KTH Engineering Sciences

Laboration 2

Differentialekvationer

Före redovisningen ska ni skicka in Matlab*-filer och vissa resultatfiler via Kurswebben. Vad som ska skickas in står angivet efter respektive uppgift. På redovisningen ska ni (båda i laborationsgruppen om ni är två) kunna redogöra för teori och algoritmer som ni använt. Ni ska också kunna svara på frågepunkterna (•) och förklara hur era* Matlab*-program fungerar. Kom väl förberedda!*

1. Pendeln

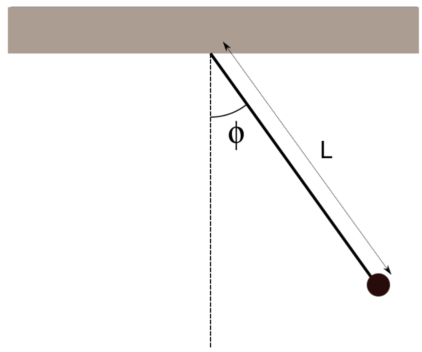
Vi studerar en dämpad pendeln som beskrivs av den ordinära differentialekvationen (ODEn)

$$m\phi'' + \alpha\phi' + \frac{mg}{L}\sin(\phi) = 0, \qquad (1)$$

med begynnelsedata

$$\phi(0) = \phi_0, \quad \phi'(0) = 0.$$

Den okända funktionen $\phi(t)$ är vinkeln för pendelns utslag. Parametrarna i ekvationen är pendelns massa m, friktionskoefficienten α, tyngdaccelerationen $g = 9.81$ och pendelns längd L. Se figur.



a) För små utslag kan man approximera $\sin(\phi) \approx \phi$. Det ger en *linjär* ODE som approximerar (1),

$$m\phi'' + \alpha\phi' + \frac{mg}{L}\phi = 0, \qquad \phi(0) = \phi_0, \quad \phi'(0) = 0. \qquad (2)$$

I denna och nästa deluppgift ska ni använda $\phi_0 = 0.1$, $m = 0.1$, $L = 1$ och $\alpha = 2$. För detta värde på friktionskoefficienten α är systemet överdämpat.

Skriv först om (2) till ett system av första ordningens ekvationer. Implementera sedan Framåt Euler-metoden i Matlab och lös problemet med tidssteget $h = 0.01$ fram till $t = 10$. Plotta lösningen $\phi(t)$. (Hjälp: $\phi(10) \approx 0.67 \cdot 10^{-3}$.)

När ni fått detta att fungera ska ni skriva om Matlab-programmet till en Matlab-funktion, `feuler` som tar följande indataargument:

- begynnelsedata `u0` (en vektor med två komponenter),
- sluttiden `T`,

- antal tidssteg `n` (notera: $h = T/n$),
- värdet på friktionskoefficienten `alpha`.

Funktionen ska returnera två variabler:

- kolumnvektorn `t` som innehåller alla tidpunkter, dvs `t = [0,h,2h, ..., T]`^{T} (längd `n+1`),
- matrisen `y` som innehåller två kolumner med approximationerna av ϕ respektive ϕ' i motsvarande tidpunkter (storlek $n+1$ rader, 2 kolumner).

Notera: Med givna returvärden `t` och `y` från `feuler.m` kan lösningsplotten nu erhållas med kommandot `plot(t,y(:,1))`.

Verifiera att er implementation av Framåt Euler har noggrannhetsordning ett genom att beräkna $\phi(10)$ för flera olika `n` med hjälp av `feuler.m`. Beräkna kvoter av differenser mellan lösningar med n, $2n$ och $4n$ på samma sätt som i uppgift 4, Laboration 1. Sammanställ en tabell med kvoter och motsvarande noggrannhetsordningar.

Skicka in: Funktionsfilen `feuler.m`; anropas som "`[t,y] = feuler(u0,T,n,alpha)`"
(Tabellen skickar ni in nedan.)

b) Implementera Runge–Kutta 4 och skapa en funktionsfil med samma argument och returvärden som för Framåt Euler. Verifiera dess noggrannhetsordning på samma sätt som ni gjorde för Framåt Euler ovan. Lägg till kolumner i tabellen med kvoter och noggrannhetsordningar för Runge–Kutta 4.

- Vad är den teoretiska noggrannhetsordningen?
- Kvantifiera med ett exempel hur mycket noggrannare Runge–Kutta 4 är jämfört med Framåt Euler.

Skicka in: 1) Funktionsfilen `rk4.m`; anropas som "`[t,y] = rk4(u0,T,n,alpha)`". 2) Tabellen `kvottabell` med kvoterna och motsvarande noggrannhetsordningar för Framåt Euler och Runge–Kutta 4. Formatet kan vara vanlig ASCII.

c) Gå nu tillbaka till den exaktare olinjära modellen (1). Använd samma parametrar som i deluppgift (a) och (b) förutom friktionskoefficienten som ni byter till $\alpha = 0.05$. Det ger ett underdämpat system. Skriv om (1) till ett system av första ordningens ekvationer. Skapa den nya funktionsfilen `rk4olin.m` i vilken ni modifierar er Runge–Kutta 4-lösare från deluppgift (b) så att den istället löser detta olinjära problem. Funktionsfilen ska ha samma argument och returvärden som tidigare.

Ni ska nu jämföra lösningarna till det linjära och det olinjära problemet. Använd `rk4.m` för att lösa det linjära problemet (2) och `rk4olin.m` för att lösa det olinjära problemet (1) med samma begynnelsedata och parametrar fram till $t = 10$. Plotta båda lösningarna i samma figur.

Detta ska ni upprepa för fyra olika startvinklar: $\phi_0 = 0.1,\ 1,\ 1.5,\ 3.1$. Skapa en figur med fyra delrutor med hjälp av kommandot `subplot` (se `help subplot`, och även uppgift 1 i Matlab-övningarna). Det blir fyra olika jämförande plottar, en för varje vinkel, som ska ligga varsin delruta.

- För vilka startvinklar ϕ_0 är den enklare linjära modellen en bra approximation av den olinjära modellen?

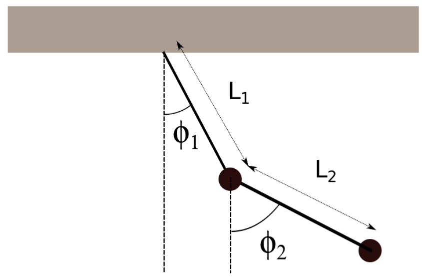
Skicka in: 1) Funktionsfilen `rk4olin.m`; anropas som "`[t,y] = rk4olin(u0,T,n,alpha)`",
2) Figuren `pendlar` med de fyra delrutorna, antingen i `.fig`-format (gör Save i figur-fönstret) eller i `.png`-format (gör `print -dpng pendlar.png` vid prompten).

d) Animera den olinjära pendeln (1) i MATLAB för $t \in [0, 10]$ med parametrarna valda som ovan. Ni kan använda kodsnutten nedan. Den förutsätter att de approximerade vinklarna ligger i vektorn `P` och att pendellängden L är definierad i `L`.

```
for k=1:length(P)
    plot([-L L],[0,0],[0 L*sin(P(k))],[0 -L*cos(P(k))],'-o');
    axis equal
    axis(1.2*[-L L -L L]);
    drawnow
end
```

Välj en stor startvinkel ($\phi_0 > 2$) för bäst effekt. Ni kan justera animationshastigheten (fps) genom att variera antalet tidssteg n i den numeriska lösningen.

e) Om man kopplar ihop två vanliga pendlar får man en dubbelpendel. Se figur. Denna typ av pendel har ett betydligt mer komplicerat beteende än den enkla pendeln. Även om man bortser från friktion blir rörelsen sällan periodisk, och ofta kaotisk; små förändringar i begynnelsedata ger mycket stora förändringar i lösningen. Dubbelpendelns dynamik utan friktion beskrivs av två kopplade andra ordningens ODE för utslagsvinklarna ϕ_1 och ϕ_2,



$$\phi_1'' = f_1(\phi_1, \phi_1', \phi_2, \phi_2'), \qquad \phi_2'' = f_2(\phi_1, \phi_1', \phi_2, \phi_2'), \tag{3}$$

där

$$f_1(\phi_1, \phi_1', \phi_2, \phi_2') = \frac{-m_2 L_2 {\phi_2'}^2 s - g(m_1 + m_2)\sin(\phi_1) - c m_2 \left(L_1 {\phi_1'}^2 s - g\sin(\phi_2)\right)}{L_1[m_1 + m_2 s^2]}$$

$$f_2(\phi_1, \phi_1', \phi_2, \phi_2') = \frac{(m_1 + m_2)\left(L_1 {\phi_1'}^2 s - g\sin(\phi_2)\right) + c\left(m_2 L_2 {\phi_2'}^2 s + g(m_1 + m_2)\sin(\phi_1)\right)}{L_2[m_1 + m_2 s^2]},$$

och

$$s = \sin(\phi_1 - \phi_2), \qquad c = \cos(\phi_1 - \phi_2).$$

Parametrarna är pendeldelarnas längder, L_1, L_2 och massorna m_1, m_2. ODEerna (3) kan skrivas om som ett system av fyra första ordningens ODE. Genom att sätta $u_1 = \phi_1$, $u_2 = \phi_1'$, $u_3 = \phi_2$ och $u_4 = \phi_2'$ får vi

$$\frac{d\boldsymbol{u}}{dt} = \boldsymbol{F}(\boldsymbol{u}), \qquad \boldsymbol{u} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix}, \qquad \boldsymbol{F}(\boldsymbol{u}) = \begin{pmatrix} u_2 \\ f_1(u_1, u_2, u_3, u_4) \\ u_4 \\ f_2(u_1, u_2, u_3, u_4) \end{pmatrix}.$$

Denna funktion finns implementerad i filen `fpendel.m` som ligger på kurswebben. Parametervalen där är $L_1 = 1.5$, $L_2 = 1$, $m_1 = 1$ och $m_2 = 1.5$. Observera att funktionen tar två argument, trots att t-parametern inte används. Detta beror på att MATLABs inbyggda ODE-funktioner kräver ett sådant format.

- Använd Matlabs inbyggda funktion `ode45` för att lösa (3) med begynnelsedata $\phi_1(0) = 2$, $\phi_2(0) = 0.8$ och $\phi_1'(0) = \phi_2'(0) = 0$. Beräkna lösningen fram till $t = 5$. Plotta vinklarna $\phi_1(t)$ och $\phi_2(t)$.
- Animera dubbelpendelns rörelse för tiden $t \in [0, 50]$ i en ny figur[1]. Ni kan tex använda en lätt modifierad version av animeringskoden från deluppgift (d) för detta.

  Notera att `ode45` i allmänhet ger lösningen vid icke ekvidistanta tidpunkter, till skillnad från er Runge–Kutta-lösare. Det gör animeringen lite ryckig. Försök göra den mjukare. T.ex. kan ni i anropet till `ode45` specificera för vilka tidpunkter lösningen ska beräknas. (Se `help ode45`.)

  Prova gärna att variera pendeldelarnas längder och massor.

Skicka in: Filen `dubbelpendel.m` som innehåller programmet.

2. Randvärdesproblem

Följande differentialekvationsproblem beskriver temperaturfördelningen $T(x)$ i en cylindrisk stav av längden L.

$$-\frac{d}{dx}\left(k\frac{dT}{dx}\right) = Q(x), \qquad T(0) = t_0, \quad T(L) = t_1. \tag{4}$$

Vänster och höger ändpunkt på staven har den konstanta temperaturen t_0 resp t_1. Konstanten k är stavens värmeledningsförmåga och $Q(x)$ är den värmemängd som per tidsenhet och volymsenhet genereras i staven, tex genom radioaktivitet eller yttre uppvärmning. Antag att $L = 2$ $[m]$, $k = 3$ $[J/(K \cdot m \cdot s]$, $t_0 = 290$ $[K]$, $t_1 = 320$ $[K]$ och $Q(x)$ $[J/(s \cdot m^3)]$ är funktionen

$$Q(x) = 5000\, e^{-500(x-0.25L)^2} + 1200\, e^{-10(x-0.7L)^2}, \qquad 0 \le x \le L.$$

Differentialekvation och randvillkor (4) kan lösas numeriskt med hjälp av finita differensmetoden. Om vi diskretiserar intervallet $[0, L]$ enligt $x_i = ih$, $i = 0, 1, 2, \ldots, n, n+1$, där $h(n+1) = L$ och approximerar andraderivatan med centraldifferens erhålles

$$\frac{-T_{i-1} + 2T_i - T_{i+1}}{h^2} = \frac{1}{k}Q(x_i), \qquad i = 1, 2, \ldots, n. \tag{5}$$

Tillsammans med randvillkoren $T_0 = t_0$ och $T_{n+1} = t_1$ leder diskretiseringen till ett linjärt ekvationssystem

$$AT = b,$$

där A är en $n \times n$-matris, T är en n-vektor med temperaturvärden i det inre av intervallet och b är en n-vektor som beror av bla randvärdena t_0 och t_1 samt $Q(x_i)$-värdena.

a) Skriv ett MATLAB-program som konstruerar matrisen A och högerledet b för en given storlek n. Matrisen kommer endast att ha ett fåtal nollskilda element. Matrisen skapar du tex med hjälp av MATLABs kommando `diag(v)`, som genererar en diagonalmatris med vektorn `v` på diagonalen, samt generaliseringen `diag(v,p)` som genererar en matris med vektor `v` på diagonal nummer `p`, där `p` kan vara både positiv och negativ.

Tips: Skriv först ner matrisen A med papper och penna för ett enkelt fall, tex $n = 4$, för att se strukturen hos matrisen. Verifiera också att programmet genererar korrekt matris för detta fall.

[1] Kommandot `figure` ger ett nytt figurfönster.

b) Bygg ut programmet så att det räknar ut temperaturen T genom att lösa det linjära ekvationssystemet $AT = b$ med backslash, \. Pröva först med $n = 10$ inre punkter. Skriv sedan en loop som beräknar temperaturen för $n = 10, 20, 40, 80$ (eventuellt fler n). Programmet ska dels plotta de erhållna resultaten i samma figur, dels skriva ut den minimala och maximala temperaturen, samt medeltemperaturen för varje n.

Lösningsvektorn T kommer att innehålla temperaturen i alla punkter x_i utom i randpunkterna. Ni ska plotta temperaturen som funktion av x på *hela* intervallet $0 \leq x \leq L$. Randvärdena ska även inkluderas vid beräkningen av min-, max- och medeltemperaturen.

- Hur tillförlitliga är de beräknade värdena?

Skicka in: Filen `stav.m` som innehåller programmet.

3. Sammanställning av filer som ska skickas in

Uppgift 1: `feuler.m`, `rk4.m`, `kvottabell`, `rk4olin.m`, `pendlar`, `dubbelpendel.m`

Uppgift 2: `stav.m`

SF1546, Numeriska metoder

Laboration 2 redovisad och godkänd! Datum: ....

Namn:... Godkänd av ....