Numerik für CE, Ing. und Phys., Übung 2, Lösungsvorschlag

Gruppenübung

G 4 (*Splines*)

Kreuzen Sie die richtigen Feststellungen an und begründen Sie ihre Antworten.

a) ☐ Eine kubische Splinefunktion ist das Polynom dritten Grades, das durch die Daten $(x_i, y_i) \quad (i = 0, \ldots, n)$ eindeutig bestimmt ist.

☐ Eine kubische Splinefunktion ist zweimal stetig differenzierbar und stimmt auf jedem Teilintervall $[x_i, x_{i+1}] \quad (i = 0, \ldots, n-1)$ mit einem Polynom dritten Grades überein.

b) ☐ Für die Eindeutigkeit eines interpolierenden kubischen Splines ist die ausschließliche Kenntnis aller Datensätze $(x_0, y_0), \ldots, (x_n, y_n)$ nicht ausreichend.

☐ Mit vier Daten $(x_0, y_0), \ldots, (x_3, y_3)$ ist ein interpolierender kubischer Spline bereits eindeutig bestimmt.

☐ Ein interpolierender kubischer Spline auf $[x_0, x_n]$ mit den Nebenbedingungen $s'(x_0) = f'(x_0)$ und $s'(x_n) = f'(x_n)$ kann bereits eindeutig berechnet werden.

a) ☐ *Eine kubische Splinefunktion ist das Polynom dritten Grades, das durch die Daten* $(x_i, y_i) \quad (i = 0, \ldots, n)$ *eindeutig bestimmt ist.*

☒ *Eine kubische Splinefunktion ist zweimal stetig differenzierbar und stimmt auf jedem Teilintervall* $[x_i, x_{i+1}] \quad (i = 0, \ldots, n-1)$ *mit einem Polynom dritten Grades überein.*

b) ☒ *Für die Eindeutigkeit eines interpolierenden kubischen Splines ist die ausschließliche Kenntnis aller Datensätze* $(x_0, y_0), \ldots, (x_n, y_n)$ *nicht ausreichend.*

☐ *Mit vier Daten* $(x_0, y_0), \ldots, (x_3, y_3)$ *ist ein interpolierender kubischer Spline bereits eindeutig bestimmt.*

☒ *Ein interpolierender kubischer Spline auf* $[x_0, x_n]$ *mit den Nebenbedingungen* $s'(x_0) = f'(x_0)$ *und* $s'(x_n) = f'(x_n)$ *kann bereits eindeutig berechnet werden.*

Zu a):

Eine kubische Splinefunktion ist auf dem Intervall $[x_0, x_n]$ *i.allg.* kein *Polynom dritten Grades. Sie setzt sich vielmehr aus Polynomen dritten Grades auf* $[x_i, x_{i+1}]$ *zusammen.*

Wesentlich ist weiterhin, daß die Teilpolynome so zusammengesetzt werden, daß s *auf ganz* $[x_0, x_n]$ *zweimal stetig differenzierbar ist.*

Zu b):

Um einen interpolierenden kubischen Spline eindeutig zu bestimmen, benötigt man neben den gegebenen Daten $(x_i, y_i), \; i = 0, \ldots, n$ *noch zwei lineare Zusatzbedingungen. In der Vorlesung sind hierfür drei Möglichkeiten angegeben worden: natürliche, hermitesche bzw. periodische Randbedingungen. Es sind aber selbstverständlich auch andere Bedingungen denkbar.*

G 5 (*Kubischer Spline?*)

Gegeben sei die Funktion

$$\Phi(x) = \frac{1}{6}\begin{cases} x^3+6x^2+12x+8 & \text{für} \quad -2 \le x < -1 \\ -3x^3-6x^2+4 & \text{für} \quad -1 \le x < 0 \\ 3x^3-6x^2+4 & \text{für} \quad 0 \le x < 1 \\ -x^3+6x^2-12x+8 & \text{für} \quad 1 \le x < 2 \\ 0 & \text{sonst.} \end{cases}$$

Man prüfe, ob Φ ein kubischer Spline ist.

Ein kubischer Spline ist definiert durch:

1. Φ *ist zweimal stetig differenzierbar, also* $\Phi \in C^2(\mathbb{R})$.
2. Φ *ist stückweise kubisch, genauer* $\Phi\big|_{[i,i+1]} \in \Pi_3$ *für* $i \in \mathbb{Z}$.

1. *Die Funktion* Φ *ist stetig, da gilt:*

$$\begin{array}{rcccl} & & 0 & = & (x^3+6x^2+12x+8)\big|_{x=-2} \\ \lim_{x\nearrow -1} x^3+6x^2+12x+8 & = & 1 & = & (-3x^3-6x^2+4)\big|_{x=-1} \\ \lim_{x\nearrow 0} -3x^3-6x^2+4 & = & 4 & = & (3x^3-6x^2+4)\big|_{x=0} \\ \lim_{x\nearrow 1} 3x^3-6x^2+4 & = & 1 & = & (-x^3+6x^2-12x+8)\big|_{x=1} \\ \lim_{x\nearrow 2} -x^3+6x^2-12x+8 & = & 0 & & \end{array}$$

Auch die erste Ableitung

$$\Phi'(x) = \frac{1}{6}\begin{cases} 3x^2+12x+12 & \textit{für} \quad -2 \le x < -1 \\ -9x^2-12x & \textit{für} \quad -1 \le x < 0 \\ 9x^2-12x & \textit{für} \quad 0 \le x < 1 \\ -3x^2+12x-12 & \textit{für} \quad 1 \le x < 2 \\ 0 & \textit{sonst.} \end{cases}$$

ist stetig:

$$\begin{array}{rcccl} & & 0 & = & (3x^2+12x+12)\big|_{x=-2} \\ \lim_{x\nearrow -1} 3x^2+12x+12 & = & 3 & = & (-9x^2-12x)\big|_{x=-1} \\ \lim_{x\nearrow 0} -9x^2-12x & = & 0 & = & (9x^2-12x)\big|_{x=0} \\ \lim_{x\nearrow 1} 9x^2-12x & = & -3 & = & (-3x^2+12x-12)\big|_{x=1} \\ \lim_{x\nearrow 2} -3x^2+12x-12 & = & 0 & & \end{array}$$

Schließlich ist auch die zweite Ableitung

$$\Phi''(x) = \frac{1}{6}\begin{cases} 6x+12 & \textit{für} \quad -2 \le x < -1 \\ -18x-12 & \textit{für} \quad -1 \le x < 0 \\ 18x-12 & \textit{für} \quad 0 \le x < 1 \\ -6x+12 & \textit{für} \quad 1 \le x < 2 \\ 0 & \textit{sonst.} \end{cases}$$

stetig:

$$\begin{array}{lcll} & 0 & = & (6x+12)\big|_{x=-2} \\ \lim_{x\nearrow -1} 6x+12 & 6 & = & (-18x-12)\big|_{x=-1} \\ \lim_{x\nearrow 0} -18x-12 & -12 & = & (18x-12)\big|_{x=0} \\ \lim_{x\nearrow 1} 18x-12 & 6 & = & (-6x+12)\big|_{x=1} \\ \lim_{x\nearrow 2} -6x+12 & 0 & & \end{array}$$

Damit ist $\Phi \in C^2(\mathbb{R})$.

2. *Dies sieht man unmittelbar an der Definition von* Φ.

G 6 (*Natürlicher Spline*)

Stellen Sie das Gleichungssystem zur Bestimmung des natürlichen interpolierenden kubischen Spline zu den gegebenen Daten auf.

x_i	-3	-2	0	1	2
y_i	0	2	2	11	18

Die Werte für einen natürlichen Spline:

i	0	1	2	3	4
x_i	-3	-2	0	1	2
$h_i = x_{i+1} - x_i$		1	2	1	1
y_i	0	2	2	11	18
$z_i = y_{i+1} - y_i$		2	0	9	7

Das ergibt das lineare Gleichungssystem:

$$\begin{pmatrix} 2\cdot(2+1) & 2 & 0 \\ 2 & 2\cdot(2+1) & 1 \\ 0 & 1 & 2\cdot(1+1) \end{pmatrix} \begin{pmatrix} M_1^* \\ M_2^* \\ M_3^* \end{pmatrix} = \begin{pmatrix} \frac{0}{2} - \frac{2}{1} \\ \frac{9}{1} - \frac{0}{2} \\ \frac{7}{1} - \frac{9}{1} \end{pmatrix}$$

oder

$$\begin{pmatrix} 6 & 2 & 0 \\ 2 & 6 & 1 \\ 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} M_1^* \\ M_2^* \\ M_3^* \end{pmatrix} = \begin{pmatrix} -2 \\ 9 \\ -2 \end{pmatrix}.$$

Hausübung

H 4 (*Denkaufgabe*)

Sei φ_f der natürliche interpolierende kubische Spline zur Funktion f. Wie lautet der natürliche interpolierende kubische Spline φ_g zur Funktion g mit $g(x) = f(x) + ax + b$?

Ansatz: $\varphi_g = \varphi_f + ax + b$
Damit gilt:

- φ_g *ist stückweise kubisch;*
- $\varphi_g(x_i) = \varphi_f(x_i) + ax_i + b = f(x_i) + ax_i + b$, *interpoliert also und ist stetig;*
- φ_g *ist die Summe von zweimal stetig differenzierbaren Funktionen also auch zweimal stetig differenzierbar;*
- $\varphi_g''(x_0) = \varphi_f''(x_0) = 0$ *und* $\varphi_g''(x_n) = \varphi_f''(x_n) = 0$, *damit sind die Randbedingungen erfüllt.*

Insgesamt ist also φ_g *der kubische natürliche interpolierende Spline zu* $f(x) + ax + b$.

H 5 ($C^1(\mathbb{R})$-*Eigenschaft*)

Kann man $p_1, p_2 \in \Pi_2$ so wählen, daß die Funktion

$$s(x) = \begin{cases} -1 & \text{für} \quad -\infty \le x < -1 \\ p_1(x) & \text{für} \quad -1 \le x < 0 \\ p_2(x) & \text{für} \quad 0 \le x < 1 \\ 1 & \text{für} \quad 1 \le x < \infty. \end{cases}$$

aus $C^1(\mathbb{R})$ ist?

Nach Voraussetzung gelten

$$p_1(x) = ax^2 + bx + c \text{ bzw } p_2(x) = dx^2 + ex + f.$$

und damit

$$p_1'(x) = 2ax + b \text{ bzw } p_2'(x) = 2dx + e.$$

Damit s *stetig ist, müssen gelten:*

$$p_1(-1) = -1,\ p_2(1) = 1 \text{ und } p_1(0) = p_2(0).$$

Also erhalten wir

$$\begin{aligned} a - b + c &= -1 \\ d + e + f &= 1 \\ c - f &= 0. \end{aligned}$$

Der Differenzierbarkeit wegen fordern wir:

$$p_1'(-1) = 0,\ p_2'(1) = 0 \text{ und } p_1'(0) = p_2'(0).$$

Das liefert folgende Gleichungen:

$$\begin{aligned} -2a + b &= 0 \\ 2d + e &= 0 \\ b - e &= 0. \end{aligned}$$

Insgesamt müssen wir also ein lineares Gleichungssystem lösen:

$$\begin{pmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Nach einiger Rechnung stellt man fest, daß die Systemmatrix den Rang sechs hat. Also ist das LGS eindeutig lösbar. Somit kann $s \in \mathbb{C}^1(\mathbb{R})$ durchaus gelten. Die Lösung des LGS ist $(1, 2, 0, -1, 2, 0)^T$. Es sind also $p_1(x) = x^2 + 2x$ bzw. $p_2(x) = -x^2 + 2x$.

H 6 (*MATLAB/OCTAVE: Periodische Splinefunktion*)

Schreiben Sie eine `MATLAB`-Funktion

```
function [m,c,d] = periodicspline(x,y)
% computes the momentum representation of the periodic cubic spline
% through the data points (x(i),y(i)) i=1,..,n=length(x)
% m stores the moments and c and d the coefficients of the linear part
```

die die Daten $(x(i), y(i)),\ i = 1, .., n$ mit $y(1) = y(n)$ durch eine periodische kubische Splinefunktion interpoliert in der Darstellung mit den Momenten $M_i^* = M_i/6$ wie im Skript beschrieben. Zur Lösung des linearen Gleichungssystems genügt es, die Matrix als vollbesetzte Matrix aufzubauen und den backslash-Operator zu benutzen. (Sie können natürlich auch den effektiveren Weg einer speziellen Eliminationsroutine für diese quasi-tridiagonale Matrix benutzen, der mit 3 Vektoren auskommt). Erstellen Sie ferner eine Funktion

```
function s=splineeval(xx,x,m,c,d)
%computes the cubic spline in its moment form at xx
```

zur Berechnung des Wertes der Splinefunktion an der Stelle xx. Berechnen Sie dabei das für xx relevante Intervall durch logarithmische Suche im Intervall $[1, n]$.

Berechnen Sie mit Hilfe dieser Funktionen periodische Splineapproximationen für $f(x) =$

$\sin(x)$ auf $[0, 2\pi]$ mit 11, 21, 41 und 81 äquidistanten Gitterpunkten und plotten Sie die Fehlerkurve, wobei eine Abtastgenauigkeit mit 401 Punkten genügt. Berechnen Sie den von n abhängigen maximalen Fehler auf diesem Gitter und die Quotienten aufeinanderfolgender maximaler Fehler. Welche $h-$Ordung der Konvergenz kann man aus den Resultaten ablesen?

Hinweis: Im Skript ist das Gitter mit $0, \ldots, n+1$ numeriert. Dies müssen Sie hier in $1, \ldots, n+2$ umsetzen, da MATLAB den Index 0 nicht kennt.

periodicspline(x,y)

```
function [m,c,d]=periodicspline(x,y)
% computes the momentum representation of the
% periodic cubic spline through the data
%points (x(i),y(i)) i=1,..,n=length(x)
% m stores the moments and c and d the coefficients of the linear part
n=length (x);
if length(y) ~= n
    error('periodic spline: x and y don''t fit');
end
if y(1) ~= y(n)
    error('periodic spline without periodic y');
end
m=zeros(n,1);
c=zeros(n-1,1);
d=zeros(n-1,1);
diffx=zeros(n-1,1);
diffdiffy=zeros(n-1,1);
diffy=zeros(n-1,1);
a=zeros(n-1,1);
b=zeros(n-1,1);
c=zeros(n-1,1);
diffx=x(2:n)-x(1:n-1);
diffy=(y(2:n)-y(1:n-1))./diffx;
diffdiffy(1:n-2)=diffy(2:n-1)-diffy(1:n-2);
diffdiffy(n-1)=diffy(1)-diffy(n-1);
rhs=diffdiffy;
% the linear system
a(1:n-2)=2*(diffx(2:n-1)+diffx(1:n-2));
a(n-1)=2*(diffx(1)+diffx(n-1));
b(1:n-2)=diffx(2:n-1);
c(1)=diffx(1);
%elimination
%system has dimension n-1
```

```
for i=1:n-2
    %there are two multipliers only: subdiagonal and last row
    %a stores the pivots
    %b stores the superdiagonal, there is one modified element
    %c stores the last column up to the superdiagonal
    mult=b(i)/a(i);  %subdiagonal
    mult1=c(i)/a(i); %last row one element
    if i <= n-4
        c(i+1)=-c(i)*mult;
        a(n-1)=a(n-1)-c(i)*mult1;
        rhs(n-1)=rhs(n-1)-rhs(i)*mult1;
    end
    if i == n-3
        b(i+1)=b(i+1)-c(i)*mult;
        a(n-1)=a(n-1)-c(i)*mult1;
        rhs(n-1)=rhs(n-1)-rhs(i)*mult1;
    end
    a(i+1)=a(i+1)-b(i)*mult;
    rhs(i+1)=rhs(i+1)-rhs(i)*mult;
end
for i=n-1:-1:1
    s=rhs(i);
    if i<=n-2
        s=s-b(i)*m(i+2);
    end
    if i<n-2
        s=s-c(i)*m(n);
    end
    m(i+1)=s/a(i);
end
m(1)=m(n);
%periodicity
% the remaining coefficients: linear part
c=diffy-diffx.*(m(2:n)-m(1:n-1));
d=y(1:n-1)-(diffx.^2).*m(1:n-1);
```

splineeval(xx,x,m,c,d)

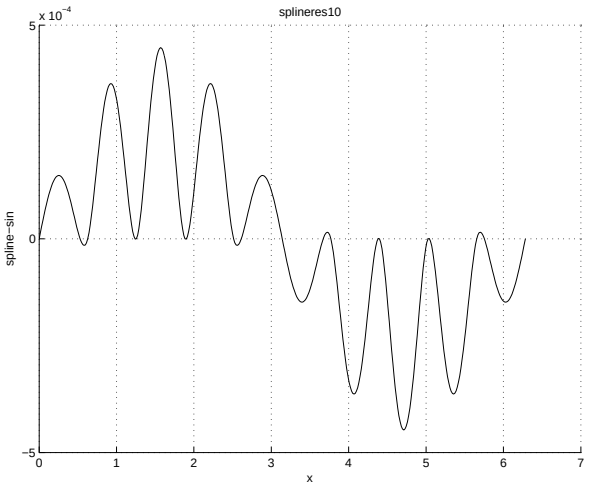
```
function s=splineeval(xx,x,m,c,d)
%computes the cubic spline in its moment form at xx
n=length(x);
%search for the relevant interval
low=1;
up=n;
```

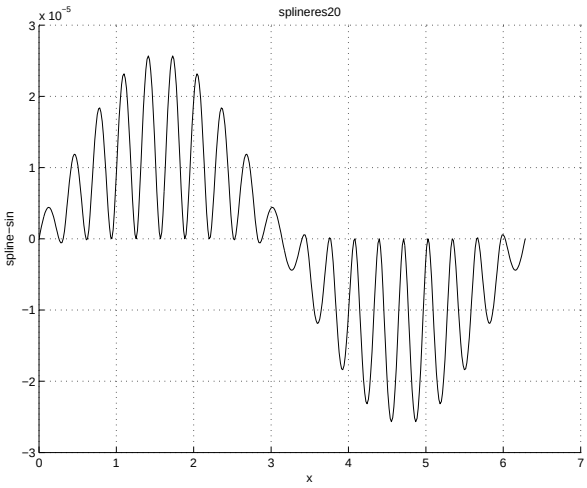
```
while up-low>1
    mid=floor((up+low)/2);
    if xx<x(mid)
        up=mid;
    else
        low=mid;
    end
end
i=low;
s=d(i)+c(i)*(xx-x(i))+( m(i)* (x(i+1)-xx)^3 + m(i+1) * ...
            (xx-x(i))^3)/(x(i+1)-x(i));
```

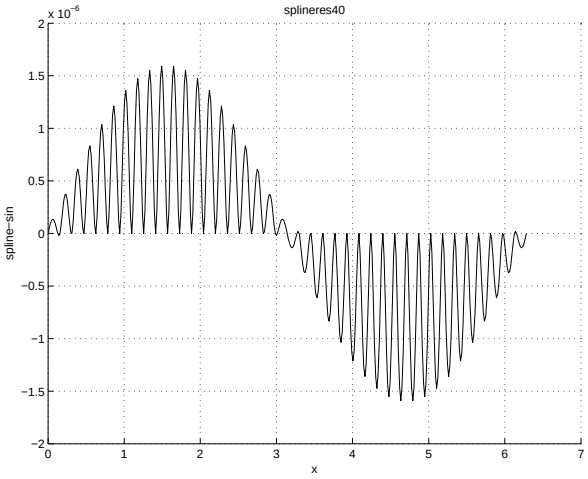
3. Programmieraufgabe

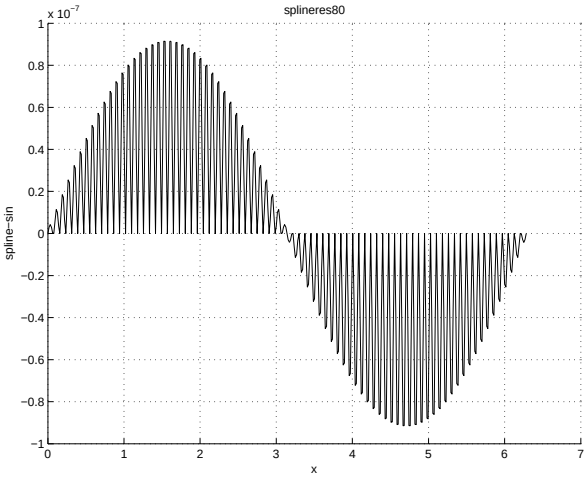
```
low=0;
up=2*pi;
maxerr=zeros(4,1);
kmax=400; % for plotting
xx=zeros(kmax+1,1);
yy=zeros(kmax+1,1);
for i=1:4
    n=2^(i-1)*10;
    x=zeros(n+1,1);
    y=zeros(n+1,1);
    h=2*pi/n;
    for j=0:n
        x(j+1)=low+j*h;
        y(j+1)=sin(x(j+1));
    end
    y(n+1)=y(1); %in order to avoid roundoff trouble
    filnam=['splineres',int2str(n)];
    for k=1:length(filnam)
        if strcmp(filnam(k),' ')
            filnam(k)='_';
        end
    end
    [m,c,d]=periodicspline(x,y);
    for k=0:kmax
        xx(k+1)=low+(up-low)/kmax*k;
        yy(k+1)=splineeval(xx(k+1),x,m,c,d);
        yy(k+1)=sin(xx(k+1))-yy(k+1);
        maxerr(i)=max(maxerr(i),abs(yy(k+1)));
    end
    hold on
```

```
    title(filnam);
    grid on;
    xlabel('x');
    ylabel('spline-sin');
    plot(xx,yy);
    pause;
    print('-deps','-r600',filnam);
    clf;
end
disp('periodic spline for sin');
disp('development of error');
disp(['max error n=10 :',num2str(maxerr(1))]);
disp('quotients of errors , doubling n');
for i=2:4
    disp(num2str(maxerr(i-1)/maxerr(i)));
end
```









Maximaler Fehler mit $n = 10$*: 0.00044726,*
Fehlerquotiente beim Verdoppeln der Stützstellen:
$n = 20$*: 17.418*
$n = 40$*: 16.1464*
$n = 80$*: 17.3951*

Theoretisch müsste der Fehlerquotient 16 betragen, da aber die maximalen Abweichungen an verschiedenen Stellen auftreten, wird es nur näherungsweise erhalten.