TANA23 Matematiska algoritmer och modeller

Laboration 4. Differentialekvationer

1 Introduktion

Redovisning sker genom att en kort rapport skrivs där svaret på alla frågor finns med. Program skrivna i Python skall bifogas. Tänk på att programmen skall skickas på ett format som gör att jag kan testköra dem. Enklast är att bara lägga in all kod i en textfil med lite kommentarer angående vilken uppgift som ett visst kodstycke löser. Bifoga även grafer. Den färdiga rapporten skall skickas med epost till `fredrik.berntsson@liu.se`. Var noga med att skriva *TANA23 Lab4* i rubriken.

2 Klassiska Runge-Kutta metoden

Vi skall först studera en enkel differentialekvation. Betrakta problemet

$$y'(t) = f(t, y) = -y(t), \quad y(0) = 1.$$

I denna övning skall vi lösa problemet numeriskt. Högerledsfunktionen kan skapas i Python genom att skriva

```
def funk( t, y ):
    return -y
```

Den exakta lösningen på problemet är $y(t) = \mathrm{e}^{-t}$.

Uppgift 2.1 För att beräkna lösningen till differentialekvationen ovan skall vi implementera Runge-Kuttas klassiska metod. Denna löser ett problem av typen: $y'(t) = f(t, y)$, $y(t_0) = y_0$. Den klassiska Runge-Kutta metoden beräknar $y_k \approx y(t_k)$ genom

$$\begin{aligned}
k_1 &= hf(t_k, y_k), \\
k_2 &= hf(t_k + h/2, y_k + k_1/2), \\
k_3 &= hf(t_k + h/2, y_k + k_2/2), \\
k_4 &= hf(t_k + h, y_k + k_3), \\
y_{k+1} &= y_k + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4).
\end{aligned}$$

Skriv en funktion i Python som beräknar lösningen till ett begynnelsevärdesproblem med Runge-Kuttas klassiska metod.

Funktionen skall ha en vektor `t`, en högerledsfunktion $f(t, y)$ och ett begynnelsevärde $y(t_0) = y_0$, som inparametrar. Din funktion skall alltså definieras som

```
def ClassicRK( t , f , y0 ):
     < Beräkna en vektor y sådan att y[k]=y(t[k]) >
     return y
```

Redovisa den funktion som implementerar Runge-Kuttas klassiska metod

Uppgift 2.2 Vi skall nu lösa problemet ovan och beräkna approximationer av $y(1)$ med Runge-Kuttas klassiska metod och olika steglängder h. Vi väljer ett antal delintervall n och skapar en vektor `t=np.linspace(0,1,n+1)`. Detta ger en steglängd $h = 1/n$. Vi löser sedan problemet med Runge-Kuttas metod och får $y_n = y(1; h) \approx y(1)$. Vi får då en tabell

n	5	10	20
h			
$y(1; h)$			

Fyll i och **redovisa tabellen**.

Uppgift 2.3 Vi skall nu utnyttja att $R_T(h) \approx C \cdot h^p$ för att hitta p. Utnyttja uttrycket för R_T och visa att

$$\frac{y(1; h) - y(1; h/2)}{y(1; h/2) - y(1; h/4)} = 2^p.$$

där $y(1; h)$ är den approximation av $y(1)$ som beräknats med steglängden h. Utnyttja sedan tabellen ovan för att bilda kvoter som borde få värdet 2^p och bestäm därigenom p. Stämmer ditt p med teorin? **Redovisa både beräkningarna och** p

2.1 En glödlampa

En glödlampa fungerar genom att en metalltråd värms och då avger ljus. När strömmen slås på tillförs effekt som höjer temperaturen enligt Ohms lag. Samtidigt kyls glödtråden enligt Boltzmanns T^4-lag. Detta ger en temperatur $T(t)$ som kan beskrivas med modellen,

$$T'(t) = E(t) - 2.76 \cdot 10^{-10}(T^4(t) - T_\infty^4) = f(t, T),$$

Strömmen slås på vid $t = 0$, då $T(0) = T_\infty$, där $T_\infty = 18\,^oC$ är rumstemperaturen. Strömmen slås av efter $1\,s$. Detta ger att

$$E(t) = \begin{cases} 10\,000, & t \leq 1.0 \\ 0, & \text{t>1.0.} \end{cases}$$

Kommentar Boltzmanns lag förutsätter att $T(t)$ anges i Kelvin istället för ^oC. Då vi räknar i Python skall enheten vara Kelvin men då resultat redovisas i grafer är det trevligare att ange temperaturer i ^oC. Denna typ av enhetsomvandligar måste alltid göras även om det är irriterande.

Uppgift 2.4 Vi börjar med att välja ett antalet gridpunkter $n = 100$. Vi skapar sedan en vektor med de tidpunkter där vi skall beräkna lösningen $T(t)$ med `t=np.linspace(0,5,n+1)`. Detta ger en steglängd $h = 5/n$. Beräkna sedan funktionen $E(t)$ ovan och plotta den på intervallet $0 < t < 5$.

Uppgift 2.5 Skriv en Python funktion `GlodTrad` som implementerar högerledsfunktionen $f(t, T)$ för glödlampan som beskrivits ovan. Lös även problemet med begynnelsevärdet $T(0) = 18\,^oC$ och $n = 100$. Plotta lösningen $T(t)$, för $0 < t < 5$. **Redovisa både grafen och Python funktionen** $f(t, T)$.

Uppgift 2.6 När man släcker en glödlampa ser man hur glödtråden svalnar och svartnar medan uppvärmning tycks gå mycket fortare. Verifieras detta fenomen av modellen?

3 Simulering av en Fotboll

En fotboll som rör sig genom luften påverkas av ett antal krafter som bestämmer dess bana genom luften. Inom Fysiken ges en matematisk beskrivning av dessa krafter. Problemet att beräkna bollbanan givet utgångsposition och hastighet är ett *begynnelsevärdesproblem* för en *differentialekvation*. Sådana är i allmänhet svåra att lösa analytiskt, annat än i enkla specialfall, men går utmärkt att lösa numeriskt med exempelvis Python.

Vårt mål med simuleringen är att beskriva bollbanan genom att hitta bollens position och hastighet som funktion av tiden. Dvs funktioner

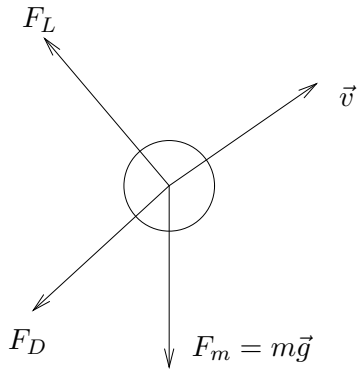
$$\vec{p}(t) = (x(t)\ y(t)\ z(t))^T, \qquad \text{och,} \qquad \vec{v}(t) = (v_x(t)\ v_y(t)\ v_z(t))^T.$$

Till vår hjälp har vi Newtons rörelselagar som säger att hastigheten är derivatan av positionen och att derivatan av hastigheten är accelerationen, eller

$$\vec{p}'(t) = \vec{v}(t), \quad \text{och} \quad \vec{v}'(t) = m^{-1}\vec{F}(t)$$

där m är bollens massa och $\vec{F}(t)$ är den kraft som verkar på bollen.

Vi skall nu studera de krafter som påverkar fotbollen i detalj och visa hur problemet kan beskrivas med en differentialekvation.



Den första kraft som påverkar fotbollen är *gravitationen*. Denna ges av uttrycket

$$\vec{F}_m = m\vec{g} = m \cdot (0, 0, -g),$$

där m är bollens massa och g är gravitationskonstanten.

Uppgift 3.1 Låt

$$S(t) = (p(t)\,,\, v(t))^T = (x(t)\ y(t)\ z(t)\ v_x(t)\ v_y(t)\ v_z(t))^T,$$

vara de okända funktioner som fullständigt beskriver bollbanan. Formulera det system av differentialekvationer som skall lösas.

Tips Vi vet att exempelvis $x'(t) = v_x(t)$ och $dv_x(t)/dt = F_x/m$, där F_x är x–komponenten av den kraft vektor som verkar på bollen vid tiden t. Vi skall uttrycka derivatan $dS(t)/dt$ med hjälp av de funktioner som ingår i $S(t)$.

Uppgift 3.2 ***(Förberedelse)*** För att studera hur bollen påverkas av gravitationskraften tittar vi på ett enkelt exempel. Antag att utgångspositionen är $x_0 = y_0 = z_0 = 0$ och hastighetsvektorn är $V(0) = (17.3, 0, 11.5)$ *m/s*. Skriv upp, och lös, ekvationerna för bollbanan i detta fall. Var landar bollen? **Redovisa beräkningarna.**

Uppgift 3.3 Titta i filen `BollBana.py`. Där finns den Python kod som krävs för att simulera bollbanan då det endast tas hänsyn till gravitationen. Stämmer det som implementerats i koden med din högerledsfunktion från ovan? Kör även simuleringen och anteckna var bollen landar. Stämmer det med den analytiska lösningen ovan?

3.1 Luftmotståndet

I praktiken kommer bollen att bromsas av *Luftmotståndet*. Denna är riktad rakt mot bollens hastighetsvektor, och ges av uttrycket

$$F_D = -\frac{1}{2}\rho A C_D |\vec{v}|\vec{v}, \qquad |\vec{v}| = \sqrt{v_x^2 + v_y^2 + v_z^2},$$

där $A = 0.0375\, m^2$ är bollens tvärsnittsarea, $\rho = 1.2\, kg/m^3$ är luftens densitet, och C_D är en konstant som beror på luftens viskositet och bollens form. Detta uttryck är betydligt mera komplicerat än gravitationskraften och det är inte längre möjligt att lösa problemet analytiskt.

Uppgift 3.4 Ändra i programmet `BollBana.py` så att simuleringen tar hänsyn till luftmotståndet. Använd koefficienten $C_D = 0.17$. Vad händer med bollbanan nu? Var landar bollen? **Redovisa grafen**.

Då man genomför simuleringen är man helt beroende av flera parametrar som är olika svåra att mäta. Gravitationskonstanten g och bollens massa m kan mätas med hög noggranhet. En höghastighetskamera kan dessutom mäta bollens utgångshastighet med bra noggranhet.

Koefficienten C_D som bestämmer luftmotståndet är betydligt svårare att mäta. Koefficienten beror på ett flertal faktorer som luftens densitet, viskositet, samt temperatur, men exakt hur sambandet ser ut är inte helt klarlagt. Istället erbjuder simuleringar med Python, i kombination med mätningar, ett sätt att indirekt mäta C_D.

Uppgift 3.5 Antag att vi i ett experiment skjuter iväg bollar med utgångshastighet

$$V_0 = (17.32, 0.45, 11.51)^T \qquad [m/s].$$

Bollen landar vid $(x_e, y_e) = (23.50, 0.61)$. Övriga parametrar har de värden som givits tidigare i övningen. Använd denna information för att bestämma luftmotståndskoefficienten C_D i det aktuella fallet. **Redovisa ditt värde på** C_D.

Tips Detta sätt att använda känd information, tillsammans med datorsimuleringar, för att indirekt mäta fysikaliska storheter är väldigt vanligt. Då man kan simulera även mycket komplicerade system är det en oerhört kraftfull teknik.

3.2 Skruva bollen in i mål!

Då man tittar på fotbollsmatcher så inser man snabbt att gravitation och luftmotstånd inte kan vara hela sanningen. Isåfall skulle det vara fysikaliskt omöjligt att skruva en fotboll. Exempel är en frispark av brasilianaren Roberto Carlos under en match 1997 (se klippet `https://www.youtube.com/watch?v=crKwlbwvr88`).

Då en boll roterar runt sin egen axel och samtidigt rör sig fås en så kallad *Magnus kraft*. Antag att fotbollen rör sig genom luften med hastighetsvektor $V(t)$ och roterar med rotationsmomentsvektor ω så fås en resulterande kraft

$$F_L = C_L \rho A (\vec{\omega} \times \vec{v}).$$

där ρ är luftens densitet, A är bollens tvärsnittsarea, C_L är en konstant, och $\vec{\omega}$ är rotationsmomentvektorn. Vektor produkten $\vec{\omega} \times \vec{v}$ ger rätt riktning på kraften. Denna kan beräknas med `np.cross()`.

Från filmen kan vi se ungefär var frisparken slogs, see Figur 1. Vi kan även uppskatta starthastighet och riktning. Det som är svårare att uppskatta är hur bollen roterar runt sin egen axel. I denna övning skall vi försöka hitta bollens rotationshastighet genom att simulera bollbanan.

Uppgift 3.6 Använd programmet `BollBana.py` som beräknar, och ritar upp, bollbanan given utgångshastighet V_0 och rotationsmomentvektor $\vec{\omega}$. Det finns även en rad som ritar upp fotbollsmålet. Kopiera denna till den plats där resten av plottningen sker.

Lämpliga värden på utgångshastigheten V_0, densiteten ρ, och övriga parametrar som enkelt kan mätas finns inkluderade i filen. Startgissningen för bollens rotation är satt till ett rent bakåt spin $\vec{\omega} = (0\ {-8.9}\ 0)^T$, vilket ger en rakt uppåt–riktad Magnus kraft. Detta ger fotbollen lite extra lyftkraft och bollen landar rätt mycket längre bort jämfört med om man inte tar hänsyn till Magnuskraften.

Figur 1: Översiktlig bild av straffsparken. Avståndet till målet är ungefär $35\,m$ och straffen slogs med utgångshastighet omkring $140\,km/h$.

Experimentera med olika värden på vektorn $\vec{\omega}$ tills du hittar ett fall där bollen precis går in i mål. Det bör räcka att experimentera olika sidspin, dvs ändra z-komponenten i $\vec{\omega}$. **Redovisa grafen och din vektor $\vec{\omega}$.**

Tips Hur stor effekt den uppåtriktade Magnus kraften har kan du undersöka genom att sätta $\vec{\omega} = (0\,0\,0)^T$. Tänk också på att bilden ser lite konstig ut eftersom z-axeln är utdragen jämfört med x- och y-axlarna. Vrid gärna på figuren för att se tydligare.

Uppgift 3.7 Sambandet mellan rotationsmomentvektor $\vec{\omega}$ och rotationshastighet (i varv/sekund) är

$$s = \frac{|\vec{\omega}|}{2\pi}$$

Hur snabbt måste fotbollen rotera för att kunna skruvas in i mål som på videoklippet?

Ingen matematisk modell av ett fysikaliskt fenomen är perfekt. Då simuleringen är gjord gäller det att bedöma resultatet, samt undersöka möjliga förbättringar.

Uppgift 3.8 Undersök hur känslig modellen är med avseende på mätfel i koefficienten C_L. Gör detta genom att beräkna lösningen för $C_L = 0.15$ och $C_L = 0.17$. Använd den vektor $\vec{\omega}$ som du hittade i föregående uppgift. Hur långt flyttas bollens nedslagsplats?

Uppgift 3.9 Finns alla fysikaliska effekter med i den matematiska modellen? Föreslå åtminstone ett sätt att förbättra simuleringsmodellen.