Aufgabe 25.9

Lösen Sie die folgenden Aufgaben mit `MATLAB`. Protokollieren Sie Ihr Vorgehen in einer `diary`-Datei und speichern Sie erstellte Plots ab.

1. Zeichnen Sie das Richtungsfeld der Differenzialgleichung aus Aufgabe 21.3 zusammen mit einigen ausgewählten Lösungen der Gleichungen in einen gemeinsamen Plot und beschriften Sie die Achsen. Beachten Sie, dass sich die Pfeile nur im Anstieg, nicht aber in ihrer Länge unterscheiden sollen.
2. Berechnen Sie die Eigenwerte und Eigenvektoren der in den Aufgaben 22.8 und 22.20 auftretenden Systemmatrizen und vergleichen Sie diese mit Ihren Ergebnissen.
3. Zeichnen Sie die Punkte (x, y), welche die Gleichung in Aufgabe 17.36 erfüllen, unter Verwendung des `contour`-Befehls und beschriften Sie die Achsen.

Öffnen Sie die erstellte `diary`-Datei (vorher mit $\gg$ `diary off` die Protokollierung abschließen) und entfernen Sie ggf. überflüssige Zeilen (z.B. Fehleingaben). Drucken Sie anschließend die bearbeitete `diary`-Datei und eventuell angefertigte Plots und `m-Files` möglichst sparsam (d.h. nach Möglichkeit duplex, mehrere Seiten pro Blatt, kleine Schriftgröße) aus.

Hinweise zur MATLABaufgabe

eig

Die Funktion `eig` dient der Bestimmung von Eigenwerten und der dazugehörigen Eigenvektoren. Zum Beispiel gibt der Befehl

```
≫ eig( [1 2 ; 2 1] )
```

die Eigenwerte der Matrix

$$\begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$$

zurück. Ist man zusätzlich an den Eigenvektoren interessiert, so werden zwei Rückgabeargumente benötigt

```
≫ [V,D] = eig( [1 2 ; 2 1] )
```

contour

Mit `contour` können die Niveaumengen (Punkte mit gleichem Funktionswert, z.B. Höhenlinien auf einer Landkarte) einer Funktion dargestellt werden. Mit den Befehlen

```
≫ x = linspace(-2, 2, 100);
≫ y = linspace(-2, 2, 100);
≫ [X,Y] = meshgrid(x,y);
≫ Z = X.^2 + Y.^2;
≫ contour(X,Y,Z)
```

werden einige Niveaumengen, also $\{(x,y) : f(x,y) = c\}$, der Funktion $f(x,y) = x^2 + y^2$ gezeichnet. Es ist möglich, die gewünschten Niveaus c selbst anzugeben, z.B. werden mit

```
≫ contour(X,Y,Z,[1 2])
```

nur die beiden Kreise zum Niveau $c = 1$ und $c = 2$ gezeichnet. Will man nur ein Niveau zeichnen, muss dieses doppelt angegeben werden, z.B.

```
≫ contour(X,Y,Z,[1 1])
```

quiver

`quiver` dient dem Zeichnen von Pfeilen. Zum Beispiel zeichnet

```
≫ quiver( [0 0 0], [-1 0 1], [2 -1 3] , [0 0 -2] )
```

die drei Vektoren

$$\vec{v}_1 = \begin{pmatrix} 2 \\ 0 \end{pmatrix}, \qquad \vec{v}_2 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, \qquad \vec{v}_3 = \begin{pmatrix} 3 \\ -2 \end{pmatrix}$$

an die zugehörigen Stellen

$$\vec{x}_1 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \qquad \vec{x}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \vec{x}_3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

siehe auch

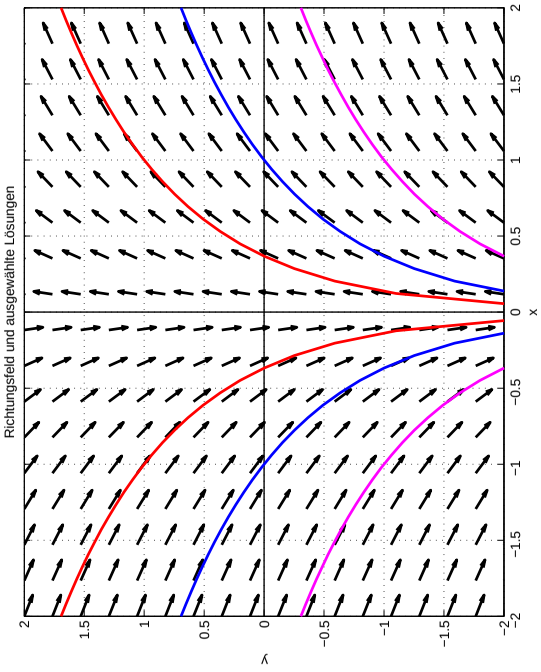
```
≫ help quiver
```

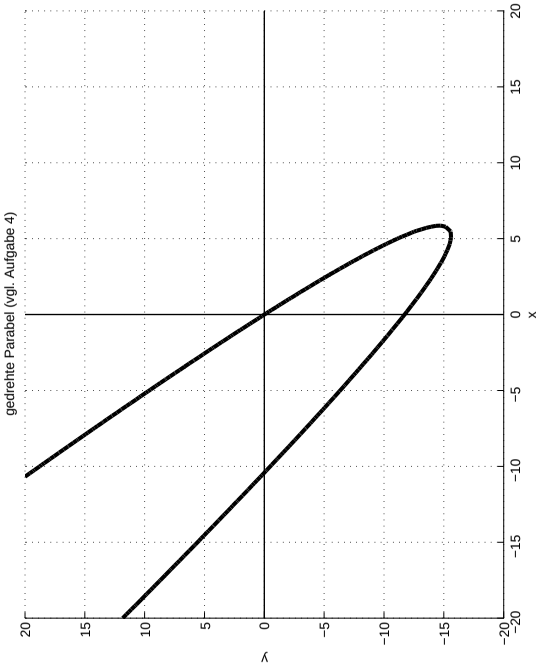
Lösung:

nachbereitete `diary`-Datei (Kommentare durch % gekennzeichnet) und Plots auf dieser und der nächsten Seite

```
% ------------------------------------------------------------------------
% Aufgabe 1
% ------------------------------------------------------------------------

% Anzahl der Pfeile pro Koordinatenrichtung
n = 18;

x = linspace(-2 , 2 , n);
y = linspace(-2 , 2 , n);
[X,Y] = meshgrid(x , y);

% Verschiebe X um eps, damit X nie exakt 0 ist
X = X + eps;

% Berechnung der Norm der Pfeile, um diese später zu Normieren
Norm = sqrt(1 + 1./X.^2);

% x Koordinate der Pfeile setzen
U = ones(n,n) ./ Norm;
% y Koordinate der Pfeile setzen
V = (1./X) ./ Norm;

% Pfeile zeichnen
figure(1); clf;
quiver(X , Y , U , V , 0.5 , 'LineWidth' , 2 , 'Color' , 'black');

%
% für Octave ('LineWidth',2,'Color','black' führen bei Octave zu Fehler):
% quiver(X , Y , U , V , 0.5)
%

% axis fest einstellen
axis([-2 2 -2 2]);
grid on;
hold on;
% Koordinatenachsen hervorheben
plot([-2,2],[0,0],'-k','LineWidth',1)
plot([0,0],[-2,2],'-k','LineWidth',1)

% Jetzt drei Lösungen einzeichnen
n = 50;
x = linspace(-2, 2, n);

f1 = log(abs(x));
f2 = log(abs(x)) + 1;
f3 = log(abs(x)) - 1;

plot(x , f1 , 'Color' , 'blue' , 'LineWidth' , 2);
plot(x , f2 , 'Color' , 'red' , 'LineWidth' , 2);
plot(x , f3 , 'Color' , 'magenta' , 'LineWidth' , 2);

% Labels, Title
xlabel('x'); ylabel('y');
title('Richtungsfeld und ausgewählte Lösungen');
print -depsc HA04_matlab_plot_1.eps
```

```
% ------------------------------------------------------------------------
% Aufgabe 2
% ------------------------------------------------------------------------

A = [ -1 4 ; -2 3 ];
EW=eig(A)'
EW =
   1.0000 - 2.0000i   1.0000 + 2.0000i

[EV,D]=eig(A); EV
EV =
   0.8165             0.8165
   0.4082 + 0.4082i   0.4082 - 0.4082i

B = [0  2 0 ; 3 -5 0 ; 2 -4 1 ];
EW=eig(B)'
EW =
     1    -6     1

[EV,D]=eig(B); EV
EV =
         0    0.2673   -0.8016
         0   -0.8018   -0.4008
    1.0000   -0.5345   -0.4436

% Ermittelt wurden normierte Eigenvektoren. Der 1. und der 3. EV gehören
% zum EW 1. Der 3. EV ergibt sich aus der oben in der Musterlösung von
% Aufgabe 6 notierten Darstellung durch Wahl von C=-0.4008 und D=-0.4436.


% ------------------------------------------------------------------------
% Aufgabe 3
% ------------------------------------------------------------------------

n = 100;
x = linspace(-20, 20, n);
y = linspace(-20, 20, n);
[X,Y] = meshgrid(x,y);

Z = 9*X.^2 + 12*X.*Y + 4*Y.^2 + 26*sqrt(13)*X + 13*sqrt(13)*Y;

figure(2); clf; hold on;
contour(X,Y,Z,[0 0],'LineWidth',3,'Color','black');

%
% für Octave (kann mit contour nur geschlossene Konturen darstellen):
% Z(1,:)=0; Z(100,:)=0; Z(:,1)=0; Z(:,100)=0;
% Zusatzparameter werden bei Octave beim contour-Befehl nicht akzeptiert:
% contour(X,Y,Z,[0,0])
%

% Label, Title
xlabel('x'); ylabel('y'); grid on; hold on;
plot([-20,20],[0,0],'-k','LineWidth',1)
plot([0,0],[-20,20],'-k','LineWidth',1)
title('gedrehte Parabel (vgl. Aufgabe 4)');
print -depsc HA04_matlab_plot_3.eps
diary off
```


Richtungsfeld und ausgewählte Lösungen
x
y
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2


gedrehte Parabel (vgl. Aufgabe 4)
x
y
20
15
10
5
0
−5
−10
−15
−20