Numerische Mathematik für StudentInnen mit Ziel Diplom in Mathematik

Gerhard Opfer WS 1997/1998 (1. Teil) SS 1998 (2. Teil)

Bogen 10: Übungen am 27. Januar 1998

Lösungshinweise

Aufgabe P28. (a) Ein Spline s ist eine auf einem Gitter $t_j < t_{j+1}$, $j = 0, 1, \ldots, n$ (oder $j \in \mathbb{Z}$, so daß $\{t_j\}$ keine Häufungspunkte besitzt) definierte, reelle Funktion, so daß die Restriktionen auf die offenen Intervalle $]t_j, t_{j+1}[$ Polynome sind. (b) Liegen alle Restriktionen von s in Π_{k-1}, $k \in \mathbb{N}$, so sagen wir, daß s die Ordnung k hat. (c) Der Raum $\mathbb{S}_2$ ist ein endlich dimensionaler Vektorraum mit Dimension nk, wenn das Gitter aus n Intervallen besteht.

Aufgabe P29. (a) Ein Spline s heißt (in unserem Kontext) interpolierend, wenn $s(t_j) = s_j$ für vorgegebene Werte s_j, $j = 0, 1, \ldots, n$. Die Werte werden also an den Knoten t_j vorgegeben. (b) Formel (3.54), S. 56. (c) Für die Fehler gibt es je nach Voraussetzungen drei Formeln: (3.59), (3.61), (3.63).

Aufgabe P30. (a) Ja. (b), (c) Gibt man s nur an einer einzigen weiteren Stelle $t \in [a, b]$, $t \neq m_j$ für alle j vor, so ist s eindeutig bestimmt. Sei zuerst $t \neq b$ und $s(t) = \tilde{s}$. Dann gibt es eindeutig ein j mit $t \in [t_j, t_{j+1}[$, und s ist in $[t_j, t_{j+1}[$ durch die beiden Werte $s(m_j) = f(m_j)$ und $s(t) = \tilde{s}$ eindeutig bestimmt. Damit sind wegen der Stetigkeit auch $s(t_j), s(t_{j+1})$ eindeutig bestimmt und somit auch s in den Nachbarintervallen von $[t_j, t_{j+1}[$, und damit in allen Intervallen. Ist $s(b)$ vorgebeben, so ist auch s in $[t_{j-1}, t_j]$, $j = n, n-1, \ldots, 1$ eindeutig bestimmt.

Aufgabe 19. Das Programm mit einigen Testergebnissen kann so aussehen:

Programm L10.1. Pascal-Programm zur Berechnung von lokalen, linearen und kubischen Splines

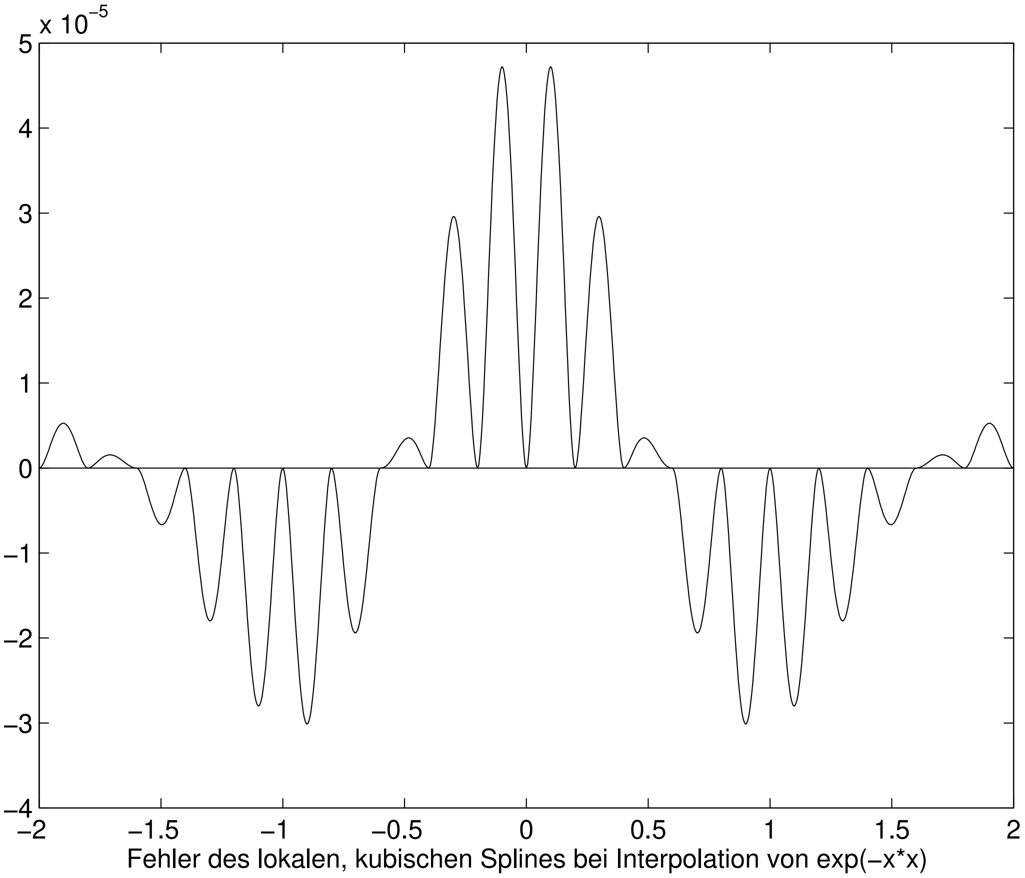
```
{Aufgab 19, Numer Math WS 1997/1998}
{Es geht darum x^5 in [0,1] und exp(-x^2) in [0,b], b=1, 1.25, 1.5, 1.75, 2}
{durch lineare und lokale, kubische Splines quidistant zu interpolieren.}
program Spline_Interpolation;
const gross=1000;
type  vektor=array[0..gross] of real;   matrix=array[1..6,0..8] of real;
var   Erg1,Erg3,alpha1,alpha3:matrix;

function linearer_Spline(x,tl,tr,fl,fr:real):real;
{Der lineare Spline auf [xl,xr] mit Werten fl, fr wird an der Stelle x}
{ausgewertet. Formel (3.54), Buch S. 56.}
var deltafj,deltatj:real;
begin
  deltafj:=fr-fl;deltatj:=tr-tl;
  linearer_Spline:=fl+deltafj/deltatj*(x-tl);
end; {linearer_Spline}

function kubischer_Spline_lokal(x,tl,tr,fl,fr,sl,sr:real):real;
{Der kubische Spline auf [xl,xr] mit Werten fl, fr und Ableitungen sl,sr}
{wird an der Stelle x ausgewertet. Formel (3.67), Buch S. 60.}
var deltafj,deltatj,delta2tj,c0,c1,c2,c3:real;
begin
  deltafj:=fr-fl;deltatj:=tr-tl;delta2tj:=sqr(deltatj);
  c0:=fl;c1:=sl;c2:=(deltafj/deltatj-sl)/deltatj;
  c3:=(sr+sl-2*deltafj/deltatj)/delta2tj;
  kubischer_Spline_lokal:=c0+(x-tl)*(c1+(x-tl)*(c2+c3*(x-tr)));
end; {kubischer_Spline_lokal}

function fkt(Fall:integer;x:real):real;
begin
  case Fall of
```

```
  1:fkt:=x*sqr(x*x); {x^5}
  2..6:fkt:=exp(-sqr(x));
  end; {case of}
  {Die Flle 2 bis 6 entsprechen verschiedenen Intervallen}
end; {fkt}

function fkt_abl(Fall:integer;x:real):real;
{Es werden die Ableitg von fkt ausgerechnet}
begin
  case Fall of
  1:fkt_abl:=5*sqr(x*x); {5*x^4}
  2..6:fkt_abl:=-2*x*exp(-sqr(x));
  end; {case of}
  {Die Flle 2 bis 6 entsprechen verschiedenen Intervallen}
end; {fkt_abl}

function maximum(g:integer;v:vektor):real;
{Die grte Komponente von v[0],...,v[g] ist das gesuchte maximum}
var h1,h2:real;  j:integer;
begin
  h1:=-1.0e-15;
  for j:=0 to g do
  begin
    h2:=v[j];
    if h2>h1 then h1:=h2;
  end; {for j}
  maximum:=h1;
end; {maximum}

function absmaximum(g:integer;v:vektor):real;
{Die grte Komponente von |v[0]|,...,|v[g]| ist das gesuchte absmaximum}
var j:integer;
begin
  for j:=0 to g do v[j]:=abs(v[j]);
  absmaximum:=maximum(g,v);
end; {absmaximum}

procedure Fehlerordnung(Fehler:matrix;var alpha:matrix);
{Die erste Spalte von alpha ist irrelevant}
var j,k:integer;
begin
  for j:=1 to 6 do
  for k:=1 to 8 do
  alpha[j,k]:=ln(Fehler[j,k-1]/Fehler[j,k])/ln(2);
end; {Fehlerordnung}

procedure Fehler(var erg1,erg3:matrix);
{erg1[j,k] ist der Fehler eines linearen Splines im Fall j}
{bei Maschenweite h:=h(k):=(b-a)2^(-k).}
{Die Flle 1 bis 6 entsprechen den gegebenen Funktionen.}
{erg3 ist entsprechend der Fehler bei kubischer Splineinterpolation}

procedure intervall(Fall:integer;var a,b:real);
begin
  a:=0;
  case Fall of
  1..2:b:=1;
  3:b:=1.25;
  4:b:=1.5;
  5:b:=1.75;
  6:b:=2;
  end; {case of}
end; {intervall}

var g,j,k,l,wieoft:integer;a,b,h,H,xl,xr,fl,fr,sl,sr:real;
    v1,v1lokal,v3,v3lokal:vektor;
begin
  for j:=1 to 6 do {die 6 Flle}
  begin
    intervall(j,a,b);
    wieoft:=1;
    for k:=0 to 8 do {Die neun Maschenweiten h=(b-a)2^-k}
    begin
```

```
      h:=(b-a)/wieoft;
      for l:=0 to wieoft-1 do {durch die Maschen turnen}
      begin
        xl:=a+l*h; xr:=xl+h;fl:=fkt(j,xl);fr:=fkt(j,xr);
        sl:=fkt_abl(j,xl);sr:=fkt_abl(j,xr);
        for g:=0 to gross do {diskretisieren}
        begin H:=h/gross;
          v1lokal[g]:=fkt(j,xl+g*H)-linearer_Spline(xl+g*H,xl,xr,fl,fr);
          v3lokal[g]:=fkt(j,xl+g*H)-kubischer_Spline_lokal(xl+g*H,xl,xr,fl,fr,sl,sr);
        end; {for g}
        v1[l]:=absmaximum(gross,v1lokal);
        v3[l]:=absmaximum(gross,v3lokal);
      end; {for l}
      erg1[j,k]:=maximum(wieoft-1,v1);
      erg3[j,k]:=maximum(wieoft-1,v3);
      wieoft:=2*wieoft;
    end; {for k}
  end; {for j}
end; {Fehler}

procedure matrix_drucken(i:integer;erg:matrix);
{Es wird die Matrix erg zeilenweise, von Spalte i an ausgegeben}
  var j,k:integer;
begin
  for j:=1 to 6 do
  begin
    write(j:1,':');
    for k:=i to 8 do
      write(erg[j,k]:10:6);
    writeln;
  end; {for j}
  writeln;
end; {matrix_drucken}

begin {Hauptprogramm [dauert bei langsamem Rechner, also Geduld]}
  writeln('dauert bei langsamem Rechner, also etwas Geduld');
  Fehler(Erg1,Erg3);
  Fehlerordnung(Erg1,alpha1);
  Fehlerordnung(Erg3,alpha3);
  writeln('Fehler bei linearer Interpolation');
  matrix_drucken(0,Erg1);
  writeln('Fehlerordnung bei linearer Interpolation');
  matrix_drucken(1,alpha1);
  writeln('Fehler bei kubischer, lokaler Interpolation');
  matrix_drucken(0,Erg3);
  writeln('Fehlerordnung bei kubischer, lokaler Interpolation');
  matrix_drucken(1,alpha3);
end. {Hauptprogramm}
{Ergebnisse:

Fehler bei linearer Interpolation
1:  0.534992  0.285424  0.106274  0.032294  0.008885  0.002329  0.000596  0.000151  0.000038
2:  0.106010  0.050237  0.014794  0.003853  0.000973  0.000244  0.000061  0.000015  0.000004
3:  0.106079  0.069463  0.022415  0.005975  0.001518  0.000381  0.000095  0.000024  0.000006
4:  0.093639  0.086234  0.031088  0.008523  0.002180  0.000548  0.000137  0.000034  0.000009
5:  0.109750  0.098712  0.040480  0.011472  0.002960  0.000746  0.000187  0.000047  0.000012
6:  0.177455  0.106010  0.050237  0.014794  0.003853  0.000973  0.000244  0.000061  0.000015

Fehlerordnung bei linearer Interpolation
1:  0.906409  1.425317  1.718466  1.861756  1.931602  1.965991  1.983044  1.991536
2:  1.077366  1.763787  1.940844  1.985209  1.996300  1.999075  1.999769  1.999942
3:  0.610827  1.631751  1.907589  1.976888  1.994221  1.998555  1.999639  1.999910
4:  0.118868  1.471868  1.866975  1.966719  1.991679  1.997919  1.999480  1.999870
5:  0.152927  1.286029  1.819025  1.954707  1.988674  1.997168  1.999292  1.999823
6:  0.743254  1.077366  1.763787  1.940844  1.985209  1.996300  1.999075  1.999769

Fehler bei kubischer, lokaler Interpolation
1:  0.157024  0.014657  0.001068  0.000072  0.000005  0.000000  0.000000  0.000000  0.000000
2:  0.004226  0.001341  0.000112  0.000007  0.000000  0.000000  0.000000  0.000000  0.000000
3:  0.012488  0.002748  0.000259  0.000018  0.000001  0.000000  0.000000  0.000000  0.000000
4:  0.044230  0.005053  0.000503  0.000037  0.000002  0.000000  0.000000  0.000000  0.000000
5:  0.095340  0.006264  0.000862  0.000067  0.000004  0.000000  0.000000  0.000000  0.000000
6:  0.159799  0.005100  0.001341  0.000112  0.000007  0.000000  0.000000  0.000000  0.000000
```

```
Fehlerordnung bei kubischer, lokaler Interpolation
1:  3.421371  3.778262  3.900579  3.952718  3.976923  3.988595  3.994331  3.997174
2:  1.655496  3.587434  3.902553  3.975950  3.994005  3.998503  3.999625  3.999898
3:  2.184066  3.408891  3.846215  3.962331  3.990627  3.997660  3.999415  3.999850
4:  3.129922  3.328067  3.775744  3.945600  3.986494  3.996630  3.999158  3.999789
5:  3.927911  2.860719  3.689997  3.925696  3.981604  3.995412  3.998854  3.999712
6:  4.969757  1.926627  3.587434  3.902553  3.975950  3.994005  3.998503  3.999625}
```

Aufgabe 20. (a) Für $f(x) = 1 - \operatorname{sign}(x)x^5$ ist $|f''(x)| = 20x^3$, $|f^{(iv)}(x)| = 120x$. Also ist $\max_{x\in[0,1]} |f''(x)| = 20$, $\max_{x\in[0,1]} |f^{(iv)}(x)| = 120$. Der Fehler ε der linearen Spline-Interpolation (nach Formel (3.59), S. 58) ist also $\varepsilon \le 20h/8^2 = (5/2)h^2$, der Fehler ε der kubischen Spline-Interpolation (Formel (3.42), S. 53 mit $k = 1$) ist also $\varepsilon \le 120h^4/384 = (5/16)h^4$. (b) Die benötigten Hermitepolynome sind $P_2(x) = 4x^2 - 2$, $P_4(x) = 16x^4 - 48x^2 + 12$. Da $\exp(-x^2)$ und auch $P_n(x)\exp(-x^2)$ für wachsende x sehr schnell abfallen, ist $\max_{x\ge 0} |P_n(x)\exp(-x^2)| = |P_n(0)|$. Damit ist $\max |f''(x)| = 2$, $\max |f^{(iv)}(x)| = 12$. Die zu erwartenden Fehler sind damit höchstens $h^2/4$ (linearer Spline), bzw $h^4/32$ (lokaler, kubischer Spline), bei $h = 0.2$ also höchstens $5\cdot 10^{-5}$. Man vgl. dazu die Figur L10.2, in der $\exp(-x^2)$ in $[-2, 2]$ mit 20 Teilintervallen durch einen lokalen, kubischen (C^1-) Spline ersetzt wurde.



Figur L10.2. Fehler eines lokalen, kubischen Splines