

MÈTODES NUMERICIS

Grau en Matemàtiques i itinerari doble Matemàtiques–Física, 2025–2026.

Pràctica 3

En aquesta pràctica calcularem una aproximació de la trajectòria de flyby lunar de la missió Artemis II. Els mètodes numèrics que emprarem són el mètode de Newton per trobar zeros de funcions i el mètode d'interpolació de Neville. Farem servir “com a caixa negra” una funció que propaga trajectòries.

1 Model dinàmic

La missió Artemis II és una missió tripulada de la NASA que ha estat completada recentment, en la que 4 astronautes han fet una passada a prop de la lluna (lunar flyby). Ha estat la primera vegada des de l'època de les missions Apollo en què éssers humans han anat més enllà de l'òrbita terrestre. L'objectiu d'aquesta pràctica és obtenir una aproximació de la trajectòria lunar emprada en aquesta missió.

El model dinàmic que emprarem és el *Problema Restringit de Tres Cossos* (en anglès *Restricted Three-Body Problem* o RTBP), que descriu el moviment d'un cos de massa negligible (la nau espacial Orion, en el nostre cas) sota l'atracció gravitatòria de dos cossos (la terra i la lluna, en el nostre cas), que se suposen en moviment circular uniforme un al voltant de l'altre. Aquests dos cossos s'anomenen *primaris*. Per tal de simplificar les equacions del moviment, es fan dues suposicions:

1. Es canvia a *unitats adimensionals*, en les quals la distància entre els primaris es manté constant igual a 1 i els primaris completen una revolució al voltant de l'altre en 2π unitats de temps.
2. Es pren un sistema de coordenades amb origen al centre de masses dels primaris i que gira solidàriament amb ells, de manera que, en aquest sistema, els primaris tenen posicions fixes a l'eix z_1 (eqs. (4) i (5)).

Suposant que el moviment del cos de massa negligible està confinat al pla de rotació dels primaris (parlem llavors del “RTBP pla”), aquest moviment ve donat per les solucions $(x(t), y(t))$ del següent sistema d'equacions diferencials:

$$\begin{aligned}\dot{z}_1 &= z_3, & \dot{z}_3 &= 2z_4 + \partial_{z_1}\Omega(z_1, z_2), \\ \dot{z}_2 &= z_4, & \dot{z}_4 &= -2z_3 + \partial_{z_2}\Omega(z_1, z_2),\end{aligned}\tag{1}$$

on

$$\begin{aligned}\Omega(z_1, z_2) &:= \frac{1}{2}(z_1^2 + z_2^2) + \frac{1-\mu}{\rho_1} + \frac{\mu}{\rho_2} + \frac{1}{2}\mu(1-\mu), \\ \rho_1 &:= ((z_1 - \mu)^2 + z_2^2)^{1/2}, \\ \rho_2 &:= ((z_1 - \mu + 1)^2 + z_2^2)^{1/2}.\end{aligned}\tag{2}$$

Al vector d'estats (z_1, z_2, z_3, z_4) , les coordenades (z_1, z_2) són posicions i les coordenades (z_3, z_4) són velocitats. Com veureu (o heu vist) al curs d'equacions diferencials, donada una condició inicial $(z_1^0, z_2^0, z_3^0, z_4^0)$, existeix una única solució $(z_1(t), z_2(t), z_3(t), z_4(t))$ de les

$$\begin{array}{ll}
a_{\oplus\zeta} = 1.215058560962404\text{e} - 2 & R_{\zeta} = 1737.5 \\
T_{\oplus\zeta} = 27.321577 & R_{\oplus} = 6371.00
\end{array}$$

Taula 1: Valors d'algunes constants

equacions (1) que satisfaci $(z_1(t), z_2(t), z_3(t), z_4(t)) = (z_1^0, z_2^0, z_3^0, z_4^0)$. Les equacions (1) depenen d'un únic paràmetre, anomenat *paràmetre de masses*, que és la massa del primari petit dividida per la suma de masses dels primaris. En el nostre cas,

$$\mu := \frac{m_{\zeta}}{m_{\oplus} + m_{\zeta}} = 1.215058560962404\text{e}-2 \quad (3)$$

Si anomenem ul la unitat de longitud i ut la unitat de temps de les equacions (1), d'acord amb les suposicions esmentades abans,

$$1 \text{ ul} = a_{\oplus\zeta} \text{ km}, \quad 2\pi \text{ ut} = T_{\oplus\zeta} \text{ dies},$$

on les constants $a_{\oplus\zeta}$, $T_{\oplus\zeta}$ estan llistades a la taula 1. També les trobareu definides als fitxers `constants.h` i `initraj.gnu`. D'acord amb les suposicions mencionades anteriorment, l'estat de la lluna és fix (no depèn del temps) igual a

$$(\mu - 1, 0, 0, 0), \quad (4)$$

mentre que el de la terra és fix igual a

$$(\mu, 0, 0, 0, 0) \quad (5)$$

Per tal de propagar trajectòries amb el model anterior, disposeu d'una funció anomenada `proptraj()` (“propagar trajectòria”), que és al mòdul `rtbp.c`. El seu prototipus és:

```
void proptraj (double mu, int nt, double *z0, double *zf, double *du,
              double pmax, FILE *fp);
```

Donat un estat inicial $z^0 = (z_1^0, z_2^0, z_3^0, z_4^0)$ dins un vector de 4 components amb base apuntada per l'argument `z0`, aquesta funció propaga la trajectòria corresponent fins a un estat final $z^f = (z_1^f, z_2^f, z_3^f, z_4^f)$ definit pel fet de ser el `nt`-èsim tall amb l'eix horitzontal ($z_2 = 0$). Els arguments de `proptraj()` són:

- `mu` (entrada): paràmetre μ (eq. (3)).
- `nt` (entrada): nombre de talls fins al qual es propaga la trajectòria, temps endavant si `nt` > 0, o temps endarrere si `nt` < 0.
- `z0` (entrada): apunta a la base d'un vector amb la condició inicial $z^0 = (z_1^0, z_2^0, z_3^0, z_4^0)$ de la trajectòria a propagar.
- `zf` (sortida, opcional): si `zf != NULL`, es guarda al vector amb base apuntada per `zf` l'estat $z^f = (z_1^f, z_2^f, z_3^f, z_4^f)$ del `nt`-èsim punt de tall amb l'eix horitzontal ($z_2^f = 0$ llevat d'errors numèrics).
- `du` (sortida, opcional): si `du != NULL`, es guarden a `du[0]` (respectivament `du[1]`, `du[2]`, `du[3]`) les derivades de z_3^f (component horitzontal de la velocitat del `nt`-èsim tall) respecte de z_1^0 (respectivament z_2^0 , z_3^0 , z_4^0).

- **pmax** (entrada): paràmetre tècnic per limitar en valor absolut el pas màxim emprat en la integració numèrica. S'ha de disminuir si, en representar trajectòries, es veuen poligonals (v. el següent argument).
- **fp** (sortida): si **fp**!=NULL, s'entèn que apunta a un fitxer obert al qual es bolca una línia per cada punt obtingut a la integració numèrica, amb columnes: t , z_1 , z_2 , z_3 , z_4 .

2 Disseny de la trajectòria lunar

2.1 Càlcul de la trajectòria

La missió Artemis II no ha arribat a orbitar la lluna, sinó que ha fet una passada propera (flyby) seguint el que es coneix com a “trajectòria de retorn lliure” (*free-return*): una trajectòria que, sense necessitat d'engegar cap motor, passa per darrere de la lluna, retorna cap a la terra i acaba fent una reentrada a l'atmosfera.

No ha estat el cas a la missió Artemis II, però, per simplificar, suposarem que, com es feia a les missions Apollo, abans d'agafar una trajectòria de retorn lliure cap a la lluna, la nau està en òrbita circular (“òrbita d'aparcament”) al voltant de la terra. Triem l'alçada d'aquesta òrbita sobre la superfície com a 175 km. Això correspon a una distància al centre de la terra de

$$r_0 := 0.01701377045867686 \text{ ul.} \quad (6)$$

El que volem fer és engegar els motors a un punt de l'òrbita d'aparcament per tal d'agafar una trajectòria de retorn lliure cap a la lluna. El problema és determinar el punt on ho hem de fer. És conegut que, degut a una simetria de les equacions diferencials (1), el RTBP pla satisfà la següent propietat:

$$(z_1(t), z_2(t), z_3(t), z_4(t)) \text{ solució} \implies (z_1(-t), -z_2(-t), -z_3(-t), z_4(-t)) \text{ solució.}$$

Aquesta propietat ens dóna la manera de trobar el punt que busquem: si trobem un punt de l'òrbita d'aparcament en el qual, en pujar la velocitat, la nova trajectòria arriba al darrere de la lluna tallant perpendicularment l'eix horitzontal ($z_2 = 0$), la trajectòria simètrica serà la mateixa trajectòria, i ens portarà de tornada cap a l'òrbita d'aparcament. Serà per tant una trajectòria de retorn lliure. Vegeu la figura 1 esquerra.

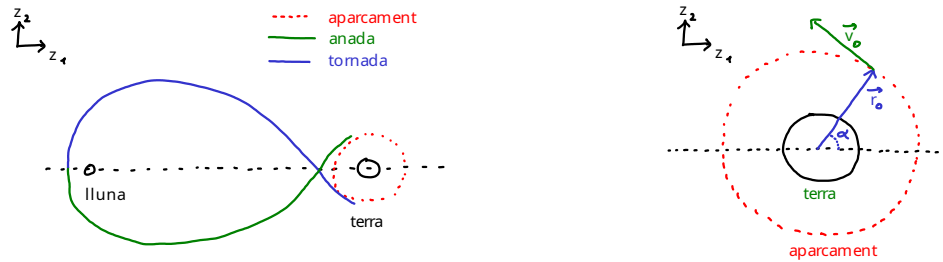


Figura 1: Esquerra: ús de la simetria per obtenir una trajectòria de retorn lliure. Dreta: cerca del punt d'entrada a una trajectòria de retorn lliure.

Suposem que, per altres mitjans, s'ha determinat que, per tal d'arribar al darrere de la lluna a uns 6000 km sobre la seva superfície des de l'òrbita d'aparcament, la velocitat s'ha d'incrementar a

$$v_0 := 10.67764174790536 \text{ ul/ut.} \quad (7)$$

D'acord amb tot el que hem dit, trobar el punt de partida de l'òrbita d'aparcament per tal d'agafar la trajectòria de retorn lliure es redueix a buscar un zero α_* de la funció

$$\alpha \longmapsto u(z^m(r_0, v_0, \alpha)),$$

on:

- $z^m(r_0, v_0, \alpha) = (z_i^m(r_0, v_0, \alpha))_{i=1}^4$ és el vector d'estats que conté la posició i la velocitat del punt de partida de l'òrbita d'aparcament. D'acord amb la figura 1 dreta,

$$\begin{aligned} z_1^m(r_0, v_0, \alpha) &= \mu + r_0 \cos \alpha, & z_3^m(r_0, v_0, \alpha) &= -v_0 \sin \alpha, \\ z_2^m(r_0, v_0, \alpha) &= r_0 \sin \alpha, & z_4^m(r_0, v_0, \alpha) &= v_0 \cos \alpha. \end{aligned}$$

La funció z^m la podeu avaluar mitjançant una crida a la funció `rva2z()` del mòdul `rtbp.c`.

- Donat un estat inicial $z = (z_1, z_2, z_3, z_4)$, la funció $u(z)$ està definida com la **coordenada horitzontal** de la velocitat del segon tall amb l'eix horitzontal de la trajectòria resultant de propagar z .

La funció u la podeu avaluar mitjançant una crida a `proptraj()` del mòdul `rtbp.c`: és la tercera component del vector d'estats **zf**. La diferencial de la funció u respecte de l'estat inicial (**el que entreu a l'argument z0**) també us la torna la funció `proptraj()`. Ho fa al vector amb base apuntada per **du**.

Amb tot el que hem dit, podem trobar un zero α_* de la funció $\alpha \mapsto u(z^m(r_0, v_0, \alpha))$, que determina un punt d'entrada a una trajectòria de retorn lliure d'acord amb la figura 1 dreta. Quan ho fareu, veureu que, amb els valors numèrics de r_0 i v_0 donats a (6) i (7), la trajectòria corresponent passa per darrere de la lluna amb una distància mínima a la superfície de 6046.558 km. Diem que l'*alçada del periseleni* de la trajectòria de retorn lliure és $h_{P_\zeta} = 6046.558$ km. Voldríem obtenir una trajectòria de retorn lliure amb alçada de periseleni de 6545 km, com l'emprada a la missió Artemis II.

Per tal de fer-ho, meditant sobre tot el que hem fet fins ara ens adonem que tot plegat es pot interpretar com que, donats r_0 i v_0 (amb valors numèrics donats per les equacions (6), (7)), hem trobat α tal que

$$u(z^m(r_0, v_0, \alpha)) = 0.$$

Aquesta expressió defineix α com a funció implícita de r_0, v_0 . De fet, no ens interessa variar r_0 , perquè llavors canviariem l'alçada de l'òrbita d'aparcament. Amb r_0 fixada, l'expressió anterior també defineix α com a funció implícita de v_0 . Anem a denotar $\alpha = \alpha(v_0)$. Variant lleugerament el valor (7) de v_0 , podem refer tot el càlcul i avaluar així la funció “alçada de periseleni de l'òrbita de retorn lliure com a funció de v_0 ”,

$$v_0 \longmapsto h_{P_\zeta}(r_0, v_0, \alpha(v_0)).$$

Si, fent proves amb valors $\{v_0^{(i)}\}_{i=0}^n$ propers al de (7), aconseguim que $6545 \in \langle h_{P_\zeta}(r_0, v_0, \alpha(v_0)) \rangle_{i=0}^n$, podrem determinar el valor v_0^* tal que $h_{P_\zeta}(r_0, v_0, \alpha(v_0^*)) = 6545$ per interpolació inversa.

2.2 Representació gràfica

Per tal de representar gràficament les trajectòries que obtingueu, disposeu d'una utilitat anomenada `shtmn` (“shoot the Moon”), que respon a la crida

```
./shtmn pmax fitxtrj
```

Aquesta utilitat llegeix per standard input línies de la forma

$$r_0 \quad v_0 \quad \alpha \quad n_t$$

Qualsevol d'aquestes quatre quantitats es pot substituir per un guió (-), i llavors **shtmn** entén que voleu fer servir el valor que heu entrat a la línia anterior. Per cadascuna d'aquestes línies, la utilitat **shtmn** propaga la trajectòria corresponent a l'estat inicial $z^m(r_0, v_0, \alpha)$ fins al n_t -èsim tall amb l'eix horitzontal, tot bolcant els punts que obté al fitxer **fitxtrj** (vegeu l'argument **fp** de la descripció de la funció **proptraj()** més amunt). També, per cada línia de standard input com l'anterior, la utilitat **shtmn** escriu per standard output la línia

$$\alpha \quad u(z^m(r_0, v_0, \alpha)) \quad h_{\zeta}(r_0, v_0, \alpha)$$

on $h_{\zeta}(r_0, v_0, \alpha)$ és la distància en km des del darrer punt de tall amb l'eix horitzontal fins a la superfície de la lluna. Aquesta distància es pren positiva si la trajectòria acaba tallant l'eix horitzontal a l'esquerra de la lluna (el darrere, vista des de la terra), o negativa si el tall és a la dreta.

Per exemple, si entrem a la terminal

```
./shtmn .05 trajsex.txt
0.02138186341586538 9.5 0.5 2
- - 0.6 -
- - 0.7 -
- - 0.8 -
- - 0.9 -
- - 1.0 -
[Ctrl-D]
```

i, després, dins **gnuplot**, entrem

```
load "initraj.gnu"
plot "trajsex.txt" u 2:3 w l
```

obtindrem les trajectòries de la figura 2.

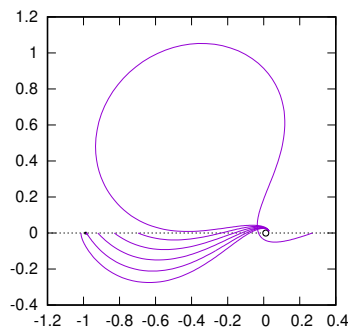


Figura 2: Trajectòries d'exemple generades amb la utilitat **shtmn**.

3 Software a lliurar

3.1 Biblioteca

A un mòdul anomenat **nwtnev.c** heu d'escriure les dues funcions següents:

1. Una funció amb prototipus

```
int newton (double *x, double tolf, double tolx, int maxit,  
           void (*fdf)(double,double*,double*,void*), void *prm, int ixrr);
```

Per a una funció $f : I \rightarrow \mathbb{R}$, on I interval, i per a un valor $x_0 \in I$, guardat a `*x`, que compleix $f(x_0) \approx 0$, la funció `newton()` ha d'aplicar el mètode de Newton fins que la funció tingui valor absolut $\leq \text{tolf}$ o fins que la distància entre dos iterats consecutius sigui $\leq \text{tolx}$. Si es compleix una d'aquestes dues condicions, `newton()` ha de parar i retornar el nombre d'iterats que ha fet. Si en `maxit` iterats no s'ha complert cap de les dues condicions anterior, la funció ha de retornar `-1`. L'argument `fdf` és un apuntador a funció. La funció apuntada per ell, suposant que se li passa un valor x com a primer argument, ha de guardar $f(x)$ a la variable apuntada pel segon argument i $f'(x)$ a la variable apuntada pel tercer argument. La funció `newton()` ha de passar `prm` com a quart argument a la funció apuntada per `f` cada vegada que la crida: se suposa que apunta a uns paràmetres arbitraris que `f` necessita. L'argument `ixrr` és l'“índex de xarrera”: si és 0, `newton()` no ha d'escriure res mentre calcula. Si és > 0 , `newton()` ha d'informar per `stderr` sobre l'evolució del mètode de Newton.

2. Una funció amb prototipus

```
double intnev (int n, double xi[], double yi[], double x);
```

que ha de retornar el valor a `x` del polinomi interpolador de grau `n` amb punts de suport $\{(xi[i], yi[i])\}_{i=0}^n$. L'ha d'avaluar mitjançant l'algorisme de Neville.

3.2 Utilitats

Heu d'escriure les següents utilitats:

1. Dins un mòdul `detcsig.c`, una utilitat que respongui a la crida

```
./detcsig r0 v0 alf0 alff nalf nt pmax
```

Per a cada valor $\alpha_i = \alpha_0 + i(\alpha_f - \alpha_0)/n_\alpha$, on α_0 és `alf0`, α_f és `alff` i n_α és `nalf`, la utilitat `detcsig` ha d'escriure una línia amb les quantitats

$$\alpha_i \quad u(z^m(r_0, v_0, \alpha_i)) \quad h_{P_\zeta}(r_0, v_0, \alpha_i)$$

separades per espais. Els arguments `nt` i `pmax` de la utilitat són per passar a la funció `proptraj()`. La quantitat $h_{P_\zeta}(r_0, v_0, \alpha_i)$ s'ha d'escriure en punt fix amb 3 decimals.

2. Dins un mòdul `trobalf.c`, una utilitat que respongui a la crida

```
./trobalf tolu maxit pmax ixrr
```

Aquesta utilitat ha de llegir per standard input línies de la forma

$$r_0 \quad v_0 \quad \alpha_0 \quad n_t$$

on r_0, v_0, α_0 són tals que $u(z^m(r_0, v_0, \alpha_0)) \approx 0$. Per cadascuna d'aquestes línies, la utilitat `trobalf` ha de fer servir el mètode de Newton en α per tal d'aconseguir $|u(z^m(r_0, v_0, \alpha))| < \text{tolu}$. Els arguments `maxit` i `ixrr` de la utilitat són per a passar a la funció `newton()`. L'argument `pmax` ha d'arribar a `proptraj()` via l'argument `prm` de `newton()`.

Per cada línia llegida per standard input, la utilitat `trobal` ha d'escriure per standard output una línia de la forma

$$r_0 \quad v_0 \quad \alpha \quad h_{P_{\mathbb{C}}}(r_0, v_0, \alpha)$$

on α és l'obtinguda a partir de α_0 mitjançant el mètode de Newton i $h_{P_{\mathbb{C}}}(r_0, v_0, \alpha)$ és l'alçada del periseleni tal com l'hem denotada abans.

3. Dins d'un mòdul `interp.c`, una utilitat que respongui a la crida

```
./interp n
```

Aquesta utilitat ha de llegir per standard input $n + 1$ línies de la forma

$$x_i \quad y_i$$

A continuació ha d'anar llegint valors x i, per cadascun d'ells, escriure per standard output una línia

$$x \quad P(x)$$

on P és el polinomi interpolador amb punts de suport $\{(x_i, y_i)\}_{i=0}^n$. El valor $P(x)$ l'ha d'obtenir mitjançant una crida a `neville()`.

4 Etapes de desenvolupament

1. Escriviu la funció `newton()` de la biblioteca.
2. Valideu la funció `newton()` mitjançant la utilitat `proves_nwt_exp.c` que trobareu al campus virtual. Aquesta utilitat busca, mitjançant el mètode de Newton, un zero de la funció $f(x) = e^x - 2$, que sabem que és $\ln 2$. Podeu fer servir els paràmetres de la següent taula:

x	tolf	tolx	maxit	ixrr
2	1e-8	1e-8	10	1
2	1e-8	-1	10	1
2	-1	1e-8	10	1
10	1e-12	1e-8	10	1
10	1e-12	1e-8	20	1

Heu d'explicar a la memòria com es comporta `newton()` amb aquests paràmetres i per què aquest comportament és correcte.

3. Escriviu la utilitat `detcsig`, i executeu

```
./detcsig 0.01701377045867686=r0 10.67764174790536=v0 \
0.95=alf0 1.25=alf1 100=nalf 2=nt .05=pmax > csig.txt
```

(podeu escriure tota la comanda en una línia sense el `\`). Després d'això, executeu dins `gnuplot`

```
set y2tics
plot "csig.txt" u 1:2 w l, "" u 1:3 ax x1y2 w l
```

Podeu prémer **g** a la finestra del gràfic per a activar/desactivar la graella, i **r** per a activar/desactivar el regle. Observareu que detectem dos zeros de $\alpha \mapsto u(z^m(r_0, v_0, \alpha))$. El primer correspon a una alçada del periseleni d'uns 6000 km, mentre que l'alçada de l'altre és d'uns 30000 km. Ens interessa el primer zero. Feu zooms amb el botó dret del ratolí per tal de trobar-ne una bona aproximació. Per saber fins a on té sentit fer zooms, us pot ajudar redibuixar el gràfic amb línies i punts:

```
plot "csig.txt" u 1:2 w lp, "" u 1:3 ax x1y2 w lp
```

Anomenem α_0 l'aproximació del zero de $\alpha \mapsto u(z^m(r_0, v_0, \alpha))$ que heu trobat.

4. Escriviu la utilitat **trobalf**. Feu-la servir per tal de trobar α^* a prop de α_0 tal que

$$|u(z^m(r_0, v_0, \alpha^*))| \leq 10^{-10}.$$

Observareu que la corresponent trajectòria de retorn lliure té una alçada de periseleni de 6046.558 km.

Representeu amb la utilitat **shtmn** la trajectòria de retorn lliure que heu obtingut.

5. Escriviu la funció **neville()** de la biblioteca i la utilitat **interp**. Feu els testos que considereu adients per comprovar que funcionen correctament.
6. A continuació, com està dit abans, volem ajustar v_0 per tal d'aconseguir una alçada de periseleni de 6545 km. Modifiqueu v_0 (amb delicadesa: esteu fent vol espacial!) i executeu **trobalf** per tal d'aconseguir valors d'alçada de periseleni que envoltin 6545. Anomenem $\{v_0^{(i)}\}_{i=0}^n$ els valors de v_0 modificats (feu servir $n = 4$ o $n = 5$). Per tal de minimitzar l'error d'interpolació, convé que 6545 sigui proper al centre de $\langle h_{P_{\mathcal{C}}}^{(i)} \rangle_{i=0}^n$, on $h_{P_{\mathcal{C}}}^{(i)}$ denota l'alçada de periseleni obtinguda a partir de $v_0^{(i)}$.

Un cop hagueu determinat els valors $\{v_0^{(i)}\}_{i=0}^n$, suposant que el fitxer **trobalf.txt** és un fitxer d'entrada per **trobalf** que té els valors $\{v_0^{(i)}\}_{i=0}^n$ com a segona columna, podeu generar la taula d'interpolació inversa mitjançant la comanda¹

```
./trobalf 1e-10=tolu 10=maxit .05=imax 0=ixrr < trobalf.inp \
| awk '{ print $4,$2 }'
```

(com abans, la podeu entrar tota en una línia sense el caràcter \).

7. Feu servir **interp** a partir de la taula d'interpolació anterior per determinar la velocitat v_0^* que dóna lloc a una trajectòria de retorn lliure amb alçada de periseleni 6545.000 km. Si no ho aconseguíu, repetiu el pas 6 però amb valors de $v_0^{(i)}$ més acumulats al voltant del valor de v_0^* que heu trobat per interpolació inversa.
8. (Repte opcional) Us atreviu a rebaixar l'alçada del periseleni a 111 km, que és la que va fer servir l'Apollo 11?

5 Lliurament

Dins el termini de lliurament d'aquesta pràctica, heu de lliurar al campus virtual un únic fitxer **NIA.zip**, on **NIA** és el vostre **NIA**, que contingui la biblioteca **nwtnev.c**, les utilitats **detcsig.c**, **trobalf.c**, **interp.c** i un fitxer anomenat **memoria.pdf** amb la memòria. Afegiu-hi també el codi que hagueu escrit per fer validacions.

¹Fent copy-paste des del pdf pot ser que no us funcioni per culpa dels caràcters apòstrof.