Aufgabe 25.7

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

1. Zeichnen Sie die Funktion $f(x) = \sin(x)$ und die Taylor-Polynome ungeraden Grades bis zur 11ten Ordnung mit der Entwicklungsstelle $x_0 = 0$ im Intervall $[-4\pi, 4\pi]$ in einen gemeinsamen Plot. Beschriften Sie die Achsen und erstellen Sie eine Legende. Verwenden Sie den Befehl `axis`, um einen angemessenen Bildausschnitt auszuwählen.

2. a) Implementieren Sie das Newton-Verfahren. Erstellen Sie dazu ein extra `m-File` und arbeiten Sie mit `function-handles`. Ihrer Funktion werden die folgenden Parameter übergeben

      - die Funktion F, auf die das Verfahren angewandt werden soll (in Form eines `function-handles`),
      - die Ableitung der Funktion F (in Form eines `function-handles`),
      - der Startwert x_0.

      Zurückgegeben werden soll ein Vektor $x = (x_0, x_1, \ldots, x_n)$, welcher die gesamte Iterationsfolge enthält. Verwenden Sie $\varepsilon = 10^{-8}$ als Parameter für das Abbruchkriterium.

   b) Wenden Sie Ihr `m-File` zum Newton-Verfahren auf die Funktion aus Aufgabe 2a) aus Aufgabe 25.6 ($f(x) = \ln(x+1)\sin(x^2)$) mit den Startwerten

      i. x_0=`1.50`
      ii. x_0=`1.43`
      iii. x_0=`1.40`

      an. Zeichnen Sie jeweils die Funktion f und den Verlauf der Iterierten.

   c) Setzen Sie nun das Newton-Verfahren ein, um x mit $f'(x) = 0$ zu finden. Verwenden Sie die Startwerte

      i. x_0=`1.70`
      ii. x_0=`3.17`
      iii. x_0=`0.70`

      und zeichnen Sie die Funktion f und f' sowie jeweils den Verlauf der Iterierten. Bestätigen Sie anhand der zweiten Ableitung, dass es sich bei der jeweils letzten Iterierten um ein lokales Maximum, ein lokales Minimum bzw. einen Sattelpunkt handelt.

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

Schleifen

MATLAB unterstützt die Verwendung von `while`-Schleifen und `for`-Schleifen, siehe:

```
≫ help while
≫ help for
```

Als Beispiel dient das folgende kleine Programm. Es tabelliert die Sinus-Funktion im Intervall $[0, 2\pi]$ im Abstand von $\frac{\pi}{4}$.

```
≫ m=1;
≫ fprintf('\n\nn    n*pi/4          sin(n*pi/4)\n')
≫ while m<10,
≫     x(m)=(m-1)*pi/4;
≫     fprintf('%1.0f%16.10f%16.10f\n',m-1,x(m),sin(x(m)))
≫     m=m+1;
≫ end
```

function-handles

`Function-handles` ermöglichen es, Funktionen an Funktionen zu übergeben. Als Beispiel wollen wir hier in einer Funktion die Werte $f(0) + f(1) + \ldots f(5)$ berechnen, wobei die Funktion f durch ein `function-handle` übergeben wird. Dazu erstellen wir das `m-File` `Summe0bis5.m` mit dem Inhalt:

```
function Summe = Summe0bis5(f)
    Summe = f(0) + f(1) + f(2) + f(3) + f(4) + f(5);
```

Wir nutzen als Beispiel für die Funktion f zunächst die `sin`-Funktion,

```
≫ Summe0bis5(@sin)
```

Ein `function-handle` für die `sin`-Funktion wird also durch das voranstellen des `@` erreicht. Anstatt für jede erdenkliche Funktion f eine eigene Funktion zur Summation zu verwenden, genügt die Funktion `Summe0bis5`. Sie können auch Ihre eigenen Funktionen übergeben, sofern diese in `m-Files` ausgelagert sind:

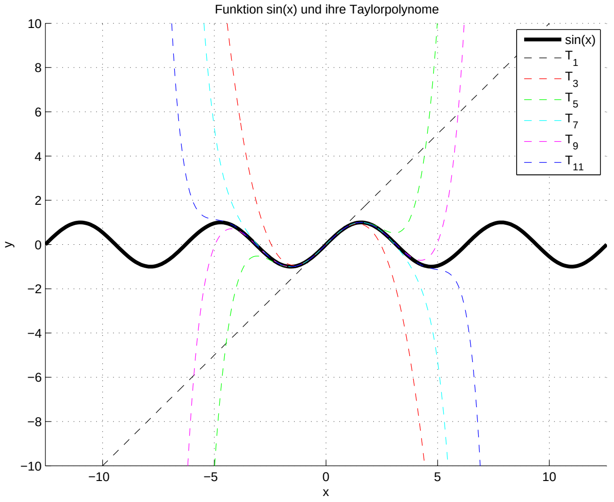
```
≫ Summe0bis5(@Name_des_m-Files)
```

Weitere Hinweise zu `function-handles` finden Sie in der MATLAB-Hilfe,

```
≫ help function_handle
```

Lösung:
nachbereitete `diary`-Datei (Kommentare durch % gekennzeichnet) und Plots auf dieser und den nächsten Seiten

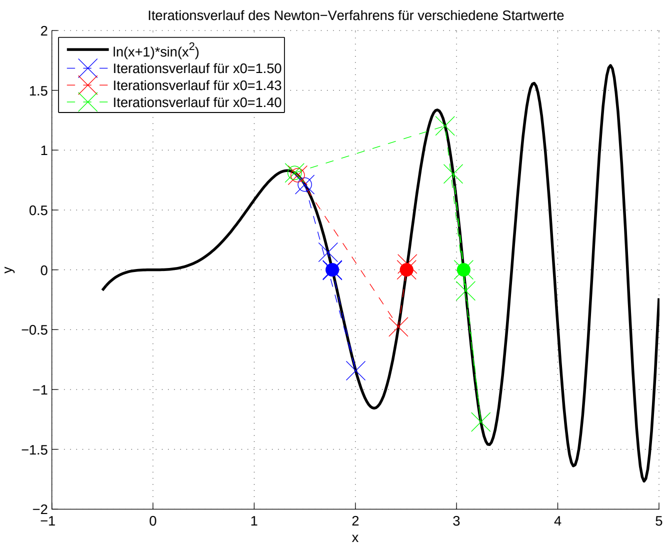
```
% -----------------------------------------------------------------------
% Aufgabe 1
% -----------------------------------------------------------------------

x=linspace(-4*pi,4*pi,300);
% Funktion sin auswerten
y=sin(x);

% Taylorpolynome auswerten
T1=x;
T3= T1 - 1/6*x.^3;
T5= T3 + 1/120*x.^5;
T7= T5 - 1/5040*x.^7;
T9= T7 + 1/362880*x.^9;
T11= T9 - 1/39916800*x.^11;

% Funktionen zeichnen
figure(1); clf; hold on;
plot(x,y,'k-','LineWidth',3);
plot(x,T1,'k--');
plot(x,T3,'r--');
plot(x,T5,'g--');
plot(x,T7,'c--');
plot(x,T9,'m--');
plot(x,T11,'b--');
% Dargestellten Bereich anpassen
axis([-4*pi,4*pi, -10,10])

% Label, Title, Legend
xlabel('x'); ylabel('y'); grid on;
title('Funktion sin(x) und ihre Taylorpolynome');
legend('sin(x)', 'T_1', 'T_3', 'T_5', 'T_7', 'T_9', 'T_{11}', 'Location',
       'NorthEast');
print -depsc HA02_matlab_plot_1.eps
```




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


% a) --------------------------------------------------------------------

% Listing des m-Files HA02_matlab_solution_Newton_Verfahren.m
type HA02_matlab_solution_Newton_Verfahren.m


function x=Newton_Verfahren(F, dF,  x0)

m=1;
x=x0;
fprintf('\n\n n    x_n             F(x_n)\n%2.0f%16.10f%16.10f\n',
        m-1,x(m),F(x(m)))

while abs(F(x(m))) > 10^-8,

    x(m+1) = x(m) - F(x(m)) / dF(x(m));
    fprintf('%2.0f%16.10f%16.10f\n',m,x(m+1),F(x(m+1)))
    m=m+1;

end


% -----------------------------------------------------------------------
% Der Einfachheit halber wurde in dem vorstehenden m-File auch die
% Ausgabe der Iterationsschritte mit dem fprintf-Befehl realisiert.
%
% Wenn ein solches m-File für aufwändigere Berechnungen genutzt wird,
% sollte dieses möglichst keine Ausgabebefehle für Zwischenschritte
% enthalten.
% -----------------------------------------------------------------------
```

```
% b) -------------------------------------------------------------------

% Listing des m-Files HA01_matlab_solution_f.m für die Funktion f
type HA01_matlab_solution_f.m


function y=HA01_matlab_solution_f(x)
   y=log(x+1).*sin(x.^2);

% ---------------------------------------------------------------------

% Listing des m-Files HA01_matlab_solution_df.m für die Ableitung von f
type HA01_matlab_solution_df.m


function y=HA01_matlab_solution_df(x)
    y = sin(x.^2)./(x+1) + 2*x.*log(x+1).*cos(x.^2);

% ---------------------------------------------------------------------

% 1. Startwert setzen
x0=1.50;
% Newton-Verfahren auf Funktion f(x) = ln(x+1) * sin(x^2) anwenden
x_newton_1=HA02_matlab_solution_Newton_Verfahren( @HA01_matlab_solution_f,
                                           @HA01_matlab_solution_df , x0);
 n     x_n              F(x_n)
 0     1.5000000000    0.7129412590
 1     2.0036533031   -0.8427791426
 2     1.7306319547    0.1466439023
 3     1.7739377079   -0.0053689696
 4     1.7724552348   -0.0000050025
 5     1.7724538509   -0.0000000000

% 2. Startwert setzen
x0=1.43;
% Newton-Verfahren auf Funktion f(x) = ln(x+1) * sin(x^2) anwenden
x_newton_2=HA02_matlab_solution_Newton_Verfahren( @HA01_matlab_solution_f,
                                           @HA01_matlab_solution_df , x0);
 n     x_n              F(x_n)
 0     1.4300000000    0.7899588841
 1     2.4259215887   -0.4773508498
 2     2.5144168797    0.0491399509
 3     2.5066500514    0.0001369752
 4     2.5066282748    0.0000000013

% 3. Startwert setzen
x0=1.40;
% Newton-Verfahren auf Funktion f(x) = ln(x+1) * sin(x^2) anwenden
x_newton_3=HA02_matlab_solution_Newton_Verfahren( @HA01_matlab_solution_f,
                                           @HA01_matlab_solution_df , x0);
 n     x_n              F(x_n)
 0     1.4000000000    0.8099937619
 1     2.8871820468    1.2031045389
 2     3.2405112138   -1.2715088923
 3     2.9672152060    0.8011657528
 4     3.0903309575   -0.1761336286
 5     3.0700121012   -0.0002755927
 6     3.0699801242   -0.0000000030
```

```
x=linspace(-0.5,5,300);
% Funktion auswerten
y=HA01_matlab_solution_f(x);
y_newton_1=HA01_matlab_solution_f(x_newton_1);
y_newton_2=HA01_matlab_solution_f(x_newton_2);
y_newton_3=HA01_matlab_solution_f(x_newton_3);

% Funktion zeichnen
figure(2); clf; hold on;
plot(x,y,'k-','LineWidth',2);
% Iterationsverläufe einzeichnen
plot(x_newton_1, y_newton_1,'xb--','MarkerSize',20);
plot(x_newton_2, y_newton_2,'xr--','MarkerSize',20);
plot(x_newton_3, y_newton_3,'xg--','MarkerSize',20);
% Startwerte hervorheben
plot(x_newton_1(1), y_newton_1(1),'ob','MarkerSize',10);
plot(x_newton_2(1), y_newton_2(1),'or','MarkerSize',10);
plot(x_newton_3(1), y_newton_3(1),'og','MarkerSize',10);
% Letzte Iterierte hervorheben
plot(x_newton_1(end), y_newton_1(end),'.b','MarkerSize',30);
plot(x_newton_2(end), y_newton_2(end),'.r','MarkerSize',30);
plot(x_newton_3(end), y_newton_3(end),'.g','MarkerSize',30);

% Label, Title, Legend
xlabel('x'); ylabel('y'); grid on;
title('Iterationsverlauf des Newton-Verfahrens für verschiedene
Startwerte');
legend('ln(x+1)*sin(x^2)', 'Iterationsverlauf für x0=1.50', '
Iterationsverlauf für x0=1.43', 'Iterationsverlauf für x0=1.40', '
Location', 'NorthWest');
print -depsc HA02_matlab_plot_2b.eps
```



```
% c) -----------------------------------------------------------------

% Listing des m-Files HA02_matlab_solution_ddf.m für die 2. Abl. von f
type HA02_matlab_solution_ddf.m

function y=HA02_matlab_solution_ddf(x)
   y = -sin(x.^2) ./ (x+1).^2 + 4*cos(x.^2).*x ./ (x+1)
                 - 4*log(x+1).*sin(x.^2).*x.^2 + 2*log(x+1).*cos(x.^2);

% --------------------------------------------------------------------

% 1. Startwert setzen
x0=1.70;
% Newton-Verfahren auf die Ableitung der Funktion f(x)= ln(x+1) * sin(x^2)
% anwenden
x_newton_1=HA02_matlab_solution_Newton_Verfahren(@HA01_matlab_solution_df,
                                          @HA02_matlab_solution_ddf , x0);
 n    x_n              F(x_n)
 0    1.7000000000   -3.1785342143
 1    1.2619286162    0.3973648832
 2    1.3344789358   -0.0528211827
 3    1.3268414970   -0.0005550841
 4    1.3267595102   -0.0000000645
 5    1.3267595006   -0.0000000000

% 2. Startwert setzen
x0=3.17;
% Newton-Verfahren auf die Ableitung der Funktion f(x)= ln(x+1) * sin(x^2)
% anwenden
x_newton_2=HA02_matlab_solution_Newton_Verfahren(@HA01_matlab_solution_df,
                                          @HA02_matlab_solution_ddf , x0);
 n    x_n              F(x_n)
 0    3.1700000000   -7.4864270413
 1    3.4300358051    6.9426812567
 2    3.3028109460   -1.0692309434
 3    3.3197918903    0.0157454002
 4    3.3195484906    0.0000023162
 5    3.3195484548    0.0000000000

% 3. Startwert setzen
x0=0.70;
% Newton-Verfahren auf die Ableitung der Funktion f(x)= ln(x+1) * sin(x^2)
% anwenden
x_newton_3=HA02_matlab_solution_Newton_Verfahren(@HA01_matlab_solution_df,
                                          @HA02_matlab_solution_ddf , x0);
 n    x_n              F(x_n)
 0    0.7000000000    0.9323057960
 1    0.1633666388    0.0723612045
 2    0.0776187517    0.0171950662
 3    0.0378533558    0.0041934572
 4    0.0186937033    0.0010354985
 5    0.0092893247    0.0002572838
 6    0.0046303689    0.0000641232
 7    0.0023116220    0.0000160061
 8    0.0011549218    0.0000039985
 9    0.0005772388    0.0000009992
10    0.0002885639    0.0000002498
11    0.0001442681    0.0000000624
12    0.0000721306    0.0000000156
13    0.0000360644    0.0000000039

x=linspace(-0.5,5,300);
% Funktion auswerten
y=HA01_matlab_solution_f(x);
y_newton_1=HA01_matlab_solution_f(x_newton_1);
y_newton_2=HA01_matlab_solution_f(x_newton_2);
y_newton_3=HA01_matlab_solution_f(x_newton_3);

% Funktion zeichnen
figure(3); clf; hold on;
plot(x,y,'k-','LineWidth',2);
% Iterationsverläufe einzeichnen
plot(x_newton_1, y_newton_1,'xb--','MarkerSize',20);
plot(x_newton_2, y_newton_2,'xr--','MarkerSize',20);
plot(x_newton_3, y_newton_3,'xg--','MarkerSize',20);
% Startwerte hervorheben
plot(x_newton_1(1), y_newton_1(1),'ob','MarkerSize',10);
plot(x_newton_2(1), y_newton_2(1),'or','MarkerSize',10);
plot(x_newton_3(1), y_newton_3(1),'og','MarkerSize',10);
% Letzte Iterierte hervorheben
plot(x_newton_1(end), y_newton_1(end),'.b','MarkerSize',30);
plot(x_newton_2(end), y_newton_2(end),'.r','MarkerSize',30);
plot(x_newton_3(end), y_newton_3(end),'.g','MarkerSize',30);

% Label, Title, Legend
xlabel('x'); ylabel('y'); grid on;
title('Iterationsverlauf des Newton-Verfahrens für Nullstellen der
Ableitung für verschiedene Startwerte');
legend('ln(x+1)*sin(x^2)', 'Iterationsverlauf für x0=1.70', '
Iterationsverlauf für x0=3.17', 'Iterationsverlauf für x0=0.70', '
Location', 'NorthWest');
print -depsc HA02_matlab_plot_2c1.eps

% Analog für die Ableitung von f
% Ableitung auswerten
dy=HA01_matlab_solution_df(x);
dy_newton_1=HA01_matlab_solution_df(x_newton_1);
dy_newton_2=HA01_matlab_solution_df(x_newton_2);
dy_newton_3=HA01_matlab_solution_df(x_newton_3);

% Ableitung zeichnen
figure(4); clf; hold on;
plot(x,dy,'k-','LineWidth',2);
% Iterationsverläufe einzeichnen
plot(x_newton_1, dy_newton_1,'xb--','MarkerSize',20);
plot(x_newton_2, dy_newton_2,'xr--','MarkerSize',20);
plot(x_newton_3, dy_newton_3,'xg--','MarkerSize',20);
% Startwerte hervorheben
plot(x_newton_1(1), dy_newton_1(1),'ob','MarkerSize',10);
plot(x_newton_2(1), dy_newton_2(1),'or','MarkerSize',10);
plot(x_newton_3(1), dy_newton_3(1),'og','MarkerSize',10);
% Letzte Iterierte hervorheben
plot(x_newton_1(end), dy_newton_1(end),'.b','MarkerSize',30);
plot(x_newton_2(end), dy_newton_2(end),'.r','MarkerSize',30);
plot(x_newton_3(end), dy_newton_3(end),'.g','MarkerSize',30);
```

```
% Label, Title, Legend
xlabel('x'); ylabel('y'); grid on;
title('Iterationsverlauf des Newton-Verfahrens für Nullstellen der
Ableitung für verschiedene Startwerte');
legend('Ableitung von f(x)=ln(x+1)*sin(x^2)', 'Iterationsverlauf für x0=1.
70', 'Iterationsverlauf für x0=3.17', 'Iterationsverlauf für x0=0.70', '
Location', 'NorthWest');
print -depsc HA02_matlab_plot_2c2.eps


% Zweite Ableitungen an den letzten Iterierten bestimmen

HA02_matlab_solution_ddf(x_newton_1(end))
ans =
   -6.7688
% Zweite Ableitung negativ ==> lokales Maximum

HA02_matlab_solution_ddf(x_newton_2(end))
ans =
   64.6704
% Zweite Ableitung positiv ==> lokales Minimum

HA02_matlab_solution_ddf(x_newton_3(end))
ans =
  2.1638e-004
% Zweite Ableitung nahe Null ==> möglicherweise Sattelpunkt

diary off
```