KTH
VETENSKAP
OCH KONST

KTH Engineering Sciences

Laboration 1: Modul 1: Icke-linjära ekvationer, Modul 2: Linjära ekvationssystem, Modul 3: Kurvanpassning och Modul 4: Numerisk integrering

Målsättningen med denna laboration är att du ska få förståelse för olika metoder för ekvationslösning, kurvanpassning och numerisk integrering. Laborationen består av tre uppgifter. Metoder som behandlas är

- Fixpunktsmetoden
- Newtons metod för olinjära skalära ekvationer
- Lösning av linjära ekvationssystem med Gausseliminering
- Minstakvadratmetoden
- Interpolation
- Trapetsregeln
- Simpsonsregel
- Richardsonextrapolation

Viktiga koncept som behandlas i laborationen är

- Konvergensordning
- Noggrannhetsordning

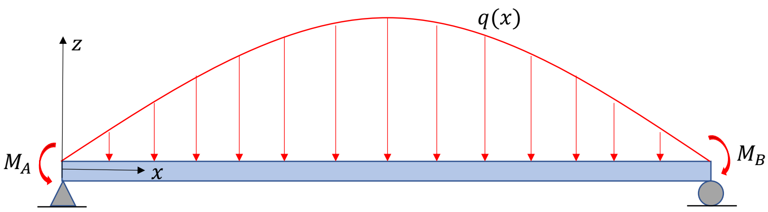
Följande specifika kunskaper i PYTHON är bra att ha inför arbetet med laborationen

- Funktioner
- `while`-loopar eller `for`-loopar
- Hantering av vektorer och matriser
  - Skapa vektorer av bestäm längd
  - Skapa matriser
  - Beräkna normen av en vektor
  - Lösa linjära ekvationssystem
  - Summera element i en vektor
- Olika plot-funktioner

Tips: Tänk på att skriva Python-programmen/funktionerna på ett sådant sätt att det är enkelt att använda samma program/funktion för snarlika problem. Exempelvis bör ett program för Newtons metod enkelt kunna modifieras beroende på vilken ekvation man vill hitta nollställe till.

1 Balkböjning - Modul 1: icke-linjär ekvation [1]

En balk med böjstyvhet EI och längd L ligger fritt upplagd på två stöd. Balken belastas med moment i bägge ändarna (M_A och M_B) och en utbredd last $q(x) = -q_0 \sin(\pi x/L)$ längs balken, se Fig. 1.



Figur 1: En fritt upplagd balk som belastas med moment och utbredd last.

På grund av momenten och lasten kommer balken att böja sig och den vertikala utböjningen (deformationen) ges av

$$u_z(x) = \frac{L^2}{EI}\left[\frac{M_A}{6}\left(2\frac{x}{L} - 3\left(\frac{x}{L}\right)^2 + \left(\frac{x}{L}\right)^3\right) + \frac{M_b}{6}\left(\frac{x}{L} - \left(\frac{x}{L}\right)^3\right) - \frac{q_0 L^2}{\pi^4}\sin\left(\frac{\pi x}{L}\right)\right]$$

Om vi inför en lastparameter M_0 och sätter

$$M_A = M_0, \qquad M_B = \frac{2}{3}M_0, \qquad q_0 = \frac{1}{9}\frac{M_0\pi^4}{L^2}$$

får vi

$$u_z(x) = \frac{M_0 L^2}{6EI}\left[\frac{8}{3}\frac{x}{L} - 3\left(\frac{x}{L}\right)^2 + \frac{1}{3}\left(\frac{x}{L}\right)^3 - \frac{2}{3}\sin\left(\frac{\pi x}{L}\right)\right]$$

där funktionen

$$f(x) = \frac{8}{3}\frac{x}{L} - 3\left(\frac{x}{L}\right)^2 + \frac{1}{3}\left(\frac{x}{L}\right)^3 - \frac{2}{3}\sin\left(\frac{\pi x}{L}\right) \tag{1}$$

bestämmer utböjningsformen.

Utböjningen kommer att bli noll i några punkter längs balken (x-axeln). Dels vid stöden men även innanför stöden. Nu gäller det att bestämma var på balken utböjningen blir noll dvs hitta de värden på x för vilka $f(x) = 0$.

UPPGIFT 1

a) Sätt $L = 1$ och ta reda på hur många nollställen ekvation (1) har på intervallet $0 < x < L$ genom att rita upp funktionen i PYTHON.

b) För att bestämma nollställena numeriskt ställer vi upp följande fixpunktsiteration

$$x_{n+1} = \frac{3L}{8}\left[3\left(\frac{x_n}{L}\right)^2 - \frac{1}{3}\left(\frac{x_n}{L}\right)^3 + \frac{2}{3}\sin\left(\frac{\pi x_n}{L}\right)\right] \qquad n = 0, 1, 2\ldots \tag{2}$$

Vilket eller vilka av nollställen kan bestämmas med denna metod? Motivera ditt svar teoretiskt.

[1] I samarbete med Carl Dahlberg, Hållfasthetslära, KTH

c) Skriv ett PYTHON-program som beräknar ett av de nollställen på intervallet $0 < x < L$ för vilka fixpunktsmetoden konvergerar.

Programmet ska ha en utskrift som visar hur metoden konvergerar till nollstället samt returnera ett svar med ett fel som är mindre än en given tolerans $\tau = 10^{-10}$. Använd lämpligt avbrottsvillkor för att säkerställa detta.

d) Utöka PYTHON-programmet ovan så att det beräknar ett av nollställena på intervallet $0 < x < L$ till (1) med hjälp av Newtons metod. Välj ett nollställe som inte går att bestämma med fixpunktsmetoden.

Programmet ska ha en utskrift som visar hur metoden konvergerar till nollstället samt returnera ett svar med ett fel som är mindre än en given tolerans $\tau = 10^{-10}$. Använd lämpligt avbrottsvillkor för att säkerställa detta.

e) Nu ska konvergenshastigheten för de två metoderna jämföras. Välj ett nollställe där bägge metoderna konvergerar. Använd samma startgissning för båda fallen. Beräkna nollstället och plotta $|x_{n+1} - x_n|$ som funktion av n för bägge metoderna i samma figur. Du kan använda en `matplotlib.pyplot.semilogy`-plot för att få logskala på y-axeln. Vad kan man utläsa av figuren? Ser figuren ut som du förväntar dig?

2 Solens upp- och nedgång - Modul 2: linjära ekvationssystem och Modul 3: Kurvanpassning

Tabell 1 innehåller medelvärdet i minuter som solen är uppe i Stockholm under årets tolv månader. I figuren 2 ses data för varje dag tillsammans med de beräknade medelvärdena. [2]

t [månad]	1	2	3	4	5	6	7	8	9	10	11	12
Soltid [min]	421	553	709	871	1021	1109	1066	929	771	612	463	374

Tabell 1: Soltid i minuter per månad.

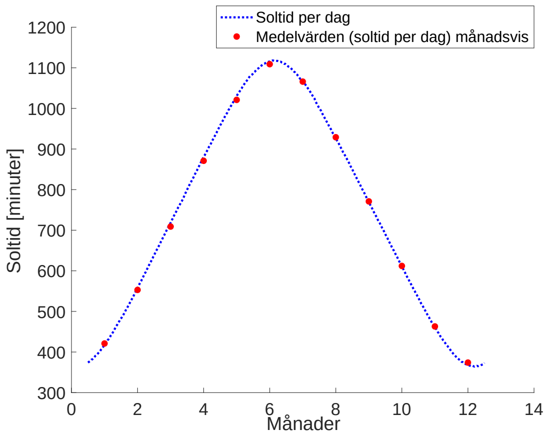
UPPGIFT 2

I denna uppgift ska du skriva ett PYTHON-program som gör kurvanpassningarna i uppgifterna **a)**, **c)**-**e)** till data i tabellen.

För varje kurvanpassning:

- plotta den anpassade modellen tillsammans med data från tabellen,
- beräkna minstakvadratfelet (i kvadrat),
- du får använda den inbyggda funtionen `numpy.vander` i uppgift **a)** men du får inte använda följande Python-funktioner: `numpy.polyfit`; `numpy.polyval`,
- du får inte använda den inbyggda funtionen `scipy.optimize.curve_fit` i uppgift **c till e)**.

[2] Data för varje dag kommer från *https://www.astroinfo.se/tider-for-solen-och-manen/solens-upp-och-nedgangar-for-hela-aret/*. Medelvärden är beräknade per månad och avrundade till heltal.



Figur 2: Soltid i minuter per dag och i medelvärde

a) Modell: ett interpolationspolynom till alla data. Vilket gradtal blir det på polynomet? Beräkna interpolationspolynomet på tre olika sätt:

- Naiv ansats: $p_1(t) = c_0 + c_1 t + c_2 t^2 + \cdots$
- Centrerad ansats: $p_2(t) = c_0 + c_1(t - t_m) + c_2(t - t_m)^2 \cdots$, där t_m är medelvärdet i tidsvektorn $t = [1\ \ 2 \cdots 12]^T$.
- Newtons ansats: $p_3(t) = c_0 + c_1(t - t_1) + c_2(t - t_1)(t - t_2) \cdots$, där t_1 är komponent 1 i tidsvektorn t osv.

Evaluera polynomen p_1, p_2, p_3 i 1000 punkter mellan 0 och 12 och räkna ut maximala skillnaden på intervallet mellan p_1, p_2, p_3. Du behöver bara plotta polynomet för en av ansatserna ovan. Varför?

b) Beräkna konditionstalen i maxnorm för matriserna som uppstår med de olika ansatserna i **a)**.

c) Modell: ett minstakvadratanpassat andragradspolynom där du använder data från april till augusti.

d) Modell: ett minstakvadratanpassat tredjegradspolynom där du använder data från april till augusti.

e) Modell: en minstakvadratanpassad trigonometrisk funktion med ansatsen

$$p(t) = d_0 + d_1 \cos(\omega t) + d_2 \sin(\omega t) \quad \text{där} \quad \omega = 2\pi/12$$

f) Gör nu om **b)**-uppgiften för matrisen i normalekvationerna i **c)-e)**. Jämför med dina resultat i **b)** och kommentera på resultaten.

Diskutera: Vilken av modellerna i uppgifterna **a)**, **c)-e)** skulle du rekommendera för att bestämma hur många minuter solen är uppe under sommaren och under vintern? Motivera ditt svar.

3 Effekt från solceller - Modul 4: numerisk integrering (och lite om Modul 3: kurvanpassning)

I tabell 2 finns värden på effekten $f(t)$, avrundade till heltal, från ett pilotprojekt med solceller. Genom att skatta integralen

$$\int_{2014}^{2022} f(t)dt \tag{3}$$

vill vi bestämma totala mängden energi mätt i kWår som producerats under åren 2014-2022 och även bedöma om det är värt att gå vidare med ett större projekt.

t [år]	2014	2015	2016	2017	2018	2019	2020	2021	2022
$f(t)$ [kW]	12.00	15.10	19.01	23.92	30.11	37.90	47.70	60.03	75.56

Tabell 2: Uppmätta värden för t och $f(t)$.

UPPGIFT 3

Skriv ett PYTHON-program som löser uppgifterna nedan.

a) Skriv en funktion `trapets.py` som numeriskt approximerar en integral trapetsregeln. Funktionen ska ta ett funktionshandtag, antal intervall och en vektor med integralgränser som indata. Verifiera att din funktion fungerar genom testa den på integralen

$$I = \int_0^2 x^3 e^x dx = 6 + 2e^2 \approx 20.7781. \tag{4}$$

b) Gör en konvergensstudie för trapetsregeln applicerad på integralen (4). Beräkna felet för en skattning med trapetsregeln med steglängd h, T_h, enligt $e_h = |I - T_h|$. Beräkna noggrannhetsordningen för trapetsregeln med hjälp av steglängdshalvering och kontrollera att resultatet stämmer väl med teorin.

c) Låt steglängden $h = 1$ år och skatta integralen (3) med trapetsregeln. Inspireras gärna av den funktion du skrivit i **a)**. Vad blir det skattade värdet på totala mängden energi under åren 2014-2022?

d) För integralen (3) har vi vare sig en exakt lösning eller en kontinuerlig funktion för $f(t)$. Vi kan ändå kontrollera att metoden är korrekt implementerad genom att göra en konvergensstudie med befintliga data från tabell 2. Gör en konvergensstudie genom att skatta integralen med trapetsregeln med fyra olika värden på h. Uppskatta felen som skillnaden mellan två skattningar med h respektive $2h$, dvs $e_{2h} \approx |T_h - T_{2h}|$. Kontrollera att noggrannhetsordningen för trapetsregeln stämmer rimligt väl med teorin.

e) Gör en mer noggrann skattning genom Richardssonextrapolation av de två mest noggranna skattningarna med trapetsregeln. Approximera sedan integralen (3) med Simpsons regel och $h = 1$. Verifiera att de två sätten ger samma resultat.

f) Antag att $f(t)$ kan beskrivas med en exponentiell modell på formen $f(t) = ae^{b(t-2014)}$. Linjärisera modellen och gör en linjär minstakvadratanpassning för att bestämma koefficienterna a och b.

g) Använd den exponentiella modellen från **f)** för att uppskatta hur mycket energi som produceras år 2023. Lägg sedan till detta värde i tabellen och skatta $\int_{2014}^{2023} f(t)dt$. Pilotprojektet anses lyckat om minst ett av följande villkor är uppfyllda: effekten under 2023 överstiger 100 kW eller totala mängden energi under 2014-2023 överstiger 350 kWår. Är något eller båda villkoren uppfyllda?

4 Inför redosvisning

Samtliga program bör kontrolleras så att de exekveras felfritt innan redovisningen. Dessa kontroller kan med fördel genomföras i ett tidigt skede, dels som förberedelse inför laborationen, dels för att säkerställa funktionaliteten i samtliga uppgifter. Inför den muntliga redovisningen gäller att samtliga Python-filer som har använts ska vara tillgängliga på den dator som används vid redovisningen. Filerna ska vara tydligt namngivna för att möjliggöra en effektiv åtkomst och körning. Samtliga filer ska dessutom laddas upp via Canvas före redovisningstillfället. Under redovisningen ska ni, individuellt, kunna redogöra för den teori och de algortimer ni har använt. Ni ska också kunna svara på frågor som ställs under redovisningen och förklara hur era Python-program fungerar. **Kom väl förberedda!**