.19261	8.038622
perielio	41.79	-2.5841869	0.074596
	41.8	-2.5847286	-0.02212
y min	45.16	11.2068415	-8.03863

(b)

Figura 2.14: Fig. 2.14(a): un pianeta di massa m_2 =10^8kg in rotazione attorno ad un Sole di massa M_1 =10^{12}kg, fermo nell'origine, con velocità e distanza iniziali $v_y = 1\frac{m}{s}, 25m$: schema, orbita calcolata con il modello IM, condizioni di integrazione; Fig. 2.14(b): caratteristiche delle orbite nei riferimenti in cui uno dei due corpi è fermo, calcolate e ricavate dall'orbita simulata.

Per calcolare le caratteristiche dell'orbita di M_1 *nel riferimento in cui* m_2 *è fermo, si procede allo stesso modo*: si riferiscono posizioni e velocità di M_1 a quelle di m_2 e si calcolano energia e momento di M_1 considerando $M_1 + m_2$ al posto di m_2:

$$E_{12eff} = \frac{1}{2} M_1 (v_{x012}^2 + v_{y012}^2) - \frac{C_{12}}{r_{12}} \qquad C_{12} = G(M_1 + m_2) M_1$$

Il risultato è riportato nella tabella di Fig. 2.14(b): come dedotto dalle Eq. 1.42, le caratteristiche dell'orbita di M_1 vista da m_2 sono identiche a quelle dell'orbita di m_2 vista da M_1. Purtroppo, *con IM è difficile trovare sistemi per cui si riescano a simulare accettabilmente entrambe le situazioni*. IM ammette estremi negativi degli slider, ma il moto di corpi di grande massa attorno a corpi di massa minore richiede passi di integrazione molto piccoli e tempi complessivi brevi, condizioni ben diverse da quelle in Fig. 2.14(b), per ottenere l'orbita prima studiata. La soluzione didattica, come si vedrà, è usare unità di misura che cambino l'ordine di grandezza della costante di gravitazione universale G.

2.2.4 Due stelle mobili in diversi riferimenti

Riferimento inerziale

Osservato il moto del pianeta nel riferimento in cui il piano xy è quello dell'orbita ed il Sole è fermo, si simula anche il moto del Sole in un riferimento inerziale (Par. 1.1.4). A tal fine si considerano le Eq. 2.7, 2.8 per le componenti della quantità di moto e le coordinate sia del pianeta, sia del Sole, modificando in Fig. 2.15(a) il modello di Fig. 2.11(a), *replicando per il Sole la stessa struttura grafica costruita per il pianeta*, fatta di Stocks per coordinate x, y e componenti della quantità di moto, e flussi per le forze e le velocità, tutti identificati con nomi opportuni (`https://insightmaker.com/insight/3h116Sq4lFdO6H1WKhhBVK`).

Fig. 2.15(b) mostra come la funzione distanza debba ridefinirsi come $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, mentre *sia meglio fare il calcolo delle forze una sola volta: per il terzo principio della dinamica, la forza del pianeta sul Sole è pari a quella del Sole sul pianeta cambiata di segno*. Grande attenzione deve essere quindi posta nell'imporre che tutti i flussi (forze e velocità) abbiano *doppio verso, tutti rispetto ad uno stesso verso positivo* (freccia bianca interna).

I risultati di due situazioni del tipo di quella illustrata in Fig. 2.15(a) sono riportati nelle Figg. 2.15(c) e 2.15(d). In Fig. 2.15(c) si vedono le traiettorie nel caso in cui il Sole parta da fermo ed il pianeta con una certa quantità di moto. Avendo il Sole una massa enorme rispetto al pianeta ($M_1 = 1.2 \cdot 10^{20}$kg, m_2 =3000kg), ci si aspetterebbe un'orbita ellittica imperturbata, ma non lo è: come evidente dal grafico, si muove su una scala di lunghezze infinitesima rispetto a quella del pianeta. Dando invece quantità di moto iniziale anche al Sole, nel caso lungo l'asse x ($p_{1x} = 1.4 \cdot 10^{22}$Ns), ecco che si ottiene Fig. 2.15(d): il Sole comincia a spostarsi mostrando l'effetto della piccola influenza del pianeta (il piccolo ma periodico spostamento lungo l'asse y); il pianeta, dal canto suo, insegue il Sole ruotando attorno alla sua traiettoria in un'orbita "'aperta"', a spirale ellittica.

Riferimento siderale

In tutti i modelli finora mostrati si saranno notati già dei punti di forza di IM, come l'aggiornamento dei grafici da slider, l'estrema semplicità delle operazioni necessarie, la modularità delle strutture grafiche, ma anche almeno tre grandi punti di debolezza didattica: la scelta automatica e non modificabile delle scale, l'esigenza di scrivere le unità di misura nelle didascalie (qui spesso disattesa. . .) e soprattutto l'impossibilità di mostrare due grafici x, y su uno stesso piano, il che rende molto scomodo il confronto delle traiettorie.

È perciò significativo modificare il modello di Fig. 2.15 come in Fig. 2.16, al fine di *mostrare le due orbite nel sistema siderale (Par. 1.1.4), ossia il riferimento in cui è*

fermo il Centro di Massa (CdM)[3] *della coppia pianeta-Sole*, di coordinate Eq. 1.45, dove esse sono più comode da studiare, anche se separate. Come mostrato in Fig. 2.16(a), partendo dal modello di Fig. 2.15, basta scalare le coordinate x, y di Sole e pianeta dei valori istantanei di x_G, y_G, calcolati dall'Eq. 1.45. Ciò è in accordo con Eq. 1.55, ma chiaramente nel modello il problema è già impostato nel piano dell'orbita, quindi le coordinate z non ci sono. Dal punto di vista tecnico, *occorre fare attenzione a definire le nuove coordinate* $xG1, yG2, xG2, yG2$ *non come stocks, ma come variabili, in quanto non vengono da integrazioni.* Le Figg. 2.16(b) e 2.16(c) mostrano le traiettorie di Sole e pianeta, rispettivamente, nel sistema inerziale o nel sistema siderale: si vede bene come nel primo (Fig. 2.16(b)), sebbene ci siano condizioni iniziali molto diverse ed in particolare $m_2 = \frac{1}{2}M_1$, le traiettorie siano spirali (forse) ellittiche, mentre nel sistema siderale esse siano due orbite ellittiche chiuse, entrambe con un fuoco proprio nell'origine, ossia in G (Fig. 2.16(c)) e gli altri due allineati ma non lungo l'asse x, avendo scelto il vettore velocità iniziale nel piano xy ma non ortogonale all'asse stesso.

Dal riferimento siderale al riferimento sinodale

Modificando il modello di Fig. 2.16 come in Fig. 2.17(a) (`https://insightmaker.com/insight/CFUT3ptZa0XBR2vH0BQsL`), si porta avanti uno studio quasi completo del problema dei due corpi nei sistemi siderale e sinodale (Par. 1.1.4). In Fig. 2.17(b) si mostrano le condizioni di partenza dei due corpi e, nelle tabelle, *i valori di energia, momento e parametri delle orbite calcolati nei riferimenti inerziale e siderale*. Nel riferimento inerziale, basta utilizzare quanto detto a proposito di Eq. 2.10. Per calcolare invece le caratteristiche delle orbite di i e di j riferite al CdM G, origine del sistema siderale, si ricorre a quanto fatto nel Par. 1.1.1. Si consideri prima l'orbita di i:

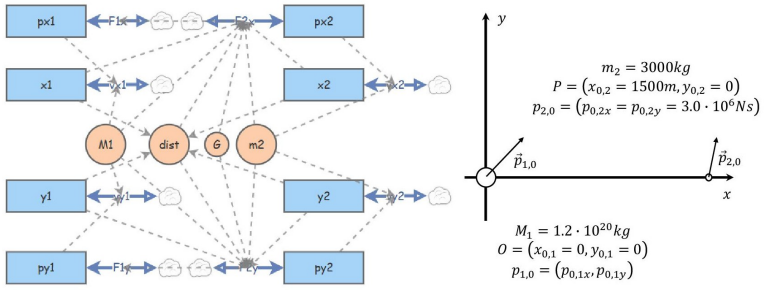
- si riferiscono posizioni e velocità di i a quelle di G;
- si calcolano l'energia di i sostituendo m_j con $m_j \left(1 + \frac{m_i}{m_j}\right)^{-2}$ in:

$$E_{ijeff} = \frac{1}{2} m_i (v_{x0ji1}^2 + v_{y0ji}^2) - \frac{C_{ji}}{r_{12}} \quad C_{ji} = G m_i m_j \left(1 + \frac{m_i}{m_j}\right)^{-2} \tag{2.12}$$

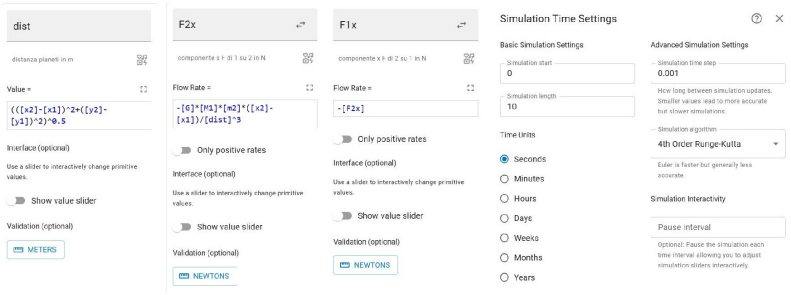
- si calcola anche il momento L_i con le nuove componenti della velocità, trovando a, b, ϵ, T per l'orbita di i vista da G fermo (Par. 1.1.1).

Per j, si ripete tutto quanto detto per $i \leftrightarrow j$. L'analisi delle orbite ottenute dal modello, *nel quale non si fa alcuna assunzione sul moto di* G, conferma tale algoritmo. Le orbite dei due corpi (Figg. 2.17(c), 2.18(a)) sono ellissi non riferite agli assi, ma selezionando sul grafico *configure* e poi *table* si ottiene un file `.csv` con tutte le coordinate (t, x, y) delle traiettorie, dalle quali si selezionano i dati nella

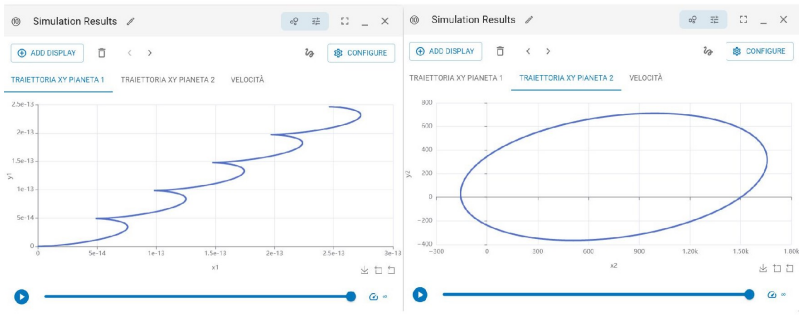
[3] Chiamato spesso anche G. Nei modelli IM è difficile confonderlo con la costante di gravitazione universale G.



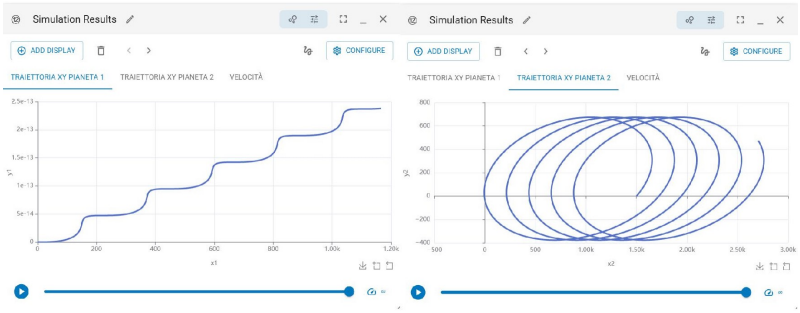
(a)



(b)

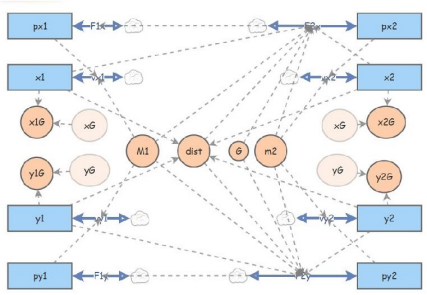


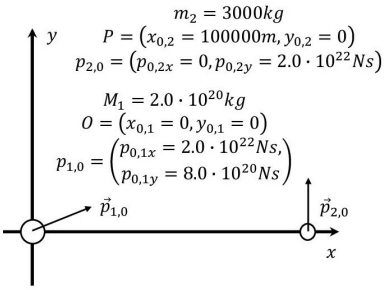
(c)

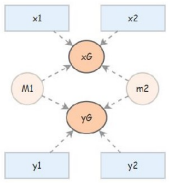


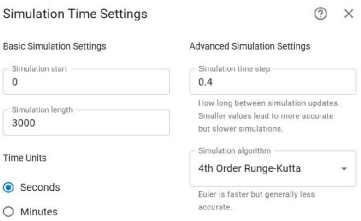
(d)

Figura 2.15: 2.15(a): modello per osservare in un riferimento inerziale le traiettorie sia del Sole, sia del pianeta nel piano delle loro orbite, con schema della situazione fisica simulata; 2.15(b): definizione della funzione distanza, imposizione del terzo principio della dinamica e caratteristiche della simulazione; 2.15(c): traiettorie nel caso in cui il Sole, molto più massivo del pianeta, parta da fermo; 2.15(d): traiettorie nel caso in cui il Sole parta con una notevole quantità di moto lungo l'asse x. Il modello è disponibile a `https://insightmaker.com/insight/3h116Sq4lFdO6H1WKhhBVK`

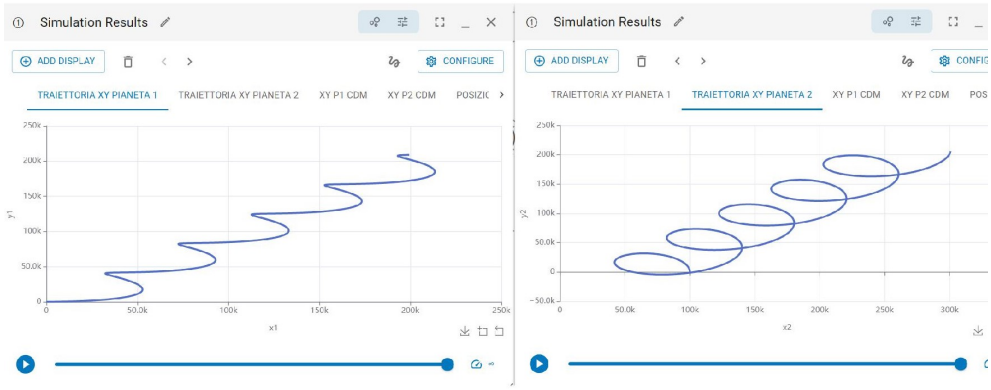








(a)



(b)

(c)

Figura 2.16: 2.16(a): modello per osservare le traiettorie del pianeta e del Sole (la cui massa è solo il doppio di quella del pianeta) nel riferimento del loro centro di massa G, con schema della situazione fisica simulata nel piano delle orbite; 2.16(a): traiettorie in un riferimento inerziale; 2.16(b): traiettorie nel riferimento del centro di massa G, fermo nell'origine. Il modello è disponibile a `https://insightmaker.com/insight/4tJ2Vsa7uyRJKIhc4jqv7a`

tabella in basso a sinistra in Fig. 2.17(b): istanti e posizioni di afelio e perielio forniscono valori di a, b, T in ottimo accordo con quelli calcolati.

Ulteriore conferma si ha ricavando le due orbite nel riferimento sinodale, centrato nel CdM e con l'asse x passante per il Sole ed il pianeta. A tal fine, nel riferimento siderale si calcola $x1S(t)$ come distanza di un pianeta dal CdM, trovando poi $x2S(t)$ come detto più volte:

$$x_{1,sin} = \sqrt{x_{1,sid}^2 + y_{1,sid}^2} \qquad x_{2,sin} = -\frac{m_1}{m_2} x_{1,sin} \tag{2.13}$$

In Fig. 2.18(c) si vede il risultato: ricavando sempre da un file `.csv` le distanze dal CdM (nell'origine), si vede che esse oscillano fra valori di afelio e perielio delle due parti praticamente identici a quelli di Fig. 2.17(b)).

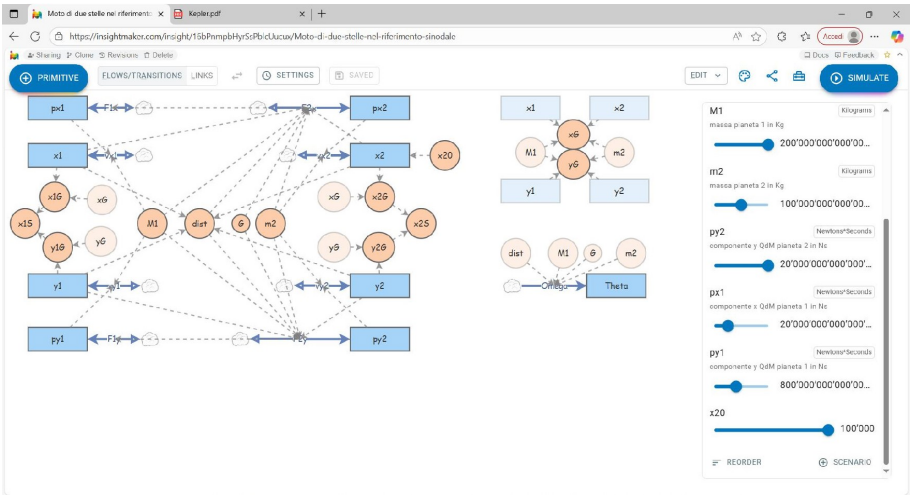
Sempre dalle coordinate siderali, una semplice funzione arcotangente fornisce anche l'angolo $\theta(t)$ formato dagli assi x dei riferimenti sinodale e siderale nel tempo (Fig. 2.18(c)). Coerentemente con l'Eq. 1.53, ha derivata massima/minima nei punti in cui i pianeti sono a distanza minima/massima.

Uso dell'Equazione di Kepler

Come visto in teoria (Eq. 1.12) e verificato nei modelli (Fig. 2.16), nel sistema siderale le traiettorie del pianeta e del Sole sono coniche riferite agli assi solo qualora i vettori delle velocità iniziali siano ortogonali ad uno degli assi stessi. In caso contrario sono ruotate (e deformate da IM), creando problemi di confronto fra quanto calcolato e rappresentato. Ciò è stato appena visto nello studio del sistema in Fig.2.17(b), portato avanti trovando i valori di energia, momento angolare e caratteristiche dell'orbita ellittica *di ciascun corpo i visto in moto rispetto al CdM, fermo nell'origine e di massa efficace $m_j \left(1 + \frac{m_i}{m_j}\right)^{-2}$, come se l'altro j non ci fosse*, ottenendo proprio i parametri delle due ellissi in Figg. 2.17(c) e 2.18(a).

Per confermare ulteriormente la correttezza del metodo confrontando i valori calcolati e simulati dell'afelio e del perielio, conviene calcolare $r_1(t), r_2(t)$ nel sistema sinodale *anche mediante l'equazione di Kepler* (Par. 1.1.3). Ciò mostra anche che in due dimensioni il problema di avere coniche ruotate non è insormontabile, mentre in tre dimensioni richiede strumenti come i cambiamenti di base (Par. 1.1.5) che non entrano nei programmi prima del terzo anno di studio. Trovare pertanto un metodo alternativo è decisamente opportuno.

A tal fine, in Fig. 2.19(a) si riporta un modello che, date le condizioni iniziali $x_{0i}, y_{0i}, v_{x0i}, v_{y0i}, i = 1, 2$ in un riferimento inerziale, prima le trasla in un riferimento dove il CdM è fermo (parte del modello in alto a sinistra), poi calcola in quest'ultimo tutte le grandezze necessarie a stabilire le caratteristiche delle orbite proprio nel riferimento siderale (come energia e momento angolare: parte in alto a destra), infine usa questi ultimi risultati per disegnare direttamente le orbite riferite agli assi, integrando l'Eq. 1.37 per trovare l'anomalia eccentrica u per ogni corpo, e da questa prima le coordinate polari, a partire dall'anomalia vera θ (Eq.1.36), e poi



(a)

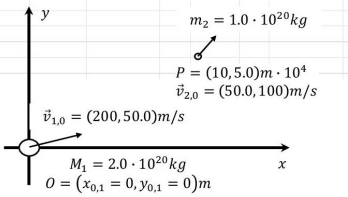
Riferimento inerziale					
G	6.7E-11	Nm²/kg²	dist12	111803	m
M1	2E+20	kg	xG	3.33E+04	m
m2	1.00E+20	kg	yG	1.67E+04	m
C	1.33E+30	Nm²	vxG	1.50E+02	m/s
			vyG	6.67E+01	m/s
x01	0	m			
y01	0	m	Etot12	-7.06E+24	J
x02	100000	m	L1	0	Nsm
y02	50000	m	L2	7.50E+26	Nsm
vx01	200	m/s	px01	4E+22	Ns
vy01	50	m/s	py01	1.00E+22	Ns
vx02	50	m/s	px02	5.00E+21	Ns
vy02	100	m/s	py02	1.00E+22	Ns

Riferimento del centro di massa G					
x01G	-3.33E+04	m	dist1G	37267.8	m
y01G	-1.67E+04	m	dist2G	74535.6	m
vx01G	5.00E+01	m/s			
vy01G	-1.67E+01	m/s	C1	1.48E+29	Nm²
x02G	6.67E+04	m	C2	5.93E+29	Nm²
y02G	3.33E+04	m			
vx02G	-1.00E+02	m/s	E1Geff	-3.70E+24	J
vy02G	3.33E+01	m/s	E2Geff	-7.40E+24	J
			L1G	2.78E+26	Nsm
Etot12G	-1.11E+25	J	L2G	5.56E+26	Nsm

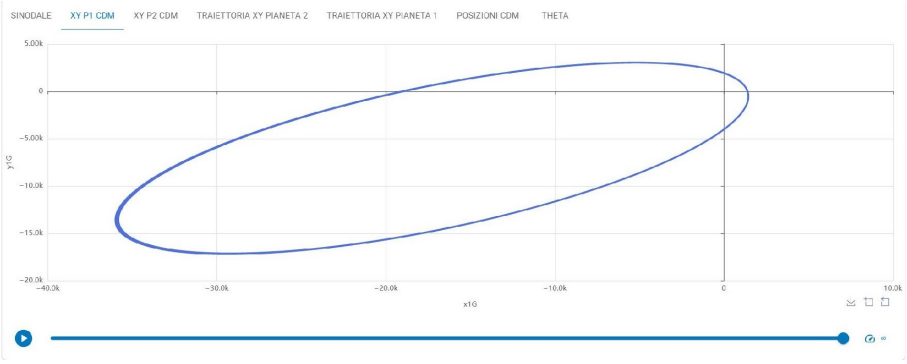
			IM (CdM)		IM (Eq. Kepler)
p1G	2602.8652	m			
eps1G	0.9328				
aG1	20033.0523	m			
bG1	7221.0342	m			7221
TG1	654.4235	s	652.6	1.4	
c	18686.3546	m			
eps1G	0.9328				
a-c	**1346.7**	m	1349.1	3.2	1346
a+c	**38719**	m	38649	35	38719
p2G	5205.7305	m			
eps2G	0.9328				
aG2	40066.1046	m			
bG2	14442.0685	m			14442
TG2	654.4235	s			
c	37372.7093	m			
eps1G	0.9328				
a-c	**2693.4**	m	2698.2	6.5	2692.5
a+c	**77439**	m	77297	71	77438

Dati IM, riferimento del CdM				
	t(s)	r1(m)	r2(m)	
Perielio	248.4	1347.2712	2694.54	
Afelio	575.2	38684.022	77368	
Perielio	902	1348.5408	2697.08	653.6
Afelio	1229.6	38648.283	77296.6	654.4
Perielio	1554.4	1353.8321	2707.66	652.4
Afelio	1880.4	38613.452	77226.9	650.8
Perielio	2206.4	1346.8118	2693.62	652

Dati IM, Eq. Di Kepler				
t(s)	x1G(m)	y1G(m)	x2G(m)	y2G(m)
322.32	**38719.406**	-1.75159	**-77439**	3.50319
322.4	**38719.407**	1.11805	**-77439**	-2.23611
583.12	18687.849	**7221.03**	-37376	**-14442.1**
649.52	**-1345.9479**	62.4734	**2691.9**	-124.947
649.6	**-1346.6207**	-20.021	**2693.2**	40.0419
716	18679.929	**-7221.03**	-37360	**14442.1**

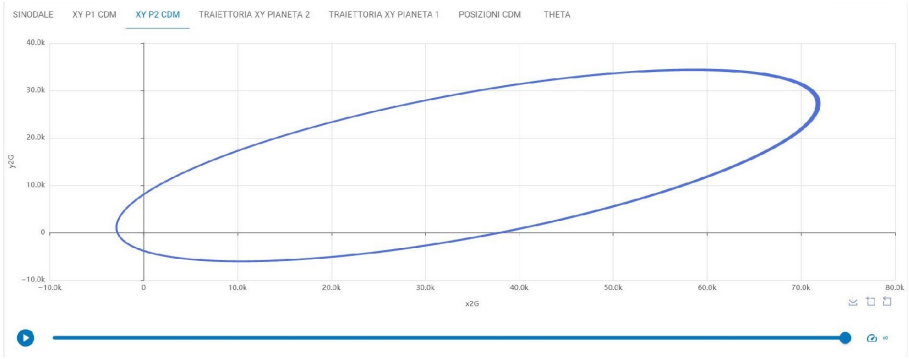


(b)



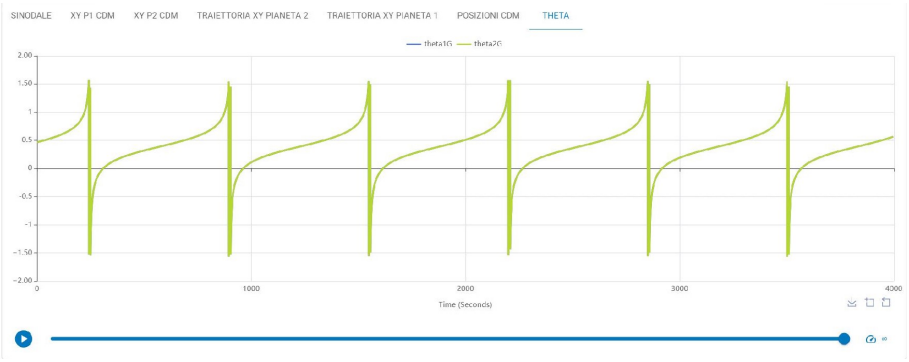
(c)

Figura 2.17: 2.17(a): modello per osservare le traiettorie di due pianeti nei riferimenti siderale e sinodale nel tempo (`https://insightmaker.com/insight/CFUT3ptZa0XBR2vH0BQsL`); 2.17(b): situazione fisica simulata, calcolo delle caratteristiche delle orbite nei riferimenti inerziale e siderale e loro confronto con i parametri calcolati da più modelli IM; 2.17(c): prima traiettoria nel sistema siderale.



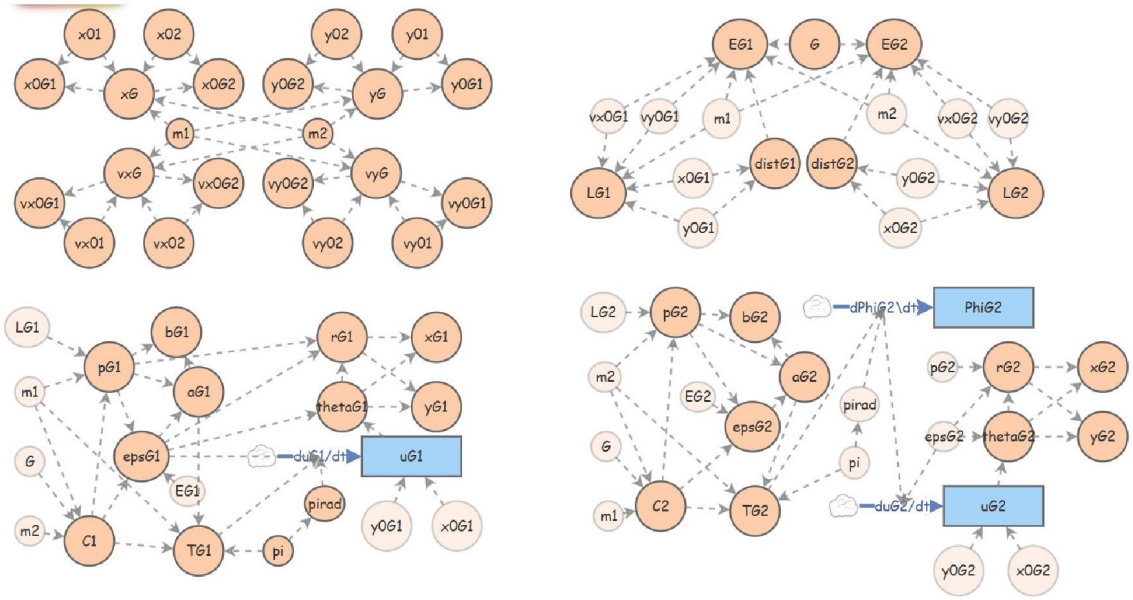
(a)

(b)

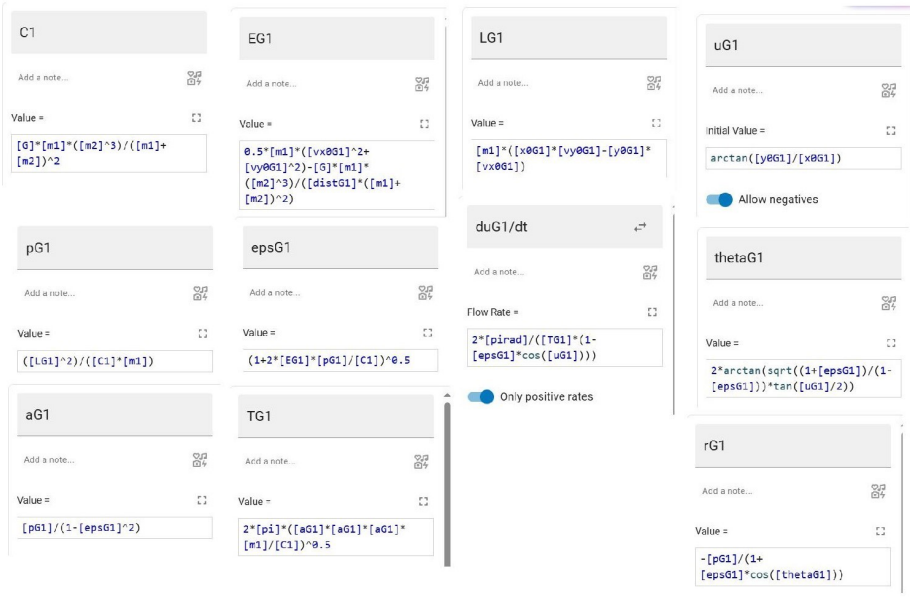


(c)

Figura 2.18: Moto di due corpi in un sistema siderale e sinodale secondo il modello di Fig. 2.17(a) (`https://insightmaker.com/insight/CFUT3ptZa0XBR2vH0BQsL`); 2.18(a): seconda traiettoria nel sistema siderale, della situazione fisica in Fig. 2.17(b); 2.18(b): traiettorie nel riferimento sinodale, con controllo di afelio e perielio; 2.18(c): andamento nel tempo dell'angolo $\theta(t)$ formato dagli assi x dei riferimenti sinodale e siderale nel tempo.

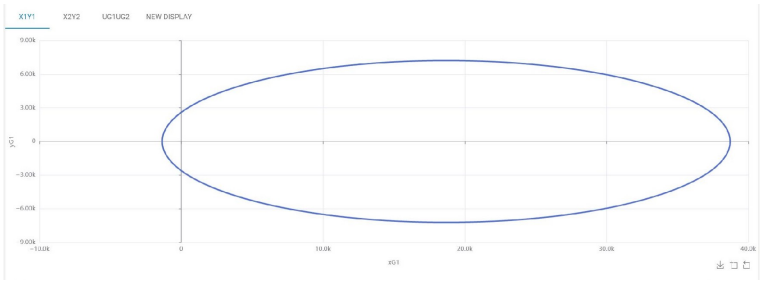


(a)

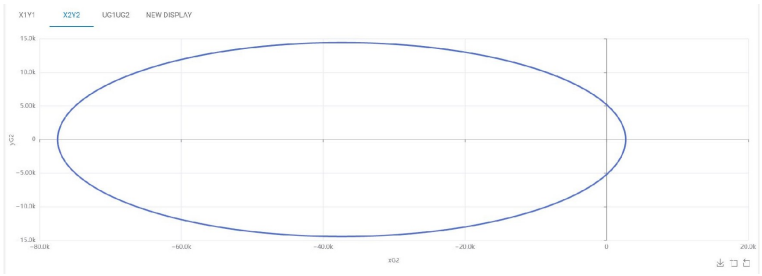


(b)

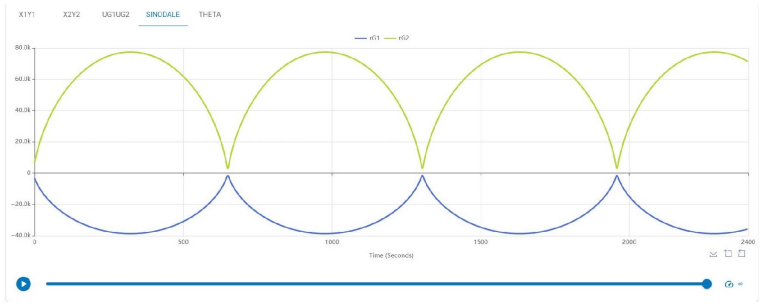
Figura 2.19: 2.19(a): modello per rappresentare orbite riferite agli assi in un sistema siderale in due dimensioni. Spiegazione delle 4 zone, dall'alto a sinistra, in senso orario: le condizioni iniziali sono riferite al CdM; calcolo di energia e momento angolare dei corpi nel sistema del CdM; calcolo delle anomalie media Φ, eccentrica u e vera θ, e da queste le coordinate polari e cartesiane del secondo corpo (Par. 1.1.3); stessa cosa per il primo corpo, senza integrazione dell'anomalia media Φ; 2.19(b): definizioni di molte delle grandezze: attenzione nel definire θ_{0i} e nell'interpretare le posizioni delle due ellissi dopo aver scelto i segni in $r_i = \frac{\pm p_i}{1 \pm \epsilon \cos\theta_i}$. Il modello è disponibile a `https://insightmaker.com/insight/2qd1jComdv4Y8CEP9WCcQq`.



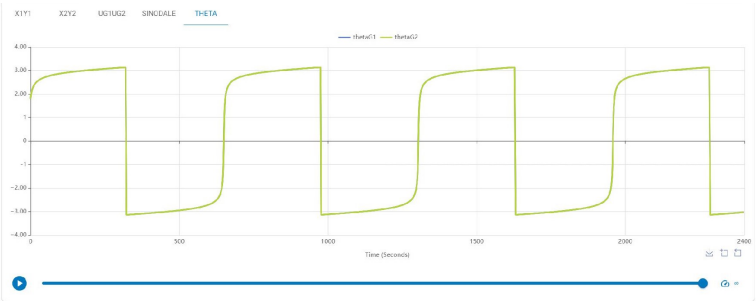
(a)



(b)



(c)



(d)

Figura 2.20: 2.20(a), 2.20(b): risolvendo l'equazione di Kepler in un riferimento siderale, le ellissi di Figg. 2.17(c), 2.18(a) sono subito riferite agli assi; 2.20(c): rappresentazione nel sistema sinodale delle distanze dal CdM: identica a Fig. 2.18(b); 2.20(d): andamento nel tempo dell'angolo $\theta(t)$ fra gli assi x dei riferimenti sinodale e siderale nel tempo. Il modello è disponibile a `https://insightmaker.com/insight/2qd1jComdv4Y8CEP9WCcQq`.

quelle cartesiane, secondo la teoria illustrata nel Par. 1.1.3. A parte la necessità di definire bene le grandezze (Fig. 2.19(b)), ed interpretare il ruolo di segni e condizioni iniziali sugli angoli Φ, u, θ, nelle Figg. 2.20(a), 2.20(b) è rappresentata la stessa coppia di ellissi delle Figg. 2.17(c) e 2.18(a), così come in Fig. 2.20(c) si vede quanto già rappresentato in Fig. 2.17(c). C'è invece discrepanza fra gli andamenti di $\theta(t)$ nelle Figg. 2.20(d) e 2.18(c) in quanto come detto occorre riferirli ad uno stesso valore di partenza.

Una ulteriore accortezza da considerare è che *qualora si voglia partire da questo modello per lo studio del moto di un terzo corpo interagente con gli altri due*, va ricordato che l'equazione di Kepler è applicabile solo nel caso ideale, *quando la dinamica dei due corpi non è influenzata da quella del terzo*. Più opportuno sarebbe estendere pertanto il modello di Fig. 2.17(a), valutando come definire le interazioni, se nel riferimento inerziale o in quello sinodale (si ricorda, non inerziale). Se ne riparlerà nel Cap. 3.

2.3 Codici C++

Se il punto di forza della modellizzazione dinamica mediante IM è nel fatto che il software è tradotto in (o volendo "'nascosto da"') una mappa concettuale la cui struttura evidenzia il senso fisico dei problemi, la sua debolezza è nella perdita della discussione dell'algoritmo e soprattutto della visione in 3D. Si tratta d'altra parte di scelte didattiche: tutto ciò può recuperarsi con una programmazione più classica [32], come quella che segue, in C++.

2.3.1 Con il Sole fermo nell'origine

Un codice che simuli in 3D il moto di un pianeta attorno ad un Sole fermo (Par. 2.2.3) è riportato nell'Appendice A.1. Dopo il preambolo, esso comincia con la vettorializzazione (di dimensione P) del tempo T e delle componenti X, Y, Z della posizione, della velocità Vx, Vy, Vz e del momento angolare Lx, Ly, Lz del pianeta. Vengono definiti vettori anche per il momento angolare e l'energia totali del pianeta, nonché per la sua energia potenziale e cinetica, in modo da poterne studiare l'andamento nel tempo. Sono poi definite le variabili che servono per ottenere tutte queste grandezze istante per istante: le componenti dell'accelerazione ax, ay, az, quelle del momento angolare e i valori di energia cinetica e potenziale:

```
1 # include <iostream >
2 (...)
3 using namespace std;
4 const int P = 110000;
5 double G, M, m;
6 double T[P], X[P], Y[P], Z[P], Vx[P], Vy[P], Vz[P], Ep[P], Ec[P], Et[P], Lx[P], Ly[
      P], Lz[P], Lt[P];
7 double ax, ay, az, Epot, Ecin, Lax, Lay, Laz;
```

Le costanti ed i parametri iniziali del sistema sono inseriti in un file `input.txt`, il che permette una certa flessibilità (dopo la compilazione basta cambiare l'input ed eseguire il file `.exe`), ma richiede anche molta consapevolezza nell'uso. *Ad esempio, in tutti i codici sviluppati le unità di misura sono scelte in modo che la costante di gravitazione universale risulti* $G = 1$. Si lascia a chi legge la scelta di modificare il tutto secondo altre convenzioni [33], [35].

Letti i parametri dal file input.txt, si definiscono i nomi delle procedure di integrazione (funzioni `void`) e delle funzioni accelerazione, velocità, energia e momento angolare. Sono introdotte ben 4 possibili procedure di integrazione, ossia gli algoritmi di Eulero, Eulero-Cromer, Verlet e Runge-Kutta al 4. ordine, i cui dettagli sia matematici, sia informatici, si trovano in qualunque testo di simulazione, sebbene l'opera di riferimento sia [31]:

```
1 void euler(const double& Ti, const double& Tf, const double& X0, const double& Vx0,
       const double& Y0, const double& Vy0, const double& Z0, const double& Vz0,
      const int& Nt);
2 void euler_cromer(...);
3 void velocity_verlet(...);
4 void RK(...);
5 void RKSTEP(double& t, double& x, double& vx, double& y, double& vy, double& z,
      double& vz, const double& dt);
6 double acc(double& x, double& y, double& z, double& vx, double& vy, double& vz,
      double& ax, double& ay, double& az, double& t);
7 double vel(double& x, double& y, double& z, double& vx, double& vy, double& vz,
      double& t);
8 double En(double& x, double& y, double& z, double& vx, double& vy, double& vz,
      double& Ecin, double& Epot);
9 double La(double& x, double& y, double& z , double& vx, double& vy, double& vz,
      double& Lax, double& Lay, double& Laz);
```

Le procedure richiedono gli istanti di inizio T_i e fine T_f del moto, le posizioni $X0, Y0, Z0$ e le velocità iniziali $Vx0, Vy0, Vz0$, nonché il numero dei passi Nt, che deve essere minore dell'estensione P dei vettori. Le funzioni accelerazione e velocità richiedono invece molte più grandezze di quelle necessarie, un sovradimensionamento del codice che risulta tuttavia utile qualora chi legge voglia modificarlo per integrare un qualunque altro sistema di equazioni differenziali (con forze variabili nel tempo, come in un comune oscillatore armonico smorzato e forzato).

Comincia quindi il programma principale, che legge i dati in ingresso, confronta Nt con P e scrive in output le caratteristiche del sistema:

```
1  int main()
2  {
3  double Ti, Tf, X0, Y0, Z0, Vx0, Vy0, Vz0;
4  int Nt, i;
5    ifstream ifile("input.txt");
6      while (!ifile.eof()) {
7              ifile >> G >> M >> m >> Ti >> Tf >> Nt >> X0 >> Y0 >> Z0 >> Vx0 >> Vy0
       >> Vz0;}
8      ifile.close();
9      if(Nt>=P) {cerr << "Error : Nt>=P\n"; exit(1);}
10     cout << "Massa Sole del sistema: M=" << M << "\n";
11     cout << "Massa Terra del sistema m=" << m << "\n";
12     cout << "Posizione Sole (fissa): x=" << 0 << " y=" << 0 << "\n";
13     cout << "Posizione iniziale Terra: x=" << X0 << " y=" << Y0 << " z=" << Z0 <<
       "\n";
```

```
14      cout << "Velocita' iniziale Terra: vx=" << Vx0 << " vy=" << Vy0 << " vz=" <<
        Vz0 << "\n";
15      cout << "Tempo di simulazione: " << Ti << "-" << Tf << " e numero passi: " <<
        Nt << "\n";
```

Fatto ciò il codice esegue l'integrazione con più algoritmi, aprendo poi per ciascuno un file di output (`.xls`, o `.txt`). Per il RK4, ad esempio:

```
RK(Ti, Tf, X0, Vx0, Y0, Vy0, Z0, Vz0, Nt);
ofstream myfile4 ("rk4.xls");
myfile4.precision(17);
for(i=0; i<Nt; i++)
myfile4 << T[i] << "\t" << X[i] << "\t" << Y[i] << "\t" << Z[i] << "\t" << Vx[i] <<
     "\t" << Vy[i] << "\t" << Vz[i] << "\t" << endl;
myfile4.close();
ofstream myfile5 ("rk4.txt");
(...)
}
```

Concluso così il `main`, sono definite le funzioni. La prima è quella che esprime le componenti dell'accelerazione del pianeta a partire da quelle della forza:

```
double acc(double& x, ...)
{double r;
r=sqrt(x*x+y*y+z*z);
ax=-G*M*x/(r*r*r);
ay=-G*M*y/(r*r*r);
az=-G*M*z/(r*r*r);}
```

Per poter generalizzare il software, segue una funzione che esplicita le componenti della velocità, che qui... non fa nulla (chi legge potrebbe semplificare il codice togliendo tutti i riferimenti alle componenti della velocità).

Seguono le funzioni che calcolano energia cinetica, potenziale e le componenti del momento angolare:

```
double En(double& x, double& y, double& z, double& vx, double& vy, double& vz,
    double& Ecin, double& Epot)
{double r, vsqr;
r=sqrt(x*x+y*y+z*z);
vsqr=vx*vx+vy*vy+vz*vz;
Epot=-G*M*m/r;
Ecin=0.5*m*vsqr;}

double La(double& x, double& y, double& z , double& vx, double& vy, double& vz,
    double& Lax, double& Lay, double& Laz)
{Lax=m*(y*vz-z*vy);
Lay=m*(z*vx-x*vz);
Laz=m*(x*vy-y*vx);}
```

Infine, sono esplicitate le procedure di integrazione, come quella classica di Eulero:

```
void euler(const double& Ti, ...)
{int i; double h;
T[0]=Ti; Vx[0]=Vx0; Vy[0]=Vy0; Vz[0]=Vz0; X[0]=X0; Y[0]=Y0; Z[0]=Z0;
acc(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], ax, ay, az, T[0]);
h=Tf/(Nt-1);
for(i=1; i<Nt; i++)
{T[i]=T[i-1]+h;
Vx[i]=Vx[i-1]+ax*h;
Vy[i]=Vy[i-1]+ay*h;
Vz[i]=Vz[i-1]+az*h;
```

```
X[i]=X[i-1]+Vx[i-1]*h;
Y[i]=Y[i-1]+Vy[i-1]*h;
Z[i]=Z[i-1]+Vz[i-1]*h;
acc(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], ax, ay, az, T[i]);}}
```

Molto più complessa è la Runge-Kutta 4. In una prima funzione *void*, chiamata `RK`, si inizializzano i vettori posizione, velocità, energia (cinetica, potenziale, totale) e momento angolare (componenti e totale) richiamando le rispettive funzioni, ed infine le variabili interne usate per l'integrazione (`XS, YS...`). Si esegue quindi quest'ultima sull'intero intervallo di tempo `dt`, richiamando passo passo una seconda funzione *void*, `RKstep`, in un ciclo `for`, ottenendo così i valori di posizione, velocità, energia e momento da memorizzare nei vettori di uscita:

```
void RK(const double& Ti, ...)
{double dt;
double TS, XS, VxS, YS, VyS, ZS, VzS;
int i;
dt =(Tf-Ti)/(Nt-1);    // passo di integrazione
T[0]=Ti; X[0]=X0; Vx[0]=Vx0; Y[0]=Y0; Vy[0]=Vy0; Z[0]=Z0; Vz[0]=Vz0; //
    inizializzazione output

En(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], Ecin, Epot);  // calcolo energia
Ep[0]=Epot; Ec[0]=Ecin; Et[0]=Epot+Ecin;

La(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], Lax, Lay, Laz); // calcolo momento
Lx[0]=Lax; Ly[0]=Lay; Lz[0]=Laz; Lt[0]=sqrt(Lax*Lax+Lay*Lay+Laz*Laz);

TS=Ti;                 // inizializzazione variabili interne
XS=X0; VxS=Vx0; YS=Y0; VyS=Vy0; ZS=Z0; VzS=Vz0;
for(i+1;i<Nt;i++)      // ciclo
  {
  RKSTEP(TS,XS,VxS,YS,VyS,ZS,VzS,dt);  // passo (funzione RKSTEP)
  T[i]=TS;       // caricamento risultati del passo nell'output
  X[i]=XS;  Vx[i]=VxS;
  Y[i]=YS;  Vy[i]=VyS;
  Z[i]=ZS;  Vz[i]=VzS;

  En(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], Ecin, Epot);
  Ep[i]=Epot; Ec[i]=Ecin; Et[i]=Epot+Ecin;
  La(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], Lax, Lay, Laz);
  Lx[i]=Lax; Ly[i]=Lay; Lz[i]=Laz;
  Lt[i]=sqrt(Lax*Lax+Lay*Lay+Laz*Laz);
  }
}
```

Nella funzione `RKstep`, l'intervallo di tempo `dt` è suddiviso in 4 parti, per ognuna delle quali sono richiamate le funzioni accelerazione e velocità (nel caso, silente) ottenendo le quantità k_{ij} poi sommate con i determinati pesi dell'algoritmo di 4. grado [31]:

```
void RKSTEP(double& t, double& x, double& vx, double& y, double& vy, double& z,
    double& vz, const double& dt)
{
  double xf, yf, zf, vxf, vyf, vzf, tf;
  double k11x, k12x, k13x, k14x, k21x, k22x, k23x, k24x;
  double k11y, k12y,k13y, k14y, k21y, k22y, k23y, k24y;
  double k11z, k12z, k13z, k14z, k21z, k22z, k23z, k24z;
  double h, h2, h6;
  h=dt; h2=0.5*h; h6=h/6.0;
  xf=x;  yf=y;  zf=z;
```

```
  vxf=vx;  vyf=vy;  vzf=vz;
  tf=t;
  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
  vel(x, y, z, vx, vy, vz, t);
  t=tf+h2;
  k21x=ax;  k11x=vx;  vx=vxf+h2*k21x; x=xf+h2*k11x;
  k21y=ay;  k11y=vy;  vy=vyf+h2*k21y; y=yf+h2*k11y;
  k21z=az;  k11z=vz;  vz=vzf+h2*k21z; z=zf+h2*k11z;
  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
  vel(x, y, z, vx, vy, vz, t);
  t=t;           // per ricordare che resta t=tf+h2
  k22x=ax;  k12x=vx;  k22y=ay;  k12y=vy;  k22z=az;  k12z=vz;
  vx=vxf+h2*k22x; x=xf+h2*k12x;
  vy=vyf+h2*k22y; y=yf+h2*k12y;
  vz=vzf+h2*k22z; z=zf+h2*k12z;
  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
  vel(x, y, z, vx, vy, vz, t);
  t=tf+h;
  k23x=ax;  k13x=vx;  k23y=ay;  k13y=vy;  k23z=az;  k13z=vz;
  x=xf+h*k13x;  y=yf+h*k13y;  z=zf+h*k13z;
  vx=vxf+h*k23x;  vy=vyf+h*k23y;    vz=vzf+h*k23z;
  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
  vel(x, y, z, vx, vy, vz, t);
  t=t;         // per ricordare che resta t=tf+h
  k14x=vx;  k24x=ax;  k14y=vy;  k24y=ay;  k14z=vz;  k24z=az;
  x=xf+h6*(k11x+2.0*(k12x+k13x)+k14x);  vx=vxf+h6*(k21x+2.0*(k22x+k23x)+k24x);
  y=yf+h6*(k11y+2.0*(k12y+k13y)+k14y);  vy=vyf+h6*(k21y+2.0*(k22y+k23y)+k24y);
  z=zf+h6*(k11z+2.0*(k12z+k13z)+k14z);  vz=vzf+h6*(k21z+2.0*(k22z+k23z)+k24z);
}
```

Nelle Fig.2.21 si mostra come il codice possa essere aperto, modificato e compilato, ad esempio con `Dev-C++` (Fig. 2.21(a)). Può essere eseguito anche come file `.exe`, avendo avuto cura di selezionare bene le condizioni iniziali nel file di input (Fig. 2.21(b)).

Come IM, anche la programmazione classica ha i suoi problemi. Il primo del `C++` è la mancanza di strumenti grafici semplici, ragione per cui l'output è stato portato in un file `excel`, dal quale possono rappresentarsi gli andamenti nel tempo delle tre componenti di posizione e velocità, periodici, se si sono scelte opportunamente le condizioni iniziali (Fig. 2.21(a)). Sempre il file excel consente di verificare la *sostanziale* conservazione dell'energia totale (Fig. 2.22(a)) e di *ciascuna* delle componenti del momento angolare (Fig. 2.22(b)). Ingrandendo gli andamenti di E_{tot}, L_{tot}, si osserva in realtà quanto già visto in Fig. (Fig. 2.13(a)): la conservazione è in realtà "'a gradini"', in quanto ogni volta che il pianeta si trova al perielio, l'andamento r^{-2} delle forze introduce errori numerici, la cui entità complessiva pregiudica l'affidabilità della simulazione su tempi lunghi[4].

La rappresentazione dell'orbita più utile didatticamente è quella in 3D mediante `Geogebra` (Fig. 2.22(c)). Importando le coordinate in output in un foglio elettronico interno e creando una *spezzata aperta* (selezionate le colonne con le coordinate x, y, z, dal tasto destro del mouse si sceglie *crea*), l'orbita (in questo caso ellittica)

[4]Per capire senza dover aprire il file excel, a quale forma di orbita portano i dati in input e l'affidabilità della simulazione, si potrebbe calcolare e scrivere in output, dopo l'esecuzione (Fig. 2.21(b)), l'energia iniziale del pianeta e le variazioni relative (in percentuale) delle grandezze conservate sull'intervallo di tempo scelto.

viene rappresentata in 3D, con la possibilità di cambiare dinamicamente il punto di vista. Ciò consente di osservare chiaramente che il Sole sia nel fuoco (origine del riferimento), che l'orbita sia piana, e di ingrandire scoprendo che la scelta del tempo (0.3) e del passi (500) è migliorabile: l'orbita si chiude appena. Il problema di tale rappresentazione è la notevole quantità di memoria richiesta: per ottenere immagini significative senza che `Geogebra` si blocchi, è meglio *importare al massimo 2000 punti* usandoli solo per creare la traiettoria come linea spezzata, ossia, senza rappresentarli.

2.3.2 Le due stelle mobili in più riferimenti

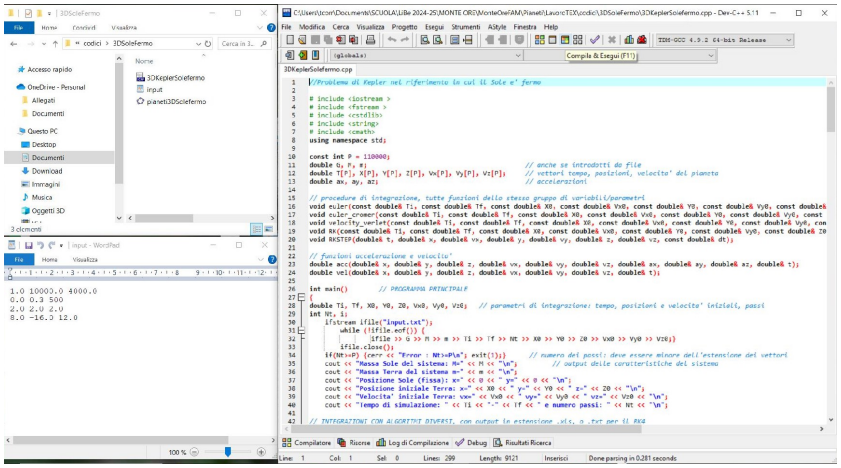
Nell'App. A.2 si riportano due codici: uno simula i moti di un pianeta e del suo Sole in un generico riferimento inerziale, l'altro li riporta in un riferimento centrato nel centro di massa (CdM) del sistema a due corpi.

Dal punto di vista informatico, quanto sviluppato per il solo pianeta nel codice del Par. 2.3.1 viene ora implementato anche per il Sole in entrambi i nuovi codici, con sole tre modifiche importanti. La prima riguarda le componenti dell'accelerazione nella relativa funzione, calcolate a partire dal terzo principio della dinamica:

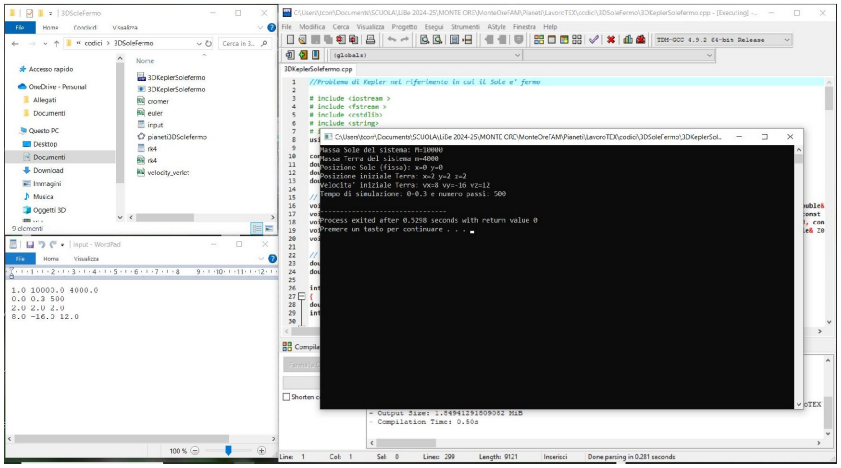
```
double acc(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
    double& ax1, double& ay1, double& az1, double& ax2, double& ay2, double& az2)
{
double r12=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2)+(z1-z2)*(z1-z2));
double f12=-G*m1*m2/(r12*r12);
ax1=(f12*(x1-x2)/r12)/m1;
ax2=-ax1*m1/m2;
ay1=(f12*(y1-y2)/r12)/m1;
ay2=-ay1*m1/m2;
az1=(f12*(z1-z2)/r12)/m1;
az2=-az1*m1/m2;
}
```

La seconda è che si utilizza il solo algoritmo Runge-Kutta4, estendendolo a due corpi (il listato è nell'App. A.2). La terza, che riguarda solo il codice che riporta i moti ad un riferimento centrato nel CdM (App. A.2.2), è che la traslazione è operata non nella funzione `RKSTEP`, già molto estesa, ma nella `RK`, nella quale si riferiscono al CdM sia le condizioni iniziali, sia le posizioni e le velocità calcolate passo per passo nel ciclo:

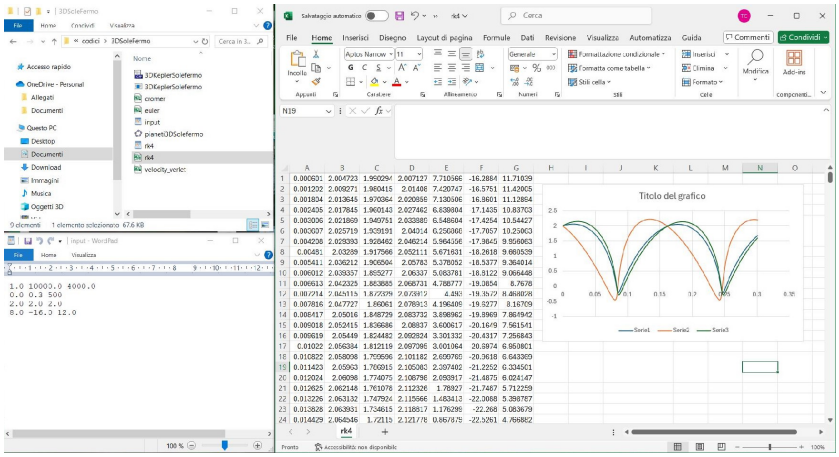
```
void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
    ...)
{
double dt; double TS, XS1, VxS1, YS1, VyS1, ZS1, VzS1, XS2, VxS2, YS2, VyS2, ZS2,
    VzS2;
int i; dt =(Tf-Ti)/(Nt-1);
T[0]=Ti;
xG=(m1*X01+m2*X02)/(m1+m2);  // caratteristiche iniziali del CdM
yG=(m1*Y01+m2*Y02)/(m1+m2);
zG=(m1*Z01+m2*Z02)/(m1+m2);
vxG=(m1*Vx01+m2*Vx02)/(m1+m2);
vyG=(m1*Vy01+m2*Vy02)/(m1+m2);
vzG=(m1*Vz01+m2*Vz02)/(m1+m2);
X1[0]=X01-xG;  Vx1[0]=Vx01-vxG;   // traslazione per il Sole
Y1[0]=Y01-yG;  Vy1[0]=Vy01-vyG;
```



(a)

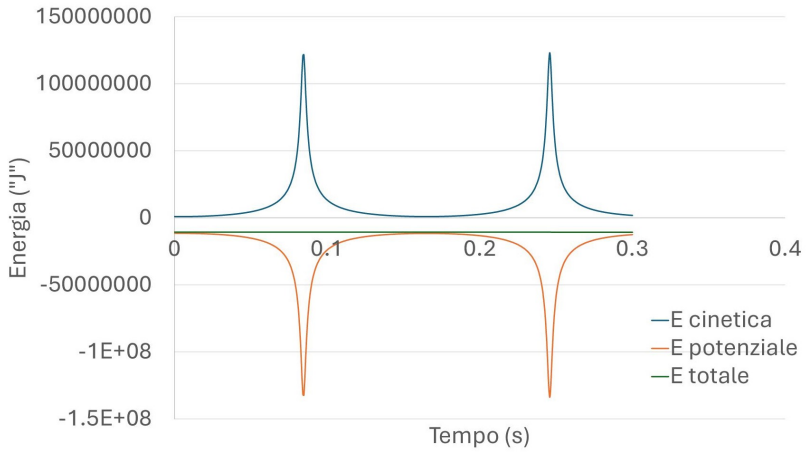


(b)

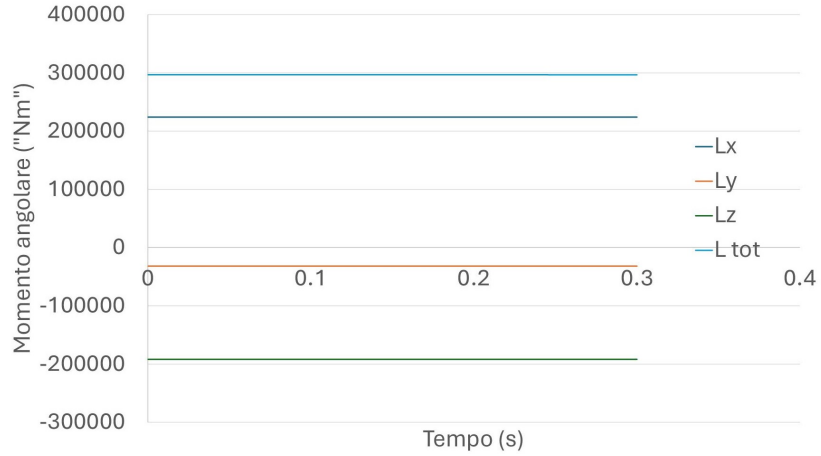


(c)

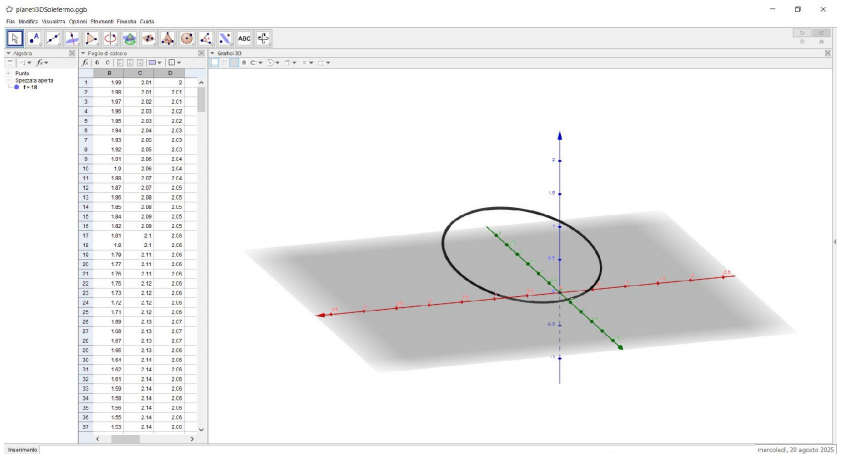
Figura 2.21: Simulazione `C++` di un pianeta che ruota attorno ad un Sole fermo nell'origine (App. A.1). 2.21(a): gestione del codice e del file di input: ricordare l'ordine di entrata dei dati (!), nel caso: $m_1 = 10000.0, m_2 = 4000.0, T_i = 0.0, T_f = 0.3, Nt = 500, x_{02} = 2.0, y_{02} = 2.0, z_{02} = 2.0, v_{0x2} = 8.0, v_{0y2} = -16, v_{0z2} = 12$, tutte in unità tali che la costante di gravitazione universale diventi $G = 1$; 2.21(b): compilazione e generazione del file `.exe`; 2.21(c): gestione del file `.xls` di output: ricordare l'ordine di uscita dei dati (!), nel caso: $t, x_2, y_2, z_2, v_{x2}, v_{y2}, v_{z2}, E_{cin}, E_{pot}, E_{tot}, L_x, L_y, L_z, L_{tot}$.



(a)



(b)



(c)

Figura 2.22: Simulazione C++ di un pianeta che ruota attorno ad un Sole fermo nell'origine, come in Fig. 2.21 (App. A.1). 2.22(a): andamenti nel tempo dell'energia cinetica, potenziale e totale del pianeta (in "'J"' ma le unità sono quelle per cui $G = 1$). L'energia totale è conservata: cambia dello 0.857% sull'intera simulazione; 2.22(c): andamenti delle componenti e valore totale del momento angolare del pianeta nel tempo (in "'Nm"'). Anche il momento totale è conservato, componente per componente: L_{tot} cambia dello -0.0311% sull'intera simulazione; 2.22(a): orbita ellittica del pianeta, orientabile in 3D, rappresentata in un file Geogebra.

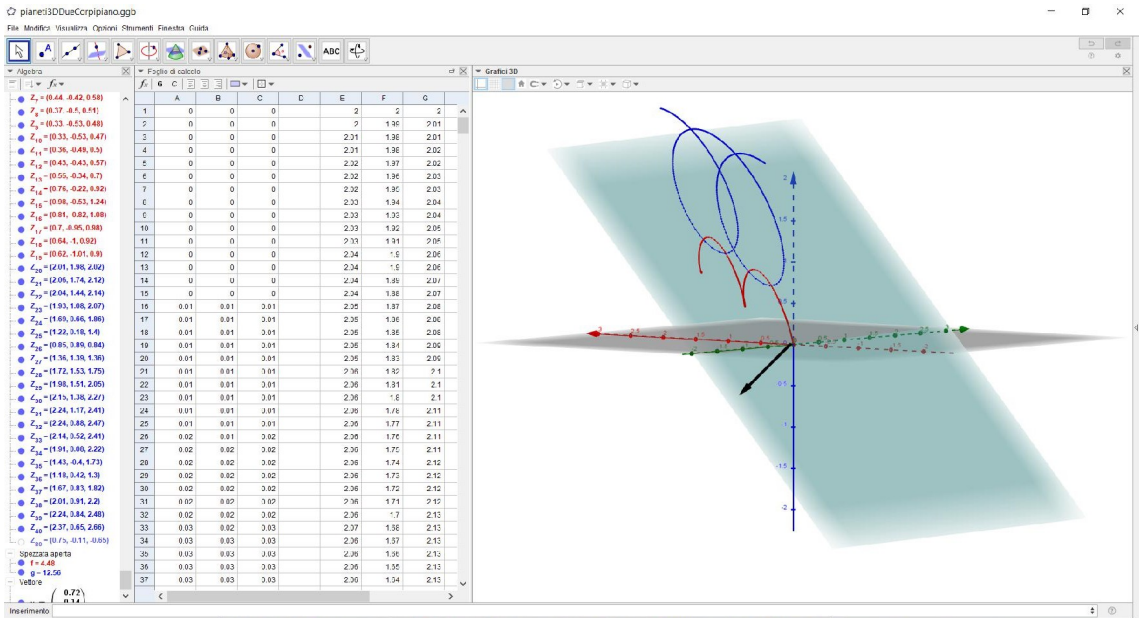
```
Z1[0]=Z01-zG;  Vz1[0]=Vz01-vzG;
X2[0]=X02-xG; (...)          // traslazione per il pianeta
TS=Ti;
XS1=X1[0];     VxS1=Vx1[0];  // inizializzazione vettori Sole
YS1=Y1[0];     VyS1=Vy1[0];
ZS1=Z1[0];     VzS1=Vz1[0];
XS2=X2[0];     (...)        // inizializzazione vettori pianeta

for(i=1;i<Nt;i++)                          // inizio ciclo
  {
  RKSTEP(TS,XS1,VxS1,YS1,VyS1,ZS1,VzS1,XS2,VxS2,YS2,VyS2,ZS2,VzS2,dt);  // passo
    del ciclo (integrazione)
  xG=(m1*XS1+m2*XS2)/(m1+m2); // nuove caratteristiche del CdM
  yG=(m1*YS1+m2*YS2)/(m1+m2);
  zG=(m1*ZS1+m2*ZS2)/(m1+m2);
  vxG=(m1*VxS1+m2*VxS2)/(m1+m2);
  vyG=(m1*VyS1+m2*VyS2)/(m1+m2);
  vzG=(m1*VzS1+m2*VzS2)/(m1+m2);
  XS1=XS1-xG;   VxS1=VxS1-vxG;   // nuova traslazione per il Sole
  YS1=YS1-yG;   VyS1=VyS1-vyG;
  ZS1=ZS1-zG;   VzS1=VzS1-vzG;
  XS2=XS2-xG;   (...)            // nuova traslazione per il pianeta
  T[i]=TS;      // caricamento nei vettori di output
  X1[i]=XS1;  Vx1[i]=VxS1;  // Sole
  Y1[i]=YS1;  Vy1[i]=VyS1;
  Z1[i]=ZS1;  Vz1[i]=VzS1;
  X2[i]=XS2;  Vx2[i]=VxS2;  // pianeta
  (...)
  }
}
```

Riportando i risultati numerici in immagini 3D orientabili, con `Geogebra`, si possono studiare aspetti molto interessanti. In Fig. 2.23(a) si mostra come appare in un riferimento inerziale il moto già rappresentato in Fig. 2.22(a): invece della sola orbita del pianeta si osserva anche quella del Sole, che inizialmente fermo nell'origine, lo segue. Come evidenziato rappresentando un piano per tre punti di una sola delle orbite, entrambe si trovano su di esso, che risulta *ortogonale al momento angolare totale del sistema*, conservato, di cui si mostra il versore, coincidente in questo caso con quello del momento angolare iniziale del solo pianeta (dato che il Sole parte da fermo).

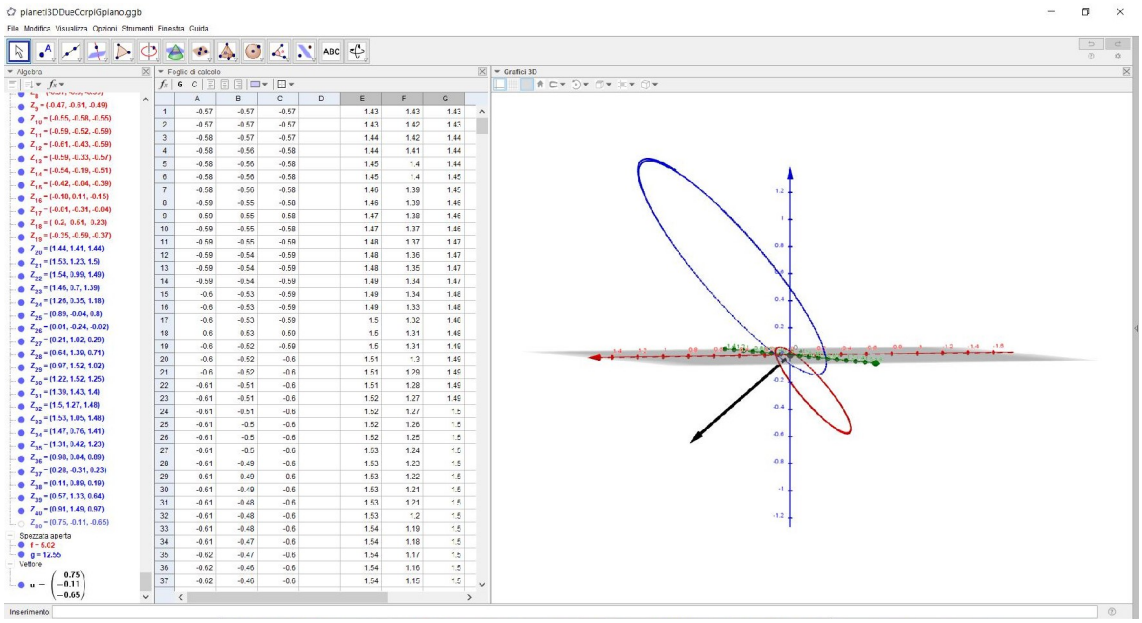
In Fig. 2.24(a) l'importanza di quest'ultimo risultato è evidenziata in un sistema nel quale le due parti, nel sistema inerziale, hanno quantità di moto iniziali *rappresentate da vettori sghembi*. I due vettori momento angolare:

$$\vec{L}_i = \vec{r}_{0,i} \times \vec{p}_{0,i} = m_i \begin{pmatrix} \hat{i} & \hat{j} & \hat{k} \\ x_{0,i} & y_{0,i} & z_{0,i} \\ v_{x0,i} & v_{y0,i} & v_{z0,i} \end{pmatrix} \quad i = 1, 2 \qquad \vec{L}_t = \vec{L}_1 + \vec{L}_2 \quad (2.14)$$

e la loro somma $\vec{L}_t$ (si mostrano i versori di tutti e tre) sembrano infatti del tutto estranei alle caratteristiche delle due orbite che si avvitano nello spazio. Riferendo invece al CdM tanto le traiettorie (traslando di $\vec{r}_G(t)$ le posizioni istantanee), quanto il calcolo dei momenti angolari delle parti e totale (traslando di $\vec{v}_G(t)$ le velocità istantanee), ecco che si ottiene Fig. 2.24(b): *le orbite sono ancora entrambe sul piano ortogonale al momento angolare totale*, il cui versore, fra l'altro, coincide con

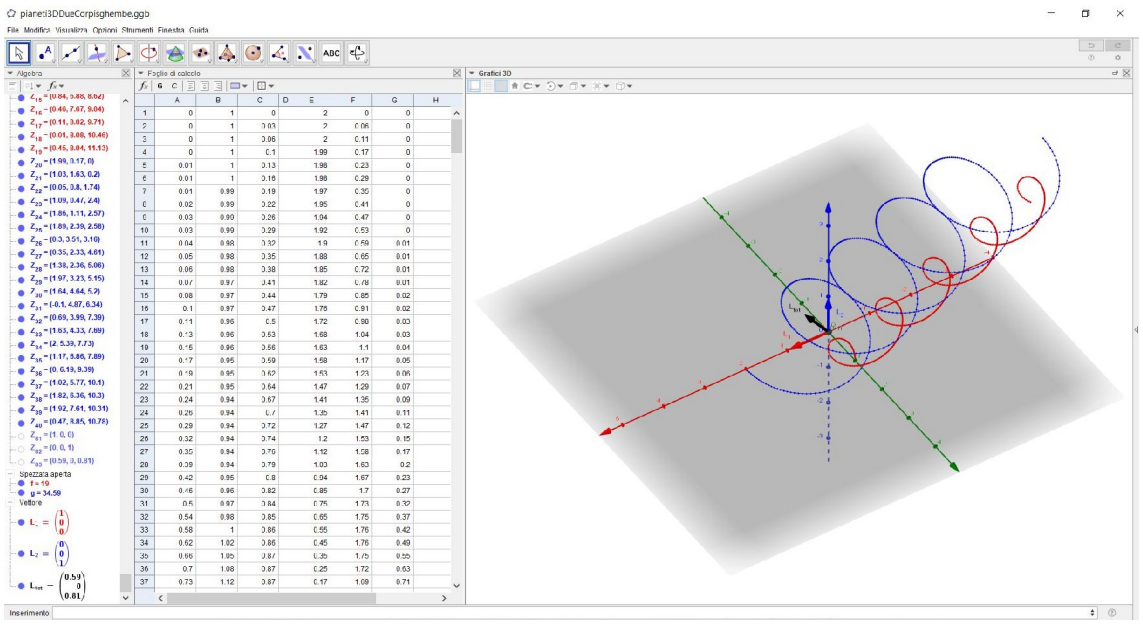


(a)

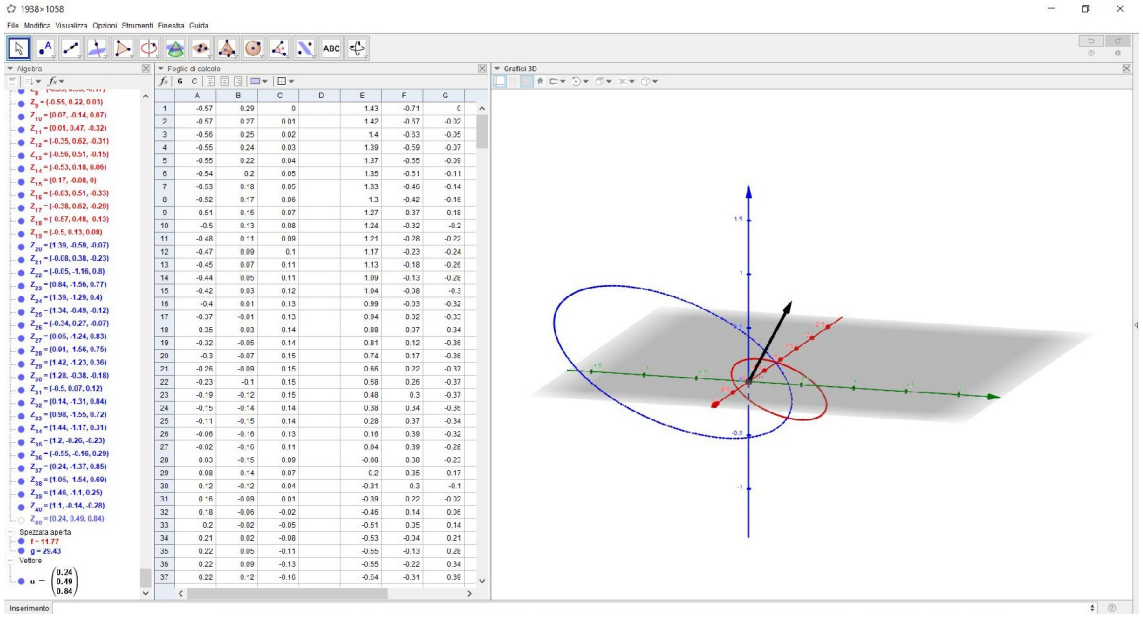


(b)

Figura 2.23: Rappresentazioni 3D orientabili (da Geogebra) dei risultati di simulazioni C++ del sistema Sole-pianeta mostrato in Fig. 2.22(a) ($m_1 = 10000.0, m_2 = 4000, T_i = 0.0, T_f = 0.5, Nt = 1000, x_{01} = 0.0, y_{01} = 0.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 0.0, x_{02} = 2.0, y_{02} = 2.0, z_{02} = 2.0, v_{0x2} = 8.0, v_{0y2} = -16, v_{0z2} = 12$). 2.23(a): in un riferimento inerziale; 2.23(b): in un riferimento centrato nel CdM. Le unità sono tali che la costante di gravitazione universale diventi $G = 1$. Il vettore rappresenta sempre il versore del momento angolare totale *iniziale*.



(a)



(b)

Figura 2.24: Rappresentazioni 3D orientabili (da `Geogebra`) dei risultati di simulazioni `C++` di un sistema Sole-pianeta con condizioni iniziali $m_1 = 10000.0, m_2 = 4000.0, T_i = 0.0, T_f = 0.5, Nt = 500, x_{01} = 0.0, y_{01} = 1.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 32.0, x_{02} = 2.0, y_{02} = 0.0, z_{02} = 0.0, v_{0x2} = 0.0, v_{0y2} = 55.0, v_{0z2} = 0.0$, tutte in unità tali che la costante di gravitazione universale diventi $G = 1$. 2.24(a): in un riferimento inerziale; 2.24(b): in un riferimento centrato nel CdM, dove le orbite sono sul piano ortogonale alla somma dei vettori momento angolare dei due pianeti (posti volutamente, nel riferimento inerziale, su rette sghembe).

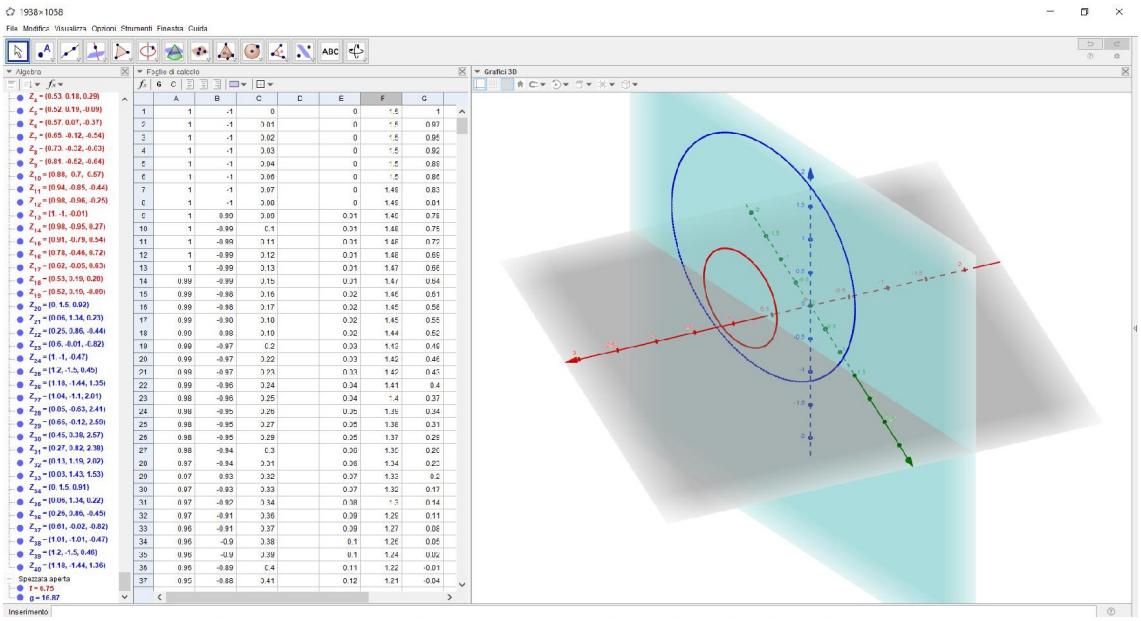


Figura 2.25: Orbite circolari ottenute nel sistema inerziale per $m_1 = 10000, m_2 = 4000, T_i = 0.0, T_f = 0.3, Nt = 500, x_{01} = 1.0, y_{01} = -1.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 18.0702, x_{02} = 0.0, y_{02} = 1.5, z_{02} = 1.0, v_{0x2} = 0.0, v_{0y2} = 0.0, v_{0z2} = -45.1754$, tutte in unità tali che la costante di gravitazione universale diventi $G = 1$. Le velocità necessarie ad avere orbite circolari sono state calcolate dalle Eq. 2.15.

entrambi i versori dei momenti angolari (sempre riferiti al CdM), del pianeta e del Sole.

In Fig. 2.25 si mostra infine un risultato apparentemente "'magico"'. I codici in App. A.2 suggeriscono di scegliere $x_i = -\frac{m_j x_j}{m_i}, y_i = -\frac{m_j y_j}{m_i}, z_i = -\frac{m_j z_j}{m_i}$ qualora si voglia avere subito il CdM nell'origine, in accordo con l'intera teoria costruita (Par. 1.1.4). Molto più interessante è quindi *il calcolo a priori dei valori dei moduli delle velocità che devono avere entrambe le parti qualora si desideri che si muovano di moto circolare uniforme*, dato da:

```
1 v1circ=m2*sqrt(G/((m1+m2)*sqrt((X01-X02)*(X01-X02)+(Y01-Y02)*(Y01-Y02)+(Z01-Z02)*(
      Z01-Z02))));
2 v2circ=-m1*v1circ/m2;
```

La dimostrazione di ciò è la seguente. Nell'Eq. 1.53 si è mostrato come la velocità angolare con cui ruotano i due corpi nel tempo attorno al loro CdM è $\Omega(t) = \sqrt{\frac{Gm_j^3}{r_i^3(m_1+m_2)^2}}(1 - \epsilon\cos\theta)$. Ora, se nel riferimento del CdM il corpo 1 si muove di moto circolare uniforme, allora $\epsilon = 0$ e la sua velocità tangenziale sarà $v_1 = \Omega r_1$, con Ω ed r_1 costanti nel tempo. In tale riferimento, d'altra parte, deve anche essere $m_1 r_1 = m_2 r_2$, che con $r_{12} = r_1 + r_2$ fornisce $r_1 = \frac{m_2}{m_1+m_2} r_{12}$. Ne segue il valore di v_1 scritto nel codice, cui si aggiunge quello che deve avere v_2 affinché il CdM sia nell'origine:

$$v_1 = \Omega r_1 = \sqrt{\frac{Gm_2^3}{r_1(m_1+m_2)^2}} = \sqrt{\frac{Gm_2^2}{r_{12}(m_1+m_2)}} \qquad v_2 = -\frac{m_1}{m_2} v_1 \tag{2.15}$$

Da tutto ciò si deduce che *qualunque siano le posizioni iniziali e le masse, pur non sapendo dov'è il CdM del sistema, per avere due moti circolari uniformi attorno ad esso è sufficiente che i moduli delle velocità iniziali dei due corpi abbiano i valori calcolati nelle Eq. 2.15* (per semplicità, basta assegnarli ad una sola componente, con le altre nulle). Dando condizioni iniziali arbitrarie in un sistema inerziale, prevedere subito quali orbite siano circolari è un risultato abbastanza sorprendente, qualora sfugga, vista la dimostrazione, che si è imposto fin dall'inizio che r_{12} fosse costante, trovato proprio dalle condizioni iniziali.

Capitolo 3

Il problema dei 3 corpi

3.1 Dal N corpi a "'solo"' $N = 3$

Nel Par.1.1.5 si è introdotto il problema di determinare il moto di N stelle o pianeti sotto l'azione dalle loro reciproche interazioni gravitazionali. Come già per N=2, essendovi fra tutti solo forze interne, in un riferimento inerziale il loro centro di massa (CdM) *complessivo* deve muoversi di moto rettilineo uniforme, e devono conservarsi l'energia *totale* E_{tot} ed il vettore momento angolare *complessivo* $\vec{L}_{tot}$.

Tali condizioni sono necessarie ma non sufficienti a trovare cambiamenti di riferimento che permettano di determinare le orbite (Fig. 3.1). Certamente, conviene studiare le equazioni in un riferimento centrato nel CdM, ma se per 2 corpi si sceglie il riferimento siderale, centrato nel CdM e posto nel piano delle due orbite (riferite agli assi), per N corpi si dovrebbe scegliere il piano ortogonale al vettore $\vec{L}_{tot}$ senza averne tuttavia un vantaggio certo, dato che *nella maggioranza dei casi gli N corpi non hanno orbite su un piano comune*. Se per $N = 3$ tale affermazione può non convincere (visto che per tre punti dello spazio passa uno ed un solo piano), ma è smentita dall'osservazione sperimentale (Fig. 1), per $N \geq 4$ è di immediata dimostrazione: basta che le condizioni iniziali dei corpi ne pongano solo tre su un piano. Le difficoltà sono anche maggiori qualora si cerchi di definire un riferimento sinodale per $N \geq 3$, in quanto *nel caso generale il CdM non è allineato con due dei corpi*, e non si dispone pertanto di un criterio assoluto per selezionarne una coppia fra le N parti, attraverso le cui posizioni istantanee definire un asse x_{sin}.

La strategia più scientificamente consapevole è rimandare la soluzione esatta del problema in attesa di una dimostrazione che la determini o che ne provi l'inesistenza[1], e procedere nei soli casi in cui si sa operare con metodo. Ad esempio, si può *stabilire quali corpi costituiscano "'il sistema"' e quali siano le interazioni ad esso interne od esterne*, valutando così *quanto* il moto dei corpi esterni influenzi quello di quelli interni. Già avere $N = 3$ richiede strategie diverse da quelle usate per $N = 2$. Come già detto, sebbene in un riferimento inerziale debba esistere un piano privilegiato

[1] È questo il caso per $N = 3$, come ha dimostrato Poincaré [34].

ortogonale a $\vec{L}_{tot}$, *i riferimenti siderale e sinodale per* $N = 3$ *potranno definirsi come per* $N = 2$ *solo qualora il terzo corpo abbia massa così piccola rispetto a quella degli altri due da non perturbare significativamente le loro posizioni e velocità istantanee*, condizione necessaria a considerare ancora il CdM sempre allineato con essi e la loro velocità di rotazione attorno ad esso solo "'perturbata"'. In caso contrario, si può passare al sistema sinodale calcolando istante per istante gli angoli della doppia rotazione rotazione descritta nel Par.1.1.5 (Eqq. 1.56, 1.57).

Tale complessità già solo per $N = 3$ mostra come didatticamente basti affrontare solo *il problema dei 3 corpi* per sollevare molte questioni epistemologiche. Una è la già detta necessità di distinguere fra un caso *ideale*, nel quale i due corpi di masse non trascurabili (da adesso in poi detti *principali*) determinano l'orbita del terzo muovendosi tuttavia come se esso non ci fosse, e i casi *reali*, dove *le interazioni vanno invece tutte considerate, in qualunque riferimento* (Par. 1.1.5). Nel caso ideale la dinamica dei due corpi principali sarà quella del problema di Kepler in qualunque riferimento, *suggerendo di studiare pertanto la dinamica del solo terzo corpo nel sistema sinodale* (Par. 1.1.4). Nel caso reale si può definire comunque il sistema siderale collettivo, o più riferimenti sinodali (centrati nel CdM dei due soli corpi che si scelgono come principali), ma quel che si è scoperto sul caos deterministico o sul teorema KAM [35], fa sì che si possa arrivare con gran fatica a risultati anche interessanti, come orbite periodiche ad N parti, ma comunque particolari. La distinzione didattica fra casi ideali e reali serve infatti anche a superare consuetudini storiche rischiose, come parlare di *masse trascurabili* invece di *interazioni tracurabili*, o sottolineare l'estrema particolarità del *problema dei 3 corpi ideale circolare*, caso in cui i corpi principali sono in moto circolare uniforme, o del *problema dei 3 corpi ideale, circolare e ristretto*, in cui per di più i 3 corpi si muovono su un piano.

3.2 Forze apparenti nel riferimento sinodale

Fintanto che si resta nel caso ideale, un'ottima strategia per ottenere la traiettoria del terzo corpo in un sistema sinodale è porre i due corpi principali nelle posizioni in cui sarebbero in esso (in generale, variabili nel tempo), aggiungendo alle forze gravitazionali che applicano al terzo corpo nel riferimento siderale (inerziale) quelle apparenti, che esso vede nel riferimento sinodale (non inerziale perché in moto rotatorio accelerato). Per trovarle, siano $O(x, y, z)$ il sistema siderale, con l'asse z ortogonale al piano in cui si muovono i due corpi principali, e $O'(x', y', z')$ il sistema sinodale. In esso, la posizione di un oggetto del sistema O sarà espressa mediante coordinate e versori che per O sono funzioni del tempo:

$$\vec{r} = x'(t)\hat{i}'(t) + y'(t)\hat{j}'(t) + z'(t)\hat{k}'(t) \tag{3.1}$$

Le velocità rilevate da O, derivando, sono quindi:

$$\begin{aligned}\dot{\vec{r}} = \dot{x}'(t)\hat{i}'(t) + x'(t)\dot{\hat{i}}'(t) + \dot{y}'(t)\hat{j}'(t) + y'(t)\dot{\hat{j}}'(t) + \dot{z}'(t)\hat{k}'(t) + z'(t)\dot{\hat{k}}'(t) = \\ v'_x(t)\hat{i}'(t) + v'_y(t)\hat{j}'(t) + v'_z(t)\hat{k}'(t) + x'(t)\dot{\hat{i}}'(t) + y'(t)\dot{\hat{j}}'(t) + z'(t)\dot{\hat{k}}'(t)\end{aligned} \tag{3.2}$$

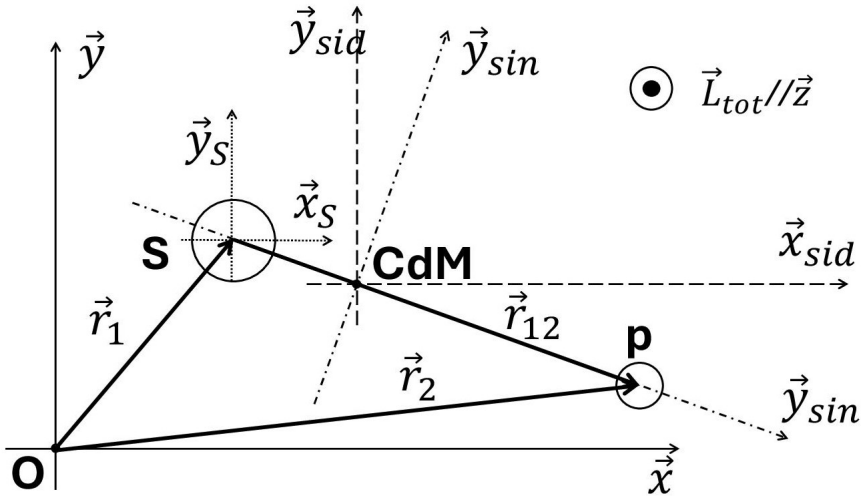


Figura 3.1: Si riporta qui Fig. 1.5, al fine di ricordare i 4 riferimenti in cui è studiato il problema dei 2 corpi: : **riferimento inerziale** (x, y) (assi a linee continue), fermo e centrato in O, con il *S*ole in r_1 ed il *p*ianeta in r_2; **riferimento in cui S è fermo**, (x_S, y_S) (linee puntate), in genere non inerziale; **riferimento siderale** (x_{sid}, y_{sid}) (tratti), centrato nel Centro di Massa (CdM), inerziale; **riferimento sinodale** (x_{sin}, y_{sin}) (tratto-punto), non inerziale, dove il CdM G è fermo ma gli assi ruotano nel tempo in modo che l'asse x_{sin} passi sempre per le due parti del sistema. Tutti sono nel piano su cui si trovano le orbite, ortogonale al vettore momento angolare totale $\vec{L}_{tot}$, pensato lungo l'asse z, in chiralità destrorsa.

La derivata temporale di un versore si deduce dalla trasformazione delle coordinate in una rotazione. In due dimensioni:

$$x(t) = x'(t)\cos\alpha(t) - y'(t)\sin\alpha(t) \qquad y(t) = x'(t)\sin\alpha(t) + y'(t)\cos\alpha(t) \quad (3.3)$$

ossia, invertendo le equazioni (la matrice inversa è la trasposta), nel sistema rotante:

$$x'(t) = x(t)\cos\alpha(t) + y(t)\sin\alpha(t) \qquad y'(t) = -x(t)\sin\alpha(t) + y(t)\cos\alpha(t) \quad (3.4)$$

Rispetto al sistema O, i versori $\hat{i}'$ e $\hat{j}'$ del sistema O' sono dunque:

$$\hat{i}' = \hat{i}\cos\alpha(t) + \hat{j}\sin\alpha(t) \qquad \hat{j}' = -\hat{i}\sin\alpha(t) + \hat{j}\cos\alpha(t) \quad (3.5)$$

Dato che O vede i suoi versori fermi, interpreta la derivata di quelli di O' come:

$$\dot{\hat{i}}' = (-\hat{i}\sin\alpha(t) + \hat{j}\cos\alpha(t))\dot{\alpha}(t) \quad \dot{\hat{j}}' = (-\hat{i}\cos\alpha(t) - \hat{j}\sin\alpha(t))\dot{\alpha}(t) \quad (3.6)$$

Se si definisce il vettore velocità angolare $\vec{\Omega}$ con cui O' ruota rispetto ad O come:

$$\vec{\Omega}(t) = \dot{\alpha}(t)\hat{k} = \Omega(t)\hat{k} \quad (3.7)$$

si ottiene così che *la derivata di un versore è il suo prodotto vettore con* $\vec{\Omega}(t)$:

$$\vec{\Omega}\times\hat{i}' = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ 0 & 0 & \Omega \\ \cos\alpha(t) & \sin\alpha(t) & 0 \end{vmatrix} = \dot{\hat{i}}' \quad \vec{\Omega}\times\hat{j}' = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ 0 & 0 & \Omega \\ -\sin\alpha(t) & \cos\alpha(t) & 0 \end{vmatrix} = \dot{\hat{j}}' \quad (3.8)$$

Un teorema di Eulero [21] assicura che ciò vale anche in tre dimensioni[2], quindi l'Eq 3.2 diventa:

$$\vec{v}(t) = \dot{\vec{r}}(t) = \vec{v}'(t) + \vec{\Omega}(t) \times (x'(t)\hat{i}'(t) + y'(t)\hat{j}'(t) + z'(t)\hat{k}'(t)) = \vec{v}'(t) + \vec{\Omega}(t) \times \vec{r}'(t) \quad (3.9)$$

dove v' è la velocità nel sistema O'.

L'accelerazione $\vec{a}(t) = \ddot{\vec{r}}(t)$ si calcola derivando il vettore velocità rispetto al tempo e ottenendo alla fine:

$$\vec{a}(t) = \vec{a}'(t) + 2\vec{\Omega}(t) \times \vec{v}'(t) + \vec{\Omega}(t) \times (\vec{\Omega}(t) \times \vec{r}'(t)) + \dot{\vec{\Omega}}(t) \times \vec{r}'(t) \quad (3.10)$$

In base all'Eq. 3.10, l'accelerazione in un sistema inerziale O espressa mediante le grandezze che caratterizzano in esso il moto di un sistema non inerziale O', risulta somma del-:

- l'accelerazione lineare a';
- l'accelerazione di Coriolis $\vec{\Omega}(t) \times \vec{v}'(t)$ (su una piattaforma circolare rotante, lanciando una pallina con velocità costante e radiale dal centro verso l'esterno o viceversa, viene deviata a destra o a sinistra a seconda del verso di rotazione);
- l'accelerazione centri*fuga* $\vec{\Omega}(t) \times (\vec{\Omega}(t) \times \vec{r}'(t))$ (un corpo nel sistema O' la percepisce verso l'esterno, per il 3. principio);
- l'accelerazione di Eulero $\dot{\vec{\Omega}}(t) \times \vec{r}'(t)$, nulla qualora $\vec{\Omega}$ sia costante nel tempo.

Ovviamente interessa di più l'accelerazione di un corpo nel sistema non inerziale O': invertendo Eq. 3.10 e moltiplicando per la sua massa, si vede che in O' esso sente quattro forze:

$$m\vec{a}'(t) = m\vec{a}(t) - 2m\vec{\Omega}(t) \times \vec{v}'(t) - m\vec{\Omega}(t) \times (\vec{\Omega}(t) \times \vec{r}'(t)) - m\dot{\vec{\Omega}}(t) \times \vec{r}'(t) \quad (3.11)$$

La prima è la (sola) forza cui sarebbe sottoposto nel sistema inerziale, le altre sono forze apparenti, dovute cioè non ad altri campi, ma solo al movimento del sistema non inerziale: la forza di Coriolis, la forza centrifuga e quella di Eulero, legata alla variazione della velocità angolare.

3.3 Moto dei tre corpi nel caso ideale

Applicheremo ora quanto visto nel Par. 3.2 al caso del problema ideale dei tre corpi, nel quale cioè i due di masse non trascurabili si muovano su traiettorie ellittiche. Da considerazioni geometriche, sappiamo già che nel piano xy del sistema siderale devono essere congiunti da un segmento di lunghezza $R(t)$ che ruota attorno all'origine $G = O = O'$ con pulsazione $\Omega^2(t) = G\frac{m_1+m_2}{R(r)^3}$. Nel sistema sinodale O' devono

[2]Senza troppo rigore: con tre versori, si ragiona come fatto per ogni coppia di essi.

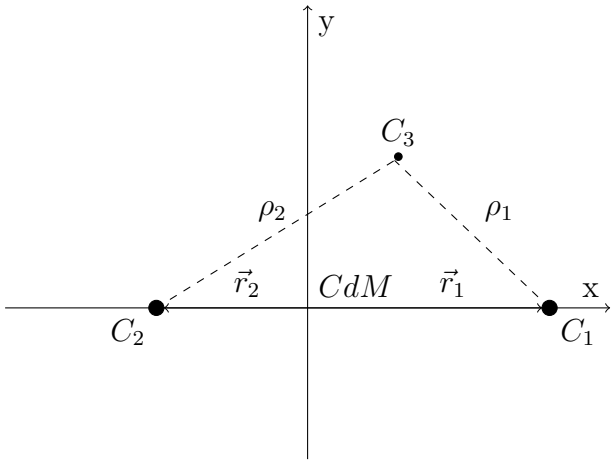


Figura 3.2: *Geometria del problema dei tre corpi nel sistema sinodale: definizione delle distanze ρ_1 e ρ_2 [3].*

invece oscillare in fase mantenendosi sul solo asse x'. Dovendo essere $G = O' = O$, in questo riferimento la loro distanza $R(t)$ sull'asse x' deve essere tale che:

$$m_1\vec{r}_1 + m_2\vec{r}_2 = 0 \qquad \vec{r}_1 + \vec{r}_2 = \vec{r}_{12} \tag{3.12}$$

Ma in O' i due vettori posizione partono dal centro $O' = G$ paralleli all'asse x'. Quindi, assumendo che $r'1$ abbia verso opposto all'asse, con i segni corretti (ossia, con già $x'_1 < 0$:

$$m_1x'_1 + m_2x'_2 = 0 \qquad x'_2 - x'_1 = r_{12} \tag{3.13}$$

così che, istante per istante:

$$x'_1 = -\frac{m_2}{m_1}x'_2 = -\frac{m_2}{m_1+m_2}r_{12} = -\alpha r_{12} \qquad x'_2 = \frac{m_1}{m_1+m_2}r_{12} = (1-\alpha)r_{12} \tag{3.14}$$

3.3.1 Il problema ideale circolare

Qualora i due corpi principali ruotino di moto circolare uniforme, il sistema si semplifica molto: nel sistema sinodale le loro posizioni lungo l'asse x' sono costanti nel tempo, così come la loro distanza R. Anche il vettore $\vec{\Omega} = \Omega\hat{k}$ è pertanto costante nel tempo, sempre lungo l'asse z, ortogonale al piano xy in cui i due corpi di masse non trascurabili ruotano:

$$\Omega^2 = G\frac{m_1+m_2}{r_{12}^3} \tag{3.15}$$

L'accelerazione totale di un eventuale terzo corpo nel sistema sinodale è dunque:

$$\vec{a}'(t) = \vec{a}(t) - 2\Omega(t) \times \vec{v}'(t) - \Omega(t) \times (\Omega(t) \times \vec{r}'(t)) \tag{3.16}$$

Le forze apparenti sono dunque solo due, in quanto $\vec{\Omega} = \Omega\hat{k}$, costante:

$$-2\vec{\Omega} \times \vec{v}' = 2 \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ 0 & 0 & \Omega \\ \dot{x}' & \dot{y}' & \dot{z}' \end{vmatrix} = 2\Omega(\dot{y}\hat{i} - \dot{x}\hat{j})$$

$$-\vec{\Omega} \times (\vec{\Omega} \times \vec{r}') = -\vec{\Omega} \times \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ 0 & 0 & \Omega \\ x' & y' & z' \end{vmatrix} = \Omega \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ 0 & 0 & \Omega \\ y' & -x' & 0 \end{vmatrix} = \Omega^2(x'\hat{i} + y'\hat{j}) \quad (3.17)$$

Così, componente per componente, per il solo terzo corpo, indicando Ω_0^2 per sottolineare che è costante:

$$\begin{cases} \ddot{x}' = a_x - 2\Omega_0\dot{y}' + \Omega_0^2 x' = -Gm_1 \frac{x'+\alpha r_{12}}{\rho_1^3} - \frac{x'-(1-\alpha)r_{12}}{\rho_2^3} - 2\Omega_0\dot{y}' + \Omega_0^2 x' \\ \ddot{y}' = a_y - 2\Omega_0\dot{x}' + \Omega_0^2 y' = -Gm_1 \frac{y'}{\rho_1^3} - \frac{y'}{\rho_2^3} - 2\Omega_0\dot{x}' + \Omega_0^2 y' \\ \ddot{z}' = a_z = -Gm_1 \frac{z'}{\rho_1^3} - \frac{z'}{\rho_2^3} \end{cases} \quad (3.18)$$

dove:

$$\rho_1(x,y,z) = \sqrt{(x'+\alpha r_{12})^2 + y'^2 + z'^2} \qquad \rho_2(x,y,z) = \sqrt{(x'+(1-\alpha)r_{12})^2 + y'^2 + z'^2} \quad (3.19)$$

Per comprendere come siano state trovate le espressioni delle accelerazioni a nel sistema inerziale, si torni all'immagine del sistema siderale. In esso la forza gravitazionale sul terzo corpo dipende dalle distanze r_1 e r_2, le quali non cambiano fra O e O' (le rotazioni sono isometrie: non cambiano le distanze) e devono comunque esprimersi in funzione delle coordinate sinodali per ottenere equazioni differenziali esplicite (cosa che, fra l'altro, è più facile con i due corpi fissi sull'asse x').

L'integrale di Jacobi

Determinate le equazioni nel caso circolare, si può proseguire togliendo gli apici: oramai è stabilito che si è nel sistema sinodale. Si nota allora che, come scoperto da Jacobi, definita la funzione:

$$U(x,y,z) = -\frac{Gm_1}{\rho_1} - \frac{Gm_2}{\rho_2} - \frac{1}{2}\Omega_0^2(x^2+y^2) \quad (3.20)$$

le sue derivate parziali sono tali che:

$$\partial_x U = 2\Omega_0\dot{y} - \ddot{x} \qquad \partial_y U = -2\Omega_0\dot{x} - \ddot{y} \qquad \partial_z U = -\ddot{z} \Rightarrow$$

$$-\frac{d}{dt}U = -\partial_x U\dot{x} - \partial_y U\dot{y} - \partial_z U\dot{z} = \dot{x}\ddot{x} + \dot{y}\ddot{y} + \dot{z}\ddot{z} = \frac{1}{2}\frac{d}{dt}(\dot{x}^2 + \dot{y}^2 + \dot{z}^2) \Rightarrow$$

$$\frac{d}{dt}(2U + v^2) = 0 \Rightarrow 2U + v^2 = C \quad (3.21)$$

Ne segue che deve esistere una costante C, detta *costante o integrale di Jacobi*, a seconda del cui valore, fissato dalle condizioni iniziali, viene stabilito dove può svolgersi il moto, perché per $v = 0$, si ha:

$$U = \frac{C}{2} = -\frac{Gm_1}{\rho_1} - \frac{Gm_2}{\rho_2} - \frac{1}{2}\Omega_0^2(x^2+y^2) \quad (3.22)$$

Variabile	**Unità**
Massa	UM, $1\,\text{UM} = 1,498 \cdot 10^{12}\,\text{Kg}$
Lunghezza	m
Tempo	s
Forza	$\text{UF} = \frac{\text{UM}\cdot\text{m}}{\text{s}^2}$, $1\,\text{UF} = 1,498 \cdot 10^{12}\,\text{N}$
Energia	$\text{UE} = \frac{\text{UM}\cdot\text{m}^2}{\text{s}^2}$, $1\,\text{UE} = 1,498 \cdot 10^{12}\,\text{J}$
Costante di gravitazione universale (G)	$\frac{\text{UF}\cdot\text{m}^2}{\text{UM}^2}$
Costante di Jacobi (C)	$\frac{\text{m}^2}{\text{s}^2}$

Tabella 3.1: *Unità di misura utilizzate e unità del Sistema Internazionale [3].*

Si tratta di curve nel piano xy che, traslate a qualunque quota sull'asse z, definiscono cilindri detti *zone di Hill che non possono essere attraversate nel moto*, in quanto su di esse la velocità del corpo è nulla, e non può pertanto superarle. Pertanto, *se il corpo parte all'interno di una zona di Hill, non può abbandonarla; se viceversa parte al di fuori di essa, non può accedervi.* In Fig. **??** si mostrano esempi di zone di Hill al variare delle masse m_1, m_2, delle posizioni C_1, C_2 e della distanza R tra i corpi principali e della costante di Jacobi C, ottenute mediante il software geogebra *https://www.geogebra.org/calculator/trzjkfcw* [3]. Nella scelta dei parametri, si nota innanzi tutto che:

$$\frac{C}{2} = -\frac{Gm_1}{\rho_1} - \frac{Gm_2}{\rho_2} - \frac{\Omega^2}{2}\left(x^2 + y^2\right) \tag{3.23}$$

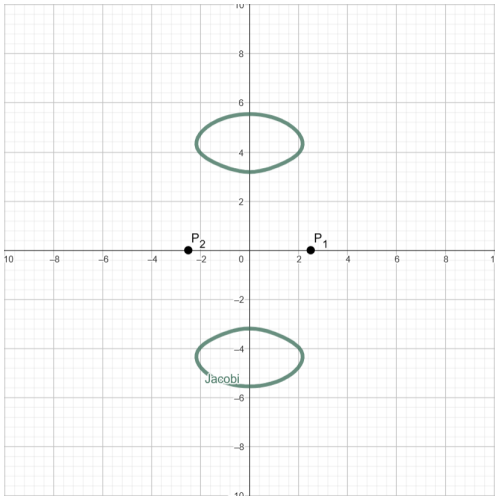
e considerando quindi che $\left(x^2 + y^2\right), G, m_1, m_2, \rho_1, \rho_2, \Omega^2$ sono tutti valori positivi:

$$\frac{C}{2} + G\left(\frac{m_1}{\rho_1} + \frac{m_2}{\rho_2}\right) < 0 \iff C < -2G\left(\frac{m_1}{\rho_1} + \frac{m_2}{\rho_2}\right) \tag{3.24}$$

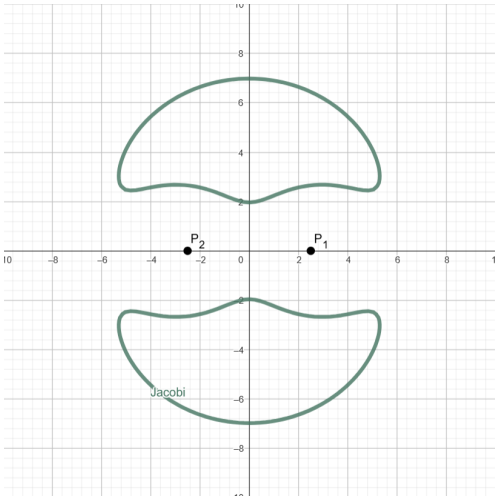
Inoltre, è necessario specificare le unità di misura utilizzate: le masse m_1 ed m_2 vengono espresse in UM, la distanza fra i due corpi R in m, la costante di Jacobi C in $\frac{\text{m}^2}{\text{s}^2}$ mentre la costante di gravitazione universale G in $\frac{\text{UF}\cdot\text{m}^2}{\text{UM}^2}$. I fattori di conversione con le unità del Sistema Internazionale sono riportati in Tab. 3.1 [3].

I punti di Lagrange

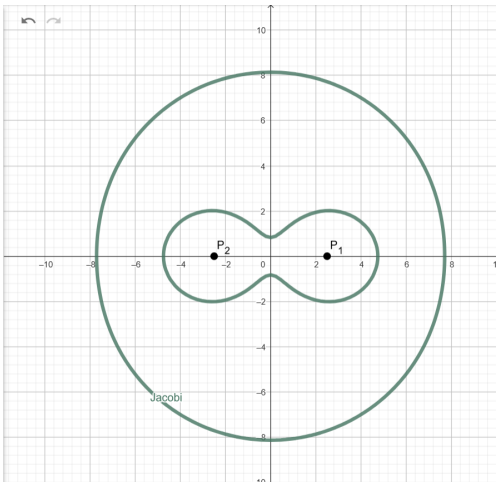
Imponendo che l'accelerazione e la velocità del terzo corpo siano nulle componente per componente, si trovano i *punti di Lagrange*, punti dove esso dovrebbe dunque restare fermo, sottoposto ad una risultante nulla. Per trovare tali punti, annulliamo



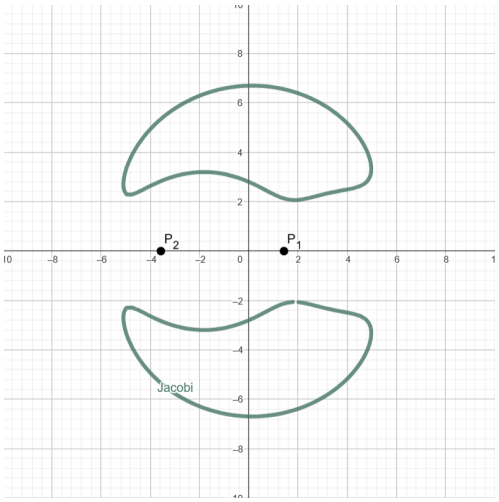
(a) $m_1 = 2, m_2 = 2, R = 5, C = -230$



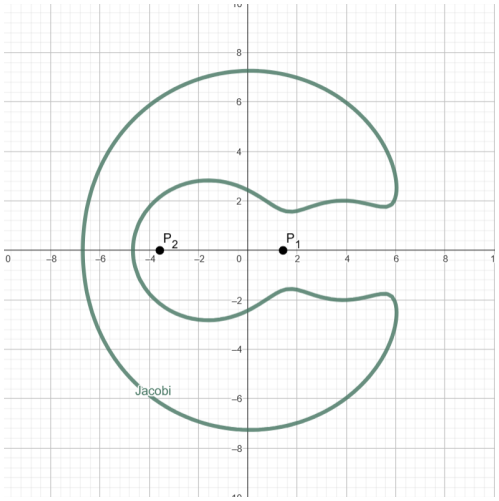
(b) $m_1 = 2, m_2 = 2, R = 5, C = -264$



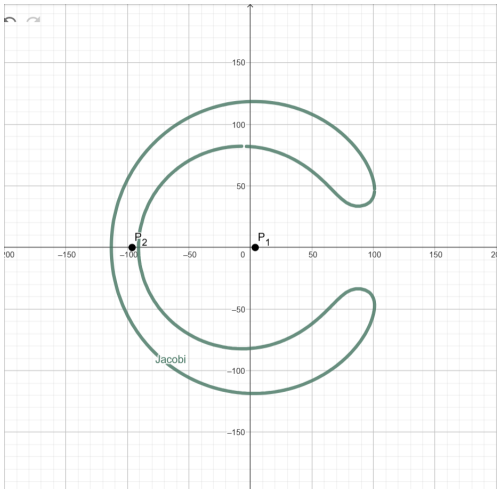
(c) $m_1 = 2, m_2 = 2, R = 5, C = -306$



(d) $m_1 = 5, m_2 = 2, R = 5, C = -450$



(e) $m_1 = 5, m_2 = 2, R = 5, C = -480$



(f) $m_1 = 50, m_2 = 2, R = 100, C = -160$

Figura 3.3: Zone di Hill, ottenute assumendo $G = 100$ in *https://www.geogebra.org/calculator/trzjkfcw* (unità di misura in Tab. 3.1) [3].

le equazioni:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -G\left(\dfrac{m_1(x-MR)}{\rho_1^3} + \dfrac{m_2(x+(1-M)R)}{\rho_2^3}\right) + \Omega^2 x \\ -G\left(\dfrac{m_1 y}{\rho_1^3} + \dfrac{m_2 y}{\rho_2^3}\right) + \Omega^2 y \\ -G\left(\dfrac{m_1 z}{\rho_1^3} + \dfrac{m_2 z}{\rho_2^3}\right) \end{bmatrix} \tag{3.25}$$

Sostituendo poi $\Omega^2 = G\frac{m_1+m_2}{R^3}$, si ottiene:

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \dfrac{m_1(x-MR)}{\rho_1^3} + \dfrac{m_2(x+(1-M)R)}{\rho_2^3} - \dfrac{m_1+m_2}{R^3}x \\ \dfrac{m_1 y}{\rho_1^3} + \dfrac{m_2 y}{\rho_2^3} - \dfrac{m_1+m_2}{R^3}y \\ \dfrac{m_1 z}{\rho_1^3} + \dfrac{m_2 z}{\rho_2^3} \end{bmatrix} \tag{3.26}$$

Sostituendo ancora ρ_1 e ρ_2, si ottiene il sistema:

$$\begin{cases} 0 = \dfrac{m_1(x-MR)}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \dfrac{m_2(x+(1-M)R)}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} - \dfrac{m_1+m_2}{R^3}x \\ 0 = y\left(\dfrac{m_1}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \dfrac{m_2}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} - \dfrac{m_1+m_2}{R^3}\right) \\ 0 = z\left(\dfrac{m_1}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \dfrac{m_2}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3}\right) \end{cases} \tag{3.27}$$

Se ora introduciamo una nuova variabile K, definita come:

$$K = \frac{m_1}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \frac{m_2}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \tag{3.28}$$

possiamo riscrivere il sistema come:

$$\begin{cases} 0 = x\left(K - \dfrac{m_1+m_2}{R^3}\right) + KMR - \dfrac{m_2 R}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \\ 0 = y\left(K - \dfrac{m_1+m_2}{R^3}\right) \\ 0 = zK \end{cases} \tag{3.29}$$

La seconda e la terza equazione si possono annullare in tre modi: se $y = 0$ e $z = 0$, se $y = 0$ e $K = 0$ o se $z = 0$ e $K = \frac{m_1+m_2}{R^3}$. Analizziamo quindi singolarmente ciascun caso, vedendo come annullare anche la prima equazione [3].

- Ponendo $y = 0$ e $z = 0$ (primo caso), con $K \neq 0$ e $K \neq \frac{m_1+m_2}{R^3}$:

$$\begin{cases} 0 = x\left(K - \dfrac{m_1+m_2}{R^3}\right) + KMR - \dfrac{m_2 R}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \\ K = \dfrac{m_1}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \dfrac{m_2}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \end{cases}$$

$$\begin{cases} 0 = x\left(K - \dfrac{m_1+m_2}{R^3}\right) + KMR - \dfrac{m_2 R}{(x+(1-M)R)^3} \\ K = \dfrac{m_1}{(x-MR)^3} + \dfrac{m_2}{(x+(1-M)R)^3} \end{cases}$$

$$\implies \frac{m_2}{(x+(1-M)R)^3}R - \left(\frac{m_1}{(x-MR)^3} + \frac{m_2}{(x+(1-M)R)^3}\right)MR =$$

$$= x\left(\frac{m_1}{(x-MR)^3} + \frac{m_2}{(x+(1-M)R)^3} - \frac{m_1+m_2}{R^3}\right)$$

$$0 = \frac{m_1}{(x-MR)^3}\cdot(x+MR) + \frac{m_2}{(x+(1-M)R)^3}\cdot(x+(M-1)R) - \frac{m_1+m_2}{R^3}x$$

$$0 = -\frac{m_1}{(x-MR)^2} - \frac{m_2}{(x+(1-M)R)^2} - \frac{m_1+m_2}{R^3}x \quad (3.30)$$

Si ottiene pertanto un'equazione algebrica di quinto grado, il cui polinomio in x è detto *funzione di Lagrange*. Esso presenta tre rami separati da asintoti verticali, posti nelle ascisse dei due corpi principali, $x_1 = MR$ e $x_2 = -(1-M)R$. Il punto più vicino all'asse x di ogni ramo è un punto di Lagrange, da cui pertanto tre *punti collineari di Lagrange*: il primo, detto L_1 ha $x \in]x_1; x_2[$; il secondo L_2, ha $x \in]-\infty; x_1[$; il terzo L_3 ha $x \in]x_2; +\infty[$, come visibile in figura 3.4 [**?**].

- Ponendo $y = 0$ e $z \neq 0$, con $K = 0$ (secondo caso) si trova:

$$\begin{cases} 0 = x\left(K - \dfrac{m_1+m_2}{R^3}\right) + KMR - \dfrac{m_2 R}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \\ 0 = K = \dfrac{m_1}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \dfrac{m_2}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \end{cases}$$

$$\iff \begin{cases} \dfrac{m_1+m_2}{R^3}x = -\dfrac{m_2 R}{\sqrt{(x+(1-M)R)^2+z^2}^3} \\ \dfrac{m_1}{\sqrt{(x-MR)^2+z^2}^3} = -\dfrac{m_2}{\sqrt{(x+(1-M)R)^2+z^2}^3} \end{cases} \quad (3.31)$$

Se ci si sofferma sulla seconda equazione si nota subito come questa sia impossibile, perché la frazione a sinistra e quella a destra dell'uguale dovrebbero essere di segni discordi e diventare concordi grazie al segno $-$ presente a destra dell'uguale. Questo è però chiaramente impossibile perché sia entrambi i numeratori sono positivi, in quanto masse strettamente maggiori di 0, sia entrambi i denominatori sono positivi, in quanto risultato di una radice quadrata. Questo secondo caso non porta quindi a nessun altro punto di Lagrange, ma ci permette di comprendere come ogni punto di Lagrange debba forzatamente avere $z = 0$. L'impossibilità di risolvere questo sistema di equazioni è dovuta

dalle condizioni iniziali imposte. Abbiamo infatti fissato $K = 0$, il che è però impossibile in base alla definizione stessa di K.

- Ponendo $z = 0$, $y \neq 0$, con $K = \frac{m_1+m_2}{R^3}$ si ricava:

$$\begin{cases} 0 = x\left(K - \dfrac{m_1+m_2}{R^3}\right) + KMR - \dfrac{m_2 R}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \\ \dfrac{m_1+m_2}{R^3} = K = \dfrac{m_1}{\sqrt{(x-MR)^2+y^2+z^2}^3} + \dfrac{m_2}{\sqrt{(x+(1-M)R)^2+y^2+z^2}^3} \end{cases}$$

$$\iff \begin{cases} \dfrac{m_2 R}{\sqrt{(x+(1-M)R)^2+y^2}^3} = \dfrac{m_1+m_2}{R^2} M \\ \dfrac{m_1+m_2}{R^3} - \dfrac{m_1}{\sqrt{(x-MR)^2+y^2}^3} = \dfrac{m_2}{\sqrt{(x+(1-M)R)^2+y^2}^3} \end{cases}$$

$$\implies \frac{m_1+m_2}{R^3} M = \frac{m_1+m_2}{R^3} - \frac{m_1}{\sqrt{(x-MR)^2+y^2}^3}$$

$$\implies \left(\frac{m_1+m_2}{m_1 R^3}(M-1)\right)^{\frac{2}{3}} = \frac{1}{(x-MR)^2+y^2}$$

$$\iff y^2 = R^2 - x^2 + 2xMR - M^2R^2$$

Ne segue:

$$\frac{m_2 R}{\sqrt{x^2+R^2+M^2R^2+2xR-2xMR-2MR^2+R^2-x^2+2xMR-M^2R^2}^3} = \frac{m_1+m_2}{R^2} M$$

$$\frac{1}{\sqrt{2R^2+2xR-2MR^2}^3} = \frac{m_1+m_2}{m_2 R^3} \cdot \frac{m_2}{m_1+m_2} \iff \frac{1}{2R^2+2xR-2MR^2} = \left(\frac{1}{R^3}\right)^{\frac{2}{3}} = \frac{1}{R^2}$$

$$R^2 = 2R^2 + 2xR - 2MR^2 \iff x = \frac{R(2M-1)}{2}$$

da cui la soluzione dell'equazione precedente:

$$y^2 = R^2 - \left(\frac{R(2M-1)}{2}\right)^2 + 2MR\frac{R(2M-1)}{2} - M^2R^2$$

$$y^2 = R^2 - \frac{4M^2R^2-4MR^2+R^2}{4} + \frac{4M^2R^2-2MR^2}{2} - M^2R^2$$

$$y^2 = \frac{4R^2-4M^2R^2+4MR^2-R^2+8M^2R^2-4MR^2-4M^2R^2}{4}$$

$$y^2 = \frac{3}{4}R^2 \implies y = \pm\frac{\sqrt{3}}{2}R, x = \frac{R(2M-1)}{2} \tag{3.32}$$

Gli ultimi due punti di Lagrange, formano quindi ognuno un triangolo equilatero con vertici i due corpi C_1 e C_2 e se stessi, e vengono dunque chiamati *punti equilateri (o triangolari) di Lagrange* [**?**].

Ricapitolando dunque, i cinque punti di Lagrange sono:

$$L_1 = (]MR; -(1-M)R[\,; 0; 0) \quad L_2 = (]-\infty; MR[\,; 0; 0) \quad L_3 = (]-(1-M)R; +\infty[\,; 0; 0)$$

$$L_4 = \left(\frac{R(2M-1)}{2}; \frac{\sqrt{3}}{2}R; 0\right) \qquad L_5 = \left(\frac{R(2M-1)}{2}; -\frac{\sqrt{3}}{2}R; 0\right) \tag{3.33}$$

Nelle Figg. 3.4 si mostrano i punti di Lagrange ottenuti in sei casi, al variare delle masse m_1 e m_2 di C_1 e C_2 e della loro distanza R, mediante un calcolatore geogebra disponibile a *https://www.geogebra.org/calculator/x2ganjcv* (nelle unità di misura di Tab. 3.1) [3]. I due corpi sono nelle ascisse degli asintoti verticali della curva rappresentata; L_4, L_5 sono calcolati, L_1, L_2, L_3 sono in corrispondenza dei massimi locali o degli zeri della curva [17].

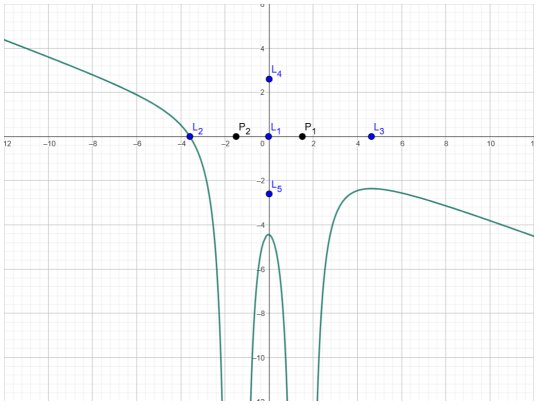
3.3.2 Il problema ideale ellittico

Qualora in condizioni ideali i due corpi principali non si muovano di moto circolare, conviene descrivere le loro orbite mediante quanto trovato studiando l'equazione di Kepler Eq. 1.34. Come visto nel Par. 1.1.3, partendo dai semiassi maggiori a_i, l'eccentricità ϵ ed il periodo T, calcolati ad esempio in base alle condizioni iniziali di uno dei due nel sistema siderale, *per ciascuno dei due corpi*, i, j si procede così:

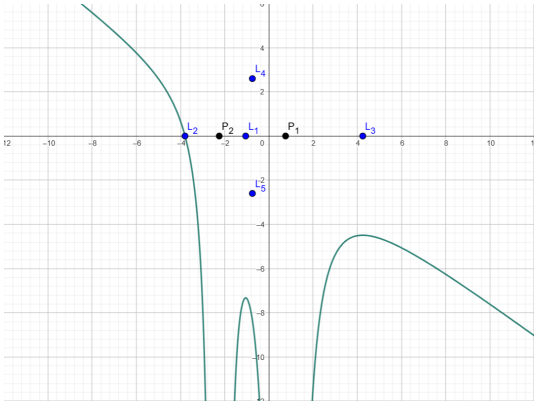
- noto T, si calcola $\Phi(t) = 2\pi \frac{t-t_0}{T}$ (Eq. 1.32);
- nota ϵ, si inverte l'equazione di Kepler Eq. 1.34, trovando $u(\Phi)$;
- da u si trova $\theta(u) = 2 \arctan\left(\sqrt{\frac{1+\epsilon}{1-\epsilon}} \tan \frac{u}{2}\right)$, Eq.1.36;
- note ai, a_j, da θ si trova $r_i(\theta) = a_i \frac{1-\epsilon^2}{1+\epsilon \cos\theta}$ (Eq.1.31), e volendo (x_i, y_i).

In tale algoritmo, le uniche grandezze diverse per i due corpi sono le r_i, *il che conferma di per sé l'omotetia delle orbite.*

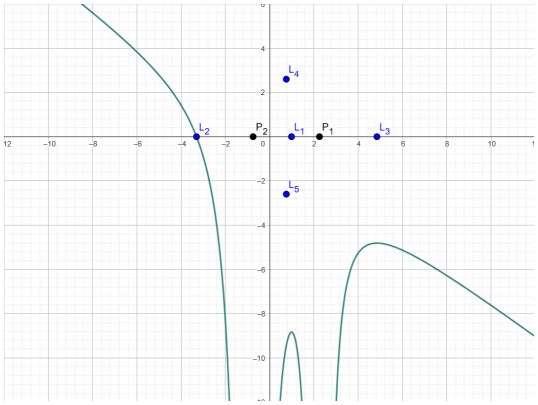
Trovate le coordinate dei due corpi, conviene porsi immediatamente nel riferimento sinodale e determinare in esso le forze sul terzo corpo. *Ovviamente, ciò è possibile solo nel caso ideale.*



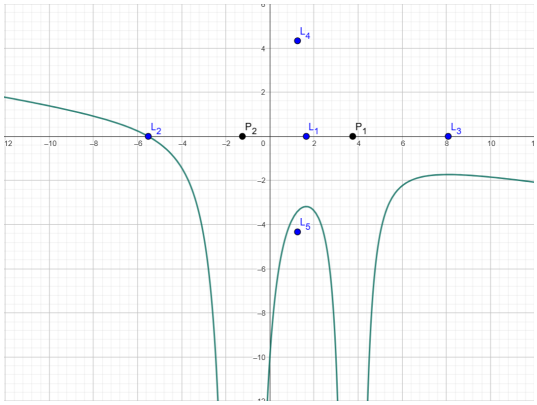
(a) $m_1 = 5, m_2 = 5, R = 3$



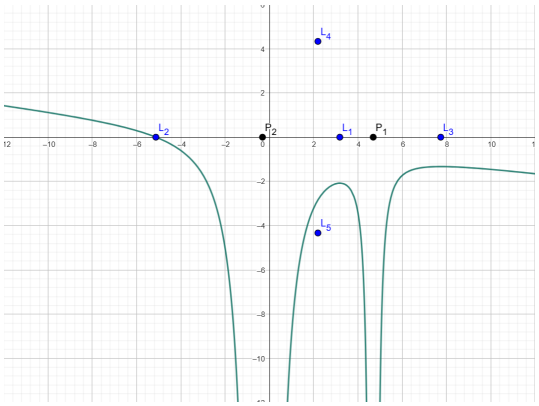
(b) $m_1 = 15, m_2 = 5, R = 3$



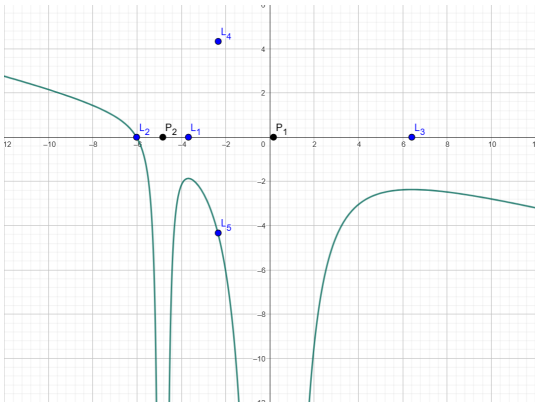
(c) $m_1 = 5, m_2 = 15, R = 3$



(d) $m_1 = 5, m_2 = 15, R = 5$



(e) $m_1 = 1, m_2 = 15, R = 5$



(f) $m_1 = 30, m_2 = 1, R = 5$

Figura 3.4: I cinque punti Lagrangiani calcolati dalla funzione di Lagrange mediante *https://www.geogebra.org/calculator/x2ganjcv* (nelle unità di misura di Tab. 3.1) [3].

Capitolo 4

Simulazione del moto dei 3 corpi

4.1 Codici C++

4.1.1 Tre stelle mobili in riferimenti inerziali

Per quanto detto in apertura del Cap. 3, simulare numericamente le traiettorie di tre corpi interagenti in un riferimento inerziale tridimensionale è tecnicamente molto più semplice che determinare da quali riferimenti il loro moto sia meglio interpretabile. Possibili codici si trovano nell'App. A.3, per un generico riferimento inerziale (A.3.1) o uno centrato nel CdM del sistema (A.3.1). Le caratteristiche di questi algoritmi sono del tutto analoghe a quanto visto per 2 corpi: nella definizione delle componenti delle tre accelerazioni va imposto il terzo principio della dinamica; si utilizza il solo algoritmo Runge-Kutta4 suddiviso in due funzioni (la `RK`, dove si operano le trasformazioni geometriche, la `RKSTEP` per l'integrazione); si estende a 3 corpi la funzione `RKSTEP`, unica parte nuova.

È ovviamente importante cercare casi studio che provino l'affidabilità del codice nel riprodurre almeno le proprietà fisiche sulle quali non si hanno dubbi. Come primo esempio, nelle Figg. 4.1 e 4.2 si mostrano i risultati della simulazione del sistema Sole-pianeta già mostrato nelle Figg. 2.22(a) e 2.23, a cui si aggiunge un terzo corpo con una velocità iniziale e due diversi valori della massa. In Fig. 4.1(a) si ha il caso in cui m_3=10 (espressa in unità tali che $G = 1$, al solito), molto inferiore delle masse delle altre due parti. Come evidente dai versori dei momenti angolari *iniziali* del corpo 2, $\hat{L}_{02}$, del corpo 3, $\hat{L}_{03}$ e del sistema totale $\hat{L}_{tot}$ (il corpo 1 è inizialmente fermo), $\hat{L}_{tot}$ *è quasi indistinguibile dal versore del momento iniziale della coppia costituita dai due corpi principali* ($\vec{L}_{01} + \vec{L}_{02} \approx \vec{L}_{tot}$)*, che si muovono sul piano ad esso ortogonale con orbite ellittiche pressoché imperturbate dal moto del terzo corpo*. Anche ponendo il CdM nell'origine (Fig. 4.1(b)), il terzo corpo *non si pone sul piano degli altri due*, che continua ad essere *quasi* ortogonale a $\vec{L}_{tot}$, *conservato nel tempo*. Si tratta in sostanza di un sistema pressoché ideale, il cui comportamento potrebbe essere approssimato da una simulazione in un sistema

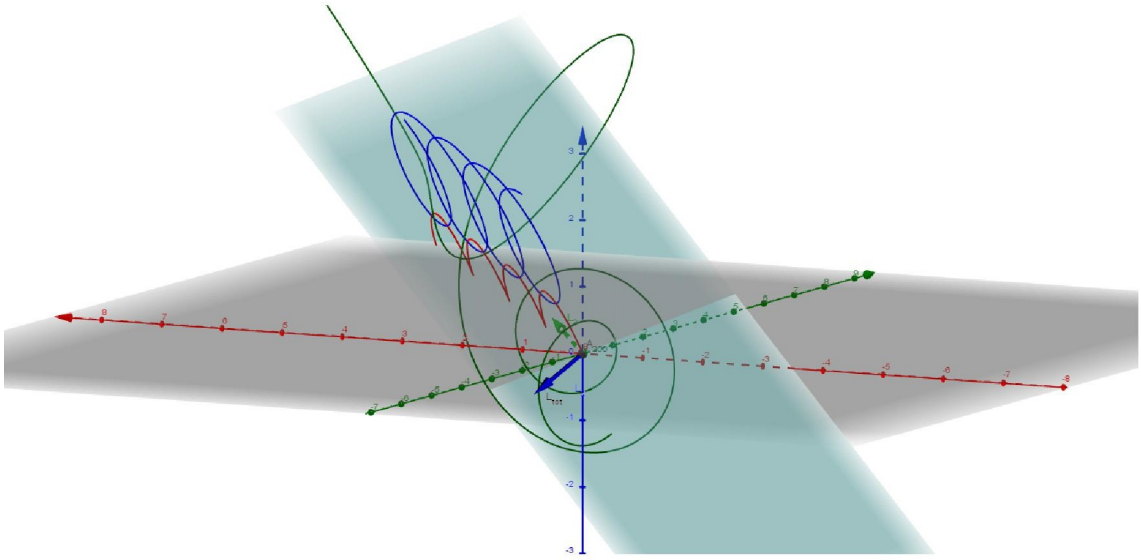
sinodale definito per due soli corpi, con il terzo soggetto a sole forze apparenti (Par. 3.2).

Nelle Figg. 4.3(a), 4.3(b), si riportano gli andamenti nel tempo di energia totale e momento angolare totale, rispettivamente, del sistema nel CdM. Come si vede, entrambe le grandezze sono *accettabilmente conservate*, nel senso che le approssimazioni numeriche ne alterano comunque il valore iterazione dopo iterazione (in special modo quando i corpi si avvicinano ed in particolare l'energia), ma sull'intero tempo di simulazione la variazione è regolare ed contenuta, come percentuale dei valori iniziali (sotto l'1% per l'energia, ancor meno per il momento). In sostanza, si tratta di un errore *sistematico*, controllabile aumentando il numero Nt dei passi di integrazione e diminuendo il tempo di simulazione $T_f - T_i$.

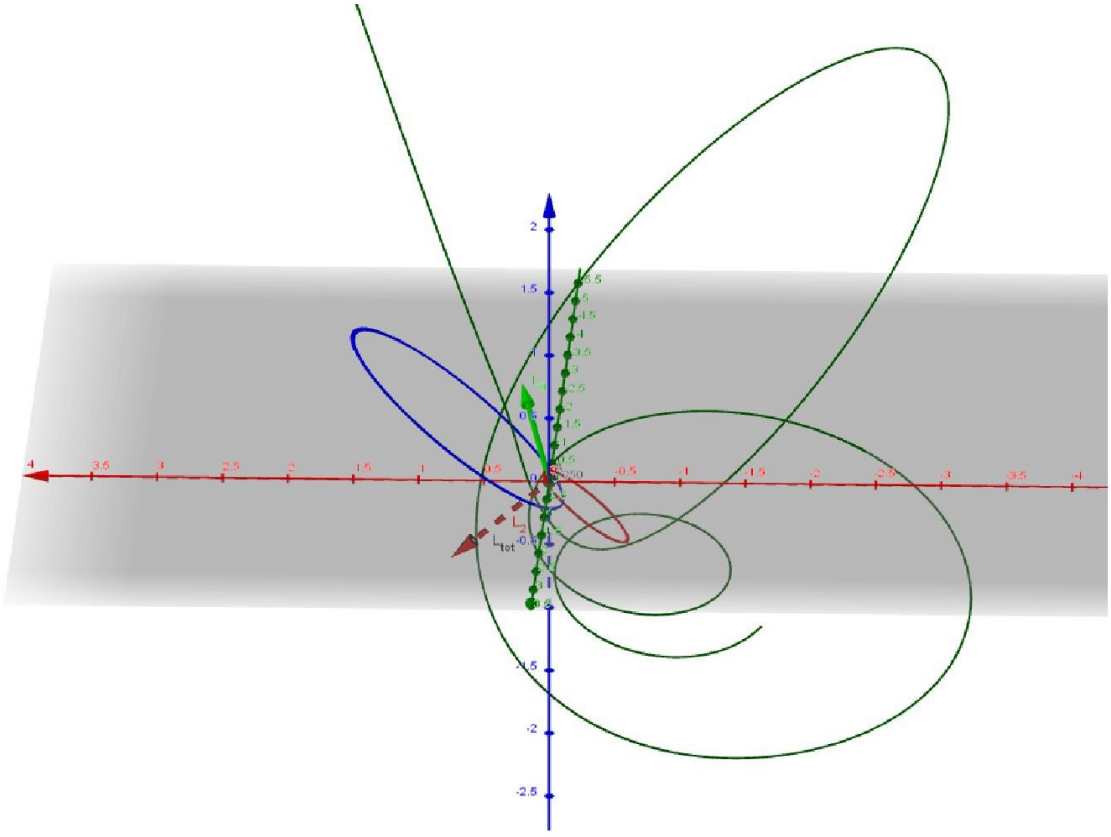
In Fig. 4.2(a) si vedono invece le orbite nel sistema inerziale qualora ai corpi nelle Figg. 2.22(a) e 2.23 si aggiunga un terzo corpo con m_3=2000, confrontabile con m_2, e la stessa velocità di Fig. 4.1. Stavolta $\hat{L}_{tot}$ è ben distinto sia da $\hat{L}_{02}$, sia da $\hat{L}_{03}$. Nessuna delle parti si muove pertanto su un piano identificabile già dai momenti angolari iniziali, che a parte $\vec{L}_{tot}$ cambiano nel tempo, e lo stesso si osserva nel riferimento del CdM (4.2(b)). Data l'assenza di criteri geometrici, è particolarmente importante stabilire se la simulazione sia attendibile, soprattutto considerando il punto di vista della computer graphics e dei videogames: per creare immagini *verosimiglianti*, bastano leggi empiriche (ad esempio, i modelli di illuminazione). L'esempio è infatti particolarmente significativo in quanto ad un certo punto della simulazione c'è un tale avvicinamento dei tre corpi da far sì che il terzo fugga via lasciando la coppia 1-2 legata su orbite ellittiche. Ora, bastano poche prove di simulazione per osservare che i casi in cui gli incontri generano errori numerici tali da invalidare del tutto i calcoli, finiscono con traiettorie lineari (si veda Fig. 2.12(a)), e *non con un moto a 2 corpi coerente con la teoria*. Ma allora, se dopo l'incontro *la simulazione mostra qualcosa che si è dimostrato essere possibile, perché dubitare dei suoi risultati ?*

Il primo passo verso una risposta viene dagli andamenti nel tempo di energia totale e momento angolare totale nel riferimento del CdM, in Figg. 4.3(c), 4.3(d) rispettivamente. Come si vede, al procedere delle iterazioni entrambe le grandezze passano da una *variazione accettabile, in termini di errore sistematico dovuto alle approssimazioni numeriche*, ad un brusco cambiamento dovuto all'espulsione del terzo corpo dal sistema, dopo il quale *energia e momento sono decisamente costanti*. In sostanza, il passaggio fra due "'fasi"' di fatto accettabili avviene con una *variazione notevole ed improvvisa delle grandezze che dovrebbero essere conservate*, e non si può pertanto ritenere affidabile quanto si ottiene *dopo il salto*. Ciò porta ovviamente a dover riflettere su cosa significhi simulare un sistema sapendo cosa sia il *caos deterministico* e la *sensibile dipendenza dalle condizioni iniziali*, potendo arrivare a spiegare le tecniche per *normalizzare* le equazioni nei sistemi caotici proprio per trattare i casi come quello mostrato [35].

Ma pur senza arrivare a tanto, dal punto di vista didattico certamente *apre una*

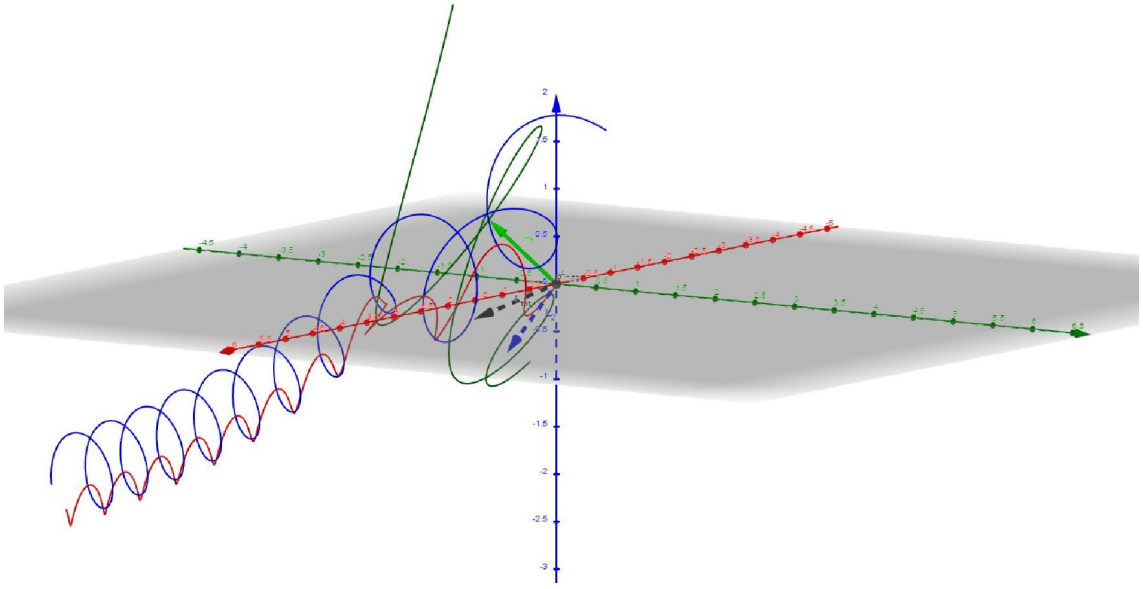


(a)

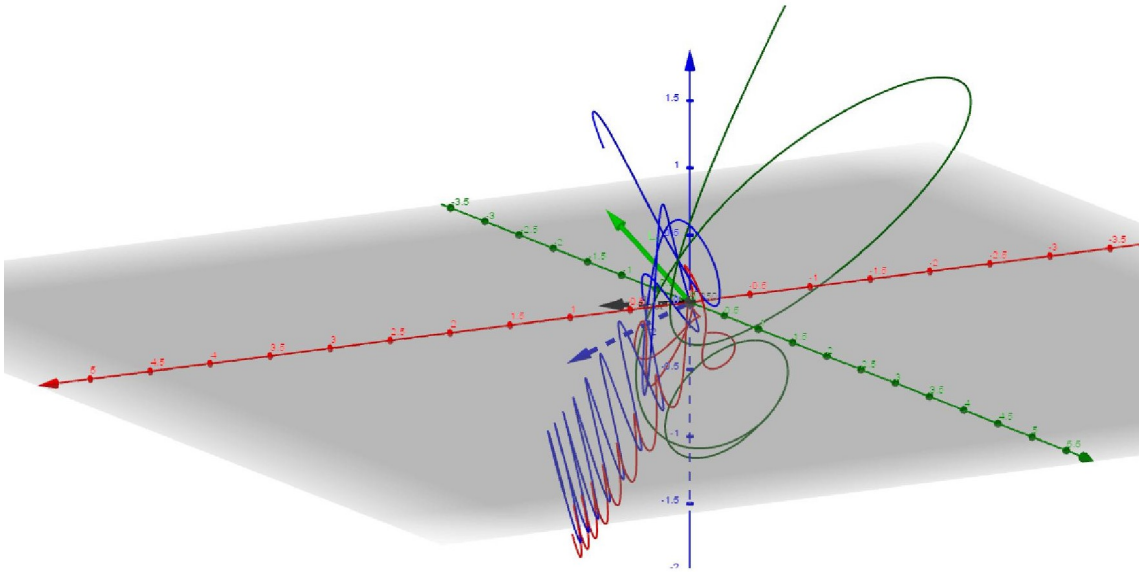


(b)

Figura 4.1: Rappresentazioni 3D (da Geogebra) dei risultati di simulazioni C++ del sistema Sole-pianeta mostrato nelle Fig. 2.22(a) e 2.23, ($m_1 = 10000.0, m_2 = 4000, T_i = 0.0, T_f = 0.5, Nt = 2000, x_{01} = 0.0, y_{01} = 0.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 0.0, x_{02} = 2.0, y_{02} = 2.0, z_{02} = 2.0, v_{0x2} = 8.0, v_{0y2} = -16, v_{0z2} = 12$), qualora interagisca con un terzo corpo con $x_{03} = -1.0, y_{03} = -1.0, z_{03} = -1.0, v_{0x3} = 28, v_{0y3} = -20, v_{0z3} = -30$ e massa m_3=10, in un riferimento inerziale (4.1(a)) o nel CdM (4.1(b)). Le unità sono tali che la costante di gravitazione universale sia $G = 1$. I vettori sono i versori dei momenti angolari *iniziali*: blu, corpo 2; verde corpo 3; nero, totale.



(a)



(b)

Figura 4.2: Rappresentazioni 3D (da Geogebra) dei risultati di simulazioni C++ del sistema Sole-pianeta mostrato nelle Fig. 2.22(a) e 2.23, ($m_1 = 10000.0, m_2 = 4000, T_i = 0.0, T_f = 0.5, Nt = 2000, x_{01} = 0.0, y_{01} = 0.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 0.0, x_{02} = 2.0, y_{02} = 2.0, z_{02} = 2.0, v_{0x2} = 8.0, v_{0y2} = -16, v_{0z2} = 12$), qualora interagisca con un terzo corpo con $x_{03} = -1.0, y_{03} = -1.0, z_{03} = -1.0, v_{0x3} = 28, v_{0y3} = -20, v_{0z3} = -30$ e massa m_3=2000, in un riferimento inerziale (4.2(a)) o nel CdM (4.2(b)). Le unità sono tali che la costante di gravitazione universale sia $G = 1$. I vettori sono i versori dei momenti angolari *iniziali*: blu, corpo 2; verde corpo 3; nero, totale.

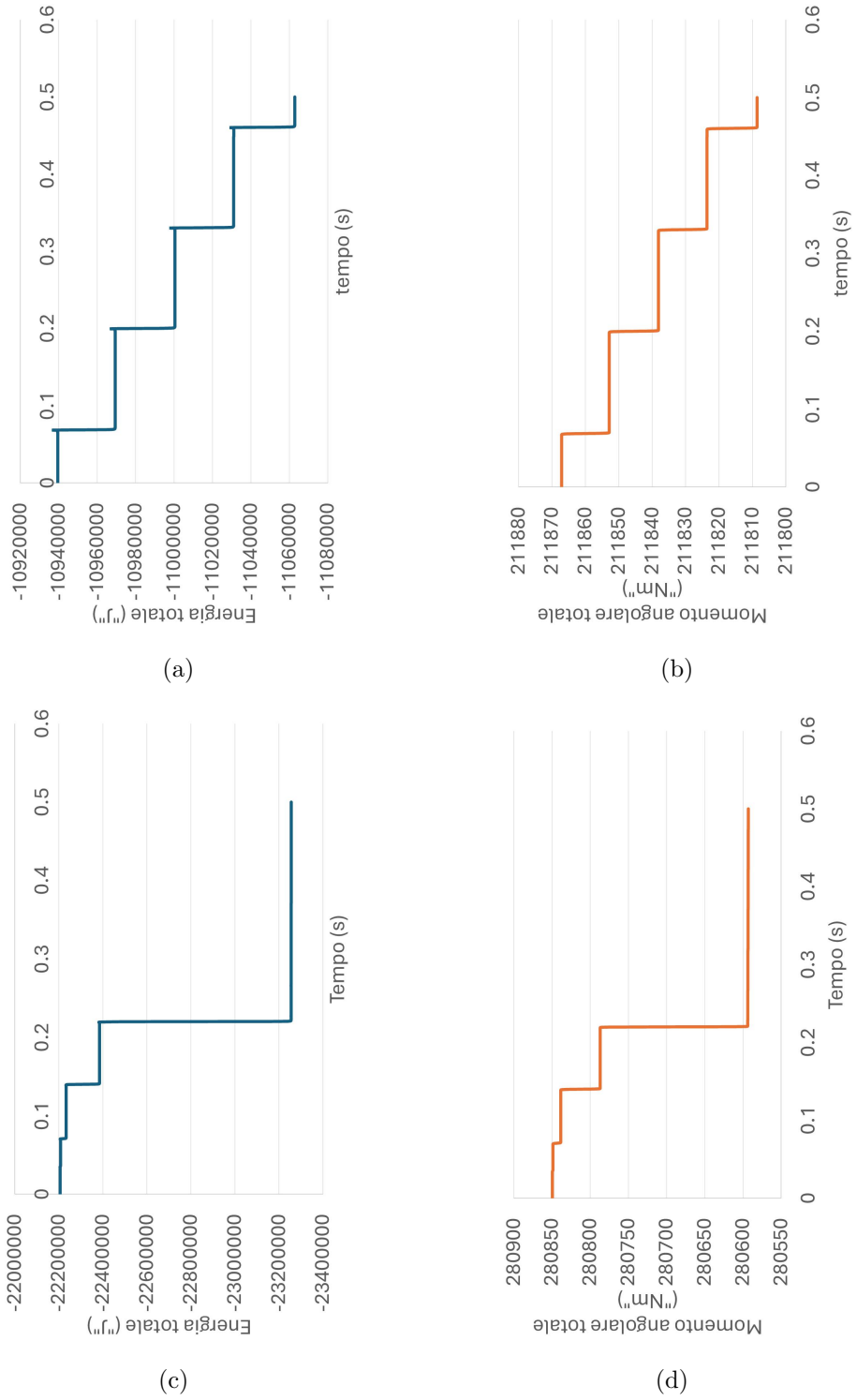


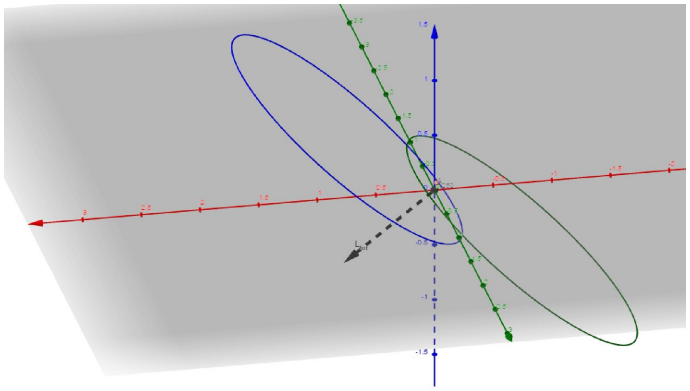
Figura 4.3: Andamenti nel tempo di energia totale e momento angolare totale (in unità di misura nominali indicate fra virgolette, in quanto scalate di costanti in modo tale che $G = 1$) nelle simulazioni C++ del sistema Sole-pianeta nelle Figg. 4.1 e 4.2: per il sistema di Fig. 4.1(b), Energia totale nel CdM, 4.3(a), e Momento angolare totale nel CdM, 4.3(b); per il sistema di Fig. 4.2(b), Energia totale nel CdM, 4.3(c), e Momento angolare totale nel CdM, 4.3(d).

riflessione sul senso della simulazione come sorta di laboratorio gravitazionale, con un suo insieme di *pratiche finalizzate ad acquisire consapevolezza* di ciò che si sta facendo, al di là della sua virtuale bellezza grafica. Per inciso infatti, *ogni simulazione è oggettivamente falsa*: si può solo dire se si ritengono accettabili o meno le approssimazioni a cui porta (Fig. 2.13). Ma *se per il problema dei 2 corpi ciò è possibile, in quanto si dispone di una soluzione matematicamente esatta per valutarle, nel caso del problema a 3 corpi il problema si riapre completamente*: sebbene appaia plausibile, nell'incontro dei tre pianeti *non accadrebbe quanto rappresentato*, perché si è avuta un'approssimazione tale di energia e momento angolare da compromettere tutto il seguito.

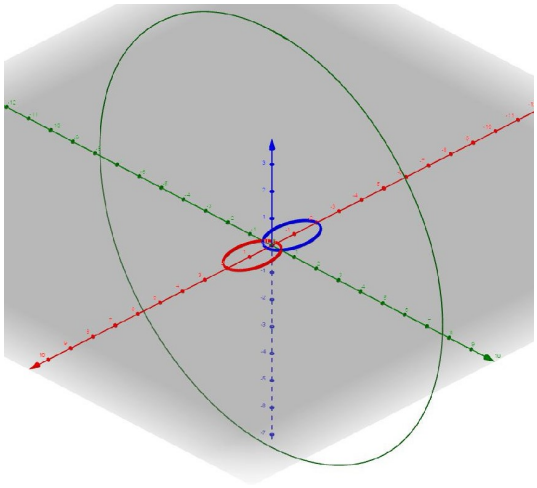
Dato quanto detto, appare quindi essenziale didatticamente, perché immediata, la possibilità di controllare l'affidabilità della simulazione da proprietà geometriche determinate dai principi di conservazione, primo fra tutti quello del momento angolare. In Fig. 4.4(a), alla coppia di pianeti di Figg. 2.22(a) e 2.23 viene ancora aggiunto un terzo corpo, ma con $m_3 = m_2$ e posizione e velocità opposte al corpo 2, *così che 2 e 3 abbiano lo stesso momento angolare, ed il CdM dell'intero sistema sia fermo nell'origine*. Come si comprende intuitivamente, si formano due orbite ellittiche sul piano ortogonale al momento angolare totale, che resta costante. Pur senza controllare energia e momento, la simulazione è stabile e affidabile sull'intervallo di tempo scelto (0.5s suddivisi in 2000 passi): solo ingrandendo le orbite si vede l'inizio del propagarsi di errori numerici.

Alla luce di tale risultato, nelle Figg. 4.4(b), 4.4(c) si simula nel riferimento del CdM un sistema analogo a quello di Fig. 1, osservato sperimentalmente: due stelle di stessa massa sono fatte partire su uno stesso piano con posizioni e velocità iniziali opposte rispetto all'origine, mentre un terzo corpo di massa 100 volte minore viene fatto partire con posizione e velocità tali da ruotare stabilmente su un piano ortogonale all'asse individuato dalle posizioni iniziali degli altri due, passante per l'origine. Il sistema risulta periodico nel tempo, ma molto più complesso di quanto si immagini: l'ingrandimento di Fig. 4.4(c) mostra come il terzo corpo sia comunque in grado di perturbare le orbite dei corpi 1 e 2, che restano piane, ma non ortogonali al momento angolare totale. Variando la massa m_3 a parità di tempo e passo di integrazione, si può modulare proprio tale effetto, ben simulato dal codice. Per m_3 decrescente, le orbite dei pianeti 1 e 2 sono meglio definite, ellittiche, ortogonali al momento angolare totale ed indipendenti dall'orbita del 3, che resta nel suo piano (pur perturbata), anche per $m_3 = 10^{-11} m_1$. Nelle Figg. 4.6(a), 4.6(b) si riportano le energie di uno dei due corpi principali e del terzo nel tempo, molto regolari e senza bruschi cambiamenti.

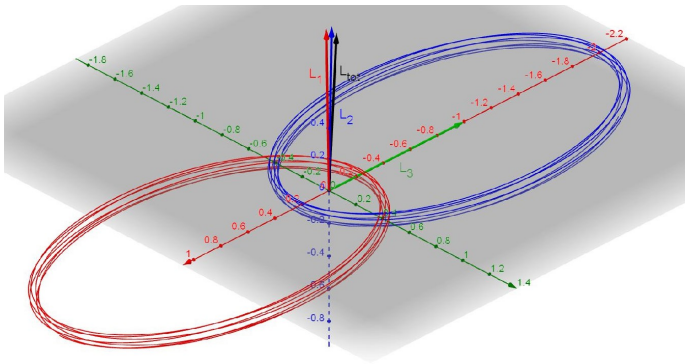
Per m_3 crescente, si perde invece gradualmente l'affidabilità della simulazione: anche per $m_3 = 4000$ si osservano ancora le periodicità, ma l'influenza del corpo 3 sugli altri porta ad orbite ben più complesse di quelle di Fig. 4.4(c), e da $m_3 = 6000$ in poi non bastano criteri geometrici: lo studio dell'energia e del momento totali (non riportati) mostra come un incontro fra i corpi porti ancora sistematicamente ad un



(a)



(b)

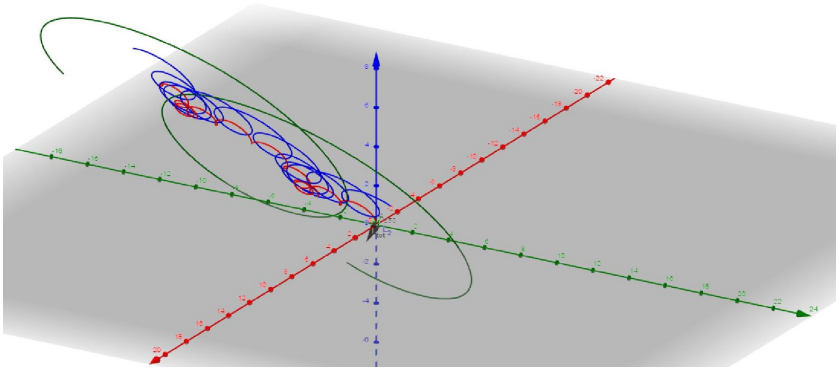


(c)

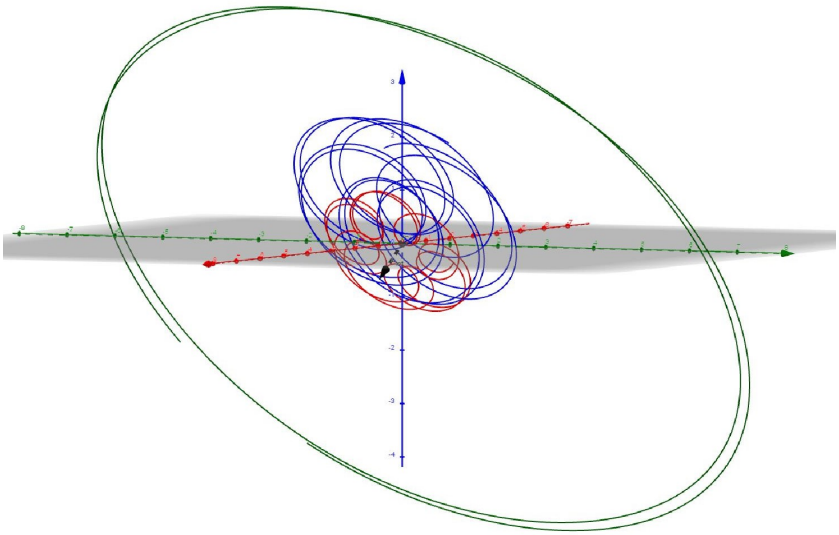
Figura 4.4: Rappresentazioni 3D (da Geogebra) dei risultati di simulazioni C++ di sistemi a 3 corpi. In Fig. 4.4(a), al sistema delle Figg. 2.22(a), 2.23 ($m_1 = 10000.0, m_2 = 4000, T_i = 0.0, T_f = 0.5, Nt = 2000, x_{01} = 0.0, y_{01} = 0.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 0.0, x_{02} = 2.0, y_{02} = 2.0, z_{02} = 2.0, v_{0x2} = 8.0, v_{0y2} = -16, v_{0z2} = 12$) viene aggiunto un terzo corpo con $m_3 = m_2$, posizione e velocità opposte al corpo 2, così che il CdM sia fermo; in Fig. 4.4(b) si simula nel riferimento del CdM un sistema analogo a quello di Fig. 1: due stelle con $m_1 = 10000 = m_2$ sono lanciate da $x_0 = \pm 2$ con velocità $v_{0y} = \pm 16$, mentre un terzo corpo di massa $m_3 = 100$ parte da $y_{03} = 8$ con $v_{0y3} = 48$; in Fig. 4.4(c) si ingrandiscono le orbite dei corpi 1 e 2 ed i versori dei momenti angolari *iniziali*: rosso, corpo 1; blu, 2; verde 3; nero, totale. Le unità sono tali che la costante di gravitazione universale sia $G = 1$.

"'salto"' brusco, dunque inaccettabile, fra due "'fasi"' in cui le grandezze conservate variano invece accettabilmente. Ovviamente, si possono modificare il tempo ed il passo di integrazione, ottenendo, con 4000 passi su 0.2s, comportamenti periodici ben definiti perfino per $m_3 = m_1$, ma ciò non risolve il problema del salto: i diversi criteri fisici di affidabilità, tradotti magari in proprietà geometriche, mostrano che la simulazione non è affidabile sebbene "'unisca due mondi possibili"'.

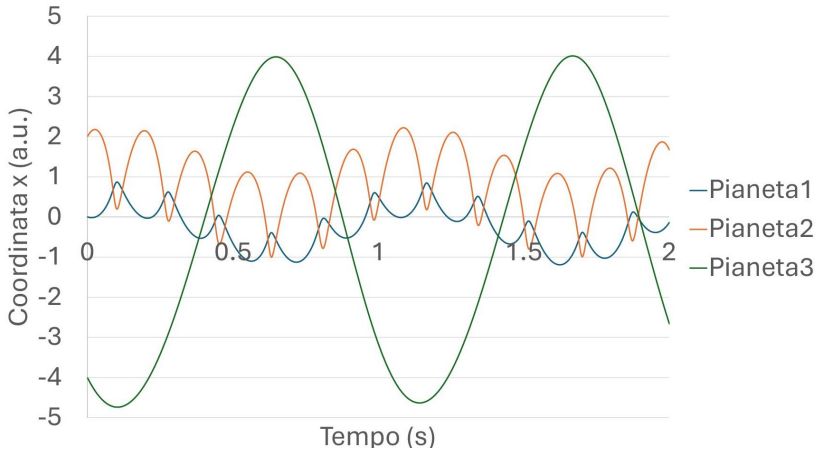
È pertanto ancora più significativo l'esempio riportato nelle Figg.4.5, dove il corpo 1 è fermo nell'origine mentre *masse, posizioni e velocità iniziali dei corpi 2 e 3 sono scelte in modo che abbiano all'inizio diversa quantità di moto ma stesso vettore momento angolare*. In tal modo il CdM del sistema deve muoversi, ma *le tre orbite devono restare su uno stesso piano ortogonale al momento angolare totale iniziale*, come in effetti si riscontra (Fig. 4.5(a)). Inoltre, il moto risulta periodico, con orbite molto complesse per i corpi 1 e 2, come si vede sia traslando il CdM dell'intero sistema nell'origine (Fig. 4.5(b)), sia studiando nel tempo le coordinate delle singole parti (4.5(c)). Dato il permanere del sistema sul piano ortogonale al momento angolare totale iniziale, si può affermare che gli effetti delle approssimazioni numeriche sono ancora contenuti, sebbene si osservino soprattutto sull'orbita del corpo 3. A conferma di quanto sia complesso ma anche molto interessante valutare tali "'esperimenti gravitazionali"', il controllo dell'energia e del momento totali nelle Figg. 4.6(c), 4.6(d) mostra come in questo caso un piccolo "'salto"' nelle grandezze sia presente, ma nel quadro di una variazione periodica dell'errore, e quindi ancora considerabile come un errore sistematico, più rilevante di quello in Fig. 4.3(a), ma meno di quello in Fig. 4.3(c).



(a)



(b)



(c)

Figura 4.5: Rappresentazioni 3D (da Geogebra) dei risultati di simulazioni C++ di sistemi a 3 corpi. In Fig. 4.5(a), nel riferimento inerziale si scelgono $m_1 = 10000.0, m_2 = 4000, m_3 = 2000, T_i = 0.0, T_f = 0.5, Nt = 2000, x_{01} = 0.0, y_{01} = 0.0, z_{01} = 0.0, v_{0x1} = 0.0, v_{0y1} = 0.0, v_{0z1} = 0.0, x_{02} = 2.0, y_{02} = 2.0, z_{02} = 2.0, v_{0x2} = 16.0, v_{0y2} = -32, v_{0z2} = 24, x_{03} = -4.0, y_{03} = -4.0, z_{03} = -4.0, v_{0x3} = -12, v_{0y3} = 24, v_{0z3} = -18$; in Fig. 4.5(b) si mostra lo stesso moto nel riferimento del CdM e in Fig. 4.5(c) le posizioni x dei tre corpi nel tempo. Le unità sono tali che la costante di gravitazione universale diventi $G = 1$. I vettori sono i versori dei momenti angolari *iniziali*: blu, corpo 2; verde corpo 3; nero, totale.

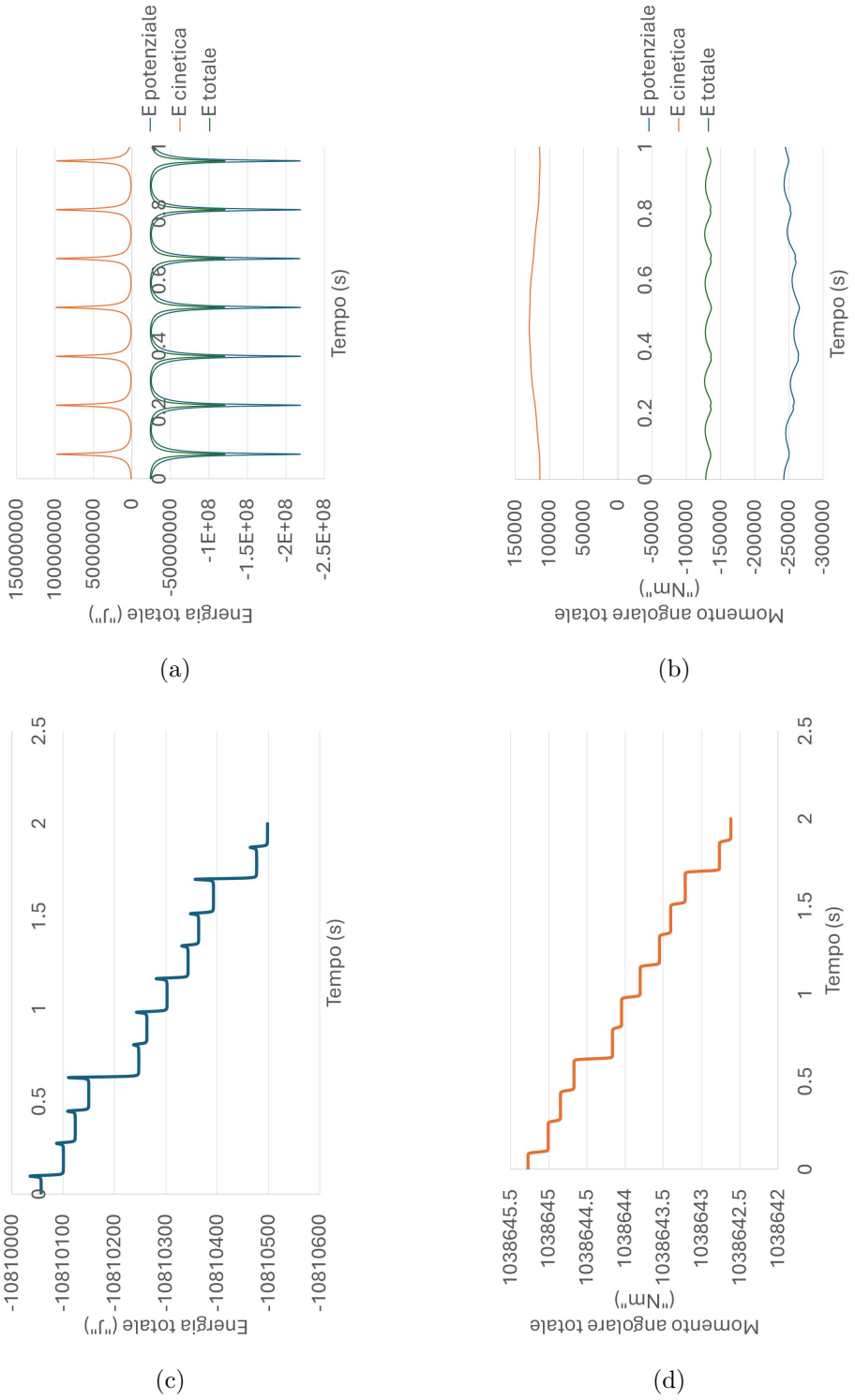


Figura 4.6: Andamenti nel tempo di energia totale e momento angolare totale (in unità di misura nominali indicate fra virgolette, in quanto scalate di costanti in modo tale che $G = 1$) nelle simulazioni `C++` dei sistemi nelle Figg. 4.4 4.5: per il sistema di Fig. 4.4(c), Energia totale nel CdM per uno dei due corpi principali, 4.6(a), e del terzo corpo, 4.6(b); per il sistema di Fig. 4.5(b), Energia totale nel CdM, 4.6(c), e Momento angolare totale nel CdM, 4.6(d).

Appendice A

Listati dei Codici C++

A.1 Un Sole fermo nell'origine

```
# include <iostream >
# include <fstream >
# include <cstdlib>
# include <string>
# include <cmath>
using namespace std;

const int P = 110000;
double G, M, m;     // Anche se introdotti da file, vanno dichiararli prima, o nella funzione accelerazione!
double T[P], X[P], Y[P], Z[P], Vx[P], Vy[P], Vz[P], Ep[P], Ec[P], Et[P], Lx[P], Ly[P], Lz[P], Lt[P];   // vettori tempo, posizioni, velocita', energie, momento
double ax, ay, az, Epot, Ecin, Lax, Lay, Laz;      // accelerazioni, energie, momento angolare

// procedure di integrazione: tutte sono funzione della stessa "parentesi" di variabili/parametri
void euler(const double& Ti, const double& Tf, const double& X0, const double& Vx0, const double& Y0, const double& Vy0, const double& Z0, const double& Vz0, const int& Nt);
void euler_cromer(const double& Ti, const double& Tf, const double& X0, const double& Vx0, const double& Y0, const double& Vy0, const double& Z0, const double& Vz0, const int& Nt);
void velocity_verlet(const double& Ti, const double& Tf, const double& X0, const double& Vx0, const double& Y0, const double& Vy0, const double& Z0, const double& Vz0, const int& Nt);
void RK(const double& Ti, const double& Tf, const double& X0, const double& Vx0, const double& Y0, const double& Vy0, const double& Z0, const double& Vz0, const int& Nt);
void RKSTEP(double& t, double& x, double& vx, double& y, double& vy, double& z, double& vz, const double& dt);

// funzioni accelerazione, velocita', energia potenziale e cinetica, momento angolare
double acc(double& x, double& y, double& z, double& vx, double& vy, double& vz, double& ax, double& ay, double& az, double& t);
double vel(double& x, double& y, double& z, double& vx, double& vy, double& vz, double& t);
double En(double& x, double& y, double& z, double& vx, double& vy, double& vz, double& Ecin, double& Epot);
double La(double& x, double& y, double& z , double& vx, double& vy, double& vz, double& Lax, double& Lay, double& Laz);
```

```
25
26 // PROGRAMMA PRINCIPALE
27 int main()
28 {
29 double Ti, Tf, X0, Y0, Z0, Vx0, Vy0, Vz0;   // parametri di integrazione: tempo,
       posizioni e velocita' iniziali, passi
30 int Nt, i;
31   ifstream ifile("input.txt");
32     while (!ifile.eof()) {
33             ifile >> G >> M >> m >> Ti >> Tf >> Nt >> X0 >> Y0 >> Z0 >> Vx0 >> Vy0
       >> Vz0;}
34     ifile.close();
35     if(Nt>=P) {cerr << "Error : Nt>=P\n"; exit(1);}   // Nt deve essere minore
       della dimensione P
36     // output delle caratteristiche del sistema
37     cout << "Massa Sole del sistema: M=" << M << "\n";
38     cout << "Massa Terra del sistema m=" << m << "\n";
39     cout << "Posizione Sole (fissa): x=" << 0 << " y=" << 0 << "\n";
40   cout << "Posizione iniziale Terra: x=" << X0 << " y=" << Y0 << " z=" << Z0 << "\n
       ";
41     cout << "Velocita' iniziale Terra: vx=" << Vx0 << " vy=" << Vy0 << " vz=" <<
       Vz0 << "\n";
42   cout << "Tempo di simulazione: " << Ti << "-" << Tf << " e numero passi: " << Nt
       << "\n";
43
44 // INTEGRAZIONI CON FUNZIONI DIVERSE, con output in estensione .xls, o .txt per il
       RK4
45
46 RK(Ti, Tf, X0, Vx0, Y0, Vy0, Z0, Vz0, Nt);            // Runge-Kutta 4
47 ofstream myfile4 ("rk4.xls");
48 myfile4.precision(17);
49 for(i=0; i<Nt; i++)
50 myfile4<<T[i]<<"\t"<<X[i]<<"\t"<<Y[i]<<"\t"<<Z[i]<<"\t"<<Vx[i]<<"\t"<<Vy[i]<<"\t"<<
       Vz[i]<<"\t"<<Ec[i]<<"\t"<<Ep[i]<<"\t"<<Et[i]<<"\t"<<Lx[i]<<"\t"<<Ly[i]<<"\t"<<
       Lz[i]<<"\t"<<Lt[i]<<endl;
51 myfile4.close();
52 ofstream myfile5 ("rk4.txt");
53 myfile5.precision(17);
54 for(i=0; i<Nt; i++)
55 myfile5<<T[i]<<"\t"<<X[i]<<"\t"<<Y[i]<<"\t"<<Z[i]<<"\t"<<Vx[i]<<"\t"<<Vy[i]<<"\t"<<
       Vz[i]<<"\t"<<Ec[i]<<"\t"<<Ep[i]<<"\t"<<Et[i]<<"\t"<<Lx[i]<<"\t"<<Ly[i]<<"\t"<<
       Lz[i]<<"\t"<<Lt[i]<<endl;
56 myfile5.close();
57
58 euler(Ti, Tf, X0, Vx0, Y0, Vy0, Z0, Vz0, Nt);         // Eulero
59 ofstream myfile1 ("euler.xls");
60 myfile1.precision(17);
61 for(i=0; i<Nt; i++)
62 myfile1<<T[i]<<"\t"<<X[i]<<"\t"<<Y[i]<<"\t"<<Z[i]<<"\t"<<Vx[i]<<"\t"<<Vy[i]<<"\t"<<
       Vz[i]<<"\t"<<Ec[i]<<"\t"<<Ep[i]<<"\t"<<Et[i]<<"\t"<<Lx[i]<<"\t"<<Ly[i]<<"\t"<<
       Lz[i]<<"\t"<<Lt[i]<<endl;
63 myfile1.close();
64
65 euler_cromer(Ti, Tf, X0, Vx0, Y0, Vy0, Z0, Vz0, Nt);   // Eulero Cromer
66 ofstream myfile2 ("cromer.xls");
67 myfile2.precision(17);
68 for(i=0; i<Nt; i++)
69 myfile2<<T[i]<<"\t"<<X[i]<<"\t"<<Y[i]<<"\t"<<Z[i]<<"\t"<<Vx[i]<<"\t"<<Vy[i]<<"\t"<<
       Vz[i]<<"\t"<<Ec[i]<<"\t"<<Ep[i]<<"\t"<<Et[i]<<"\t"<<Lx[i]<<"\t"<<Ly[i]<<"\t"<<
       Lz[i]<<"\t"<<Lt[i]<<endl;
70 myfile2.close();
71
72 velocity_verlet(Ti, Tf, X0, Vx0, Y0, Vy0, Z0, Vz0, Nt);     // Velocity Verlet
73 ofstream myfile3 ("velocity_verlet.xls");
```

```
74 myfile3.precision(17);
75 for(i=0; i<Nt; i++)
76 myfile3<<T[i]<<"\t"<<X[i]<<"\t"<<Y[i]<<"\t"<<Z[i]<<"\t"<<Vx[i]<<"\t"<<Vy[i]<<"\t"<<
       Vz[i]<<"\t"<<Ec[i]<<"\t"<<Ep[i]<<"\t"<<Et[i]<<"\t"<<Lx[i]<<"\t"<<Ly[i]<<"\t"<<
       Lz[i]<<"\t"<<Lt[i]<<endl;
77 myfile3.close();
78 }
79
80 // funzioni
81
82 double acc(double& x, double& y, double& z , double& vx, double& vy, double& vz,
       double& ax, double& ay, double& az, double& t)
83 {
84 double r;
85 r=sqrt(x*x+y*y+z*z);
86 ax=-G*M*x/(r*r*r);
87 ay=-G*M*y/(r*r*r);
88 az=-G*M*z/(r*r*r);
89 }
90
91 double vel(double& x, double& y, double& z , double& vx, double& vy, double& vz,
       double& t)
92 {
93 vx=vx;
94 vy=vy;
95 vz=vz;
96 }
97
98 double En(double& x, double& y, double& z, double& vx, double& vy, double& vz,
       double& Ecin, double& Epot)  // energia
99 {
100 double r, vsqr;
101 r=sqrt(x*x+y*y+z*z);
102 vsqr=vx*vx+vy*vy+vz*vz;
103 Epot=-G*M*m/r;
104 Ecin=0.5*m*vsqr;
105 }
106
107 double La(double& x, double& y, double& z , double& vx, double& vy, double& vz,
       double& Lax, double& Lay, double& Laz)   // momento angolare
108 {
109 Lax=m*(y*vz-z*vy);
110 Lay=m*(z*vx-x*vz);
111 Laz=m*(x*vy-y*vx);
112 }
113
114 // integratori
115
116 void euler(const double& Ti, const double& Tf, const double& X0, const double& Vx0,
        const double& Y0, const double& Vy0, const double& Z0, const double& Vz0,
       const int& Nt)
117 {
118 int i;
119 double h;
120 T[0]=Ti;
121 Vx[0]=Vx0;
122 Vy[0]=Vy0;
123 Vz[0]=Vz0;
124 X[0]=X0;
125 Y[0]=Y0;
126 Z[0]=Z0;
127 acc(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], ax, ay, az, T[0]);
128 h=Tf/(Nt-1);
129 for(i=1; i<Nt; i++)
```

```
{
T[i]=T[i-1]+h;
Vx[i]=Vx[i-1]+ax*h;
Vy[i]=Vy[i-1]+ay*h;
Vz[i]=Vz[i-1]+az*h;
X[i]=X[i-1]+Vx[i-1]*h;
Y[i]=Y[i-1]+Vy[i-1]*h;
Z[i]=Z[i-1]+Vz[i-1]*h;
acc(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], ax, ay, az, T[i]);
}
}

void euler_cromer(const double& Ti, const double& Tf, const double& X0, const
    double& Vx0, const double& Y0, const double& Vy0, const double& Z0, const
    double& Vz0, const int& Nt)
{
int i;
double h;
T[0]=Ti;
Vx[0]=Vx0;
Vy[0]=Vy0;
Vz[0]=Vz0;
X[0]=X0;
Y[0]=Y0;
Z[0]=Z0;
acc(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], ax, ay, az, T[0]);
h=Tf/(Nt-1);
for (i=1; i<Nt; i++){
T[i]=T[i-1]+h;
Vx[i]=Vx[i-1]+ax*h;
Vy[i]=Vy[i-1]+ay*h;
Vz[i]=Vz[i-1]+az*h;
X[i]=X[i-1]+Vx[i]*h;
Y[i]=Y[i-1]+Vy[i]*h;
Z[i]=Z[i-1]+Vz[i]*h;
acc(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], ax, ay, az, T[i]);
}
}

void velocity_verlet(const double& Ti, const double& Tf, const double& X0, const
    double& Vx0, const double& Y0, const double& Vy0, const double& Z0, const
    double& Vz0, const int& Nt)
{
int i;
double h, ax1, ay1, az1;
T[0]=Ti;
Vx[0]=Vx0;
Vy[0]=Vy0;
Vz[0]=Vz0;
X[0]=X0;
Y[0]=Y0;
Z[0]=Z0;
acc(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], ax, ay, az, T[0]);
h=Tf/(Nt-1);
for(i=1; i<Nt; i++)
{
T[i]=T[i-1]+h;
X[i]=X[i-1]+Vx[i-1]*h+0.5*ax*h*h;
Y[i]=Y[i-1]+Vy[i-1]*h+0.5*ay*h*h;
Z[i]=Z[i-1]+Vz[i-1]*h+0.5*az*h*h;
ax1=ax;
ay1=ay;
az1=az;
acc(X[i], Y[i], Z[i], Vx[i-1], Vy[i-1], Vz[i-1], ax, ay, az, T[i-1]);
```

```
Vx[i]=Vx[i-1]+0.5*(ax1+ax)*h;
Vy[i]=Vy[i-1]+0.5*(ay1+ay)*h;
Vz[i]=Vz[i-1]+0.5*(az1+az)*h;
}
}


void RK(const double& Ti, const double& Tf, const double& X0, const double& Vx0,
    const double& Y0, const double& Vy0, const double& Z0, const double& Vz0, const
     int& Nt)
{
double dt;
double TS, XS, VxS, YS, VyS, ZS, VzS;
int i;
dt =(Tf-Ti)/(Nt-1);
T[0]=Ti;
X[0]=X0;
Vx[0]=Vx0;
Y[0]=Y0;
Vy[0]=Vy0;
Z[0]=Z0;
Vz[0]=Vz0;

En(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], Ecin, Epot);      // calcolo energia
Ep[0]=Epot;
Ec[0]=Ecin;
Et[0]=Epot+Ecin;
La(X[0], Y[0], Z[0], Vx[0], Vy[0], Vz[0], Lax, Lay, Laz);  // calcolo momento
    angolare
Lx[0]=Lax;
Ly[0]=Lay;
Lz[0]=Laz;
Lt[0]=sqrt(Lax*Lax+Lay*Lay+Laz*Laz);

TS=Ti;

XS=X0;
VxS=Vx0;
YS=Y0;
VyS=Vy0;
ZS=Z0;
VzS=Vz0;

for(i=1;i<Nt;i++)
  {
  RKSTEP(TS,XS,VxS,YS,VyS,ZS,VzS,dt);
  T[i]=TS;
  X[i]=XS;
  Vx[i]=VxS;
  Y[i]=YS;
  Vy[i]=VyS;
    Z[i]=ZS;
  Vz[i]=VzS;

  En(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], Ecin, Epot);
  Ep[i]=Epot;
  Ec[i]=Ecin;
  Et[i]=Epot+Ecin;
  La(X[i], Y[i], Z[i], Vx[i], Vy[i], Vz[i], Lax, Lay, Laz);
  Lx[i]=Lax;
  Ly[i]=Lay;
  Lz[i]=Laz;
  Lt[i]=sqrt(Lax*Lax+Lay*Lay+Laz*Laz);
  }
```

```
}

void RKSTEP(double& t, double& x, double& vx, double& y, double& vy, double& z,
    double& vz, const double& dt)
{
  double xf, yf, zf, vxf, vyf, vzf, tf;
  double k11x, k12x, k13x, k14x, k21x, k22x, k23x, k24x;
  double k11y, k12y,k13y, k14y, k21y, k22y, k23y, k24y;
    double k11z, k12z, k13z, k14z, k21z, k22z, k23z, k24z;

  double h, h2, h6;

  h=dt;
  h2=0.5*h;
  h6=h/6.0;

    xf=x;
    yf=y;
    zf=z;
    vxf=vx;
    vyf=vy;
    vzf=vz;
  tf=t;

  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
    vel(x, y, z, vx, vy, vz, t);
    t=tf+h2;
  k21x=ax;
  k11x=vx;
  vx=vxf+h2*k21x;
  x=xf+h2*k11x;
  k21y=ay;
    k11y=vy;
    vy=vyf+h2*k21y;
  y=yf+h2*k11y;
  k21z=az;
  k11z=vz;
  vz=vzf+h2*k21z;
    z=zf+h2*k11z;

  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
    vel(x, y, z, vx, vy, vz, t);
  t=t;           // per ricordare che resta t=tf+h2;
  k22x=ax;
  k12x=vx;
  k22y=ay;
  k12y=vy;
  k22z=az;
  k12z=vz;
  vx=vxf+h2*k22x;
  x=xf+h2*k12x;
  vy=vyf+h2*k22y;
  y=yf+h2*k12y;
    vz=vzf+h2*k22z;
  z=zf+h2*k12z;

  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
    vel(x, y, z, vx, vy, vz, t);
    t=tf+h;
  k23x=ax;
  k13x=vx;
  k23y=ay;
  k13y=vy;
  k23z=az;
```

```
  k13z=vz;
  x=xf+h*k13x;
  y=yf+h*k13y;
  z=zf+h*k13z;
  vx=vxf+h*k23x;
  vy=vyf+h*k23y;
  vz=vzf+h*k23z;

  acc(x, y, z, vx, vy, vz, ax, ay, az, t);
    vel(x, y, z, vx, vy, vz, t);
    t=t;        // per ricordare che resta t=tf+h;
  k14x=vx;
  k24x=ax;
  k14y=vy;
  k24y=ay;
    k14z=vz;
  k24z=az;

  x=xf+h6*(k11x+2.0*(k12x+k13x)+k14x);
  vx=vxf+h6*(k21x+2.0*(k22x+k23x)+k24x);
  y=yf+h6*(k11y+2.0*(k12y+k13y)+k14y);
  vy=vyf+h6*(k21y+2.0*(k22y+k23y)+k24y);
  z=zf+h6*(k11z+2.0*(k12z+k13z)+k14z);
  vz=vzf+h6*(k21z+2.0*(k22z+k23z)+k24z);

}
```

A.2 Due stelle mobili

A.2.1 Riferimento inerziale

```
// Problema di Kepler in un sistema inerziale

# include <iostream >
# include <fstream >
# include <cstdlib>
# include <string>
# include <cmath>
using namespace std;

const int P = 210000;
double G, m1, m2;      // anche se introdotti da file
double T[P];           // vettore tempo
double X1[P], Y1[P], Z1[P], Vx1[P], Vy1[P], Vz1[P], Ec1[P], Et1[P], Lx1[P], Ly1[P],
     Lz1[P], Lt1[P];   // vettori tempo, posizioni, velocita', energie, momento,
    pianeta 1
double X2[P], Y2[P], Z2[P], Vx2[P], Vy2[P], Vz2[P], Ec2[P], Et2[P], Lx2[P], Ly2[P],
     Lz2[P], Lt2[P];   // vettori tempo, posizioni, velocita', energie, momento,
    pianeta 2
double Ept[P], ET[P], LT[P]; // energia potenziale comune, energia e momento
    angolare totali del sistema
double ax1, ay1, az1, Epot, Ecin1, Lax1, Lay1, Laz1;      // accelerazioni, energie
    , momento angolare pianeta 1
double ax2, ay2, az2, Ecin2, Lax2, Lay2, Laz2;      // accelerazioni, energie,
    momento angolare pianeta 2
double v1circ, v2circ;    // velocita' per moto circolare

// procedura di integrazione
void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
    const double& Y01, const double& Vy01, const double& Z01, const double& Vz01,
    const double& X02, const double& Vx02, const double& Y02, const double& Vy02,
    const double& Z02, const double& Vz02, const int& Nt);
```

```
22 void RKSTEP(double& t, double& x1, double& vx1, double& y1, double& vy1, double& z1
       , double& vz1, double& x2, double& vx2, double& y2, double& vy2, double& z2,
       double& vz2, const double& dt);
23
24 // funzioni accelerazione, velocita', energia e momento
25 double acc(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
       double& ax1, double& ay1, double& az1, double& ax2, double& ay2, double& az2);
26 double vel(double& vx1, double& vy1, double& vz1, double& vx2, double& vy2, double&
        vz2);
27 double Ep(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
       double& Epot);
28 double Enc(double& m1, double& vx, double& vy, double& vz, double& Ecin);
29 double La(double& m1, double& x, double& y, double& z , double& vx, double& vy,
       double& vz, double& Lax, double& Lay, double& Laz);
30
31 int main()   // PROGRAMMA PRINCIPALE
32 {
33 double Ti, Tf, X01, Y01, Z01, Vx01, Vy01, Vz01, X02, Y02, Z02, Vx02, Vy02, Vz02;
         // parametri di integrazione: tempo, posizioni e velocita' iniziali, passi
34 int Nt, i;
35   ifstream ifile("input.txt");
36     while (!ifile.eof()) {
37             ifile >> G >> m1 >> m2 >> Ti >> Tf >> Nt >> X01 >> Y01 >> Z01 >> Vx01
       >> Vy01 >> Vz01 >> X02 >> Y02 >> Z02 >> Vx02 >> Vy02 >> Vz02;}
38     ifile.close();
39     if(Nt>=P) {cerr << "Error : Nt>=P\n"; exit(1);}  // deve essere Nnt<P
40
41   v1circ=m2*sqrt(G/((m1+m2)*sqrt((X01-X02)*(X01-X02)+(Y01-Y02)*(Y01-Y02)+(Z01-Z02)
       *(Z01-Z02))));
42   v2circ=-m1*v1circ/m2;
43   cout << "Per avere il centro di massa nell'origine, scegliere x1*m1=-x2*m2, y1*m1
       =-y2*m2, z1*m1=-z2*m2 \n";
44   cout << "Per orbite circolari dalle posizioni scelte, scegliere il modulo v1=" <<
        v1circ << " e v2=-m1*|v1|/m2=" << v2circ << "\n\n";
45   cout << "Massa Pianeta 1 del sistema: m1=" << m1 << "\n";          // output delle
        caratteristiche del sistema
46     cout << "Massa Pianeta 2 del sistema m2=" << m2 << "\n";
47     cout << "Posizione iniziale Pianeta 1: x=" << X01 << " y=" << Y01 << " z=" <<
       Z01 << "\n";
48     cout << "Velocita' iniziale Pianeta 1: vx=" << Vx01 << " vy=" << Vy01 << " vz="
        << Vz01 << "\n";
49   cout << "Posizione iniziale Pianeta 2: x=" << X02 << " y=" << Y02 << " z=" << Z02
        << "\n";
50     cout << "Velocita' iniziale Pianeta 2: vx=" << Vx02 << " vy=" << Vy02 << " vz="
        << Vz02 << "\n";
51   cout << "Tempo di simulazione: " << Ti << "-" << Tf << " e numero passi: " << Nt
       << "\n";
52
53 RK(Ti, Tf, X01, Vx01, Y01, Vy01, Z01, Vz01, X02, Vx02, Y02, Vy02, Z02, Vz02, Nt);
       // INTEGRAZIONE RK4, con output in estensione .xls
54 ofstream myfile1 ("rk43D2corpiXV.xls");
55 myfile1.precision(17);
56 for(i=0; i<Nt; i++)
57 myfile1<<T[i]<<"\t"<<X1[i]<<"\t"<<Y1[i]<<"\t"<<Z1[i]<<"\t"<<X2[i]<<"\t"<<Y2[i]<<"\t
       "<<Z2[i]<<"\t"<<Vx1[i]<<"\t"<<Vy1[i]<<"\t"<<Vz1[i]<<"\t"<<Vx2[i]<<"\t"<<Vy2[i
       ]<<"\t"<<Vz2[i]<<endl;
58 myfile1.close();
59 ofstream myfile2 ("rk43D2corpiEL.xls");
60 myfile2.precision(17);
61 for(i=0; i<Nt; i++)
62 myfile2<<T[i]<<"\t"<<Ept[i]<<"\t"<<Ec1[i]<<"\t"<<Et1[i]<<"\t"<<Ec2[i]<<"\t"<<Et2[i
       ]<<"\t"<<ET[i]<<"\t"<<Lx1[i]<<"\t"<<Ly1[i]<<"\t"<<Lz1[i]<<"\t"<<Lx2[i]<<"\t"<<
       Ly2[i]<<"\t"<<Lz2[i]<<"\t"<<LT[i]<<endl;
63 myfile2.close();
```

```
64 }
65
66 // Funzioni
67 double acc(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
       double& ax1, double& ay1, double& az1, double& ax2, double& ay2, double& az2)
68 {
69 double r12=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2)+(z1-z2)*(z1-z2));
70 double f12=-G*m1*m2/(r12*r12);
71 ax1=(f12*(x1-x2)/r12)/m1;
72 ax2=-ax1*m1/m2;
73 ay1=(f12*(y1-y2)/r12)/m1;
74 ay2=-ay1*m1/m2;
75 az1=(f12*(z1-z2)/r12)/m1;
76 az2=-az1*m1/m2;
77 }
78
79 double vel(double& vx1, double& vy1, double& vz1, double& vx2, double& vy2, double&
       vz2)
80 {
81 vx1=vx1;
82 vy1=vy1;
83 vz1=vz1;
84 vx2=vx2;
85 vy2=vy2;
86 vz2=vz2;
87 }
88
89 double Ep(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
       double& Epot)  // energia potenziale
90 {
91 double r;
92 r=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2)+(z1-z2)*(z1-z2));
93 Epot=-G*m1*m2/r;
94 }
95
96 double Enc(double& m1, double& vx, double& vy, double& vz, double& Ecin)  //
       energia cinetica
97 {
98 double v;
99 v=sqrt(vx*vx+vy*vy+vz*vz);
100 Ecin=0.5*m1*v*v;
101 }
102
103 double La(double& m1, double& x, double& y, double& z , double& vx, double& vy,
       double& vz, double& Lax, double& Lay, double& Laz)   // momento angolare
104 {
105 Lax=m1*(y*vz-z*vy);
106 Lay=m1*(z*vx-x*vz);
107 Laz=m1*(x*vy-y*vx);
108 }
109
110 // integratori
111
112 void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
       const double& Y01, const double& Vy01, const double& Z01, const double& Vz01,
       const double& X02, const double& Vx02, const double& Y02, const double& Vy02,
       const double& Z02, const double& Vz02, const int& Nt)
113 {
114 double dt;
115 double TS, XS1, VxS1, YS1, VyS1, ZS1, VzS1, XS2, VxS2, YS2, VyS2, ZS2, VzS2;
116 int i;
117 dt =(Tf-Ti)/(Nt-1);
118
119 T[0]=Ti;
```

```
120 X1[0]=X01;      // inizializzazione vettori pianeta 1
121 Vx1[0]=Vx01;
122 Y1[0]=Y01;
123 Vy1[0]=Vy01;
124 Z1[0]=Z01;
125 Vz1[0]=Vz01;
126
127 X2[0]=X02;      // inizializzazione vettori pianeta 2
128 Vx2[0]=Vx02;
129 Y2[0]=Y02;
130 Vy2[0]=Vy02;
131 Z2[0]=Z02;
132 Vz2[0]=Vz02;
133
134 Ep(X1[0], Y1[0], Z1[0], X2[0], Y2[0], Z2[0], Epot);  // energia pianeta 1
135 Ept[0]=Epot;
136
137 Enc(m1, Vx1[0], Vy1[0], Vz1[0], Ecin1);
138 Ec1[0]=Ecin1;
139 Et1[0]=Epot+Ecin1;
140 La(m1, X1[0], Y1[0], Z1[0], Vx1[0], Vy1[0], Vz1[0], Lax1, Lay1, Laz1);  // momento
        pianeta 1
141 Lx1[0]=Lax1;
142 Ly1[0]=Lay1;
143 Lz1[0]=Laz1;
144 Lt1[0]=sqrt(Lax1*Lax1+Lay1*Lay1+Laz1*Laz1);
145
146 Enc(m2, Vx2[0], Vy2[0], Vz2[0], Ecin2);  // energia pianeta 2
147 Ec2[0]=Ecin2;
148 Et2[0]=Epot+Ecin2;
149 La(m2, X2[0], Y2[0], Z2[0], Vx2[0], Vy2[0], Vz2[0], Lax2, Lay2, Laz2);  // momento
        pianeta 2
150 Lx2[0]=Lax2;
151 Ly2[0]=Lay2;
152 Lz2[0]=Laz2;
153 Lt2[0]=sqrt(Lax2*Lax2+Lay2*Lay2+Laz2*Laz2);
154
155 ET[0]=Epot+Ecin1+Ecin2;    // energia totale sistema
156 LT[0]=sqrt((Lax1+Lax2)*(Lax1+Lax2)+(Lay1+Lay2)*(Lay1+Lay2)+(Laz1+Laz2)*(Laz1+Laz2))
        ;    // momento angolare totale sistema
157
158 TS=Ti;
159
160 XS1=X01;    // inizializzazione variabili ciclo, pianeta 1
161 VxS1=Vx01;
162 YS1=Y01;
163 VyS1=Vy01;
164 ZS1=Z01;
165 VzS1=Vz01;
166
167 XS2=X02;    // inizializzazione variabili ciclo, pianeta 2
168 VxS2=Vx02;
169 YS2=Y02;
170 VyS2=Vy02;
171 ZS2=Z02;
172 VzS2=Vz02;
173
174 for(i=1;i<Nt;i++)   // ciclo
175   {
176   RKSTEP(TS,XS1,VxS1,YS1,VyS1,ZS1,VzS1,XS2,VxS2,YS2,VyS2,ZS2,VzS2,dt);   // passo
        del ciclo (integrazione)
177   T[i]=TS;
178   X1[i]=XS1;   // caricamento sui vettori di output, pianeta 1
179   Vx1[i]=VxS1;
```

```
180   Y1[i]=YS1;
181   Vy1[i]=VyS1;
182     Z1[i]=ZS1;
183   Vz1[i]=VzS1;
184
185     X2[i]=XS2;   // caricamento sui vettori di output, pianeta 2
186   Vx2[i]=VxS2;
187   Y2[i]=YS2;
188   Vy2[i]=VyS2;
189     Z2[i]=ZS2;
190   Vz2[i]=VzS2;
191
192   Ep(X1[i], Y1[i], Z1[i], X2[i], Y2[i], Z2[i], Epot);  // energia potenziale comune
193   Ept[i]=Epot;
194
195   Enc(m1, Vx1[i], Vy1[i], Vz1[i], Ecin1);
196   Ec1[i]=Ecin1;
197   Et1[i]=Epot+Ecin1;
198   La(m1, X1[i], Y1[i], Z1[i], Vx1[i], Vy1[i], Vz1[i], Lax1, Lay1, Laz1);  //
        momento pianeta 1
199   Lx1[i]=Lax1;
200   Ly1[i]=Lay1;
201   Lz1[i]=Laz1;
202   Lt1[i]=sqrt(Lax1*Lax1+Lay1*Lay1+Laz1*Laz1);
203
204   Enc(m2, Vx2[i], Vy2[i], Vz2[i], Ecin2);   // energia pianeta 2
205   Ec2[i]=Ecin2;
206   Et2[i]=Epot+Ecin2;
207   La(m2, X2[i], Y2[i], Z2[i], Vx2[i], Vy2[i], Vz2[i], Lax2, Lay2, Laz2);  //
        momento pianeta 2
208   Lx2[i]=Lax2;
209   Ly2[i]=Lay2;
210   Lz2[i]=Laz2;
211   Lt2[i]=sqrt(Lax2*Lax2+Lay2*Lay2+Laz2*Laz2);
212
213   ET[i]=Epot+Ecin1+Ecin2;    // energia totale sistema
214   LT[i]=sqrt((Lax1+Lax2)*(Lax1+Lax2)+(Lay1+Lay2)*(Lay1+Lay2)+(Laz1+Laz2)*(Laz1+Laz2
        ));    // momento angolare totale sistema
215   }
216 }
217
218 void RKSTEP(double& t, double& x1, double& vx1, double& y1, double& vy1, double& z1
      , double& vz1, double& x2, double& vx2, double& y2, double& vy2, double& z2,
      double& vz2, const double& dt)
219 {
220     double h, h2, h6, tf;
221   h=dt;
222   h2=0.5*h;
223   h6=h/6.0;
224   tf=t;
225
226   double xf1, yf1, zf1, vxf1, vyf1, vzf1;
227   double k11x1, k12x1, k13x1, k14x1, k21x1, k22x1, k23x1, k24x1;
228   double k11y1, k12y1, k13y1, k14y1, k21y1, k22y1, k23y1, k24y1;
229     double k11z1, k12z1, k13z1, k14z1, k21z1, k22z1, k23z1, k24z1;
230     xf1=x1;
231     yf1=y1;
232     zf1=z1;
233     vxf1=vx1;
234     vyf1=vy1;
235     vzf1=vz1;
236
237     double xf2, yf2, zf2, vxf2, vyf2, vzf2;
238   double k11x2, k12x2, k13x2, k14x2, k21x2, k22x2, k23x2, k24x2;
```

```
239   double k11y2, k12y2, k13y2, k14y2, k21y2, k22y2, k23y2, k24y2;
240     double k11z2, k12z2, k13z2, k14z2, k21z2, k22z2, k23z2, k24z2;
241     xf2=x2;
242     yf2=y2;
243     zf2=z2;
244     vxf2=vx2;
245     vyf2=vy2;
246     vzf2=vz2;
247
248     acc(x1, y1, z1, x2, y2, z2, ax1, ay1, az1, ax2, ay2, az2);
249     vel(vx1, vy1, vz1, vx2, vy2, vz2);
250     t=tf+h2;
251
252   k21x1=ax1;
253   k11x1=vx1;
254   vx1=vxf1+h2*k21x1;
255   x1=xf1+h2*k11x1;
256   k21y1=ay1;
257     k11y1=vy1;
258     vy1=vyf1+h2*k21y1;
259   y1=yf1+h2*k11y1;
260   k21z1=az1;
261   k11z1=vz1;
262   vz1=vzf1+h2*k21z1;
263     z1=zf1+h2*k11z1;
264
265     k21x2=ax2;
266   k11x2=vx2;
267   vx2=vxf2+h2*k21x2;
268   x2=xf2+h2*k11x2;
269   k21y2=ay2;
270     k11y2=vy2;
271     vy2=vyf2+h2*k21y2;
272   y2=yf2+h2*k11y2;
273   k21z2=az2;
274   k11z2=vz2;
275   vz2=vzf2+h2*k21z2;
276     z2=zf2+h2*k11z2;
277
278
279   acc(x1, y1, z1, x2, y2, z2, ax1, ay1, az1, ax2, ay2, az2);
280     vel(vx1, vy1, vz1, vx2, vy2, vz2);
281   t=t;         // per ricordare che resta t=tf+h2;
282
283   k22x1=ax1;
284   k12x1=vx1;
285   k22y1=ay1;
286   k12y1=vy1;
287   k22z1=az1;
288   k12z1=vz1;
289   vx1=vxf1+h2*k22x1;
290   x1=xf1+h2*k12x1;
291   vy1=vyf1+h2*k22y1;
292   y1=yf1+h2*k12y1;
293     vz1=vzf1+h2*k22z1;
294   z1=zf1+h2*k12z1;
295
296     k22x2=ax2;
297   k12x2=vx2;
298   k22y2=ay2;
299   k12y2=vy2;
300   k22z2=az2;
301   k12z2=vz2;
302   vx2=vxf2+h2*k22x2;
```

```
x2=xf2+h2*k12x2;
vy2=vyf2+h2*k22y2;
y2=yf2+h2*k12y2;
  vz2=vzf2+h2*k22z2;
z2=zf2+h2*k12z2;

acc(x1, y1, z1, x2, y2, z2, ax1, ay1, az1, ax2, ay2, az2);
  vel(vx1, vy1, vz1, vx2, vy2, vz2);
  t=tf+h;

k23x1=ax1;
k13x1=vx1;
k23y1=ay1;
k13y1=vy1;
k23z1=az1;
k13z1=vz1;
x1=xf1+h*k13x1;
y1=yf1+h*k13y1;
z1=zf1+h*k13z1;
vx1=vxf1+h*k23x1;
vy1=vyf1+h*k23y1;
vz1=vzf1+h*k23z1;

  k23x2=ax2;
k13x2=vx2;
k23y2=ay2;
k13y2=vy2;
k23z2=az2;
k13z2=vz2;
x2=xf2+h*k13x2;
y2=yf2+h*k13y2;
z2=zf2+h*k13z2;
vx2=vxf2+h*k23x2;
vy2=vyf2+h*k23y2;
vz2=vzf2+h*k23z2;

acc(x1, y1, z1, x2, y2, z2, ax1, ay1, az1, ax2, ay2, az2);
  vel(vx1, vy1, vz1, vx2, vy2, vz2);
  t=t;       // per ricordare che resta t=tf+h;

k14x1=vx1;
k24x1=ax1;
k14y1=vy1;
k24y1=ay1;
  k14z1=vz1;
k24z1=az1;

k14x2=vx2;
k24x2=ax2;
k14y2=vy2;
k24y2=ay2;
  k14z2=vz2;
k24z2=az2;

x1=xf1+h6*(k11x1+2.0*(k12x1+k13x1)+k14x1);
vx1=vxf1+h6*(k21x1+2.0*(k22x1+k23x1)+k24x1);
y1=yf1+h6*(k11y1+2.0*(k12y1+k13y1)+k14y1);
vy1=vyf1+h6*(k21y1+2.0*(k22y1+k23y1)+k24y1);
z1=zf1+h6*(k11z1+2.0*(k12z1+k13z1)+k14z1);
vz1=vzf1+h6*(k21z1+2.0*(k22z1+k23z1)+k24z1);

  x2=xf2+h6*(k11x2+2.0*(k12x2+k13x2)+k14x2);
vx2=vxf2+h6*(k21x2+2.0*(k22x2+k23x2)+k24x2);
y2=yf2+h6*(k11y2+2.0*(k12y2+k13y2)+k14y2);
```

```
367   vy2=vyf2+h6*(k21y2+2.0*(k22y2+k23y2)+k24y2);
368   z2=zf2+h6*(k11z2+2.0*(k12z2+k13z2)+k14z2);
369   vz2=vzf2+h6*(k21z2+2.0*(k22z2+k23z2)+k24z2);
370 }
```

A.2.2 Riferimento del CdM

Il codice è sostanzialmente quello dell'App. A.2.1, con le seguenti modifiche. Dopo il preambolo, identico, nella definizione delle variabili sono dichiarate le componenti della posizione e della velocità del centro di massa:

```
1  # include <iostream >
2  (...)
3  using namespace std;
4
5  const int P = 110000;
6  double G, m1, m2;
7  (...)
8  double ax2, ay2, az2;
9  double xG, yG, zG;
10 double vxG, vyG, vzG;
11 double v1circ, v2circ;
```

Con le stesse identiche righe del codice dell'App. A.2.1, segue poi la dichiarazione delle funzioni `RK, RKSTEP, acc, vel` ed il programma principale, immutato. Anche il codice delle funzioni è immutato, tranne per la `RK`, da sostituirsi con:

```
1  void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
       const double& Y01, const double& Vy01, const double& Z01, const double& Vz01,
       const double& X02, const double& Vx02, const double& Y02, const double& Vy02,
       const double& Z02, const double& Vz02, const int& Nt)
2  {double dt;
3  double TS, XS1, VxS1, YS1, VyS1, ZS1, VzS1, XS2, VxS2, YS2, VyS2, ZS2, VzS2;
4  int i;
5  dt =(Tf-Ti)/(Nt-1);
6
7  T[0]=Ti;
8
9  xG=(m1*X01+m2*X02)/(m1+m2);   // caratteristiche iniziali del CdM
10 yG=(m1*Y01+m2*Y02)/(m1+m2);
11 zG=(m1*Z01+m2*Z02)/(m1+m2);
12 vxG=(m1*Vx01+m2*Vx02)/(m1+m2);
13 vyG=(m1*Vy01+m2*Vy02)/(m1+m2);
14 vzG=(m1*Vz01+m2*Vz02)/(m1+m2);
15
16 X1[0]=X01-xG;   // traslazione
17 Vx1[0]=Vx01-vxG;
18 Y1[0]=Y01-yG;
19 Vy1[0]=Vy01-vyG;
20 Z1[0]=Z01-zG;
21 Vz1[0]=Vz01-vzG;
22
23 X2[0]=X02-xG;
24 Vx2[0]=Vx02-vyG;
25 Y2[0]=Y02-yG;
26 Vy2[0]=Vy02-vyG;
27 Z2[0]=Z02-zG;
28 Vz2[0]=Vz02-vzG;
29
30 TS=Ti;
31
32 XS1=X1[0];   // inizializzazione vettori
```

```
VxS1=Vx1[0];
YS1=Y1[0];
VyS1=Vy1[0];
ZS1=Z1[0];
VzS1=Vz1[0];

XS2=X2[0];
VxS2=Vx2[0];
YS2=Y2[0];
VyS2=Vy2[0];
ZS2=Z2[0];
VzS2=Vz2[0];

for(i=1;i<Nt;i++) // inizio ciclo
  {
  RKSTEP(TS,XS1,VxS1,YS1,VyS1,ZS1,VzS1,XS2,VxS2,YS2,VyS2,ZS2,VzS2,dt);  // passo
    del ciclo (integrazione)

  xG=(m1*XS1+m2*XS2)/(m1+m2);   // nuove caratteristiche del CdM
  yG=(m1*YS1+m2*YS2)/(m1+m2);
  zG=(m1*ZS1+m2*ZS2)/(m1+m2);
  vxG=(m1*VxS1+m2*VxS2)/(m1+m2);
  vyG=(m1*VyS1+m2*VyS2)/(m1+m2);
  vzG=(m1*VzS1+m2*VzS2)/(m1+m2);

  XS1=XS1-xG; // nuova traslazione
  YS1=YS1-yG;
  ZS1=ZS1-zG;
  VxS1=VxS1-vxG;
  VyS1=VyS1-vyG;
  VzS1=VzS1-vzG;

  XS2=XS2-xG;
  YS2=YS2-yG;
  ZS2=ZS2-zG;
  VxS2=VxS2-vxG;
  VyS2=VyS2-vyG;
  VzS2=VzS2-vzG;

  T[i]=TS;    // caricamento nel vettore

  X1[i]=XS1;
  Vx1[i]=VxS1;
  Y1[i]=YS1;
  Vy1[i]=VyS1;
  Z1[i]=ZS1;
  Vz1[i]=VzS1;

  X2[i]=XS2;
  Vx2[i]=VxS2;
  Y2[i]=YS2;
  Vy2[i]=VyS2;
  Z2[i]=ZS2;
  Vz2[i]=VzS2;
  }
}
```

La funzione `RKSTEP` invece non cambia.

A.3 Tre stelle mobili

Per meglio gestire i dati in uscita, rispetto al caso a due corpi vengono creati due files, uno con posizioni e velocità, l'altro con energie e momenti angolari.

A.3.1 Riferimento inerziale

```
// Problema dei 3 corpi reale in un sistema inerziale
# include <iostream >
# include <fstream >
# include <cstdlib>
# include <string>
# include <cmath>
using namespace std;

const int P = 420000;
double G, m1, m2, m3;    // anche se introdotti da file
double T[P];             // vettore tempo
double X1[P], Y1[P], Z1[P], Vx1[P], Vy1[P], Vz1[P], Ep1[P], Ec1[P], Et1[P], Lx1[P],
     Ly1[P], Lz1[P], Lt1[P];   // vettori tempo, posizioni, velocita', energie,
    momento, pianeta 1
double X2[P], Y2[P], Z2[P], Vx2[P], Vy2[P], Vz2[P], Ep2[P], Ec2[P], Et2[P], Lx2[P],
     Ly2[P], Lz2[P], Lt2[P];   // vettori tempo, posizioni, velocita', energie,
    momento, pianeta 2
double X3[P], Y3[P], Z3[P], Vx3[P], Vy3[P], Vz3[P], Ep3[P], Ec3[P], Et3[P], Lx3[P],
     Ly3[P], Lz3[P], Lt3[P];   // vettori tempo, posizioni, velocita', energie,
    momento, pianeta 3

double ET[P], LT[P]; // energia e momento angolare totali del sistema
double ax1, ay1, az1, Epot12, Ecin1, Lax1, Lay1, Laz1;      // accelerazioni,
    energie, momento angolare pianeta 1
double ax2, ay2, az2, Epot23, Ecin2, Lax2, Lay2, Laz2;      // accelerazioni,
    energie, momento angolare pianeta 2
double ax3, ay3, az3, Epot13, Ecin3, Lax3, Lay3, Laz3;      // accelerazioni,
    energie, momento angolare pianeta 3

// procedura RK4
void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
    const double& Y01, const double& Vy01, const double& Z01, const double& Vz01,
    const double& X02, const double& Vx02, const double& Y02, const double& Vy02,
    const double& Z02, const double& Vz02, const double& X03, const double& Vx03,
    const double& Y03, const double& Vy03, const double& Z03, const double& Vz03,
    const int& Nt);
void RKSTEP(double& t, double& x1, double& vx1, double& y1, double& vy1, double& z1
    , double& vz1, double& x2, double& vx2, double& y2, double& vy2, double& z2,
    double& vz2, double& x3, double& vx3, double& y3, double& vy3, double& z3,
    double& vz3, const double& dt);

// funzioni accelerazione, velocita', energia e momento
double acc(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
    double& x3, double& y3, double& z3, double& ax1, double& ay1, double& az1,
    double& ax2, double& ay2, double& az2, double& ax3, double& ay3, double& az3);
double vel(double& vx1, double& vy1, double& vz1, double& vx2, double& vy2, double&
     vz2, double& vx3, double& vy3, double& vz3);
double Ep(double& m1, double& m2, double& x1, double& y1, double& z1, double& x2,
    double& y2, double& z2, double& Epot);
double Enc(double& m1, double& vx, double& vy, double& vz, double& Ecin);
double La(double& m1, double& x, double& y, double& z , double& vx, double& vy,
    double& vz, double& Lax, double& Lay, double& Laz);

int main()  // PROGRAMMA PRINCIPALE
```

```
{
double Ti, Tf, X01, Y01, Z01, Vx01, Vy01, Vz01, X02, Y02, Z02, Vx02, Vy02, Vz02,
    X03, Y03, Z03, Vx03, Vy03, Vz03;
int Nt, i;              // tempo, posizioni e velocita' iniziali, passi
  ifstream ifile("input.txt");
    while (!ifile.eof()) {
            ifile >> G >> m1 >> m2 >> m3 >> Ti >> Tf >> Nt >> X01 >> Y01 >> Z01 >>
    Vx01 >> Vy01 >> Vz01 >> X02 >> Y02 >> Z02 >> Vx02 >> Vy02 >> Vz02 >> X03 >> Y03
     >> Z03 >> Vx03 >> Vy03 >> Vz03;}
    ifile.close();
    if(Nt>=P) {cerr << "Error : Nt>=P\n"; exit(1);}      // Nt deve essere <=P

    cout << "Massa Pianeta 1 del sistema: m1=" << m1 << "\n";   // output
    caratteristiche del sistema
    cout << "Massa Pianeta 2 del sistema: m2=" << m2 << "\n";
    cout << "Massa Pianeta 3 del sistema: m3=" << m3 << "\n";
    cout << "Posizione iniziale Pianeta 1: x=" << X01 << " y=" << Y01 << " z=" <<
    Z01 << "\n";
    cout << "Velocita' iniziale Pianeta 1: vx=" << Vx01 << " vy=" << Vy01 << " vz="
     << Vz01 << "\n";
  cout << "Posizione iniziale Pianeta 2: x=" << X02 << " y=" << Y02 << " z=" << Z02
     << "\n";
    cout << "Velocita' iniziale Pianeta 2: vx=" << Vx02 << " vy=" << Vy02 << " vz="
     << Vz02 << "\n";
    cout << "Posizione iniziale Pianeta 3: x=" << X03 << " y=" << Y03 << " z=" <<
    Z03 << "\n";
    cout << "Velocita' iniziale Pianeta 3: vx=" << Vx03 << " vy=" << Vy03 << " vz="
     << Vz03 << "\n";
  cout << "Tempo di simulazione: " << Ti << "-" << Tf << " e numero passi: " << Nt
    << "\n";
  // aumentando il tempo, si ottengono risultati riproducibili solo se si aumentano
     proporzionalmente i passi. Es. 3000 passi per ogni 0.1 s di simulazione

RK(Ti, Tf, X01, Vx01, Y01, Vy01, Z01, Vz01, X02, Vx02, Y02, Vy02, Z02, Vz02, X03,
    Vx03, Y03, Vy03, Z03, Vz03, Nt);       // Runge-Kutta 4
ofstream myfile1 ("rk43D3corpiXV.xls");
myfile1.precision(17);
for(i=0; i<Nt; i++)
myfile1<<T[i]<<"\t"<<X1[i]<<"\t"<<Y1[i]<<"\t"<<Z1[i]<<"\t"<<X2[i]<<"\t"<<Y2[i]<<"\t
    "<<Z2[i]<<"\t"<<X3[i]<<"\t"<<Y3[i]<<"\t"<<Z3[i]<<"\t"<<Vx1[i]<<"\t"<<Vy1[i]<<"\
    t"<<Vz1[i]<<"\t"<<Vx2[i]<<"\t"<<Vy2[i]<<"\t"<<Vz2[i]<<"\t"<<Vx3[i]<<"\t"<<Vy3[i
    ]<<"\t"<<Vz3[i]<<endl;
myfile1.close();
ofstream myfile2 ("rk43D3corpiEL.xls");
myfile2.precision(17);
for(i=0; i<Nt; i++)
myfile2<<T[i]<<"\t"<<Ep1[i]<<"\t"<<Ec1[i]<<"\t"<<Et1[i]<<"\t"<<Ep2[i]<<"\t"<<Ec2[i
    ]<<"\t"<<Et2[i]<<"\t"<<Ep3[i]<<"\t"<<Ec3[i]<<"\t"<<Et3[i]<<"\t"<<ET[i]<<"\t"<<
    Lx1[i]<<"\t"<<Ly1[i]<<"\t"<<Lz1[i]<<"\t"<<Lx2[i]<<"\t"<<Ly2[i]<<"\t"<<Lz2[i]<<"
    \t"<<Lx3[i]<<"\t"<<Ly3[i]<<"\t"<<Lz3[i]<<"\t"<<LT[i]<<endl;
myfile2.close();
}

// Funzioni
double acc(double& x1, double& y1, double& z1, double& x2, double& y2, double& z2,
    double& x3, double& y3, double& z3, double& ax1, double& ay1, double& az1,
    double& ax2, double& ay2, double& az2, double& ax3, double& ay3, double& az3)
{
double r12=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2)+(z1-z2)*(z1-z2));
double r13=sqrt((x1-x3)*(x1-x3)+(y1-y3)*(y1-y3)+(z1-z3)*(z1-z3));
double r23=sqrt((x2-x3)*(x2-x3)+(y2-y3)*(y2-y3)+(z2-z3)*(z2-z3));
double f12=-G*m1*m2/(r12*r12);
double f13=-G*m1*m3/(r13*r13);
double f23=-G*m2*m3/(r23*r23);
```

```
ax1=(f12*(x1-x2)/r12+f13*(x1-x3)/r13)/m1;
ax2=(-f12*(x1-x2)/r12+f23*(x2-x3)/r23)/m2;
ax3=(-f13*(x1-x3)/r13-f23*(x2-x3)/r23)/m3;
ay1=(f12*(y1-y2)/r12+f13*(y1-y3)/r13)/m1;
ay2=(-f12*(y1-y2)/r12+f23*(y2-y3)/r23)/m2;
ay3=(-f13*(y1-y3)/r13-f23*(y2-y3)/r23)/m3;
az1=(f12*(z1-z2)/r12+f13*(z1-z3)/r13)/m1;
az2=(-f12*(z1-z2)/r12+f23*(z2-z3)/r23)/m2;
az3=(-f13*(z1-z3)/r13-f23*(z2-z3)/r23)/m3;
}

double vel(double& vx1, double& vy1, double& vz1, double& vx2, double& vy2, double&
     vz2, double& vx3, double& vy3, double& vz3)
{
vx1=vx1;
vy1=vy1;
vz1=vz1;
vx2=vx2;
vy2=vy2;
vz2=vz2;
vx3=vx3;
vy3=vy3;
vz3=vz3;
}

double Ep(double& m1, double& m2, double& x1, double& y1, double& z1, double& x2,
    double& y2, double& z2, double& Epot)  // energia potenziale
{
double r;
r=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2)+(z1-z2)*(z1-z2));
Epot=-G*m1*m2/r;
}

double Enc(double& m1, double& vx, double& vy, double& vz, double& Ecin)  //
    energia cinetica
{
double vsq;
vsq=vx*vx+vy*vy+vz*vz;
Ecin=0.5*m1*vsq;
}

double La(double& m1, double& x, double& y, double& z, double& vx, double& vy,
    double& vz, double& Lax, double& Lay, double& Laz)   // momento angolare
{
Lax=m1*(y*vz-z*vy);
Lay=m1*(z*vx-x*vz);
Laz=m1*(x*vy-y*vx);
}

// integratori

void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
    const double& Y01, const double& Vy01, const double& Z01, const double& Vz01,
    const double& X02, const double& Vx02, const double& Y02, const double& Vy02,
    const double& Z02, const double& Vz02, const double& X03, const double& Vx03,
    const double& Y03, const double& Vy03, const double& Z03, const double& Vz03,
    const int& Nt)
{
double dt;
double TS, XS1, VxS1, YS1, VyS1, ZS1, VzS1, XS2, VxS2, YS2, VyS2, ZS2, VzS2, XS3,
    VxS3, YS3, VyS3, ZS3, VzS3;
int i;
dt =(Tf-Ti)/(Nt-1);
T[0]=Ti;
```

```
130 X1[0]=X01;
131 Vx1[0]=Vx01;
132 Y1[0]=Y01;
133 Vy1[0]=Vy01;
134 Z1[0]=Z01;
135 Vz1[0]=Vz01;
136
137 X2[0]=X02;
138 Vx2[0]=Vx02;
139 Y2[0]=Y02;
140 Vy2[0]=Vy02;
141 Z2[0]=Z02;
142 Vz2[0]=Vz02;
143
144 X3[0]=X03;
145 Vx3[0]=Vx03;
146 Y3[0]=Y03;
147 Vy3[0]=Vy03;
148 Z3[0]=Z03;
149 Vz3[0]=Vz03;
150
151 Ep(m1, m2, X1[0], Y1[0], Z1[0], X2[0], Y2[0], Z2[0], Epot12);  // energia pianeta 1
152 Ep(m1, m3, X1[0], Y1[0], Z1[0], X3[0], Y3[0], Z3[0], Epot13);
153 Ep1[0]=Epot12+Epot13;
154
155 Enc(m1, Vx1[0], Vy1[0], Vz1[0], Ecin1);
156 Ec1[0]=Ecin1;
157 Et1[0]=Epot12+Epot13+Ecin1;
158 La(m1, X1[0], Y1[0], Z1[0], Vx1[0], Vy1[0], Vz1[0], Lax1, Lay1, Laz1);  // momento
    pianeta 1
159 Lx1[0]=Lax1;
160 Ly1[0]=Lay1;
161 Lz1[0]=Laz1;
162 Lt1[0]=sqrt(Lax1*Lax1+Lay1*Lay1+Laz1*Laz1);
163
164 Ep(m1, m2, X1[0], Y1[0], Z1[0], X2[0], Y2[0], Z2[0], Epot12);  // energia pianeta 2
165 Ep(m2, m3, X2[0], Y2[0], Z2[0], X3[0], Y3[0], Z3[0], Epot23);
166 Ep2[0]=Epot12+Epot23;
167
168 Enc(m2, Vx2[0], Vy2[0], Vz2[0], Ecin2);  // energia pianeta 2
169 Ec2[0]=Ecin2;
170 Et2[0]=Epot12+Epot23+Ecin2;
171 La(m2, X2[0], Y2[0], Z2[0], Vx2[0], Vy2[0], Vz2[0], Lax2, Lay2, Laz2);  // momento
    pianeta 2
172 Lx2[0]=Lax2;
173 Ly2[0]=Lay2;
174 Lz2[0]=Laz2;
175 Lt2[0]=sqrt(Lax2*Lax2+Lay2*Lay2+Laz2*Laz2);
176
177 Ep3[0]=Epot13+Epot23;
178
179 Enc(m3, Vx3[0], Vy3[0], Vz3[0], Ecin3);  // energia pianeta 3
180 Ec3[0]=Ecin3;
181 Et3[0]=Epot13+Epot23+Ecin3;
182 La(m3, X3[0], Y3[0], Z3[0], Vx3[0], Vy3[0], Vz3[0], Lax3, Lay3, Laz3);  // momento
    pianeta 3
183 Lx3[0]=Lax3;
184 Ly3[0]=Lay3;
185 Lz3[0]=Laz3;
186 Lt3[0]=sqrt(Lax3*Lax3+Lay3*Lay3+Laz3*Laz3);
187
188 ET[0]=Epot12+Epot13+Epot23+Ecin1+Ecin2+Ecin3;    // energia totale sistema
189 LT[0]=sqrt((Lax1+Lax2+Lax3)*(Lax1+Lax2+Lax3)+(Lay1+Lay2+Lay3)*(Lay1+Lay2+Lay3)+(
    Laz1+Laz2+Laz3)*(Laz1+Laz2+Laz3));    // momento angolare totale sistema
```

```
190
191
192 TS=Ti;
193
194 XS1=X01;
195 VxS1=Vx01;
196 YS1=Y01;
197 VyS1=Vy01;
198 ZS1=Z01;
199 VzS1=Vz01;
200
201 XS2=X02;
202 VxS2=Vx02;
203 YS2=Y02;
204 VyS2=Vy02;
205 ZS2=Z02;
206 VzS2=Vz02;
207
208 XS3=X03;
209 VxS3=Vx03;
210 YS3=Y03;
211 VyS3=Vy03;
212 ZS3=Z03;
213 VzS3=Vz03;
214
215 for(i=1;i<Nt;i++)
216   {
217   RKSTEP(TS,XS1,VxS1,YS1,VyS1,ZS1,VzS1,XS2,VxS2,YS2,VyS2,ZS2,VzS2,XS3,VxS3,YS3,VyS3
      ,ZS3,VzS3,dt);
218   T[i]=TS;
219   X1[i]=XS1;
220   Vx1[i]=VxS1;
221   Y1[i]=YS1;
222   Vy1[i]=VyS1;
223     Z1[i]=ZS1;
224   Vz1[i]=VzS1;
225
226     X2[i]=XS2;
227   Vx2[i]=VxS2;
228   Y2[i]=YS2;
229   Vy2[i]=VyS2;
230     Z2[i]=ZS2;
231   Vz2[i]=VzS2;
232
233     X3[i]=XS3;
234   Vx3[i]=VxS3;
235   Y3[i]=YS3;
236   Vy3[i]=VyS3;
237     Z3[i]=ZS3;
238   Vz3[i]=VzS3;
239
240   Ep(m1, m2, X1[i], Y1[i], Z1[i], X2[i], Y2[i], Z2[i], Epot12);  // energia pianeta
       1
241   Ep(m1, m3, X1[i], Y1[i], Z1[i], X3[i], Y3[i], Z3[i], Epot13);
242   Ep1[i]=Epot12+Epot13;
243
244   Enc(m1, Vx1[i], Vy1[i], Vz1[i], Ecin1);
245   Ec1[i]=Ecin1;
246   Et1[i]=Epot12+Epot13+Ecin1;
247   La(m1, X1[i], Y1[i], Z1[i], Vx1[i], Vy1[i], Vz1[i], Lax1, Lay1, Laz1);  //
      momento pianeta 1
248   Lx1[i]=Lax1;
249   Ly1[i]=Lay1;
250   Lz1[i]=Laz1;
```

```
251   Lt1[i]=sqrt(Lax1*Lax1+Lay1*Lay1+Laz1*Laz1);
252
253   Ep(m2, m3, X2[i], Y2[i], Z2[i], X3[i], Y3[i], Z3[i], Epot23);   // energia
        pianeta 2
254   Ep2[i]=Epot12+Epot23;
255
256   Enc(m2, Vx2[i], Vy2[i], Vz2[i], Ecin2);  // energia pianeta 2
257   Ec2[i]=Ecin2;
258   Et2[i]=Epot12+Epot23+Ecin2;
259   La(m2, X2[i], Y2[i], Z2[i], Vx2[i], Vy2[i], Vz2[i], Lax2, Lay2, Laz2);  //
        momento pianeta 2
260   Lx2[i]=Lax2;
261   Ly2[i]=Lay2;
262   Lz2[i]=Laz2;
263   Lt2[i]=sqrt(Lax2*Lax2+Lay2*Lay2+Laz2*Laz2);
264
265   Ep3[i]=Epot13+Epot23;
266
267   Enc(m3, Vx3[i], Vy3[i], Vz3[i], Ecin3);  // energia pianeta 3
268   Ec3[i]=Ecin3;
269   Et3[i]=Epot13+Epot23+Ecin3;
270   La(m3, X3[i], Y3[i], Z3[i], Vx3[i], Vy3[i], Vz3[i], Lax3, Lay3, Laz3);  //
        momento pianeta 3
271   Lx3[i]=Lax3;
272   Ly3[i]=Lay3;
273   Lz3[i]=Laz3;
274   Lt3[i]=sqrt(Lax3*Lax3+Lay3*Lay3+Laz3*Laz3);
275
276   ET[i]=Epot12+Epot13+Epot23+Ecin1+Ecin2+Ecin3;    // energia totale sistema
277   LT[i]=sqrt((Lax1+Lax2+Lax3)*(Lax1+Lax2+Lax3)+(Lay1+Lay2+Lay3)*(Lay1+Lay2+Lay3)+(
        Laz1+Laz2+Laz3)*(Laz1+Laz2+Laz3));    // momento angolare totale sistema
278   }
279 }
280
281 void RKSTEP(double& t, double& x1, double& vx1, double& y1, double& vy1, double& z1
      , double& vz1, double& x2, double& vx2, double& y2, double& vy2, double& z2,
      double& vz2, double& x3, double& vx3, double& y3, double& vy3, double& z3,
      double& vz3, const double& dt)
282 {
283
284     double h, h2, h6, tf;
285   h=dt;
286   h2=0.5*h;
287   h6=h/6.0;
288   tf=t;
289
290   double xf1, yf1, zf1, vxf1, vyf1, vzf1;
291   double k11x1, k12x1, k13x1, k14x1, k21x1, k22x1, k23x1, k24x1;
292   double k11y1, k12y1, k13y1, k14y1, k21y1, k22y1, k23y1, k24y1;
293     double k11z1, k12z1, k13z1, k14z1, k21z1, k22z1, k23z1, k24z1;
294     xf1=x1;
295     yf1=y1;
296     zf1=z1;
297     vxf1=vx1;
298     vyf1=vy1;
299     vzf1=vz1;
300
301     double xf2, yf2, zf2, vxf2, vyf2, vzf2;
302   double k11x2, k12x2, k13x2, k14x2, k21x2, k22x2, k23x2, k24x2;
303   double k11y2, k12y2, k13y2, k14y2, k21y2, k22y2, k23y2, k24y2;
304     double k11z2, k12z2, k13z2, k14z2, k21z2, k22z2, k23z2, k24z2;
305     xf2=x2;
306     yf2=y2;
307     zf2=z2;
```

```
308     vxf2=vx2;
309     vyf2=vy2;
310     vzf2=vz2;
311
312   double xf3, yf3, zf3, vxf3, vyf3, vzf3;
313   double k11x3, k12x3, k13x3, k14x3, k21x3, k22x3, k23x3, k24x3;
314   double k11y3, k12y3, k13y3, k14y3, k21y3, k22y3, k23y3, k24y3;
315     double k11z3, k12z3, k13z3, k14z3, k21z3, k22z3, k23z3, k24z3;
316     xf3=x3;
317     yf3=y3;
318     zf3=z3;
319     vxf3=vx3;
320     vyf3=vy3;
321     vzf3=vz3;
322
323
324   acc(x1, y1, z1, x2, y2, z2, x3, y3, z3, ax1, ay1, az1, ax2, ay2, az2, ax3, ay3,
        az3);
325     vel(vx1, vy1, vz1, vx2, vy2, vz2, vx3, vy3, vz3);
326     t=tf+h2;
327
328   k21x1=ax1;
329   k11x1=vx1;
330   vx1=vxf1+h2*k21x1;
331   x1=xf1+h2*k11x1;
332   k21y1=ay1;
333     k11y1=vy1;
334     vy1=vyf1+h2*k21y1;
335   y1=yf1+h2*k11y1;
336   k21z1=az1;
337   k11z1=vz1;
338   vz1=vzf1+h2*k21z1;
339     z1=zf1+h2*k11z1;
340
341     k21x2=ax2;
342   k11x2=vx2;
343   vx2=vxf2+h2*k21x2;
344   x2=xf2+h2*k11x2;
345   k21y2=ay2;
346     k11y2=vy2;
347     vy2=vyf2+h2*k21y2;
348   y2=yf2+h2*k11y2;
349   k21z2=az2;
350   k11z2=vz2;
351   vz2=vzf2+h2*k21z2;
352     z2=zf2+h2*k11z2;
353
354     k21x3=ax3;
355   k11x3=vx3;
356   vx3=vxf3+h2*k21x3;
357   x3=xf3+h2*k11x3;
358   k21y3=ay3;
359     k11y3=vy3;
360     vy3=vyf3+h2*k21y3;
361   y3=yf3+h2*k11y3;
362   k21z3=az3;
363   k11z3=vz3;
364   vz3=vzf3+h2*k21z3;
365     z3=zf3+h2*k11z3;
366
367
368   acc(x1, y1, z1, x2, y2, z2, x3, y3, z3, ax1, ay1, az1, ax2, ay2, az2, ax3, ay3,
        az3);
369     vel(vx1, vy1, vz1, vx2, vy2, vz2, vx3, vy3, vz3);
```

```
370  t=t;  // per ricordare che resta t=tf+h2;
371
372  k22x1=ax1;
373  k12x1=vx1;
374  k22y1=ay1;
375  k12y1=vy1;
376  k22z1=az1;
377  k12z1=vz1;
378  vx1=vxf1+h2*k22x1;
379  x1=xf1+h2*k12x1;
380  vy1=vyf1+h2*k22y1;
381  y1=yf1+h2*k12y1;
382    vz1=vzf1+h2*k22z1;
383  z1=zf1+h2*k12z1;
384
385    k22x2=ax2;
386  k12x2=vx2;
387  k22y2=ay2;
388  k12y2=vy2;
389  k22z2=az2;
390  k12z2=vz2;
391  vx2=vxf2+h2*k22x2;
392  x2=xf2+h2*k12x2;
393  vy2=vyf2+h2*k22y2;
394  y2=yf2+h2*k12y2;
395    vz2=vzf2+h2*k22z2;
396  z2=zf2+h2*k12z2;
397
398    k22x3=ax3;
399  k12x3=vx3;
400  k22y3=ay3;
401  k12y3=vy3;
402  k22z3=az3;
403  k12z3=vz3;
404  vx3=vxf3+h2*k22x3;
405  x3=xf3+h2*k12x3;
406  vy3=vyf3+h2*k22y3;
407  y3=yf3+h2*k12y3;
408    vz3=vzf3+h2*k22z3;
409  z3=zf3+h2*k12z3;
410
411  acc(x1, y1, z1, x2, y2, z2, x3, y3, z3, ax1, ay1, az1, ax2, ay2, az2, ax3, ay3,
       az3);
412    vel(vx1, vy1, vz1, vx2, vy2, vz2, vx3, vy3, vz3);
413    t=tf+h;
414
415  k23x1=ax1;
416  k13x1=vx1;
417  k23y1=ay1;
418  k13y1=vy1;
419  k23z1=az1;
420  k13z1=vz1;
421  x1=xf1+h*k13x1;
422  y1=yf1+h*k13y1;
423  z1=zf1+h*k13z1;
424  vx1=vxf1+h*k23x1;
425  vy1=vyf1+h*k23y1;
426  vz1=vzf1+h*k23z1;
427
428    k23x2=ax2;
429  k13x2=vx2;
430  k23y2=ay2;
431  k13y2=vy2;
432  k23z2=az2;
```

```
433   k13z2=vz2;
434   x2=xf2+h*k13x2;
435   y2=yf2+h*k13y2;
436   z2=zf2+h*k13z2;
437   vx2=vxf2+h*k23x2;
438   vy2=vyf2+h*k23y2;
439   vz2=vzf2+h*k23z2;
440
441     k23x3=ax3;
442   k13x3=vx3;
443   k23y3=ay3;
444   k13y3=vy3;
445   k23z3=az3;
446   k13z3=vz3;
447   x3=xf3+h*k13x3;
448   y3=yf3+h*k13y3;
449   z3=zf3+h*k13z3;
450   vx3=vxf3+h*k23x3;
451   vy3=vyf3+h*k23y3;
452   vz3=vzf3+h*k23z3;
453
454   acc(x1, y1, z1, x2, y2, z2, x3, y3, z3, ax1, ay1, az1, ax2, ay2, az2, ax3, ay3,
        az3);
455     vel(vx1, vy1, vz1, vx2, vy2, vz2, vx3, vy3, vz3);
456     t=t;  // per ricordare che resta t=tf+h;
457
458   k14x1=vx1;
459   k24x1=ax1;
460   k14y1=vy1;
461   k24y1=ay1;
462     k14z1=vz1;
463   k24z1=az1;
464
465   k14x2=vx2;
466   k24x2=ax2;
467   k14y2=vy2;
468   k24y2=ay2;
469     k14z2=vz2;
470   k24z2=az2;
471
472   k14x3=vx3;
473   k24x3=ax3;
474   k14y3=vy3;
475   k24y3=ay3;
476     k14z3=vz3;
477   k24z3=az3;
478
479   x1=xf1+h6*(k11x1+2.0*(k12x1+k13x1)+k14x1);
480   vx1=vxf1+h6*(k21x1+2.0*(k22x1+k23x1)+k24x1);
481   y1=yf1+h6*(k11y1+2.0*(k12y1+k13y1)+k14y1);
482   vy1=vyf1+h6*(k21y1+2.0*(k22y1+k23y1)+k24y1);
483   z1=zf1+h6*(k11z1+2.0*(k12z1+k13z1)+k14z1);
484   vz1=vzf1+h6*(k21z1+2.0*(k22z1+k23z1)+k24z1);
485
486     x2=xf2+h6*(k11x2+2.0*(k12x2+k13x2)+k14x2);
487   vx2=vxf2+h6*(k21x2+2.0*(k22x2+k23x2)+k24x2);
488   y2=yf2+h6*(k11y2+2.0*(k12y2+k13y2)+k14y2);
489   vy2=vyf2+h6*(k21y2+2.0*(k22y2+k23y2)+k24y2);
490   z2=zf2+h6*(k11z2+2.0*(k12z2+k13z2)+k14z2);
491   vz2=vzf2+h6*(k21z2+2.0*(k22z2+k23z2)+k24z2);
492
493     x3=xf3+h6*(k11x3+2.0*(k12x3+k13x3)+k14x3);
494   vx3=vxf3+h6*(k21x3+2.0*(k22x3+k23x3)+k24x3);
495   y3=yf3+h6*(k11y3+2.0*(k12y3+k13y3)+k14y3);
```

```
  vy3=vyf3+h6*(k21y3+2.0*(k22y3+k23y3)+k24y3);
  z3=zf3+h6*(k11z3+2.0*(k12z3+k13z3)+k14z3);
  vz3=vzf3+h6*(k21z3+2.0*(k22z3+k23z3)+k24z3);
}
```

A.3.2 Riferimento del CdM

Il codice è sostanzialmente quello dell'App. A.3.1, con le seguenti modifiche. Dopo il preambolo, identico, nella definizione delle variabili sono dichiarate le componenti della posizione e della velocità del centro di massa:

```
# include <iostream >
(...)
using namespace std;

const int P = 420000;
double G, m1, m2, m3;
(...)
double ax3, ay3, az3;
double xG, yG, zG;
double vxG, vyG, vzG;
```

Con le stesse identiche righe del codice dell'App. A.3.1, segue poi la dichiarazione delle funzioni `RK, RKSTEP, acc, vel` ed il programma principale, immutato. Anche il codice delle funzioni è immutato, tranne per la `RK`, da sostituirsi con:

```
void RK(const double& Ti, const double& Tf, const double& X01, const double& Vx01,
    const double& Y01, const double& Vy01, const double& Z01, const double& Vz01,
    const double& X02, const double& Vx02, const double& Y02, const double& Vy02,
    const double& Z02, const double& Vz02, const double& X03, const double& Vx03,
    const double& Y03, const double& Vy03, const double& Z03, const double& Vz03,
    const int& Nt)
{
double dt;
double TS, XS1, VxS1, YS1, VyS1, ZS1, VzS1, XS2, VxS2, YS2, VyS2, ZS2, VzS2, XS3,
    VxS3, YS3, VyS3, ZS3, VzS3;
int i;
dt =(Tf-Ti)/(Nt-1);
T[0]=Ti;

xG=(m1*X01+m2*X02+m3*X03)/(m1+m2+m3);
yG=(m1*Y01+m2*Y02+m3*Y03)/(m1+m2+m3);
zG=(m1*Z01+m2*Z02+m3*Z03)/(m1+m2+m3);
vxG=(m1*Vx01+m2*Vx02+m3*Vx03)/(m1+m2+m3);
vyG=(m1*Vy01+m2*Vy02+m3*Vy03)/(m1+m2+m3);
vzG=(m1*Vz01+m2*Vz02+m3*Vz03)/(m1+m2+m3);

X1[0]=X01-xG;
Vx1[0]=Vx01-vxG;
Y1[0]=Y01-yG;
Vy1[0]=Vy01-vyG;
Z1[0]=Z01-zG;
Vz1[0]=Vz01-vzG;

X2[0]=X02-xG;
Vx2[0]=Vx02-vxG;
Y2[0]=Y02-yG;
Vy2[0]=Vy02-vyG;
Z2[0]=Z02-zG;
Vz2[0]=Vz02-vzG;

X3[0]=X03-xG;
```

```
Vx3[0]=Vx03-vxG;
Y3[0]=Y03-yG;
Vy3[0]=Vy03-vyG;
Z3[0]=Z03-zG;
Vz3[0]=Vz03-vzG;

TS=Ti;

XS1=X1[0];
VxS1=Vx1[0];
YS1=Y1[0];
VyS1=Vy1[0];
ZS1=Z1[0];
VzS1=Vz1[0];

XS2=X2[0];
VxS2=Vx2[0];
YS2=Y2[0];
VyS2=Vy2[0];
ZS2=Z2[0];
VzS2=Vz2[0];

XS3=X3[0];
VxS3=Vx3[0];
YS3=Y3[0];
VyS3=Vy3[0];
ZS3=Z3[0];
VzS3=Vz3[0];

for(i+1;i<Nt;i++)
  {
  RKSTEP(TS,XS1,VxS1,YS1,VyS1,ZS1,VzS1,XS2,VxS2,YS2,VyS2,ZS2,VzS2,XS3,VxS3,YS3,VyS3
    ,ZS3,VzS3,dt);

  xG=(m1*XS1+m2*XS2+m3*XS3)/(m1+m2+m3);
  yG=(m1*YS1+m2*YS2+m3*YS3)/(m1+m2+m3);
  zG=(m1*ZS1+m2*ZS2+m3*ZS3)/(m1+m2+m3);
  vxG=(m1*VxS1+m2*VxS2+m3*VxS3)/(m1+m2+m3);
  vyG=(m1*VyS1+m2*VyS2+m3*VyS3)/(m1+m2+m3);
  vzG=(m1*VzS1+m2*VzS2+m3*VzS3)/(m1+m2+m3);

  XS1=XS1-xG;
    YS1=YS1-yG;
    ZS1=ZS1-zG;
    VxS1=VxS1-vxG;
    VyS1=VyS1-vyG;
    VzS1=VzS1-vzG;

  XS2=XS2-xG;
  YS2=YS2-yG;
  ZS2=ZS2-zG;
  VxS2=VxS2-vxG;
  VyS2=VyS2-vyG;
  VzS2=VzS2-vzG;

  XS3=XS3-xG;
  YS3=YS3-yG;
  ZS3=ZS3-zG;
  VxS3=VxS3-vxG;
  VyS3=VyS3-vyG;
  VzS3=VzS3-vzG;

  T[i]=TS;
  X1[i]=XS1;
```

```
  Vx1[i]=VxS1;
  Y1[i]=YS1;
  Vy1[i]=VyS1;
  Z1[i]=ZS1;
  Vz1[i]=VzS1;

  X2[i]=XS2;
  Vx2[i]=VxS2;
  Y2[i]=YS2;
  Vy2[i]=VyS2;
  Z2[i]=ZS2;
  Vz2[i]=VzS2;

  X3[i]=XS3;
  Vx3[i]=VxS3;
  Y3[i]=YS3;
  Vy3[i]=VyS3;
  Z3[i]=ZS3;
  Vz3[i]=VzS3;
  }
}
```

La funzione `RKSTEP` invece non cambia.

A.4 Codice VPython (F. De Vito, [3])

```

Web VPython 3.2
from vpython import *
# GlowScript 2.7 VPython

# Settings
step = 0.01 # in secondi -> in ordine centesimale imprecisione di L centesimale
G = 100
step_rate = 120000
lagrange_on = False
render_vettori = False # Momento angolare
screen_times = [200,600,100,1300,1600]
time = 0
sinodale = True

# Sistema di riferimento in O
scene1 = canvas(width=725, height=300,
                align='left', background=color.white)

# Grafici
gd_L = graph(title="Momento Angolare L", xtitle="t[s]", ytitle="L[kg*m^2/s]", width
    =363,
    height=300,align='right')
curva_x = gcurve(graph=gd_L, color=color.red)
curva_y = gcurve(graph=gd_L, color=color.yellow)
curva_z = gcurve(graph=gd_L, color=color.blue)

gd_E = graph(title="Energia totale E", xtitle="t[s]", ytitle="E[J]", width=362,
    height=300,align='right')
curva_E = gcurve(graph=gd_E, color=color.yellow)

# Sistema di riferimento in G
scene2 = canvas(width=725, height=300,
                align='left', background=color.white)

```

```
35 # Sistema di riferimento rispetto a un corpo / sinodale
36 scene3 = canvas(width=725, height=300,
37                 align='right', background=color.white)
38
39 # Lista di sistemi
40 scenes = [scene1,scene2,scene3]
41
42 # vcos, vsin
43 def vcos(x):
44   if x == 0:
45     return 1
46   elif x == pi/2:
47     return 0
48   elif x == pi:
49     return -1
50   elif x == 3*pi/2:
51     return 0
52   else:
53     return cos(x)
54
55 def vsin(x):
56   if x == 0:
57     return 0
58   elif x == pi/2:
59     return 1
60   elif x == pi:
61     return 0
62   elif x == 3*pi/2:
63     return -1
64   else:
65     return sin(x)
66
67 # Vectors
68 def vettore(angles,mod):
69   vt = vector(0,0,0)
70   vt.x = mod*vsin(angles[0])*vcos(angles[1])
71   vt.y = mod*vsin(angles[0])*vsin(angles[1])
72   vt.z = mod*vcos(angles[0])
73   return vt
74
75 def modulo(vt):
76   mod = vt.x**2+vt.y**2+vt.z**2
77   mod = mod**0.5
78   return mod
79
80 # Angles
81 def angle(locA, locB):
82     delta = locB - locA
83     distance = modulo(vt=delta)
84     theta = acos(delta.z/distance)
85     fi = atan2(delta.y,delta.x)
86     return [theta, fi]
87
88 # Position -> Speed -> Acceleration
89 def from_spd_to_locat(loc_,speed,original):
90   loc = loc_ + speed*step
91   return loc
92
93 def from_accel_to_spd(speed,accel):
94   spd = speed + accel*step
95   return spd
96
97 def from_locat_to_accel(angles_,locA,locB,massB):
98   delta = locA - locB
```

```
99    distance = modulo(vt=delta)
100   accel = (massB*G)/(distance**2)
101   accel = vettore(angles=angles_,mod=accel)
102   return accel
103
104 # Body
105 class Body:
106   def __init__(self, name, mass, loc, clr, spd, fixed=False):
107     self.name = name
108     self.rd = 20
109     self.clr = clr
110     self.mass = mass
111     self.locat = []
112     self.speed = []
113     self.accel = []
114     self.fixed = fixed
115     self.momentum = []
116     self.angular_momentum = []
117     self.kinetik_energy = []
118     self.klist = []
119     self.jlist = []
120
121     for i in range(3):
122       self.locat.append(loc)
123       self.speed.append(spd)
124       self.accel.append(vector(0,0,0))
125       self.momentum.append(vector(0,0,0))
126       self.angular_momentum.append(vector(0,0,0))
127       self.kinetik_energy.append(0)
128
129     for i in range(4):
130       self.klist.append(vector(0,0,0))
131       self.jlist.append(vector(0,0,0))
132
133     self.update()
134
135   def update(self):
136     for i in range(3):
137       self.momentum[i] = self.mass*self.speed[i]
138       self.angular_momentum[i] = cross(self.locat[i],self.momentum[i])
139       self.kinetik_energy[i] = modulo(vt=self.speed[i])**2*self.mass*0.5
140
141 # Sistem
142 class System:
143   def __init__(self, bodies):
144     self.bodies = bodies
145     self.position = vector(0,0,0)
146     self.speed = vector(0,0,0)
147     self.angular_momentum = []
148     self.energy = []
149     self.lagrange = [sphere(canvas=scene3, pos=vector(0,0,0), radius=12, color=
      color.white,
150       opacity=0.5, make_trail=False),sphere(canvas=scene3, pos=vector(0,0,0),
      radius=12,
151       color=color.white, opacity=0.5, make_trail=False)]
152
153     for i in range(3):
154       self.angular_momentum.append(vector(0,0,0))
155       self.energy.append(0)
156
157     self.next_step()
158
159   def next_step(self):
160     self.update_self()
```

```
161     self.relocing()
162     self.update()
163     if len(self.bodies) == 3 and lagrange_on:
164       self.lagrange_points()
165
166
167   def update_self(self):
168     static_moment = vector(0,0,0)
169     dstatic_moment = vector(0,0,0)
170     total_mass = 0
171     for body in self.bodies:
172       static_moment += body.mass*body.locat[0]
173       dstatic_moment += body.mass*body.speed[0]
174       total_mass += body.mass
175     self.position = static_moment/total_mass
176     self.speed = dstatic_moment/total_mass
177
178   def relocing(self):
179     for body in self.bodies:
180       # Sistema di riferimento in G
181       body.locat[1] = body.locat[0]
182       body.locat[1] -= self.position
183       body.speed[1] = body.speed[0]
184       body.speed[1] -= self.speed
185
186       # Sistema di riferimento del primo corpo
187       if len(self.bodies) != 3 or not sinodale:
188         body.locat[2] = body.locat[0]
189         body.locat[2] -= self.bodies[0].locat[0]
190         body.speed[2] = body.speed[0]
191         body.speed[2] -= self.bodies[0].speed[0]
192
193     # Sistema di riferimento sinodale
194     if len(self.bodies) == 3 and sinodale:
195       static_moment = self.bodies[0].mass*self.bodies[0].locat[0]
196         +self.bodies[1].mass*self.bodies[1].locat[0]
197       total_mass = self.bodies[0].mass+self.bodies[1].mass
198       gposition = static_moment/total_mass
199       base = self.bodies[0].locat[0]-gposition
200       alt = self.bodies[2].locat[0]-gposition
201       dbase = self.bodies[0].speed[0]
202       dalt = self.bodies[2].speed[0]
203       dis_1 = modulo(vt=vector(base.x,base.y,0))
204       dis_2 = modulo(vt=base)
205       s_a = base.y/dis_1
206       c_a = base.x/dis_1
207       s_b = base.z/dis_2
208       c_b = dis_1/dis_2
209       self.bodies[0].locat[2]=vector(dis_2,0,0)
210       self.bodies[1].locat[2]=-self.bodies[0].locat[2]*self.bodies[0].mass/self.
    bodies[1].mass
211       comp_x = (alt.x*c_a+alt.y*s_a)*c_b+alt.z*s_b
212       comp_y = -alt.x*s_a+alt.y*c_a
213       comp_z = -(alt.x*c_a+alt.y*s_a)*s_b+alt.z*c_b
214       self.bodies[2].locat[2]=vector(comp_x,comp_y,comp_z)
215       self.bodies[0].speed[2]=vector((base.x*dbase.x+base.y*dbase.y+base.z*dbase.z)
    /
216         s1.bodies[0].locat[2].x,0,0)
217       self.bodies[1].speed[2]=-self.bodies[0].speed[2]*self.bodies[0].mass/self.
    bodies[1].mass
218       d_a = (base.x*dbase.y-base.y*dbase.x)/(dis_1**2)
219       d_b = ((dis_2)**2*dbase.z-base.z*(base.x*dbase.x+base.y*dbase.y+base.z*dbase.
    z))/
220         ((dis_2**2)*((dis_2**2-base.z**2)**0.5))
```

```
221       comp_a = alt.x*c_a+alt.y*s_a
222       dcomp_a = dalt.x*c_a+dalt.y*s_a+d_a*(alt.y*c_a-alt.x*s_a)
223       dcomp_x = dcomp_a*c_b-comp_a*d_b*s_b+dalt.z*s_b+alt.z*c_b*d_b
224       dcomp_y = -dalt.x*s_a+dalt.y*c_a-d_a*(alt.x*c_a+alt.y*s_a)
225       dcomp_z = -dcomp_a*s_b-comp_a*d_b*c_b+dalt.z*c_b+alt.z*s_b*d_b
226       self.bodies[2].speed[2]=vector(dcomp_x,dcomp_y,dcomp_z)
227       self.bodies[0].speed[2] = vector(0,0,0)
228       self.bodies[1].speed[2] = vector(0,0,0)
229       self.bodies[2].speed[2] = vector(0,0,0)
230
231   def update(self):
232     # Update variabili
233     for body in self.bodies:
234       body.update()
235
236     for i in range(3):
237       self.angular_momentum[i] = vector(0,0,0)
238       self.energy[i] = 0
239       for j in range(len(self.bodies)):
240         self.angular_momentum[i] += self.bodies[j].angular_momentum[i]
241         self.energy[i] += self.bodies[j].kinetik_energy[i]
242         for k in range(len(self.bodies)):
243           if k > j:
244             delta = self.bodies[k].locat[i] - self.bodies[j].locat[i]
245             distance = modulo(vt=delta)
246             if i == 0: # Sistema inerziale
247               self.energy[i] -= G*self.bodies[j].mass*self.bodies[k].mass/distance
248
249       if i == 2 and len(self.bodies) == 2: # Centrato nel primo corpo
250         self.energy[i] -= G*self.bodies[1].mass*(self.bodies[0].mass+self.bodies
    [1].mass)/
251             modulo(self.bodies[0].locat[2]-self.bodies[1].locat[2])
252
253   def lagrange_points(self):
254     R = abs(self.bodies[1].locat[2].x - self.bodies[0].locat[2].x)
255     M = self.bodies[1].mass/(self.bodies[0].mass+self.bodies[1].mass)
256     #Langrange 4 && 5
257     self.lagrange[0].pos = vector(R*(2*M-1)/2,R*(3**0.5)/2,0)
258     self.lagrange[1].pos = vector(R*(2*M-1)/2,-R*(3**0.5)/2,0)
259
260
261 # RK4:
262 def runge_kutta(rk_bodies):
263   # Klist -> Speed
264   for kdex in range(4):
265     for body in rk_bodies:
266       if kdex == 0:
267         body.klist[kdex] = body.speed[0]
268       elif kdex == 1 or kdex == 2:
269         body.klist[kdex] = body.speed[0] + body.klist[kdex-1]*step/2
270       elif kdex == 3:
271         body.klist[kdex] = body.speed[0] + body.klist[kdex-1]
272       body.klist[kdex] = from_spd_to_locat(loc_=body.locat[0], speed=body.klist[
    kdex],
273           original=body.speed[0])
274
275     for index in range(len(rk_bodies)):
276       rk_bodies[index].accel = vector(0,0,0)
277       for altdex in range(len(rk_bodies)):
278         if altdex != index:
279           new_angle = angle(locA=rk_bodies[index].klist[kdex],locB=rk_bodies[altdex
    ].klist[kdex])
280           parz = from_locat_to_accel(angles_=new_angle,locA=rk_bodies[index].klist[
    kdex],
```

```
281             locB=rk_bodies[altdex].klist[kdex],massB=rk_bodies[altdex].mass)
282           rk_bodies[index].accel = rk_bodies[index].accel + parz
283     for body in rk_bodies:
284       body.klist[kdex] = body.accel
285   for body in rk_bodies:
286     total = body.klist[0] + body.klist[1]*2 + body.klist[2]*2 + body.klist[3]
287     total = total*step/6
288     if not body.fixed:
289       body.speed[0] = body.speed[0] + total
290
291   # Jlist -> Position
292   for jdex in range(4):
293     for body in rk_bodies:
294       if jdex == 0:
295         body.jlist[jdex] = body.locat[0]
296       elif jdex == 1 or jdex == 2:
297         body.jlist[jdex] = body.locat[0] + body.jlist[jdex-1]*step/2
298       elif jdex == 3:
299         body.jlist[jdex] = body.locat[0] + body.jlist[jdex-1]
300     for index in range(len(rk_bodies)):
301       rk_bodies[index].accel = vector(0,0,0)
302       for altdex in range(len(rk_bodies)):
303         if altdex != index:
304           new_angle = angle(locA=rk_bodies[index].jlist[jdex],locB=rk_bodies[altdex
    ].jlist[jdex])
305           rk_bodies[index].accel = rk_bodies[index].accel + from_locat_to_accel(
    angles_=new_angle,
306             locA=rk_bodies[index].jlist[jdex], locB=rk_bodies[altdex].jlist[jdex],
307             massB=rk_bodies[altdex].mass)
308     for body in rk_bodies:
309       body.jlist[jdex] = body.accel
310       body.jlist[jdex] = from_accel_to_spd(speed=body.speed[0],accel=body.jlist[
    jdex])
311
312   for body in rk_bodies:
313     total = body.jlist[0] + body.jlist[1]*2 + body.jlist[2]*2 + body.jlist[3]
314     total = total*step/6
315     if not body.fixed:
316       body.locat[0] = body.locat[0] + total
317
318 # Bodies
319 G = 10
320 b1 = Body(name="Stella", mass=1000, loc=vector(0,0,0), spd=vector(0,0,0), clr=color
    .blue)
321 b2 = Body(name="Pianeta", mass=500, loc=vector(1000,0,0), spd=vector(0,sqrt(15),0),
     clr=color.red)
322 b3 = Body(name="Satellite", mass=1, loc=vector(1000,0,200), spd=vector(0,0,0), clr=
    color.green)
323 # System
324 s1 = System(bodies=[b1,b2,b3])
325
326 # Axes
327 for i in range(len(scenes)):
328   asse_x = arrow(canvas=scenes[i], axis=vector(500,0,0), color=color.black,
      shaftwidth=1)
329   asse_y = arrow(canvas=scenes[i], axis=vector(0,500,0), color=color.black,
      shaftwidth=1)
330   asse_z = arrow(canvas=scenes[i], axis=vector(0,0,500), color=color.black,
      shaftwidth=1)
331   assi = [asse_x,asse_y,asse_z]
332
333 # Graphics Bodies
334 body_render = []
335 for i in range(len(scenes)):
```

```
336   body_render.append([])
337   for body in s1.bodies:
338     body_render[i].append(sphere(canvas=scenes[i], pos=vector(0,0,-1), radius=body.
      rd,
339     color=body.clr, make_trail=True, trail_type='points', interval=200, retain
      =1000))
340
341 angular_momentum_render = []
342 for i in range(len(scenes)):
343   angular_momentum_render.append([])
344   for body in s1.bodies:
345     angular_momentum_render[i].append(arrow(canvas=scenes[i], color=color.green,
      shaftwidth=3))
346   angular_momentum_render[i].append(arrow(canvas=scenes[i], color=color.green,
      shaftwidth=3))
347
348 # Trigger iniziale per dati aggiuntivi
349 print(f"G: {G}")
350 print(f"step: {step}")
351
352 # Cattura schermata && Dati
353 def screenshot():
354   s1.update_self()
355   s1.relocing()
356   # Catture
357   scene1.capture('sistema0.png')
358   scene2.capture('sistemaG.png')
359   scene3.capture('sistemaS.png')
360
361   # Stampe
362   print(time) # Istante
363   for body in s1.bodies:
364     print(body.name)
365     print(body.locat)
366     print(body.speed)
367     print(body.momentum)
368     print(body.angular_momentum)
369   print("Sistema:")
370   print(s1.position)
371   print(s1.speed)
372
373 # Loop
374 while True:
375   rate(step_rate)
376   time += step
377   runge_kutta(rk_bodies=s1.bodies)
378   s1.next_step()
379   for i in range(len(scenes)):
380     for j in range(len(body_render[i])):
381       body_render[i][j].pos = s1.bodies[j].locat[i]
382       if render_vettori:
383         angular_momentum_render[i][j].pos = s1.bodies[j].locat[i]
384         angular_momentum_render[i][j].axis = s1.bodies[j].angular_momentum[i]/
385           modulo(s1.bodies[j].angular_momentum[i])*s1.bodies[j].rd*4
386     if render_vettori:
387       angular_momentum_render[i][-1].axis = s1.angular_momentum[i]/modulo(s1.
      angular_momentum[i])*100
388
389   # Aggiornamento continuo del grafico
390   curva_x.plot(time, s1.angular_momentum[0].x)
391   curva_y.plot(time, s1.angular_momentum[0].y)
392   curva_z.plot(time, s1.angular_momentum[0].z)
393   curva_E.plot(time, s1.energy[0])
394
```

```
395   # Trigger automatico per Screenshot
396   if time >= screen_times[0]:
397     screen_times.pop(0)
398     screenshot()
```

Bibliografia

[1] K. G. Stassun, *A tilted "Tatooine planet" whose two suns aren't stars at all*, Sci. Adv. 11, eadx3902 (2025).

[2] T.A. Baycroft, L. Sairam, A.H.M.J. Triaud, A.C.M. Correia, *Evidence for a polar circumbinary exoplanet orbiting a pair of eclipsing brown dwarfs*, Sci. Adv. 11, eadu0627 (2025).

[3] F. De Vito, *Il problema dei tre corpi*, Lavoro di Maturità in Fisica ed Informatica, supervisionato da I. Arini e T. Corridoni, Liceo di Bellinzona, 2026.

[4] N.R. Hanson, *I modelli della scoperta scientifica*, Feltrinelli (1978).

[5] T.S. Kuhn, *La rivoluzione copernicana*, Einaudi (1972).

[6] M. Gardner, *Fads and Fallacies in the name of science*, Dover (1956).

[7] *Le trame concettuali delle discipline scientifiche*, a cura di G. Cortini, La Nuova Italia, (1985).

[8] E. Morin, *On Complexity*, Cresskill-New Jersey, Hampton Press Inc. (2008).

[9] Profilo di Competenza del Docente, `https://www4.ti.ch/decs/ds/sims/docenti/profilo-docente`

[10] T. Corridoni, *Le due biblioteche. Dal metodo scientifico a una didattica che costruisca metodi*, Scuola Ticinese 341, 45 (2021), e referenze interne.

[11] D. Hestenes, *Notes for a Modeling Theory of Science, Cognition and Instruction*, Proceedings of the 2006 GIREP conference: Modelling in Physics and Physics Education, `https://www.researchgate.net/publication/253847244_Notes_for_a_Modeling_Theory_of_Science_Cognition_and_Instruction`.

[12] A. Kopff, *I fondamenti della relatività einsteiniana. Valore e interpretazione della teoria*, Ulrico Hoepli, Milano (1923).

[13] G. Keller, *Vergleich von Kepler- und Kreisbahn: Eine Erzängung*, Bulletin 158, 28, VSMP, SSPMP, SSIMF (2025).

[14] L. Landau, E. Lifchitz *Mécanique*, MIR, Moscou (1966).

[15] S. Strelkov, *Mécanique*, MIR, Moscou (1978).

[16] I.N Bronstein, K.A. Semendjajew, G. Musiol, H. Mühlig, *Taschenbuch der Mathematik*, Harri Deutsch Verlag, Frankfurt am Main (2005).

[17] M. Monaci, M., *Il problema dei 3 corpi: generalizzazione in 3D al caso ellittico con perturbazioni esterne risolto tramite metodo di Runge-Kutta al quarto ordine*, Corso di Sistemi Planetari - Prof. P. Paolicchi, 2018.

[18] M. Valtonen, H. Karttunen, *The three-body problem*, Cambridge University Press (2006).

[19] M. Folkerts, D. Launert, A. Thom, *Jost Bürgi's Method for Calculating Sines*. https://arxiv.org/abs/1510.03180 (Oct 2015).

[20] D. Roegel, *Bürgi's "Progress Tabulen" (1620): logarithmic tables without logarithms*, https://locomat.loria.fr/buergi1620/buergi1620doc.pdf.

[21] K. Kanatani, *3D Rotations: Parameter Computation and Lie Algebra based Optimization*, Chapman and Hall, CRC (2020).

[22] H.P. Schecker, *Physik-Modellieren. Grafikorientierte Modellbildungssysteme im Physikunterricht*, Klett Verlag Gmbh, Stuttgart (1998).

[23] D.H. Meadows, D.L. Meadows, J. Randers, W.W. Behrens III, *The Limits to Growth*, Universe Book, New York (1972); https://donellameadows.org/the-limits-to-growth-now-available-to-read-online/.

[24] https://en.wikipedia.org/wiki/Fermi%E2%80%93Pasta%E2%80%93Ulam%E2%80%93Tsingou_problem; https://web.physics.utah.edu/~detar/phys6720/handouts/fpu/FermiCollectedPapers1965.pdf.

[25] E.D. Lorenz *Deterministic non periodic flow*, Journal of the Atmospherical Science, 20, 130 (1963) https://journals.ametsoc.org/view/journals/atsc/20/2/1520-0469_1963_020_0130_dnf_2_0_co_2.xml.

[26] https://iseesystems.com/store/products/stella-online.aspx.

[27] https://berkeley-madonna.myshopify.com/.

[28] https://vensim.com/download/.

[29] https://stochsd.sourceforge.io/homepage/#/.

[30] https://insightmaker.com.

[31] K. N. Anagnostopoulos, *Computational Physics*, National Technical University of Athens, Athens (2016).

[32] D. Acheson, *From calculus to chaos*, Oxford University Press, Oxford UK (1997).

[33] L. Liu, *Algorithms for Satellite Orbital Dynamics*, Springer, NUP (2023).

[34] H. Poincaré, *Sue le problème des trois corpes et les équations de la dynamique*, Acta mathematica, XIII, pp.1-270 (1890).

[35] H. Dang-Vu, C. Delconte, *Bifurcations et chaos*, Ellipses, Paris (2000).

[36] M. Zanlorenzi, *Soluzioni triangolari nel problema ristretto ellittico dei 3 corpi*, Tesi di laurea (Relatore, Prof. Francesco Fassò) Università degli studi di Padova, Padova, (2010).

[37] M. Giancola, *Il problema ristretto dei tre corpi*, Matematicamente.it Magazine, n.19, pag. 41, Aprile (2013).