R. Brigola, TH Nürnberg Georg Simon Ohm, 2014

Mathematica - Notebooks als Bonusmaterial zum Lehrbuch

[1] Rolf Brigola Fourier-Analysis und Distributionen,
Eine Einführung mit Anwendungen,
edition swk, Hamburg 2013

Teil 4 Demonstrationsbeispiele zu Tschebyscheff-Polynomen (Chebyshev-Polynomen) mit *Mathematica*

Referenzen zu Kapiteln, Abschnitten, Seitenzahlen beziehen sich im Folgenden meist auf das genannte Lehrbuch des Autors. Einige wenige weitere Referenzen findet man am Ende des Notebooks.

Die URL aller meiner *Mathematica*-Notebooks zur Fourier-Analysis ist:
http://www.stiftung-swk.de/mathematica/

In diesem Notebook werden einige Eigenschaften der Tschebyscheff-Polynome (oft auch in der Schreibweise Chebyshev-Polynome zu finden) mit Hilfe von *Mathematica* gezeigt. Approximation und Interpolation mit Hilfe der Tschebyscheff-Polynome und ihre enge Beziehung zur Diskreten Cosinus-Transformation (DCT I und DCT II) bilden das wesentliche Beispielmaterial. Wir beschränken uns auf die sogenannten *Tschebyscheff-Polynome erster Art*.

Anmerkung des Autors: Wie schon in vorangehenden Notebooks bemerkt, bin ich kein Experte für die schier unerschöpflichen Möglichkeiten, die ein wirklich geübter, vertiefter Umgang mit *Mathematica* bietet. Ich habe mich daher im Wesentlichen bemüht, den Stoff mit diesem Angebot zu veranschaulichen und zu zeigen, wie man mit (zum Teil wahrscheinlich oft zu umständlichen, dafür aber auch für *Mathematica*-Anfänger wie mich selbst transparenten) Anweisungen die behandelten Inhalte erschließen kann.

Tschebyscheff – Polynome (erster Art)

(vgl. [1], Kapitel 5, Abschnitt 5.7)

1. Definition, erste Eigenschaften

Das Tschebyscheff-Polynom T[n,x] erster Art vom Grad n ist - zunächst auf [-1,1] - für n⩾0 definiert durch

$$T[n,x] = \cos[\, n \arccos[x]].$$

Mit den Additionstheoremen für die Cosinusfunktion findet man schnell die Rekursionsgleichung (n=1,2,...)

$$T[n+1,x]=2x\,T[n,x] - T[n-1,x].$$

Dies zeigt, dass T[n,x] ein Polynom vom Grad n und als solches auf ganz $\mathbb{R}$ und $\mathbb{C}$ definiert ist.

Für gerade n sind die T[n,x] gerade, für ungerade n sind sie ungerade Funktionen und es gilt stets

$$|\,T[n,x]\,| \leqslant 1 \text{ auf } [-1,1].$$

Der Koeffizient a_n von $a_n x^n$ in T[n,x] ist 2^{n-1}. Weil cos[n x] in $[0,\pi]$ genau n Nullstellen hat, besitzt T[n,x] genau n *reelle Nullstellen in* [-1,1]. Diese Nullstellen x[k] sind

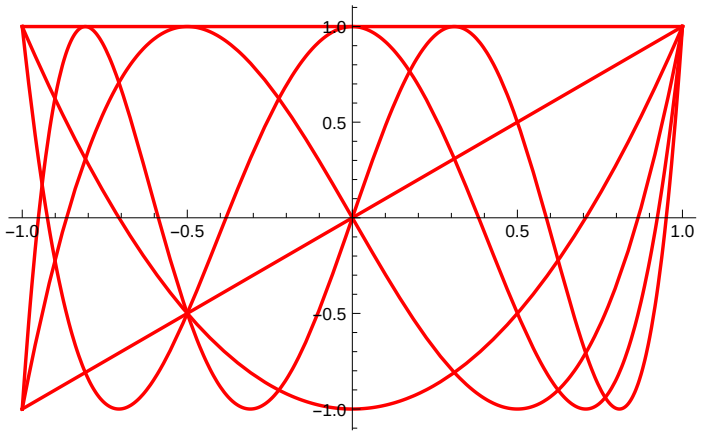
$$x[k] = \cos[\,(2k-1)\pi/(2\,n)] \quad \text{für } k=1,...,n.$$

Mathematica kennt die Tschebyscheff-Polynome in der Schreibweise ChebyshevT[n,x].
Wir plotten die ersten 5 solchen Polynome. Beachten Sie die Symmetrie-Eigenschaften und die Nullstellen.

In[200]:=

```
Plot[Evaluate[Table[ChebyshevT[n, x], {n, 0, 5}]], {x, -1, 1},
 PlotStyle → Directive[Red, Thickness[0.005`]], PlotRange → All]
```

Out[200]=



2. Die Tschebyscheff-Polynome erster Art als Orthogonalsystem

Die betrachteten Polynome bilden ein Orthogonalsystem bzgl. des inneren Produkts

$$< f, g >_w = \int_{-1}^{+1} f(x)\,g(x) \Big/ \left(\sqrt{1-x^2}\right) dx$$

mit der Gewichtsfunktion w[x]=1/$\left(\sqrt{1-x^2}\right)$, betrachtet für alle reellwertigen Funktion f und g auf [-1,1], für welche $< f, f >_w = ||\,f\,||^2_w$ und $< g, g >= ||\,g\,||^2_w$ existieren.

Wie üblich identifizieren wir Funktionen f und g, für die $||\,f-g\,||_w = 0$ ist, und bezeichnen den Vektorraum der so entstehenden Äquivalenzklassen, meist einfach wieder Funktionen f wie ein Repräsentant einer solchen Klasse genannt, als $H = L_w{}^2([-1,1])$. Dieser Raum ist mit dem definierten inneren Produkt ein Hilbertraum und die entsprechend normierten

Tschebyscheff-Polynome bilden ein vollständiges Orthonormalsystem in diesem Raum.

Zunächst die ersten 3 dieser Polynome in unnormierter Form:

In[201]:=
```
T0 = ChebyshevT[0, x]
T1 = ChebyshevT[1, x]
T2 = ChebyshevT[2, x]
```

Out[201]= 1

Out[202]= x

Out[203]= $-1 + 2\,x^2$

Nun mit den zugehörigen Normierungsfaktoren, so dass $||\,T[n,\,x]\,||_w = 1$ ist,
hier für n=0,1,2.

In[204]:=
```
T0n[x_] = ChebyshevT[0, x] / Sqrt[Pi]
Integrate[T0n[x]^2 / Sqrt[1 - x^2], {x, -1, 1}]
```

Out[204]= $\frac{1}{\sqrt{\pi}}$

Out[205]= 1

In[206]:=
```
T1n[x_] = Sqrt[2 / Pi] ChebyshevT[1, x]
Integrate[T1n[x]^2 / Sqrt[1 - x^2], {x, -1, 1}]
```

Out[206]= $\sqrt{\frac{2}{\pi}}\;x$

Out[207]= 1

In[208]:=
```
T2n[x_] = Sqrt[2 / Pi] ChebyshevT[2, x]
Integrate[T2n[x]^2 / Sqrt[1 - x^2], {x, -1, 1}]
```

Out[208]= $\sqrt{\frac{2}{\pi}}\;\left(-1 + 2\,x^2\right)$

Out[209]= 1

Aufgrund ihrer Definition entspricht eine Reihenentwicklung eines Elements f(x) aus H gerade der Fourierreihen-Entwicklung von f(cos(ϕ)), betrachtet als 2π-periodische gerade Funktion. Die zugehörige Reihe konvergiert in der Norm von H und für stetige, stückweise stetig differenzierbare f sogar gleichmäßig, wie wir aus der Theorie der Fourierreihen wissen (vgl. [1]). Die Koeffizienten der Entwicklung

(1) $$f(x) = \frac{a_0}{2}\,T[0,x] + \sum_{k=1}^{\infty} a_k\,T[k,\,x]$$

sind also gerade die Fourierkoeffizienten $a_0/2$ und a_k ($k = 1,\,2,\,\ldots$) von f(cos(ϕ)).

Beispiel: Approximation der Signum-Funktion

Wir betrachten als Beispiel einige Approximationen der Signum-Funktion auf [-1,1] mittels Tschebyscheff-Polynomen durch Partialsummen der obigen Reihenentwicklung und beobachten

wie bei Fourierreihen an der Sprungstelle typischerweise auch hier das ***Gibbsphänomen***.
Wir betrachten die "Zielfunktion" Signum und die Partialsummen bis T[5,x], T[9,x] und T[19,x].
Es treten nur ungerade Polynompotenzen auf, weil die Signumfunktion ja ungerade ist.

Zunächst die Partialsummen der Reihenentwicklung bis zu den Tschebyscheff-Polynome T[n,x] für n=5,9 und 19, dann die genannten polynomialen Approximationen und ihre grafische Darstellung.
Wir berechnen die nötigen Koeffizienten einmal mit dem inneren Produkt $< ., . >_w$ von oben und zur Demonstration des vorher Gesagten als Fourierkoeffizienten von Sign[Cos[ϕ]]. Auf beide Art findet man die gleichen Koeffizienten.

In[210]:=
```
f[x_] = Sign[x]
a0halbe = Integrate[f[x] ChebyshevT[0, x] / Sqrt[1 - x^2], {x, -1, 1}] / Pi
(* natürlich Null, weil f ungerade ist *)
```

Out[210]= Sign[x]

Out[211]= 0

In[212]:=
```
FKa0halbe = 1 / Pi Integrate[f[Cos[phi]], {phi, 0, 2 Pi}]
```

Out[212]= 0

In[213]:=
```
ak[k_] := 2 / Pi Integrate[f[x] ChebyshevT[k, x] / Sqrt[1 - x^2], {x, -1, 1}]
FKak[k_] := 2 / Pi Integrate[f[Cos[phi]] Cos[k phi], {phi, 0, Pi}]
```

Hier ein kurzer Test, dass die gleichen Koeffizienten damit berechnet werden.

In[215]:=
```
ak[1]        (* Koeffizient zu T[1,x] mittels innerem Produkt in (1) oben  *)
FKak[1]      (* Koeffizient als Fourierkoeffizient von f(cos(phi)) *)
```

Out[215]= $\frac{4}{\pi}$

Out[216]= $\frac{4}{\pi}$

In[217]:=
```
ak[3]        (* Koeff. zu T[3,x]  *)
FKak[3]
```

Out[217]= $-\frac{4}{3\,\pi}$

Out[218]= $-\frac{4}{3\,\pi}$

Nun die genannten Approximationen mit 5, 9 und 19 Tschebyscheff - Polynomen, die Koeffizienten jeweils unterschiedlich berechnet, im letzten Fall durch *numerische* Integration.
Danach ein grafische Darstellung der Funktion f=Sign und ihrer berechneten Näherungen.
Man erkennt deutlich das Gibbsphänomen. Welche Näherung zu welchem Polynomgrad gehört, ist leicht zu sehen: Je näher der erste “Buckel” der Näherung in [0,1] der Sprungstelle ist, desto höher der Polynomgrad.

In[219]:=
```
fct2[t_] = Sum[FKak[k] ChebyshevT[k, t], {k, 1, 5, 2}]
  (* Näherung mit Polynomgrad 5 *)
```

Out[219]=
$$\frac{4\,t}{\pi} - \frac{4\left(-3\,t+4\,t^3\right)}{3\,\pi} + \frac{4\left(5\,t-20\,t^3+16\,t^5\right)}{5\,\pi}$$

In[220]:=
```
Simplify[N[%]] (* hier numerisch *)
```

Out[220]=
$$3.81972\,t - 6.79061\,t^3 + 4.07437\,t^5$$

In[221]:=
```
fct3[t_] = Sum[ak[k] ChebyshevT[k, t], {k, 1, 9, 2}]
  (* Näherung mit Polynomgrad 9 *)
```

Out[221]=
$$\frac{4\,t}{\pi} - \frac{4\left(-3\,t+4\,t^3\right)}{3\,\pi} + \frac{4\left(5\,t-20\,t^3+16\,t^5\right)}{5\,\pi} - \frac{4\left(-7\,t+56\,t^3-112\,t^5+64\,t^7\right)}{7\,\pi} + \frac{1}{9\,\pi}4\left(9\,t-120\,t^3+432\,t^5-576\,t^7+256\,t^9\right)$$

In[222]:=
```
Simplify[N[%]] (* Man beachte: Die Koeff. der Tschebyscheff-
   Polynome bis zum Grad 5 sind die gleichen wie oben,
aber die Koeff. zu den Monomen t^k in der Entwicklung bis zum
 Grad 9 nun durchaus bei t, t^2, .. ,t^5 anders als oben *)
```

Out[222]=
$$6.3662\,t - 33.9531\,t^3 + 85.5617\,t^5 - 93.1284\,t^7 + 36.2166\,t^9$$

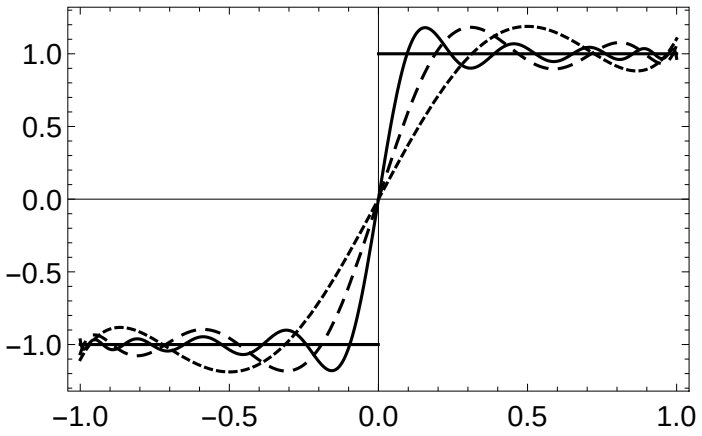
In[223]:=
```
fct4[t_] = Simplify[N[2/Pi]
   Sum[NIntegrate[f[Cos[x]] Cos[n x], {x, 0, Pi}] ChebyshevT[n, t], {n, 1, 19, 2}]]
  (* Näherung mit Polynomgrad 19, numerische Ausgabe *)
```

Out[223]=
$$12.7324\,t - 280.113\,t^3 + 3226.9\,t^5 - 19\,976.\,t^7 + 72\,505.6\,t^9 - 161\,789.\,t^{11} + 224\,654.\,t^{13} - 189\,138.\,t^{15} + 88\,351.4\,t^{17} - 17\,567.\,t^{19}$$

Nun der Plot der Approximationen

In[224]:=
```
g1 := Plot[f[t], {t, -1, 1}, Frame → True,
   FrameStyle → Directive[Black, FontSize → 15,
     FontWeight → Plain],
   PlotStyle → {GrayLevel[0.0], Thickness[0.005]}]
g2 := Plot[fct2[t], {t, -1, 1}, Frame → True,
   FrameStyle → Directive[Black, FontSize → 15, FontWeight → Plain],
   PlotStyle → {GrayLevel[0.0], Dashing[0.01], Thickness[0.005]}]
g3 := Plot[fct3[t], {t, -1, 1}, Frame → True,
   FrameStyle → Directive[Black, FontSize → 15, FontWeight → Plain],
   PlotStyle → {GrayLevel[0.0], Dashing[0.02], Thickness[0.005]}]
g4 := Plot[fct4[t], {t, -1, 1}, Frame → True,
   FrameStyle → Directive[Black, FontSize → 15, FontWeight → Plain],
   PlotStyle → {GrayLevel[0.0], Thickness[0.005]}]
Show[g1, g2, g3, g4, PlotRange → All ]
(* Export["ChebApp.eps",%,"EmbeddedFonts"→False] ,
falls Sie das Bild als eps-Grafik z.B. nach Latex exportieren wollen *)
```

Out[228]=



Vergleich der Approximation mit Legendre-Polynomen und Tschebyscheff-Polynomen am Beispiel einer Sinus-Funktion

Im ersten Notebook über Fourierreihen auf dieser Website haben wir schon das Beispiel einer Approximation der Sinus-Funktion mittels Taylorpolynomen und mittels Legendre-Polynomen angesehen. Hier vergleichen wir noch im gleichen Beispiel die Näherung mit Legendre- und Tschebyscheff-Polynomen.

Zuerst noch einmal die Approximation von Sin[3x] im quadratischen Mittel mit den Legendre-Polynomen bis zum Grad 5 wie schon gehabt (http://www.stiftung-swk.de/mathematica/Fourierreihen-Teil1.nb)

In[229]:=
```
Faktor[n_] := NIntegrate[LegendreP[n, x] Sin[3 x], {x, -1, 1}]
n2[x_] = Expand[Sum[Faktor[n] LegendreP[n, x] / (2 / (2 n + 1)), {n, 0, 5}]]
```

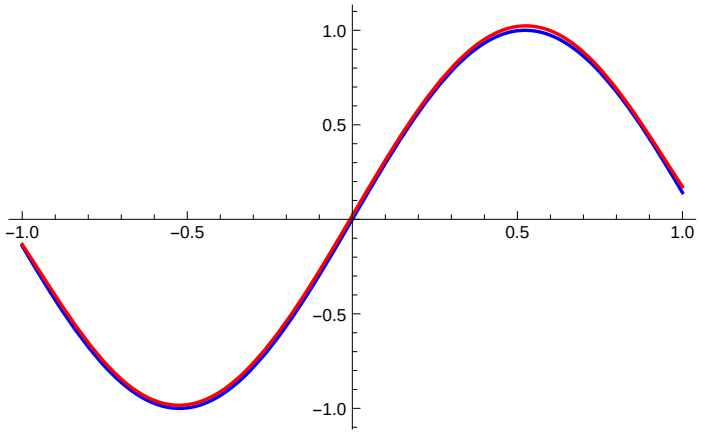
Out[230]=
$$0. + 2.97177\,x - 4.23916\,x^3 + 1.42043\,x^5$$

Hier das so erhaltene Näherungspolynom vom Grad 5 wie oben zur Ansicht rot

mit einem Offset von 0.02 gezeichnet, damit man die Näherung überhaupt von der Zielfunktion Sin[3 x] unterscheiden kann.

In[231]:=
```
p1 := Plot[Sin[3 x], {x, -1, 1},
  PlotStyle → Directive[Blue, Thickness[0.005]], PlotRange → All]
p2 := Plot[n2[x] + 0.02, {x, -1, 1},
  PlotStyle → Directive[Red, Thickness[0.005]], PlotRange → All]
Show[p1, p2]
```

Out[233]=



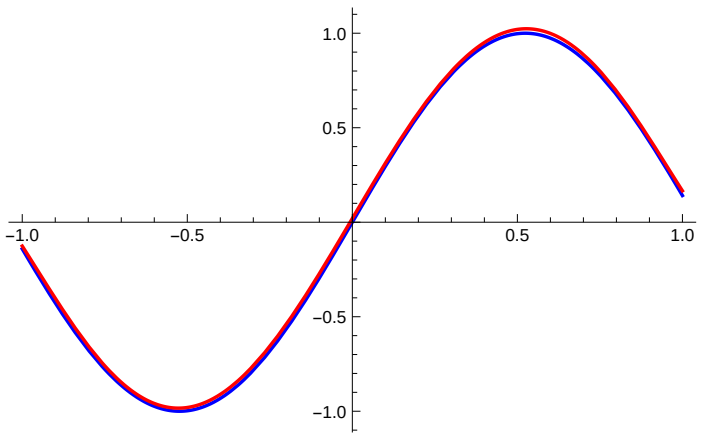
Nun die Approximation mit Tschebyscheff - Polynomen an Stelle der Legendre - Polynome und der Plot, wieder das Näherungspolynom mit Offset +0.02. Das Auge kann die beiden Näherungen kaum unterscheiden. Der Unterschied wird aber deutlich, wenn man die Fehler der beiden Näherungen vergleicht.

In[234]:=
```
f[x_] := Sin[3 x]
n3[x_] = Simplify[N[2 / Pi Sum[
      Integrate[f[Cos[phi]] Cos[n phi], {phi, 0, Pi}] ChebyshevT[n, x], {n, 1, 5}]]]
```

Out[235]=
$$2.96278\,x - 4.19364\,x^3 + 1.37691\,x^5$$

In[236]:=
```
p3 := Plot[n3[x] + 0.02, {x, -1, 1},
  PlotStyle → Directive[Red, Thickness[0.005]], PlotRange → All]
Show[p1, p3]
```

Out[237]=

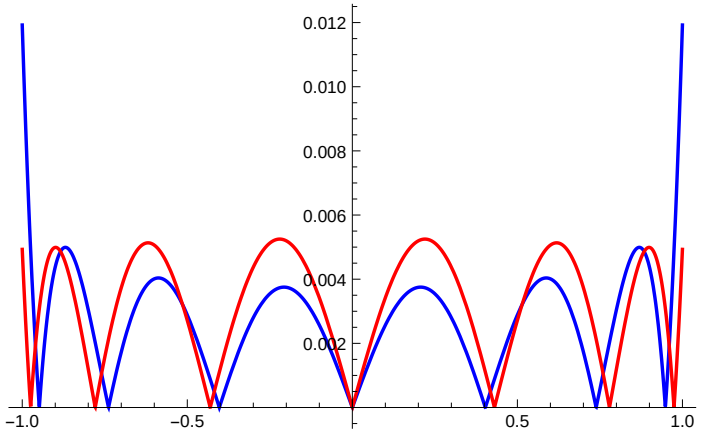


In[238]:=
```
p4 := Plot[Abs[Sin[3 x] - n2[x]], {x, -1, 1},
  PlotStyle → Directive[Blue, Thickness[0.005]], PlotRange → All]
p5 := Plot[Abs[Sin[3 x] - n3[x]], {x, -1, 1},
  PlotStyle → Directive[Red, Thickness[0.005]], PlotRange → All]
```

Nun die Fehlerkurven, die Approximation mit Legendre-Polynomen in blau, die mit Tschebyscheff-Polynomen in rot. Die letztere zeigt etwas größere Fehler im Inneren von [-1,1], dagegen eine weit bessere Näherung am Rand des Intervalls. Dies ist ein Effekt der verwendeten Gewichtsfunktion, welche große Fehler am Rand mit größerer Norm des Fehlers bestraft, nach der ja bei der Orthogonalprojektion in $L_w^2([-1,1])$, die dem Näherungspolynom zugrunde liegt, minimiert wird..

In[240]:=
```
Show[p4, p5]
```

Out[240]=



Zum Abschluss dieses ersten Abschnitts sei noch einmal darauf hingewiesen, dass *auch bei der Verwendung anderer Orthogonalsysteme* als den trigonometrischen Funktionen oder den Systemen der Legendre- oder Tschebyscheff-Polynome bei Funktionen mit Sprungstellen vielfach wieder das *Gibbssche Phänomen* auftritt. Zur Bekämpfung kann man auch hier Faltungen mit geeigneten Summationskernen verwenden, wie wir es mit den Fejér-Kernen vorher schon bei Fourierreihen kennen gelernt haben.

3. Polynomiale Interpolation mit Tschebyscheff-Polynomen, Zusammenhang mit der DCT I und der DCT II, Beispiel von C. Runge, Aliaseffekte

Bei der Verarbeitung von Signalen in der Technik ist es häufig eine wichtige Aufgabe, aus Abtastwerten eines kontinuierlichen Signals eine trigonometrische oder polynomiale Interpolation zu berechnen. Mit den Koeffizienten der entsprechenden Interpolationsfunktion hat man dann eine Näherungsfunktion durch ein paar wenige Zahlen beschrieben und kann diese in Rechnern “verarbeiten” (Rechner können ja schließlich nur mit Zahlen arbeiten und

nicht etwa mit kontinuierlichen Signalen.)

In der Fourier-Analysis erhält man trigonometrische Interpolationsformeln mit Hilfe der DFT, DCT I oder auch DCT II, vgl. [1], Abschnitt 5.7, S. 75 und 77. Beispiele dazu wurden schon im Notebook http://stiftung-swk.de/mathematica/Fourierreihen-Teil3.nb meines Materials auf dieser Website vorgestellt.

Wegen der engen Beziehung der Tschebyscheff-Approximation zu Fourierreihen-Entwicklungen von Funktionen ergeben sich schnell auch Möglichkeiten, zu gegebenen Funktionen *polynomiale* Interpolationen mit Hilfe einer diskreten Cosinustransformation DCT I oder DCT II zu finden. *Dabei ersetzt man die Fourierkoeffizienten in (1) oben durch ihre mit einer DCT aus Abtastwerten berechneten Näherungswerte*. Schon aus den Überlegungen zur DFT ist klar, dass auch hier wieder die sog. *Alias-Effekte* zu beachten sind. Wir gehen weiter unten noch darauf ein.

Wir betrachten nun im Folgenden stetige, stückweise stetig differenzierbare reellwertige Funktionen f auf [-1,1]. Durch Umskalierung lassen sich die Ergebnisse natürlich auch für Funktionen auf anderen Intervallen verwenden (Übung für Sie als Eigenarbeit).

Man erhält folgende Interpolationsformeln (vgl. [1], Abschnitt 5.7, S. 84):

Mit Hilfe einer ***DCT I****:*

Mit den m+1 (m ∈ ℕ) Interpolationsknoten der Form $\boldsymbol{x_n = \cos(n\pi/m)}$, n=0,...,m *in* [-1,1] *ist*
(2) $P_{2,T}(x) = C_0 + 2 \sum_{k=1}^{m-1} C_k \, T[k,x] + C_m T[m,x]$
ein reellwertiges Interpolationspolynom. Dabei erhält man die Koeffizienten C_k, *k=0,..,m aus einer* DCT I *der Abtastwerte* $f(x_n)$, n=0,...,m.

Mit Hilfe einer ***DCT II*** *und den sogenannten* ***Tschebyscheffabszissen*** $\boldsymbol{x_n}$*:*

Mit den m+1 (m ∈ ℕ) Interpolationsknoten der Form $\boldsymbol{x_n = \cos((2n+1)\pi/(2m+2))}$, n=0,...,m *in* [-1,1], *d.h. mit den Nullstellen des Tschebyscheff-Polynoms* T[m+1,x], *ist*
(3) $P_{3,T}(x) = A_0 + \sum_{k=1}^{m} A_k \, T[k, x]$
ein reellwertiges Interpolationspolynom. Dabei erhält man die Koeffizienten A_k, *k=0,..,m aus einer* DCT II *der Abtastwerte* $f(x_n)$, n=0,...,m.

Wir betrachten zur Demonstration das berühmte Beispiel der Funktion $f(x) = 1/(1 + 25\,x^2)$ auf [-1,1] von C. Runge (1901). Für dieses Beispiel ergeben polynomiale Interpolationen mit äquidistanten Knoten eine mit wachsender Kontenzahl immer schlechtere Approximation von f, während die Interpolationen mit Tschebyscheffknoten wie oben eine mit wachsender Knotenzahl sogar gleichmäßig gegen f konvergente Folge von Interpolationspolynomen hervorbringt.

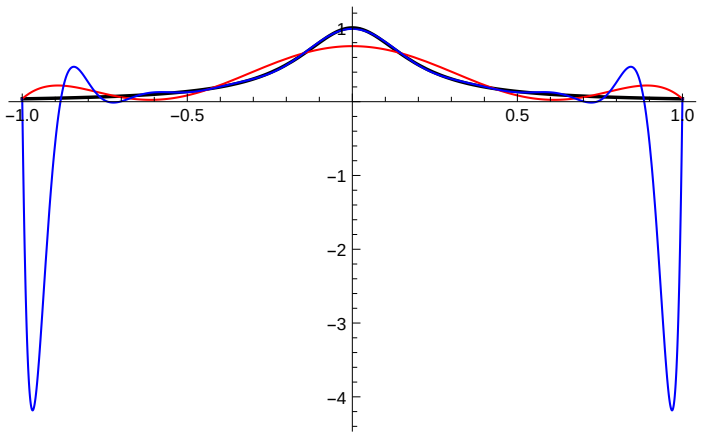
Zunächst das Beispiel mit 8 und 17 äquidistanten Knoten zur Veranschaulichung, dass dies eine schlechte Wahl zur polynomialen Näherung von f ist. Testen Sie selbst mit noch mehr Knoten.

In[241]:=
```
f[x_] := 1 / (1 + 25 x^2)  (* Das Runge-Beispiel *)
a := -1; b := 1;           (* Intervallgrenzen *)
X[m_, n_] := a + (b - a) m / n; Y[m_, n_] := f[X[m, n]];
(* n+1 Knoten X und Abtastwerte Y von f *)
Lagr[n_, k_, x_] := (∏_{j=0}^{k-1} (x - X[j, n]) / (X[k, n] - X[j, n])) (∏_{j=k+1}^{n} (x - X[j, n]) / (X[k, n] - X[j, n]));
InterpolyLagrange[n_, x_] := ∑_{k=0}^{n} Y[k, n] Lagr[n, k, x];
(* Das Lagrange-Interpolationspolynom *)
plot1 := Plot[f[x], {x, a, b}, PlotStyle → Directive[
    Black, Thickness[0.005]], PlotRange → All]
plot2 := Plot[InterpolyLagrange[7, x], {x, a, b}, PlotStyle → Directive[
    Red, Thickness[0.003]], PlotRange → All]
plot3 := Plot[InterpolyLagrange[17, x], {x, a, b}, PlotStyle → Directive[
    Blue, Thickness[0.003]], PlotRange → All]
```

Nun die Plots der Runge - Funktion (schwarz) und der Lagrange - Interpolationspolynome vom Grad 7 (rot) und Grad 17 (blau) :

In[249]:=
```
Show[plot1, plot2, plot3]
```

Out[249]=



Nun stattdessen Interpolation mit Tschebyscheff-Polynomen.

Zuerst mit Hilfe einer DCT I. Wir verwenden die *Mathematica*-Version der DCT I.

Dabei muss beachtet werden, dass bei der *Mathematica*-DCT im Vergleich zur Darstellung in [1] der Skalierungsfaktor angepasst werden muss, hier als Vorfaktor $1/\sqrt{2\,m}$, wenn wir m+1 Abtastwerte von f in [-1,1], oder anders gesagt m+1 Abtastwerte von f(cos(ϕ)) für $\phi \in [0,\pi]$ nehmen. In [1] wurde der Vorfaktor dagegen so gewählt, dass die DCT-Koeffizienten wie oft gewünscht den Amplituden der an der Näherung beteiligten Schwingungen entsprechen. (*Tipp*: Wenn Sie einmal fertige Programmroutinen zur DFT (FFT), DCT verwenden, dann testen Sie vielleicht die Normierungen, indem Sie einfach eine Cosinusfunktion mit den Routinen behandeln. Das kann auf einfache Weise unliebsame Überraschungen verhindern.)

Wir wählen also für das Beispiel 17 äquidistante Knoten in [0,π], beginnend bei Null, und die zugehörigen Abtastwerte von f(cos(ϕ)):

In[250]:=
```
m := 16;  (* weil die Zählung mit Null beginnt *)
list1 := Table[f[Cos[ n π / m]], {n, 0, m}]
```

Nun die DCT I mit diesen Werten und dem richtigen Vorfaktor.
Jeder Koeffizient mit gerader Nummer ist Null, d.h. passend zur geraden Symmetrie von f werden keine ungeraden Tschebyscheff-Polynome in der Interpolation verwendet.

In[252]:=
```
coeff = 1 / Sqrt[2 m] FourierDCT[list1, 1]
```

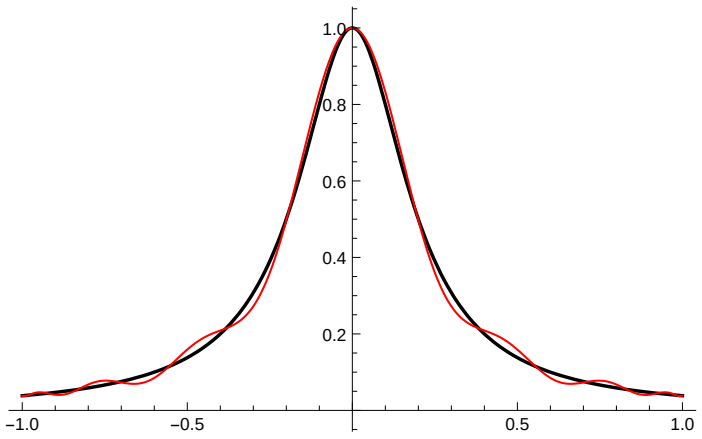
Out[252]=
```
{0.196797, 0., -0.132541, 0., 0.0894909, 0., -0.0607598, 0.,
 0.0417501, 0., -0.0294205, 0., 0.0217982, 0., -0.0176636, 0., 0.0163552}
```

Jetzt das Interpolationspolynom und seine grafische Darstellung (rot) zusammen mit f (schwarz)

In[253]:=
```
InterpolyTscheb1[x_] = Simplify[
  coeff[[1]] + 2 Sum[coeff[[k]] ChebyshevT[k - 1, x] + coeff[[m + 1]] ChebyshevT[m, x]]
plot4 := Plot[InterpolyTscheb1[x], {x, a, b}, PlotStyle → Directive[
    Red, Thickness[0.003]], PlotRange → All]
Show[plot1, plot4]
```

Out[253]=
$$1. - 18.4579\,x^2 + 180.138\,x^4 - 931.478\,x^6 + 2718.63\,x^8 - 4638.33\,x^{10} + 4585.72\,x^{12} - 2433.11\,x^{14} + 535.928\,x^{16}$$

Out[255]=



Nun das gleiche Spiel mit einer DCT II

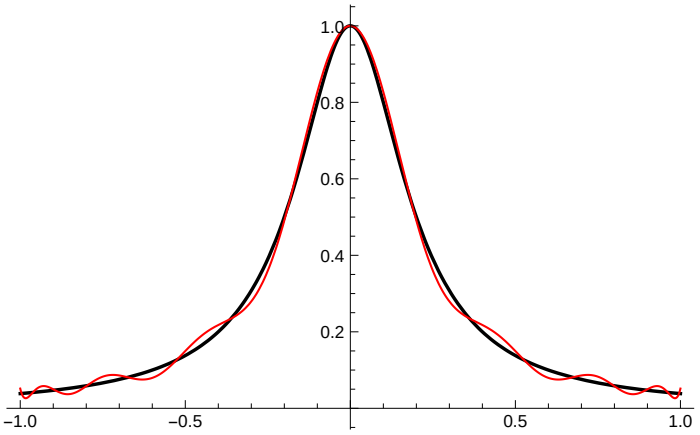
Nachfolgend eine DCT II der Abtastwerte mit den sogenannten Tschebyscheff-Abszissen zur Berechnung der Koeffizienten des Interpolationspolynoms. Wieder müssen wir beachten, den richtigen Normierungsfaktor bei der DCT II von *Mathematica* zu wählen, hier also den Faktor $1/\sqrt{m+1}$ (man vgl. wieder [1], S. 77). Wir berechnen also das Interpolationspolynom, wieder mit m+1=17 Knoten, und plotten das Ergebnis zusammen mit der Funktion von C. Runge:

In[256]:=
```
list2 := Table[N[f[Cos[(2 n + 1) π / (2 m + 2)]]], {n, 0, m}]
(* Abtastwerte an den Tschebyscheff-Abszissen *)
coeff2 = N[1 / Sqrt[m + 1]] Chop[FourierDCT[list2, 2] ]
(* DCT II dieser Wertefolge *)
InterpolyTscheb2[x_] = Simplify[coeff2[[1]] + 2 Sum[coeff2[[k]] ChebyshevT[k - 1, x], {k, 2, m + 1}]]
(* resultierendes Polynom *)
plot5 := Plot[InterpolyTscheb2[x], {x, a, b}, PlotStyle → Directive[
    Red, Thickness[0.003]], PlotRange → All]
Show[plot1, plot5]
```

Out[257]=
```
{0.196573, 0., -0.132299, 0., 0.0891931, 0., -0.0603577, 0.,
 0.0411796, 0., -0.0285902, 0., 0.0205753, 0., -0.0158524, 0., 0.0136658}
```

Out[258]=
$$1. - 19.192\, x^2 + 201.018\, x^4 - 1122.49\, x^6 + 3529.36\, x^8 - 6457.85\, x^{10} + 6814.73\, x^{12} - 3842.14\, x^{14} + 895.603\, x^{16}$$

Out[260]=



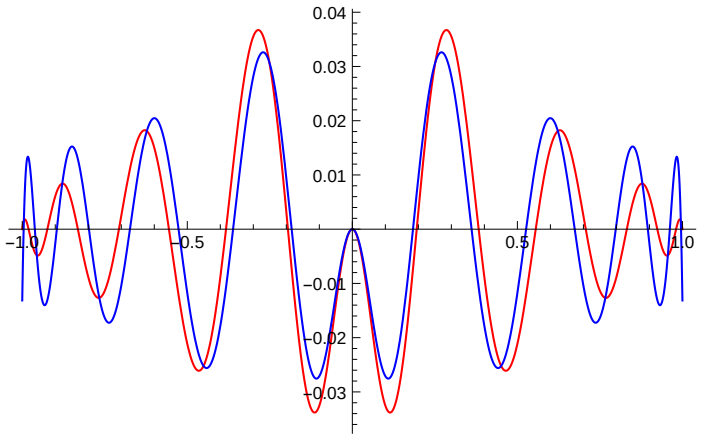
Wie zu erwarten sind die beiden Versionen der Interpolationspolynome mit dem bloßen Auge schwer zu unterscheiden. Wir betrachten daher die zugehörigen Fehlerfunktionen, um einen besseren Eindruck zu erhalten. Der Fehler der Interpolation mittels DCT I in rot, der mittels DCT II in blau.
Mit wachsenden Knotenzahlen liefern beide Interpolationen auch gute Approximationen über das gesamte Intervall [-1,1] betrachtet, die schließlich für m→∞ sogar gleichmäßig gegen die Zielfunktion konvergieren (vgl. [1], Satz auf S. 85).

Fehler der beiden Interpolationen

In[261]:=
```
plot6 := Plot[f[x] - InterpolyTscheb1[x], {x, a, b}, PlotStyle → Directive[
    Red, Thickness[0.003]], PlotRange → All]
```

In[262]:=
```
plot7 := Plot[f[x] - InterpolyTscheb2[x], {x, a, b}, PlotStyle → Directive[
    Blue, Thickness[0.003]], PlotRange → All]
Show[plot6, plot7]
```

Out[263]=



Nun zum Alias - Effekt

Wegen des geschilderten engen Zusammenhangs der beiden besprochenen Interpolationen mittels Tschebyscheff-Polynomen mit der DCT ist klar, dass auch hier bei der Koeffizientenberechnung Alias-Effekte wie bei der DFT, DCT eine Rolle spielen. Ganz analog dazu, dass bei einer DFT mit N Abtastwerten über einen Zeitraum [0,T[die Werte aller Schwingungen $\exp(i\,(k+mN)2\pi t/T)$ an den Abtaststellen zusammenfallen ($k,m \in \mathbb{N}_0$) und so den Alias-Effekt erzeugen, kann man auch die Werte gewisser Tschebyscheff-Polynome an den verwendeten Interpolationsknoten nicht unterscheiden. Man kann den Approximationsfehler der resultierenden Interpolationspolynome aus dieser Sicht als eine Konsequenz der so entstehenden Alias-Effekte bei den Polynomkoeffizienten lesen und daraus auch Fehlerabschätzungen gewinnen (vgl. etwa [3]).

Zur Verdeutlichung betrachten wir den letzten Fall einer Interpolation mit den Tschebyscheff-Abszissen als Knoten $x_n = \cos((\pi(2n+1)/(2m+2)),\ n = 0,\ ...,\ m$.
Aus $T[k, \mathrm{Cos}[x_n]] = \mathrm{Cos}[k\,x_n]$ folgt mit etwas Rechenarbeit aus den Additionstheoremen (eine empfehlenswerte Rechenübung für Sie), dass die Polynomwerte von T[k,x] und $(-1)^l\,T[l(2m+2)\pm k,x]$ für $l \in \mathbb{N}$ an allen Knoten x_n übereinstimmen. Für eine stetige Funktion f auf [-1,1] und ihr Interpolationspolynom mit m+1 Tschebyscheff-Abszissen als Knoten erhalten wir:

Für die Koeffizienten A_k in Gleichung (3) oben gilt die Aliasbeziehung:

Für $l \in \mathbb{N}$ und $k \in \mathbb{N}_0$ gilt mit $C_0 = 1/2$ und $C_k = 1$ für $k \neq 0$ gilt mit den Koeffizienten der a_k der Reihendarstellung (1) für f mit den m+1 Tschebyscheff-Abszissen als Knoten

(4) $$A_k = C_k\left(a_k + \sum_{l=1}^{\infty}(-1)^l\left(a_{l(2m+2)+k} + a_{l(2m+2)-k}\right)\right)$$

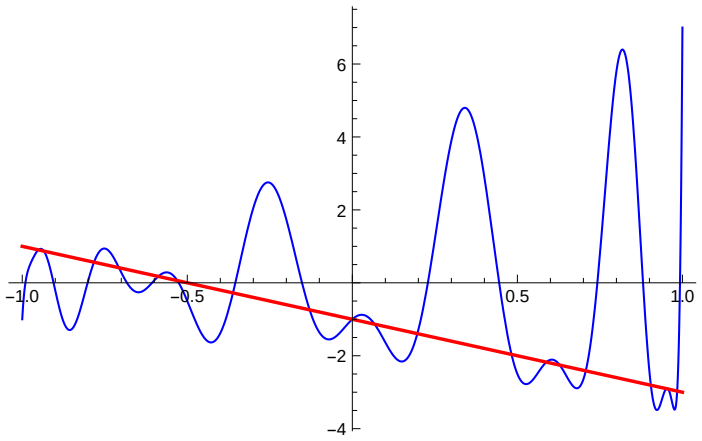
Wir demonstrieren den Effekt an einem einfachen Beispiel:

Wir interpolieren f(x) = T[9,x]+2 T[10,x] + 2 T[11,x] +T[20,x] +T[21,x] mit 5 Tschebyscheff-Abszissen als Knoten in [-1,1] wie besprochen. Die Koeffizienten $a_{10} = a_{11} = 2$ und $a_{20} = a_{21} = 1$ ergeben wegen des Alias-Effekts mit m=4 das Interpolationspolynom $P_{3,T}$[x]=-T[0,x]-2T[1,x]=-1-2x, denn $A_0 = \frac{1}{2}(-2\,a_{10} + 2\,a_{20}) = -1$ und $A_1 = -a_{11} - a_9 + a_{21} = -2$. Wir lassen $P_{3,T}$ wie oben berechnen und plotten das Polynom (rot) zusammen mit f (blau). Natürlich ist von vorneherein klar, dass die Anzahl der Knoten und damit der Grad des zugehörigen Interpolationspolynoms bei weitem nicht ausreicht, um mit der Näherung f geeignet zu approximieren. Demonstriert wird nur, was in den Koeffizienten A_k seinen Niederschlag findet.

In[264]:=
```
f[x_] := ChebyshevT[9, x] + 2 ChebyshevT[10, x] +
   2 ChebyshevT[11, x] + ChebyshevT[20, x] + ChebyshevT[21, x]
m := 4
list4 := Table[N[f[Cos[(2 n + 1) π / (2 m + 2)]]], {n, 0, m}]
(* Abtastwerte an den Tschebyscheff-Abszissen *)
coeff3 := N[1 / Sqrt[m + 1]] Chop[FourierDCT[list4, 2] ]
(* DCT II dieser Wertefolge *)
InterpolyTscheb3[x_] = Simplify[coeff3[[1]] + 2 Sum[coeff3[[k]] ChebyshevT[k - 1, x], {k, 2, m + 1}]]
(* resultierendes Polynom *)
plot6 := Plot[f[x], {x, a, b}, PlotStyle → Directive[
    Blue, Thickness[0.003]], PlotRange → All]
plot7 := Plot[InterpolyTscheb3[x], {x, a, b}, PlotStyle → Directive[
    Red, Thickness[0.005]], PlotRange → All]
Show[plot6, plot7] (*  Funktion f blau,Interpolationspolynom rot ;
der Koeffizient bei x^3 kommt durch numerische Fehler bei der DCT-
 Berechnung zustande *)
```

Out[268]= $-1. - 2.\,x - 2.52051 \times 10^{-9}\,x^3$

Out[271]=



An solche Effekte muss man denken, wenn man etwa in nichtlinearen Gleichungen mit Abtastwerten und Interpolationspolynomen arbeitet und damit vielleicht Terme wie eine Funktion f^3 durch ein Näherungspolynom approximieren will. *Zu achten ist wie bei der DFT, DCT also auf eine genügend hohe Anzahl von Abtastwerten, wenn das*

Interpolationspolynom die Funktion f gut nachbilden soll.

Übung: Leiten Sie eine entsprechende Aliasbeziehung für die Interpolation mit den Knoten $x_n = \cos(n\pi/m)$, $n = 0, \ldots, m$ in $[-1, 1]$ her und testen Sie Ihre Formel.

Fehlerabschätzungen finden Sie bei Interesse in den unten angegebenen Quellen [3] und [4]

Zum Abschluss dieses Notebooks :

Eine Extremaleigenschaft der Tschebscheff - Polynome, welche sie für die Schaltungsentwicklung in der Elektrotechnik interessant macht.

Es gilt folgender Satz (Beweis in [1], S. 87)

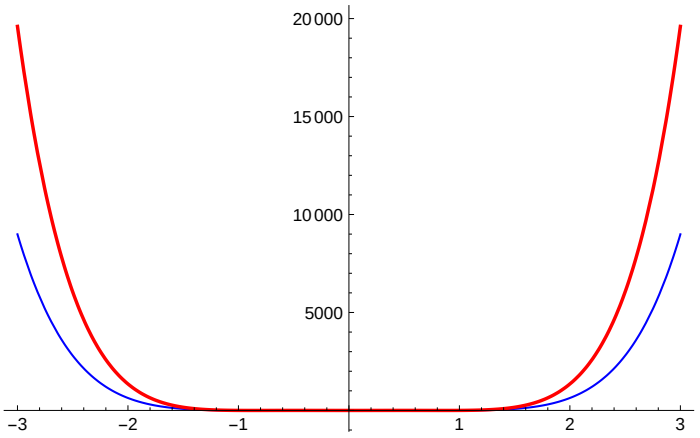
1. *Für jedes x_0 außerhalb von [-1,1] hat das Polynom $T[n, x]/T[n, x_0]$ unter allen Polynomen P vom Grad n mit $P(x_0) = 1$ minimale Supremumsnorm.*
2. *Unter allen Polynomen P vom Grad n mit $|P(x)| \leq 1$ auf [-1,1] wächst T[n,x] außerhalb von [-1,1] am schnellsten, d.h. dort ist $|T[n,x]| \geq |P(x)|$.*

Durch diese Eigenschaften eignen sich die Tschebyscheff-Polynome hervorragend für den Entwurf von Tiefpassfiltern mit rationalen Frequenzgängen, da sie schnell eine hohe Dämpfung außerhalb des sog. Passbands liefern. Ich gehe darauf in noch folgenden Notebooks auf dieser Website näher ein und zeige dort, wie man ein Tschebyscheff-Tiefpassfilter für einen vorgegebenen Dämpfungsplan entwirft und es als analoges oder diskretes Filter realisieren kann (durch Hardware im analogen Fall, durch Software im diskreten Fall). Der Entwurf von Tiefpassfiltern ist eine zentrale Aufgabe der Schaltungsentwicklung in der Nachrichtentechnik, weil man davon abgeleitet schnell auch Hochpass- oder Bandpass-Filter berechnen und realisieren kann. Die digitale Signalverarbeitung in vielen Übertragungsverfahren moderner Technik ist ohne den Einsatz solcher Filter nicht vorstellbar. Dazu jedoch etwas mehr in einem Nachfolge-Notebook.

Wir betrachten als Schlussbeispiel im Vergleich den Verlauf eines Legendre-Polynoms und eines Tschebyscheff-Polynoms (erster Art) gleichen Grades etwa in [-3,3] und sehen das unterschiedliche Wachstum im Komplement von [-1,1]. Leser können mit anderen Polynomen das gleiche Spiel betrachten, um vielleicht noch mehr Gefühl für den Sachverhalt zu entwickeln.

In[272]:=
```
plot8 := Plot[LegendreP[6, x], {x, -3, 3}, PlotStyle → Directive[
    Blue, Thickness[0.003]], PlotRange → All]
plot9 := Plot[ChebyshevT[6, x], {x, -3, 3}, PlotStyle → Directive[
    Red, Thickness[0.005]], PlotRange → All]
Show[plot8, plot9] (*  Legendre-Polynom blau,Tschebyscheff-Polynom rot *)
```

Out[274]=



Weitere empfehlenswerte Referenzen:

[2]	C. Runge	Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten, Zeitschrift für Mathematik und Physik 46 (1901), 224-243
[3]	M. Hanke-Bourgeois	Grundlagen der Numerischen Mathematik und des Wissenschaftlichen Rechnens, Teubner, 2008
[4]	J.C. Mason, D. Handscomb	Chebyshev Polynomials, CRC Press, 2002
[5]	C.W. Clenshaw, A.R. Curtis	A method for numerical integration on an automatic computer Numerische Mathematik 2 (1960), 197-205

u.v.a.m., die Sie im Literaturverzeichnis am Ende von [1] finden können.