

# Aufgabe 1 Beispiel 5 Arcustangens

Choi Hyungi und Matkovich Sebastian

bei Winfried Auzinger und Ewa Weinmüller

**Zusammenfassung**—Es wird ein Programm zur Berechnung der Arcustangensfunktion  $\arctan(x)$  entwickelt. Die Annäherung erfolgt durch die Taylorreihe um  $x_0 = 0$ , die für  $|x| \leq 1$  konvergiert. Die notwendigen Taylorreihenglieder zur Erreichung einer Genauigkeit von mindestens 100 eps werden berechnet und grafisch dargestellt.

Relative Fehler bei konkreten  $x$ -Werten werden untersucht und mit dem malleigenen Arcustangens verglichen, wobei die Genauigkeit bei natürlicher und umgekehrter Summationsreihenfolge betrachtet wird. Für  $x$ -Werte nahe 1 wird eine alternative, schneller konvergierende Reihe verwendet und getestet. Schließlich wird ein Vergleich mit Lagrangeinterpolation mit äquidistanten Stützstellen und mit durch Chebyshevpolynome definierte Stützstellen hergestellt. Verwendet werden ein HP Envy x360 und ein Razer Blade Stealth 13 2020.

## Inhaltsverzeichnis

<b>1</b>	<b>Parameters</b>	<b>1</b>
1.1	Beispiel 5.1	1
1.2	Beispiel 5.2	1
<b>2</b>	<b>Taylorentwicklung um 0</b>	<b>2</b>
2.1	Beispiel 5.3	2
2.2	Beispiel 5.4	2
<b>3</b>	<b>Taylorentwicklung um 1</b>	<b>2</b>
3.1	Beispiel 5.5	2
<b>4</b>	<b>selektive Taylorentwicklung</b>	<b>2</b>
4.1	Beispiel 5.6, $n$ numerisch	2
4.2	Beispiel 5.6, $n$ analytisch	3
<b>5</b>	<b>Lagrange Interpolation</b>	<b>3</b>
5.1	Beispiel 5.7	3
<b>6</b>	<b>Quellcode</b>	<b>4</b>
6.1	Mains	4
6.2	Anzahl der Taylorterme	5
6.3	Taylorreihe um 0	6
6.4	Taylorreihe um 1	7
6.5	selective Taylor	7
6.6	Lagrange Interpolation	8
6.7	Lagrange Interpolation	8
6.8	Lagrange Interpolation	8

## 1. Parameters

### 1.1. Beispiel 5.1

Es ist sinnvoll für den Bereich  $-1$  bis  $1$  die Approximation zu berechnen. Da der Wertebereich um  $0$  antisymmetrisch ist, kann er auf das Intervall  $0$  bis  $1$  eingeschränkt werden. Außerhalb dieses Bereichs nähert sich der Funktionswert  $\frac{\pi}{2}$  an. Die trivialen Funktionswerte sind für das Argument  $0$ :  $0$ , für das Argument  $1$ :  $\frac{\pi}{4}$  und für das Argument  $-1$ :  $-\frac{\pi}{4}$ . Für die Entwicklungen haben die folgende Reihe verwendet:

$$x = 1 - 2^{-k}, \quad k = 1 \dots, 19$$

### 1.2. Beispiel 5.2

Die Anzahl nötiger Reihenglieder für eine gewünschte Genauigkeit lassen sich mit dem Term abschätzen, der entsteht, wenn in der Taylorreihenentwicklung des Arcustangens der Summenindex  $n$ , bis zu dem Summiert wird, um  $1$  erhöht wird. Dann entsteht folgender

Term:  $\frac{x^{2n+3}}{2n+3}$ . Mit der Division durch  $\arctan(x)$  erhalten wir eine Abschätzung für den relativen Fehler. Wenn wir diesen relativen Fehler gleich 100eps setzen, können wir numerisch nach  $n$  auflösen. Wenn der Betrag von  $x$  kleiner als  $1$  ist, konvergiert dieser Fehler und 100eps können erreicht werden. Genau bei  $1$  müssten in etwa  $10^{14}$  Werte Summiert werden, da bei uns 100eps in etwa  $10^{-14}$  ist. Bei Werten im Betrag kleiner  $1$  könnte auch eine Umformung folgendermaßen geschehen:

$$\begin{aligned} \frac{x^{2n+3}}{2n+3} &\leq 100\text{eps} \quad \left| \cdot \arctan(x) \right. \\ \frac{x^{2n+3}}{2n+3} &\leq 100\text{eps} \arctan(x) \quad \left| \frac{d}{dx} \right. \\ x^{2n+2} &\leq \frac{100\text{eps}}{1+x^2} \quad \left| \ln \right. \\ (2n+2) \ln(x) &\leq \ln\left(\frac{100\text{eps}}{1+x^2}\right) \quad \left| \frac{1}{\ln(x)} \right. \\ (2n+2) &\leq \frac{\ln\left(\frac{100\text{eps}}{1+x^2}\right)}{\ln(x)} \quad \left| \frac{1}{2} \right. \quad \left| -1 \right. \\ n &\leq \frac{\ln\left(\frac{100\text{eps}}{1+x^2}\right)}{2 \ln(x)} - 1 \end{aligned}$$

Dieser Ausdruck liefert eine schnellere Näherung an die Anzahl benötigter Reihenglieder um die Genauigkeit von 100eps zu erreichen. Vorallem wächst die Rechenzeit der numerische Methode exponentiell je näher wir bei eins sind im Vergleich zur analytische Methode.

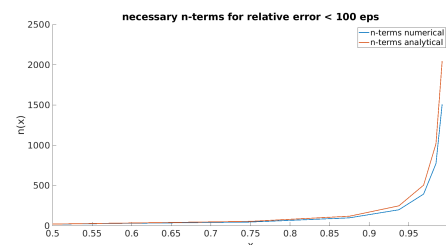


Abbildung 1. Anzahl benötigter Taylorreiheterme bis nahe 1.

Bei genau 1 verhält sich die analytische Methode asymptotisch während die numerische im Bereich von  $10^{14}$  ist.

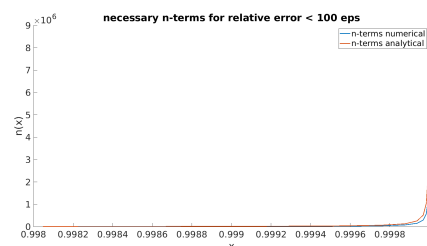


Abbildung 2. Anzahl benötigter Taylorreiheterme bis nahe 1.

## 2. Taylorentwicklung um 0

### 2.1. Beispiel 5.3

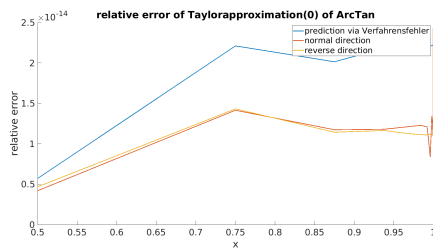


Abbildung 3. relativer Fehler bei Entwicklung bis nahe 1.

Die relative Genauigkeit wird für  $|x| < 1$  erreicht, jedoch müssen immer mehr Werte aufsummiert werden, damit diese erreicht wird.

### 2.2. Beispiel 5.4

Wie wir an der Abschätzung erkennen können müssen für Argumente, die sich im Betrag 1 annähern, immer mehr Werte summiert werden, um die geforderte Genauigkeit zu erreichen. Bei Summation in umgekehrter Reihenfolge sind die zu summierenden Werte eher in gleicher Größenordnung und damit kann die Fehlerquelle, dass kleine Zahlen zu großen dazuaddiert, diese Zahl nicht mehr erhöhen, weil die Größenordnungen zu weit auseinander liegen, weitestgehend ausgeschlossen werden. Deshalb wird die Genauigkeit der umgekehrten Reihenfolge im Bereich 1, wo die  $n$  Werte exponential ansteigen, deutlich höher.

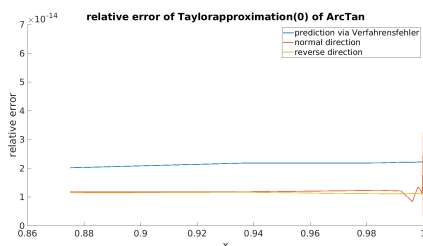


Abbildung 4. relativer Fehler bei Entwicklung nahe 1.

Zusätzlich können wir bei niedrigen  $x$ -Werten von Abb.3 auch beobachten, dass bei niedrigen  $n$  Werten zufällige Rechenfehler auftauchen, so dass das Kommutativgesetz nicht erhalten bleibt.

## 3. Taylorentwicklung um 1

### 3.1. Beispiel 5.5

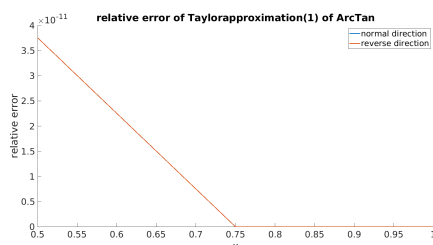


Abbildung 5. relativer Fehler bei Entwicklung um 1.

Hier sieht man, dass Reihenentwicklung des Arcustanges um 1 nur für  $x$  werten größer als ca. 0,75 sinnvoll ist. Und in der folgenden Abbildung können wir auch sehen dass die relative Genauigkeit der umgekehrten Summationsreihenfolge generell um Größenordnungen besser als die normale.

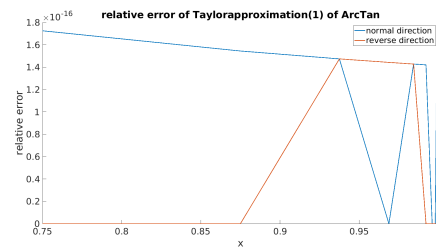


Abbildung 6. relativer Fehler bei Entwicklung zwischen 0.75 und 1.

Und im Vergleich zu der Taylorentwicklung um 0 ist die Taylorentwicklung um 1 ab 0.75 besonders hervorragend.

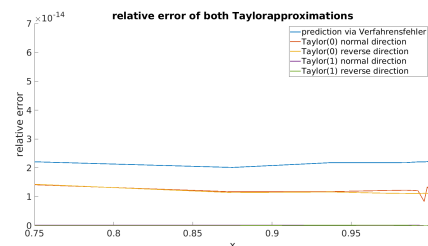


Abbildung 7. relativer Fehler beider Taylorentwicklungen zwischen 0.75 und 1.

## 4. selektive Taylorentwicklung

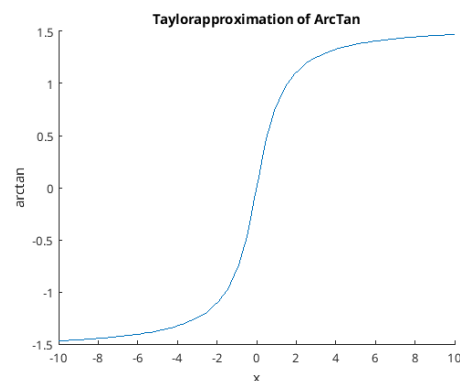


Abbildung 8. Taylorapproximation von arctan.

### 4.1. Beispiel 5.6, $n$ numerisch

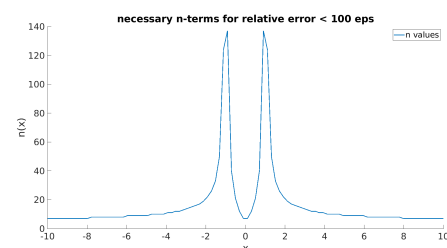


Abbildung 9. Anzahl benötigter Reihenglieder.

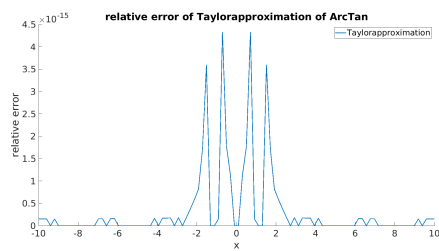


Abbildung 10. Relativer Fehler mit Taylorreihenentwicklung.

#### 4.2. Beispiel 5.6, n analytisch

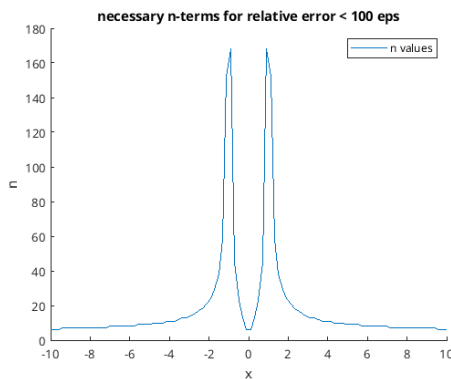


Abbildung 11. Anzahl benötigter Reihenglieder.

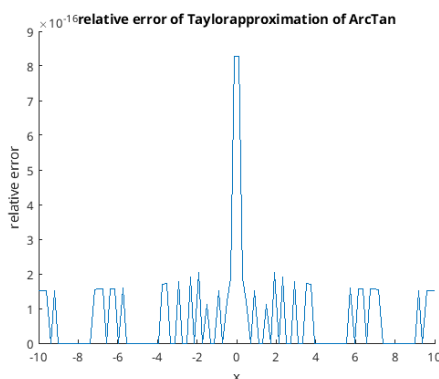


Abbildung 12. Relativer Fehler mit Taylorreihenentwicklung.

### 5. Lagrange Interpolation

#### 5.1. Beispiel 5.7

Die relative Genauigkeit bei Lagrange Interpolation mit äquidistanter Stützstellen zwischen 0 und 1, ist bei 20 Stützstellen maximal. Die extreme Ungenauigkeit an die Grenzwerten kommt durch die Unstetigkeit der Stützstellen in diesem Bereich zustande.

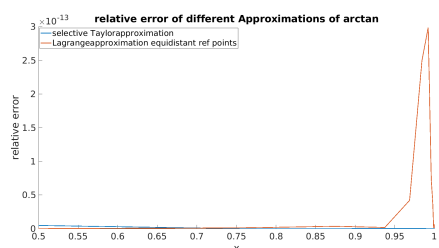


Abbildung 13. Lagrange Interpolation mit äquidistanter Stützstellen.

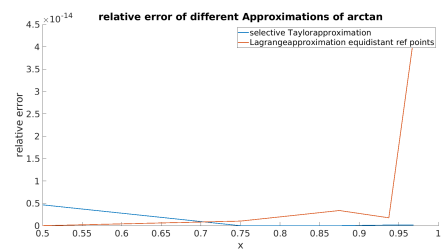


Abbildung 14. Lagrange Interpolation mit äquidistanter Stützstellen bis 0.95.

Bei der Lagrange Interpolation mit Chebyshev Knoten zwischen  $[-1, 1]$  sehen ebenfalls einen extremen Anstieg an den Grenzwerten aus dem selben Grund. Allerdings wächst bei Chebyshev Knoten die relative Genauigkeit mit der Anzahl der Stützstellen, falls man den Bereich um die 1 vernachlässigt. Um den gesamten Bereich unter 100 eps zu halten ist die Anzahl von 36 Stützstellen optimal.

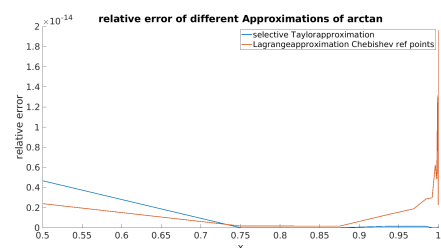


Abbildung 15. Lagrange Interpolation mit Chebyshev Knoten.

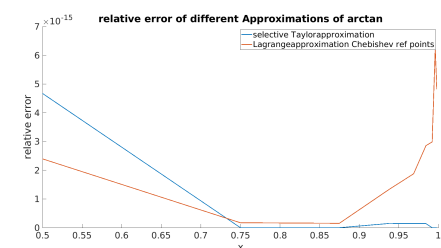


Abbildung 16. Lagrange Interpolation mit Chebyshev Knoten bis nahe 1.

In den folgenden Graphen kann man beobachten das die Lagrange Interpolationen generell eine höhere Genauigkeit aufweisen als die Taylorentwicklung. Die höhere Genauigkeit der Taylorreihe für  $x > 0.75$  kommt dadurch zustande weil die Taylorentwicklung um 1 mit so vielen Gliedern errechnet wird, dass ihre Genauigkeit, wie in Abb.7 erkennbar, in eine anderen Größen Ordnung ist.

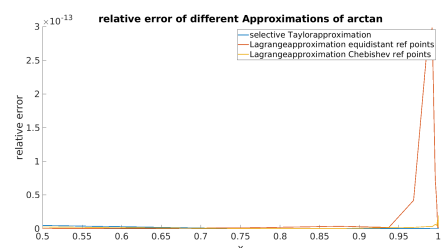


Abbildung 17. relativer Fehler von verschiedener Approximationen.

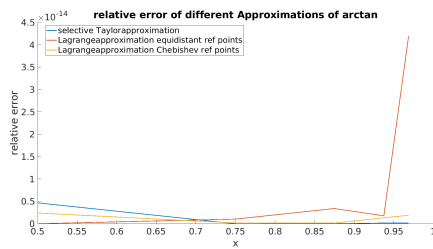


Abbildung 18. relativer Fehler von verschiedener Approximationen bis nahe 1.

## 6. Quellcode

### 6.1. Mains

```

1 clear variables;
2 close all;
3 %{
4 Task:
5 focused primarily on Analysing Approximations in
6 the interval [0, 1]
7 %}
8
9 %-----
10 % x Values
11 %-----
12 % Parameters
13 xpts = 19;
14 xmin = 0;
15 xmax = 1;
16
17 xs = xvalues(xmax, xpts);
18 %xs = linspace(xmin, xmax, xpts);
19
20 %-----
21 % minimum n terms for the Taylorapproximation
22 via Verfahrensfehler
23 % to keep relative Error below 100 eps
24 %-----
25 [ns_num, ers_num] = n_numerical(xs); %
26 numerically
27 [ns_ana, ers_ana] = n_analytical(xs); %
28 analytically
29
30 ns = ns_num; % n values
31 ers_ns = ers_num; % relative error according
32 to Verfahrensfehler
33 disp('ns ok');
34
35 % plot n-terms
36 figure;
37 idx_strt = 1;
38 idx_end = fix(length(xs)*0.4); % 1 for
39 fullrange; 0.4 for relevant range
40
41 hold on;
42 plot(xs(idx_strt:idx_end), ns_num(idx_strt:
43 idx_end), 'DisplayName', 'n-terms numerical');
44 plot(xs(idx_strt:idx_end), ns_ana(idx_strt:
45 idx_end), 'DisplayName', 'n-terms analytical');
46
47 title('necessary n-terms for relative error
48 < 100 eps');
49 xlabel('x');
50 ylabel('n');
51 legend;
52 hold off;
53
54 %-----
55 % Taylorsequence around 0
56 %-----
57 [arctans_taylor0, ers_taylor0] = taylor0_arctan(
58 xs, ns); % Taylorapprox(0)
59 normal direction

```

```

48 [rev_arctans_taylor0, rev_ers_taylor0] =
49 reversetaylor0_arctan(xs, ns); %
50 Taylorapprox(0) reverse direction
51
52 %plot relative Error
53 figure
54 idx_strt = 1;
55 idx_end = fix(length(xs)*0.7);
56
57 hold on;
58 plot(xs(idx_strt:idx_end), ers_ns(idx_strt:
59 idx_end), 'DisplayName', 'prediction via
60 Verfahrensfehler');
61 plot(xs(idx_strt:idx_end), ers_taylor0(
62 idx_strt:idx_end), 'DisplayName', 'normal
63 direction');
64 plot(xs(idx_strt:idx_end), rev_ers_taylor0(
65 idx_strt:idx_end), 'DisplayName', 'reverse
66 direction');
67
68 title('relative error of Taylorapproximation
69 (0) of ArcTan');
70 xlabel('x');
71 ylabel('relative error');
72 legend;
73 hold off;
74
75 %-----
76 % Taylorapprox around 1
77 %-----
78 [arctans_taylor1, ers_taylor1] = taylor1_arctan(
79 xs, ns); % Taylorapprox(1)
80 normal order
81 [rev_arctans_taylor1, rev_ers_taylor1] =
82 reversetaylor1_arctan(xs, ns); %
83 Taylorapprox(1) reverse order
84
85 %plot relative Error to arctan(x)
86 figure
87 idx_strt = fix(length(xs)*0.15); %fix(length
88 (xs)*0.15) f r relevante range
89 idx_end = length(xs);
90
91 hold on;
92 %plot(xs(idx_strt:idx_end), ers_ns(idx_strt:
93 idx_end), 'DisplayName', 'prediction via
94 Verfahrensfehler');
95 plot(xs(idx_strt:idx_end), ers_taylor1(
96 idx_strt:idx_end), 'DisplayName', 'normal
97 direction');
98 plot(xs(idx_strt:idx_end), rev_ers_taylor1(
99 idx_strt:idx_end), 'DisplayName', 'reverse
100 direction');
101
102 title('relative error of Taylorapproximation
103 (1) of ArcTan');
104 xlabel('x');
105 ylabel('relative error');
106 legend;
107 hold off;
108
109 %-----
110 % Compare Taylorapprox(0) and Taylorapprox(1)
111 %-----
112 figure
113 idx_strt = fix(length(xs)*0.15);
114 idx_end = length(xs);
115
116 hold on;
117 plot(xs(idx_strt:idx_end), ers_ns(idx_strt:
118 idx_end), 'DisplayName', 'prediction via
119 Verfahrensfehler');
120 plot(xs(idx_strt:idx_end), ers_taylor0(
121 idx_strt:idx_end), 'DisplayName', 'Taylor(0)
122 normal direction');
123 plot(xs(idx_strt:idx_end), rev_ers_taylor0(
124 idx_strt:idx_end), 'DisplayName', 'Taylor(0)
125 reverse direction');
126 plot(xs(idx_strt:idx_end), ers_taylor1(
127 idx_strt:idx_end), 'DisplayName', 'Taylor(1)
128 normal direction');

```

```

100 plot(xs(idx_strt:idx_end), rev_ers_taylor1(
    idx_strt:idx_end), 'DisplayName', 'Taylor(1)
    reverse direction')
101
102 title('relative error of both
    Taylorapproximations');
103 xlabel('x');
104 ylabel('relative error');
105 legend;
106 hold off;
107
108 %-----
109 % selective Taylorapprox
110 %-----
111 [arctans_taylor, ers_taylor] = fulltaylor_arctan(
    xs, ns); % selective Taylorapprox(0 or 1)
    , reverse order
112
113 %-----
114 % Lagrangeapprox
115 %-----
116 refmin = 0;
117 refmax = 1;
118 refpts1 = 20; % max approximation at 20
119 refs1 = linspace(refmin, refmax, refpts1);
120 refpts2 = 36; % min 36 to be comparable to
    taylor
121 refs2 = chebyshevnodes(refpts2);
122
123 [arctans4, ers4] = lagrangeapprox(xs, refs1);
124 [arctans5, ers5] = lagrangeapprox(xs, refs2);
125
126 figure
127     idx_strt = 1;
128     idx_end = fix(length(xs)*0.3); % 0.5 for rel
129
130     hold on;
131     plot(xs(idx_strt:idx_end), ers_taylor(
        idx_strt:idx_end), "DisplayName", 'selective
        Taylorapproximation')
132     plot(xs(idx_strt:idx_end), ers4(idx_strt:
        idx_end), "DisplayName", '
        Lagrangeapproximation equidistant ref points
        ')
133     plot(xs(idx_strt:idx_end), ers5(idx_strt:
        idx_end), "DisplayName", '
        Lagrangeapproximation Chebishev ref points')
134
135     title('relative error of different
        Approximations of arctan');
136     xlabel('x');
137     ylabel('relative error');
138     legend;
139     hold off;
140
141 figure
142     idx_strt = 1;
143     idx_end = length(xs);
144
145     hold on;
146     plot(xs(idx_strt:idx_end), arctans_taylor(
        idx_strt:idx_end), "DisplayName", 'selective
        Taylorapproximation')
147     plot(xs(idx_strt:idx_end), arctans4(idx_strt:
        idx_end), "DisplayName", '
        Lagrangeapproximation equidistant ref points
        ')
148     plot(xs(idx_strt:idx_end), arctans5(idx_strt:
        idx_end), "DisplayName", '
        Lagrangeapproximation Chebishev ref points')
149
150     title('arctan of different Approximations');
151     xlabel('x');
152     ylabel('arctan(x)');
153     legend;
154     hold off;

```

Code 1. main01.

```

1 clear variables;
2 close all;
3
4 %-----
5 % x Values
6 %-----
7 % Parameters
8 xpts = 100;
9 xmin = -10;
10 xmax = 10;
11
12 %xs = xvalues(xmax, kmax);
13 xs = linspace(xmin, xmax, xpts);
14
15 %-----
16 % minimum n Values to each x
17 %-----
18 [ns_num, ers_num] = n_numerical(xs);
19 [ns_ana, ers_ana] = n_analytical(xs);
20
21 ns = ns_ana; % n values
22 ers_ns = ers_ana; % errors via
    Verfahrensfehler
23 ns = ns + 1;
24 disp('ns, ok')
25
26 % plot necessary n-terms
27 figure;
28 hold on;
29 plot(xs, ns, 'DisplayName', 'n values');
30 title('necessary n-terms for relative error <
    100 eps');
31 xlabel('x');
32 ylabel('n');
33 legend;
34 hold off;
35
36 %-----
37 % Taylorapproximation of arctan
38 %-----
39 [arctans, ers] = fulltaylor_arctan(xs, ns);
40
41 figure
42     idx_strt = 1;
43     idx_end = length(xs);
44
45     hold on;
46     plot(xs(idx_strt:idx_end), arctans(idx_strt:
        idx_end))
47
48     title('Taylorapproximation of ArcTan');
49     xlabel('x');
50     ylabel('arctan');
51     hold off;
52
53 figure
54     idx_strt = 1;
55     idx_end = length(xs);
56
57     hold on;
58     plot(xs(idx_strt:idx_end), ers(idx_strt:
        idx_end), 'DisplayName', 'Taylorapproximation
        ')
59     plot(xs(idx_strt:idx_end), ers_ns(idx_strt:
        idx_end), 'DisplayName', 'Verfahrensfehler')
60
61     title('relative error of Taylorapproximation
        of ArcTan');
62     xlabel('x');
63     ylabel('relative error');
64     legend;
65     hold off;

```

Code 2. main02.

## 6.2. Anzahl der Taylortermes

```

1 function xs = xvalues(xmax, kmax)
2     xs = zeros(kmax, 1);

```

```

3   for k = 1 : kmax
4       x = xmax - 2^(-k);
5       xs(k) = x;
6   end
7   end

```

Code 3. x werte

```

34      ns(k) = n;
35      ers(k) = er;
36   end
37   end

```

Code 5. n numerisch.

```

1   function [ns, ers] = n_analytical(xs)
2   %{
3   Parameter:
4       array of x values
5   Todo:
6       calculate n values analytically
7   Return:
8       [array, array]
9       1st array: n-values to each x
10      2nd array: relative error to each x from
11      Verfahrensfehler
12  %}
13  vmax = length(xs);
14  ns = zeros(vmax,1); % array of n values
15  ers = zeros(vmax,1); % array of relative
16  errors
17  for k = 1 : vmax
18      o = xs(k);
19      if abs(o) <= 1
20          x = abs(o);
21      else
22          x = abs(1/o);
23      end
24      comp = atan(x);
25      n = log(100*eps/(1+x^2)) / (2*log(x)) -
26      1;
27      er = (x^(2*n+3)/(2*n+3))/comp;
28      ers(k) = abs(er);
29      ns(k) = fix(n);
30  end
31  end

```

Code 4. n analytisch.

```

1   function [ns, ers] = n_numerical(xs)
2   %{
3   Parameter:
4       array of x values
5   Todo:
6       determine n values numerically
7   Return:
8       [array, array]
9       1st array: n-values to each x
10      2nd array: relative error to each x from
11      Verfahrensfehler
12  %}
13  vmax = length(xs);
14  ns = zeros(vmax, 1); % array of n values
15  ers = zeros(vmax, 1); % array of relative
16  errors
17  for k = 1 : vmax
18      er = 1;
19      o = xs(k);
20      if abs(o) <= 1
21          x = abs(o);
22      else
23          x = abs(1/o);
24      end
25      n = -1;
26      comp = atan(x);
27      if comp == 0
28          n = 0;
29      else
30          while er > 100*eps
31              n = n+1;
32              er = (x^(2*n+3)/(2*n+3))/comp;
33              er = abs(er);
34          end
35      end
36  end
37  end

```

### 6.3. Taylorreihe um 0

```

1   function termN = nterm_taylor0_arctan(x, n)
2   %{
3   Parameter:
4       x: x value
5       n: order of term
6   Return:
7       n-th term of the Taylorsequence around 0
8   %}
9   term = 2*n + 1;
10  termN = (-1)^n * x^term/term;
11  end

```

Code 6. n-ter glied von Taylorreihe um 0.

```

1   function [arctans, ers] = taylor0_arctan(xs, ns)
2   %{
3   Parameter:
4       xs: array of x values
5       ns: array of n values
6   Todo:
7       1.) approximate arctan(x) via Taylorsequence
8       (0) in normal order
9       2.) calculate relative error of that
10      approximation
11   Return:
12      [array01, array02]
13      array01: array of arctan(x) to each x
14      array02: array of relative errors to each x
15  %}
16  arctans = zeros(length(xs),1); % array of
17  arctan(x)
18  ers = zeros(length(xs), 1); % array of
19  relative errors
20  for k = 1 : length(xs)
21      x = xs(k);
22      arctan = 0;
23      comp = atan(x);
24      for i = 0 : ns(k)
25          arctan = arctan +
26          nterm_taylor0_arctan(x,i);
27      end
28      arctans(k) = arctan;
29      ers(k) = abs((arctan - comp)/comp);
30  end
31  end

```

Code 7. Taylorentwicklung um 0 in normal.

```

1   function [arctans, ers] = reversetaylor0_arctan(
2   xs, ns)
3   %{
4   Parameter:
5       xs: array of x values
6       ns: array of n values
7   Todo:
8       1.) approximate arctan(x) via Taylorsequence
9       (0) in reverse order
10      2.) calculate relative error of that
11      approximation
12   Return:
13      [array01, array02]
14      array01: array of arctan(x) to each x
15      array02: array of relative errors to each x
16  %}
17  arctans = zeros(length(xs),1); % array of
18  arctan(x)

```

```

15     ers = zeros(length(xs), 1);      % array of
16     relative errors
17     for k = 1 : length(xs)
18         x = xs(k);
19         arctan = 0;
20         comp = atan(x);
21         for i = ns(k) : (-1) : 0
22             arctan = arctan +
23             nterm_taylor0_arctan(x,i);
24         end
25         arctans(k) = arctan;
26         ers(k) = abs((arctan - comp)/comp);
27     end
28 end

```

Code 8. Taylorentwicklung um 0 in umgekehrt.

#### 6.4. Taylorreihe um 1

```

1 function arctan = nterm_taylor1_arctan(delta, k)
2 %{
3 Parameter:
4     delta: delta between x and 1
5     k: order of term
6 Return:
7     k-th term of the Taylorsequence around 1
8 %}
9 arctan = 2^(-k/2)*delta^k*sin(3*k*pi/4)/k;
10 end

```

Code 9. n-ter glied von Taylorreihe um 1

```

1 function [arctans, ers] = taylor1_arctan(xs, ns)
2 %{
3 Parameter:
4     xs: array of x values
5     ns: array of n values
6 Todo:
7     1.) approximate arctan(x) via Taylorsequence
8     (1) in normal order
9     2.) calculate relative error of that
10    approximation
11 Return:
12    [array01, array02]
13    array01: array of arctan(x) to each x
14    array02: array of relative errors to each x
15 %}
16 arctans = zeros(length(xs),1); % array of
17 arctan(x)
18 ers = zeros(length(xs), 1); % array of
19 relative errors
20 for k = 1 : length(xs)
21     x = xs(k);
22     delta = x - 1;
23     arctan = pi/4;
24     comp = atan(x);
25     for i = 1 : ns(k)
26         arctan = arctan +
27         nterm_taylor1_arctan(delta,i);
28     end
29     arctans(k) = arctan;
30     ers(k) = abs((arctan - comp)/comp);
31 end
32 end

```

Code 10. Taylorentwicklung um 1 normal

```

1 function [arctans, ers] = reversetaylor1_arctan(
2     xs, ns)
3 %{
4 Parameter:
5     xs: array of x values
6     ns: array of n values
7 Todo:
8     1.) approximate arctan(x) via Taylorsequence
9     (1) in reverse order

```

```

8     2.) calculate relative error of that
9     approximation
10 Return:
11    [array01, array02]
12    array01: array of arctan(x) to each x
13    array02: array of relative errors to each x
14 %}
15 arctans = zeros(length(xs),1); % array of
16 arctan(x)
17 ers = zeros(length(xs), 1); % array of
18 relative errors
19 for k = 1 : length(xs)
20     x = xs(k);
21     delta = x - 1;
22     arctan = 0;
23     comp = atan(x);
24     for i = ns(k) : (-1) : 1
25         arctan = arctan +
26         nterm_taylor1_arctan(delta,i);
27     end
28     arctan = arctan + pi/4;
29     arctans(k) = arctan;
30     ers(k) = abs((arctan - comp)/comp);
31 end
32 end

```

Code 11. Taylorentwicklung um 1 umgekehrt.

#### 6.5. selective Taylor

```

1 function [arctans, ers] = fulltaylor_arctan(xs,
2     ns)
3 % selective taylorapproximation of arctan
4 arctans = zeros(length(xs),1);
5 ers = zeros(length(xs), 1);
6
7 for k = 1 : length(xs)
8     % selective x value
9     o = xs(k);
10    if abs(o) <= 1
11        x = abs(o);
12    else
13        x = abs(1/o);
14    end
15
16    % Taylorapproximation
17    arctan = 0;
18    comp = atan(o);
19    if(x<0.75)
20        for i = ns(k) : (-1) : 0
21            arctan = arctan +
22            nterm_taylor0_arctan(x,i);
23        end
24    else
25        delta = x - 1;
26        for i = ns(k) : (-1) : 1
27            arctan = arctan +
28            nterm_taylor1_arctan(delta,i);
29        end
30        arctan = arctan + pi/4;
31    end
32
33    % range adjustments
34    if abs(o) > 1
35        arctan = pi/2 - arctan;
36    end
37
38    % sign adjustments
39    if o < 0
40        arctan = arctan * (-1);
41    end
42
43    % return
44    arctans(k) = arctan;
45    ers(k) = abs((arctan - comp)/comp);
46 end
47 end

```

Code 12. selective Taylorentwicklung.



## 6.6. Lagrange Interpolation

```

1 function xs = chebishevnodes(n)
2     xs = zeros(n, 1);
3     for i = 0 : (n-1)
4         x = cos(i/(n-1)*pi);
5         xs(i+1) = x;
6     end
7 end

```

Code 13. Chebishev Knoten.

## 6.7. Lagrange Interpolation

```

1 function f = lagrangepolynom(xs)
2     syms x;
3     n = length(xs);
4     ys = zeros(n,1);
5     for i = 1 : n
6         ys(i) = atan(xs(i));
7     end
8     Ls = sym(zeros(1,n));
9     for i = 1 : n
10        Ls(i) = 1;
11        for j = [1:i-1, i+1:n]
12            Ls(i) = Ls(i) * (x-xs(j))/(xs(i)-xs(
13                j));
14        end
15        disp("Ls, ok")
16        p = Ls*ys;
17        p = matlabFunction(p);
18        f = p;
19 end

```

Code 14. Lagrangepolynom.

## 6.8. Lagrange Interpolation

```

1 function [arctans, ers] = lagrangeapprox(xs,
2     refs)
3     max = length(xs);
4     arctans = zeros(max,1);
5     ers = zeros(max,1);
6     p = lagrangepolynom(refs);
7
8     for i = 1 : max
9         x = xs(i);
10        arctan = p(x);
11        comp = atan(x);
12        er = abs(arctan - comp)/comp;
13
14        arctans(i) = arctan;
15        ers(i) = er;
16    end
end

```

Code 15. Lagrangeapproximation.

```

1 \RequirePackage[
2     backend=biber,
3     style=ieee,
4     sorting=ynt
5     ]{biblatex}

```

Code 16. References style.