6. november 2012/pch

02601 Introduktion til numeriske algoritmer — Efterår 2012

3. obligatoriske opgave

Rapporten skal uploades som **pdf-fil** på CampusNet senest
fredag den 7. december, klokken 24:00 (dvs midnat mellem fredag og lørdag).

`http://www.youtube.com/watch?v=eCMmmEEyOO0`

Inspirationen til denne opgave kommer fra en video Slinky Drop på YouTube, der viser hvordan en "slinky" – altså en stor fjeder – falder til jorden når man slipper den. Først trækker den sig sammen, således af den *nederste* del af fjederen står stille i luften, og derefter falder fjederen samlet mod jorden.

Vores opgave går ud på at lave en simulering af dette "slinky drop" som bl.a. kan bruges til at vise, at denne opførsel er uafhængig af fjederkonstanten.

Undervejs i arbejdet med at løse denne opgave skal vi, i lighed med første og anden obligatoriske opgave, også træne nogle vigtige teknikker og beregningsmetoder fra *numerisk analyse*. Besvarelsen af disse spørgsmål træner de teknikker og metoder, som skal bruges til at designe og og analysere vores algoritmer – vi hjælper med disse træningsopgaver, mens opgaverne knyttet til algoritme-design skal løses selvstændigt.

Rapporten skal indeholde besvarelser både på spørgsmål 1–6 (som omhandler de numeriske metoder) og opgave **A**, **B** og **C** (der bl.a. omhandler "slinky drop"). Vedlæg Matlab-kode til alle relevante spørgsmål og opgaver i appendix.

Nedenfor står 6 spørgsmål, som træner de numeriske metoder der skal bruges til at løse opgaverne.

- Spørgsmål 1–2 skal besvares, inden man kan lave opgave **A**.
- Spørgsmål 3–5 skal besvares, inden man kan lave opgave **B**.
- Spørgsmål 6 skal besvares, inden man kan lave opgave **C**.

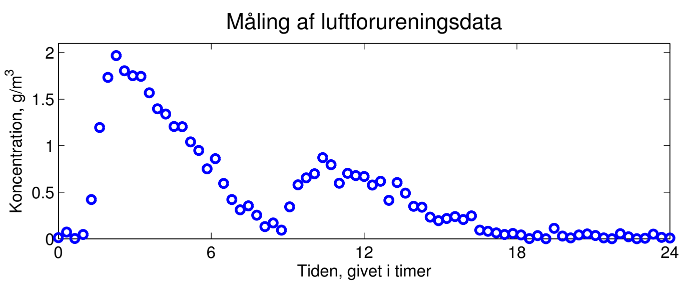
Spørgsmål 1. Optag af luftforurening (1½ timer)

Formålet med dette spørgsmål er at introducere et problem der involverer numerisk beregning af et integrale, og vise hvor nemt det er at bruge Matlabs indbyggede funktion `quad`.

Jod-isotopen ^{131}I forekommer i store mængder efter udslip på et atomkraftværk. Den forekommer på gas-form og optages i blodet ved indånding, og den ophobes primært i skjoldbugskirtlen. Vi skal beregne hvor meget jod der optages fra luften ved et tænkt uheld, hvor ^{131}I fortyndes i atmosfæren og flyver forbi København. Hvis vi antager at koncentrationen af ^{131}I i luften, som funktion af tiden t, er given ved funktionen $f(t)$ – målt i g/m^3 – så kan vi modellere det samlede optag i blodet, over hele perioden fra $t = 0$ til $t = T = 24$ timer – ved dette integral:

$$I = \ell \int_0^T f(t)\, dt \;, \qquad (1)$$

hvor $\ell = 1\,\text{m}^3/\text{time}$ er luftintaget i lungerne.



Over perioden på 24 timer findes der ialt $n = 75$ fejlbehæftede målinger (t_i, y_i) af koncentrationen af ^{131}I for $i = 1, \ldots, n$, se figuren ovenfor. Disse data ligger på CampusNet i filen `airpollutiondata.mat`.

1.1) Beregn en grov tilnærmelse I_{grov} til optaget, dvs integralet I i (1), ved at bruge formlen

$$I_{\text{grov}} = \Delta \sum_{i=1}^{n} y_i \;, \qquad \Delta = T/n \;. \qquad (2)$$

Vi kan beregne et bedre resultat ved at fitte en god model til måledata, og derefter beregne integralet af denne funktion. I denne opgave skal vi bruge modellen

$$\begin{aligned} f(t) \;=\; & a_1 \exp\left(-\left(\frac{t-t_1}{\sigma_1}\right)^2\right) \Big(1 + \text{erf}\big(d_1\,(t-t_1)\big)\Big) + \\ & a_2 \exp\left(-\left(\frac{t-t_2}{\sigma_2}\right)^2\right) \Big(1 + \text{erf}\big(d_2\,(t-t_2)\big)\Big) \end{aligned} \qquad (3)$$

hvor funktionen "erf" er indbygget i Matlab.

1.2) Brug teknikkerne fra den anden obligatoriske opgave til at fitte funktionen $f(t)$ i (3) til data. Som startgæt på parametrene kan du bruge:

$$a_1 = a_2 = 1, \qquad t_1 = 2, \quad t_2 = 8, \qquad \sigma_1 = \sigma_2 = 4, \qquad d_1 = d_2 = 1 \;. \qquad (4)$$

Plot fittet sammen med data og angiv de fundne værdier af alle parametrene.

1.3) Brug Matlab-funktionen `quad` til at beregne integralet I (du skal få en værdi som større end I_{grov}). Brug `doc quad` til at få hjælp om brugen af `quad`.

Spørgsmål 2. Kondiløberen ($1^1/_2$ timer)

Vi betragter en kondiløber, hvis rute som funktion af tiden kan beskrives ved koordinaterne $\big(x(t), y(t)\big)$. Den totale distance D, når man løber i tidsrummet fra $t = 0$ til $t = T$, er da givet ved

$$D = \int_0^T \sqrt{x'(t)^2 + y'(t)^2}\, dt \ , \tag{5}$$

hvor $x'(t)$ og $y'(t)$ er de afledede af hhv $x(t)$ og $y(t)$ mht tiden t (i matematikken kaldes D for "buelængden" for kurven beskrevet ved $x(t)$ og $y(t)$, og D er uafhængig af enheden for t).

På kondiløberens sidste tur medbragte han en GPSmodtager, som optog hans position som funktion af tiden; disse data har formen (t_i, x_i, y_i) for $i = 1, 2, \ldots, n$ og de ligger i filen `kondi.mat` i form af tre vektorer `t`, `x` og `y`. Enheden for `x` og `y` er kilometer, mens værdierne for `t` ligger mellem 0 og 1 (dvs de er skaleret mht den samlede løbetid).

2.1) Brug Matlab-funktionen `polyfit` til at beregne to polynomier $p_x(t)$ og $p_y(t)$ af grad 12, som fitter hhv x- og y-data. Plot måledata (x_i, y_i) sammen med den fittede kurve $\big(p_x(t), p_y(t)\big)$; brug Matlab-funktionen `polyval` til at beregne polynomierne.

2.2) Brug Matlab-funktionen `polyder` til at beregne de afledede $p'_x(t)$ og $p'_y(t)$ af de to polynomier, og plot dem som funktion af t.

2.3) Brug `quad` til at beregne distancen D vha formel (5) med en nøjagtighed på `tol = 1e-6`.

Spørgsmål 3. ODE metoder og skridtlængde ($1^1/_2$ timer)

I dette spørgsmål skal vi anvende koden `eulode.m` fra bogen (ligger på CampusNet) samt lave en kode `heunode.m` for Heun's metode. Formålet er at undersøge metodernes nøjagtighed i forhold til den skridtlængde man vælger.

Vi vil løse følgende begyndelsesværdiproblem:

$$\frac{dy}{dt} = yt^3 - 1.5y \ , \tag{6}$$

for $y(0) = 1$ over intervallet fra $t = 0$ til $t = 2$. Den analytiske løsning til differentialligningen i (6) er

$$y(t) = y(0)e^{0.25t^4 - 1.5t} \ . \tag{7}$$

3.1) Brug koden `eulode.m` til at løse begyndelsesværdiproblemet i (6) numerisk med skridtlængde på hhv. $h = 0.5$ og $h = 0.25$. Plot de to løsningerne samt den analytiske løsning i samme figur. Hvor stor er den relative fejl til tiden $t = 2$ ved brug af Eulers metode for de to forskellige skridtlængder?

Eulers metode (formel (22.5) i lærebogen) til numerisk løsning af (6) skrives

$$y_{i+1} = y_i + f(t_i, y_i)h \ , \tag{8}$$

hvor h er skridtlængden og $f(t, y) = \frac{dy}{dt}$.

Ligeledes kan Heun's metode (formel (22.18)+(22.19) i lærebogen) skrives

$$
\begin{aligned}
y_{i+1}^0 &= y_i + f(t_i, y_i)h \ , \\
y_{i+1} &= y_i + \frac{f(t_i, y_i) + f(t_{i+1}, y_{i+1}^0)}{2} h \ ,
\end{aligned}
\tag{9}
$$

hvor den øverste formel er identisk med Eulers metode.

3.2) Koden `eulode.m` implementerer Eulers metode i formel (8). Du skal modificere denne kode, således at den implementerer Heun's metode, dvs. de to formler i (9). Kald din modificede m-fil for `heunode.m`.

3.3) Brug din kode `heunode.m` til at løse begyndelsesværdiproblemet i (6) numerisk med skridtlængde på hhv. $h = 0.5$ og $h = 0.25$. Plot løsningerne i samme figur som løsningerne til spørgsmål **3.1** og den analytiske løsning. Hvor stor er den relative fejl til tiden $t = 2$ ved brug af Heun's metode for de to forskellige skridtlængder?

3.4) Beregn den relative fejl til tiden $t = 2$ med koden `eulode.m` og `heunode.m` med små skridtlængder $h = 0.1$, $h = 0.01$ og $h = 0.001$. Brug resultaterne til at demonstrere, at den relative fejl ved Eulers metode og Heun's metode er proportional med henholdsvis $O(h)$ og $O(h^2)$.

Spørgsmål 4. Epidemi i en befolkning (1 time)

Formålet med dette spørgsmål er at træne brugen af Matlabs indbyggede ODE løser `ode45`. Kermack-McKendricks model for forløbet af en epidemi i en befolkning er et begyndelsesværdiproblem beskrevet ved følgende system af ODEer,

$$
\begin{aligned}
\frac{dy_1}{dt} &= -c\, y_1\, y_2 \ , \\
\frac{dy_2}{dt} &= c\, y_1\, y_2 - d\, y_2 \ , \\
\frac{dy_3}{dt} &= d\, y_2 \ ,
\end{aligned}
\tag{10}
$$

hvor y_1 repræsenterer den procentdel af befolkningen, der er modtagelig overfor smitte, y_2 repræsenterer den procentdel af befolkningen, der er inficerede og y_3 repræsenterer den procentdel af befolkningen, der er immune. Parametrene c og d bestemmer henholdsvis inficeringsraten og helbredelsesraten.

Det normale forløb når vilkårerne for en epidemi er til stede er at antallet af inficerede individer stiger til at begynde med og derefter aftager langsomt mod nul.

4.1) Brug Matlabs indbyggede funktion `ode45` til at løse systemet i (10) for tiden $t = 0$ til $t = 1$. Som begyndelsesværdier og parametre skal du bruge:

$$
y_1(0) = 99, \quad y_2(0) = 1, \quad y_3(0) = 0, \quad c = 1, \quad d = 5. \tag{11}
$$

Husk, at hvis du erklærer `odefun` som en anonym funktion skal den første variabel være `t`, altså `f = @(t,y) ....`

4.2) Plot løsningen for hver variabel y_1, y_2 og y_3 i samme figur for $t = 0$ til $t = 1$. Angiv hvor stor en procentdel af befolkningen der er inficeret når denne procentdel er maksimal. Angiv f.eks. ved at aflæse på kurven, ca. det tidspunkt mellem $t = 0$ og $t = 1$ hvor der er ligeså mange inficerede personer som der er immune personer.

Spørgsmål 5. Omskrivning af anden-ordens ligninger (1 time)

Dette spørgsmål træner hvorledes vi omskriver en anden-ordens differentialligning til et system af første-ordens differentialligninger. Denne generelle teknik er vigtig, fordi den gør at vi kan nøjes med udvikle og bruge software til løsning af systemer af første-ordens ligninger!

På side 548 i bogen er forklaret, hvorledes man transformerer anden-ordens ligningen

$$m\,\frac{d^2x}{dt^2} + c\,\frac{dx}{dt} + k\,x = 0 \tag{12}$$

til dette system af første-ordens ligninger:

$$\frac{dx}{dt} = v, \qquad m\,\frac{dv}{dt} + c\,v + k\,x = 0\ . \tag{13}$$

Hvis vi indfører vektorerne $y = \begin{pmatrix} x \\ v \end{pmatrix}$ og $\frac{dy}{dt} = \begin{pmatrix} dx/dt \\ dv/dt \end{pmatrix}$, så vil den tilsvarede Matlab-funktion `odefun`, der skal bruges in `ode45`, se således ud:

```
function dydt = odefun(t,y)
m = ___;  c = ___; k = ___;
x = y(1);
v = y(2);
dydt = [ v ; - (c/m)*v - (k/m)*x ];
```

5.1) Udled $\frac{dy}{dt}$ for ligning (13) og forklar hvad der sker i Matlab-funtionen `odefun`.

5.2) Betragt nu anden-ordens ligningen

$$\frac{d^2x}{dt^2} - (1 - x^2)\,\frac{dx}{dt} + x = 0\ . \tag{14}$$

Omskriv dette til et system af første-ordens ligninger, og skriv den tilhørende Matlab-funktion `odefun`, analogt med ovenstående. Du skal bare skrive koden, som skal have korrekt syntaks; men du skal **ikke** løse differentialligningen vha `ode45`.

Spørgsmål 6. Finite difference metoden (1 time)

I dette spørgsmål skal vi løse et simpelt randværdiproblem med finite difference metoden.

Vi betragter følgende andenordens differentialligning,

$$7\frac{d^2y}{dx^2} + \frac{dy}{dx} - y + x = 0, \tag{15}$$

over intervallet $x = 0$ til $x = 20$ og med randbetingelserne givet ved

$$y(0) = 5, \quad y(20) = 8. \tag{16}$$

Vi vil finde løsningen til randværdiproblemet ved at omskrive det til et lineært ligningssystem $\mathtt{Ay} = \mathtt{b}$.

Hvis vi diskretiserer intervallet $0 \leq x \leq 20$ med

$$x_i = (i-1)\Delta x, \quad i = 1, 2, \ldots, N, \tag{17}$$

hvor $N = \frac{20}{\Delta x} + 1$ og Δx er skridtlængden, kan den første afledede $\frac{dx}{dy}$ udtrykkes med finite difference tilnærmelsen,

$$\frac{dy}{dx} = \frac{y_{i+1} - y_{i-1}}{2\Delta x}, \tag{18}$$

og den anden afledede udtrykkes med finite difference tilnærmelsen,

$$\frac{d^2y}{dx^2} = \frac{y_{i+1} - 2y_i + y_{i-1}}{\Delta x^2}, \tag{19}$$

hvor $y_i = y(x_i)$.

6.1) Indsæt de to finite difference tilnærmelser (18) og (19) i differentialligningen (15) og erstat x med x_i og y med y_i. Omskriv dette udtryk til formen

$$c\, y_{i-1} + d\, y_i + e\, y_{i+1} + x_i = 0, \tag{20}$$

hvor c, d, og e er funktioner af Δx. Angiv udtrykkene for c, d, og e.

Vi kan nu sætte $\Delta x = 0.1$ og opstille et lineært ligningssystem $\mathtt{Ay} = \mathtt{b}$, givet ved

$$\mathtt{A} = \begin{pmatrix} d & e & & & \\ c & d & e & & \\ & \ddots & \ddots & \ddots & \\ & & c & d & e \\ & & & c & d \end{pmatrix}, \qquad \mathtt{b} = \begin{pmatrix} -x_2 - c\, y(0) \\ -x_3 \\ -x_4 \\ \vdots \\ -x_{N-2} \\ -x_{N-1} - e\, y(20) \end{pmatrix} \tag{21}$$

Matricen `A` kan laves med følgende indbyggede Matlab kommando:

```
A = full(gallery('tridiag',N-2,c,d,e));
```

Højresiden `b` kan laves med følgende kode (hvor $\mathtt{dx} = \Delta x$):

```
x = (0:dx:20)'; b = -x(2:N-1); b(1) = b(1)-c*5; b(end) = b(end)-e*8;
```

6.2) Find den diskrete løsning til randværdiproblemet i (15) for $\Delta x = 0.1$ ved at løse det lineære ligningssystem og vis løsningen på en figur (dvs. plot de diskrete punkter (x_i, y_i), $i = 1, 2 \ldots, N$).

Finite difference metoden kan også anvedes til at løse begyndelsesværdiproblemer. Hvis vi i udtrykket (20) isolerer y_{i+1} på venstresiden får vi:

$$y_{i+1} = -\frac{1}{e}(c\, y_{i-1} + d\, y_i + x_i), \tag{22}$$

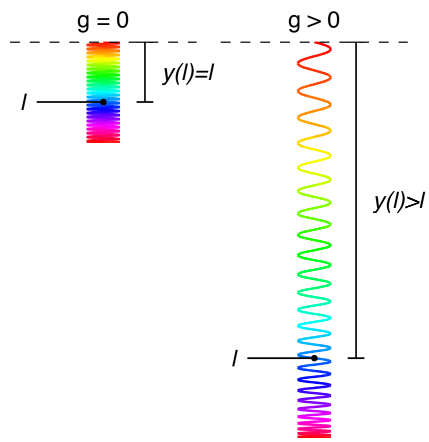
6.3) Brug denne formel i en `for` løkke til at løse differentialligningen (15) som begyndelsesværdiproblem med begyndelsesværdier: $y(0) = 5$ og $\frac{dy}{dx}(0) = -0.8434$ (dvs. start iterationen med $y_0 = 5$ og $y_1 = y_0 - 0.8434\,\Delta x$). Plot din løsning i en figur sammen med løsningen fra forrige spørgsmål. Hvor langt ved siden af $y(20) = 8$ rammer vi med $\Delta x = 0.1$ og med $\Delta x = 0.01$?

Opgave A. Slinky der holdes fast i den ene ende (3 timer)

I denne opgave vil vi finde formen og længden af en Slinky, der holdes fast i strakt arm i den ene ende mens den anden ende hænger frit (se første billede på figuren side 1). Vi antager, at vores Slinky har følgende specifikationer:

- Masse: $M = 0.2\,\mathrm{kg}$.
- Fjederkonstant: $k = 1\,\mathrm{kg/s^2}$.
- Ustrækket længde: $L_0 = 0.1\,\mathrm{m}$.
- Diameter: $0.1\,\mathrm{m}$.
- Viklinger: 25.

Tyngdeaccelerationen sættes til: $g = 9.82\,\mathrm{m/s^2}$.



Lad os først betragte den ustrækkede Slinky. Hvis der ikke var nogen tyngdekraft ville Slinky'en blot have form og længde som den ustrækkede Slinky. Vi vil lade parameteren l repræsentere et sted på Slinky'en, således at $0 \le l \le L_0$. Vi kan derefter beskrive Slinky'ens form ved en funktion $y(l)$. Her er l er et punkt på Slinky'en og $y(l)$ er højden under den fastholdte ende, som vist på figuren.

Uden tyngdekraft får vi formen af den ustrækkede Slinky, givet ved

$$y(l) = l, \quad 0 \le l \le L_0. \tag{23}$$

Det oplyses desuden, at der *med* tyngdekraft haves følgende funktion for formen,

$$y(l) = l + a\,g\left(\frac{2l}{L_0} - \frac{l^2}{L_0^2}\right), \quad 0 \le l \le L_0, \tag{24}$$

hvor a er en konstant, g er tyngdeaccelerationen og L_0 er den ustrækkede længde.

Strækket der mærkes i punktet l på Slinky'en er givet ved $y'(l) - 1$, hvor $y' = \frac{dy}{dl}$.

Den totale potentielle energy for Slinky'en er givet ved integralet

$$I = \int_0^{L_0} \left[-\frac{1}{L_0} M\,g\,y(l) + \frac{1}{2} k\,L_0\,(y'(l) - 1)^2 \right] dl, \tag{25}$$

hvor første led angiver den potentielle energi fra tyngdekraften og andet led er den potentielle energi fra fjederkraften i den strækkede Slinky.

Vi ønsker at bestemme værdien af konstanten a når Slinky'en hænger stille så vi kan beregne dens form fra formel (24). Denne ligevægtstilstand opnås præcis når den totale potentielle energy, givet ved integralet i formel (25), har sit minimum.

A.1) Skriv et udtryk for den afledede $y'(l)$ både for formel (23) og (24). Antag nu, at vi kan sætte konstanten $a = 1/g$ i formel (24). Hvad er værdien af strækket $y'(0) - 1$, dvs. i den ende af Slinky'en vi holder fast i, for formel (23) og (24)?

A.2) Vi kan beregne værdien af integralet I i formel (25) ved brug af den indbyggede Matlab-funktion `quad`. Du skal lave to separate Matlab-funktioner til beregning af I;

én for formel (23) og én for formel (24). Hvis man igen bruger $a = 1/g$, hvad er den beregnede værdi af I for de to funktioner?

A.3) Brug `fminbnd` til at bestemme den værdi af a hvor den totale potentielle energi har sit minimum. Sammenlign værdien du fandt med $a = M/(2k)$, som kan udledes analytisk. Hvad er værdien I af integralet for formen (24) med dit a?

Vi ønsker at sikre os at Slinky'en ikke rammer jorden mens vi forsøger at få den til at hænge stille. Lad os antage, at vi som udgangspunkt har Slinky'en sammenfoldet i begge hænder hvorefter vi giver promte slip med den ene hånd. Slinky'en vil derefter udvide sig til en maksimum længde $L_{\max}$ og så svinge op igen.

Når Slinky'en er sammenfoldet har den formen (23) med den totale potentielle energi som blev beregnet i **A.2**. Når Slinky'en har den maksimale længde har den samme totale potentielle energi men med formen

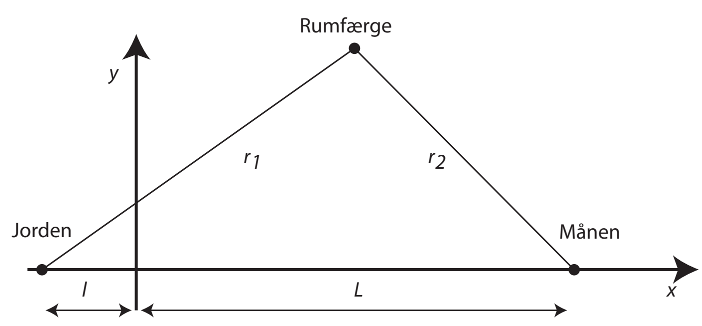
$$y(l,s) = s\left[l + a\,g\left(\frac{2l}{L_0} - \frac{l^2}{L_0^2}\right)\right], \quad 0 \le l \le L_0, \tag{26}$$

der svarer til formen (24) blot skaleret længere med en faktor s.

A.4) Lav en Matlab-funktion som kan beregne værdien af integralet $I = I(s)$ i formel (25) for den skalerede form (26) som funktion af s (konstanten a er stadig dit a fra **A.3**). Lad I_{ustr} være den værdi af I du fandt i **A.2** for formen (23). Brug `fzero` til at beregne den værdi $s > 1$, som tilfredsstiller $I(s) = I_{\text{ustr}}$. Angiv dit fundne s og $L_{\max} = y(L_0, s)$.

Opgave B. Rumfærgens bane mellem jord og måne (1½ timer)

Formålet med dette spørgsmål er at prøve at løse anden-ordens differentialligninger som koblede første-ordens differentialligninger.



Vi vil simulere en rumfærges kredsløb mellem jorden og månen ved at beskrive rumfærgens position (x, y) i et to-dimensionelt koordinatsystem som funktion af tiden t. Vi antager, at koordinatsystemet bevæger sig med jorden og månen således at begge deres positioner kan betragtes som faste. Figuren viser opstillingen. Newton's anden lov giver følgende system af *anden-ordens* differentialligninger:

$$\begin{aligned} \frac{d^2x}{dt^2} &= a\,x + b + 2\Omega\frac{dy}{dt}, \\ \frac{d^2y}{dt^2} &= c\,y - 2\Omega\frac{dx}{dt}, \end{aligned} \tag{27}$$

hvor

$$
\begin{aligned}
a &= \Omega^2 - GM/r_1^3 - Gm/r_2^3, \\
b &= -GM\mu L/r_1^3 + Gm\mu^* L/r_2^3, \\
c &= \Omega^2 - GM/r_1^3 - Gm/r_2^3.
\end{aligned} \tag{28}
$$

Der indgår her adskillige konstanter, hvor G er gravitationskonstanten, M og m er masserne af jorden og månen, μ^* og μ er de tilsvarende reducerede masser, og Ω er hastigheden månen roteret rundt om jorden. Værdierne af konstanterne samt formler for beregning af r_1 og r_2 er givet ved:

$$
\begin{aligned}
\Omega &= 2.661 \cdot 10^{-6}\,/\mathrm{s}, & \mu^* &= m/(m+M) \\
G &= 6.67259 \cdot 10^{11}\,\mathrm{m}^3/(\mathrm{kg\,s})^2 & L &= 3.844 \cdot 10^8\,\mathrm{m} \\
M &= 5.974 \cdot 10^{24}\,\mathrm{kg} & l &= 4.669 \cdot 10^6\,\mathrm{m} \\
m &= 7.348 \cdot 10^{22}\,\mathrm{kg} & r_1 &= [(x+l)^2 + y^2]^{1/2} \\
\mu &= M/(m+M) & r_2 &= [(L-l-x)^2 + y^2]^{1/2}.
\end{aligned} \tag{29}
$$

På CampusNet har vi uploadet koden `parametre_opgB.m`, som ud fra de oplyste konstanter beregner de tre parametre a, b og c i formel (28) for et givet (x, y).

B.1) Omskriv systemet af de to koblede anden-ordens differentialligninger i formel (27) til et system af fire koblede *første-ordens* differentialligninger. Det nemmeste er at definere en vektor z med fire variable,

$$
z = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = \begin{pmatrix} x \\ y \\ \frac{dx}{dt} \\ \frac{dy}{dt} \end{pmatrix}, \tag{30}
$$

og herefter opskrive den afledede $\frac{dz}{dt}$ givet ved z selv og højresiderne af lign. (27).

B.2) Lav en Matlab-funktion `odefun`, der givet en vektor `z` beregner en vektor `dzdt` for den afledede $\frac{dz}{dt}$ som du fandt i **6.1**. Funktionen skal kalde `parametre_opgB`. Husk at det første argument i `odefun` skal være t også selvom tiden ikke bruges til beregningen af $\frac{dz}{dt}$.

B.3) Brug den indbyggede Matlab-funktion `ode45` til at beregne rumfærgens bane $\big(x(t), y(t)\big)$ som funktion af tiden for begyndelsesværdierne

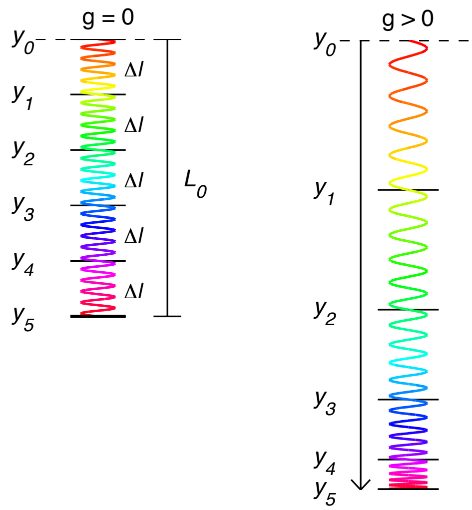
$$
x(0) = 4.613 \cdot 10^8, \quad y(0) = 0, \quad \frac{dx}{dt}(0) = 0, \quad \frac{dy}{dt}(0) = -1074, \tag{31}
$$

fra $t = 0$ til $t = 2.4 \cdot 10^6$ s. Lav et plot af rumfærgens bane hvor du også indikerer positionen af jorden og månen på figuren. Du kan evt. lave baggrunden af figuren sort (som rummet) med kommandoen `set(gcf,'color','black'); axis off;`.

Opgave C. Den faldende Slinky (2 timer)

I denne opgave vil vi simulere hvad der sker med en Slinky over tid når den slippes efter at have været holdt fast i den ene ende (se billede 2-4 på figuren side 1).

Vi vil modellere Slinky'en som værende opbygget af n små stykker hver med længde $\Delta l = L_0/n$, som vist til venstre på figuren nedenfor for $n = 5$.



Modellen skal beskrive bevægelsen over tid af de små stykkers endepunkter, dvs. vi knytter en differentialligning til hvert endepunkt y_j, $j = 0, 1, \ldots, n$. Vi bemærker, at der vil være ialt $n + 1$ endepunkter til n små stykker.

Til tiden $t = 0$ vil vi gå ud fra at Slinky'en har den udstrakte form fra Opgave A, således at

$$y_j = y(l_j), \quad l_j = j\Delta l, \quad j = 0, 1, \ldots, n. \tag{32}$$

hvor $y(l)$ er givet ved formel (24).

Vi vil desuden antage at Slinky'en hænger helt stille til at begynde med, således at

$$\frac{dy_j}{dt} = 0, \quad j = 0, 1, \ldots, n, \quad t = 0. \tag{33}$$

C.1) Opret vektorerne `y` og `l` i Matlab defineret i formel (32) for $n = 400$ til tiden $t = 0$. For a anvendes det analytiske udtryk $a = M/(2k)$. Brug funktionen `draw_slinky(t,y,l,diameter,viklinger)`, der er lagt på CampusNet, til at tegne en figur af Slinky'en til $t = 0$.

For tiden $t > 0$ løses et system af $n + 1$ anden-ordens differentialligninger, givet ved

$$\frac{d^2 y_j}{dt^2} = f_j, \quad j = 0, 1, \ldots, n, \quad t > 0, \tag{34}$$

hvor f_j, $j = 0, 1, \ldots, n$ definerer en vektor `f` af længde $n + 1$, der beregnes med funktionen `ffun(l,y,dl,M,k,L0,g)` lagt på CampusNet (se `ffun.m`).

Til at løse (34) skal du bruge finite-difference metoden,

$$\frac{d^2 y_j}{dt^2} = \frac{y_j^{i-1} - 2y_j^i + y_j^{i+1}}{\Delta t^2} = f_j, \quad j = 0, 1, \ldots, n, \tag{35}$$

som kan omskrives til formlen

$$y_j^{i+1} = 2y_j^i - y_j^{i-1} + \Delta t^2 f_j, \quad j = 0, 1, \ldots, n, \tag{36}$$

hvor y_j^i er positionen af punkt j til tiden $t = i\Delta t,\ i = 0, 1, 2, \ldots,$ og Δt er et lille tidsskridt.

C.2) Lav et script som implementerer finite-difference skridtene i formel (36) og dermed simulerer Slinky'ens fald. Du skal simulere fra $t = 0$ til $t = 0.5$ og anvende tidsskridt $\Delta t = 0.001$. Lav en figur af Slinky'en til tiden $t = 0.1$, $t = 0.25$ og $t = 0.5$.

Hint: Det kan være en ide at have en vektor `y` med elementerne y_j^i og en vektor `yold` med elementerne y_j^{i-1} og så lave en ny vektor `ynew` med elementerne y_j^{i+1} efter formel (36). Herefter kan man så skrive `yold = y;` og `y = ynew;` før næste iteration.

Animation: Bemærk at `draw_slinky` også kan vise en animation af simulationen hvis hele outputtet fra de forskellige tidsskridt indsættes som matricer. Dette kræves ikke! Spørg evt. hjælpelærerne hvis du vil prøve det.