Semesteroppgave

TMA4215 Numerisk Matematikk
Gruppe nr. 6
Kristian Stormark og Bjørn E. Vesterdal

Høst 2003

Sammendrag Fra datasettet med termodynamiske egenskaper for vann ved konstant temperatur 25°C fikk gruppen vår tildelt de tabulerte verdiene for trykk og tetthet, og da spesielt de verdiene med trykk i intervallet [105,200] bar. Ved hjelp av polynominterpolasjon og naturlige kubiske spliner har vi med trykkverdiene som abscisser kurvetilpasset dette datasettet. Til polynominterpolasjonen brukte vi Nevilles algoritme, noe modifisert i forhold til slik den er presentert i læreboka [B&F]. Selve implementeringen av algoritmene har vi gjort i Matlab.

1. Generelt om interpolasjon

Gitt et sett datapunkter. Dersom man antar at disse punktene kommer fra en eller flere kontinuerlige fordelingsfunksjoner kan man tenke seg at de kjente punktene vil kunne brukes til å komme med plausible approksimasjoner for ytterligere punkter. En vil videre forvente at disse approksimasjonene vil være best nær de tabulerte punktene.
En av måtene å gjøre slike approksimasjoner på er å finne en eller flere funksjoner som tilfredsstiller visse spesifiserte krav, og da som et typisk minimum at en evaluering av funksjonen(e) for de tabulerte abscisse-verdiene vil være konsistente med de gitte datapunktene.
En av de mest brukte typene funksjoner for slike approksimasjoner er polynomer, og da hovedsaklig på grunn av at disse gis et teoretisk grunnlag for approksimasjon av kontinuerlige funksjoner ved Weierstrass' teorem, samt at disse er forholdsvis simpele å derivere og integrere.
De to interpolasjonsteknikkene vi har brukt i denne oppgave bygger begge på polynomer. Lagrange-polynomene som Nevilles algoritme bygger på genererer et polynom som intererpolerer hele datasettet, mens de naturlige kubiske splinene genererer et polynom som gjelder mellom hver par av de gitte datapunktene.

2.0 Metode-presentasjon: Nevilles algoritme

Gitt et sett av *n+1* distinkte abscisse-punkter med tilhørende funksjonsverdier, dvs. $\{(x_i, f_i)\}$, for $i = 0, 1, ..., n$, hvor $f_i = f(x_i)$.
Da finnes det ved teorem 3.2 i læreboka [B&F] *ett unikt* polynom av grad n eller lavere som interpolerer disse punktene. Dette kalles det *n-te Lagrange-interpolasjonspolynomet.*
Disse polynomene er entydig gitt ved hjelp av forholdsvis enkle formler (jfr. læreboka [B&F]), og det er derfor uproblematisk å generere disse. Videre kan det utledes et estimat for feilen i approksimeringen, men dette krever kjennskap til den *n+1*-te deriverte til funksjonen som approksimeres, noe som generelt ikke er tilfelle.
Derimot er det vanlig å anta at en ved å øke antall datapunkter som brukes i interpoleringen vil få et bedre estimat. I så måte brukes differansen mellom et gitt estimat og estimatet som fåes ved å inkludere ett ytterligger datapunkt som feilskranke for aproksimasjonen. Ved bruk av de generelle formlene for Lagrange-interpolasjonspolynomene genereres polynomene sekvensielt. Og da det derfor ikke er mulig å evaluere denne feilskranken før de to påfølgende polynomene faktisk er genererte, er dette en relativt tidkrevedne prosess som resulterer i tilsynelatende mye unødig arbeid.
Men i stedet for å bruke de eksplisitte formlene for å generere Lagrange-interpolasjonspolynomene kan en bruke en rekursiv metode kalt *Nevilles algoritme.* Denne algoritmen benytter seg av de lavere ordens polynomene til å generere et polynom av én grad høyere. På denne måten reduseres arbeidsmengden som kreves for å inkludere et nytt datapunkt betraktelig. Det teoretiske resultatet som danner basis for Nevilles algoritme vil her bli kort referert.
Av hensyn til oversiktelighet i teksten innføres først følgende notasjon for Lagrang-interpolasjonspolynomer:

Definisjon 1 *Gitt datasettet* $\{(x_i, f_i)\}$*, for* $i = 0, 1, ..., n$*, og la* $m_1, m_2, ..., m_g$ *være distinkte heltall i intervallet* $[0, n]$*. Lagrange-interpolasjonspolynomet som interpolerer de g punktene* $\{(x_t, f_t)\}$*, for* $t = m_1, m_2, ..., m_g$ *betegnes da som* $P_{m_1,m_2,...,m_g}$*.*

Med denne notasjonen kan den rekursive polynom-genereringen i Nevilles algoritme uttrykkes på følgende måte:
Gitt datasett $\{(x_b, f_b)\}$, for $b = 0, 1, ..., k$. La x_i og x_j være to distinkte abscissepunkter i dette settet. Da vil

$$P(x) = \frac{(x - x_j)P_{0,1,...,j-1,j+1,...k}(x) - (x - x_i)P_{0,1,...,i-1,i+1,...k}(x)}{x_i - x_j} \qquad (1)$$

ved teorem 3.5 i læreboka [B&F] være det k-te Lagrange-interpolasjonspolynomet som interpolerer punktene $\{(x_b, f_b)\}$, for $b = 0, 1, ..., k$.

2.1 Kort om implementering av Nevilles algoritme

Selve implementeringen av Nevilles algoritme ble gjort ved å lage en Matlab-metode som tar inn et datasett, ønsket approksimasjonspunkt, samt en grenseverdi for ønsket approksimasjonspresisjon som input-parametre. Metoden forutsetter datasettet sortert mhp. abscisse-punktene. Videre er det valgt å la metoden kreve at approksimasjonspunktet er i det lukkede intervallet mellom laveste og høyeste abscissepunkt.
Etter en del initielle tester av input-parametrene, samt opprettelse av en matrise for mellom-lagring lokaliserer metoden så det abcsissepunktet som ligger nærmest approksimasjonspunktet på oversiden. Så tilordnes den tilhørende funksjonsverdien til det korresponderende 0.grads-polynomet som interpolerer dette punktet. Deretter starter en løkke som suksessivt legger til nye punkter, fortrinnsvis alternerende mellom undersiden og oversiden, og nye approksimasjoner gjøres vha. (1).
For hver iterasjon sjekkes i tillegg endringen i estimatet etter at det siste punkt ble inkludert mot den spesifiserte grenseverdien for ønsket presisjon, og dersom denne betingelsen skulle vise seg å være ivaretatt bryter løkken av og returnerer approksimasjonen. I motsatt fall itererer løkken videre til alle datapunktene er tatt med og returnerer *deretter* den siste approksimasjonen. Videre returnerer metoden også i alle tilfeller endring i estimat etter siste inkluderte punkt, antall datapunkt som ble brukt i estimeringen (dvs. graden på interpolasjonspolynomet - 1), samt antall resterende datapunkt. I de tilfellene hvor den ønskede presisjonen ikke ble oppnådd returneres NaN for endringen i estimatet.

3.0 Metode-presentasjon: Naturlige kubiske spliner

Interpolasjon ved kubiske spliner er en metode der man interpolerer et sett av $n+1$ datapunkter, dvs. $\{(x_i, f_i)\}$, for $i = 0, 1, ..., n$, hvor $f_i = f(x_i)$, med n polynomer av grad 3, der hvert delintervall $[x_j, x_{j+1}]$ har et eget polynom. Disse polynomene tilsammen kalles en kubisk spline S. S er definert som følger:

Definisjon 2 *Gitt en funksjon f definert på [a,b] og et sett skjøter $a = x_0 < x_1 < ... < x_n = b$, er en kubisk spline S for f en funksjon som oppfyller følgende krav:*

a. $S(x)$ er et kubisk polynom, gitt som $S_j(x)$, på delintervallet $[x_j, x_{j+1}]$ for alle $j = 0, 1, ..., n-2$
b.$S(x_j) = f(x_j)$ for alle $j = 0, 1, ..., n$
c.$S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ for alle $j = 0, 1, ..., n-2$
d.$S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ for alle $j = 0, 1, ..., n-2$
e.$S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ for alle $j = 0, 1, ..., n-2$
f.$S''(x_0) = S''(x_n) = 0$

Det man trenger er koeffisientene i de kubiske polynomene

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \tag{2}$$

for alle $j = 0, 1, ..., n-1$.
a_j-ene er kjent, siden disse er funksjonsverdiene i skjøtene. Ved å benytte (c) og definere $h_j = x_{j+1} - x_j$ og $a_n = f(x_n)$ fås uttrykket

$$a_{j+1} = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3 \tag{3}$$

Et uttrykk for b_j-ene kan finnes ved å derivere (2), definere $b_n = S'(x_n)$ og benytte (d) for å få

$$b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2 \tag{4}$$

På samme måte kan man derivere (2) igjen og definere $c_n = S''(x_n)/2$ og anvende (e). Da er c_j-ene gitt ved

$$c_{j+1} = c_j + 3d_j h_j \tag{5}$$

Ved å kombinere likningene (3), (4) og (5) kan man sette inn for alle b_j-er og d_j-er og ende opp med et system av $n-1$ lineære likninger uttrykt ved

$$h_{j-1}c_{j-1} + 2(h_{j-1} + h_j)c_j + h_j c_{j+1} = \frac{3}{h_j}(a_{j+1} - a_j) - \frac{3}{h_{j-1}}(a_j - a_{j-1}) \tag{6}$$

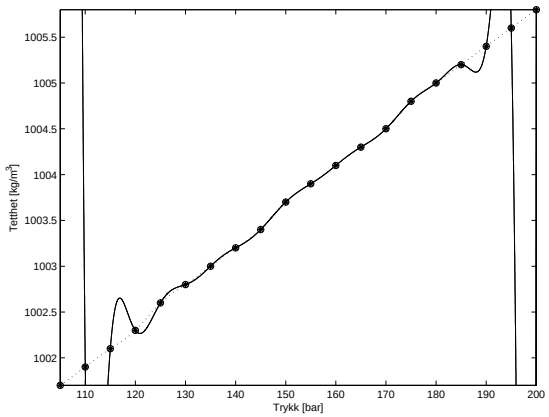
Dette systemet kan representeres som et tridiagonalt likningssystem $Ax = b$, der x er koeffisientene $c_1, ..., c_{n-1}$. c_0 og c_n er per definisjon lik 0, fra (f). Elementene i b er gitt ved høyresiden i (6). c_i-koeffisientene kan derfor bestemmes ved å løse dette ligningssystemet. Og med h_i allerede kjent kan d_i da finnes fra (5), og deretter følger b_j fra (3).

3.1 Kort om implementering av kubiske spliner

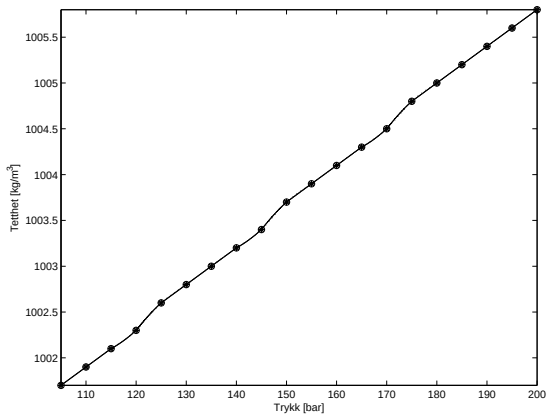
Algoritmen har som mål å regne ut koeffisientene b_j, c_j og d_j i (2), og metoden kalles på det gitte datasettet av skjøter og funksjonsverdier. Det første metoden gjør er å legge skjøtene inn i en vektor x og funksjonsverdiene inn i en vektor a. Deretter setter den opp en h-vektor av differansene mellom de gitte x-verdiene og genererer b-vektoren i ligningssystemet fra (6). Elementene i båndmatrisen A er gitt ved h_j-ene fra høyresiden til (6) og disse benyttes til å løse systemet vha. Crout-faktorisering for tridiagonale lineære likningssystemer, slik at man får ut c_j-ene. Crout-faktoriseringen er implemetert i metoden. Deretter løses b_j-ene og d_j-ene enkelt ut ved hjelp av (3), (4) og (5).

4.0 Resultater

Evaluering av de implementerte algoritmene på det spesifiserte intervallet resulterte i følgende kurver for tetthet plottet mot trykk. For hvert par av de trykkverdiene som benyttes i interpolasjonen approksimeres funksjonsverdien i ca 1000 mellomliggende punkter. Neville-interpolasjonspolynomet er representert ved den heltrukne linjen, og den kubiske splinen ved den stiplede linjen. Figuren viser tettheten som tilnærmet proporsjonal med trykket over hele intervallet som ble undersøkt.



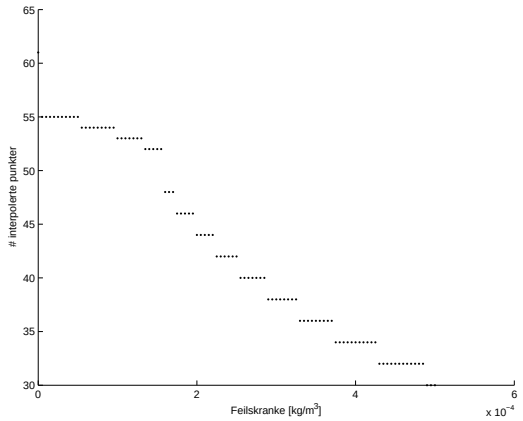
Figur 1: Feilskranke = 0



Figur 2: Feilskranke $= 1 \cdot 10^{-4}$

I figur 1 er Neville-interpolasjonspolynomet beregnet med en feilskranke på 0. Dette gir store fluktasjoner nær endepunktene av intervallet, og splinen gir trolig en bedre tilnærming i disse områdene. I figur 2 er derimot Neville-interpolasjonspolynomet beregnet med en feilskranke på $2.5 \cdot 10^4$, noe som eliminerer dette problemet fullstendig. Fenomenet skyldes at forespørselen

om absolutt presisjonen som regel vil tvinge metoden til å måtte interpolere samtlige gitte datapunkter i et *forsøk på* å oppnå dette, og graden på det interpolerende polynomet vil da bli tilsvarende høy. Mens ved å redusere den forespurte nøyaktighet minimalt vil metoden kunne komme med approskimasjonene på grunnlag av lavere-ordens polynomer og fluktasjonene på endene unngås. Faktisk ble det endelige estimatet av approksimasjonsfeilen generelt sett større når metoden ble kalt med 0 som feilskranke enn når den ble kalt me $2.5 \cdot 10^{-4}$ som feilskranke.
En enkel sammenlikning av tidsbruken til de to metodene viste at spline-metoden kjørte betraktelig raskere enn det Neville-metoden gjorde, og tidsbruken for spline-metoden var i langt mindre grad avhengig av antall punkter beregnet til plottingen. Dette er ikke videre overraskende, siden spline-metoden genererer de forskjellige polynomer én gang, for deretter å evaluere disse direkte. Neville-metoden må derimot i praksis regne ut et nytt polynom for hvert eneste punkt som skal evalueres. Ved et stort antall punkter tar dette naturlig nok relativt lang tid, og flere punkter gir en betydelig økning i tidsbruken.
Figur 3 viser grad på polynomet som Neville-metoden genererer i ett punkt



Figur 3:

(P=116.5 Pa) for varierende feilskranke. Figuren illustrerer tydelig hvordan minkende feilskranke fører til at metoden må inkludere stadig flere punkter, og dermed lage høyere grads polynomer. Derimot er figuren ikke representativ for alle punktene i settet, da dette settet i stor grad er linært og karakteristisk sett består av en rett linje med noen få, beskjedne knekk på. I områdene nært disse "knekkpunktene" er oppførselen typisk slik som vist på figur 3, men i mer enn ett abcsisse-punkts avstand fra nærmerste slike punkt så oppfører metoden seg litt anderledes. Da bruker den enten alle inerpolasjonspunktene dersom den blir kalt med feil-parameteren satt til 0, ellers bruker den 3.

5.0 Konklusjon

I denne oppgaven har de naturlige kubiske splinene vist seg bedre egnet til å interpolere det utdelte datasettet enn Neville-interpolasjonspolynomene. Splinene gir videre vesentlig kortere beregningstid enn Neville, og en unngår problemer med fluktasjoner nær endepunktene.

Kilder:


```
[B&F] - Richard L. Burden & J. Douglas Faires
        Numerical Analysis
        Brooks/Cole 2001
```