Staatsexamensarbeit:

Fraktionale Differentialgleichungen mit MAPLE

Dietmar Blume

Übersicht

Inhaltsverzeichnis

1. Einführung

1.1 Vorbemerkungen

In der nachfolgenden Arbeit werde ich im zweiten Kapitel theoretisch fraktionale Ableitungen vorstellen, um dann im dritten Kapitel praktisch mit MAPLE fraktionale Ableitungen zu veranschaulichen. Genauso werde ich auch das Gebiet der fraktionalen Differentialgleichungen einführen, d.h. zuerst wird ein theoretischer Teil über Lösungsmethoden behandelt und darauf folgend ein praktischer Teil, in dem mittels MAPLE diverse Gleichungen gelöst werden.

1.2 Historie

Die fraktionale Differentialrechnung hat ihren Ursprung in der Frage, ob die Bedeutung des Ausdrucks $\frac{d^n}{dx^n}$, $n \in \mathbb{N}$ erweitert werden kann, wenn $n \in \mathbb{Q}$, $\mathbb{R}$ oder gar $\mathbb{C}$ ist. Den hier aufgeführten letzten Fall werde ich in dieser Arbeit nur noch in den Schlussbemerkungen aufgreifen. Jedoch wird $n \in \mathbb{Q}$ bzw. $\mathbb{R}$ zentral besprochen werden.

Als 1695 Leibniz auf die Frage von l'Hospital, was denn $n = \frac{1}{2}$ bedeuten könne, antwortete, "Es wird in einem Widerspruch enden", nahm er zusätzlich noch, um nicht später eines besseren belehrt zu werden, in seine Antwort auf "Von diesem Widerspruch sich aber eines Tages nützliche Konsequenzen ergeben werden".

1819 wurde dann eine Ableitung mit willkürlichem n das erste mal in einem Text erwähnt. Lacroix widmete in einem 700 Seiten starken Text über Differential- und Integralrechnung weniger als zwei Seiten dem Thema der gebrochenen Ableitungen. Er fand heraus, dass für $y(x) = x^n$, $n, m \in \mathbb{N}$ die Ableitung $\frac{d^m}{dx^m} y(x) = \frac{n!}{(n-m)!} x^{n-m}$ ist, und dies auch für $m = \frac{1}{2}$ gelten könne, wenn man die Fakultäten durch die entsprechende Γ-Funktion ersetzen würde. Heutige Definitionen fraktionaler Ableitungen bestätigen diese Vermutung.

Liouville veröffentlichte zwischen 1832 und 1835 einige Aufsätze zu diesem Thema. Motiviert durch einen Aufsatz eines Studenten begann 1847 Riemann sich mit gebrochenen Ableitungen zu beschäftigen und vervollständigte Liouvilles Vorarbeit zu der heute bekannten Definition. Erst 1967 veröffentlichte Caputo seine “umgekehrt“ definierte Version der gebrochenen Ableitung, die, wie wir später sehen werden, “näher an der realen Welt“ liegen wird.

2. Definitionen fraktionaler Ableitungen

2.0 Klassisches Integral und Ableitung

Beginnend mit den klassischen Varianten von Integral und Ableitung werden zuerst zwei Operatoren für Integration und Differentiation eingeführt und dann von $n \in \mathbb{N}$ auf $n \in \mathbb{R}_+$ erweitert.

Def. 2.0.1 Sei $f : I \to \mathbb{R}$ stetige Funktion auf dem Intervall I : $[a,b]$.
Mit I_a^n , $n \in \mathbb{N}$ wird der Integraloperator und mit D^n , $n \in \mathbb{N}$ der Differentialoperator bezeichnet, m.a.W. für $x \in [a,b]$

$$I_a f(x) = \int_a^x f(t)dt \qquad , I_a^n = I_a I_a^{n-1} \qquad n \geq 2$$

$$Df(x) = \frac{d}{dx} f(x) \qquad , D^n = DD^{n-1} \qquad n \geq 2$$

Für höhere Integrale ergibt sich daraus

Lemma 2.0.1 Sei f Riemann integrierbar auf $[a,b]$, dann gilt für $x \in [a,b]$ und $n \in \mathbb{N}$

$$I_a^n f(x) = \frac{1}{(n-1)!} \int_a^x (x-t)^{n-1} f(t)dt \qquad , I_a^n = I_a I_a^{n-1} \qquad n \geq 2$$

Beweis. Sei $n \in \mathbb{N}$. Per Induktion ergibt sich

1. Induktionsanfang: $n = 1$

$$I_a^1 f(x) = \frac{1}{0!} \int_a^x (x-t)^0 f(t)dt = \int_a^x f(t)dt$$

2. Induktionsannahme: Lemma 2.0.1 für $n \in \mathbb{N}$ bereits bewiesen.
3. Induktionsschritt: $n \to n+1$

$$\begin{aligned} I_a^{n+1} f(x) = I_a I_a^n f(x) &= I_a \left[\frac{1}{(n-1)!} \int_a^x (x-t)^{n-1} f(t)dt \right] && (*_1) \\ &= \frac{1}{(n-1)!} \int_a^x \left(\int_a^t (t-\tau)^{n-1} f(\tau)d\tau \right) dt && (*_2) \\ &= \frac{1}{(n-1)!} \int_a^x \left(\int_\tau^x (t-\tau)^{n-1} f(\tau)dt \right) d\tau && (*_3) \end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{(n-1)!} \int_a^x \left[\frac{1}{n} (t-\tau)^n f(\tau) \right]_\tau^x d\tau \\
&= \frac{1}{(n-1)!} \int_a^x \frac{1}{n} (x-\tau)^n f(\tau) d\tau \\
&= \frac{1}{n!} \int_a^x (x-t)^n f(t) dt.
\end{aligned}
$$

Hierbei wurde in $(*_1)$ im Induktionsschritt die Integrationsvariable t durch τ ersetzt, damit t für die äußere Integration in $(*_2)$ verwendet werden kann. Von $(*_2)$ nach $(*_3)$ wurde Fubinis Theorem verwendet, nach dem die Integrationsreihenfolge vertauscht werden darf. Da allerdings die Integrationsgrenzen im Integranden vorkommen, müssen auch diese entsprechend geändert werden.

□

Nun wollen wir uns überlegen, wie die beiden Operatoren aufeinander wirken. Gemäß dem bekannten Fundamentalsatz der Differential- und Integralrechnung (siehe [1]) zeigt sich, dass die Integration gerade die Umkehrung der Differentiation ist. In Lemma 2.0.3 kommt dazu, dass die Stammfunktion nur bis auf eine additive Konstante eindeutig bestimmt ist. Des weiteren halte ich fest, dass die Operatoren linear sind, was hier zwar nicht vorgerechnet werden soll, aber an späterer Stelle noch nützlich sein wird.

Lemma 2.0.2 Sei $n \in \mathbb{N}$ und f integrierbar auf $x \in [a,b]$. Es gilt

$$D^n I_a^n f = f$$

Beweis. Aus $D^n = D \overset{n}{\cdots} D$ und $I_a^n = I_a \overset{n}{\cdots} I_a$ genügt es die Aussage für $n = 1$ zu zeigen. Sei F eine Stammfunktion von f, so folgt

$$DI_a f(x) = \frac{d}{dx} \int_a^x f(t) dt = \frac{d}{dx} (F(x) - F(a)) = F'(x) = f(x)$$

□

Lemma 2.0.3 Sei $n \in \mathbb{N}$ und f differenzierbar auf $x \in [a,b]$. Es gilt

$$I_a^n D^n f = f - c,$$

mit $c = \sum_{k=0}^{n-1} \frac{f^{(n)}(a)}{k!} x^k \in \mathbb{R}$.

Beweis. Sei $n \in \mathbb{N}$. Per Induktion ergibt sich

1. Induktionsanfang: $n = 1$

$$I_a D = \int_a^x \frac{d}{dx} f(x) dt = \int_a^x f'(t) dt = f(x) - f(a)$$

2. Induktionsannahme: Lemma 2.0.3 für $n \in \mathbb{N}$ bereits bewiesen.

3. Induktionsschritt: $n \to n+1$

$$\begin{aligned} I_a^{n+1} D^{n+1} f(x) &= I(I^n D^n) D f(x) \\ &= I\left(Df - \sum_{k=0}^{n-1} \frac{f^{(k+1)}(a)}{k!} x^k \right) \\ &= f(x) - f(a) - I \sum_{k=0}^{n-1} \frac{f^{(k+1)}(a)}{k!} x^k \\ &= f(x) - f(a) - \sum_{k=0}^{n-1} \frac{f^{(k+1)}(a)}{k!(k+1)} x^{k+1} \\ &= f(x) - f(a) - \sum_{k=1}^{n} \frac{f^{(k+1)}(a)}{k!} x^k \\ &= f(x) - \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!} x^k \end{aligned}$$

□

Lemma 2.0.4 Die Operatoren I_a^n und D^n sind linear.

Zum Beweis siehe z.B. [1].

Def. 2.0.2 Die Funktion $\Gamma : (0, \infty) \to \mathbb{R}$ ist definiert durch

$$\Gamma(x) := \int_0^\infty t^{x-1} e^{-t} dt$$

und wird Eulers Gamma Funktion genannt. Es folgt die Funktionalgleichung der Γ - Funktion als

Lemma 2.0.5 Es gilt $\Gamma(x+1) = x\Gamma(x)$, $x > 0, x \in \mathbb{R}$.

Beweis. Sei $x > 0, x \in \mathbb{R}$.

$$\begin{aligned} \Gamma(x+1) &= \int_0^\infty t^x e^{-t} dt = \lim_{\epsilon \to 0, R \to \infty} \int_\epsilon^R t^x e^{-t} dt \\ &= \lim_{\epsilon \to 0, R \to \infty} \left(\left[-e^{-t} t^x \right]_\epsilon^R + x \int_\epsilon^R t^{x-1} e^{-t} dt \right) \\ &= x \int_0^\infty t^{x-1} e^{-t} dt = x\Gamma(x). \end{aligned}$$

□

Lemma 2.0.6 Es gilt $(n-1)! = \Gamma(n)$, $n \in \mathbb{N}$.

Beweis. Sei $n \in \mathbb{N}$. Per Induktion zeige ich genau wie oben zuerst
1. Induktionsanfang: $n = 1$

$$\Gamma(1) = (1-1)! = 0! = 1$$

2. Induktionsannahme: Lemma 2.0.6 für $n \in \mathbb{N}$ bereits bewiesen.
3. Induktionsschritt: $n \to n+1$

Durch wiederholte Anwendung von Lemma 2.0.5 .
$\Gamma(n+1) = n\Gamma(n) = \ldots = n(n-1)\cdot\ldots\cdot 1\cdot\Gamma(1) = n!$. □

Die Γ - Funktion interpoliert also die Fakultät. Sie bietet sich somit an, den Schritt innerhalb der Integraldefinition von $n \in \mathbb{N} \to n \in \mathbb{R}$ zu vervollkommnen. In der Definition des fraktionalen Integrals ändern sich im Vergleich zu Def. 2.0.1 zwei Dinge. Zum einen ist nun die Funktion aus $L_1[a,b]$, dem Raum der Lebesgue-messbaren Funktionen, für die $\int_a^b |f(x)|dx < \infty$ gilt, zum anderen wird die Fakultät durch die Γ - Funktion ersetzt. Dass diese Änderungen keine Schwierigkeiten machen, wird später noch ersichtlich.

Zunächst allerdings noch zwei weitere Funktionen, die wir im Verlauf noch benötigen.

Def. 2.0.3 Die Beta - Funktion B zweier komplexer Veränderlichen z und w mit $Re(z) > 0$ und $Re(w) > 0$ ist definiert durch

$$B(z,w) = \int_0^1 \tau^{z-1}(1-\tau)^{w-1}d\tau.$$

Um den engen Zusammenhang zwischen der Γ-Funktion und der B-Funktion zu zeigen, betrachten wir noch das folgende

Lemma 2.0.7 Es gilt

$$B(z,w) = \int_0^1 \tau^{z-1}(1-\tau)^{w-1}d\tau = \frac{\Gamma(z)\Gamma(w)}{\Gamma(z+w)}.$$

Beweis. Hierzu benutzen wir die Technik der Laplace-Transformation des Faltungsproduktes, welches in 4.1.2.1 noch näher erläutert wird. Sämtliche Transformationen und Rücktransformationen sind in [5] ersichtlich. Der Beweis wurde [2] entnommen.

Betrachten wir die Funktion

$$h_{z,w}(t) = \int_0^t \tau^{z-1}(1-\tau)^{w-1} d\tau,$$

welche eine Faltung zweier Funktionen $h_1(t) = t^{z-1}$ und $h_2(t) = t^{w-1}$ darstellt. Weiterhin sind $h_{z,w}(1) = B(z,w)$ und $H_1(s) = \frac{\Gamma(z)}{s^z}$ bzw. $H_2(s) = \frac{\Gamma(w)}{s^w}$ die Laplace-Transformierten von $h_1(t)$ bzw. $h_2(t)$. Damit ergibt sich die Laplace-Transformierte von $h_{z,w}(t)$ als Produkt von H_1 und H_2 zu

$$H_{z,w}(s) = \frac{\Gamma(z)}{s^z}\frac{\Gamma(w)}{s^w} = \frac{\Gamma(z)\Gamma(w)}{s^{z+w}}.$$

Da der Zähler konstant ist ergibt die Rücktransformation von $H_{z,w}(s)$

$$h_{z,w}(t) = \frac{\Gamma(z)\Gamma(w)}{\Gamma(z+w)} t^{z+w-1},$$

woraus für $t = 1$ obige Aussage folgt.

□

Def. 2.0.4 Die Mittag-Leffler Funktion $E_{\alpha,\beta}(t)$ ist definiert durch

$$E_{\alpha,\beta}(t) = \sum_{k=0}^{\infty} \frac{t^k}{\Gamma(\alpha k + \beta)} \quad , \quad \alpha, \beta > 0.$$

Für $\alpha = \beta = 1$ ergibt sich die Exponentialfunktion

$$E_{1,1}(t) = \sum_{k=0}^{\infty} \frac{t^k}{k!} = e^t.$$

2.1 Riemann-Liouville Integral und Ableitung

Def. 2.1.1 Der Riemann-Liouville Integraloperator

Sei $n \in \mathbb{R}_+$. Der Riemann-Liouville Integraloperator I_a^n ist auf $L_1[a,b]$ durch

$$I_a^n f(x) := \frac{1}{\Gamma(n)} \int_a^x (x-t)^{n-1} f(t) dt \quad , \quad a \leq x \leq b$$

definiert, wobei für $n = 0$ der Operator I_a^0 der identische Operator I ist.

Die Existenz dieses Integrals ist gewährleistet, da der Integrand aus dem Produkt der stetigen Funktion $(x-t)^{n-1}$ und der integrierbaren Funktion $f(x)$ besteht.

Nun soll auch gleich der Differentialoperator eingeführt werden. Weiter unten werden noch einige Eigenschaften dieser beiden Operatoren angeführt.

Def. 2.1.2 Der Riemann-Liouville Differentialoperator

Sei $n \in \mathbb{R}_+$ und $m = \lceil n \rceil$ mit $\lceil n \rceil = \min \{z \in \mathbb{Z} : z \geq x\}$. Der Riemann-Liouville Differentialoperator D_a^n ist durch

$$D_a^n f := D^m I_a^{m-n} f$$

definiert.

Zunächst zwei Beispiele.

Beispiel. 2.1.1 $I_a^n (t-a)^\nu \quad$ mit $\quad \nu > -1 \quad$ und $\quad n > 0$.

Im ersten Schritt wird durch die Substitution $\xi(\tau) = \frac{(\tau - a)}{(t-a)}$ eine Verschiebung der Integrationsgrenzen erreicht, um anschließend die Definition von Eulers Beta Funktion zu verwenden.

$$\begin{aligned} I_a^n (t-a)^\nu &= \frac{1}{\Gamma(n)} \int_a^t (t-\tau)^{n-1} (t-a)^\nu d\tau \\ &= \frac{1}{\Gamma(n)} \int_0^1 (t-a-\xi(t-a))^{n-1} (\xi(t-a))^\nu (t-a) d\xi \\ &= \frac{1}{\Gamma(n)} \int_0^1 (1-\xi)^{n-1} (t-a)^{n-1} \xi^\nu (t-a)^{\nu+1} d\xi \end{aligned}$$

$$
\begin{aligned}
&= \frac{(t-a)^{\nu+n}}{\Gamma(n)} \int_0^1 (1-\xi)^{n-1} \xi^{\nu} d\xi \\
&= \frac{(t-a)^{\nu+n}}{\Gamma(n)} \frac{\Gamma(\nu+1)\Gamma(n)}{\Gamma(\nu+n+1)} \\
&= \frac{\Gamma(\nu+1)}{\Gamma(\nu+n+1)} (t-a)^{\nu+n}
\end{aligned}
$$

Beispiel. 2.1.2 $D_a^n (t-a)^{\nu}$ mit $\nu > -1$ und $n > 0$ und $m = \lceil n \rceil$.

Unter Verwendung von Beispiel 2.1.1 ergibt sich.

$$
\begin{aligned}
D_a^n (t-a)^{\nu} &= D^m I_a^{m-n} (t-a)^{\nu} \\
&= \frac{\Gamma(\nu+1)}{\Gamma(\nu+m-n+1)} \frac{d^m}{dt^m} (t-a)^{\nu+m-n} \\
&= \frac{\Gamma(\nu+1)}{m!\Gamma(\nu-n+1)} m! (t-a)^{\nu-n} \\
&= \frac{\Gamma(\nu+1)}{\Gamma(\nu-n+1)} (t-a)^{\nu-n}
\end{aligned}
$$

Genau wie im klassischen Fall soll nun $D_a I_a$ und $I_a D_a$ berechnet werden. Zuvor jedoch das später benötigte

Lemma 2.1.1 Sei $m, n \geq 0$. Für jedes $f \in L_1[a,b]$ gilt

$$
I_a^m I_a^n f = I_a^{m+n} f.
$$

Beweis. Seien m, n und f wie in Lemma 2.1.1 definiert, dann ergibt sich unter Verwendung des Theorems von Fubini (wie in Lemma 2.0.1) und gleicher Substitution wie in Beispiel 2.1.1

$$
\begin{aligned}
I_a^m \left(I_a^n f(x)\right) &= \frac{1}{\Gamma(m)} \int_a^x (x-t)^{m-1} I_a^n f(t) dt \\
&= \frac{1}{\Gamma(m)\Gamma(n)} \int_a^x (x-t)^{m-1} \int_a^t (t-\tau)^{n-1} f(\tau) d\tau \, dt \\
&= \frac{1}{\Gamma(m)\Gamma(n)} \int_a^x f(\tau) \int_\tau^x (x-t)^{m-1} (t-\tau)^{n-1} dt \, d\tau \\
&= \frac{1}{\Gamma(m)\Gamma(n)} \int_a^x f(\tau) (x-\tau)^{m+n-1} \int_0^1 (1-z)^{m-1} z^{n-1} dz \, d\tau \\
&= \frac{1}{\Gamma(m)\Gamma(n)} \int_a^x f(\tau) (x-\tau)^{m+n-1} \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)} d\tau
\end{aligned}
$$

$$= \frac{1}{\Gamma(m+n)} \int_a^x (x-\tau)^{m+n-1} f(\tau) d\tau$$
$$= I_a^{m+n} f(x).$$

□

Lemma 2.1.2 Sei $n \geq 0$. Für jedes $f \in L_1[a,b]$ gilt

$$D_a^n I_a^n f = f.$$

Beweis. Den Fall $n = 0$ brauchen wir nicht zu beweisen, da dann beide Operatoren die identischen Operatoren sind. Für $n > 0$ benötigen wir $m = \lceil n \rceil$ und es folgt

$$D_a^n I_a^n f(x) = D^m I_a^{m-n} I_a^n f(x) = D^m I_a^m f(x) = f(x)$$

Im ersten Schritt haben wir die Definition der fraktionalen Ableitung eingesetzt, im zweiten Schritt Lemma 2.1.1 benutzt und im dritten Schritt ausgenutzt, dass $m \in \mathbb{N}$ ist und somit Lemma 2.0.2 gilt.

□

Lemma 2.1.3 Sei $n > 0$. Wenn eine Funktion $\phi \in L_1[a,b]$ existiert, zu der $f = I_a^n \phi$ Stammfunktion ist, dann gilt

$$I_a^n D_a^n f = f$$

Beweis. Zum Beweis benutzen wir Lemma 2.1.2

$$I_a^n D_a^n f = I_a^n [D_a^n I_a^n \phi] = I_a^n \phi = f$$

□

Ist f allerdings keine Stammfunktion einer Funktion ϕ, so ergibt sich

Lemma 2.1.4 Sei $n > 0$ und $m = \lfloor n \rfloor + 1$. Sei weiterhin $I_a^{m-n} f(x)$ $(m-1)$ mal absolut stetig differenzierbar, dann ist

$$I_a^n D_a^n f(x) = f(x) - \sum_{k=0}^{m-1} \frac{(x-a)^{n-k-1}}{\Gamma(n-k)} \lim_{z \to a_+} D^{m-k-1} I_a^{m-n} f(z)$$

und speziell für $0 < n < 1$ erhält man

$$I_a^n D_a^n f(x) = f(x) - \frac{(x-a)^{n-1}}{\Gamma(n)} \lim_{z \to a_+} I_a^{1-n} f(z).$$

Beweis. $I_a^{m-n} f(x)$ ist $(m-1)$ mal absolut stetig differenzierbar, was die Stetigkeit von $D^{m-1} I_a^{m-n} f$ garantiert. Deshalb existiert eine Funktion $\phi \in L_1$, so dass $D^{m-1} I_a^{m-n} f = D^{m-1} I_a^{m-n} f(a) + I_a^1 \phi$ ist. Diese klassische Differentialgleichung der Ordnung $(m-1)$ wird von der Funktion

$$I_a^{m-n} f(x) = \sum_{k=0}^{m-1} \frac{(x-a)^k}{k!} \lim_{z \to a_+} D^k I_a^{m-n} f(z) + I_a^m \phi(x) \qquad (*)$$

gelöst, was man durch einsetzen in die Differentialgleichung bestätigen kann. Daraus folgt

$$\begin{aligned} I_a^n D_a^n f(x) &= I_a^n D^m I_a^{m-n} f(x) \\ &= I_a^n D^m \left[\sum_{k=0}^{m-1} \frac{(\cdot - a)^k}{k!} \lim_{z \to a_+} D^k I_a^{m-n} f(z) + I_a^m \phi \right](x) \\ &= I_a^n D^m I_a^m \phi(x) + \sum_{k=0}^{m-1} \frac{I_a^n D^m [(\cdot - a)^k](x)}{k!} \lim_{z \to a_+} D^k I_a^{m-n} f(z) \\ &= I_a^n \phi(x), \end{aligned}$$

weil der m-te Ableitungsoperator D^m jeden Summanden Null werden lässt. Es bleibt noch zu zeigen, dass $I_a^n \phi(x)$ die rechte Seite von Lemma 2.1.4 bildet. Wendet man den Operator D_a^{m-n} auf $(*)$ an, so erhält man unter Verwendung von Lemma 2.1.2

$$\begin{aligned} f(x) &= \sum_{k=0}^{m-1} \frac{D_a^{m-n}[(\cdot - a)^k](x)}{k!} \lim_{z \to a_+} D^k I_a^{m-n} f(z) + D_a^{m-n} I_a^m \phi(x) \\ &= \sum_{k=0}^{m-1} \frac{D_a^{m-n}[(\cdot - a)^k](x)}{k!} \lim_{z \to a_+} D^k I_a^{m-n} f(z) + D_a^1 I_a^{1-m+n} I_a^m \phi(x) \\ &= \sum_{k=0}^{m-1} \frac{(x-a)^{k+n-m}}{\Gamma(k+n-m+1)} \lim_{z \to a_+} D^k I_a^{m-n} f(z) + I_a^n \phi(x), \qquad (**) \end{aligned}$$

wobei im letzten Schritt Ableitungsbeispiel 2.1.2 und Lemma 2.1.3 verwendet wurde. Zum Schluss wird k in obiger Summe durch $(m-k-1)$ gemäß der Ableitung D^{m-k-1} aus der rechten Seite von Lemma 2.1.4 substituiert und die Gleichung $(**)$ nach $I_a^n \phi(x)$ aufgelöst. Man erhält

$$I_a^n D_a^n f(x) = I_a^n \phi(x) = f(x) - \sum_{k=0}^{m-1} \frac{(x-a)^{n-k-1}}{\Gamma(n-k)} \lim_{z \to a_+} D^{m-k-1} I_a^{m-n} f(z).$$

□

2.2 Caputo Ableitung

Der im letzten Abschnitt 2.1 vorgestellte Riemann-Liouville Differentialoperator weist in der Praxis angewendet entscheidende Nachteile auf. An dieser Stelle wollen wir einen diesbezüglich verbesserten Differentialoperator vorstellen, der genau wie der Riemann-Liouville Differentialoperator den Riemann-Liouville Integraloperator benutzt. Der Vergleich der beiden Differentialoperatoren wird im nächsten Kapitel 2.3 erfolgen.

Def. 2.2.1 Der Caputo Differentialoperator

Sei $n > 0$, $n, a \in \mathbb{R}$ und $m = \lceil n \rceil$. Sei D_a^n der in Def. 2.1.2 definierte Riemann-Liouville Differentialoperator und $T_m[f;a]$ das Taylor-Polynom der Funktion f der Ordnung m mit Entwicklungspunkt a. Der Caputo Differentialoperator $D_{a_C}^n$ ist durch

$$D_{a_C}^n f := D_a^n (f - T_{m-1}[f;a])$$

definiert.
Für $n \in (0,1)$ ergibt sich daraus die übersichtlichere Darstellung

$$D_{a_C}^n f(x) := D_a^n (f(x) - f(a))$$

bzw. mit g(x) := f(x) - f(a)

$$D_{a_C}^n f(x) := D_a^n g(x)$$

Zu dieser Definition gibt es noch eine äquivalente andere Möglichkeit, die ohne Taylor-Reihe auskommt und nur die m-te Ableitung enthält.

Def. 2.2.2 Seien m, n, a, f und D_a^n anaolog Def. 2.2.1 definiert und $f^{(n)}$ die n-te Ableitung der Funktion f. Der Caputo Differentialoperator kann durch

$$D_{a_C}^n f(x) = \frac{1}{\Gamma(m-n)} \int_a^x (x-\tau)^{m-n-1} f^{(m)}(\tau) d\tau$$

bzw. in der Operatorschreibweise

$$D_{a_C}^n f(x) = I_a^{m-n} D^m f(x)$$

definiert werden.

Die Äquivalenz beider Definitionen ergibt sich von Def. 2.2.2 ausgehend und auf diese den Operator I_a^n angewendet unter Verwendung von Lemmata 2.0.3 und 2.1.1 wie folgt:

$$\begin{aligned} I_a^n D_{a_C}^n f(x) &= I_a^n I_a^{m-n} D^m f(x) \\ &= I_a^m D^m f(x) \\ &= f(x) - \sum_{k=0}^{m-1} \frac{f^{(k)}(a)}{k!} x^k \\ &= f(x) - T_{m-1}[f;a]) \end{aligned}$$

Erneutes Anwenden des Operators D_a^n auf die Gleichung ergibt Def. 2.2.1 .

□

Genau wie bei dem Riemann-Liouville Operator zeige ich nun, wie fraktionale Integration und Ableitung aufeinander wirken. Der Integraloperator ist dabei immer der Riemann-Liouville Integraloperator. Es zeigt sich, das hier die Eigenschaften der klassischen Integration und Ableitung zum Vorschein kommen.

Lemma 2.2.1 Sei f stetig und $n > 0$, dann gilt

$$D_{a_C}^n I_a^n f = f.$$

Beweis. Sei $m = \lceil n \rceil$. Gemäß der Definition des Caputo-Differentialoperators im ersten Schritt und unter Anwendung von Lemma 2.1.2 im zweiten Schritt erhält man

$$D_{a_C}^n [I_a^n f] = D_a^m (I_a^n f - T_{m-1}[I_a^n f;a]) = f - D_a^m (T_{m-1}[I_a^n f;a])$$

Der zweite Summand rechts ist Null, was ersichtlich wird, wenn man das fraktionale Integral in der Taylor-Reihe darstellt. Es enthält lediglich die aus den Ableitungen des Integrals an der Stelle a entstehenden Konstanten geteilt durch entsprechende Fakultäten und die Terme $(x-a)^k$ für $k = 0, ..., m-1$. Letztere Terme fallen aber bei der anschließenden Integration des $D_a^n = D^m - I_a^{m-n}$ weg, weil der Integrand $(x-\tau)^{m-n-1}(\tau - a)^{m-1}$ nach Einsetzen von x und a für τ immer Null wird.

□

Lemma 2.2.2 Sei $n > 0$, $m = \lceil n \rceil$ und f $m-1$ mal absolut stetig differenzierbar. Dann gilt

$$I_a^n D_{a_C}^n f(x) = f(x) - \sum_{k=0}^{m-1} \frac{D^k f(a)}{k!}(x-a)^k.$$

Und speziell für $0 < n < 1$ erhält man

$$I_a^n D_{a_C}^n f(x) = f(x) - f(a).$$

Beweis. Wenn $g := f - T_{m-1}[f;a]$ gewählt wird, müssen wir nur noch $I_a^n D_{a_C}^n f = g$ beweisen.

$$I_a^n D_{a_C}^n f(x) = I_a^n D_a^n g(x) = g(x) - \sum_{k=0}^{m-1} \frac{(x-a)^{n-k-1}}{\Gamma(n-k)} \lim_{z \to a_+} D^{m-k-1} I_a^{m-n} g(z)$$

Das zweite Gleichheitszeichen gilt auf Grund von Lemma 2.1.3 . Weiterhin verschwindet der Summand rechts bzw. die Summe, da $g(z)$ bei a eine $m-$fache Nullstelle besitzt und somit das Integral I_a^{m-n} Null wird.

□

2.3 Vergleich zwischen Riemann-Liouville - und Caputo Ableitung

In diesem Abschnitt will ich drei Unterschiede hervorheben, die für die mathematische Behandlung fraktionaler Differentialgleichungen wesentlich sind. Ein Unterschied zeigt sich in der Verkettung beider Operatoren in der Art $I_a^n D_a^n$ bzw. $I_a^n D_{a_C}^n$. Als zweiten Unterschied möchte ich die Ableitung einer konstanten Funktion $f(x) = c$ behandeln und als dritten das Verhalten der Operatoren für $t \to a$, d.h. das Verhalten der Ableitung für "kleine Zeiten" t.

Bemerkung 2.3.1 Den ersten Unterschied habe ich dabei eigentlich schon behandelt, vergleicht man nämlich Lemma 2.1.3 und Lemma 2.2.2, so fällt auf, dass im Falle $0 < n < 1$, $n \in \mathbb{R}_+$ der Caputo-Operator dem Verhalten

der klassischen Ableitung (für $n \in \mathbb{N}$) entspricht, hingegen der Riemann-Liouville-Operator zu einem eher kontraintuitivem Ergebnis führt. Dies rührt daher, dass bei der Riemann-Liouville Ableitung (genau wie bei der Caputo Ableitung höherer Ordnung) innerhalb der Ableitung integriert wird und der Integrand den Term $(x-t)^{n-1}$ enthält, d.h. die Integrations"konstante" von der unabhängigen Variablen x abhängt und somit beim Ableiten nicht verschwindet.

Bemerkung 2.3.2 Wendet man den Riemann-Liouville Operator D_a^n für $n \in \mathbb{R}_+$ auf die Funktion $f(x) = c$ an, so ergibt sich

$$\begin{aligned} D_a^n c &= D\frac{1}{\Gamma(n)}\int_a^x (x-t)^{n-1} c\,dt \\ &= -D[(x-t)^n c]_a^x = Dc(x-a)^n = c(n-1)(x-a)^{n-1} \end{aligned}$$

eine Funktion, die im Gegensatz zur klassischen Ableitung nur für $n = 1$ gleich Null ist, in allen anderen Fällen aber keineswegs diese gewohnte Eigenschaft zeigt. Wird hingegen der Caputo Operator $D_{a_C}^n$ auf $f(x) = c$ angewendet, so folgt mit $m = \lceil n \rceil$

$$\begin{aligned} D_{a_C}^n c &= D_a^n (c - T_{m-1}[c;a]) \\ &= D_a^n (c - (c + (x-a)D^1 c + \cdots + \frac{1}{(m-1)!}(x-a)^{m-1} D^{m-1} c)) = 0 \end{aligned}$$

das der klassischen Ableitung entsprechende Ergebnis.

Bemerkung 2.3.3 Betrachtet man den Grenzwert $x \to a$, so fallen auch hier Unterschiede der beiden Ableitungs-Definitionen auf. Im Riemann-Liouville Fall erhalten wir unter bestimmten Voraussetzungen für f eine Divergenz durch den Term im Integranden $(x-t)^{n-1}$, wobei diese Divergenz durch zwei Eigenschaften von $f(x)$ verhindert wird. Erstens, falls die Funktion $f(x)$ Terme der Form $(x-a)^w$ mit $w \geq n$ enthält die nach Integration für $x \to a$ einen Wert $D_a^n f(a) < \infty$ annehmen, oder zweitens, falls die ersten $\lfloor n \rfloor$ Ableitungen (mit $\lfloor n \rfloor = \max\{z \in \mathbb{Z} : z \leq x\}$), d.h. alle Ausdrücke der Form $f^{(k)}(a) = 0$ mit $k = 0, 1, ..., \lfloor n \rfloor$ sind. Im letzteren Fall gilt auch für die Ableitung $D_a^n f(a) = 0$. Im Caputo Fall kann auch für $f(a) \neq 0$ für $0 < n < 1$ die Divergenz der Ableitung in diesem Punkt ausgeschlossen werden (es sei denn f selbst ist divergent), da hier keine die Divergenz verursachenden Nennerterme der Form $(x-a)$ vorkommen.

3. Fraktionale Ableitungen mit MAPLE

3.0 Einführung in MAPLE

Zu Beginn dieses Kapitels will ich kurz die Vorgehensweise erklären, die hier zugrunde gelegt ist. Der theoretische Teil ist im vorangegangenen Kapitel erklärt worden und in diesem Kapitel soll dies nun mittels MAPLE verdeutlicht und praktisch veranschaulicht werden. Dabei werden zum einen die notwendigen Befehle erklärt, der Programmcode gezeigt und die Protokollierung von MAPLE dargestellt und erläutert.

Alle hier dargestellten Programmcodes und die zugehörige Protokollierung von MAPLE sind in Internet erhältlich. Näheres dazu im Anhang.

Der in MAPLE einzugebende Programmcode wird in `Maschinenschrift` linksbündig und die MAPLE - Protokollierung *kursiv* zentriert oder als Grafik dargestellt. Hierzu einige Beispiele:

Soll mittels MAPLE die Summe zweier Zahlen berechnet werden, so kann man dies mit der üblichen + Verknüpfung anweisen. Jeder Befehl muss an seinem Ende ein Semikolon erhalten, um MAPLE anzuweisen die Befehle abzuarbeiten.

```
3 + 5;
```

$$8$$

Für Operatoren oder Berechnungen, die MAPLE nicht von Hause aus zur Verfügung stellt, gibt es in MAPLE die Möglichkeit, Prozeduren zu erstellen, die mehrere Befehle und Funktionen enthalten und die sich auch gegenseitig aufrufen können. Dies geschieht mittels Prozeduren, einer leistungsstarken Entwicklungsumgebung, in der eingegebener eigener Programmcode von MAPLE durch Aufruf der Prozedur in einem Arbeitsgang bearbeitet wird.

Eine Beispielprozedur sei hier dargestellt:

```
Beispiel_2:=proc(n::nonnegint)
  RETURN(sum('i^2','i'=1..n);)
end:
```

Da in der Prozedur-Definition hinter dem der Prozedur zu übergebenden Parameter der Typ dieses Parameters nach dem Doppelpunkt mit angegeben wurde, wird MAPLE veranlasst, diesen Parameter direkt bei Prozeduraufruf auf Korrektheit zu überprüfen. Schlägt diese Überprüfung fehl, wird von MAPLE eine Fehlermeldung ausgegeben und die Prozedur wird nicht ausgeführt.

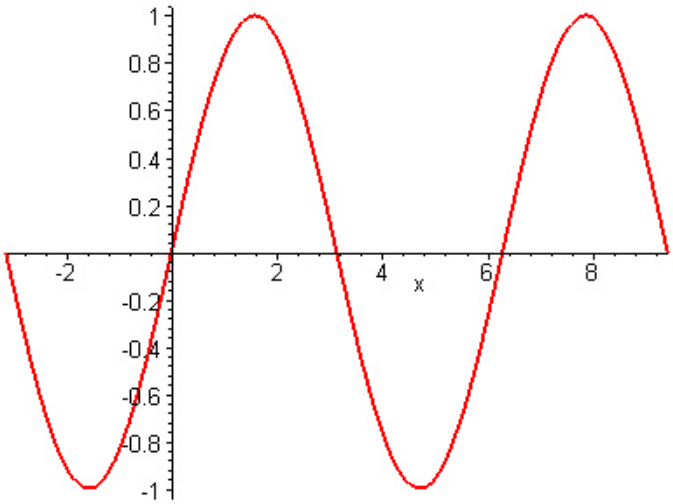
Der Prozedur-Aufruf erfolgt durch die Eingabe des Prozedurnamens gefolgt von den der Prozedur zu übergebenden Parametern in Klammern:

```
Beispiel_2(5);
```

55

Zur graphischen Ausgabe kann grundsätzlich der plot-Befehl benutzt werden. Diesem werden zwei (optional weitere die Ausgabe steuernde) Parameter übergeben. Der erste ist die darzustellende Funktion und der zweite der Wertebereich der plot-Ausgabe. Hierzu ein Beispiel:

```
plot(sin(x),x=-Pi..3*Pi);
```



Innerhalb von Prozeduren können auch logische Operatoren und Schleifen programmiert werden.

```
Beispiel_4:=proc(n::nonnegint)
for i from 1 to n do
  if isprime(i) then print(i); fi;
od: end:
```

Diese Prozedur würde alle Primzahlen bis `n` ausgeben.

3.1 Riemann-Liouville Integral und Ableitung mit MAPLE

3.1.1 Die Operatoren `I_RL` und `D_RL`

Bevor die im letzten Kapitel eingeführten Operatoren benutzt werden können, müssen sie zuerst MAPLE bekannt gemacht werden, da MAPLE diese nicht in den programmeigenen Funktionsbibliotheken zur Verfügung stellt.

Gemäß den Definitionen 2.1.1 und 2.1.2 des Integral - und Ableitungsoperators müssen zur Darstellung der fraktionalen Ableitung einer Funktion $f(x)$ zuerst diese Operatoren als Prozeduren eingegeben werden. Der Prozedurname ist dabei bis auf MAPLE-eigene Namen frei wählbar.

```
I_RL:=proc(a::numeric,n::nonneg,f::procedure)
if n=0 then
  eval(f)(x);
else
  1/GAMMA(n)*int((x-t)^(n-1)*f(t),t=a..x);
fi:
end:
```

```
D_RL:=proc(a::numeric,n::nonneg,f::procedure)
m:=ceil(n);
if m = 0 then
  eval(f)(x);
else
  diff(unapply(I_RL(a,m-n,f),x),x$m);
fi;
end:
```

In der `I_RL`-Prozedur wird zuerst überprüft, ob nicht vielleicht der identische Operator zurückgegeben werden soll. Dies geschieht dann mittels `eval(f)`, wobei der Rückgabewert keine Prozedur sein soll, was durch das nachfolgende `(x)` erreicht wird, welches aus der Prozedur `eval(f)` wieder einen Ausdruck erzeugt.

Der Befehl `ceil(n)` in der D_RL-Prozedur gibt die kleinste Zahl zurück, die größer oder gleich n ist analog dem in Def. 2.1.2 verlangten. Da der Rückgabewert der I_RL-Prozedur ein Ausdruck ist, muss dieser per `unapply`-Befehl in eine Funktion umgewandelt werden, die dann per `diff`-Befehl m-mal abgeleitet werden kann.

Nun sollen fraktionale Integrale und Ableitungen veranschaulicht werden.

3.1.2 Riemann-Liouville Integrale in einem Diagramm

Um ein anschaulich einfaches Beispiel zu erhalten definiere ich zunächst eine konstante Funktion $f(x) = 1$, was mit MAPLE folgendermaßen realisiert wird.

```
f := x -> 1;
```

$$f := 1$$

Anschließend werden diverse Integrale berechnet und die Ergebnisse in einer Variablen, genauer einem MAPLE-Ausdruck, gespeichert.

```
I_RL_0:=I_RL(0,0,f);
I_RL_0_2:=I_RL(0,0.2,f);
I_RL_0_8:=I_RL(0,0.8,f);
I_RL_1:=I_RL(0,1,f);
I_RL_1_5:=I_RL(0,1.5,f);
I_RL_2:=I_RL(0,2,f);
```

$$I\_RL\_0 := 1$$

$$I\_RL\_0\_2 := 1.089124421\, x^{\left(\frac{1}{5}\right)}$$

$$I\_RL\_0\_8 := 1.073671274\, x^{\left(\frac{4}{5}\right)}$$

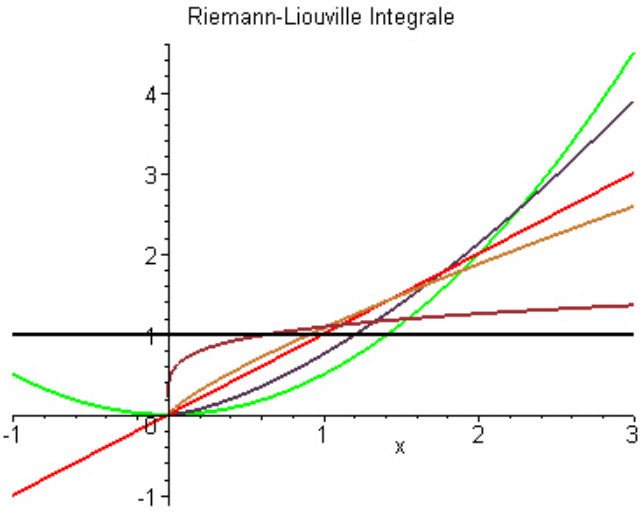
$$I\_RL\_1 := x$$

$$I\_RL\_1\_5 := .7522527780\, x^{\left(\frac{3}{2}\right)}$$

$$I\_RL\_2 := \frac{1}{2} x^2$$

Diese Ausdrücke können nun mittels des bereits erwähnten `plot`-Befehls gedruckt werden.

```
plot([f(x),I_RL_0(x)-.01,I_RL_0_2(x),I_RL_0_8(x), \
  I_RL_1(x),I_RL_1_5(x),I_RL_2(x)],x=-1..3,       \
  color=[black,blue,brown,gold,red,violet,green], \
  thickness=2,title=(“Riemann-Liouville Integrale“));
```



Die fraktionalen Integrale sind nur für $x \geq 0$ definiert und man kann in guter Näherung sagen, dass diese "zwischen" den klassischen Integralen, d.h. für $n \in \mathbb{N}$ liegen. In gewohnter Weise sind diese klassischen Integrale auch für negative x definiert.

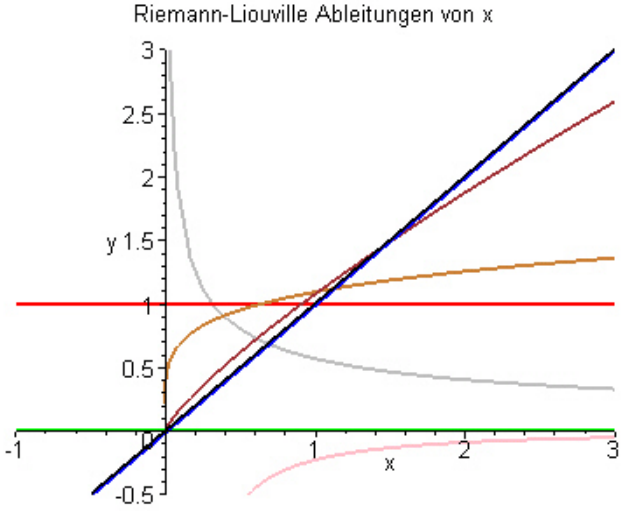
3.1.3 Riemann-Liouville Ableitungen in einem Diagramm

Das selbe kann man nun auch mit fraktionalen Ableitungen programmieren, d.h. zuerst werden die Ableitungen berechnet und in MAPLE-Ausdrücken gespeichert und anschließend mittels `plot`-Befehl ausgedruckt. Der Anschaulichkeit halber verwende ich hierzu allerdings eine andere Funktion $g(x) = x$ und unterdrücke die Protokollierung durch den Doppelpunkt am Ende jedes Befehls.

```
D_RL_0:=D_RL(0,0,g):
D_RL_0_2:=D_RL(0,0.2,g):
D_RL_0_8:=D_RL(0,0.8,g):
D_RL_1:=D_RL(0,1,g):
D_RL_1_5:=D_RL(0,1.5,g):
D_RL_2:=D_RL(0,2,g):
D_RL_2_3:=D_RL(0,2.3,g):
```

Das Diagramm wird analog dem obigen Verfahren programmiert.

```
plot([g(x),D_RL_0(x),D_RL_0_2(x),D_RL_0_8(x),          \
  D_RL_1(x),D_RL_1_5(x),D_RL_2(x),D_RL_2_3(x)],x=-1..3, \
  color=[black,blue,brown,gold,red,violet,green,pink],  \
  thickness=2,title=(“Riemann-Liouville Ableitungen von x“));
```



Hier wird deutlich, dass eine fraktionale Ableitung “zwischen“ den klassischen Ableitungen liegen kann (vgl. der braune [0.2-te Ableitung] und der goldene [0.8-te Ableitung] Graph liegen “zwischen“ dem blauen [0. Ableitung] und dem roten [1. Ableitung] Graphen). Eine fraktionale Ableitung kann allerdings für $x \to a$ auch divergent sein (vgl. der graue [1.5-te Ableitung] Graph), wenn für $k \in \mathbb{N}, 0 \leq k \leq n$ mindestens eine k-te Ableitung am Entwicklungspunkt a ungleich Null ist , d.h. die Funktion dort keine $\lfloor n \rfloor$-fache Nullstelle besitzt.

Dieses Verhalten kann man sich sehr schön auch in drei Dimensionen anschauen, wobei als Parameter für den 2-dimensionalen Wertebereich die unabhängige Variable x und die Ordnung der Ableitung gewählt werden.

3.1.4 Riemann-Liouville Ableitungen dargestellt in 3 Dimensionen

Dazu muss allerdings einige Vorarbeit geleistet werden, um MAPLE anzuweisen die berechneten Werte der entsprechenden Ableitungen einer Funktion $f(x)$ in einem Diagramm anzuzeigen.

Zuerst die Prozedur, die die $\mathbb{R}^2_+ \to \mathbb{R}$ - Funktion darstellt:

```
D_RL_gr:=proc(a,FFF::procedure,A1,s_A,A2,X1,s_X,X2)
global LL;
Z:=0;
for AA from A1 by s_A to A2 do
  F:=unapply(D_RL(a,AA,FFF),x);
  L:=[[AA,0,0]];
  for xx from X1 by s_X to X2 do
    L := [ op(L), [AA,xx,evalf(F(xx))] ];
  od:
  if Z = 0 then
    LL := [L];
    Z:=1;
  else
    LL := [op(LL),L];
  fi;
od: 'ok';
end:
```

Der Prozedur werden mehrere Parameter übergeben. Zum einen den Entwicklungspunkt a, dann natürlich die Funktion selber, die höchste mit der Schrittweite s zu berechnende Ableitung A , d.h. es werden alle Ableitungen $n = ks$ mit $k = 0, ..., \frac{A}{s}$ berechnet und als letzte zwei Parameter Anfangs - und Endstellen des Intervalls über das differenziert werden soll. Die Ausgabe erfolgt in einer Liste `LL`, die dann mittels einer zweiten Prozedur dargestellt wird, wozu zu Beginn obiger Prozedur die Variable `LL` global definiert sein muss, damit auch von Außerhalb dieser Prozedur auf die Variable zugegriffen werden kann.

Die Vorgehensweise in dieser Prozedur ist die folgende: Es werden zwei ineinander liegende Schleifen durchlaufen, wobei in der äußeren Schleife `s` die Ableitungen und in der inneren Schleife `xx` die x-Werte laufen. Für jede Ableitung wird zuerst mit der `I_RL`-Prozedur die Ableitung berechnet und dann in diese Ableitung alle beliebigen x-Werte zwischen `X1` und `X2` eingesetzt, wobei nach jedem einsetzen jedes Ergebnis als Liste mit drei Einträgen (die drei Koordinaten des Punktes) gespeichert wird. Diese einzelnen Koordinatenlisten werden in einer größeren Liste `L` gespeichert, d.h. nach jedem Schleifenschritt der inneren Schleife `xx` wird die neue Koordinatenliste an `L` angehängt. Nach komplettem Durchlauf von Schleife `xx` enthält `L` die kompletten Daten für eine Ableitung. Als zweiter Teil der äußeren Schleife `s` wird nun die Liste `L` ein Eintrag in der alle Ableitungen enthaltenen Liste `LL`. Die interne Befehlsstruktur von MAPLE macht es dabei notwendig, das im ersten Schritt `LL := L` erzeugt wird und dann in jedem weiteren Schritt neue Listen `L` an die Liste `LL` angehängt werden. Dazu dient die Variable `Z`.

Kommen wir nun zur zweiten Prozedur, die aus der Liste `LL` die graphische Ausgabe erzeugt. Diese Prozedur `D_RL_graph` ist die eigentliche Prozedur, die später aufgerufen wird. Die vorherige Prozedur `D_RL_gr` wird dabei intern von `D_RL_graph` aufgerufen.

```
D_RL_graph:=proc(a,FFFF::procedure,A1,s_A,A2,
                 X1,s_X,X2,Z1,Z2,delta,phi)
D_RL_gr(a,FFFF,A1,s_A,A2,X1,s_X,X2);
printf(''\ n %a. bis %a. Ableitung der \
  Funktion f(x) = %A'',A1,A2,eval(FFFF)(x)):
```

```
PLOT3D(MESH(LL),VIEW(A1..A2,X1..X2,Z1..Z2), \
  AXESLABELS(Abl,x,z),AXES(BOXED),ORIENTATION(delta,phi));
end:
```

Dieser Prozedur werden die selben Parameter wie der vorherigen übergeben und zusätzlich noch vier weitere die graphische Ausgabe betreffende. Zum einen soll das Diagramm die z-Achse von `Z1` bis `Z2` enthalten, zum anderen soll das Diagramm durch die Parameter `delta` und `phi` so gedreht werden, dass der Betrachter leicht einen Überblick über den Verlauf der Funktion bekommen kann.

Ruft man die Prozedur `D_RL_graph` auf, wird zuerst das Ergebnis durch die Prozedur `D_RL_gr` berechnet und von dieser in der global definierten Variablen `LL` gespeichert. Anschließend erfolgt eine formatierte `printf`-Anweisung, die dem Betrachter wiederholt, was nach der Berechnung im Diagramm angezeigt wird. Kern dieser Prozedur ist die Kombination der Befehle `PLOT3D()` und `MESH()`. Im Teil 3.0 dieses Kapitels wurde bereits der `plot`-Befehl eingeführt, der für entsprechende Funktionen durch `plot3d` auf drei Dimensionen erweitert wird. Diese Befehle erzeugen aus der Funktion entsprechend dem ihnen übergebenen Wertebereich durch einsetzen eine Datenstruktur `PLOT` bzw. `PLOT3D`, die von MAPLE grafisch angezeigt werden kann. Da unsere darzustellende Funktion als Koordinatenliste `LL` vorliegt, kann dieser Schritt übersprungen werden und wir können gleich den `PLOT3D`-Befehl verwenden, um die Daten darzustellen. Der `MESH`-Befehl ist eine Erweiterung des `GRID`-Befehls von zwei auf drei Dimensionen. Wie bereits erwähnt liegen unsere Daten als Listen vor, wobei jede Koordinate eine Liste aus drei Elementen ist. Diese Koordinatenlisten sind Einträge in einer Liste, welche alle Funktionswerte einer Ableitung darstellt. Alle Ableitungen zusammen bilden eine Liste, die folglich eine Liste von Koordinatenlisten enthält. Der `MESH`-Befehl enthält nun ein Protokoll, das ihn in die Lage versetzt diese Einträge in entsprechender Reihenfolge abzuarbeiten und dem `PLOT3D`-Befehl zur Verfügung zu stellen.

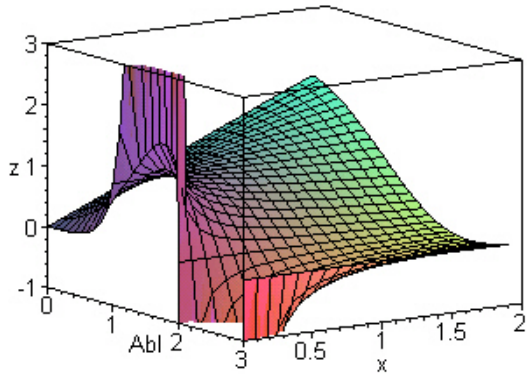
Sollen z.B. die fraktionalen Ableitungen der Funktion

```
g := x -> x:
```

graphisch dargestellt werden, so erfolgt dies mittels Aufruf von

```
D_RL_graph(0,g,0,.1,3,.001,.1,2,-1,3,-35,75);
```

0. bis 3. Ableitung der Funktion f(x) = x

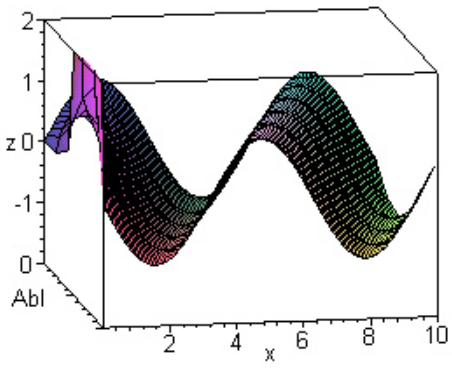


Für $x \to a$ konvergieren die gebrochenen Ableitungen für $0 < n < 1$ und divergieren für $1 < n < 2$ und $2 < n < 3$. Weiterhin ist die erste Ableitung konstant 1 und die zweite Ableitung konstant 0 analog den klassischen Ergebnissen.

Bevor wir zu den fraktionalen Ableitungsdefinitionen von Caputo kommen, hier noch ein letztes Beispiel.

```
h := x -> sin(x):
D_RL_graph(0,h,0,.4,2,.001,.1,10,-2,2,-10,75);
```

0. bis 2. Ableitung der Funktion f(x) = sin(x)



3.2 Caputo Ableitung mit MAPLE

Genau wie bei der Programmierung des Riemann-Liouville Ableitungsoperators `D_RL` können wir nun auch bei dem Caputo-Ableitungsoperator vorgehen, wobei hier nur Definition 2.2.1 vorgestellt wird. Die äquivalente Definition 2.2.2 befindet sich im Anhang der Datei `3_2_C_abl.mws`.

3.2.1 Der Operator `D_C`

Wir definieren diesen Operator gemäß Definition 2.2.1 wie folgt, wobei dieser Operator den `D_RL`-Operator, d.h. die Riemann-Liouville Integral- und Differentialoperatoren verwendet, welcher hier nicht noch einmal wiederholt werden.

```
D_C:=proc(a::numeric,n::nonneg,f)
local g,h;
global G;
m:=ceil(n);
if m=0 then
  eval(f)(x);
else
  if m=1 then
    g:=x->f(x)-f(a);
    simplify(diff(D_RL(a,n,g),x$m));
  else
    G:=taylor(f(x),x=a,m-1);
    GG:=unapply(convert(G,polynom),x);
    GGG:=x->f(x)-GG(x);
    simplify(D_RL(a,n,GGG));
  fi;
fi;
end:
```

Dem Operator wird der Entwicklungspunkt a, die Ableitungshöhe n, $n \in \mathbb{R}_+$ und die Funktion f. Nun wird überprüft, welche Ableitung berechnet werden soll, da für $n = 0$ der identischen Operator und für $0 < n < 1$ die in Def. 2.2.1 gezeigte einfachere Darstellung zurückgegeben werden kann.

Die Taylor-Reihe wird nur für $n > 1$ benötigt, d.h. für diesen Fall wird in `G` die Taylor-Reihe der Funktion f gespeichert, anschließend der Fehlerterm am Ende dieser Reihe abgeschnitten, die Differenz $f - T_{m-1}[f;a]$ gebildet und diese Funktion an den Riemann-Liouville Differentialoperator `D_RL` übergeben, wobei ein anschließendes `simplify` die Ausgabe evtl. übersichtlicher darstellen kann. Der Fehlerterm der Taylor-Reihe wird dabei durch Ausnutzung des `convert(G,polynom)`-Befehls realisiert, in dem der Ausdruck `G` in ein Polynom konvertiert wird. Intern wird in diesem Fall lediglich der letzte Term der Taylorreihe, d.h. der Term der die Fehlerordnung angibt, durch 0 substituiert. Die global definierte Variable `G` dient lediglich Kontrollzwecken, um die Taylor-Reihe der Funktion f auch außerhalb der Prozedur verwenden zu können.

3.2.2 Caputo-Ableitungen in einem Diagramm

Mittels dieser Prozedur kann nun ebenfalls die Funktion

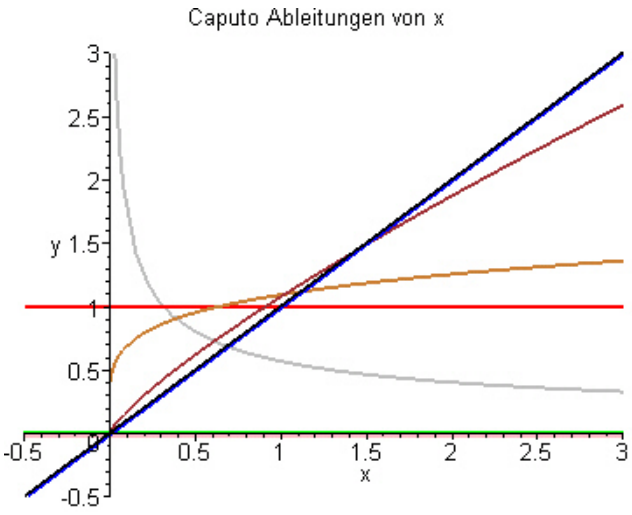
```
g := x -> x:
```

dargestellt werden, wobei wieder das selbe Verfahren wie bei den Riemann-Liouville Ableitungen verwendet werden kann, entweder in zwei oder drei Dimensionen.

```
D_C_0:=D_C(0,0,g):
D_C_0_2:=D_C(0,0.2,g):
D_C_0_8:=D_C(0,0.8,g):
D_C_1:=D_C(0,1,g):
D_C_1_5:=D_C(0,1.5,g):
D_C_2:=D_C(0,2,g):
D__2_3:=D_C(0,2.3,g):
```

Das Diagramm wird analog dem obigen Verfahren programmiert.

```
plot([g(x),D_C_0(x),D_C_0_2(x),D_C_0_8(x),          \
  D_C_1(x),D_C_1_5(x),D_C_2(x),D_C_2_3(x)],x=-1..3,  \
  color=[black,blue,brown,gold,red,grey,green,pink], \
  thickness=2,title=(“Caputo Ableitungen von x“));
```



Das Ergebnis ist identisch dem der Riemann-Liouville Ableitung, weil $g(x)$ am Entwicklungspunkt $a = 0$ Null ist. Einzig die höhere Ableitung für $n > 2$ ist, anders als im Riemann-Liouville Fall gleich Null. Der Grund dafür wird im nächsten Kapitel besprochen werden.

3.2.3 Caputo-Ableitungen dargestellt in 3 Dimensionen

Und nun soll eine dreidimensionale Vorstellung einer Caputo-Ableitung vermittelt werden. Innerhalb der Prozeduren wird nun die entsprechende Caputo-Ableitungs-Prozedur aufgerufen. Alle anderen Befehle sind identisch.

```
D_C_gr:=proc(a,FFF::procedure,A1,s_A,A2,X1,s_X,X2)
global LL;
Z:=0;
for AA from 0 by s to A do
  F:=unapply(D_C(a,AA,FFF),x);
  L:=[[AA,0,0]];
  for xx from X1 by .1 to X2 do
    L := [ op(L), [AA,xx,evalf(F(xx))] ];
  od:
  if Z = 0 then
    LL := [L];
    Z:=1;
  else
```

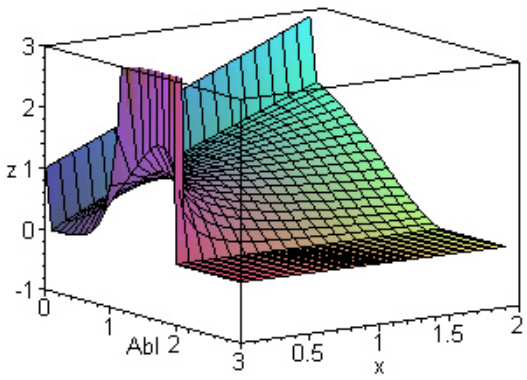
```
    LL := [op(LL),L];
  fi;
od:
'ok';
end:

D_C_graph:=proc(a,FFFF::procedure,A1,s_A,A2,
                X1,s_X,X2,Z1,Z2,delta,phi)
D_C_gr(a,FFFF,s,A,X1,X2);
printf("\ n 0. bis %a. Ableitung der \
  Funktion f(x) = %A",A,eval(FFFF)(x)):
PLOT3D(MESH(LL),VIEW(A1..A2,X1..X2,Z1..Z2), \
  AXESLABELS(Abl,x,z),AXES(BOXED),ORIENTATION(delta,phi));
end:
```

Als Beispiel sei die Funktion $g(x) = x + 1$ gewählt, da diese am Entwicklungspunkt a ungleich Null ist und so besser Unterschiede zwischen den Ableitungsdefinitionen diskutiert werde kann. Die Caputo-Ableitung einer Funktion $g(x) = x$ würde in ihren Ableitungen den selben Verlauf zeigen wie die Riemann-Liouville-Ableitungen.

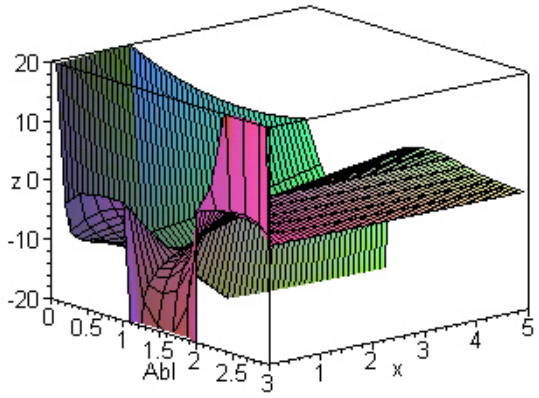
```
g := x -> x+1:
D_C_graph(0,g,0,.1,3,.001,.1,2,-1,3,-35,75):
```

0. bis 3. Caputo-Ableitung der Funktion f(x) = x+1



```
h := x -> (x-6)^2:
D_C_graph(0,h,0,.2,3,.001,.1,5,-20,20,-40,70):
```

0. bis 3. Caputo-Ableitung der Funktion f(x) = (x-6)^2



3.3 Vergleich zwischen Riemann-Liouville - und Caputo - Ableitung mit MAPLE

Nun wollen wir uns den drei in Kapitel 2.3 angesprochenen Unterschieden der beiden Operatoren widen. Die einzelnen Unterschiede sollen mit MAPLE visualisiert und anschließend erläutert werden.

3.3.1 Verkettung von $I_a^n D_a^n$ und $I_a^n D_{a_C}^n$

Dazu betrachten wir die Funktion $f(x) = x$ und lassen uns $I_a^n D_a^n$ und $I_a^n D_{a_C}^n$ grafisch darstellen. Auf eine dreidimensionale Darstellung sei hier verzichtet, da der Vergleich in zwei Dimensionen übersichtlicher ist. Ebenfalls lasse ich die MAPLE-Protokollierung der Eingabe weg, da der Programmcode selbsterklärend ist.

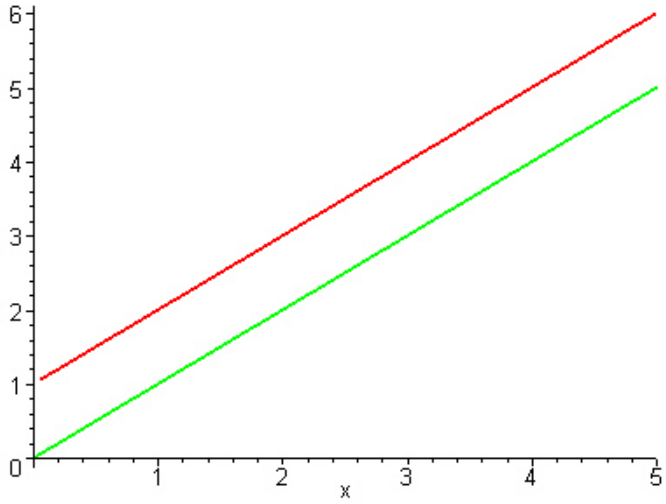
```
f := x -> x + 1:
D_RL_0_2:=unapply(D_RL(0,0.2,f),x):
I_RL_D_RL_0_2:=unapply(I_RL(0,0.2,D_RL_0_2),x):
```

Zum Vergleich die gleiche Berechnung mit dem Caputo-Operator

```
D_C_0_2:=unapply(D_C(0,0.2,f),x):
I_RL_D_C_0_2:=unapply(I_RL(0,0.2,D_C_0_2),x):
```

und anschließend die Ausgabe in einem Diagramm.

```
plot([I_RL_D_RL_0_2(x),I_RL_D_C_0_2(x)],x=0..5,color=[red,green]);
```

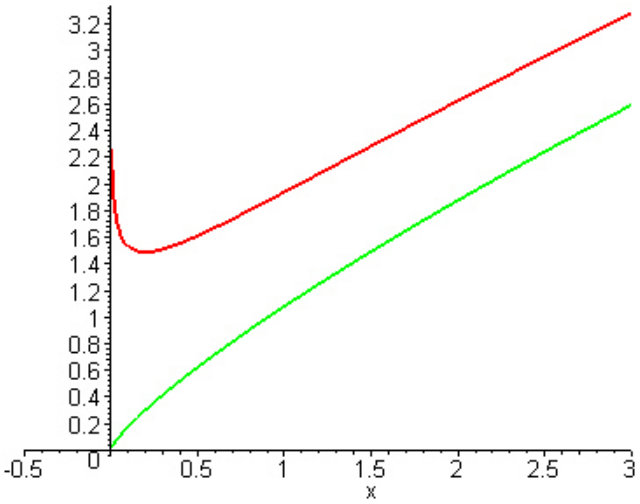


Deutlich zeigt sich, dass die beiden Operatoren unterschiedlich wirken, wie das auch schon bei der theoretischen Herleitung ersichtlich wurde.

3.3.2 Verhalten der Ableitungsoperatoren für $x \to a$

Um eine Idee der Unterschiede zwischen den Operatoren für $x \to a$ zu bekommen, werden nachfolgend Ableitungen mit den jeweiligen Operatoren berechnet und im Intervall $[0, 3]$ in einem Diagramm dargestellt, wobei wieder die Protokollierung von MAPLE unterdrückt und nur das Diagramm ausgegeben wird.

```
f := x -> x + 1:
D_RL_0_2:=D_RL(0,0.2,f):
D_C_0_2:=D_C(0,0.2,f):
plot([D_RL_0_2,D_C_0_2],x=-.5..3,color=[red,green],thickness=2);
```



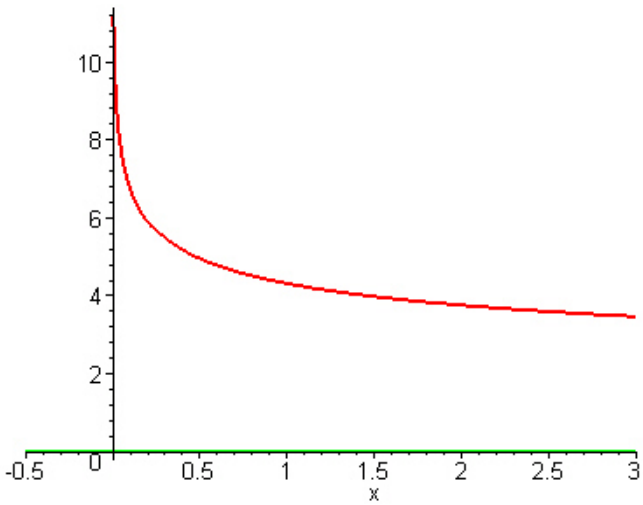
3.3.3 Ableitung einer konstanten Funktion $f(x) = c$

Die Funktion

```
f := x -> 5:
```

wird zuerst mit beiden Operatoren abgeleitet und dann in einem Diagramm dargestellt.

```
D_RL_0_2:=D_RL(0,0.2,f):
D_C_0_2:=D_C(0,0.2,f):
plot([D_RL_0_2,D_C_0_2],x=-.5..3,color=[red,green],thickness=2);
```



Der Caputo-Operator zeigt die dem klassischen Fall entsprechende Ableitung, hingegen weist der Riemann-Liouville Operator eine in der Praxis nicht immer vorteilhafte Divergenz auf.

4. Fraktionale Differentialgleichungen

In diesem Kapitel werden Lösungsmethoden zu gewöhnlichen fraktionalen Differentialgleichungen vorgestellt. Als analytische Verfahren werden die Picard-Iteration und die Laplace-Transformation behandelt, wobei erstere nur für den Grenzwert $n \to \infty$ der Iteration analytisch ist.

Anschließend wird ein numerisches Verfahren behandelt, welches eine fraktionale Variante des klassischen Adams-Bashforth-Moulton Verfahrens darstellt. Darüber hinaus kann die erwähnte Picard-Iteration für $n < \infty$ ebenfalls zu den numerischen Verfahren gezählt werden.

4.1 Analytische Lösungsverfahren

4.1.1 Picard-Iteration

Das Verfahren kann sowohl für Riemann-Liouville - als auch für Caputo Differentialgleichungen verwendet werden, wobei o.B.d.A. nachfolgend der Entwicklungspunkt $a = 0$ gesetzt wird.

Die Differentialgleichung $D_0^n y(x) = f(x, y(x))$ wird in eine Integralgleichung $y(x) = R + I_0^n f(x, y(x))$ mit R als "Summenterm" des in Lemma 2.1.4 bzw. Lemma 2.2.2 erläuterten und aus $I_0^n D_0^n y(x)$ entstandenen Terms umgeformt. Genügt f einer Lipschitz-Bedingung in der zweiten Variablen y, so kann mittels der rechten Seite dieser Gleichung ein Operator A definiert werden, der angewendet auf ein y_0 im Vektorraum der stetigen Funktionen auf einem noch näher zu bestimmenden Intervall einen Fixpunkt y^* besitzt. Dieser Fixpunkt muss die gesuchte Lösung sein und die Iteration $y_i = Ay_{i-1}$ für $i = 1, 2, ...$ führt für $i \to \infty$ auf y^*, da gemäß Weissinger's Fixpunktsatz der Fixpunkt eindeutig ist und y^* eine Lösung obiger Differentialgleichung ist, welche gemäß dem Satz von Picard-Lindelöf ebenfalls eindeutig ist.

Ein Beweis des Verfahrens wird in [3] gegeben.

Im Falle einer Riemann-Liouville Differentialgleichung muss das Existenzintervall bei 0 halboffen, d.h. $(0,h]$, $h \in \mathbb{R}$ sein, sofern die Anfangsbedingung $\lim_{z\to 0_+} I_0^{m-n} y(z) \neq 0$, $m = \lceil n \rceil$ ist, da sonst das Integral in der Integralgleichung rechts nicht existiert. Bei Caputo Differentialgleichungen muss diese Einschränkung nicht gemacht werden und das Existenzintervall $[0,h]$ ist abgeschlossen.

4.1.1.1 Riemann-Liouville Differentialgleichungen

Sei $n > 0, n \notin \mathbb{N}$ und $m = \lceil n \rceil$. Die Riemann-Liouville Differentialgleichung

$$D_0^n y(x) = f(x, y(x))$$

ausgestattet mit den Anfangsbedingungen

$$D_0^{n-k} y(0) = c_k, \quad (k = 1, ..., m-1), \quad \lim_{z\to 0_+} I_0^{m-n} y(z) = c_m$$

dargestellt in der äquivalenten Integralgleichung

$$y(x) = \sum_{k=1}^{m} \frac{c_k x^{n-k}}{\Gamma(n-k+1)} + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t, y(t)) dt$$

kann mit dem Operator

$$Ay(x) := \sum_{k=1}^{m} \frac{c_k x^{n-k}}{\Gamma(n-k+1)} + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t, y(t)) dt$$

angewendet auf die Funktion

$$y_0(x) = \sum_{k=1}^{m} \frac{c_k x^{n-k}}{\Gamma(n-k+1)}$$

durch die Iteration $(A^j y_0)_{j=1}^{\infty}$ eindeutig gelöst werden.

4.1.1.2 Beispiel : $D_0^n y(x) = \lambda y(x)$, $0 < n < 1$, $\lim_{z\to 0_+} I_0^{m-n} y(z) = c_1$ und $m = \lceil n \rceil$.

Integration der Differentialgleichung mit I_0^n liefert

$$y(x) := \frac{c_1 x^{n-1}}{\Gamma(n)} + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} \lambda y(t) dt$$

und Anwendung des Operators A auf y_0 ergibt

$$\begin{aligned}
y_1(x) = Ay_0(x) &= \frac{c_1}{\Gamma(n)}\left(x^{n-1} + \frac{\lambda}{\Gamma(n)}\int_0^x (x-t)^{n-1}t^{n-1}dt\right)\\
&= \frac{c_1}{\Gamma(n)}\left(x^{n-1} + \frac{\lambda}{\Gamma(n)}\int_0^1 (x-x\xi)^{n-1}(x\xi)^{n-1}x\,d\xi\right)\\
&= \frac{c_1}{\Gamma(n)}\left(x^{n-1} + \frac{\lambda}{\Gamma(n)}\int_0^1 x^{n-1}(1-\xi)^{n-1}x^{n-1}\xi^{n-1}x\,d\xi\right)\\
&= \frac{c_1}{\Gamma(n)}\left(x^{n-1} + \frac{\lambda x^{2n-1}}{\Gamma(n)}\frac{\Gamma(n)\Gamma(n)}{\Gamma(2n)}\right)\\
&= \frac{c_1}{\Gamma(n)}x^{n-1} + \frac{\lambda}{\Gamma(2n)}x^{2n-1}\\
&= c_1x^{n-1}\left(\frac{1}{\Gamma(n)} + \frac{\lambda x^n}{\Gamma(2n)}\right),
\end{aligned}$$

wobei die Integralgrenzen mittels der Substitution $\xi(t) = \frac{t}{x}$ erfolgte und dann Eulers Beta-Funktion benutzt wurde.

Anwendung des Operators A auf y_1 ergibt

$$\begin{aligned}
y_2(x) = Ay_1(x) &= \frac{c_1}{\Gamma(n)}\left(x^{n-1} + \int_0^x (x-t)^{n-1}\lambda(\frac{t^{n-1}}{\Gamma(n)} + \frac{\lambda t^{2n-1}}{\Gamma(2n)})dt\right)\\
&= \frac{c_1}{\Gamma(n)}\left(x^{n-1} + \frac{\lambda}{\Gamma(n)}\int_0^x (x-t)^{n-1}t^{n-1}dt + \frac{\lambda^2}{\Gamma(2n)}\int_0^x (x-t)^{n-1}t^{2n-1}dt\right)\\
&= \frac{c_1}{\Gamma(n)}x^{n-1} + \frac{\lambda}{\Gamma(2n)}x^{2n-1} + \frac{\lambda}{\Gamma(3n)}x^{3n-1}\\
&= c_1x^{n-1}\left(\frac{1}{\Gamma(n)} + \frac{\lambda x^n}{\Gamma(2n)} + \frac{\lambda^2x^{2n}}{\Gamma(3n)}\right),
\end{aligned}$$

wobei ab hier die Regelmäßigkeit ersichtlich wird, die eine kompaktere Schreibweise unter Verwendung der Mittag-Leffler Funktion erlaubt, nämlich

$$y(x) = c_1\sum_{k=0}^{\infty}\frac{\lambda^k x^{(k+1)n-1}}{\Gamma((k+1)n)} = c_1x^{n-1}\sum_{k=0}^{\infty}\frac{\lambda^k x^{kn}}{\Gamma((k+1)n)} = c_1x^{n-1}E_{n,n}(\lambda x^n).$$

Aber auch kompliziertere Differentialgleichungen lassen sich mit dieser Methode lösen, was durch das nächste Beispiel ersichtlich werden soll.

4.1.1.3 Beispiel : $D_0^n y(x) = x^2 + xy(x)$, $0 < n < 1$, $\lim\limits_{z\to 0_+} I_0^{m-n}y(z) = c_1$.

Integration der Differentialgleichung mit I_0^n liefert

$$y(x) := \frac{c_1x^{n-1}}{\Gamma(n)} + \frac{1}{\Gamma(n)}\int_0^x (x-t)^{n-1}(t^2 + ty(t))dt$$

und Anwendung des Operators A auf y_0 analog wie im Beispiel 4.1.1.2 ergibt

$$y_1 = \frac{c_1 x^{n-1}}{\Gamma(n)} + \frac{2}{\Gamma(n+3)} x^{n+2} + \frac{c_1 n}{\Gamma(2n+1)} x^{2n}.$$

Anwendung des Operators A auf y_1 ergibt

$$y_2 = y_1 + \frac{2(n+3)}{\Gamma(2n+4)} x^{2n+3} + \frac{c_1 n(2n+1)}{\Gamma(3n+2)} x^{3n+1}$$

und Anwendung des Operators A auf y_2 ergibt

$$y_3 = y_2 + \frac{2(n+3)(2n+4)}{\Gamma(3n+5)} x^{3n+4} + \frac{c_1 n(2n+1)(3n+2)}{\Gamma(4n+3)} x^{4n+2},$$

was auf Grund der auftretenden Regelmäßigkeiten eine kompaktere Schreibweise

$$y(x) = \frac{c_1 x^{n-1}}{\Gamma(n)} + \sum_{k=1}^{\infty} a_k x^{kn+k+1} + c_1 b_k x^{(k+1)n+k-1},$$

mit $a_k = \frac{\prod_{j=0}^{k-1}(jn+j+2)}{\Gamma(kn+k+2)}$ und $b_k = \frac{\prod_{j=0}^{k-1}((j+1)n+j-1)}{\Gamma((k+1)n+k)}$ erlaubt.

4.1.1.4 Caputo Differentialgleichungen

Sei $n > 0$ und $m = \lceil n \rceil$. Die Caputo Differentialgleichung

$$D_{0_C}^n y(x) = f(x, y(x))$$

ausgestattet mit den Anfangsbedingungen

$$D^k y(0) = y_0^{(k)}, \quad (k = 0, 1, ..., m-1)$$

dargestellt in der äquivalenten Integralgleichung

$$y(x) = \sum_{k=0}^{m-1} \frac{D^k y(0)}{n!} x^k + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t, y(t)) dt$$

kann mit dem Operator

$$Ay(x) := \sum_{k=0}^{m-1} \frac{D^k y(0)}{n!} + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t, y(t)) dt$$

angewendet auf die Funktion

$$y_0(x) = \sum_{k=0}^{m-1} \frac{D^k y(0)}{n!} x^k$$

durch die Iteration $(A^j y_0)_{j=1}^{\infty}$ eindeutig gelöst werden.

4.1.1.5 Beispiel : $D^n_{0_C} y(x) = \lambda y(x)$, $0 < n < 1$, $y(0) = y_0$.

Integration der Differentialgleichung mit I^n_0 liefert

$$y(x) := y_0 + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} \lambda y(t) dt$$

und Anwendungen des Operators A auf $y_i(x), i = 0, 1, ...$ ergibt

$$y_1(x) = Ay_0(x) = y_0 + \frac{\lambda y_0}{n\Gamma(n)} x^n$$
$$y_2(x) = y_0 + \frac{\lambda y_0}{\Gamma(n+1)} x^n + \frac{\lambda^2 y_0}{\Gamma(2n+1)} x^{2n}$$

und daraus

$$y(x) = y_0 \sum_{k=0}^{\infty} \frac{\lambda^k}{\Gamma(kn+1)} x^{kn} = y_0 E_{n,1}(\lambda x^n)$$

wobei hier wie im Beispiel 4.1.1.2 die Mittag-Leffler Funktion benutzt wurde.

4.1.1.6 Beispiel : $D^n_{0_C} y(x) = x^2 + xy(x)$, $0 < n < 1$, $y(0) = y_0$.

Wieder wird der Riemann-Liouville Operator I^n_0 auf die Differentialgleichung angewendet und anschließend mit dem so definierten Operator A das Iterationsverfahren aufgebaut. Aus den $y_i, i = 0, 1, ..$ läßt sich so die exakte Lösung als Summe darstellen.

Es ergeben sich für

$$y_1(x) = Ay_0(x) = y_0 + \frac{2}{\Gamma(n+3)} x^{n+2} + \frac{y_0}{\Gamma(n+2)} x^{n+1}$$
$$y_2(x) = y_1(x) + \frac{2(n+3)}{\Gamma(2n+4)} x^{2n+3} + \frac{y_0(n+2)}{\Gamma(2n+3)} x^{2n+2}$$
$$y_3(x) = y_2(x) + \frac{2(n+3)(2n+4)}{\Gamma(3n+5)} x^{3n+4} + \frac{y_0(n+2)(2n+3)}{\Gamma(3n+4)} x^{3n+3}$$

womit wie in vorangegangenen Beispielen eine Summenschreibweise realisiert werden kann, und als Lösung folgt

$$y(x) = y_0 + \sum_{k=1}^{\infty} a_k x^{kn+k+1} + y_0 b_k x^{kn+k},$$

mit $a_k = \frac{\prod_{j=0}^{k-1}(jn+j+2)}{\Gamma(kn+k+2)}$ und $b_k = \frac{\prod_{j=0}^{k-1}((j+1)n+j+1)}{\Gamma((k+1)n+k+1)}$.

Sämtliche Lösungen können durch Einsetzen in die entsprechenden Differenzialgleichungen verifiziert werden.

4.1.2 Verfahren mittels Laplace-Transformation

Eine elegante Methode zur Lösung von Differentialgleichungen macht Gebrauch von der Laplace-Transformation. Das sogenannte Laplace - Integral eignet sich besonders zur Behandlung von Differentialgleichungen und Differentialgleichungssystemen mit Anfangsbedingungen. Dabei wird die unbekannte Funktion in der Differentialgleichung einer Transformation unterzogen, die die Differentialgleichung in eine algebraische Gleichung überführt. Die Lösung dieser algebraischen Gleichung muss mittels Rücktransformation in den ursprünglichen Raum zurückgeführt werden, was insgesamt i.a. einfacher ist, als die Differentialgleichung direkt zu lösen.

Die Laplace-Transformation ist eine Integraltransformation, die jeder Funktion $y(x)$, $x > 0$, eine Bildfunktion $Y(s)$ der komplexen Veränderlichen s gemäß

$$\mathcal{L}(y) = Y(s) := \int_0^\infty y(x)e^{-sx}dx$$

zuordnet, wobei ich nachfolgend die Originalfunktionen in Kleinbuchstaben und die zugehörigen Bildfunktionen in entsprechenden Grossbuchstaben schreiben werde. Damit das uneigentliche Integral und damit die Bildfunktion $Y(s)$ überhaupt existiert, muss das Integral für jedes s einen endlichen Wert annehmen, was durch folgende zwei Bedingungen an $y(x)$ erreicht wird.

Bedingung 1) $y : [0,\infty) \to \mathbb{R}$ ist eine stückweise stetige Funktion, d.h. der Definitionsbereich der Funktion kann in endlich viele Teilintervalle unterteilt werden, in denen die Funktion stetig und beschränkt ist.

Bedingung 2) $y : [0,\infty) \to \mathbb{R}$ wächst nicht schneller als eine Exponentialfunktion $e^{\alpha x}$ mit geeignetem α, d.h. es gibt ein $X > 0$ und Konstanten $\alpha \geq 0, M > 0$, so dass $|y(x)| \leq M\, e^{\alpha x}$ für $x \geq X$. $y(x)$ wird dann von höchstens exponentiellem Wachstum der Ordnung α genannt.

Ist $Y(s)$ die Laplace-Transformierte einer stetigen Funktion, dann gibt es genau eine stetige Funktion $y(x)$ von höchstens exponentiellem Wachstum, so dass $\mathcal{L}(y) = Y(s)$, und $y(x)$ gegeben ist durch

$$y(x) = \mathcal{L}^{-1}(Y) := \frac{1}{2\pi i}\int_{\delta-i\infty}^{\delta+i\infty} Y(s)e^{sx}ds\ ,\ \delta = Re(s) > 0$$

In der Praxis wird die Rücktransformation jedoch nicht immer über die Berechnung des Integrals durchgeführt. Andere Hilfsmittel sind z.B. Tabellen, Computerprogrammen wie z.B. MAPLE oder das Ausnutzen der Korrespondenz von Originalfunktion $y(x)$ und Laplace-Transformierter $Y(s)$, was wir an späterer Stelle auch tun werden.

Wichtige Eigenschaften der Laplace-Transformation, die wir später brauchen werden und deren Beweise in [2] und [4] aufgeführt werden, sind

i) Linearität

Die Laplace-Transformation einer Summe ist gleich der Summe der Laplace-Transformationen, wobei konstante Faktoren vor das Laplace - Integral gezogen werden können.

ii) Laplace-Transformierte von $D^n y(x)$, $n \in \mathbb{N}$

$$\mathcal{L}(D^n y(x)) = s^n \mathcal{L}(y(x)) - \sum_{k=0}^{n-1} s^{n-k-1} y^{(k)}(0) = s^n \mathcal{L}(y(x)) - \sum_{k=0}^{n-1} s^k y^{(n-k-1)}(0)$$

iii) Laplace-Transformierte des Faltungsprodukts $(y_1 * y_2)(x)$

Unter dem Faltungsprodukt zweier Funktionen $y_1(x)$ und $y_2(x)$ versteht man

$$(y_1 * y_2)(x) = \int_0^x y_1(x-\xi) y_2(\xi) d\xi.$$

Sind $Y_1(s)$ und $Y_2(s)$ die Bildfunktionen von $y_1(x)$ und $y_2(x)$, so ist die Laplace-Transformierte des Faltungsproduktes $(y_1 * y_2)(x)$ gegeben durch

$$\mathcal{L}(y_1 * y_2)(s) = Y_1(s) Y_2(s).$$

In der Praxis geht man bei der Rücktransformation einer Bildfunktion häufig wie folgt vor: Man zerlegt $Y(s)$ in ein Produkt $Y(s) = Y_1(s)Y_2(s)$ von Bildfunktionen von denen die zugehörigen Originalfunktionen $y_1(x)$ und $y_2(x)$ bekannt sind. Die zu $Y(s)$ gehörende Originalfunktion ist dann $y(x) = (y_1 * y_2)(x)$.

Schaut man sich Def. 2.1.1 des Riemann-Liouville Integraloperators an, sieht man, dass uns diese Herangehensweise noch nützlich sein wird.

Beispiel: Die Laplace-Transformierten einer speziellen Mittag-Leffler Funktion. Es gilt

$$\mathcal{L}\left(t^{\alpha k+\beta-1}E^{(k)}_{\alpha,\beta}(\pm at^{\alpha})\right) = \frac{k!s^{\alpha-\beta}}{(s^{\alpha}\mp a)^{k+1}} \quad , \text{ mit } \quad E^{(k)}_{\alpha,\beta}(x) = \frac{d^k}{dx^k}E_{\alpha,\beta}(x).$$

Um obiges Beispiel plausibel zu machen, betrachten wir die in [5] ersichtliche Laplace-Transformierte

$$\mathcal{L}(\frac{t^{n-1}}{(n-1)!}e^{\pm\alpha t}) = \frac{1}{(s\mp\alpha)^n},$$

die sich durch Umstellung und Substitution von $k = n-1$ zu

$$\mathcal{L}(t^k e^{\pm\alpha t}) = \frac{k!}{(s\mp\alpha)^{k+1}}$$

ergibt. Obige Gleichung erhält man unter Verwendung der Reihendarstellung der Exponentialfunktion und Definition 2.0.2 der Γ-Funktion wie folgt.

$$\begin{aligned}\int_0^{\infty} e^{-t}e^{\pm zt}dt &= \int_0^{\infty} e^{-t}\sum_{k=0}^{\infty}\frac{(\pm zt)^k}{k!}dt \\ &= \sum_{k=0}^{\infty}\frac{(\pm z)^k}{k!}\int_0^{\infty}e^{-t}t^k dt = \sum_{k=0}^{\infty}(\pm z)^k = \frac{1}{1\mp z}\end{aligned}$$

k-maliges differenzieren nach z angewendet auf beide Seiten dieser Gleichung ergibt

$$\int_0^{\infty} e^{-t}t^k e^{\pm zt}dt = \frac{k!}{(1\mp z)^{k+1}},$$

woraus nach Substitution von $\pm z = 1-s\pm a$ die Laplace-Transformierte

$$\int_0^{\infty} e^{-st}t^k e^{\pm at}dt = \frac{k!}{(s\mp a)^{k+1}}$$

folgt. Analoges Vorgehen bezüglich der Funktion $t^{\beta-1}E_{\alpha,\beta}(\pm zt^{\alpha})$ ergibt obige Laplace-Transformierte dieser speziellen Mittag-Leffler Funktion.

Näheres dazu ist in [2] ersichtlich.

Beispiel: Die Differentialgleichung $ay'(x) - by(x) = f(x)$

Im ersten Schritt wird die Laplace-Transformierte auf die Differentialgleichung angewendet

$$\mathcal{L}(ay' - by) = \mathcal{L}(f)$$
$$a\mathcal{L}(y') - b\mathcal{L}(y) = \mathcal{L}(f)$$

Dann erfolgt im zweiten Schritt die Berechnung der Laplace-Transformierten der Ableitung links und die der Inhomogenität rechts.

$$a(s\mathcal{L}(y) - y_0) - b\mathcal{L}(y) = \mathcal{L}(f) = F(s)$$

Im nächsten Schritt löst man nach der Laplace-Transformierten $\mathcal{L}(y(x))$ auf und berechnet die Originalfunktion der entstandenen rechten Seite, wodurch man $y(x)$ erhält.

$$\mathcal{L}(y)(s) = \frac{F(s) + ay_0}{a(s+b)}$$
$$y(x) = \mathcal{L}^{-1}\left(\frac{F(s) + ay_0}{a(s+b)}\right)$$

Die hier einfach aussehende Rücktransformation kann allerdings schnell sehr kompliziert werden. Betrachtet man das vorherige Beispiel der Laplace-Transformierten einer speziellen Mittag-Leffler Funktion und bedenkt, dass diese Funktion die inverse Laplace-Transformierte der rechten Seite ist, so wird klar, dass selbst relativ einfache Bildfunktionen komplizierte Originalfunktionen haben können. Das Lösen einer Differentialgleichung beschränkt sich so meist auf das finden dieser Originalfunktion.

4.1.2.1 Riemann-Liouville Differentialgleichungen

Um nach obiger Methode fraktionale Differentialgleichungen lösen zu können, muss zuerst die Laplace-Transformierte der fraktionalen Ableitung bekannt sein. Dazu bestimmen wir zuerst die Laplace-Transformierte des Riemann-Liouville Integrals und wenden dann die Regel der Laplace- Transformierten einer Ableitung auf dieses Integral an.

Das Riemann-Liouville fraktionale Integral einer Funktion $f(x)$ kann als Faltung zweier Funktionen $f(x)$ und $g(x) = \frac{x^{n-1}}{\Gamma(n)}$, $n \in \mathbb{R}_+$, aufgefaßt werden.

$$I_0^n f(x) := \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t) dt = g(x) * f(x)$$

Es gilt $\mathcal{L}(x^{n-1}) = \Gamma(n) s^{-n}$ und somit ergibt sich die Laplace-Transformierte des Riemann-Liouville Integrals als Produkt der Laplace-Transformierten $G(s)F(s)$ zu

$$\mathcal{L}(I_0^n f(x)) = \mathcal{L}(g(x))\mathcal{L}(f(x)) = G(s)F(s) = s^{-n}F(s).$$

Sei $m = \lfloor n \rfloor + 1$. Für die Berechnung der Laplace-Transformierten der fraktionalen Riemann-Liouville Ableitung definieren wir uns eine neue Funktion

$$g(x) := I_0^{m-n} f(x) := \frac{1}{\Gamma(m-n)} \int_0^x (x-t)^{m-n-1} f(t) dt,$$

womit die m-te Ableitung von $g(x)$, also gemäß der Definition der Riemann-Liouville Ableitung $g^{(m)}(x) = \frac{d^m}{dx^m} g(x) = D_0^n f(x)$ ist. Die Laplace-Transformierte ergibt sich zu

$$\mathcal{L}(g^{(m)}(x)) = s^m G(s) - \sum_{k=0}^{m-1} s^k \left[g^{(m-k-1)}(x) \right]_{x=0}.$$

Die Laplace-Transformierte des Riemann-Liouville Integrals haben wir bereits berechnet und für die Ableitungen der Funktion $g(x)$ gilt

$$\begin{aligned} g^{(m-k-1)}(x) &= D_0^{n-k-1} f(x) = D^{m-k-1} I_0^{m-k-1-n+k+1} f(x) \\ &= D^{m-k-1} I_0^{m-n} f(x) = D_0^{n-k-1} f(x), \end{aligned}$$

woraus die Laplace-Transformierte der Riemann-Liouville Ableitung

$$\mathcal{L}(D_0^n f(x)) = s^n F(s) - \sum_{k=0}^{m-1} s^k \left[D_0^{n-k-1} f(x) \right]_{x=0}$$

folgt mit der symbolischen Schreibweise $D_0^{-\alpha} \equiv I_0^{\alpha}$, $\alpha \in \mathbb{R}_+$.

Berechnen wir nun noch einmal das Beispiel 4.1.1.2 mit der Methode der Laplace-Transformation, um beide Methoden vergleichen zu können.

4.1.2.2 Beispiel : $D_0^n y(x) = \lambda y(x)$, $0 < n < 1$ mit der Anfangsbedingung $\lim_{z \to 0_+} I_0^{1-n} y(z) = y_I$ und $m = \lceil n \rceil$.

Zu Beginn wird das Laplace-Integral auf die Gleichung angewendet, anschließend nach der Laplace-Transformierten von $y(x)$ aufgelöst und dann die Originalfunktion dieser Transformierten gesucht.

$$\begin{aligned} & \mathcal{L}(D_0^n y(x)) = \lambda \mathcal{L}(y(x)) \\ \Rightarrow\ & s^n \mathcal{L}(y(x)) - \lambda \mathcal{L}(y(x)) = s^0 y_I \\ \Rightarrow\ & \mathcal{L}(y(x)) = \frac{y_I}{s^n - \lambda} \\ \Rightarrow\ & y(x) = y_I t^{n-1} E_{n,n}(\lambda t^n) \end{aligned}$$

Die inverse Laplace-Transformierte wurde unter Verwendung des ersten Beispiels in 4.1.2.1 gefunden.

4.1.2.3 Beispiel : $D_0^n y(x) = x^2 + y(x)$, $1 < n < 2$, $m = \lceil n \rceil$ mit den Anfangsbedingungen $[I_0^{2-n} y(x)]_{x=0} = y_I$ und $[D_0^{n-1} y(x)]_{x=0} = y_D$.

Die Herangehensweise ist analog dem vorherigen Beispiel, lediglich bei der Laplace-Transformierten von D_0^n werden beide Anfangsbedingungen benötigt.

$$\begin{aligned} & \mathcal{L}(D_0^n y(x)) = \mathcal{L}(x^2) + \mathcal{L}(y(x)) \\ \Rightarrow\ & s^n \mathcal{L}(y(x)) - y_I - s y_D = 2s^{-3} + \mathcal{L}(y(x)) \\ \Rightarrow\ & s^n \mathcal{L}(y(x)) - \mathcal{L}(y(x)) = 2s^{-3} + s y_I + y_D \\ \Rightarrow\ & \mathcal{L}(y(x)) = \frac{1}{s^n - 1}(2s^{-3} + s y_I + y_D) \\ \Rightarrow\ & \mathcal{L}(y(x)) = \frac{2s^{-3}}{s^n - 1} + \frac{s y_I}{s^n - 1} + \frac{y_D}{s^n - 1}) \\ \Rightarrow\ & y(x) = 2t^{n+2} E_{n,n+3}(t^n) + y_I t^{n-2} E_{n,n-1}(t^n) + y_D t^{n-1} E_{n,n}(t^n) \end{aligned}$$

Fraktionale Differentialgleichungen mit mehreren Ableitungstermen oder solche mit nichtkonstanten Koeffizienten können nur sehr schwer mit der Methode der Laplace-Transformation gelöst werden, da sich die Rücktransformation in diesen Fällen komplizierter wenn nicht gar unmöglich zeigt. Näheres zu Gleichungen mit mehreren Ableitungstermen ist in [2] beschrieben. Bei Gleichungen mit nichtkonstanten Koeffizienten kommen u.a. Terme der Form $\frac{d}{ds}\mathcal{L}(y)(s)$ vor und die Gleichung kann nicht nach $\mathcal{L}(y(x))$ aufgelöst werden.

4.1.2.4 Caputo Differentialgleichungen

Um die Laplace-Transformierte der Caputo Ableitung zu bestimmen, benutzen wir Def. 2.2.2 und schreiben

$$D_{0_C}^n f(x) = D_0^{m-n} f^{(m)}(x).$$

Die Laplace-Transformierte der Riemann-Liouville Ableitung wurde bereits in Abschnitt 4.1.2.1 berechnet. Mit ihrer Hilfe und der Laplace- Transformierten einer ganzzahligen Ableitung einer Funktion ergibt sich

$$\begin{aligned}\mathcal{L}(D_{0_C}^n f(x)) &= s^{-(m-n)}\mathcal{L}(f^{(m)}(x)).\\ &= s^{-(m-n)}\left(s^m\mathcal{L}(f(x)) - \sum_{k=0}^{m-1} s^{m-k-1}\left[f^{(k)}(x)\right]_{x=0}\right)\\ &= s^n F(s) - \sum_{k=0}^{m-1} s^{n-k-1}\left[f^{(k)}(x)\right]_{x=0}\end{aligned}$$

Die nachfolgenden Beispiele berechnen sich analog der Beispiele 4.1.2.3 und 4.1.2.4, wobei lediglich die Anfangsbedingungen wie bei “klassischen“ Differentialgleichungen ganzzahlige Ableitungen enthalten.

4.1.2.5 Beispiel : $D_{0_C}^n y(x) = \lambda y(x)$, $0 < n < 1$, $y(0) = y_0$.

Es ergibt sich

$$\begin{aligned}&\mathcal{L}(D_{0_C}^n y(x)) = \lambda\mathcal{L}(y(x))\\ \Rightarrow\ & s^n\mathcal{L}(y(x)) - \lambda\mathcal{L}(y(x)) = s^{n-1}y_0\\ \Rightarrow\ & \mathcal{L}(y(x)) = \frac{y_0 s^{n-1}}{s^n - \lambda}\\ \Rightarrow\ & y(x) = y_0 E_{n,1}(\lambda t^n)\end{aligned}$$

4.1.2.6 Beispiel : $D_0^n y(x) = x^2 + y(x)$, $1 < n < 2$, $m = \lceil n \rceil$ mit den Anfangsbedingungen $y(0) = y_0$ und $y^{(1)}(x) = \frac{d}{dx} y(x) = y_1$.

Es ergibt sich

$$\begin{aligned}
&\mathcal{L}(D_{0_C}^n y(x)) = \mathcal{L}(x^2) + \mathcal{L}(y(x)) \\
\Rightarrow\ & s^n \mathcal{L}(y(x)) - s^{n-1} y_0 - s^{n-2} y_1 = 2s^{-3} + \mathcal{L}(y(x)) \\
\Rightarrow\ & s^n \mathcal{L}(y(x)) - \mathcal{L}(y(x)) = 2s^{-3} + s^{n-1} y_0 + s^{n-2} y_1 \\
\Rightarrow\ & \mathcal{L}(y(x)) = \frac{1}{s^n - 1}(2s^{-3} + s^{n-1} y_0 + s^{n-2} y_1) \\
\Rightarrow\ & \mathcal{L}(y(x)) = \frac{2s^{-3}}{s^n - 1} + \frac{s^{n-1} y_0}{s^n - 1} + \frac{s^{n-2} y_1}{s^n - 1}) \\
\Rightarrow\ & y(x) = 2t^{n+2} E_{n,n+3}(t^n) + y_0 E_{n,1}(t^n) + y_1 t E_{n,2}(t^n)
\end{aligned}$$

Alle Lösungen können durch Einsetzen in die entsprechenden Differentialgleichungen verifiziert werden.

4.2 Ein Numerisches Lösungsverfahren

Die im letzten Abschnitt beschriebenen analytischen Verfahren haben den Nachteil, dass sie nur für relativ einfache fraktionale Differentialgleichungen anwendbar sind. In diesem Abschnitt wird ein Verfahren vorgestellt, mit dessen Hilfe kompliziertere fraktionale Differentialgleichungen gelöst und graphisch dargestellt werden können.

Fraktionales Adams-Bashforth-Moulton Verfahren

Um die Idee dieser Methode zu verdeutlichen, soll hier zuerst das Prinzip der klassischen Variante des Adams-Bashforth-Moulton Verfahrens vorgestellt werden. Dabei handelt es sich um ein Prädiktor-Korrektor Verfahren, d.h. es wird mit einer ersten Integralapproximation ein vorläufiger nächster y-Wert berechnet und dieser Wert in einer zweiten Integralapproximation benutzt, um den endgültigen nächsten y-Wert zu erhalten. Äquidistante Schrittweiten vereinfachen das Verfahren erheblich und sollen hier vorausgesetzt werden.

Gegeben sei eine gewöhnliche Differentialgleichung

$$y'(x) = f(x, y(x))$$

mit der Anfangsbedingung

$$y(0) = y_0.$$

Durch Integration der obigen Gleichung erhält man die äquivalente Integralgleichung

$$y(x) = y_0 + \int_0^x f(t, y(t))dt,$$

und die Idee des Verfahrens ist nun, für eine gegebene Approximation $y_j \approx y(t_j), j = 1, 2, ..., k$ eine Approximation y_{k+1} durch

$$y(x_{k+1}) = y(x_k) + \int_{x_k}^{x_{k+1}} f(t, y(t))dt,$$

zu berechnen. Dabei kann das Integral auf der rechten Seite durch verschiedene Approximationen numerisch angenähert werden. Zum einen durch die sogenannte Rechteck - oder Euler-Formel

$$\int_a^b f(x)dx \approx hf(a) \ , \ h = (b - a),$$

welche das Integral, also die Fläche unter dem Graphen der Funktion f durch ein Rechteck annähert, zum anderen durch die Trapez-Formel

$$\int_a^b f(x)dx \approx \frac{h}{2}(f(a) + f(b),$$

welche das Integral durch ein Trapez annähert, wobei an dieser Stelle wichtig ist, dass in dieser Integralapproximation der Funktionswert $f(b)$ benötigt wird. Das Adams-Bashforth-Moulton Verfahren berechent nun aus einem bekannten y_k-Wert durch die Rechteck-Formel den Prädiktor, d.h. den nächsten y_{k+1}-Wert, und benutzt dann diese beiden Werte, um mittels Trapez-Formel den endgültigen y_{k+1}-Wert zu erhalten. Korrektor und Prädiktor ergeben sich somit wie folgt aus

$$y_{k+1} = y_k + \frac{h}{2}(f(x_k, y(x_k)) + f(x_{k+1}, y^P(x_{k+1})))$$

mit

$$y^P_{k+1} = y^P(x_{k+1}) = y_k + hf(x_k, y(x_k)).$$

4.2.1 Riemann-Liouville Differentialgleichungen

Gegeben sei eine Riemann-Liouville Differentialgleichung

$$D_0^n y(x) = f(x, y(x))$$

mit $n > 0, n \notin \mathbb{N}$ und $m = \lceil n \rceil$, ausgestattet mit den Anfangsbedingungen

$$D_0^{n-j} y(0) = c_j, \quad (j = 1, ..., m-1), \quad \lim_{z \to 0_+} I_0^{m-n} y(z) = c_m$$

dargestellt in der äquivalenten Integralgleichung

$$y(x) = \sum_{k=1}^{m} \frac{c_k x^{n-k}}{\Gamma(n-k+1)} + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t, y(t)) dt.$$

Obwohl das fraktionale Verfahren analog dem klassischen Verfahren verläuft, muss die Integralapproximation auf Grund des komplizierteren Integranden aufwendiger berechnet werden. Das fraktionale Integral der Funktion $f(x, y(x))$ besitzt durch den Faktor $(x-t)^{n-1}$ eine Art Gedächtnis, weil durch Einsetzen von $t \in [0, x]$ durch $(x-t)$ auf der Abszisse nach links gegangen wird. Diesem Sachverhalt wird in der Integralapproximation dadurch Rechnung getragen, dass nicht ein Rechteck bzw. Trapez, sondern eine Art Summe von Rechtecken bzw. Summe von Trapezen die Approximation liefern. Diese Gewichte ergeben sich für den Prädiktor durch

$$b_{j,k+1} = \frac{h^n}{n}((k+1-j)^n - (k-j)^n)$$

und für den Korrektor durch

$$a_{j,k+1} = \begin{cases} \frac{h^n}{n(n+1)}(k^{n+1} - (k-n)(k+1)^n) & , j = 0 \\ \frac{h^n}{n(n+1)}((k-j+2)^{n+1} + (k-j)^{n+1} - 2(k-j+1)^{n+1}) & , 1 \leq j \leq k \\ \frac{h^n}{n(n+1)} & , j = k+1 \end{cases}$$

wodurch sich insgesammt das Prädiktor-Korrektor Verfahren durch

$$y_{k+1} = \sum_{j=0}^{m-1} \frac{c_{j+1} x_{k+1}^{n-j-1}}{\Gamma(n-j)} + \frac{1}{\Gamma(n)} \left(\sum_{j=0}^{k} \left(a_{j,k+1} f(x_j, y_j) \right) + a_{k+1,k+1} f(x_{k+1}, y_{k+1}^P) \right)$$

mit

$$y_{k+1}^P = \sum_{j=0}^{m-1} \frac{c_{j+1} x_{k+1}^{n-j-1}}{\Gamma(n-j)} + \frac{1}{\Gamma(n)} \left(\sum_{j=0}^{k} \left(b_{j,k+1} f(x_j, y_j) \right) \right)$$

ergibt und nach [7] der Startwert $y_0 := 0$ für $y(0) = \infty$ bzw. $y_0 := y(0)$ für $y(0) < \infty$ gesetzt wird. Beim Adams-Bashforth-Moulton Verfahren für Riemann-Liouville Differentialgleichungen werden somit $m+1$ Anfangsbedingungen benötigt, da zu den "normalen" Anfangsbedingungen noch ein Startwert hinzukommt.

4.2.2 Beispiel : $D_0^n y(x) = \lambda y(x)$, $0 < n < 1$, $\lim\limits_{z \to 0_+} I_0^{m-n} y(z) = y_I$ und $m = \lceil n \rceil$.

Für die im Vorangegangenen schon mehrfach gelöste Differentialgleichung soll hier das Iterationsverfahren praktisch angewendet gezeigt werden. Wie aus der analytischen Lösung bekannt ist, divergiert die Lösung $y(x)$ für $x \to 0$, womit als Startwert $y_0 := 0$ benutzt wird und sich

$$y_{k+1} = \frac{y_I x_{k+1}^{n-1}}{\Gamma(n)} + \frac{1}{\Gamma(n)} \left(\sum_{j=0}^{k} \left(a_{j,k+1} f(x_j, y_j) \right) + a_{k+1,k+1} f(x_{k+1}, y_{k+1}^P) \right)$$

mit

$$y_{k+1}^P = \frac{y_I x_{k+1}^{n-j}}{\Gamma(n-j+1)} + \frac{1}{\Gamma(n)} \left(\sum_{j=0}^{k} \left(b_{j,k+1} f(x_j, y_j) \right) \right)$$

ergibt.

Wertetabelle für $n = 0.4$, $\lambda = 1$, $y_I = 1$ und Schrittweite $h = 0.01$ und Ausgabe jedes 20. Wertes :

x_k	y_k	$y(x_k)$	$\lvert y(x_k) - y_k \rvert$
0.00	0.0000000	∞	∞
0.20	3.3654677	3.9201591	0.5546915
0.40	3.6054988	4.2009368	0.5954380
0.60	4.1830702	4.8755399	0.6924697
0.80	4.9777536	5.8037138	0.8259602
1.00	5.9896005	6.9857356	0.9961351
1.20	7.2483194	8.4565053	1.2081860
1.40	8.7997934	10.2698336	1.4700402
1.60	10.7040057	12.4960608	1.7920552
1.80	13.0361018	15.2233204	2.1872186
2.00	15.8888588	18.5604454	2.6715866

Eine graphische Darstellung sowie weitere Beispiele werden in Kapitel 5 über fraktionale Differentialgleichungen mit MAPLE gegeben.

4.2.3 Caputo Differentialgleichungen

Das fraktionale Adams-Bashforth-Moulton Verfahren ist auch auf Differentialgleichungen anwendbar, in denen der Caputo Differentialoperator benutzt wurde. Das Verfahren der Integralapproximation des Integrals über der rechten Seite der Differentialgleichung läuft analog dem für Riemann-Liouville Differentialgleichungen, weil beide Ableitungsdefinitionen den Riemann-Liouville Integraloperator benutzen. Wie in den vorangegangenen Abschnitten sind auch hier lediglich die Anfangsbedingungen von anschaulicherer Form, d.h. genau so, wie wir sie von klassischen Differentialgleichungen gewohnt sind. Das hat zur Folge, dass, anders als im Riemann-Liouville Fall, kein zusätzlicher Startwert y_0 benötigt wird.

Gegeben sei die Caputo Differentialgleichung

$$D^n_{0_C} y(x) = f(x, y(x))$$

mit $n > 0$ und $m = \lceil n \rceil$, ausgestattet mit den Anfangsbedingungen

$$D^j y(0) = y_0^{(j)}, \quad (j = 0, 1, ..., m-1)$$

und dargestellt in der äquivalenten Integralgleichung

$$y(x) = \sum_{k=0}^{m-1} \frac{D^k y(0)}{n!} x^k + \frac{1}{\Gamma(n)} \int_0^x (x-t)^{n-1} f(t, y(t)) dt.$$

Prädiktor, Prädiktorgewichte, Korrektor und Korrektorgewichte ergeben sich wie folgt :

$$y^P_{k+1} = \sum_{j=0}^{m-1} \frac{x^j_{k+1}}{j!} y_0^{(j)} + \frac{1}{\Gamma(n)} \sum_{j=0}^{k} b_{j,k+1} f(x_j, y_j)$$

mit

$$b_{j,k+1} = \frac{h^n}{n}((k+1-j)^n - (k-j)^n)$$

und

$$y_{k+1} = \sum_{j=0}^{m-1} \frac{x^j_{k+1}}{j!} y_0^{(j)} + \frac{1}{\Gamma(n)} \sum_{j=0}^{k} \left(a_{j,k+1} f(x_j, y_j) + a_{k+1,k+1} f(x_{k+1}, y^P_{k+1}) \right)$$

mit

$$a_{j,k+1} = \begin{cases} \frac{h^n}{n(n+1)}(k^{n+1} - (k-n)(k+1)^n) & , j = 0 \\ \frac{h^n}{n(n+1)}((k-j+2)^{n+1} + (k-j)^{n+1} - 2(k-j+1)^{n+1}) & , 1 \leq j \leq k \\ \frac{h^n}{n(n+1)} & , j = k+1 \end{cases}$$

4.2.4 Beispiel : $D^n_{0_C} y(x) = \lambda y(x)$, $0 < n < 1$, $y(0) = y_0$.

Das Adams-Bashforth-Moulton Verfahren ergibt sich wie folgt aus

$$y_{k+1} = y_0 + \frac{1}{\Gamma(n)} \sum_{j=0}^{k} \left(a_{j,k+1} f(x_j, y_j) + a_{k+1,k+1} f(x_{k+1}, y^P_{k+1}) \right)$$

mit

$$y^P_{k+1} = y_0 + \frac{1}{\Gamma(n)} \sum_{j=0}^{k} b_{j,k+1} f(x_j, y_j).$$

Wertetabelle für $n = 0.4$, $\lambda = 1$, $y_I = 1$ und Schrittweite $h = 0.01$ und Ausgabe jedes 20. Wertes :

x_k	y_k	$y(x_k)$	$\lvert y(x_k) - y_k \rvert$
0.00	1.0000000	1.0000000	0.0000000
0.20	2.0983941	2.1036192	0.0052251
0.40	2.8982953	2.9056662	0.0073710
0.60	3.7983778	3.8087316	0.0103538
0.80	4.8582585	4.8725845	0.0143260
1.00	6.1275192	6.1470751	0.0195559
1.20	7.6596891	7.6860890	0.0263999
1.40	9.5171653	9.5524785	0.0353132
1.60	11.7746340	11.8215099	0.0468759
1.80	14.5224360	14.5842593	0.0618233
2.00	17.8703538	17.9514410	0.0810871

Eine graphische Darstellung sowie weitere Beispiele werden in Kapitel 5 über fraktionale Differentialgleichungen mit MAPLE gegeben.

4.3 Vergleich zwischen Caputo - und Riemann-Liouville Differentialgleichungen

Ziel dieses Abschnitts soll es sein, die wesentlichen Unterschiede zwischen Lösungen fraktionaler Differentialgleichungen mit Riemann-Liouville Differentialoperator und solcher mit Caputo Differentialoperator aufzuzeigen. Unterschiede zwischen den Ableitungsoperatoren sind bereits in Abschnitt 2.3 besprochen worden.

4.3.1 Verhalten der Lösung für $x \to a$

Betrachtet man sich die Laplace-Transformierte der n-ten Riemann-Liouville Ableitung

$$\mathcal{L}(D_a^n f(x)) = s^n F(s) - \sum_{k=0}^{m-1} s^k \left[D_a^{n-k-1} f(x) \right]_{x=a},$$

so enthält diese für $k = 0$ immer einen Summanden der Form

$$\lim_{x \to a+} I_a^{1-n} f(x) = \frac{1}{\Gamma(1-n)} \int_a^\tau (x-\tau)^{-n} f(\tau)\, d\tau,$$

welcher im Integranden den Term $\frac{1}{(x-\tau)^n}$ enthält. Dies bedeutet, dass unter den in Bemerkung 2.3.3 genannten Bedingungen die Lösung der Riemann-Liouville Differentialgleichung am Entwicklungspunkt a divergiert, weil dann die Rücktransformierten Terme der Form x^{n-w} mit $w > n$ enthalten.

Die Laplace-Transformierte der Caputo Ableitung enthält zwar ebenfalls Terme der Form s^{n-k-1} mit $k = 0, 1, ..., \lfloor n \rfloor$, allerdings werden hier durch die Rücktransformation keine negativen Exponenten der unabhängigen Variablen x erhalten. Und die Anfangsbedingung $y(0) < \infty$ bestimmt für $x \to 0$ das Verhalten der Lösung.

4.3.2 Unterschiedliche Anfangsbedingungen

4.3.2.1 Physikalische Interpretation

Ein weiterer wichtiger Unterschied ist die Form der Anfangsbedingungen. Im Falle von Caputo Differentialgleichungen haben wir, analog dem klassischen Fall, Anfangsbedingungen der Form $D^n y(x) = y_n$ für $n = 0, 1, ...$

gegeben, für die die physikalischen Interpretationen von Ort, Geschwindigkeit, Beschleunigung, und für höhere Ordnungen generalisiert die Änderung der vorherigen Größe, ihre Gültigkeit in unzähligen Versuchen bestätigt haben.

Die mathematisch notwendige Formulierung der Anfangsbedingungen für Riemann-Liouville Differentialgleichungen beinhaltet allerdings keine solche physikalische Interpretation, so dass Lösungen solcher Differentialgleichungen nur bedingt Aussagen über reale Prozesse erlauben. Näheres dazu siehe [2].

4.3.2.2 Unterschiedliche Lösungen

Will man Riemann-Liouville - und Caputo Lösunge direkt vergleichen, so ist zu bedenken, dass sich Lösungen $y_{RL}(x)$ und $y_C(x)$ für kleine x-Werte evtl. unterscheiden, da die unterschiedlichen Anfangsbedingungen erst für größere x-Werte vom Verhalten der Funktion $y(x)$ überlagert werden und sich dann erst Riemann-Liouville - und Caputo Lösungen angleichen. Bei der Umformulierung einer Riemann-Liouville Differentialgleichung in eine Caputo Differentialgleichung müssen folglich die Anfangsbedingungen der Caputo Differentialgleichung an die Lösung der Riemann- Liouville Differentialgleichung angepasst werden. Divergiert die Riemann-Liouville Lösung z.B. für $x \to 0$ gegen $+\infty$, so muss die Anfangsbedingung der Caputo Differentialgleichung $D^1 y(0) \leq 0$ sein, um einen ähnlichen Funktionsverlauf zu erreichen.

5. Fraktionale Differentialgleichungen mit MAPLE

5.1 Picard-Iteration

In diesem Kapitel sollen die in Kapitel 4 vorgestellen Lösungsmethoden mit MAPLE realisiert werden. Da MAPLE die Iterationen im Picard-Verfahren nicht unendlich oft durchlaufen kann und auch nicht in der Lage ist, Umformungen zur Vereinfachung von Termen wie wir sie in Kapitel 4.1.1 vorgenommen haben, durchzuführen, ist dieses Verfahren weder numerisch noch analytisch mit MAPLE zu verwenden. Bereits nach wenigen Iterationen sind die notwendigen Integralberechnungen selbst für einfache Differentialgleichungen mit MAPLE unmöglich.

MAPLE kann in diesem Lösungsverfahren aber dennoch zur Vereinfachung von Termen verwendet werden. Der MAPLE-Befehl

```
simplify(Ausdruck);
```

ist in der Lage `Ausdruck` auf bekannte einfachere Formen zu reduzieren. Die in Beispiel 4.1.1.3 auftretenden Koeffizienten

```
a_k = product(j*n+j+2,j=0..k-1)/GAMMA(k*n+k+2);
b_k = product(j*n+j-1,j=0..k-1)/GAMMA((k+1)*n+k);
```

können z.B. mit

```
simplify(a_k);
simplify(b_k);
```

$$\frac{2\,\frac{\left(\frac{4}{3}\right)^k \Gamma\left(k+\frac{3}{2}\right)}{\sqrt{\pi}\,\Gamma\left(\frac{4}{3}k+2\right)}}{-\frac{3}{8}\,\frac{\left(\frac{4}{3}\right)^k \Gamma\left(k-\frac{3}{4}\right)\sqrt{2}\,\Gamma\left(\frac{3}{4}\right)}{\pi\,\Gamma\left(\frac{4}{3}k+\frac{1}{3}\right)}}$$

mit MAPLE vereinfacht werden.

5.2 Laplace-Transformation

Um fraktionale Differentialgleichungen mit Hilfe der Laplace-Transformation zu lösen sind mehrere Schritte nötig. Es müssen die Laplace-Transformierte der fraktionalen Ableitung und die der rechten Seite der Differentialgleichung bestimmt werden, dann muss nach der Laplace-Transformierten aufgelöst werden, und schließlich muss die Rücktransformation der neu entstandenen rechten Seite durchgeführt werden.

Hat man die Laplace-Transformierte der fraktionalen Ableitung bestimmt, so kann MAPLE dazu benutzt werden die Lapace-Transformierte der rechten Seite der Differentialgleichung zu finden, da es von einer Vielzahl von Funktionen diese berechnen kann. Notwendig dazu ist allerdings das Nachladen eines zusätzlichen Programmpaketes `inttrans` mit dem Befehl

```
with(inttrans);.
```

Nun stehen u.a. die beiden Befehle `laplace(Ausdruck,var_1,var_2)` und `invlaplace(Ausdruck,var_1,var_2` zur Verfügung, die `Ausdruck` in der ersten Variablen `var_1` in die Laplace-Transformierte der Variablen `var_2` überführen. Folgendes Beispiel kann nach Ausführung des obigen Befehls zum Nachladen des erforderlichen Paketes durchgeführt werden :

```
laplace(a*x,x,s);
```

$$\frac{a}{s^2}$$

Die Rücktransformation erfolgt über den inversen Befehl

```
invlaplace(%,s,x);
```

$$a\,x$$

wobei hier der sogenannte Ditto-Operator % verwendet wurde, mit dessen Hilfe auf das zuletzt berechnete Ergebnis zurückgegriffen wird. Der Befehl

```
invlaplace(a/s^2,s,x);
```

hätte die selbe Wirkung.

5.3 Graphische Ausgabe analytischer Lösungen in zwei Dimensionen

Da MAPLE über starke graphische Werkzeuge verfügt, wollen wir nun die berechneten Lösungen in zwei Dimensionen darstellen. Eine dreidimensionale Darstellung wird in Abschnitt 5.6 gezeigt.

Die für die exakten Lösungen häufig gebrauchte Mittag-Leffler Funktion definiere ich vorab, da sie in den MAPLE eigenen Funktionsbibliotheken nicht enthalten ist. Einen Überblick aller enthaltenen Funktionen bekommt man durch Eingabe des Befehls

```
?inifcn;
```

Die Eingabe der Mittag-Leffler Funktion erfolgt mittels Prozedur. Dabei werden dieser Prozedur drei Parameter übergeben. Im Prozedurrumpf wird eine Summe gemäß der Definition der Mittag-Leffler Funktion definiert, wobei die einfachen Anführungszeichen in der Summe eine eventuelle Doppelbelegung der Variablen `j` verhindern, d.h. selbst wenn die Variable `j` außerhalb der Summe bereits definiert ist, so beeinflussen sich diese beiden Variablen gleichen Namens nicht, weil die Variable in der Summe nur in dieser Summe lokal existiert. Auf die Summe wurde der `evalf`-Befehl angewendet, um den Rückgabewert der Summe in eine Fließkommazahl umzuwandeln, falls die Summe Gleitkommazahl ist. Dabei steht `evalf` für `evaluate to a float`.

```
ML:=proc(alpha,beta,X)
  evalf(sum('X^j/(GAMMA(alpha*j+beta))','j'=0..infinity));
end:
```

Wichtig an dieser Stelle ist, da MAPLE natürlich nicht unendlich viele Summanden auswerten kann, dass intern die Summe auf Konvergenz geprüft wird und dann versucht wird eine Funktion zu finden, die diese Summendarstellung besitzt. Schlägt dies fehl, so wird die Summe unausgewertet in symbolischer Schreibweise zurückgegeben. Wie der `plot`-Befehl damit umgeht ist in Beispiel 5.3.3 ersichtlich.

Zur Kontrolle kann gemäß Def. 2.0.4 mittels

```
ML(1,1,x)
```

$$e^{x}$$

die Exponentialfunktion berechnet werden.

Die analytisch berechnete Lösung kann dann mittels der Prozedur `ML(var_1,var_2,var_3)` und dem bereits bekannten `plot`-Befehl graphisch ausgegeben werden.

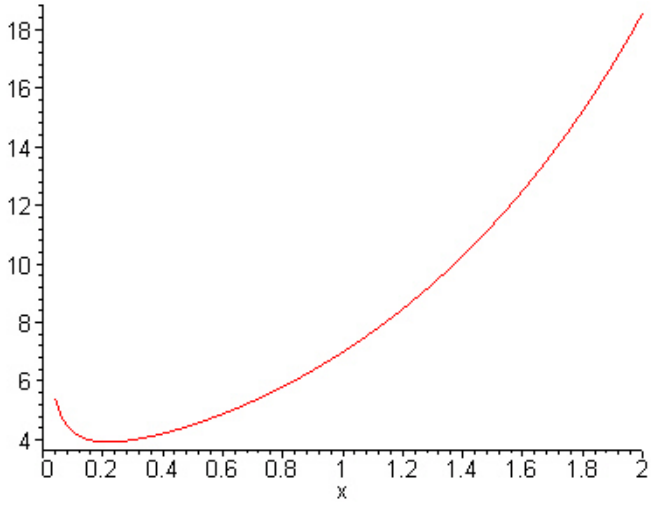
5.3.1 Beispiel: $D_0^n y(x) = ay(x)$ mit $0 < n < 1$, $a \in \mathbb{R}$, $[I_0^{1-n} y(x)]_{x=0} = y_I$

Mittels Laplace-Transformation und Rücktransformation ergibt sich folgende Lösung, die hier als MAPLE-Prozedur programmiert ist.

```
y_RL_D_1_exakt:=proc(X,n,a,ab)
  evalf(ab*X^(n-1)*ML(n,n,a*X^n));
end:
```

Der Prozedur werden die unabhängige Variable `X` , die Ordnung `n` der Differentialgleichung, der Koeffizient `a` und die Anfangsbedingung `ab` übergeben, gemäß der exakten Lösung berechnet und der Rückgabewert als Fließkommazahl ausgegeben. Nun kann mit dem üblichen `plot`-Befehl die graphische Ausgabe erfolgen.

```
plot(y_RL_D_1_exakt('x',0.4,1,1),x=0..2);
```



Ersichtlich wird hier die für Lösungen von Riemann-Liouville Differentialgleichungen typische Divergenz am Ursprung. Mittels der eben beschriebenen Methode können natürlich auch Lösungen von Caputo Differentialgleichungen graphisch dargestellt werden. Dazu berechnet man die exakte Lösung gemäß der in Kapitel 4 vorgestellten Methoden, programmiert die Lösung als MAPLE-Prozedur und gibt diese Prozedur als Graphik durch den `plot`-Befehl aus. So lassen sich beide Lösungen genau wie in Kapitel 3 die Ableitungen gut vergleichen.

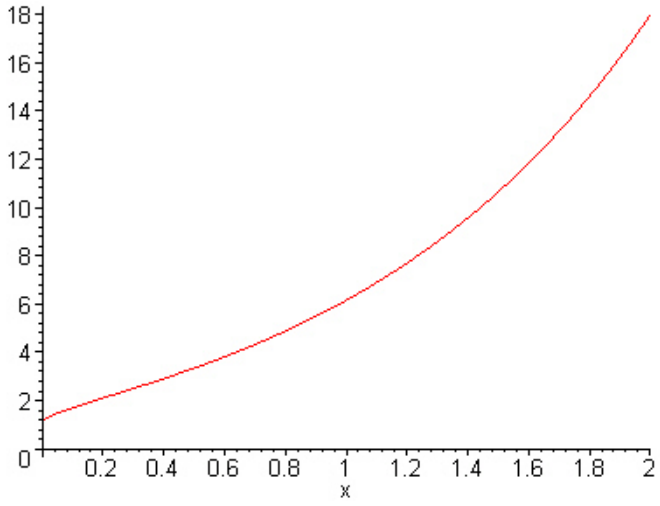
5.3.2 Beispiel: $D^n_{0_C} y(x) = ay(x)$ mit $0 < n < 1$, $a \in \mathbb{R}$ und $y(0) = y_0$

Die exakte Lösung als MAPLE-Prozedur, wobei hier für $x = 0$ eine weitere Zeile Programmcode notwendig wird, weil MAPLE sonst in der `ML`-Prozedur durch 0^0 eine Fehlermeldung zurückgibt. Die Korrektur bewirkt lediglich, dass die Anfangsbedingung nicht verletzt wird.

```
y_C_D_1_exakt:=proc(X,n,a,ab)
  if X=0 then RETURN(ab) fi;
  evalf(ab*ML(n,1,a*X^n));
end:
```

Der anschließende Aufruf der `plot`-Befehls bewirkt die Ausgabe.

```
plot(y_C_D_1_exakt('x',0.4,1,1),x=0..2);
```



Deutlich kann man hier den in Kapitel 4 theoretisch erarbeiteten Unterschied der Lösungen sehen.

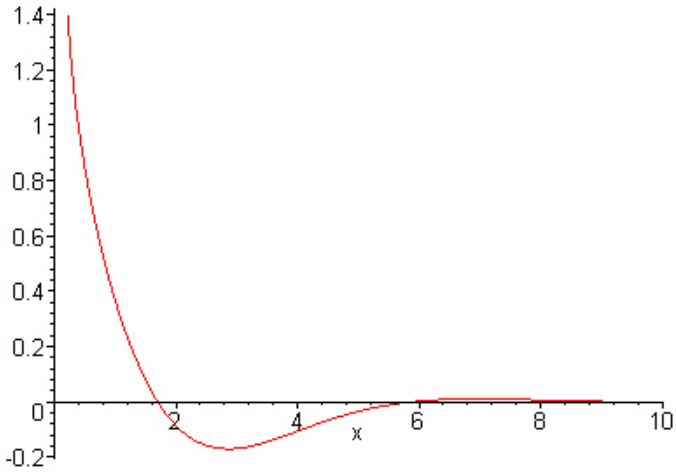
5.3.3 Beispiel : $D_0^n y(x) = ay(x)$, $1 < n < 2$, $m = \lceil n \rceil$ mit den Anfangsbedingungen $[I_0^{2-n} y(x)]_{x=0} = y_I$ und $[D_0^{n-1} y(x)]_{x=0} = y_D$.

Berechnet wurde folgende exakte Lösung, wobei nun zwei Anfangsbedingungen an die Prozedur übergeben werden müssen :

```
y_RL_D_2_exakt:=proc(X,n,a,ab_D,ab_I)
  evalf(ab_I*X^(n-1)*ML(n,n,a*X^n) + ab_D*X^(n-2)*ML(n,n-1,a*X^n));
end:
```

Die Ausgabe mittels `plot`-Befehl :

```
plot(y_RL_D_2_exakt('x',1.4,-1,1,1),x=0..2);
```



Wieder divergiert die Lösung für kleine x-Werte. Auffällig ist auch, dass MAPLE ab etwa $x = 9.5$ nicht mehr in der Lage ist, die Mittag-Leffler Funktion auszuwerten. Die der `plot`-Funktion symbolisch zurückgegebene Ausdruck für die Summe, führt jedoch nicht zu einer Fehlermeldung, da die `plot`-Funktion bereits korrekte Daten erhalten hat. Die `plot`-Funktion gibt nur dann eine Fehlermeldung zurück, wenn alle ihr übergebenden Daten nicht verarbeitet werden konnten.

Nun wieder der Vergleich mit der Caputo Lösung des analogen Problems :

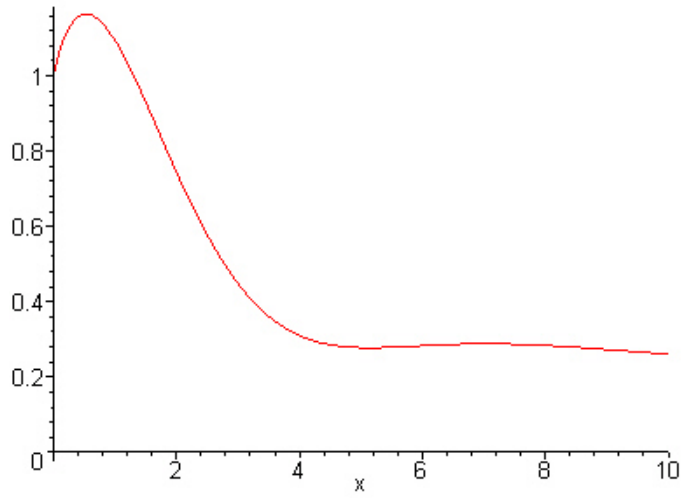
5.3.4 Beispiel: ${}_{0C}^{n}D\,y(x) = ay(x)$ mit $1 < n < 2$, $a \in \mathbb{R}$, $y(0) = y_0$ und $y'(0) = y_1$

Die exakte Lösung :

```
y_C_D_2_exakt:=proc(X,n,a,ab_0,ab_1)
  if X=0 then RETURN(ab_0) fi;
  evalf(ab_0*ML(n,1,a*X^n) + ab_1*X*ML(n,2,a*X^n));
end:
```

Die Ausgabe mittels `plot`-Befehl :

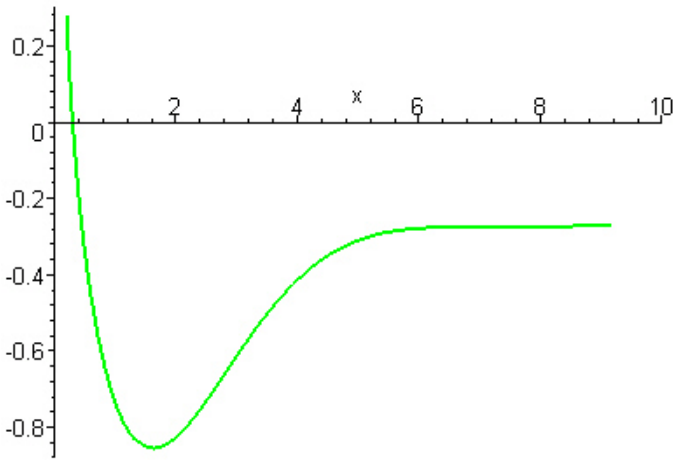
```
plot(y_C_D_2_exakt('x',1.4,-1,1,1),x=0..10);
```



Und hier wird ein ebenfalls bereits in Abschnitt 4.3 besprochener Unterschied zwischen Riemann-Liouville Lösungen und Caputo Lösungen deutlich. Obwohl die Gleichungen bis auf den Ableitungsoperator identisch sind, haben die Anfangsbedingungen, auch wenn diese alle gleich gewählt werden, eine gänzlich andere Wirkungsweise. Die Riemann-Liouville Lösung hat für kleine x-Werte eine negative Steigung trotz positiver Anfangsbedingungen, hingegen die Anfangsbedingung der Caputo Differentialgleichung von der Lösung dort eine positive Steigung verlangt.

Um den Unterschied zwischen den letzten beiden Beispielen, also den Unterschied zwischen der RL-Lösung und der C-Lösung zu verdeutlichen, können mit MAPLE beide Graphen in einem Diagramm dargestellt werden.

```
plot(y_RL_D_2_exakt('x',1.4,-1,1,1) -
y_C_D_2_exakt('x',1.4,-1,1,1),x=0..10,color=green,thickness=2);
```



5.4 Graphische Ausgabe numerischer Lösungen in zwei Dimensionen mittels fraktionalem Adams-Bashforth-Moulton Verfahren

Auf Grund der schnell sehr komplizierten Verfahren zur analytischen Lösung fraktionaler Differentialgleichungen, bieten numerische Verfahren eine mächtige Alternative, selbst komplizierte fraktionale Differentialgleichungen relativ genau lösen zu können.

In diesem Kapitel soll eine Prozedur vorgestellt werden, die gemäß dem fraktionalen Adams-Bashforth-Moulton Verfahren fraktionale Differentialgleichungen numerisch lösen kann.

Der Prozedur werden die numerischen Startwerte `X0` und `Y0` , die Schrittweite `hh` , das Ende `XX` des Berechnungsintervalls, die rechte Seite `f` der Differentialgleichung, die Anfangsbedingungen `c` als Array und die Ordnung `nn` der Differentialgleichung übergeben. Da die Prozedur nicht die graphische Ausgabe realisiert, sondern nur die später zu plottenden Punktepaare $(x_k, y(x_k))$ berechnet, werden zu Beginn der Prozedur die Variablen `x` und `y` global definiert, damit sie für andere Prozeduren zugänglich sind. Aus dem selben Grund sind auch die Ordnung der Differentialgleichung und die Schrittweite global definiert worden (siehe dazu 5.6).

5.4.1 ABM-Verfahren für Riemann-Liouville Differentialgleichungen

```
ABM_RL:=proc(X0::nonneg,Y0::nonneg,hh::nonneg, \
           XX::numeric,f::procedure,c,nn)
  global x,y,b,n,h;

  n:=nn;
  h:=hh;
  N:=XX/h;
  m:=ceil(n);
  x.0:=X0;
  y.0:=Y0;
  Y.0:=Y0;

  for j from 1 to nops(c) do
   C[j]:=op(j,c);
  od;

  for k from 0 to N do
   x.(k+1):=(k+1)*h;
   anfbed := sum('C[j+1]*(x.(k+1))^(n-j-1) \
           /GAMMA(n-j)','j'=0..m-1);

  # Berechnung des Prädiktors
     for j from 0 to k do
      b[j,k+1]:=h^n/n*((k+1-j)^n-(k-j)^n);
     od;

     Y.(k+1):=evalf( anfbed + 1/GAMMA(n)* \
           sum('b[j,k+1]*f(x.j,y.j)','j'=0..k));

  # Berechnung des Korrektors
     for j from 0 to k+1 do
      if j = 0 then
       a[j,k+1]:=h^n/(n*(n+1))*(k^(n+1)-(k-n)*(k+1)^n);
     elif j = k+1 then
      a[j,k+1]:=h^n/(n*(n+1));
```

```
    else
     a[j,k+1]:=h^n/(n*(n+1))*((k-j+2)^(n+1)+(k-j)^(n+1)- \
          2*(k-j+1)^(n+1));
     fi;
    od;

  y.(k+1):=evalf( anfbed + 1/GAMMA(n)* \
           (sum('a[j,k+1]*f(x.j,y.j)','j'=0..k) + \
  a[k+1,k+1]*f(x.(k+1),Y.(k+1))));

  od;
  print(ok);
end:
```

Die Prozedur wird wie folgt durchlaufen :

Zuerst werden Schrittweite `hh` und Ordnung `nn` in neue Variablen übergeben, damit diese global definiert werden können. Dann wird die Anzahl der Iterationen berechnet und die Startwerte in die Variablen `y.0` und `Y.0` geschrieben, wobei in `y.0` der Korrektor und in `Y.0` der Prädiktor gespeichert werden wird. Als nächstes folgt das Auslesen der in der Variablen `c` gespeicherten Anfangsbedingungen in ein Array mit Hilfe der Befehle `nops(var_1)` und `op(var_1,var_2` von denen erster die Länge von `var_1` bestimmt und letzterer den Operand mit der Nummer `var_1` im Ausdruck `var_2` zurückgibt.

Die Berechnungen der Anfangsbedingungen `anfbed`, des Prädiktors `Y.k` und des Korrektors `y.k` erfolgt gemäß dem in Kapitel 4.2.1 vorgestellten Verfahren. Dabei werden nach jeder Berechnung von Prädiktor und Korrektor die Ergebnisse als Fließkommazahl in `Y.k` und `y.k` gespeichert. Nach erfolgreichem Durchlauf der Prozedur wird ein `ok` zurückgegeben. Die berechneten Werte sind nun in den global definierten Variablen `x.k` und `y.k` gespeichert.

5.4.2 ABM-Verfahren für Caputo Differentialgleichungen

Da das ABM-Verfahren das Integral über der rechten Seite der Differentialgleichung approximiert besteht der einzige Unterschied in den Anfangsbedingungen der fraktionalen Ableitung auf der linken Seite der Differentialgleichung. Um folglich eine ABM-Prozedur für Caputo Differentialgleichungen

```
ABM_C:=proc(X0::nonneg,Y0::nonneg,hh::nonneg, \
            XX::numeric,f::procedure,c,nn)
```

zu schreiben, muss man den Programmcode der ABM-Prozedur für Riemann-Liouville Differentialgleichungen übernehmen und lediglich die Anfangsbedingungen

```
 anfbed := sum('C[j+1]*(x.(k+1))^(j)/(j!)','j'=0..m-1);
```

ersetzen.

5.4.3 Beispiele

Hat man nun die Anfangsbedingungen, die rechte Seite und die Ordnung der Differentialgleichung definiert und die Prozedur vom Startwert `X0` an bis zu `XX` durchlaufen lassen, so kann man sich, nachdem man das für `pointplot` erforderliche Paket

```
with(plots):
```

geladen hat, mittels

```
points:= { seq([x.k,y.k],k=0..XX) }:
pointplot(points);
```

den numerisch berechneten Graphen ausgeben lassen.

5.4.3.1 Beispiel: $D_0^n y(x) = ay(x)$ mit $n = 0.4$, $a = 1$, $[I_0^{1-n} y(x)]_{x=0} = y_I = 1$

Daraus ergeben sich die rechte Seite und die Anfangsbedingung

```
f:=(x,y)->y;
c:=1:
```

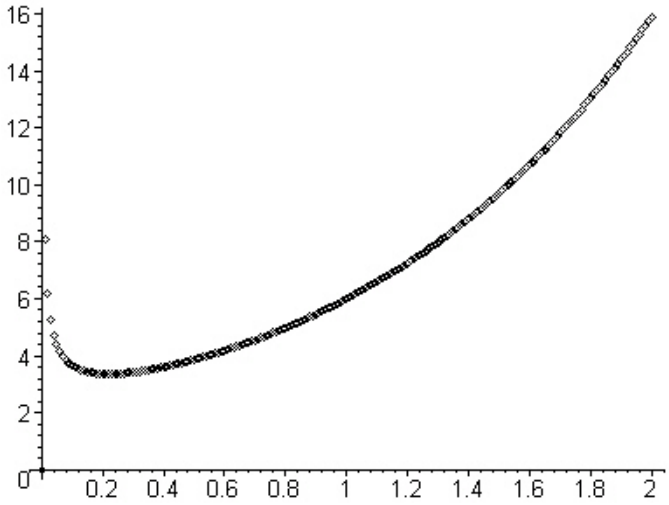
und mittels des Prozeduraufrufs

```
ABM_RL(0,0,.01,2,f,c,.4);
```

ok

kann das dem analytischen Verfahren sehr ähnliche Ergebnis betrachtet werden.

```
points:= { seq([x.k,y.k],k=0..200) }:
pointplot(points);
```



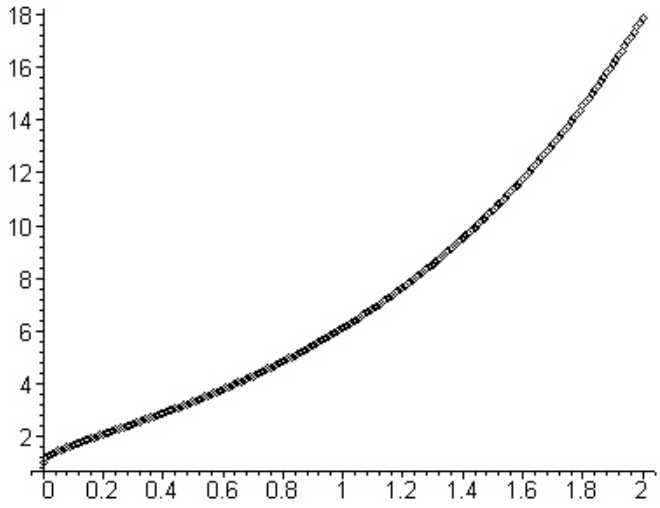
5.4.3.2 Beispiel: $D^n_{0_C} y(x) = ay(x)$ mit $n = 0.4$, $a = 1$ und $y(0) = 1$

```
f:=(x,y)->y:
c:=1:

ABM_C(0,.01,2,f,c,.4);
```

ok

```
points:=  seq([x.k,y.k],k=0..200) :
pointplot(points);
```



Ein genauerer Vergleich der berechneten Werte über eine Tabelle wird in Kapitel 5.5 gegeben.

5.4.3.3 Beispiel : $D^n_0 y(x) = ay(x)$, $n = 1.4$, $a = -1$ mit den Anfangsbedingungen $[I^{2-n}_0 y(x)]_{x=0} = y_I = 1$ und $[D^{n-1}_0 y(x)]_{x=0} = y_D = 1$.

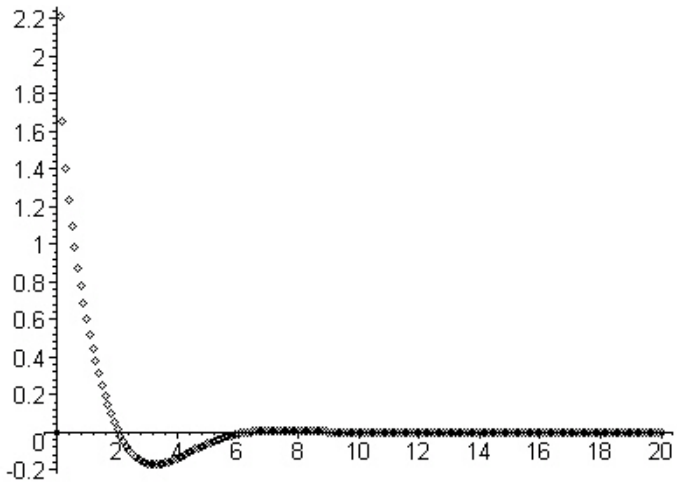
Die Prozeduraufrufe :

```
f:=(x,y)->-y:
c:=[1,1]:

ABM_RL(0,0,.01,2,f,c,1.4);
```

ok

```
points:= { seq([x.k,y.k],k=0..200) }:
pointplot(points);
```



5.4.3.4 Beispiel: $D^n_{0_C} y(x) = ay(x)$ mit $n = 1.4$, $a = 1$, $y(0) = 1$ und $y'(0) = 1$

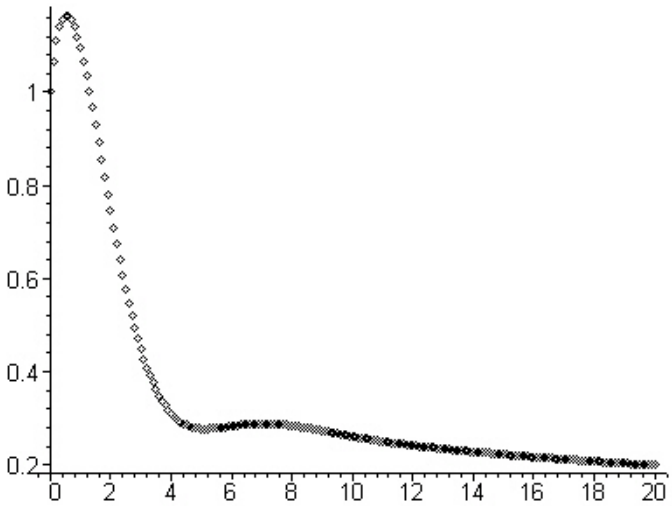
Die Prozeduraufrufe :

```
f:=(x,y)->-y:
c:=[1,1]:

ABM_C(0,.1,20,f,c,1.4);
```

ok

```
points:=  seq([x.k,y.k],k=0..200) :
pointplot(points);
```



5.5 Darstellung der Lösung als Tabelle

Auch wenn eine graphische Ausgabe schnell einen Überblick über den Verlauf einer Lösung erlaubt, so läßt sie nur schwer einen genauen Vergleich der berechneten Werte zu. Besser ist dann eine Darstellung in Tabellenform, was im Nachfolgenden in MAPLE programmiert werden soll.

Dabei sollen numerische Werte mit analytisch berechneten Werten verglichen und der Absolutbetrag ihrer Differenzen ebenfalls ausgegeben werden. Tabellen dieser Art wurden bereits in Kapitel 4.2 verwendet.

Für eine solche Tabelle ist nachfolgend der Programmcode aufgeführt. Notwendig zum Aufruf dieser Prozedur ist, dass vorher die Prozedur `ABM_RL()` und die an diese übergebenen Parametern in den entsprechenden Variablen gespeichert sind, da sonst der Tabellenaufruf mit evtl. falschen Werten erfolgt. Auch muss die Prozedur der exakten Lösung `y_exakt_RL()` und wegen dieser die Prozedur der Mittag-Leffler Funktion `ML()` verfügbar sein.

Der Tabellen-Prozedur werden vier Parameter übergeben. Der Koeffizient `a` der rechten Seite, die Ordnung `n` der Differentialgleichung, der Wert `ss`, der angibt wieviele berechnete Schritte pro Tabellenzeile übersprungen werden sollen und der Wert `x_max`, mit dem man der Tabellen-Prozedur den zu letzt zu berechnenden `y.k`-Wert übergibt.

```
ABM_RL_TABELLE:=proc(a,n,ss,x_max)
global CC;

CC:=seq(op(j,c),j=1..nops(c));

# Überschrift
printf(“ \n Wertetabelle einer RL Differentialgleichung \n“);
printf(“ \n n = %4.2f \ nSchrittweite = %4.2f \n \n“,n,h);

# Tabellenkopf
printf(“ x.k|   y.k   |   y(x.k)  | | y(x.k) - y.k | \ n“);
printf(“----|---------|---------|----------------\n“);
```

```
# Schleifendurchlauf pro Tabellenzeile
for i from 0 by ss to x_max/h do

 # Berechnung der exakten Werte y(x.k)
 if i = 0 then
  y_ex.i := infinity;
 else
  y_ex.i := evalf(y_exakt_RL(x.i,n,a,CC)):
 fi;

# Ausgabe der Tabellenzeile
printf(“ %6.2f | %16.14a | %16.14a | %16.14a \
          \n“,x.i,y.i,y_ex.i,abs(y.i-y_ex.i)):
od:

end:
```

Zu Beginn dieser Prozedur werden der in der Prozedur `ABM_RL()` global definierten Variablen `c` die Anfangsbedingungen als Sequenz zugewiesen, damit diese Anfangsbedingungen weiter unten im Programmcode der Prozedur `y_exakt_RL()` übergeben werden können. Dann erfolgt eine formatierte `printf`-Anweisung, die eine Kontrolle von Ordnung und Schrittweite für den Betrachter ermöglichen soll. Durch sie werden Ordnung und Schrittweite nach Prozeduraufruf zurückgegeben. Anschließend wird der Tabellenkopf definiert und nach diesem enthält die Prozedur eine Schleife, die in jedem Durchlauf den jeweiligen exakten Funktionswert $y(x)$ berechnet und eine Tabellenzeile ausgibt, in der die numerischen - und analytischen Werte gelistet werden, was ebenfalls wieder über eine formatierte `printf`-Anweisung erfolgt.

Will man für eine Caputo Differentialgleichung solch eine Tabelle erstellen, so muss lediglich in der Überschrift der Hinweis auf den Differentialgleichungstyp und im Prozeduraufruf der exakten Lösung die Caputo Lösung eingetragen werden, d.h. durch das Ändern der Zeilen

```
ABM_C_TABELLE:=proc(a,n,ss,x_max)
[...]
printf(‘‘ \n Wertetabelle einer C Differentialgleichung \n‘‘);
[...]
y_ex.i := evalf(y_exakt_C(x.i,n,a,CC)):
```

wird die Prozedur für Caputo Differentialgleichungen verwendbar.

5.5.1 Beispiel : $D_0^n y(x) = ay(x)$, $n = 1.4$, $a = -1$ mit den Anfangsbedingungen $[I_0^{2-n} y(x)]_{x=0} = y_I = 1$ und $[D_0^{n-1} y(x)]_{x=0} = y_D = 1$.

Nach dem Laden der Prozeduren `ML()` , `y_RL_D2_exakt()` , `ABM_RL()` und `ABM_RL_TABELLE()` in den Arbeitsspeicher und nach Speicherung der rechten Seite, der Anfangsbedingungen und des Koeffizienten und der Prozedur zur numerischen Berechnung der Wertepaare

```
f:=(x,y)->-y:
 a:=-1:
 c:=[1,1]:
ABM_RL(0,0,.1,10,f,c,1.4);
```

ok

können nach Aufruf der Prozedur

```
ABM_RL_TABELLE(a,n,9,9);
```

```
 Wertetabelle einer RL Differentialgleichung

 n = 1.40
Schrittweite =  .10

  x.k   |       y.k        |        y(x.k)     |   | y(x.k) - y.k |
--------|------------------|-------------------|--------------------
   0.00 |                0 |          infinity |          infinity
    .90 |      .6875468638 |       .4438458890 |       .2437009748
   1.80 |     .96055154e-1 |   -.335705407e-1  |       .1296256947
   2.70 |      -.143646498 |      -.1685784056 |     .249319076e-1
   3.60 |      -.161827978 |      -.1377496596 |     .240783184e-1
   4.50 |    -.98055968e-1 |   -.6706945423e-  |    .3098651377e-1
   5.40 |    -.35469466e-1 |   -.1517855531e-  |    .2029091069e-1
   6.30 |     -.1002816e-2 |    .733975926e-2  |     .834257526e-2
   7.20 |      .9119879e-2 |   .1032317733e-1  |     .120329833e-2
   8.10 |      .7053071e-2 |   .5796120874e-2  |    .1256950126e-2
   9.00 |      .2205593e-2 |   .1039787568e-2  |    .1165805432e-2
```

die Daten der analytischen - und der numerischen Lösung verglichen werden.

5.5.2 Beispiel: $D^n_{0_C} y(x) = ay(x)$ mit $n = 1.4$, $a = -1$, $y(0) = 1$ und $y'(0) = 1$

Auch hier müssen obige Befehle und Prozeduren bzgl. der Caputo Differentialgleichung aufgerufen werden. Ein anschließender Aufruf von

```
ABM_C_TABELLE(a,n,9,9);
```

```
 Wertetabelle einer C Differentialgleichung

 n = 1.40
Schrittweite =  .10

  x.k   |       y.k        |        y(x.k)     |   | y(x.k) - y.k |
--------|------------------|-------------------|--------------------
   0.00 |                1 |               1.  |                  0
   1.00 |      1.095706133 |      1.095142694  |       .563439e-3
   2.00 |       .744407962 |       .7438388052 |      .5691568e-3
   3.00 |       .448508365 |       .4482388315 |      .2695335e-3
   4.00 |       .310048032 |       .3100538681 |        .58361e-5
   5.00 |       .277726492 |       .2778439181 |      .1174261e-3
   6.00 |       .283107966 |       .2832085270 |      .1005610e-3
   7.00 |       .288337068 |       .2883782792 |       .412112e-4
   8.00 |       .283868808 |       .2838655944 |        .32136e-5
   9.00 |       .273032980 |       .2730136303 |       .193497e-4
  10.00 |        .26089892 |       .2608834476 |       .154724e-4
```

zeigt die Tabelle.

5.6 Graphische Ausgabe in drei Dimensionen

In diesem Abschnitt widmen wir uns nun der dreidimensionalen Darstellung von Lösungen fraktionaler Differentialgleichungen. Dabei wird die Ordnung der Differentialgleichung als die dritte Dimension in das Diagramm einfließen. In der nachfolgenden Prozedur wird analog zu der in Abschnitt 3.1.4 vorgegangen, wobei die Berechnungen numerisch, also mit Hilfe des Adams-Bashforth-Moulton Verfahrens durchgeführt werden, da so eine größere Menge von Differentialgleichungen gelöst werden können.

Der Prozedur werden dreizehn Parameter übergeben und die Rückgabewerte sind eine Textzeile und die dreidimensionale Graphik.

```
DG_3D:=proc(flag,ABM_,Y0,A1,s_A,A2,
                 X1,s_X,X2,Z1,Z2,delta,phi)
global LL;
if flag = y then
Z:=0;

for AA from A1 by s_A to A2 do
  L:=[[AA,0,0]];
  ABM_(X1,Y0,s_X,X2,f,c,AA);
  for xx from X1 by s_X to X2 do
    L := [ op(L), [AA,x.(xx),y(xx)] ];
  od:
  if Z = 0 then
    LL := [L];
    Z:=1;
  else
    LL := [op(LL),L];
od:
  fi;
printf(‘‘\nLösung für Ordnungen von %03.1f bis %03.1f ‘‘,A1,A2):

PLOT3D(MESH(LL),VIEW(A1..A2,X1+hh..X2,Z1..Z2), \
  AXESLABELS(Abl,x,z),AXES(BOXED),ORIENTATION(delta,phi));
end:
```

Zu Beginn wird mittels des `flag`-Parameters geprüft, ob eine Berechnung durchgeführt, oder auf bereits global gespeicherte Daten zurückgegriffen werden soll. Das hat bei längeren Berechnungen den Vorteil, dass, wenn man z.B. das Koordinatensystem nur in einem anderen z-Achsenabschnitt ausgeben will, nicht neu berechnet, sondern nur neu ausgegeben wird. Die Werte sind ja durch die letzte Prozedurausführung noch in der Liste `LL` gespeichert.

Weitere Parameter sind `ABM_`, durch den man angibt ob eine Caputo - oder eine Riemann-Liouville Differentialgleichung zu lösen ist, der y_0-Startwert für das numerische Verfahren `Y0`, die Parameter `A1,s_A,A2`, die der Reihe nach der Prozedur die erste zu berechnende Ordnung, die Schrittweite und die letzte zu berechnende Ordnung übergeben, die Parameter `X1,s_X,X2`, die für gleiches bezüglich der x-Achse zuständig sind und die vier Parameter `Z1,Z2,delta,phi`, die das z-Achsenintervall und spezielle Winkel für die Darstellung des Diagrammes übergeben.

In der Prozedur wird analog den Beschreibungen in Kapitel 3.1.4 eine Liste `L` angelegt, in die die numerisch berechneten Werte für eine Ordnung `AA` von `A1` in `s_A` Schritten bis `A2` gespeichert werden. Ist noch keine Liste `LL` vorhanden, so wird nachfolgend eine solche durch die Liste `L` definiert. In den nachfolgenden Schleifendruchläufen wird die Liste `L` immer wieder überschrieben und das Ergebnis an die Liste `LL` angehängt. Nach dem letzten Schleifendurchlauf erfolgt die Ausgabe des Textes über kleinste und größte berechnete Ordnung und die Ausgabe durch die Befehle `PLOT3D` und `MESH()`, die ebenfalls in 3.1.4 bereits genauer erläutert wurden.

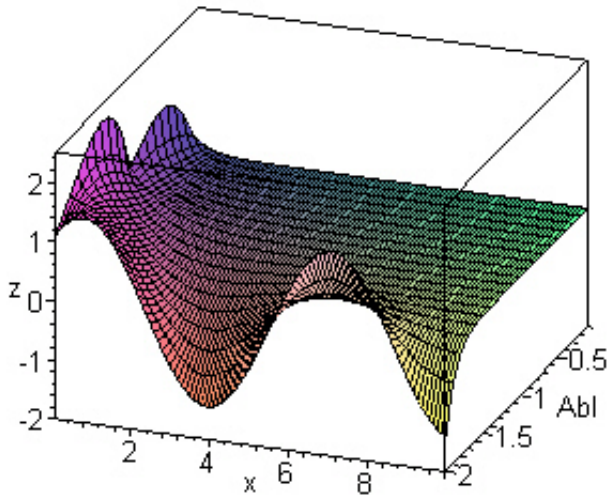
Hat man nun die Prozeduren `ABM_RL()` und `ABM_C()` in den Arbeitsspeicher geladen, so kann nach Speicherung der rechten Seite der Differentialgleichung und der Anfangsbedingung(en)

```
f:=(x,y)->-y;
c:=1:
```

und nach Aufruf der Prozedur mit den gewünschten Parametern

```
DG_3D(y,ABM_RL,0,.1,.1,2,0,.1,10,-2,2.5,20,60);
```

Lösung für Ordnungen von 0.1 bis 2.0

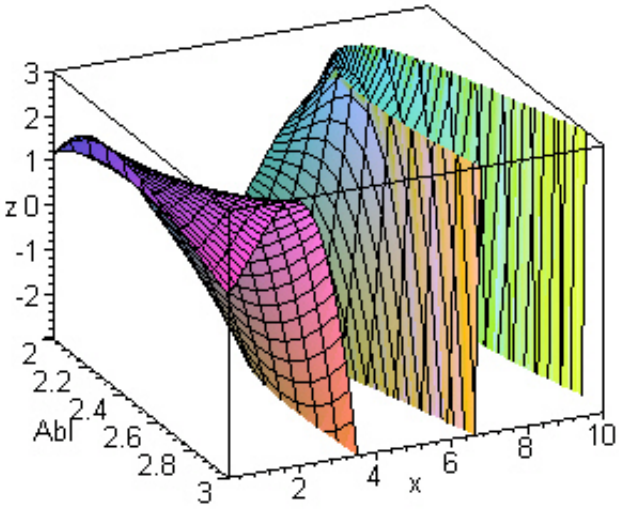


das Diagramm ausgeben lassen.

Eine weitere Berechnung soll nun für die Ordnungen von 2 bis 3 durchgeführt werden, wozu folgende Eingabe erforderlich ist

```
DG_3D(y,ABM_RL,0,2,.1,3,0,.1,10,-3,3,-25,60);
```

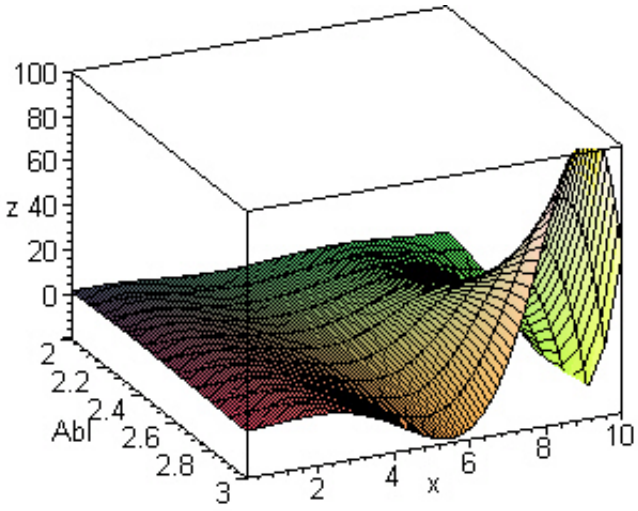
Lösung für Ordnungen von 2.0 bis 3.0



Ein “Wegzoomen“ ist nun ohne weitere Berechnungen möglich. Es muss lediglich die `flag` auf z.B. `n` gesetzt und neue Betrachtungsparameter an die Prozedur übergeben werden.

```
DG_3D(n,ABM_RL,0,.1,.1,2,0,.1,10,-20,20,-25,60);
```

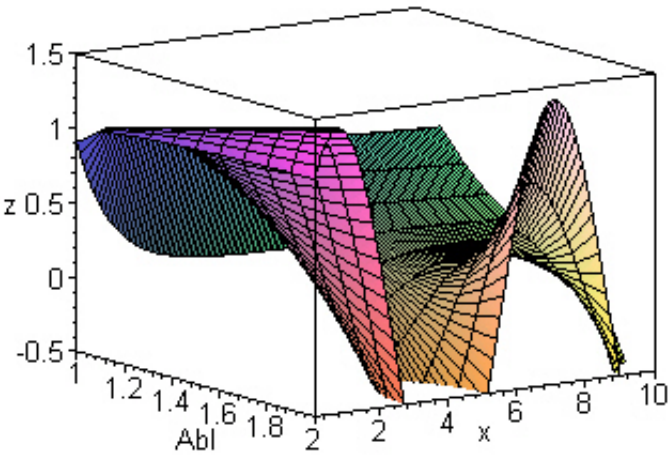
```
Lösung für Ordnungen von 2.0 bis 3.0
```



Abschließend noch eine Berechnung der sich ergebenden Lösungen von Caputo Differentialgleichungen

```
DG_3D(y,ABM_C,0,1,.1,2,0,.1,10,-.5,1.5,-35,75);
```

```
Lösung für Ordnungen von 0.1 bis 2.0
```



6. Schlussbemerkungen

Trotz der im Kapitel 4.3 besprochenen Einschränkungen auf Grund der Anfangsbedingungen von Riemann-Liouville Differentialgleichungen finden sich eine Vielzahl von Anwendungen für fraktionale Differentialgleichungen in der Natur, vor allem dann, wenn es sich um Prozesse mit Stoffen, die sich zwischen zwei Aggregatzuständen befinden, handelt.

Zum Beispiel ist die Volumensänderung eines festen Stoffes unter Druck proportional der 0. ten Ableitung eben dieses Druckes, sprich proportional des Druckes selber. Bei flüssigen Stoffen ist die Volumensänderung proportional der 1. Ableitung des Druckes, sprich proportional der Änderung des Druckes. Mit anderen Worten ändert sich ein einmal zerdrückter absolut fester Stoff nach Rücknahme des Druckes nicht mehr, eine Flüssigkeit hingegen schon. Diesbezüglich scheint es vernünftig bei zähflüssigen Stoffen auf die Proportionalität bezüglich einer Ableitung zwischen 0 und 1 zu schliessen.

Auch bei Ermüdungserscheinungen von z.B. Metallen können fraktionale Differntialgleichungen die realen Prozesse meist besser beschreiben als klassische Differentialgleichungen. Ein Metall hält in der Regel sehr lange, bis es plötzlich bricht und sofort absolut unbrauchbar wird. Der Zustand bevor das Metall bricht muss als ein sehr lange andauernder Übergang des Metalls von dem einen in den anderen Zustand betrachtet werden, auch wenn man dem Metall die Veränderung nicht ohne weiteres ansehen kann. Dennnoch kann es kurz vor dem Bruch nicht das Metall mit den selben physikalischen Eigenschaften gewesen sein, wie am Tag seiner Produktion.

Aber nicht nur in der physikalischen Welt der Körper und Flüssigkeiten, sondern auch in der Elektrizität finden sich Anwendungen für fraktionale Differntialgleichungen. So z.B. in Stromkreisen, die auf Grund ihrer Bauteile spezifische und kapazitäre Widerstände besitzen, also Widerstände die durch die Eigenschaften der Materialien vorhanden sind und Widerstände, die durch elektromagnetische Felder z.B. in Spulen auftreten. Zur Beschreibung von Prozessen in solchen elektrischen oder elektronischen Schaltungen können fraktionale Differentialgleichungen eingesetzt werden.

Weitere Anwendungen findet man z.B. auch bei der Leitfähigkeit biologischen Materials von elektrischem Strom, bei elektrolytischen Prozessen nahe der Elektrodenoberflächen oder allgemein bei Versuchen, experimentell erworbenes Datenmaterial in mathematischen Gleichungen beschreiben zu wollen und sind u.a. in [2] aufgeführt.

Jüngste Bemühungen der Entwicklungen z.B. fraktionale Varianten der Maxwell'schen Gleichungen der Elektrodynamik, fraktionale Lagrange oder fraktionale Hamilton Operatoren weisen darauf hin, dass evtl. diese neuen Gleichungen die Natur allgemeiner und treffender beschreiben würden, als es derzeit die klassisch formulierten Gleichungen tun. Die durch das fraktionale Integral einer Funktion beschriebene Faltung greift auf Zustände des Systems vor dem aktuell betrachteten Zeitpunkt zurück, verändert so diesen momentanen Zeitpunkt und kann als eine mathematische Formulierung von Gedächtnis betrachtet werden. Da jedes Material ebenfalls solch ein Gedächtnis besitzt, was sich durch Alterungs- bzw. Ermüdungserscheinungen zeigt, können allgemein fraktionale Differentialgleichungen diesem Tatbestand Rechnung tragen.

Ob allerdings die Entwicklung hin zu komplexen Ableitungen und mit ihnen hin zu komplexen Differentialgleichungen sinnvoll ist, kann zum momentanen Zeitpunkt nicht beantwortet werden. Arbeiten darüber wurde von Love im Jahre 1990 in Melbourne (Australien) begonnen, jedoch nicht fertiggestellt, da dieser im Jahre 2001 verstarb.

7. Anhang

7.1 Stichwortverzeichnis

7.2 Programmcode im Internet

Die vollständige Examensarbeit und sämtlicher MAPLE Programmcode ist im Internet erhältlich auf den Seiten

http://www.math.uni-frankfurt.de/~numerik

http://db.code23.de/mathe

Folgende Arbeitsblätter liegen in kommentierter Form bereit.

	Inhalt	Name des Maple-Worksheets
3.0	Einführende Beispiele	3_0_Einfuehrung_in_Maple.mws
3.1	Riemann-Liouville (RL) Integrale und Ableitungen	3_1_RL_abl_int.mws
3.2	Caputo (C) Ableitungen	3_2_C_abl.mws
3.3	Vergleich zwischen RL und C	3_3_RL_C.mws
5.3	analytische fDG - Lösungen graphisch in zwei Dim.	5_3_analyt_2d.mws
5.4	numerische fDG - Lösungen graphisch in zwei Dim.	5_4_numer_2d.mws
5.5	tabellarische fDG - Lösungen von RL Differentialgleichungen	5_5_Tabelle_RL.mws
5.5	tabellarische fDG - Lösungen von C Differentialgleichungen	5_5_Tabelle_C.mws
5.6	numerische fDG - Lösungen graphisch in drei Dim.	5_6_numer_3d.mws

7.3 Literaturverzeichnis

[1] FORSTER, Analysis I,
Vieweg-Verlag, Wiesbaden, 1992

[2] PODLUBNY, Fractional Differential Equations,
ACADEMIC PRESS, San Diego (USA), 1999

[3] DIETHELM, Vorlesungsskript “Fractional Differential Equations“,
TU Braunschweig, SS 2000
http://www.tu-bs.de/∼diethelm

[4] WESTERMANN, Mathematik für Ingenieure mit Maple - Band 2,
Springer-Verlag, Heidelberg, 1997

[5] BRONSTEIN, Taschenbuch der Mathematik,
Verlag Harri Deutsch, Frankfurt am Main, 1999

[6] ROSS, Fractional Calculus and its Applications,
Springer-Verlag, Heidelberg, 1975

[7] NKAMNANG, Dissertation: Diskretisierung von mehrgliedrigen Abelschen Integralgleichungen und gewöhnlichen Differentialgleichungen gebrochener Ordnung, FU Berlin, 1998
http://www.diss.fu-berlin.de/1999/23/index.html

7.4 Versicherung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst, keine anderen als die angegebenen Hilfsmittel verwendet und sämtliche Stellen, die benutzten Werken im Wortlaut oder dem Sinne nach entnommen sind, mit Quellenangaben kenntlich gemacht habe. Sämtliche Zeichnungen sind durch von mir entwickelten Programmcode entstanden.

Ort / Datum	Dietmar Blume