Erik Dalsryd, Patrick Henning
2018-11-20

SF1511, Numeriska metoder och grundläggande programmering – Laboration 4 –

Ekvationslösning, interpolation och numerisk integration

Enkel Tredimensionell Design

Deadline - 2019-02-08

Efter den här laborationen skall du känna igen problemtyperna linjära och icke-linjära ekvationer, överbestämda linjära ekvationer och minst ett sätt att lösa varje problemtyp samt att kunna uppskatta lösningsnoggrannheten. Du skall också lära dig att interpolera, integrera numeriskt och förstå begreppet noggrannhetsordning. Vidare, du ska kunna utföra enklare tredimensionell design med rotationsfigurer.

1. Icke-linjär skalär ekvation

Man vill bestämma samtliga rötter till följande skalära ekvation,

$$y(x) = x - 4\sin(2x) - 3 = 0.$$

Noggrannheten skall vara minst tio korrekta siffror.

a) Rita grafen för $y(x) = x - 4\sin(2x) - 3$ (med MATLAB). Samtliga nollställen till $y(x)$ skall vara med. Hur många rötter finns det?

b) Undersök empiriskt och teoretiskt vilka av rötterna som kan bestämmas med följande två metoder:

(1) Fixpunktsiterationen $\quad x_{n+1} = -\sin(2x_n) + \frac{5}{4}x_n - \frac{3}{4}$,

(2) Newtons metod.

Önskad noggrannhet är 10 korrekta siffror.

c) För bägge metoderna och för minst två konvergerande rötter skriv ut tabeller som visar hur iteraten konvergerar mot rötterna. Utöka därefter tabellerna så att följande storheter kan avläsas:

(1) Konvergenshastigheten (linjär eller kvadratisk)

(2) Antalet iterationer som krävs för att få rötterna med minst tio siffrors noggrannhet

2. Linjär algebra: robotarm

Figuren visar en robotarm med två länkar. Ledvinklarna ges av θ_1 och θ_2. Koordinaterna för robothanden blir

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

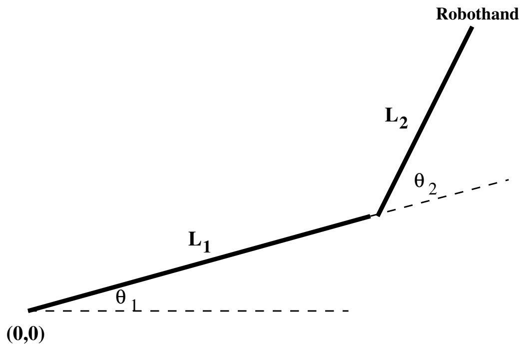
$$y = L_1 \sin\theta_1 + L_2 \sin[\theta_1 + \theta_2)$$

där L_1 och L_2 är länklängderna.
Vinklarna θ_1 och θ_2 som bestämmer robothandens rörelse kontrolleras tidsmässigt av följande polynomuttryck:

$$\theta_1(t) = \theta_1(0) + a_1 t^3 + a_2 t^4$$

$$\theta_2(t) = \theta_2(0) + b_1 t^3 + b_2 t^4$$

där $\theta_1(0)$ och $\theta_2(0)$ är startvärden för vinklarna vid tiden $t = 0$. Vinklarna anges i enheten grader och tiden i sekunder.



a Ställ upp ett linjärt ekvationssystem för bestämning av parametrarna a_1, a_2, b_1 och b_2, givet startvärdena $\theta_1(0) = 10$, $\theta_2(0) = 20$ samt vinklarnas värden då $t = 3$: $\theta_1(3) = 50.5$, $\theta_2(3) = -28.6$ och då $t = 4$: $\theta_1(4) = 131.6$, $\theta_2(4) = -140.0$. Lös ekvationssystemet i MATLAB.

b Använd dina resultat i a) för att plotta robothandens svep när tiden t går från 0 till 4 sekunder, med värdena $L_1 = 4$ m och $L_2 = 3$ m.

3. Interpolation och minstakvadratanpassning

Givet är följande tabell över termisk konduktivitet som funktion av temperaturen för elementet järn

Temperatur, T, (K):	100	200	300	400	500	600	700	800	900	1000
Kond., k, (W/cmK):	1.32	0.94	0.835	0.803	0.694	0.613	0.547	0.487	0.433	0.38

Egenskriven kod och \ ska användas, inte Matlabs färdiga funktioner.

a) Utnyttja de fyra värdena vid temperaturen 100, 400, 700 och 1000 och interpolera genom dem med ett tredjegradspolynom. Rita de fyra givna punkterna och polynomkurvan med fin diskretisering (100:20:1000).
Hur stor är konduktiviteten, k när temperaturen $T = 300K$? Jämför med det uppmätta värdet. Varför använder vi inte alla punkterna och interpolerar med ett niondegradspolynom ?

b) Anpassa ett andragradspolynom i minstakvadratmening till de tio givna data. Plotta på samma sätt som ovan och ange konduktiviteten för temperaturen $T = 300K$ enligt denna modell. Jämför med det uppmätta värdet. Beräkna även felkvadratsumman.

c) Gör om samma beräkningar som i **b)** för ett tredjegradsploynom. Är anpassningen bättre än i b)?

4. Numerisk integration

Denna del av laborationen handlar om numerisk integrering. Vi önskar beräkna ett integralvärde

$$I(f) = \int_a^b f(x)dx$$

med hjälp av någon kvadraturformel. De kvadraturformler som behandlas i denna kurs kan alla skrivas på formen

$$I_{approx}(f) = \sum_{k=0}^{M} \alpha_k f(x_k)$$

där α_k kallas kvadraturvikter och x_k kallas kvadraturpunkter. M är antalet delintervall. Detta ger att steglängden $H = (b-a)/M$.
Kom ihåg att alltid börja med att rita integranden, $f(x)$, över integrationsintervallet för att se hur $f(x)$ beter sig på intervallet.

a) Beräkna följande integral

$$I(f) = \int_{-1}^{1} \sqrt{x+4}dx$$

(1) exakt med penna och papper.

Lös sedan integralen numeriskt med

(2) trapetsregeln (egenskriven) och steglängderna $h = 2$, $h = 1$ och $h = 0.5$,

(3) trapetsregeln följd av Richardsonextrapolation.

Undersök hur approximationsfelet, E_H beror av steglängden, H, genom att plotta felkurvor som funktion av steglängen. Använd MATLABs kommando `loglog` för plottarna.
Uppskatta metodernas noggrannhetsordning, p, med hjälp av dessa plottar. För en metods noggrannhetsordning gäller

$$|E_H| = |I(f) - I_{approx}(f)| \leq \text{konstant}H^p.$$

där $I_{approx}(f)$ är det numeriskt uträknade integralvärdet med antingen trapetsregeln eller Richardson extrapolation. Stämmer detta med vad som gäller för felen enligt formlerna i kursboken?

b) I den här delen skall du räkna ut svängningtiden för en pendel. En pendels svängningstid T beror av utslagsvinkeln φ_0 enligt formeln:

$$T = 4\sqrt{\frac{L}{g}} I(\varphi_0)$$

där L är pendelns längd, g är tyngdaccelerationen och

$$I(\varphi_0) = \int_0^{\pi/2} \frac{d\alpha}{\sqrt{1-k^2(\sin\alpha)^2}}, \quad k = \sin\frac{\varphi_0}{2}$$

Låt $L = 1\ m$ och $g = 9.81\ m/s^2$. Skriv ett program som beräknar svängningstiden T för φ_0-värdena 0,5,10,....90 grader. För att beräkna integralen $I(\varphi_0)$ används trapetsregeln, egenskriven. Plotta resultatet, dvs T som funktion av φ_0 i en graf.

En ofta använd approximation av T är svängningstiden för små svängningar:

$$T_s = 2\pi\sqrt{\frac{L}{g}}$$

Denna approximation är bra för små utslagsvinklar, men relativfelet ökar med ökande utslagsvinkel. Plotta även en graf som visar relativfelet R_{T_s} som funktion av utslagsvinkeln då $0 \leq \varphi_0 \leq 90$ grader.

5. Tredimensionell design

a) Skapa en kolumnvektor z med element från -14 till 5 med något lämpligt antal steg, och definiera funktionen $x = 2 - \arctan(z)$.
Konstruera nu en radvektor ϕ med element från 0 till 2π.
Med satserna

$$X = x\cos(\phi), \quad Y = x\sin(\phi), \quad Z = z * ones(size(\phi)), \quad surfl(X, Y, Z), \quad axis\ \ equal$$

skapas en tredimensionell rotationsfigur.
Vad föreställer figuren du nu har konstruerat?

Prova gärna kommandona `axis off`, `shading flat` och `colormap hot`.

b) Skriv ett Matlab-program där värdet n = antalet punkter matas in, och därefter n stycken $(x, y)-$koordinatpar, med hjälp av `[x,y]=ginput(n)`.
Ett interpolationspolynom (gradtal $n - 1$) ska läggas genom dina inmatade punkter.
Ditt program ska beräkna koefficienterna i detta polynom. Kör programmet och rätta till eventuella fel innan du fortsätter.

Konstruera nu en radvektor ϕ med element från 0 till 2π, t.ex. med 20 steg.
Designa rotationsfiguren där ditt interpolerande polynom roterar kring $x-$axeln: Först definiera en kolumnvektor X med elementen som löper från x_1 till x_n, med samma antal komponenter som radvektorn ϕ.
Definiera sedan en kolumnvektor Y med interpolationspolynomets värden för alla element i X.

Med satserna

```
Xplot=X*ones(size(fi))
Yplot=Y*cos(fi)
Zplot=Y*sin(fi)

surfl(Xplot, Yplot, Zplot)
```

kan en figur nu plottas.
Använd gärna `axis off`, och vänd på figuren.

Alternativ beräkning:
I stället för att använda `ginput`, får du rita en personlig rotationsfigur som skapas med hjälp av ditt personnummer, om du föredrar.
Använd i så fall `X=[1:10]'` och siffrorna i ditt personnummer som 10 stycken element i vektorn Y.
Lägg ett interpolationspolynom genom punkterna, och rita en rotationsfigur enligt ovan. Uppstår Runges fenomen?

Om du vill, kan du experimentera med olika steglängder och färgskalor.