Identifikation spezieller Funktionen, die durch Rodriguesformeln gegeben sind

Dissertation zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)

am Fachbereich für Mathematik und Naturwissenschaften
der Universität Kassel

Frau Dipl.-Math. Kornelia Karla Fischer

betreut von
Prof. Dr. Wolfram Koepf

UNIKASSEL
VERSITÄT

Tag der Disputation:

19. Januar 2016

Erstgutachter:

Prof. Dr. Wolfram Koepf

Universität Kassel

Zweitgutachter:

Prof Dr. Werner Seiler

Universität Kassel

Mache die Dinge so einfach wie möglich - aber nicht einfacher. (Albert Einstein)

Inhaltsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einführung

Wir wollen in dieser Doktorarbeit Funktionenfamilien identifizieren, deren Rodriguesformel gegeben ist. Rodriguesformeln tauchen in der Theorie orthogonaler Polynome auf und werden in der Form

$$P_n = g_n D^n h_n$$

dargestellt. Dabei steht D stellvertretend für den Differentialoperator, den Differenzenoperator oder aber den D_q-Operator. Wir werden diese Operatoren im folgenden Kapitel genau einführen. Ihren Namen verdankt die Formel B. O. Rodrigues, der sie 1814 für die Legendrepolynome einführte, siehe [NU(1988), Kapitel 1].

Die Fragestellung, mit Hilfe des Zeilberger-Algorithmus (diskrete Fassung [Zeilberger(1990), Zeilberger(1991)]) Funktionenfamilien zu identifizieren, deren Rodriguesformel gegeben ist, wurde für den stetigen Fall bereits von Almkvist und Zeilberger ([AZ(1990)]) aufgeworfen und von [Koepf(2014)] ausgearbeitet und implementiert. Wir stellen die Vorgehensweise nochmals vor und testen sie an neuen Beispielen. Vor allem aber wollen wir diese Fragestellung nun auch für den diskreten und q-diskreten Fall behandeln und dafür die entsprechenden Rodriguesformeln in Summenschreibweise überführen, denn für die Anwendung des Zeilberger-Algorithmus muss uns eine Summe vorliegen. Die Umwandlung in Summen wird das theoretische Fundament dieser Arbeit bilden und stellt die wesentliche Leistung dieser Arbeit dar. Wir stellen als Werkzeuge eine Reihe von Sätzen für die Ausdrücke $D^n f(x)$ bereit, die mit Hilfe experimenteller Berechnungen mit einem Computeralgebrasystem gefun-

den werden konnten. Zu experimenteller Mathematik darf sich der Leser gerne in [BD(2011)] belesen.

Diese Arbeit baut auf dem Zeilberger-Algorithmus auf, einem Meilenstein in der Theorie der bestimmten Summation hypergeometrischer Terme. Ein Term a_k heißt dabei hypergeometrisch, wenn sein Termverhältnis $\frac{a_{k+1}}{a_k}$ rational in k ist. Der Zeilberger-Algorithmus bietet ein einfaches und wirkungsvolles Beweisschema zum Beweisen hypergeometrischer Identitäten, das in der Literatur unter dem Namen Zeilbergers Paradigma erscheint. Es erlaubt Beweise derart, dass aus linker und rechter Summe der hypergeometrischen Aussage jeweils eine Rekursionsgleichung erzeugt wird und, falls diese übereinstimmen, noch entsprechend viele Anfangswerte berechnet werden. Das Wort Rekursionsgleichung kann hierbei auch durch Differential- bzw. Differenzengleichung ersetzt werden.

Wir formen im Folgenden Rodriguesformeln in Summen um und schauen mit Hilfe von Zeilbergers Paradigma, ob sie dieselben Funktionenfamilien wie bekannte hypergeometrische Summen generieren. Dazu erzeugen wir auf Grundlage des Zeilberger-Algorithmus – sei es der stetige, der diskrete oder der q-diskrete – Rekursions- sowie Differentialgleichungen (bzw. (q-)Differenzengleichungen). Diese Schritte werden für alle Unterfälle in Mapleprogrammen umgesetzt und getestet.

1.1 Der Durchbruch in der Theorie der bestimmten Summation hypergeometrischer Terme mit dem Zeilberger-Algorithmus: ein kurzer Rückblick

Bill Gosper war es, der mit dem nach ihm benannten Gosper-Algorithmus ([Gosper(1978)]) den Grundstein für die schnelle Berechnung von geschlossenen Ausdrücken für Summen über hypergeometrische Terme legte. Er fand ihn während der Entwicklung von Macsyma, einem der ersten symbolischen Computeralgebrasysteme. Der Algorithmus erhält als Eingabeterm einen hypergeometrischen Term a_k und berechnet dazu eine hypergeometrische diskrete Stammfunktion s_k, falls diese existiert. Es liegt in der Natur der Sache, dass

hypergeometrische diskrete Stammfunktionen selten sind. Damit standen die Mathematiker weiterhin vor dem Problem, *bestimmte* Summen über hypergeometrische Terme, etwa Binomialreihen, effizient bestimmen zu können. Von daher lag Betrachtungen über Binomialreihen weiterhin langwierige und sehr technische rationale Arithmetik zugrunde. Dies zeigen die Arbeiten von Perlstadt und Cusick aus den 1980er Jahren, die sich mit Summen der Gestalt $S_n^{(r)} := \sum_{k=0}^{n} \binom{n}{k}^r$ für $r \geq 1$ befassen. Bereits im Jahre 1894 hatte Franel in [Franel(1894)] eine Rekursionsgleichung für $S_n^{(3)}$ veröffentlicht. 1895 folgte dann in [Franel(1895)] die für $S_n^{(4)}$. [Perlstadt(1987)] gab über 90 Jahre später in seiner Arbeit Rekursionsgleichungen für $S_n^{(r)}$ für die Fälle $r = 5$ und $r = 6$ an, wobei er intensive Berechnungen mit dem Computeralgebrasystem Macsyma durchführte. [Cusick(1989)] gab schließlich eine explizite Methode an, um für jedes $S_n^{(r)}$ eine $\lfloor \frac{r+3}{2} \rfloor$-Term-Rekursionsgleichung zu finden. Hierbei bezeichnet $\lfloor x \rfloor$ die ganzzahlige Abrundungsfunktion von x. Allerdings waren auch diese Rechnungen technisch und umständlich. Außerdem behandelten die genannten Arbeiten lediglich die kleine Beispielklasse $S_n^{(r)}$. Interessant wäre doch vielmehr, generell Rekursionsgleichungen für bestimmte Summen s_n über hypergeometrische Terme zu finden. Genau diese Leistung vollbringt der Zeilberger-Algorithmus aus dem Jahre 1990. Wenden wir ihn an, um beispielsweise eine Rekursionsgleichung für $S(n) = \sum_{k=0}^{n} \binom{n}{k}^7$ zu berechnen, erhalten wir in Maple binnen einer Sekunde folgendes Ergebnis:

```
> read "hsum17.mpl";
```

Package "Hypergeometric Summation", Maple V − Maple 17

Copyright 1998 − 2013, Wolfram Koepf, University of Kassel

```
> TIME:=time():
> sumrecursion(binomial(n,k)^7,k,S(n));
> time()-TIME;
```

$$
\begin{aligned}
&(427721\,n^8+6354712\,n^7+40913943\,n^6+149008897\,n^5+335597294\,n^4\\
&+478442631\,n^3+421557546\,n^2+209877096\,n+45209280)\\
&(n+3)^2\,(n+4)^6\,\mathrm{S}(n+4)-(30368191\,n^{14}+1088916563\,n^{13}\\
&+17971912105\,n^{12}+180879396742\,n^{11}+1239681510073\,n^{10}\\
&+6117625957887\,n^9+22406262825083\,n^8+61845130443640\,n^7\\
&+129212210111012\,n^6+203258395972016\,n^5\\
&+236869167238448\,n^4+198216442561728\,n^3\\
&+112552666603632\,n^2+38805627231072\,n+6127621340928)\\
&(n+3)^2\,\mathrm{S}(n+3)-(3244263785\,n^{16}+126062821360\,n^{15}\\
&+2283968506414\,n^{14}+25606027648545\,n^{13}\\
&+198784165636833\,n^{12}+1132823172700850\,n^{11}\\
&+4900968186516568\,n^{10}+16415798739266369\,n^9\\
&+43010799826545440\,n^8+88420368230599884\,n^7\\
&+142107402452328480\,n^6+176624649389228512\,n^5\\
&+166377205614902736\,n^4+114791322401632464\,n^3\\
&+54690808998655008\,n^2+16071328274727552\,n\\
&+2193807069981696)\mathrm{S}(n+2)+(15821827511\,n^{14}\\
&+504038219279\,n^{13}+7388757320392\,n^{12}+66049812430419\,n^{11}\\
&+402186441422282\,n^{10}+1764446202422005\,n^9\\
&+5750836202090468\,n^8+14144725417173505\,n^7\\
&+26380423880989287\,n^6+37123771902845896\,n^5\\
&+38801484010527532\,n^4+29207641278240480\,n^3\\
&+14968213677069888\,n^2+4674653721868800\,n\\
&+671258737065984)(n+2)^2\,\mathrm{S}(n+1)+128(427721\,n^8\\
&+9776480\,n^7+97373115\,n^6+551893883\,n^5+1946706314\,n^4\\
&+4375566933\,n^3+6119692458\,n^2+4869142152\,n\\
&+1687389120)(n+2)^2\,(n+1)^6\,\mathrm{S}(n)=0
\end{aligned}
$$

0.514

1.2 Der konkrete Aufbau dieser Arbeit

Diese Dissertation besteht aus fünf Kapiteln. Dabei dienen die ersten drei Kapitel der Einführung in das zum Verständnis nötige mathematische Vokabular sowie den Zeilberger-Algorithmus in seinen verschiedenen Varianten. Im vierten Kapitel wird dann die eigentliche Fragestellung für alle Unterfälle behandelt und die implementierten Algorithmen werden an zur Fallgruppe passenden hypergeometrischen orthogonalen Polynomfamilien getestet. Die Algorithmen wurden in dem Computeralgebrasystem Maple umgesetzt. Der Code wurde in der Arbeit dokumentiert. Er liegt in vollständiger Form auf einem der Arbeit beiliegenden Datenträger vor. Wir wollen an dieser Stelle darauf hinweisen, dass Ergebnisse dieser Arbeit, etwa aus den Abschnitten 4.2.1 und 4.3.1, nach Absprache mit Prof. Dr. Wolfram Koepf bereits in der Neuauflage seines Buches „Hypergeometric Summation“ ([Koepf(2014)]) in Kapitel 13 veröffentlicht wurden.
Im letzten Kapitel findet der Leser einen kurzen Ausblick.

Im zweiten Kapitel wird die algebraische Umgebung festgelegt, in der die folgenden Überlegungen stattfinden. Außerdem werden Begriffe definiert und Operatoren eingeführt. Sätze über Eigenschaften bestimmter Symbole als auch von Operatoren werden angeführt. Außerdem werden klassische orthogonale Polynomsysteme, die als Testobjekte für die einzelnen Testfälle dienen, vollständig gelistet.

Das dritte Kapitel stellt Gosper- und Zeilberger-Algorithmus für den diskreten (Summations-), stetigen (Integrations-) und q-diskreten Fall vor.
Der Gosper-Algorithmus für die unbestimmte Summation findet zu einem hypergeometrischen Eingabewert a_k eine hypergeometrische diskrete Stammfunktion s_k. Im Algorithmus wird nun für $\frac{a_{k+1}}{a_k}$ eine Darstellung

$$\frac{a_{k+1}}{a_k} = \frac{p_{k+1}}{p_k}\frac{q_{k+1}}{r_{k+1}} \tag{1.1}$$

mit Polynomen p_k, q_k, r_k mit der Eigenschaft

$$\gcd(q_k, r_{k+j}) = 1 \text{ für alle } j \in \mathbb{N}_0 \tag{1.2}$$

gefunden.

gcd steht hierbei kurz für den größten gemeinsamen Teiler zweier Polynome. Es stellt sich heraus, dass s_k ein rationales Vielfaches von a_k ist:

$$s_k := \frac{a_k f_{k-1} r_k}{p_k}.$$

r_k und p_k, die (1.1) mit der Eigenschaft (1.2) erfüllen, können durch gezieltes Umschreiben des Zähler- und Nennerpolynoms von $\frac{a_{k+1}}{a_k}$ bestimmt werden. Die Forderung (1.2) führt dazu, dass es sich bei f_k sogar um ein Polynom handelt. Für dieses Polynom f_k muss zunächst eine Gradschranke gefunden werden, woraufhin ein lineares Gleichungssystem zu lösen ist. Damit ist s_k vollständig bestimmt.
Außerdem stellen wir den Gosper-Algorithmus für den stetigen Fall vor. Darüber hinaus betrachten wir den q-Gosper-Algorithmus, der eine q-hypergeometrische diskrete Stammfunktion zu einem q-hypergeometrischen Eingabeterm findet. Einen Term a_k nennt man q-hypergeometrisch, falls $\frac{a_{k+1}}{a_k} \in \mathbb{F}(q^k)$ gilt. Dabei ist $\mathbb{F}$ der um q erweiterte Grundkörper. Weitere Angaben zur algebraischen Umgebung, in der diese Arbeit sich bewegt, findet der Leser in Kapitel 2. Im stetigen Gosper-Algorithmus und q-Gosper-Algorithmus werden lediglich die Begrifflichkeiten angepasst. Im q-diskreten Fall ist zu beachten, dass f_k kein Polynom, sondern ein Laurentpolynom ist. Dementsprechend werden im Gradschrankenalgorithmus eine untere und eine obere Gradschranke gefunden.

Anschließend wird der Zeilberger-Algorithmus vorgestellt. Der Zeilberger-Algorithmus entspricht einer Variante des Gosper-Algorithmus angewandt auf einen modifizierten Eingabeterm. Um $s_n = \sum_{k=-\infty}^{\infty} F(n,k)$, eine bestimmte Summe über einen hypergeometrischen Term bzgl. n als auch k, zu finden, wird der Gosper-Algorithmus schließlich auf den Eingabeterm

$$a_k = F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k)$$

angewandt, wobei die noch zu bestimmenden σ_j von n, nicht aber von k abhängen. Erfreulich daran ist, dass nach der Aktualisierung von p_k, q_k und r_k gemäß dem Gosper-Algorithmus die Unbekannten $\{\sigma_j(n)\}_{j=1,\ldots,J}$ linear in p_k auftauchen, weswegen wiederum ein lineares Gleichungssystem zu lösen bleibt. Über ein Teleskopsummenargument erhalten wir schließlich die gesuchte holonome Rekursionsgleichung.

Des Weiteren wird der Zeilberger-Algorithmus am Beispiel der Summe $S_n^{(2)} = \sum_{k=0}^{n} \binom{n}{k}^2$ konkret vorgestellt. Außerdem wird jeweils für den diskreten bzw. stetigen Fall eine holonome Differentialgleichung aus einer Summe bzw. einem Integral generiert. Auch der Zeilberger-Algorithmus wird für die Integration und den q-Fall entsprechend angepasst.

Im vierten Kapitel werden wir holonome Funktionenfamilien identifizieren, indem wir holonome Rekursionsgleichungen als auch holonome Differential- bzw. (q-)Differenzengleichungen aus Rodriguesformeln erzeugen.
Wir beginnen mit dem stetigen Fall. Diese Problematik wurde erstmals von Almkvist und Zeilberger in [AZ(1990)] aufgeworfen und in [Koepf(2014)] umgesetzt. Wir werden die Vorgehensweise nochmals verdeutlichen, da sie wegweisend für den diskreten und q-diskreten Fall ist. Wir zeigen, wie die Rodriguesformel $f_n = g_n\left(\frac{d}{dx}\right)^n h_n$ in eine Integralschreibweise überführt wird, auf die dann der stetige Zeilberger-Algorithmus Anwendung findet. Der stetige Fall hat anleitende Wirkung für die spätere Überführung in Summen im diskreten Fall. Im stetigen Fall erzeugen wir holonome Rekursionsgleichungen als auch holonome Differentialgleichungen aus Rodriguesformeln. Wir stellen die nötigen Prozeduren aus [Koepf(2014)] vor und ergänzen sie durch eine Normierungsprozedur, die die Rekursionsgleichungen in einer besonderen Form wie im Standardwerk [KLS(2010)] ausgibt. Außerdem testen wir die Prozeduren konkret am Beispiel einer Polynomfamilie aus der Klasse der klassischen orthogonalen Polynomsysteme (kurz COPS), d.h. wir identifizieren diese Polynomfamilie durch Abgleich der aus Rodriguesformel sowie aus der hypergeometrischen Reihe erzeugten Rekursionsgleichung (bzw. Differentialgleichung) und berechnen aus beiden Darstellungen entsprechend viele Anfangswerte. Wir geben darüber hinaus jeweils Tabellen mit Rekursionsgleichungen und Differentialgleichungen der COPS an, die mit den vorgestellten Prozeduren erzeugt wurden. Interessant ist, dass wir die Prozeduren außerdem noch an einer Polynomfamilie testen, die keine orthogonale Funktionenfamilie ist, denn wir wollten die Problemstellung der Identifikation allgemein für holonome Funktionenfamilien vorstellen.
Im diskreten Fall ist es uns möglich, die Rodriguesformel $f_n = g_n \Delta^n h_n$ als Summe zu schreiben. Dabei bezeichnet Δ den Vorwärtsdifferenzenoperator definiert

durch $\Delta f(x) = f(x+1) - f(x)$. Analog kann auch aus der Darstellung mit dem Rückwärtsdifferenzenoperator ∇ definiert durch $\nabla f(x) = f(x) - f(x-1)$ eine Summe generiert werden. Wir fokussieren uns auf die Darstellung mit dem Δ-Operator und wenden auf die gewonnene Summe den diskreten Zeilberger-Algorithmus an. Auf diese Weise können wir sowohl holonome Rekursions- als auch holonome Differenzengleichungen generieren. Wir setzen dies in entsprechenden Prozeduren um. Wir testen die Algorithmen an einer Polynomfamilie aus der Familie der klassischen diskreten orthogonalen Polynome (kurz CDOPS) und gleichen die erzeugte Rekursionsgleichung als auch Differenzengleichung jeweils mit der aus der hypergeometrischen Reihe erzeugten ab. Es werden für die Identifizierung entsprechend viele Anfangswerte aus beiden Darstellungen berechnet. Wir geben die Ergebnisse erneut in Tabellenform an und listen die mit den von uns implementierten Prozeduren erzeugten Rekursionsgleichungen und Differenzengleichungen der CDOPS auf.
Im q-Fall muss das Vorgehen weiter differenziert werden, denn es tauchen in [KLS(2010)] vier Arten von q-Rodriguesformeln auf:

$$f_n(x) = g_n(x) D_q^n h_n(x) \tag{1.3}$$

$$f_n(x) = g_n(x) D_{q^{-1}}^n h_n(x) \tag{1.4}$$

$$f_n(q^{-x}) = g_n(x) \nabla_q^n h_n(x) \tag{1.5}$$

$$f_n(q^x) = g_n(x) \nabla_{q+}^n h_n(x) \tag{1.6}$$

Um q-Rekursionsgleichungen als auch q-Differenzengleichungen aus diesen q-Rodriguesformeln zu generieren, werden wir Sätze bereitstellen, die dabei behilflich sind, die Formeln (1.3)-(1.6) in Summenschreibweise umzuwandeln. Die induktiven Beweise dieser Sätze sehen einfach aus, die Formeln zu finden, bedurfte einiger computeralgebraischer Experimente. Sind die vier Summenformeln schließlich gefunden, kann darauf der q-Zeilberger-Algorithmus angewandt werden und es können q-Rekursions- als auch q-Differenzengleichungen erzeugt werden. Das Vorgehen wird in entsprechenden Prozeduren umgesetzt. Außerdem werden Normierungsprozeduren geschrieben, die die q-Rekursions- als auch q-Differenzengleichungen in einer besonderen Form (wie in [KLS(2010)]) ausgeben. Wir testen die Prozeduren exemplarisch am Beispiel spezieller q-orthogonaler Polynomfamilien, deren q-Rodriguesformeln in [KLS(2010)] den Formeln (1.3)-(1.6) genügen. Wir geben tabellarisch q-Rekursions- als auch

q-Differenzengleichungen an, die mit unseren Prozeduren erzeugt wurden. Hervorzuheben bleibt, dass wir die in [KLS(2010)] angegebene q-Rodriguesformel der Stieltjes-Wigert-Polynome erfolgreich widerlegen konnten. Zwar unterscheidet sich die richtige, von uns gefundene q-Rodriguesformel nur um einen Vorfaktor von der in der Literatur ([KLS(2010)]) notierten. Doch zeigt dieses Beispiel, welch nützliches Werkzeug die von uns vorgestellten Prozeduren zum Identifizieren von Funktionenfamilien bei gegebener Rodriguesformel sind.
Wir geben an dieser Stelle einen tabellarischen Überblick über alle Testfälle. Obwohl die Algorithmen auf alle holonomen Funktionenfamilien anwendbar sind, haben wir sie systematisch am Beispiel der orthogonalen Polynomsysteme getestet. Tabelle 1.1 zeigt einen Überblick aller Testfälle und ordnet jeweils die entsprechende Testklasse orthogonaler Polynome zu.

	stetig	diskret	q-diskret
Rekursionsgleichung	COPS	CDOPS	q-OPS
Differential- bzw. (q-)Differenzengleichung	COPS	CDOPS	q-OPS

Tabelle 1.1: Alle Testfälle mit Testklassen im Überblick

Im fünften Kapitel wird ein Ausblick gegeben. So kann die gesamte Fragestellung unter Verwendung des Zeilberger-Algorithmus auch bearbeitet werden, wenn an Stelle der Rodriguesformel einer Funktionenfamilie $P_n(x)$ die erzeugende Funktion $G(z) = \sum_{n=0}^{\infty} P_n(x) z^n$ gegeben ist, die alle Informationen über die Funktionenfamilie beinhaltet.

1.3 Danksagungen

Recht herzlich danken möchte ich Prof. Dr. Wolfram Koepf. Durch seine aufrichtige Unterstützung war es mir möglich, auf dem Gebiet der Computeralgebra Fuß zu fassen. Prof. Dr. Wolfram Koepf hat durch seine langjährige Berufserfahrung, konstruktive Gespräche, seine wohlwollende Art und den Blick für das Ganze maßgeblich dazu beigetragen, dass diese Arbeit gelingen konnte.
Mein zweiter Dank gilt Prof. Dr. Werner Seiler, der mir durch den wertvollen Tipp in der Straßenbahn, doch einmal in Sloanes Online-Enzyklopädie für Folgen ganzer Zahlen zu sehen, entscheidend beim Finden der Summenformeln half. Auch ihm herzlichen Dank!
Mein weiterer Dank gilt Mario Albert, Michaela Camin, Dr. Etienne Nana Chiadjeu, Matthias Fetzer, Dr. Melanie Gerling, Carolin Heutling, Klaus Hofmann, Dr. Stefan Kopecz, Dr. Jennylee Müller, Dr. Frank Quedenfeld, Dr. Patrick Njionou Sadjang, Henning Schatz, Michael Schweinfurter, Dr. Torsten Sprenger, Emma Steinbrecher, Dr. Daniel Duviol Tcheutia, Dominik Wulf und Dr. Florian Zanger.

Kapitel 2

Die Begriffswelt

Allgemein setzen wir einen Körper $\mathbb{K}$ voraus, unter dem wir fortan den Körper der rationalen Zahlen $\mathbb{Q}$ oder endliche Körpererweiterungen desselben verstehen. Für analoge Betrachtungen im q-Fall legen wir den Körper $\mathbb{F}$ der rationalen Funktionen in q über $\mathbb{K}$ zugrunde ($\mathbb{F}$=$\mathbb{K}(q)$), wobei q stets symbolisch betrachtet wird. Der Leser darf sich unter q eine reelle Zahl mit $0 < q < 1$ vorstellen.

2.1 Der allgemeine Fall

2.1.1 Definitionen

Definition 2.1 *Das Symbol* $a^{\underline{k}} := a \cdot (a-1) \cdots (a-k+1)$ *heißt fallende Faktorielle.*

Definition 2.2 *Das Symbol* $(a)_k := a \cdot (a+1) \cdots (a+k-1)$ *heißt Pochhammersymbol. Es wird auch steigende Faktorielle genannt.*

Definition 2.3 *Die Fakultät* $k!$ *ist definiert als* $k! = k \cdot (k-1) \cdots 2 \cdot 1$.
Dabei gilt $k! = k^{\underline{k}}$ *als auch* $k! = (1)_k$.

Definition 2.4 *Die allgemeine hypergeometrische Funktion bzw. Reihe* ${}_pF_q$ *ist gegeben durch*

$$ {}_pF_q\left(\begin{matrix}\alpha_1, \ldots, \alpha_p \\ \beta_1, \ldots, \beta_q\end{matrix} \,|x\right) := \sum_{k=0}^{\infty} \frac{(\alpha_1)_k \cdot (\alpha_2)_k \cdot \ldots \cdot (\alpha_p)_k}{(\beta_1)_k \cdot (\beta_2)_k \cdot \ldots \cdot (\beta_q)_k} \frac{x^k}{k!} $$

Wir wollen außerdem definieren:

Definition 2.5 *Ein Term a_k heißt hypergeometrisch, falls gilt*

$$\frac{a_{k+1}}{a_k} \in \mathbb{K}(k).$$

Definition 2.6 *Eine differenzierbare Funktion $F(x)$ heißt hyperexponentiell, falls gilt*

$$\frac{F'(x)}{F(x)} \in \mathbb{K}(x).$$

Definition 2.7 *Eine Rekursionsgleichung für a_k heißt holonom, wenn sie homogen und linear ist und Polynomkoeffizienten $\in \mathbb{K}[k]$ hat.*

Definition 2.8 *Eine Differentialgleichung für $f(x)$ heißt holonom, wenn sie homogen und linear ist und Polynomkoeffizienten $\in \mathbb{K}[x]$ hat.*

2.1.2 Operatoren

Wir werden im Folgenden mit dem Vorwärts- und Rückwärtsdifferenzenoperator arbeiten:

Definition 2.9 *Der Vorwärtsdifferenzenoperator Δ ist definiert durch*

$$\Delta f(x) := f(x+1) - f(x).$$

Wir nennen ihn auch Delta-Operator.

Satz 2.10 *Für den Delta-Operator gilt*

$$\Delta x^{\underline{n}} = n x^{\underline{n-1}}.$$

Beweis.

$$\begin{aligned}
\Delta x^{\underline{n}} &= (x+1)^{\underline{n}} - x^{\underline{n}} \\
&= (x+1)x \cdots (x-n+2) - x(x-1)\cdots(x-n+2)(x-n+1) \\
&= x \cdots (x-n+2)\big((x+1) - (x-n+1)\big) \\
&= n x^{\underline{n-1}}.
\end{aligned}$$

□

Definition 2.11 *Der Rückwärtsdifferenzenoperator ∇ ist definiert durch*

$$\nabla f(x) := f(x) - f(x-1).$$

Wir nennen ihn auch Nabla-Operator.

Satz 2.12 *Für den Nabla-Operator gilt*

$$\nabla (x)_n = n (x)_{n-1}.$$

Beweis.

$$\begin{aligned}
\nabla (x)_n &= (x)_n - (x-1)_n \\
&= x(x+1)\cdots(x+n-1) - (x-1)x(x+1)\cdots(x+n-2) \\
&= x(x+1)\cdots(x+n-2)\cdot\big(x+n-1-(x-1)\big) \\
&= n\cdot(x)_{n-1}.
\end{aligned}$$

□

Direkt aus der Definition ergibt sich folgender Satz.

Satz 2.13 *Vorwärts- und Rückwärtsdifferenzenoperator stehen in folgender Beziehung zueinander:*

$$\Delta f(x) = \nabla f(x+1).$$

2.1.3 Klassische kontinuierliche und diskrete orthogonale Polynome

Wir werden testen, ob die Algorithmen zur Umwandlung verschiedener Darstellungsformen auch korrekt funktionieren. Dabei haben sich als Testobjekte die klassischen orthogonalen Polynomfamilien als geeignet erwiesen, da sie besonders gut und zahlreich in der Literatur vertreten sind.

Klassische orthogonale Polynome

Definition 2.14 *Die klassischen orthogonalen Polynome sind definiert als polynomiale Lösungen*

$$P_n(x) = k_n x^n + k_n' x^{n-1} + k_n'' x^{n-2} + \ldots \quad \textit{mit} \quad \deg(P_n(x), x) = n$$

der Eigenwertdifferentialgleichung

$$\sigma(x)P_n''(x) + \tau(x)P_n'(x) + \lambda_n P_n(x) = 0, \tag{2.1}$$

wobei $\sigma(x) = ax^2 + bx + c$ *mit* $a, b, c \in \mathbb{K},\ |a| + |b| + |c| \neq 0$ *und* $\tau(x) = dx + e$ *mit* $d, e \in \mathbb{K},\ d \neq 0,\ \lambda_n = -n(a(n-1) + d)$.

In Tabelle 2.1 wollen wir die Klassifizierung der klassischen orthogonalen Polynomsysteme gemäß [Bochner(1929), Lesky(2005)] angeben:

COPS	Notation	Hypergeometrische Funktion
Hermitepolynome	$\mathrm{H}_n(x)$	$(2x)^n {}_2F_0\left(\begin{matrix} -\frac{n}{2}, -\frac{n-1}{2} \\ - \end{matrix} \middle\vert -\frac{1}{x^2}\right)$
Laguerrepolynome	$\mathrm{L}_n^{(\alpha)}(x)$	$\frac{(\alpha+1)_n}{n!} {}_1F_1\left(\begin{matrix} -n \\ \alpha+1 \end{matrix} \middle\vert x\right)$
Besselpolynome	$\mathrm{B}_n^{(a)}(x)$	${}_2F_0\left(\begin{matrix} -n, n+a+1 \\ - \end{matrix} \middle\vert -\frac{x}{2}\right)$
Jacobipolynome	$\mathrm{P}_n^{(\alpha,\beta)}(x)$	$\frac{(\alpha+1)_n}{n!} {}_2F_1\left(\begin{matrix} -n, n+\alpha+\beta+1 \\ \alpha+1 \end{matrix} \middle\vert \frac{1-x}{2}\right)$

Tabelle 2.1: Die klassischen orthogonalen Polynomfamilien als hypergeometrische Funktionen

Wir werden von nun an die abkürzende Schreibweise COPS (für classical orthogonal polynomial system) für die klassischen orthogonalen Polynome verwenden. Analoges zum kontinuierlichen Fall ergibt sich für den diskreten Fall.

Klassische diskrete orthogonale Polynome

Definition 2.15 *Die klassischen diskreten orthogonalen Polynomsysteme sind definiert als polynomiale Lösungen*

$$P_n(x) = k_n x^n + k_n' x^{n-1} + k_n'' x^{n-2} + \ldots \quad \textit{mit} \quad \deg(P_n(x), x) = n$$

der Differenzengleichung

$$\sigma(x)\Delta\nabla P_n(x) + \tau(x)\Delta P_n(x) + \lambda_n P_n(x) = 0, \tag{2.2}$$

wobei $\sigma(x) = ax^2 + bx + c$ *mit* $a, b, c \in \mathbb{K},\ |a| + |b| + |c| \neq 0$ *und* $\tau(x) = dx + e$ *mit* $d, e \in \mathbb{K},\ d \neq 0,\ \lambda_n = -n(a(n-1) + d)$.

Im Folgenden wollen wir die Klassifizierung der klassischen diskreten orthogonalen Polynomsysteme gemäß [Lesky(2005)] angeben.

Wir werden von nun an die abkürzende Schreibweise CDOPS (für classical discrete orthogonal polynomial system) für die klassischen diskreten orthogonalen Polynome verwenden.

CDOPS	Notation	Hypergeometrische Funktion
Charlierpolynome	$\mathrm{C}_n(x;a)$	${}_2F_0\left(\begin{matrix}-n,-x\\-\end{matrix}\middle\vert -\frac{1}{a}\right)$
Meixnerpolynome	$\mathrm{M}_n(x;\beta,c)$	${}_2F_1\left(\begin{matrix}-n,-x\\\beta\end{matrix}\middle\vert 1-\frac{1}{c}\right)$
Krawtchoukpolynome	$\mathrm{K}_n(x;p,N)$	${}_2F_1\left(\begin{matrix}-n,-x\\-N\end{matrix}\middle\vert \frac{1}{p}\right)$
Hahnpolynome	$\mathrm{Q}_n(x;\alpha,\beta,N)$	${}_3F_2\left(\begin{matrix}-n,n+\alpha+\beta+1,-x\\\alpha+1,-N\end{matrix}\middle\vert 1\right)$

Tabelle 2.2: Die klassischen diskreten orthogonalen Polynomfamilien als hypergeometrische Funktionen

2.2 Der q-Fall

2.2.1 Definitionen im q-Fall

Definition 2.16 *Sei $a \in \mathbb{C}$. Die q-basische Zahl $[a]_q$ ist definiert als*

$$[a]_q := \frac{1-q^a}{1-q}.$$

Für $n \in \mathbb{Z}$ ist die q-basische Zahl $[n]_q$ das q-Analogon der Zahl n, da $\lim_{q\to 1}[n]_q = n$.

Definition 2.17 *Sei $n \in \mathbb{N}_0$. Die q-*Fakultät *$[n]_q!$ ist definiert als*

$$[n]_q! := \begin{cases} [n]_q \cdot [n-1]_q \cdot \ldots \cdot [1]_q, & n \in \mathbb{N} \\ 1, & n = 0. \end{cases}$$

Definition 2.18 *Das q-Pochhammersymbol $(a;q)_k$ ist für $a \in \mathbb{C}$ und $k \in \mathbb{N}_0$ definiert als*

$$(a;q)_k := \begin{cases} (1-a)\cdot(1-aq)\cdot\ldots\cdot(1-aq^{k-1}), & k \in \mathbb{N} \\ 1, & k = 0. \end{cases}$$

Wir verwenden in Folge die verkürzende Notation:

$$(a_1,\ldots,a_n;q)_k := (a_1;q)_k \ldots (a_n;q)_k.$$

Das q-Pochhammersymbol $(a;q)_\infty$ ist gegeben durch $(a;q)_\infty := \prod_{i=0}^{\infty}(1-aq^i)$.

Definition 2.19 *Für $n,k \in \mathbb{N}$ ist der q-Binomialkoeffizient definiert als*

$$\begin{bmatrix} n \\ k \end{bmatrix}_q := \frac{[n]_q!}{[k]_q!\,[n-k]_q!}.$$

Diese q-Definitionen können als Verallgemeinerungen der bekannten Begrifflichkeiten aufgefasst werden. Im Grenzprozess erhält man für $q \to 1$ die folgenden Ausdrücke. Wir betrachten die Grenzübergänge für $k, n \in \mathbb{N}_0$ und $\alpha \in \mathbb{C}$ im Einzelnen.

$$\begin{aligned}
\lim_{q\to 1} [n]_q &= \lim_{q\to 1} \frac{1-q^n}{1-q} = \lim_{q\to 1}(1+q+q^2+\cdots+q^{n-1}) = n,\\
\lim_{q\to 1} [n]_q! &= \lim_{q\to 1} [n]_q \cdot [n-1]_q \cdot \ldots \cdot [1]_q = n\cdot(n-1)\cdot\ldots\cdot 1 = n!,\\
\lim_{q\to 1} \begin{bmatrix} n \\ k \end{bmatrix}_q &= \lim_{q\to 1} \frac{[n]_q!}{[k]_q!\cdot[n-k]_q!} = \frac{n!}{k!(n-k)!} = \binom{n}{k},\\
\lim_{q\to 1} \frac{(q^\alpha;q)_k}{(1-q)^k} &= \lim_{q\to 1} \frac{(1-q^\alpha)\cdot(1-q^{\alpha+1})\cdot\ldots\cdot(1-q^{\alpha+k-1})}{(1-q)\cdot(1-q)\cdot\ldots\cdot(1-q)}\\
&= \lim_{q\to 1} [\alpha]_q \cdot [\alpha+1]_q \cdot \ldots \cdot [\alpha+k-1]_q = \alpha\cdot(\alpha+1)\cdot\ldots\cdot(\alpha+k-1)\\
&= (\alpha)_k .
\end{aligned}$$

Satz 2.20 *Die q-Binomialkoeffizienten genügen den q-Pascalregeln*

$$\begin{bmatrix} n+1 \\ k \end{bmatrix}_q = q^k \begin{bmatrix} n \\ k \end{bmatrix}_q + \begin{bmatrix} n \\ k-1 \end{bmatrix}_q \tag{2.3}$$

$$\begin{bmatrix} n+1 \\ k \end{bmatrix}_q = \begin{bmatrix} n \\ k \end{bmatrix}_q + q^{n+1-k} \begin{bmatrix} n \\ k-1 \end{bmatrix}_q . \tag{2.4}$$

Beweis. Wir führen den Beweis wie in [KC(2002), Seite 18] angegeben. Da für $1 \le k \le n$

$$\begin{aligned}
[n+1]_q &= 1+q+\cdots+q^n\\
&= \left(1+q+\cdots+q^{k-1}\right) + q^k\left(1+q+\cdots+q^{n-k}\right)\\
&= [k]_q + q^k\,[n+1-k]_q
\end{aligned}$$

gilt, ergibt sich für

$$\begin{aligned}
\begin{bmatrix} n+1 \\ k \end{bmatrix}_q &= \frac{[n+1]_q!}{[k]_q!\,[n+1-k]_q!} = \frac{[n]_q!\,[n+1]_q}{[k]_q!\,[n+1-k]_q!}\\
&= \frac{[n]_q!([k]_q + q^k\,[n+1-k]_q)}{[k]_q!\,[n+1-k]_q!}
\end{aligned}$$

$$= \frac{[n]_q!}{[k-1]_q!\,[n+1-k]_q!} + q^k \frac{[n]_q!}{[k]_q!\,[n-k]_q!}$$
$$= \begin{bmatrix} n \\ k-1 \end{bmatrix}_q + q^k \begin{bmatrix} n \\ k \end{bmatrix}_q .$$

Dies entspricht (2.3). Aus der Symmetrieeigenschaft $\left[\begin{smallmatrix} n \\ k \end{smallmatrix}\right]_q = \left[\begin{smallmatrix} n \\ n-k \end{smallmatrix}\right]_q$ und dem soeben Gezeigten folgt damit auch (2.4):

$$\begin{bmatrix} n+1 \\ k \end{bmatrix}_q = \begin{bmatrix} n+1 \\ n+1-k \end{bmatrix}_q = \begin{bmatrix} n \\ n-k \end{bmatrix}_q + q^{n+1-k} \begin{bmatrix} n \\ n+1-k \end{bmatrix}_q$$
$$= \begin{bmatrix} n \\ k \end{bmatrix}_q + q^{n+1-k} \begin{bmatrix} n \\ k-1 \end{bmatrix}_q .$$

□

Definition 2.21 *Sei f eine Funktion in x. Der Operator D_q mit*

$$D_q f(x) := \frac{f(x) - f(qx)}{(1-q)x}$$

heißt q-Differentialoperator. Er ist der Hahnoperator mit $w = 0$, denn eingeführt hat ihn [Hahn(1949)]. $D_q f(x)$ nennen wir die q-Ableitung von $f(x)$. Höhere q-Ableitungen sind rekursiv definiert

$$D_q^m f := \begin{cases} D_q D_q^{m-1} f & m \in \mathbb{N} \\ f, & m = 0. \end{cases}$$

Definition 2.22 *Die verallgemeinerte q-hypergeometrische Reihe ist nach [GR(1990)] definiert als*

$${}_r\phi_s\left(\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} \middle| q; x \right) := \sum_{k=0}^{\infty} \frac{(a_1;q)_k \cdot \ldots \cdot (a_r;q)_k}{(b_1;q)_k \cdot \ldots \cdot (b_s;q)_k} \frac{x^k}{(q;q)_k} \left((-1)^k q^{\binom{k}{2}} \right)^{1+s-r} .$$

Darüber hinaus führen wir die folgenden Begriffe ein:

Definition 2.23 *Ein Term a_k heißt q-hypergeometrisch, falls das Termverhältnis $\frac{a_{k+1}}{a_k} \in \mathbb{F}(q^k)$.*

Definition 2.24 *Eine q-holonome Rekursionsgleichung für eine Funktion f in q^n hat die Gestalt*

$$\sum_{k=0}^{m} a_k f(q^{n+k}) = 0,$$

wobei die Polynomkoeffizienten $a_k \in \mathbb{F}[q^n]$.

Definition 2.25 *Eine q-holonome Differentialgleichung für eine Funktion f von x hat die Gestalt*

$$\sum_{k=0}^{m} a_k(x) D_q^k f(x) = 0,$$

wobei die Polynomkoeffizienten $a_k(x) \in \mathbb{F}[q^n, x]$.

2.2.2 Klassische q-orthogonale Polynome

Die klassischen q-orthogonalen Polynome der Hahnklasse sind definiert als polynomiale Lösungen

$$P_n(x) = k_n x^n + k_n' x^{n-1} + k_n'' x^{n-2} + \dots \quad \text{mit} \quad \deg(P_n(x), x) = n$$

der Eigenwertgleichung

$$\sigma(x) D_q D_{q^{-1}} P_n(x) + \tau(x) D_q P_n(x) + \lambda_n P_n(x) = 0, \tag{2.5}$$

wobei $\sigma(x) = ax^2 + bx + c$ mit $a, b, c \in \mathbb{F},\ |a| + |b| + |c| \neq 0$ und $\tau(x) = dx + e$ mit $d, e \in \mathbb{F},\ d \neq 0,\ \lambda_n = -a\,[n]_{q^{-1}}\,[n-1]_q - d\,[n]_q$.
Im Folgenden geben wir eine weitere wichtige Version von Gleichung (2.5) für q$\neq$ 1 an. Dafür berechnen wir zunächst

$$(D_q P_n)(x) = \frac{P_n(x) - P_n(qx)}{x - qx} \tag{2.6}$$

sowie

$$(D_{q^{-1}} P_n)(x) = \frac{P_n(x) - P_n(\frac{x}{q})}{x - \frac{x}{q}} = \frac{q(P_n(\frac{x}{q}) - P_n(x))}{x - qx} \big(:= p_{n-1}(x) \big).$$

Somit ergibt sich

$$D_q D_{q^{-1}} P_n(x) = D_q p_{n-1}(x) = \frac{q^2 P_n(\frac{x}{q}) - q^2 P_n(x) - q P_n(x) + q P_n(qx)}{q(x - qx)^2}. \tag{2.7}$$

Setzen wir (2.6) und (2.7) in (2.5) ein, erhalten wir daher die „symmetrische“ Form

$$C(x) P_n(\frac{x}{q}) - \{C(x) + D(x)\} P_n(x) + D(x) P_n(qx) + \lambda_n P_n(x) = 0$$

mit $C(x) = \frac{\sigma(x) q^2}{q(x-qx)^2},\ D(x) = \frac{C(x)}{q} - \frac{\tau(x)}{x-qx}$.

Die folgenden Definitionen der klassischen q-orthogonalen Polynome über die Darstellung als q-hypergeometrische Funktion sind in [KLS(2010)] zu finden.

Definition 2.26 *Die klassischen q-orthogonalen Polynome der Hahnklasse sind klassifiziert gemäß [Hahn(1949), Lesky(2005)]:*

(a) die Großen q-Jacobipolynome mit

$$\mathrm{P}_n\left(x;a,b,c;q\right) := {}_3\phi_2\left(\left.\begin{matrix} q^{-n}, abq^{n+1}, x \\ aq, cq \end{matrix}\right| q;q\right).$$

Der Spezialfall $a = b = 1$ ergibt die Großen q-Legendrepolynome

$$\mathrm{P}_n\left(x;c;q\right) := {}_3\phi_2\left(\left.\begin{matrix} q^{-n}, q^{n+1}, x \\ q, cq \end{matrix}\right| q;q\right).$$

(b) die Kleinen q-Jacobipolynome mit

$$\mathrm{p}_n\left(x;a,b|q\right) := {}_2\phi_1\left(\left.\begin{matrix} q^{-n}, abq^{n+1} \\ aq \end{matrix}\right| q;qx\right).$$

Der Spezialfall $a = b = 1$ ergibt die Kleinen q-Legendrepolynome

$$\mathrm{p}_n\left(x|q\right) := {}_2\phi_1\left(\left.\begin{matrix} q^{-n}, q^{n+1} \\ q \end{matrix}\right| q;qx\right).$$

(c) die Großen q-Laguerrepolynome mit

$$\mathrm{P}_n\left(x;a,b;q\right) := {}_3\phi_2\left(\left.\begin{matrix} q^{-n}, 0, x \\ aq, bq \end{matrix}\right| q;q\right).$$

(d) die Kleinen q-Laguerrepolynome mit

$$\mathrm{p}_n\left(x;a|q\right) := {}_2\phi_1\left(\left.\begin{matrix} q^{-n}, 0 \\ aq \end{matrix}\right| q;qx\right).$$

(e) die q-Charlierpolynome mit

$$\mathrm{C}_n\left(q^{-x};a;q\right) := {}_2\phi_1\left(\left.\begin{matrix} q^{-n}, q^{-x} \\ 0 \end{matrix}\right| q;-\frac{q^{n+1}}{a}\right).$$

(f) die q-Meixnerpolynome mit

$$\mathrm{M}_n\left(q^{-x};b,c;q\right) := {}_2\phi_1\left(\left.\begin{matrix} q^{-n}, q^{-x} \\ bq \end{matrix}\right| q;-\frac{q^{n+1}}{c}\right).$$

(g) die q-Krawtchoukpolynome mit

$$\mathrm{K}_n\left(q^{-x};p,N;q\right) := {}_3\phi_2\left(\left.\begin{matrix} q^{-n}, q^{-x}, -pq^{n} \\ q^{-N}, 0 \end{matrix}\right| q;q\right) \quad \textit{mit} \quad n = 0,\ldots,N.$$

(h) die q-Hahnpolynome mit

$$\mathrm{Q}_n\left(q^{-x};a,b;N|q\right) := {}_3\phi_2\left(\begin{matrix}q^{-n},abq^{n+1},q^{-x}\\ aq,q^{-N}\end{matrix}\,\middle|\,q;q\right) \quad mit \quad n=0,\ldots,N.$$

(i) die Quantum-q-Krawtchoukpolynome mit

$$\mathrm{K}_n^{qtm}\left(q^{-x};p,N;q\right) := {}_2\phi_1\left(\begin{matrix}q^{-n},q^{-x}\\ q^{-N}\end{matrix}\,\middle|\,q;pq^{n+1}\right) \quad mit \quad n=0,\ldots,N.$$

(j) die Affinen q-Krawtchoukpolynome mit

$$\mathrm{K}_n^{aff}\left(q^{-x};p,N;q\right) := {}_3\phi_2\left(\begin{matrix}q^{-n},q^{-x},0\\ pq,q^{-N}\end{matrix}\,\middle|\,q;q\right) \quad mit \quad n=0,\ldots,N.$$

(k) die q-Besselpolynome mit

$$\mathrm{y}_n\left(q^{x};a;q\right) := {}_2\phi_1\left(\begin{matrix}q^{-n},-aq^{n}\\ 0\end{matrix}\,\middle|\,q;q^{x+1}\right).$$

(l) die Al-Salam-Carlitz I-Polynome mit

$$\mathrm{U}_n^{(a)}(x;q) := (-a)^n q^{\binom{n}{2}}\,{}_2\phi_1\left(\begin{matrix}q^{-n},x^{-1}\\ 0\end{matrix}\,\middle|\,q;\frac{qx}{a}\right).$$

(m) die Al-Salam-Carlitz II-Polynome mit

$$\mathrm{V}_n^{(a)}(x;q) := (-a)^n q^{-\binom{n}{2}}\,{}_2\phi_0\left(\begin{matrix}q^{-n},x\\ -\end{matrix}\,\middle|\,q;\frac{q^n}{a}\right).$$

(n) die Stieltjes-Wigert-Polynome mit

$$\mathrm{S}_n\left(x;q\right) := \frac{1}{(q;q)_n}\,{}_1\phi_1\left(\begin{matrix}q^{-n}\\ 0\end{matrix}\,\middle|\,q;-q^{n+1}x\right).$$

(o) die Diskreten q-Hermite I-Polynome mit

$$\mathrm{h}_n\left(x;q\right) := q^{\binom{n}{2}}\,{}_2\phi_1\left(\begin{matrix}q^{-n},x^{-1}\\ 0\end{matrix}\,\middle|\,q;-qx\right).$$

Die Diskreten q-Hermite I-Polynome sind Al-Salam-Carlitz I-Polynome mit $a=-1$. *Es gilt also* $\mathrm{h}_n\left(x;q\right) = \mathrm{U}_n^{(-1)}(x;q)$.

(p) die Diskreten q-Hermite II-Polynome mit

$$\tilde{\mathrm{h}}_n(x;q) := x^n {}_2\phi_1\left(\begin{matrix} q^{-n}, q^{-n+1} \\ 0 \end{matrix} \,\middle|\, q^2; -\frac{q^2}{x^2}\right).$$

Die Diskreten q-Hermite II-Polynome sind Al-Salam-Carlitz II-Polynome mit $a = -1$. Es gilt $\tilde{\mathrm{h}}_n(x;q) = i^{-n}\,\mathrm{V}_n^{(-1)}(ix;q)$.

Wir werden von nun an die abkürzende Schreibweise q-OPS (für q-orthogonal polynomial system) für die q-orthogonalen Polynome verwenden.

Bemerkung 2.27 *Aus der üblichen Drei-Term-Rekursion*

$$A_n P_{n+1}(x) + B_n P_n(x) + C_n P_{n-1}(x) = x P_n(x)$$

für orthogonale Polynome erhält man durch Umstellen der Terme die Gleichung

$$(x - B_n - A_n - C_n)P_n(x) = A_n P_{n+1}(x) - (A_n + C_n)P_n(x) + C_n P_{n-1}(x),$$

wobei A_n häufig in linearisierter Form auftritt und sich der Koeffizient auf der linken Seite stark vereinfacht.

Kapitel 3

Der Gosper- und der Zeilberger-Algorithmus

3.1 Der Gosper-Algorithmus und der Zeilberger-Algorithmus für die Summation

3.1.1 Der Gosper-Algorithmus für die Summation

Der Gosper-Algorithmus, erstmals vorgestellt in [Gosper(1978)], bestimmt zu einem Term a_k eine unbestimmte Summe $s_k = \sum_k a_k$, die hypergeometrisch ist.

Anders ausgedrückt: gesucht ist eine diskrete Stammfunktion s_k von a_k, die ein hypergeometrischer Term ist, d.h. s_k muss der Gleichung

$$a_k = s_{k+1} - s_k \tag{3.1}$$

genügen und es muss $\frac{s_{k+1}}{s_k} \in \mathbb{K}(k)$ gelten.

Es stellt sich über die folgende Brucherweiterung

$$\frac{a_{k+1}}{a_k} = \frac{s_{k+2} - s_{k+1}}{s_{k+1} - s_k} = \frac{s_{k+1}}{s_k} \frac{\frac{s_{k+2}}{s_{k+1}} - 1}{\frac{s_{k+1}}{s_k} - 1}$$

heraus, dass, wenn s_k hypergeometrisch ist, es auch schon der Eingabeterm a_k sein muss. Somit beschäftigt sich der Gosper-Algorithmus also mit der unbestimmten Summation hypergeometrischer Terme. Es gibt dann zwei Polynome c_k und $d_k \in \mathbb{K}[k]$ mit $\gcd(c_k, d_k) = 1$, so dass

$$\frac{a_{k+1}}{a_k} = \frac{c_k}{d_k} \in \mathbb{K}(k). \tag{3.2}$$

Dabei können an Stelle von a_k auch die beiden Polynome c_k und d_k als Eingabe des Gosper-Algorithmus betrachtet werden.
Existiert eine hypergeometrische diskrete Stammfunktion s_k, so nennen wir a_k gospersummierbar. Wir werden nun in Anlehnung an [Koepf(2014)] die wesentlichen Orakel des Gosper-Algorithmus beschreiben.

Eine hilfreiche Darstellung für $\frac{a_{k+1}}{a_k}$

Nach Lemma 5.1 und Algorithmus 5.2 aus [Koepf(2014)] gibt es für den Eingabeterm a_k Polynome p_k, q_k und $r_k \in \mathbb{K}[k]$, so dass eine Darstellung

$$\frac{a_{k+1}}{a_k} = \frac{p_{k+1}}{p_k}\frac{q_{k+1}}{r_{k+1}} \tag{3.3}$$

gefunden werden kann mit der Eigenschaft

$$\gcd(q_k, r_{k+j}) = 1 \text{ für alle } j \in \mathbb{N}_0. \tag{3.4}$$

In (3.3) steht $\frac{p_{k+1}}{p_k}$ für den polynomiellen Anteil und $\frac{q_{k+1}}{r_{k+1}}$ für den faktoriellen Anteil. Die Konstruktion der Darstellung (3.3) wird dabei wie folgt vorgenommen:
Zunächst einmal setzt man

$$p_k = 1,\ q_k = c_{k-1} \text{ und } r_k = d_{k-1}. \tag{3.5}$$

Ist mit dieser Wahl die Bedingung (3.4) bereits erfüllt, bleibt nichts mehr zu tun. Ansonsten gibt es mindestens ein $j \in \mathbb{N}_0$, so dass (3.4) nicht erfüllt ist. Die Menge all dieser j, für die (3.4) nicht erfüllt ist, bezeichnen wir als Dispersionsmenge J von q_k und r_k. Da Polynome nur endlich viele Nullstellen besitzen, ist J endlich.
Nach Definition von J gilt für jedes $j \in J$:

$$\gcd(q_k, r_{k+j}) = t_k \neq 1 \tag{3.6}$$

Für jedes $j \in J$ werden nun die Polynome p_k, q_k und r_k wie folgt aktualisiert:

$$p'_k = p_k t_k t_{k-1} \cdots t_{k-j+1}, \quad q'_k = \frac{q_k}{t_k}, \quad r'_k = \frac{r_k}{t_{k-j}} \tag{3.7}$$

Wichtig zu erwähnen ist, dass vor den einzelnen Aktualisierungsschritten die Bedingung (3.4) stets neu geprüft wird. Dieses Vorgehen terminiert, da die

Dispersionsmenge J endlich ist. Die Aktualisierung von p_k, q_k und r_k wurde in [Koepf(2014)] mit Hilfe der Mapleprozedur `update` umgesetzt, auf die ich im anschließenden Beispiel zurückgreife.

Der Gradschrankenalgorithmus

Gosper hat schließlich die Funktion f_k definiert als

$$f_k := \frac{s_{k+1}}{a_{k+1}} \cdot \frac{p_{k+1}}{r_{k+1}} \tag{3.8}$$

bzw.

$$s_k = \frac{r_k}{p_k} \cdot f_{k-1} \cdot a_k. \tag{3.9}$$

Daraus ergibt sich unmittelbar

$$a_k = s_{k+1} - s_k = \frac{r_{k+1}}{p_{k+1}} \cdot f_k \cdot a_{k+1} - \frac{r_k}{p_k} \cdot f_{k-1} \cdot a_k.$$

Multiplikation mit $\frac{p_k}{a_k}$ liefert unter Verwendung von (3.3)

$$p_k = \frac{a_{k+1}}{a_k}\frac{p_k}{p_{k+1}} r_{k+1} f_k - r_k f_{k-1} = q_{k+1} f_k - r_k f_{k-1}, \tag{3.10}$$

eine inhomogene lineare Rekursionsgleichung für f_k.
Das Argument

$$f_k = \frac{s_{k+1}}{a_{k+1}} \cdot \frac{p_{k+1}}{r_{k+1}} = \frac{s_{k+1}}{s_{k+2} - s_{k+1}}\frac{p_{k+1}}{r_{k+1}} = \frac{1}{\frac{s_{k+2}}{s_{k+1}} - 1}\frac{p_{k+1}}{r_{k+1}}$$

zeigt, dass f_k rational ist. Dass f_k sogar ein Polynom ist, geht aus Eigenschaft (3.4) hervor. Ein Gradschrankenalgorithmus bestimmt nun in Abhängigkeit von p_k, q_k und r_k eine obere Schranke für den Grad von f_k, vergleiche [Koepf(2014), Lemma 5.4].

Lemma 3.1 *Der folgende Algorithmus findet eine Gradschranke für das Polynom f_k wie in* (3.8) *definiert:*

1. *Gilt* $\deg(q_{k+1} + r_k) \leq \deg(q_{k+1} - r_k)$, *so ist*

$$\deg(f_k) = \deg(p_k) - \deg(q_{k+1} - r_k).$$

2. *Gilt* $m := \deg(q_{k+1} + r_k) > \deg(q_{k+1} - r_k)$ *und bezeichnet* c_1 *den Koeffizienten von* k^m *im Polynom* $q_{k+1} + r_k$ *und* c_2 *den Koeffizienten von* k^{m-1} *im Polynom* $q_{k+1} - r_k$, *so finden die folgenden Fälle Anwendung*

a) Ist $-2c_2/c_1$ keine nichtnegative ganze Zahl ist, so gilt

$$\deg(f_k) = \deg(p_k) - m + 1.$$

b) Ist $-2c_2/c_1$ eine nichtnegative ganze Zahl ist, so gilt

$$\deg(f_k) \leq \max\{-2c_2/c_1, \deg(p_k) - m + 1\}.$$

Lösen eines linearen Gleichungssystems für die Polynomkoeffizienten

Angenommen, die ermittelte Gradschranke ist m. Dann setzen wir das Polynom $\sum_{j=0}^{m} b_j k^j$ für f_k in die Rekursionsgleichung

$$q_{k+1} f_k - r_k f_{k-1} = p_k \tag{3.11}$$

ein und bringen die Terme auf eine Seite. Koeffizientenvergleich bzgl. der Variablen k liefert ein lineares Gleichungssystem, das elementar nach den Koeffizienten $\{b_j\}_{j=0,\ldots,m}$ gelöst wird. Ist f_k bekannt, so kennen wir gemäß (3.9) auch s_k.

3.1.2 Der Zeilberger-Algorithmus für die Summation

Zeilberger hat Anfang der 1990er Jahre erst die Bedeutung des Gosper-Algorithmus erschlossen. Der Zeilberger-Algorithmus ([Zeilberger(1990)], [Zeilberger(1991)]) findet mit Hilfe des Gosper-Algorithmus eine Rekursionsgleichung

$$s_n + \sum_{j=1}^{J} P_j(n) s_{n+j} = 0$$

mit $P_j \in \mathbb{K}(n)$ für s_n, wobei

$$s_n = \sum_{k=-\infty}^{\infty} F(n,k), \tag{3.12}$$

die nach Multiplikation mit dem Hauptnenner holonom wird. $F(n,k)$ muss dabei hypergeometrisch bezüglich n und k sein.
Die direkte Anwendung des Gosper-Algorithmus war nicht zielführend, denn für ein solches $F(n,k)$, das gospersummierbar bezüglich k ist und zudem für alle $n \in \mathbb{N}_0$ wohldefiniert ist und endlichen Träger hat, gilt $\sum_{k=-\infty}^{\infty} F(n,k) = 0$,

siehe [Koepf(2014), Satz 6.1].
Zeilberger hatte nun die Idee, den Gosper-Algorithmus auf den Term

$$a_k = F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k) \tag{3.13}$$

für ein geeignetes J anzuwenden.
Im Wesentlichen werden nun die Schritte des Gosper-Algorithmus für dieses a_k durchgeführt, was möglich ist, da (3.13) hypergeometrisch ist:

$$\frac{a_{k+1}}{a_k} = \frac{F(n,k+1) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k+1)}{F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k)} \tag{3.14}$$

$$= \frac{F(n,k+1)}{F(n,k)} \cdot \frac{1 + \sum_{j=1}^{J} \sigma_j(n) \frac{F(n+j,k+1)}{F(n,k+1)}}{1 + \sum_{j=1}^{J} \sigma_j(n) \frac{F(n+j,k)}{F(n,k)}}. \tag{3.15}$$

Anzumerken ist, dass die Dispersionsmenge mindestens den Wert 1 enthält, da es sich bei den Variablen $\{\sigma_j\}_{j=1,\dots,J}$ um Unbekannte handelt, so dass diese Variablen nach der Aktualisierung linear in p_k auftauchen. Deshalb kann nun ein lineares Gleichungssystem nach den Koeffizienten des generischen Polynoms f_k und gleichzeitig nach den k-freien Variablen $\{\sigma_j\}_{j=1,\dots,J} \subset \mathbb{K}(n)$ aufgelöst werden. Ist der Gosper-Algorithmus erfolgreich, erhalten wir einen hypergeometrischen Term $G(n,k)$ und $\{\sigma_j\}_{j=1,\dots,J} \subset \mathbb{K}(n)$, für die gilt:

$$G(n,k+1) - G(n,k) = a_k = F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k).$$

Summation liefert nach (3.12)

$$\begin{aligned} 0 = \sum_{k=-\infty}^{\infty} \big(G(n,k+1) - G(n,k)\big) &= \sum_{k=-\infty}^{\infty} a_k \\ &= \sum_{k=-\infty}^{\infty} \left(F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k) \right) \\ &= s_n + \sum_{j=1}^{J} \sigma_j(n) s_{n+j}, \end{aligned}$$

wobei die Null auf der linken Seite durch eine Teleskopsumme erzeugt wird. Nach Multiplikation mit dem Hauptnenner ist eine holonome Rekursionsgleichung der Ordnung J für s_n bestimmt.

Die von Wolfram Koepf in [Koepf(2014)] veröffentlichte Mapleprozedur `sumrecursion` findet eine holonome Rekursionsgleichung für s_n gemäß dem Zeilberger-Algorithmus. Auf diese Prozedur werde ich in meiner Arbeit zurückgreifen.

3.1.3 Der Zeilberger-Algorithmus für die Summation am Beispiel

Wir wollen nun die wesentlichen Schritte des Zeilberger-Algorithmus am Beispiel von

$$F(n,k) = \binom{n}{k}^2 \tag{3.16}$$

vorstellen und als Maple-Code wiedergeben.
Wir definieren

```
> F:=binomial(n,k)^2:
```

und bilden zunächst a_k wie in (3.13) definiert:

```
> a_k:=F+sigma[1]*subs(n=n+1,F);
```

$$a\_k := \binom{n}{k}^2 + \sigma_1 \binom{n+1}{k}^2$$

Wir bilden hiervon das Termverhältnis gemäß (3.15):

```
> rat:=ratio(a_k,k);
```

$$rat := \frac{(-n-1+k)^2\,(n^2-2\,nk+k^2+\sigma_1 n^2+2\,\sigma_1 n+\sigma_1)}{(n^2+2\,n-2\,nk+1-2\,k+k^2+\sigma_1 n^2+2\,\sigma_1 n+\sigma_1)\,(k+1)^2}$$

Nun bestimmen wir p_k, q_k und r_k wie in (3.5) festgelegt:

```
> p:=1:
> q:=subs(k=k-1,numer(rat)):
> r:=subs(k=k-1,denom(rat)):
```

und aktualisieren die Wahl von p_k, q_k und r_k mit der Prozedur `update` aus [Koepf(2014)], damit die Dispersionsmenge leer wird:

```
> upd:=update(p,q,r,k);
```

$$upd := [n^2+2\,n-2\,nk+1-2\,k+k^2+\sigma_1 n^2+2\,\sigma_1 n+\sigma_1, (-n-2+k)^2, k^2]$$

```
> p:=op(1,upd):
> q:=op(2,upd):
```

```
> r:=op(3,upd):
```

Wir bestimmen nun in Abhängigkeit dieser neu erzeugten p_k, q_k und r_k eine Gradschranke für f_k mit Hilfe des von Wolfram Koepf in [Koepf(2014)] implementierten Algorithmus

```
> deg:=degreebound(p,q,r,k);
```

$$deg := 1$$

und können daher ein generisches Polynom vom Grad 1 für f_k ansetzen:

```
> f_k:=b[0]+b[1]*k:
```

Wir setzen dieses f_k in (3.11) ein und bringen die Terme auf die linke Seite, so dass ein homogenes Gleichungssystem nach den Variablen

```
> var:={sigma[1],b[0],b[1]}:
```

zu lösen ist:

```
> rec:=collect(subs(k=k+1,q)*f_k-r*subs(k=k-1,f_k)-p,k);
```

$$rec := \left(-1+b_1+\left(-2\,n-2\right)b_1\right)k^2+\left(2+2\,n+\left(-n-1\right)^2b_1+\left(-2\,n-2\right)b_0\right)k$$
$$+\left(-n-1\right)^2b_0-\sigma_1-n^2-2\,n-\sigma_1n^2-1-2\,\sigma_1n$$

```
> sol:=solve({coeffs(rec,k)},var);
```

$$sol := \left\{b_0=1/2\,\frac{3\,n+1}{2\,n+1},b_1=-\left(2\,n+1\right)^{-1},\sigma_1=-1/2\,\frac{n+1}{2\,n+1}\right\}$$

Interessant ist für uns nur $\sigma_1(n) = -\frac{1}{2}\frac{n+1}{2n+1}$. Wir setzen $\sigma_1(n)$ in $s_n+\sigma_1(n)s_{n+1}=0$ ein und erhalten nach Multiplikation mit dem Hauptnenner

$$(4n+1)s_n-(n+1)s_{n+1}=0.$$

Umformung liefert das Termverhältnis

$$\frac{s_{n+1}}{s_n}=\frac{4(n+\frac{1}{2})}{n+1}.$$

Mit $s_0=\sum_{k=0}^{0}\binom{0}{k}^2=1$ erhält man hieraus

$$s_n=\frac{\left(\frac{1}{2}\right)_n 4^n}{n!}.$$

Wir schreiben dies als

$$s_n=\frac{4^n\left(1+3+\ldots+(2n-1)\right)}{2^n n!}$$

und erweitern mit den geraden Zahlen bis $2n$. Dies liefert

$$s_n=\frac{(2n)!}{n!n!}=\binom{2n}{n}.$$

Insgesamt haben wir damit gezeigt, dass $\sum_{k=-\infty}^{\infty}\binom{n}{k}^2=\sum_{k=0}^{n}\binom{n}{k}^2=\binom{2n}{n}$ ist.

3.1.4 Eine Variante des Zeilberger-Algorithmus: Differentialgleichungen für Summen

An Stelle einer holonomen Rekursionsgleichung wie in Abschnitt 3.1.2 suchen wir nun eine holonome Differentialgleichung

$$S^{(J)}(x) + \sum_{j=0}^{J-1} \sigma_j(x) S^{(j)}(x) = 0 \tag{3.17}$$

für

$$S(x) := \sum_{k=-\infty}^{\infty} F(x,k). \tag{3.18}$$

Eingabeterm für diese Variante des Zeilberger-Algorithmus ist der Term $F(x,k)$, der hypergeometrisch bezüglich k und hyperexponentiell bezüglich x sein muss. Es soll also gelten

$$\frac{F(x,k+1)}{F(x,k)} \in \mathbb{K}(x,k) \text{ sowie } \frac{F'(x,k)}{F(x,k)} \in \mathbb{K}(x,k).$$

Laut Lemma 10.1 aus [Koepf(2014)] hat $S(x)$ die Darstellung

$$S(x) = C \cdot e^{R(x)} \cdot \sqrt[m]{T(x)} \tag{3.19}$$

für $C \in \mathbb{K}$, R, $T \in \mathbb{K}(x)$ und $m \in \mathbb{N}$.

Für unsere Anwendungen genügt es, Gleichung (3.19) mit $m = 1$ zu betrachten. Dadurch ist sichergestellt, dass wir tatsächlich eine holonome Differentialgleichung erhalten. Während im Zeilberger-Algorithmus aus Abschnitt 3.1.2 endliche Summen betrachtet wurden, haben wir es im Falle von (3.18) mit unendlichen Summen zu tun. Deshalb muss $S(x)$ in geeigneter Weise kompakt konvergent sein, zum Begriff der kompakten Konvergenz siehe [BS(1976), Seite 86].

Nun wenden wir den Gosper-Algorithmus für die Summation auf den Term

$$a_k = F^{(J)}(x,k) + \sum_{j=0}^{J-1} \sigma_j(x) F^{(j)}(x,k) \tag{3.20}$$

an, wobei wir mit $J = 1$ beginnen. Wir erhalten eine hypergeometrische diskrete Stammfunktion $G(x,k)$, falls eine solche existiert. Diese genügt der Gleichung

$$a_k = G(x,k+1) - G(x,k). \tag{3.21}$$

Das Gleichungssystem, das im Gosper-Algorithmus nach den Koeffizienten eines Polynoms gelöst werden muss, wird nun zusätzlich nach den k-freien Variablen $\{\sigma_j(x)\}_{j=0,\dots,J-1} \subset \mathbb{K}(x)$ aufgelöst. Falls der Gosper-Algorithmus für das J aus (3.20) keine diskrete Stammfunktion gefunden hat, so wird J jeweils um 1 erhöht. Falls der Eingabeterm ein „zulässiger“ Term ist, so wird nach Korollar 7.11 in [Koepf(2014)], siehe auch [WZ(1992), GKP(1994)], ein solches J gefunden. In den von uns betrachteten Fällen sind die Eingabeterme „zulässig“. Da wir die $\{\sigma_j(x)\}_{j=0,\dots,J-1} \subset \mathbb{K}(x)$ nun kennen, brauchen wir nur noch über Gleichung (3.21) gemäß Definition (3.18) zu summieren und erhalten $S^{(J)}(x) + \sum_{j=0}^{J-1} \sigma_j(x) S^{(j)}(x) = 0$. Die rechte Seite der Differentialgleichung wird zu einer Teleskopsumme, die gegen 0 konvergiert, da $S(x)$ kompakt konvergiert. Nachdem wir außerdem mit dem kleinsten gemeinsamen Nennervielfachen multipliziert haben, erhalten wir die gesuchte Differentialgleichung. Falls $F(x,k)$ strikt hyperexponentiell bezüglich x ist, siehe [Koepf(2014), Lemma 10.1], haben wir sogar eine holonome Differentialgleichung gefunden. Dabei heißt $F(x,k)$ strikt hyperexponentiell, falls in Gleichung (3.19) der Fall $m = 1$ eintritt. Diese Variante des Zeilberger-Algorithmus wurde in [Koepf(2014)] als Prozedur `sumdiffeq` umgesetzt, die wir später verwenden werden.

3.2 Der Gosper-Algorithmus und der Zeilberger-Algorithmus für die Integration

Analog zum Gosper- und Zeilberger-Algorithmus für die Summation stellen wir nun den Gosper- und Zeilberger-Algorithmus für die Integration dar.

3.2.1 Der Gosper-Algorithmus für die Integration

Im stetigen Fall findet der Gosper-Algorithmus eine hyperexponentielle Stammfunktion $G(x)$ für eine Eingabefunktion $f(x)$, sofern diese existiert, siehe [Koepf(2014), Kapitel 11] bzw. [AZ(1990), Abschnitt 5]. Analog zum diskreten Fall muss bereits die Eingabefunktion hyperexponentiell sein, denn $G'(x)$ erfüllt die Gleichung

$$G'(x) = R(x)G(x) \text{ für ein } R(x) \in \mathbb{K}(x). \tag{3.22}$$

Mit der Produktregel folgt hieraus

$$G''(x) = R'(x)G(x) + R(x)G'(x) = \left(\frac{R'(x)}{R(x)} + R(x)\right)G'(x) \tag{3.23}$$

Und mit (3.22) und (3.23) erhalten wir, dass

$$\frac{f'(x)}{f(x)} = \frac{G''(x)}{G'(x)} = \frac{R'(x)}{R(x)} + R(x)$$

eine rationale Funktion ist.
Der Algorithmus für die Integration läuft im Wesentlichen wie der für die Summation ab. Anstatt des Termverhältnisses a_{k+1}/a_k wird nunmehr der Quotient $f'(x)/f(x)$ bezüglich des Eingabeterms $f(x) \neq 0$ betrachtet. Da $f(x)$ eine Darstellung der Form (3.19) besitzt, lässt sich dieser Quotient nach dem Kürzen der exponentiellen Terme mit Hilfe von Polynomen $c(x)$ und $d(x) \in \mathbb{K}[x]$, $\gcd(c(x), d(x)) = 1$ darstellen als

$$\frac{f'(x)}{f(x)} = \frac{c(x)}{d(x)}.$$

Danach werden die folgenden Schritte ausgeführt, die denen in Abschnitt 3.1.1 entsprechen:

1. Es werden Polynome $p(x)$, $q(x)$ und $r(x)$ berechnet, die der Gleichung

   $$\frac{f'(x)}{f(x)} = \frac{p'(x)}{p(x)} + \frac{q(x)}{r(x)},$$

   genügen und für die zudem

   $$\gcd\big(r(x), q(x) - jr'(x)\big) = 1 \text{ für alle } j \in \mathbb{N}_0 \tag{3.24}$$

   gilt, vergleiche dazu (3.3) und (3.4) in Abschnitt 3.1.1, indem zuerst einmal

   $$p(x) = 1,\ q(x) = c(x) \text{ und } r(x) = d(x)$$

   gesetzt wird.
   Sobald es ein $j \in \mathbb{N}_0$ gibt mit $g(x) = \gcd\big(r(x), q(x) - jr'(x)\big) \neq 1$, werden die Polynome $p(x)$, $q(x)$ und $r(x)$ wie folgt aktualisiert:

   $$\tilde{p}(x) = p(x)g(x)^j$$
   $$\tilde{q}(x) = j\frac{d}{dx}\left(\frac{r(x)}{g(x)}\right) + \frac{q(x) - jr'(x)}{g(x)}$$

$$\tilde{r}(x) = \frac{r(x)}{g(x)}$$

Dies erinnert an die Vorgehensweise nach (3.7) im Summationsfall.

2. Im nächsten Schritt wird analog zum diskreten Gradschrankenalgorithmus in Abschnitt 3.1.1 eine Gradschranke für das Polynom $\tilde{f}(x) \in \mathbb{K}[x]$ gefunden, das über die Gleichung

$$G(x) = \frac{r(x)\tilde{f}(x)}{p(x)} f(x) \tag{3.25}$$

definiert ist. Ist $m < 0$, existiert keine hyperexponentielle Stammfunktion des Eingabetermes.

3. Schließlich wird das generische Polynom $\sum_{j=0}^{m} b_j k^j$ für $\tilde{f}(x)$ in die Funktionalgleichung

$$p(x) = \Big(q(x) + r'(x)\Big)\tilde{f}(x) + r(x)\tilde{f}'(x)$$

eingesetzt. Nach Koeffizientenvergleich ist ein lineares Gleichungssystem nach den Koeffizienten $\{b_j\}_{j=1,\ldots,m}$ zu lösen, so dass dadurch $\tilde{f}(x)$ bestimmt ist und mittels Gleichung (3.25) auch die hyperexponentielle Stammfunktion. Hat das lineare Gleichungssystem keine Lösung, existiert keine hyperexponentielle Stammfunktion $G(x)$.

3.2.2 Der Zeilberger-Algorithmus für die Integration

Der Zeilberger-Algorithmus für die Integration sucht eine Rekursionsgleichung für I_n, wobei

$$I_n = \int_a^b F(n,t)dt, \tag{3.26}$$

falls das unbestimmte Integral über diesen Integranden kein hyperexponentieller Term ist. In diesem Fall wäre die bestimmte Integration Dank des Hauptsatzes der Differential- und Integralrechnung trivial. Almkvist und Zeilberger, [AZ(1990)], waren es, die sich diesem Problem erstmals widmeten. Wie im

diskreten Fall ist der Eingabeterm des stetigen Zeilberger-Algorithmus' ein $F(n,t) \neq 0$, das hypergeometrisch bezüglich n und hyperexponentiell bezüglich t sein muss. Für ein $J \in \mathbb{N}$ wird nun der stetige Gosper-Algorithmus auf den Eingabeterm

$$f(t) := F(n,t) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,t) \tag{3.27}$$

angewandt, so dass nach Koeffizientenvergleich nicht mehr nur nach den Koeffizienten des generischen Polynoms, sondern zusätzlich nach den Variablen $\{\sigma_j(n)\}_{j=1,\ldots,J} \subset K(n)$ aufgelöst wird. Der Gosper-Algorithmus findet dann eine hyperexponentielle Stammfunktion $G(n,t)$ mit

$$\frac{d}{dt} G(n,t) = f(t).$$

Integration von $t = a$ bis $t = b$ über Gleichung (3.27) liefert nach dem Hauptsatz der Differential- und Integralrechnung

$$I_n + \sum_{j=1}^{J} \sigma_j(n) I_{n+j} = G(n,b) - G(n,a),$$

so dass wir für geeignete Integrationsgrenzen und nach Multiplikation mit dem kleinsten gemeinsamen Nennervielfachen eine holonome Rekursionsgleichung erhalten.

Dieser Algorithmus wurde in [Koepf(2014)] als Prozedur `intrecursion` umgesetzt, worauf ich mich in meiner Arbeit beziehen werde.

3.2.3 Eine Variante des Zeilberger-Algorithmus: Differentialgleichungen für Integrale

Während wir im vorhergehenden Abschnitt Rekursionsgleichungen für Integrale gesucht haben, werden wir nun Differentialgleichungen für Integrale bestimmen. Wir definieren an Stelle von (3.26) aus Abschnitt 3.2.2 das Integral

$$I(x) = \int_a^b F(x,t)dt. \tag{3.28}$$

Dabei muss $F(x,t)$ hyperexponentiell bezüglich x und t sein. Beginnend bei $J = 1$ wird nun der stetige Gosper-Algorithmus auf den Term

$$f(t) := F(x,t) + \sum_{j=1}^{J} \sigma_j(x) \frac{\partial^j}{\partial x^j} F(x,t) \tag{3.29}$$

angewandt, vergleiche (3.27) im vorhergehenden Abschnitt. Wir lösen das lineare Gleichungssystem im Gosper-Algorithmus nun nicht mehr nur nach den Koeffizienten eines Polynoms, sondern gleichzeitig nach den Variablen $\{\sigma_j(x)\}_{j=1,\dots,J} \subset \mathbb{K}(x)$, die nicht von t abhängen. Damit erhalten wir neben der Menge $\{\sigma_j(x)\}_{j=1,\dots,J} \subset \mathbb{K}(x)$ eine hyperexponentielle Stammfunktion $G(n,t)$ für $f(t)$. Integration von $t=a$ bis $t=b$ über Gleichung (3.29) liefert nach dem Hauptsatz der Differential- und Integralrechnung

$$I(x) + \sum_{j=1}^{J} \sigma_j(x) I^{(j)}(x) = G(x,b) - G(x,a).$$

Für geeignete Integrationsgrenzen und nach Multiplikation mit dem kleinsten gemeinsamen Nennervielfachen erhalten wir die gesuchte holonome Differentialgleichung. Die soeben vorgestellte Variante des Zeilberger-Algorithmus wurde in [Koepf(2014)] als Prozedur `intdiffeq` umgesetzt. Diese Prozedur werden wir später verwenden.

3.3 Der Gosper-Algorithmus und der Zeilberger-Algorithmus für den q-Fall

Analog zum Gosper- und Zeilberger-Algorithmus im Diskreten stellen wir nun den Gosper- und Zeilberger-Algorithmus für den q-Fall dar, wobei an Stelle von $\mathbb{K}$ der Grundkörper $\mathbb{F} = \mathbb{K}(q)$ betrachtet wird.

3.3.1 Der q-Gosper-Algorithmus

Der q-Gosper-Algorithmus findet zu einem q-hypergeometrischen Eingabeterm a_k einen q-hypergeometrischen Term s_k mit

$$a_k = s_{k+1} - s_k. \tag{3.30}$$

Wir nennen s_k dann auch q-hypergeometrische Stammfunktion von a_k. Wir geben nun die wesentlichen Schritte des q-Gosper-Algorithmus an, wobei wir uns an den Darstellungen in [Koornwinder(1993)] und [BK(1999), Böing(1998)] orientieren.

Wird Gleichung (3.30) durch s_k geteilt, sieht man, dass $\frac{s_k}{a_k} \in \mathbb{F}(q^k)$. Andererseits liefert (3.30) geteilt durch a_k nach entsprechendem Erweitern die Gleichung

$$\frac{a_{k+1}}{a_k}\frac{s_{k+1}}{a_{k+1}} - \frac{s_k}{a_k} = 1 \tag{3.31}$$

Diese Gleichung lässt sich vereinfacht schreiben als

$$h_k c_{k+1} - c_k = 1, \tag{3.32}$$

wobei $c_k = \frac{s_k}{a_k} \in \mathbb{F}(q^k)$. Um eine q-hypergeometrische Stammfunktion s_k für den Eingabeterm a_k zu finden, genügt es also, eine rationale Lösung c_k der inhomogenen Rekursionsgleichung 1. Ordnung (3.32) zu finden.
Wir werden nun in Analogie zu Abschnitt 3.1.1 die wesentlichen Orakel des q-Gosper-Algorithmus angeben:

Eine hilfreiche Darstellung für $\frac{a_{k+1}}{a_k}$

Zunächst werden Polynome p_k, q_k und $r_k \in \mathbb{F}[q^k]$ bestimmt, die die Gleichung

$$h_k = \frac{a_{k+1}}{a_k} = \frac{p_{k+1}}{p_k}\frac{q_k}{r_k} \tag{3.33}$$

erfüllen und für die außerdem

$$\gcd(q_k, r_{k+j}) = 1 \text{ für alle } j \in \mathbb{N}_0 \tag{3.34}$$

sowie

$$\gcd(q_{k+1}, p_k) = 1 = \gcd(r_k, p_k) \tag{3.35}$$

gelten sollen.
Die Polynome p_k, q_k und r_k sind durch diese Eigenschaften bis auf Einheiten eindeutig bestimmt.
Gleichung (3.30) lässt sich auf Grund der Schreibweise (3.33) unter Beachtung von (3.34) und (3.35) darstellen als

$$q_{k+1} f_k - r_k f_{k-1} = p_k. \tag{3.36}$$

Diese inhomogene Rekursionsgleichung 1. Ordnung ist uns als Gleichung (3.11) bereits in Abschnitt 3.1.1 begegnet. Es stellte sich an dieser Stelle heraus, dass f_k ein Polynom sein muss. Eine ähnliche Argumentation liefert nun für den q-Fall, dass $f_k \in \mathbb{F}[q^k, q^{-k}]$ sein muss. Wegen $q^{-k} = \frac{1}{q^k}$ handelt es sich somit um ein Laurentpolynom, vergleiche [Koornwinder(1993), Seite 105].

Definition 3.2 *Sei* $f_k \in \mathbb{F}[q^k, q^{-k}]$ *ein Laurentpolynom, d.h.* $f_k := \sum_{j=m_1}^{m_2} a_j q^{jk}$; $m_1, m_2 \in \mathbb{Z}$, $m_1 \leq m_2$, $a_{m_1} \neq 0 \neq a_{m_2}$. *Dann sei* $\deg(f_k) := m_2$ *und* $\operatorname{ldeg}(f_k) := m_1$.

Eine untere und obere Gradschranke für das Laurentpolynom

Wir wollen nun mit folgendem Lemma, vergleiche dazu [Koornwinder(1993), Lemma 5.1], einen Algorithmus angeben, der aus der Kenntnis von p_k, q_k und r_k in Gleichung (3.36) eine untere und obere Gradschranke für das Laurentpolynom $f_k \in \mathbb{F}[q^k, q^{-k}]$ berechnet.

Lemma 3.3 *Sei* $f_k \in \mathbb{F}[q^k, q^{-k}]$, $f_k \neq 0$ *eine Lösung von* (3.36). *Dann werden bei der Berechnung der unteren und oberen Gradschranke von* f_k *folgende Fälle unterschieden:*

1. *Gilt* $\operatorname{ldeg} q_k \neq \operatorname{ldeg} r_k$, *so ist*

$$\operatorname{ldeg}(f_k) = \operatorname{ldeg}(p_k) - \min\{\operatorname{ldeg}(q_k), \operatorname{ldeg}(r_k)\}.$$

2. *Ist* $l := \deg(q_k) = \operatorname{ldeg}(r_k)$ *und seien* d_l *und* e_l *die Koeffizienten von* q^{kl} *in* q_k *bzw.* r_k, *so finden die folgenden Fälle Anwendung:*

   a) *Falls* $\log_q(\frac{e_l}{d_l}) \notin \mathbb{Z}$, *so gilt*

$$\operatorname{ldeg}(f_k) = \operatorname{ldeg}(p_k) - l.$$

   b) *Falls* $\log_q(\frac{e_l}{d_l}) \in \mathbb{Z}$, *so gilt*

$$\operatorname{ldeg}(f_k) \geq \min\{log_q(\frac{e_l}{d_l}), \operatorname{ldeg}(p_k)\} - l.$$

Für die obere Gradschranke von f_k *müssen folgende Fälle unterschieden werden:*

1.' *Gilt* $\deg(q_{k+1}) \neq \deg(r_k)$, *so ist*

$$\deg(f_k) = \deg(p_k) - \max\{\deg(q_k), \deg(r_k)\}.$$

2.' *Ist* $m := \deg(q_k) = \deg(r_k)$ *und seien* d_m *und* e_m *die Koeffizienten von* q^{km} *in* q_k *bzw.* r_k, *so finden die folgenden Fälle Anwendung:*

a)' Falls $\log_q(\frac{e_m}{d_m}) \notin \mathbb{Z}$, *so gilt*

$$\deg(f_k) = \deg(p_k) - m.$$

b)' Falls $\log_q(\frac{e_m}{d_m}) \in \mathbb{Z}$, *so gilt*

$$\deg(f_k) \leq \max\{\log_q(\frac{e_m}{d_m}), \deg(p_k)\} - m.$$

Lösen eines linearen Gleichungssystems für die Polynomkoeffizienten

Wir setzen nun das Laurentpolynom $\sum_{j=\mathrm{ldeg}(f_k)}^{\deg(f_k)} a_j k^j$ für f_k in die Rekursionsgleichung

$$q_{k+1} f_k - r_k f_{k-1} = p_k \tag{3.37}$$

ein, wobei im generischen Laurentpolynom q^k durch K substituiert wurde. Danach bringen wir die Terme auf eine Seite. Koeffizientenvergleich bzgl. K liefert ein lineares Gleichungssystem. Dieses kann elementar nach den Koeffizienten $\{a_j\}_{j=\mathrm{ldeg}(f_k),\ldots,\deg(f_k)}$ gelöst werden. Ist das Gleichungssystem nicht lösbar, gibt es keine q-hypergeometrische Stammfunktion s_k. Andernfalls kennen wir mit f_k gemäß der Gleichung

$$s_k = \frac{r_k}{p_k} \cdot f_{k-1} \cdot a_k \tag{3.38}$$

auch s_k, vergleiche Gleichung (3.9) in Abschnitt 3.1.1.

3.3.2 Der q-Zeilberger-Algorithmus

Wir beschreiben nun die wesentlichen Schritte des q-Zeilberger-Algorithmus entsprechend der Darstellung in [BK(1999), Böing(1998)]. Gesucht ist die Summe

$$s_n = \sum_{k=-\infty}^{\infty} F(n,k). \tag{3.39}$$

$F(n,k) \in \mathbb{F}$ muss dabei q-hypergeometrisch bezüglich n und k sein.
Wie schon in Abschnitt 3.1.2 findet der Gosper-Algorithmus für ein geeignetes J Anwendung auf den Term

$$a_k = F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k), \tag{3.40}$$

was möglich ist, da a_k q-hypergeometrisch ist nach einem q-Analogon von Gleichung (3.15).
Nachdem p_k, q_k und r_k gemäß (3.33) und ebenso eine obere und untere Gradschranke für das generische Laurentpolynom, das Lösung einer inhomogenen Rekursionsgleichung 1. Ordnung ist, gefunden wurden, wird nun der generische Ansatz für das Laurentpolynom in die Rekursionsgleichung eingesetzt und nach Koeffizientenvergleich ist ein lineares Gleichungssystem zu lösen. Ähnlich wie in Abschnitt 3.1.2 ist nicht mehr nur ein Gleichungssystem für die Koeffizienten des generischen Laurentpolynoms zu lösen, sondern gleichzeitig wird auch noch nach den k-freien Variablen $\{\sigma_j\}_{j=1,\dots,J} \subset \mathbb{F}(q^n)$ aufgelöst. Der q-Gosper-Algorithmus liefert eine q-hypergeometrische Stammfunktion $G(n,k)$ und $\{\sigma_j\}_{j=1,\dots,J} \subset \mathbb{F}(q^n)$, für die gilt:

$$G(n,k+1) - G(n,k) = a_k = F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k).$$

Summation liefert nach (3.39)

$$\begin{aligned} 0 = \sum_{k=-\infty}^{\infty} \big(G(n,k+1) - G(n,k)\big) &= \sum_{k=-\infty}^{\infty} a_k \\ &= \sum_{k=-\infty}^{\infty} \left(F(n,k) + \sum_{j=1}^{J} \sigma_j(n) F(n+j,k) \right) \\ &= s_n + \sum_{j=1}^{J} \sigma_j(n) s_{n+j}, \end{aligned}$$

wobei die Null auf der linken Seite durch eine Teleskopsumme erzeugt wird. Nach Multiplikation mit dem Hauptnenner ist eine q-holonome Rekursionsgleichung der Ordnung J für s_n bestimmt.
Die von Harald Böing und Wolfram Koepf in [BK(1999)] veröffentlichte Maple-prozedur `qsumrecursion` findet eine q-holonome Rekursionsgleichung für s_n gemäß dem q-Zeilberger-Algorithmus. Wir werden später darauf zurückgreifen.

Kapitel 4

Identifikation von Funktionenfamilien über holonome Rekursions- und Differential- bzw. Differenzengleichungen

Im vorliegenden Kapitel wollen wir Algorithmen vorstellen, um Funktionenfamilien zu identifizieren, die über eine Rodriguesformel gegeben sind, indem wir aus dieser Darstellung Rekursionsgleichungen (oder auch Differential- bzw. (q-)Differenzengleichungen) erzeugen und diese dann mit der aus der hypergeometrischen Summendarstellung erzeugten Rekursionsgleichung abgleichen. Nachdem auch noch entsprechend viele Anfangswerte überprüft worden sind, ist die Darstellung einer Funktionenfamilie über eine Rodriguesformel verifiziert.

Zunächst wollen wir uns den bereits in der Literatur dokumentierten Fall anschauen, in dem aus Rodriguesformeln stetiger Funktionenfamilien holonome Rekursionsgleichungen als auch holonome Differentialgleichungen erzeugt werden. Dabei wird die Rodriguesformel in eine Integralschreibweise überführt, auf die dann der stetige Zeilberger-Algorithmus Anwendung findet. Wir testen die Algorithmen am Beispiel der klassischen orthogonalen Polynome und vergleichen mit den Rekursions- bzw. Differentialgleichungen, die aus der hypergeometrischen Summendarstellung gewonnen werden können.

Darüber hinaus testen wir noch einmal an einer Beispielfamilie, die keine orthogonale Polynomfamilie ist.
Danach werden wir die Vorgehensweise auf den Fall diskreter Funktionenfamilien übertragen und auch hierfür Algorithmen vorstellen, die aus diskreten Rodriguesformeln holonome Rekursionsgleichungen als auch holonome Differenzengleichungen generieren. Dafür muss zunächst eine gleichwertige Summendarstellung gefunden werden. Wir wandeln die Rodriguesformel in Summenschreibweise um und wenden den diskreten Zeilberger-Algorithmus darauf an. Die gefundenen Algorithmen werden an Hand der klassischen diskreten orthogonalen Polynome getestet und wieder mit den Ergebnissen bezüglich der hypergeometrischen Summendarstellung verglichen.
Die Problemstellung wird schließlich auf den q-Fall übertragen. Im q-Fall sind in der Literatur, etwa in [KLS(2010)], verschiedene Arten von q-Rodriguesformeln verwendet worden. Für diese muss entsprechend eine Summendarstellung gefunden werden, so dass der q-Zeilberger-Algorithmus Anwendung finden kann. In diesem Fall werden die Algorithmen mit Hilfe der klassischen q-orthogonalen Polynome getestet und mit der aus der q-hypergeometrischen Summendarstellung generierten q-Rekursionsgleichung bzw. q-Differenzengleichung verglichen. Zur Orientierung für den Leser geben wir in Tabelle 4.1 eine Übersicht über die Namen der Mapleprozeduren für alle Testfälle an.

4.1 Holonome Rekursionsgleichungen und Differentialgleichungen aus Rodriguesformeln im stetigen Fall

4.1.1 Holonome Rekursionsgleichungen aus Rodriguesformeln im stetigen Fall

Wir wollen nun stetige Funktionenfamilien identifizieren, die durch eine Rodriguesformel gegeben sind. Dabei werden wir Rekursionsgleichungen aus Rodriguesformeln mit Hilfe des stetigen Zeilberger-Algorithmus generieren. Darauf hatten bereits [AZ(1990)] hingewiesen. Wir werden dies nun ausführlich darstellen, so wie es [Koepf(2014), Kapitel 13] getan hat.

	stetig	diskret	q-diskret
Rekursionsgleichung	`rodriguesrecursion`	`rodriguesDeltarec` (`rodriguesNablarec`)	1. `rodriguesDqrec` 2. `rodriguesDqinversrec` 3. `rodriguesNablaqrec` 4. `rodriguesNablaqrec2`
Differential- bzw. (q-) Differenzengleichung	`rodriguesdiffeq`	`rodriguesDeltadiffeq` (`rodriguesNabladiffeq`)	1. `rodriguesDqdiffeq` 2. `rodriguesDqinversdiffeq` 3. `rodriguesNablaqdiffeq` 4. `rodriguesNablaqdiffeq2`

Tabelle 4.1: Die Mapleprozeduren für alle Testfälle im Überblick

Die Rodriguesformel einer Funktionenfamilie $(f_n(x))_n$ ist gegeben als

$$f_n(x) = g_n(x)\left(\frac{d}{dx}\right)^n h_n(x). \tag{4.1}$$

Die Rodriguesformeln der klassischen orthogonalen Polynome lauten nach [KLS(2010)]:[1].

COPS	Notation	Rodriguesdarstellung
Hermitepolynome	$\mathrm{H}_n(x)$	$(-1)^n e^{x^2}(\frac{d}{dx})^n\left[e^{-x^2}\right]$
Laguerrepolynome	$\mathrm{L}_n^{(\alpha)}(x)$	$\frac{e^x}{x^\alpha n!}(\frac{d}{dx})^n\left[e^{-x}x^{n+\alpha}\right]$
Besselpolynome	$\mathrm{B}_n^{(a)}(x)$	$\frac{(n+a+1)_n}{2^n(n-1+a+2)_n x^a e^{-\frac{2}{x}}}(\frac{d}{dx})^n\left[x^a e^{-\frac{2}{x}}x^{2n}\right]$
Jacobipolynome	$\mathrm{P}_n^{(\alpha,\beta)}(x)$	$\frac{(-1)^n}{2^n n!(1-x)^\alpha(1+x)^\beta}(\frac{d}{dx})^n\left[(1-x)^{n+\alpha}(1+x)^{n+\beta}\right]$

Tabelle 4.2: Die Rodriguesformeln der klassischen orthogonalen Polynome

Eine komplexwertige Funktion $f(x)$, die auf einem einfach zusammenhängenden Gebiet $D \subset \mathbb{C}$ analytisch ist, lässt sich nach der Cauchyschen Integralformel unendlich oft differenzieren und die Ableitungen können wie folgt als Integrale dargestellt werden:

$$f^{(n)}(x) = \frac{n!}{2\pi i}\int_\gamma \frac{f(t)}{(t-x)^{n+1}}dt. \tag{4.2}$$

Dabei bezeichnet γ eine geschlossene Kurve in D, die x einmal im Gegenuhrzeigersinn umläuft. Ist für eine Funktionenfamilie $f_n(x)$ die Rodriguesformel (4.1) bekannt, so gilt mit (4.2) also

$$f_n(x) = g_n(x)\frac{n!}{2\pi i}\int_\gamma \frac{h_n(t)}{(t-x)^{n+1}}dt \tag{4.3}$$

Aus der Integraldarstellung kann nun über Anwendung des stetigen Zeilberger-Algorithmus (siehe Abschnitt 3.2.2), etwa die Mapleprozedur `intrecursion` aus [Koepf(2014)], eine holonome Rekursionsgleichung für die Funktionenfamilie $f_n(x)$ gefunden werden. Die Mapleprozedur `rodriguesrecursion` aus [Koepf(2014)] setzt genau dies um. Wir geben ihren Code an.

[1] Die Rodriguesdarstellung der Besselpolynome entstammt nicht [KLS(2010)], sondern sie wurde entsprechend der Bildungsvorschrift der Rodriguesformel berechnet, siehe [NU(1988), Seite 8]

```
rodriguesrecursion:=proc(g,h,x,sn)
  local S,n,t,result;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  result:=intrecursion(n!*g*subs(x=t,h)/(t-x)^(n+1),t,S(n));
end proc: #rodriguesrecursion
```

Da es sich bei (4.3) um ein Integral über eine geschlossene Kurve γ handelt, ist die ausgegebene Rekursionsgleichung automatisch homogen. Wir überprüfen die Prozedur `rodriguesrecursion` exemplarisch mit Hilfe der Hermitepolynome. Wir versetzen uns in die Rolle des Unwissenden und nehmen vorerst nur an, dass

$$\mathrm{H}_n(x) = (-1)^n e^{x^2} \left(\frac{d}{dx}\right)^n \left[e^{-x^2}\right] \tag{4.4}$$

die Rodriguesformel der Hermitepolynome ist. Wir wollen dies mit Hilfe der Prozedur `rodriguesrecursion` aus [Koepf(2014)] überprüfen.
Wir erhalten folgende Ausgabe in Maple:

```
> rodriguesrecursion((-1)^n*exp(x^2),exp(-x^2),x,H(n));
```

$$\mathrm{H}(n+2) - 2\,x\,\mathrm{H}(n+1) + 2\,(n+1)\,\mathrm{H}(n) = 0$$

Diese Ausgabe erhalten wir ebenso, wenn wir die Rekursionsgleichung unter Verwendung der Prozedur `sumrecursion` aus [Koepf(2014)] direkt aus der hypergeometrischen Summendarstellung

$$H_n(x) = (2x)^n \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-\frac{n}{2})_k (-\frac{(n-1)}{2})_k}{k!} \left(-\frac{1}{x^2}\right)^k. \tag{4.5}$$

generieren. Maple liefert uns dazu die folgende Rekursionsgleichung:

```
> sumrecursion((2*x)^n*hyperterm([-n/2,-(n-1)/2],[],-1/x^2,k),
> k,H(n));
```

$$\mathrm{H}(n+2) - 2\,x\,\mathrm{H}(n+1) + 2\,(n+1)\,\mathrm{H}(n) = 0$$

Die Rekursionsgleichungen stimmen also überein. Überprüfen wir noch, ob die beiden Anfangswerte $H_0(x)$ und $H_1(x)$ sowohl für die angenommene Rodriguesdarstellung (4.4) als auch die hypergeometrische Reihe (4.5) übereinstim-

men, haben wir gezeigt, dass (4.4) tatsächlich die Rodriguesformel der Hermitepolynome ist. Aus der Gestalt (4.4) erhalten wir sofort $H_0(x) = 1$ sowie $H_1(x) = 2x$. Wenn wir (4.5) für $n = 0$ und $n = 1$ betrachten, ergeben sich mit $(0)_0 = 1$ die Werte

$$H_0(x) = \sum_{k=0}^{0} \frac{(0)_k \left(\frac{1}{2}\right)_k}{k!} \left(\frac{-1}{x^2}\right)^k = 1 \tag{4.6}$$

sowie

$$H_1(x) = (2x) \cdot \sum_{k=0}^{0} \frac{\left(-\frac{1}{2}\right)_k (0)_k}{k!} \left(\frac{-1}{x^2}\right)^k = 2x \tag{4.7}$$

in Übereinstimmung mit den aus (4.4) erzeugten Anfangswerten. Damit ist (4.4) tatsächlich die Rodriguesdarstellung der Hermitepolynome.
Wir wollen nun eine Prozedur angeben, die die von `rodriguesrecursion` ausgegebene Rekursionsgleichung entsprechend der besonderen Form von Bemerkung 2.27 normiert. Wir stellen diese Normierungsprozedur hier vor.

```
rodriguesNorm:=proc(rec,sn)
  local S,n,result;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  result:=lhs(rec)-rhs(rec);
  result:=subs(n=n-1,result);
  result:=collect(-expand(result-coeff(result,x,0)),S,factor)=
  coeff(result,x,0);
end proc: #rodriguesNorm
```

Schließlich geben wir der Vollständigkeit halber in Tabelle 4.3 auf Seite 53 die durch Hintereinanderausführung von `rodriguesrecursion` und `rodriguesNorm` erzeugten Drei-Term-Rekursionen aller COPS an.

4.1.2 Holonome Differentialgleichungen aus Rodriguesformeln im stetigen Fall

Wir wollen nun auf ähnliche, doch andere Art und Weise stetige Funktionenfamilien $(f_n(x))_n$ identifizieren, die durch eine Rodriguesformel gegeben sind. Wir werden aus der Rodriguesformel eine holonome Differentialgleichung erzeugen und sie mit der vergleichen, die unter Verwendung der Prozedur `sumdiffeq` aus [Koepf(2014)] aus der hypergeometrischen Summendarstellung erzeugt werden kann. Nach Abgleich entsprechend vieler Anfangswerte müssen die Funktionenfamilien dann übereinstimmen. Das liegt daran, dass das Anfangswertproblem für Differentialgleichungen n-ter Ordnung unter sehr milden Bedingungen, die in unserem Fall erfüllt sind, genau eine Lösung besitzt, siehe dazu [Walter(2000), §11, II. und III.]. Ein solches Anfangswertproblem ist äquivalent zu einem System von n Differentialgleichungen erster Ordnung. Von Differentialgleichungen erster Ordnung wissen wir, dass sie nach dem Existenz- und Eindeutigkeitssatz [Walter(2000), §6, I.] bei Angabe eines Anfangswertes genau eine Lösung besitzen. Wir wenden uns also in diesem Abschnitt der Erzeugung von Differentialgleichungen zu. Dazu kann auf die Darstellung

$$f_n(x) = g_n(x)\frac{n!}{2\pi i}\int_\gamma \frac{h_n(t)}{(t-x)^{n+1}}dt \tag{4.8}$$

aus Abschnitt 4.1.1 die Variante des Zeilberger-Algorithmus zur Erzeugung von holonomen Differentialgleichungen (siehe Abschnitt 3.2.3) angewandt werden, die in [Koepf(2014)] als Mapleprozedur `intdiffeq` umgesetzt wurde. [Koepf(2014)] hat `intdiffeq` benutzt und damit die Mapleprozedur `rodriguesdiffeq` umgesetzt.

```
rodriguesdiffeq:=proc(g,h,n,sx)
  local S,x,t,result;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  result:=intdiffeq(g*subs(x=t,h)/(t-x)^(n+1),t,S(x));
end proc: #rodriguesdiffeq
```

Wir testen diese Prozedur wieder am Beispiel der Hermitepolynome:

```
> rodriguesdiffeq((-1)^n*exp(x^2),exp(-x^2),n,H(x));
```

$$2\,n\,\mathrm{H}(x) - 2\,x\,(\tfrac{d}{dx}\,\mathrm{H}(x)) + (\tfrac{d^2}{dx^2}\,\mathrm{H}(x)) = 0$$

Auch hier stellt sich heraus, dass dies diejenige Differentialgleichung ist, die mit der Prozedur `sumdiffeq` direkt aus der hypergeometrischen Reihe der Hermitepolynome erzeugt werden kann:

```
> sumdiffeq((2*x)^n*hyperterm([-n/2,-(n-1)/2],[],-1/x^2,k),
> k,H(x));
```

$$2\,n\,\mathrm{H}(x) - 2\,x\,(\tfrac{d}{dx}\,\mathrm{H}(x)) + (\tfrac{d^2}{dx^2}\,\mathrm{H}(x)) = 0$$

Überprüfen wir noch die zwei Anfangswerte $H_n(0)$ und $H_n'(0)$ für beide Darstellungen, so haben wir gezeigt, dass die Rodriguesformel (4.4) die Klasse der Hermitepolynome darstellt. Wir wollen nun $H_n(0)$ aus der Rodriguesformel (4.4) bestimmen. Wenn wir diese betrachten, bemerken wir, dass die n-te Ableitung der holomorphen Funktion e^{-x^2} nach der allgemeinen Cauchyschen Integralformel, [BS(1976), Seite 123], die Integraldarstellung

$$\left(\frac{d}{dx}\right)^n \left[e^{-x^2}\right] = \frac{n!}{2\pi i}\int_\gamma \frac{e^{-t^2}}{t^{n+1}}dt \tag{4.9}$$

besitzt. Diese Gleichung ist ein Spezialfall von Gleichung (4.3). Dabei ist γ eine geschlossene Kurve in einem Gebiet $D \subset \mathbb{C}$, die 0 einmal im Gegenuhrzeigersinn umläuft. Wir können $H_n(x)$ somit darstellen als

$$H_n(x) = (-1)^n e^{x^2}\,\frac{n!}{2\pi i}\int_\gamma \frac{e^{-t^2}}{t^{n+1}}dt \tag{4.10}$$

Wir sehen, dass

$$c_n := \frac{1}{2\pi i}\int_\gamma \frac{e^{-t^2}}{t^{n+1}}dt \tag{4.11}$$

der n-te Koeffizient der Taylorreihe der Funktion $f(x) = e^{-x^2}$ ist, vgl. den Satz zur Eindeutigkeit der Potenzreihenentwicklung in [BS(1976), Seite 133]. Um $H_n(0)$ berechnen zu können, entwickeln wir diese Funktion in eine Potenzreihe

$$f(t) = \sum_{n=0}^{\infty} c_n t^n = \sum_{m=0}^{\infty} \frac{(-1)^m t^{2m}}{m!}.$$

Eine algorithmische Methode hierfür findet sich in [Koepf(1992)], siehe auch [Koepf(2006)]. Damit erhalten wir

$$c_n = \begin{cases} \frac{(-1)^m}{m!} & \text{, falls } n = 2m \\ 0 & \text{, falls } n = 2m+1 \end{cases}, \tag{4.12}$$

wobei $m \in \mathbb{N}_0$. Zusammen mit den Gleichungen (4.10) und (4.11) ergibt sich für $x = 0$ dann

$$H_n(0) = \begin{cases} (-1)^m \frac{(2m)!}{m!} & , \text{falls } n = 2m \\ 0 & , \text{falls } n = 2m+1 \end{cases}, \tag{4.13}$$

wobei $m \in \mathbb{N}_0$. Dieses Ergebnis wird ebenfalls in [OLBC(2010), Seite 444] angegeben. Dort findet sich der Ausdruck

$$H_{2n}(0) = (-1)^n \left(n+1\right)_n .$$

Auf gleiche Weise werden wir nun den Wert $H_n'(0)$ aus der Rodriguesdarstellung (4.4) berechnen. Dafür leiten wir Gleichung (4.4) ab:

$$H_n'(x) = (-1)^n \left(2xe^{x^2} \left(\frac{d}{dx}\right)^n \left[e^{-x^2}\right] + e^{x^2} \left(\frac{d}{dx}\right)^{n+1} \left[e^{-x^2}\right] \right) \tag{4.14}$$

Der linke Summand von (4.14) verschwindet für $x = 0$. Es bleibt nur der rechte Summand von (4.14) stehen und $H_n'(0)$ kann mit Hilfe der Cauchyschen Integralformel dargestellt werden als

$$H_n'(0) = (-1)^n e^{x^2} \left(\frac{d}{dx}\right)^{n+1} \left[e^{-x^2}\right] \mid_{x=0} = (-1)^n \frac{(n+1)!}{2\pi i} \int_\gamma \frac{e^{-t^2}}{t^{n+2}} dt. \tag{4.15}$$

Also ist

$$H_n'(0) = (-1)^n (n+1)!\, c_{n+1},$$

und mit (4.12) ergibt sich explizit

$$H_n'(0) = \begin{cases} 0 & , \text{falls } n = 2m \\ \frac{(-1)^m (2m+2)!}{(m+1)!} & , \text{falls } n = 2m+1 \end{cases}, \tag{4.16}$$

wobei $m \in \mathbb{N}_0$. Dieses Ergebnis wird von [OLBC(2010)] auf Seite 444 bestätigt. Darin findet sich der Wert

$$H_{2n+1}'(0) = 2\,(-1)^n \left(n+1\right)_{n+1} .$$

Wir gleichen diese Ergebnisse für $H_n(0)$ und $H_n'(0)$ mit denen ab, die wir aus der hypergeometrischen Summendarstellung

$$H_n(x) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^k 2^n (-\frac{n}{2})_k (-\frac{(n-1)}{2})_k}{k!} x^{n-2k} \tag{4.17}$$

erhalten.

Wir beginnen mit $H_n(0)$ und unterscheiden, ob n gerade oder ungerade ist.

1. Sei n gerade. Wir betrachten

$$H_n(x) = \sum_{k=0}^{\frac{n}{2}} \frac{(-1)^k 2^n (-\frac{n}{2})_k (-\frac{(n-1)}{2})_k}{k!} x^{n-2k}. \tag{4.18}$$

   Setzen wir $x = 0$ in Gleichung (4.18) ein, bleibt nur noch der Summand für $k = \frac{n}{2}$ übrig:

$$H_n(0) = \frac{(-1)^{\frac{n}{2}} 2^n (-\frac{n}{2})_{\frac{n}{2}} (-\frac{(n-1)}{2})_{\frac{n}{2}}}{(\frac{n}{2})!}$$

   Da

$$\left(-\frac{n}{2}\right)_{\frac{n}{2}} \left(-\frac{(n-1)}{2}\right)_{\frac{n}{2}} = \frac{(-1)^n n!}{2^n},$$

   erhalten wir für $n = 2m$ $(m \in \mathbb{N}_0)$

$$H_n(0) = (-1)^m \frac{(2m)!}{m!}. \tag{4.19}$$

2. Sei n ungerade. Wir betrachten

$$H_n(x) = \sum_{k=0}^{\frac{n-1}{2}} \frac{(-1)^k 2^n (-\frac{n}{2})_k (-\frac{(n-1)}{2})_k}{k!} x^{n-2k}. \tag{4.20}$$

   Hier sieht man nun, dass für ungerades n die Gleichung $H_n(0) = 0$ gilt, da für ungerade n der Summationsindex $k = \frac{n}{2}$ nicht durchlaufen wird, bei dem der Ausdruck x^{n-2k} für $x = 0$ den Wert 1 ergeben würde.

Fassen wir beide Fälle zusammen, erhalten wir aus der hypergeometrischen Summendarstellung ebenfalls Gleichung (4.13).
Schauen wir uns nun noch auf gleiche Weise den Wert für $H_n'(0)$ aus der hypergeometrischen Summendarstellung an. Zunächst leiten wir dafür Gleichung (4.17) ab und erhalten

$$H_n'(x) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^k 2^n (-\frac{n}{2})_k (-\frac{(n-1)}{2})_k (n-2k)}{k!} x^{n-2k-1}, \tag{4.21}$$

Wir unterscheiden wieder, ob n gerade oder ungerade ist.

1. Sei n gerade. Zunächst erhalten wir im Hinblick auf $\left(-\frac{n}{2}\right)_k$ die obere Summationsgrenze $\frac{n}{2}$:

$$H_n'(x) = \sum_{k=0}^{\frac{n}{2}} \frac{(-1)^k 2^n (-\frac{n}{2})_k (-\frac{(n-1)}{2})_k (n-2k)}{k!} x^{n-2k-1} \tag{4.22}$$

   Da für gerades n der Summationsindex $k = \frac{n-1}{2}$ nicht durchlaufen wird, bei dem der Ausdruck x^{n-2k-1} für $x = 0$ den Wert 1 ergeben würde, notieren wir $H_n'(0) = 0$.

2. Sei n ungerade. Dann gilt:

$$H_n'(x) = \sum_{k=0}^{\frac{n-1}{2}} \frac{(-1)^k 2^n (-\frac{n}{2})_k (-\frac{(n-1)}{2})_k}{k!} x^{n-2k-1}. \tag{4.23}$$

   Setzen wir hierin $x = 0$ ein, bleibt nur der Summand für $k = \frac{n-1}{2}$ stehen:

$$H_n'(0) = \frac{(-1)^{\frac{n-1}{2}} 2^n (-\frac{n}{2})_{\frac{n-1}{2}} (-\frac{(n-1)}{2})_{\frac{n-1}{2}}}{(\frac{n-1}{2})!}. \tag{4.24}$$

   Da

$$\left(-\frac{n}{2}\right)_{\frac{n-1}{2}} \left(-\frac{(n-1)}{2}\right)_{\frac{n-1}{2}} = \frac{(-1)^{n-1} n!}{2^{n-1}},$$

   wird (4.24) zu

$$H_n'(0) = \frac{(-1)^{\frac{n-1}{2}} \cdot 2 \cdot n!}{(\frac{n-1}{2})!}.$$

   Diese Gleichung lässt sich einfach erweitern zu

$$H_n'(0) = \frac{(-1)^{\frac{n-1}{2}} (n+1)!}{(\frac{n+1}{2})!}.$$

   Dies entspricht für $n = 2m+1$ gerade

$$H_n'(0) = (-1)^{m-1} \frac{(2m)!}{m!}.$$

Fassen wir beide Fälle zusammen, erhalten wir aus der hypergeometrischen Summendarstellung ebenfalls Gleichung (4.16).

Insgesamt stimmen die beiden Anfangswerte $H_n(0)$ und $H_n'(0)$ der Hermitepolynome für die Rodriguesdarstellung und die hypergeometrische Summendarstellung überein und somit ist die Rodriguesdarstellung (4.4) der Hermitepolynome auch auf diesem Wege verifiziert.
Der Vollständigkeit halber geben wir in Tabelle 4.4 auf Seite 53 die mit `rodriguesdiffeq` erzeugten Differentialgleichungen der COPS an.

4.1.3 Eine Testfamilie, die kein OPS ist

In diesem Abschnitt wollen wir einmal für den Fall, dass eine Rodriguesdarstellung einer nicht-orthogonalen Funktionenfamilie vorliegt, testen, ob die Algorithmen zur Erzeugung von Rekursions- und Differentialgleichungen funktionieren. Inspiriert von dem Beispiel der Bateman-Funktionen aus [Koepf(2014), Seite 264]

$$F_n(x) = \frac{e^{-x}}{n} \sum_{k=1}^{n} \frac{(-1)^k}{(k-1)!} \binom{n}{k} (2x)^k,$$

das in [Bateman(1931)] und [Koepf(2014)] eingeführt und untersucht wurde, betrachten wir das Beispiel der Funktionenfamilie

$$G_n(x) = e^{-x^2} H_n(x) = e^{-x^2} (2x)^n \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-\frac{n}{2})_k (-\frac{(n-1)}{2})_k}{k!} \left(-\frac{1}{x^2} \right)^k \tag{4.25}$$

Die Rodriguesdarstellung der Hermitepolynome liefert die Rodriguesdarstellung der Funktionenfamilie $G_n(x)$:

$$G_n(x) = (-1)^n \left(\frac{d}{dx} \right)^n \left[e^{-x^2} \right]. \tag{4.26}$$

Wir wollen die Funktionenfamilie $G_n(x)$ nun an Hand ihrer Rodriguesformel identifizieren. Da der Vorfaktor e^{-x^2}, durch den sich $G_n(x)$ von $H_n(x)$ in (4.25) unterscheidet, nicht von n abhängt, erhalten wir dieselbe Rekursionsgleichung wie für die Hermitepolynome $H_n(x)$ in Abschnitt 4.1.1. Dieser Fall ist also für unsere Untersuchung weniger interessant. Wir werden die Funktionenfamilie $G_n(x)$ daher identifizieren, indem wir Differentialgleichungen aus den Darstellungen (4.26) sowie (4.25) erzeugen.

COPS	Drei-Term-Rekursion[a]
Hermite	$2\,xH\left(n\right)=H\left(n+1\right)+2\,n\,H\left(n-1\right)$
Laguerre	$-x\,\mathrm{L}(n)=\left(n+1\right)\mathrm{L}(n+1)-\left(2\,n+\alpha+1\right)\mathrm{L}(n)+\left(n+\alpha\right)\mathrm{L}(n-1)$
Bessel	$-\left(2\,n+a+2\right)\left(2\,n+a+1\right)\left(2\,n+a\right)x\,\mathrm{B}(n)=$ $-2\left(2\,n+a\right)\left(n+a+1\right)\mathrm{B}(n+1)+2\,a(2\,n+a+1)\,\mathrm{B}(n)+2\,n\left(2\,n+a+2\right)\mathrm{B}(n-1)$
Jacobi	$\left(2\,n+\alpha+\beta+2\right)\left(2\,n+\alpha+\beta+1\right)\left(2\,n+\alpha+\beta\right)\,x\,P\left(n\right)=$ $2\left(n+1\right)\left(n+\alpha+\beta+1\right)\left(2\,n+\alpha+\beta\right)P\left(n+1\right)-\left(2\,n+\alpha+\beta+1\right)\left(\alpha^{2}-\beta^{2}\right)P\left(n\right)$ $+2\left(n+\alpha\right)\left(n+\beta\right)\left(2\,n+\alpha+\beta+2\right)P\left(n-1\right)$

Tabelle 4.3: Die Drei-Term-Rekursionen der klassischen orthogonalen Polynome

COPS	Differentialgleichung
Hermite	$\frac{\mathrm{d}^{2}}{\mathrm{d}x^{2}}H\left(x\right)-2\,x\frac{\mathrm{d}}{\mathrm{d}x}H\left(x\right)+2\,nH\left(x\right)=0$
Laguerre	$x\left(\frac{\mathrm{d}^{2}}{\mathrm{d}x^{2}}L\left(x\right)\right)+\left(\alpha+1-x\right)\frac{\mathrm{d}}{\mathrm{d}x}L\left(x\right)+nL\left(x\right)=0$
Bessel	$x^{2}\left(\frac{\mathrm{d}^{2}}{\mathrm{d}x^{2}}B\left(x\right)\right)+\left(a\,x+2\,x+2\right)\frac{\mathrm{d}}{\mathrm{d}x}B\left(x\right)-n\left(n+a+1\right)B\left(x\right)=0$
Jacobi	$\left(x-1\right)\left(x+1\right)\left(\frac{\mathrm{d}^{2}}{\mathrm{d}x^{2}}P\left(x\right)\right)+\left(\left(\alpha+\beta+2\right)\,x+\alpha-\beta\right)\frac{\mathrm{d}}{\mathrm{d}x}P\left(x\right)-n\left(n+\alpha+\beta+1\right)P\left(x\right)=0$

Tabelle 4.4: Die Differentialgleichungen der klassischen orthogonalen Polynome

[a]Die Mapleausgaben wurden nachträglich entsprechend den Ausgaben in [KLS(2010)] sortiert.

Zunächst liefert uns `rodriguesdiffeq` eine Differentialgleichung aus der Rodriguesdarstellung (4.26):

```
> rodriguesdiffeq((-1)^n,exp(-x^2),n,G(x));
```

$$2\,(n+1)\,\mathrm{G}(x) + 2\,x\,(\tfrac{d}{dx}\,\mathrm{G}(x)) + (\tfrac{d^2}{dx^2}\,\mathrm{G}(x)) = 0$$

Andererseits stellen wir fest, dass uns die Prozedur `sumdiffeq` diese Differentialgleichung aus der hypergeometrischen Reihe (4.25) generiert:

```
> sumdiffeq(exp(-x^2)*(2*x)^n*
> hyperterm([-n/2,-(n-1)/2],[],-1/(x^2),k),k,G(x));
```

$$2\,(n+1)\,\mathrm{G}(x) + 2\,x\,(\tfrac{d}{dx}\,\mathrm{G}(x)) + (\tfrac{d^2}{dx^2}\,\mathrm{G}(x)) = 0$$

Die Differentialgleichungen stimmen also überein. Um die Rodriguesdarstellung (4.26) zu verifizieren, berechnen wir noch $G_n(0)$ und $G_n'(0)$ sowohl aus der Rodriguesdarstellung als auch aus der hypergeometrischen Summendarstellung und vergleichen sie miteinander.
Analog zu den Betrachtungen in Abschnitt 4.1.2, als $H_n(0)$ über funktionentheoretische Argumente aus der Rodriguesdarstellung berechnet wurde, erhalten wir

$$G_n(0) = \begin{cases} (-1)^m \frac{(2m)!}{m!} & \text{, falls } n = 2m \\ 0 & \text{, falls } n = 2m+1 \end{cases}, \tag{4.27}$$

wobei $m \in \mathbb{N}_0$.
Analog zu den Betrachtungen in Abschnitt 4.1.2 zur Berechnung von $H_n'(0)$ ergibt sich

$$G_n'(0) = \begin{cases} 0 & \text{, falls } n = 2m \\ \frac{(-1)^{m-1}(2m)!}{m!} & \text{, falls } n = 2m-1 \end{cases}, \tag{4.28}$$

wobei $m \in \mathbb{N}_0$ bzw. $\mathbb{N}$.
Da $e^{-x^2}\,|_{x=0} = 1$, ergeben sich für $G_n(0)$ und $G_n'(0)$ aus der hypergeometrischen Reihe (4.25) dieselben Werte wie für $H_n(0)$ und $H_n'(0)$, die in Abschnitt 4.1.2 aus der hypergeometrischen Summendarstellung berechnet wurden. Also stimmen diese Werte für $G_n(0)$ und $G_n'(0)$ mit (4.27) und (4.28) überein.
Insgesamt haben wir damit die Funktionenfamilie $G_n(x)$ gegeben durch die Rodriguesformel (4.26) verifiziert.

4.2 Holonome Rekursions- und Differenzengleichungen aus Rodriguesformeln im diskreten Fall

In Analogie zu den Ausführungen aus Kapitel 4.1 möchten wir nun diskrete Funktionenfamilien identifizieren, indem wir holonome Rekursions- als auch Differenzengleichungen aus diskreten Rodriguesformeln erzeugen. Dafür benötigen wir eine Summendarstellung der Rodriguesformel, auf die dann der diskrete Zeilberger-Algorithmus angewandt werden kann.

4.2.1 Holonome Rekursionsgleichungen aus Rodriguesformeln im diskreten Fall

In Analogie zur Darstellung (4.1) im vorhergehenden Kapitel hat die diskrete Rodriguesformel einer Funktionenfamilie $(f_n(x))_n$ folgende Gestalt, wobei der Delta-Operator den Differentialoperator $\frac{d}{dx}$ ersetzt:

$$f_n(x) = g_n(x)\Delta^n h_n(x). \tag{4.29}$$

Die Rodriguesformeln der klassischen diskreten orthogonalen Polynome lauten:

CDOPS	Notation	Rodriguesdarstellung
Charlier	$\mathrm{C}_n(x;a)$	$\frac{x!}{a^x}\Delta^n\left[\frac{a^{x-n}}{(x-n)!}\right]$
Meixner	$\mathrm{M}_n(x;\beta,c)$	$\frac{x!}{(\beta)_x\cdot c^x}\Delta^n\left[\frac{(\beta+n)_{x-n}\cdot c^{x-n}}{(x-n)!}\right]$
Krawtchouk	$\mathrm{K}_n(x;p,N)$	$\frac{1}{\binom{N}{x}\cdot(\frac{p}{1-p})^x}\Delta^n\left[\binom{N-n}{x-n}\cdot(\frac{p}{1-p})^{x-n}\right]$
Hahn	$\mathrm{Q}_n(x;\alpha,\beta,N)$	$\frac{(-1)^n\cdot(\beta+1)_n}{w(x;\alpha,\beta,N)\cdot(-N)_n}\Delta^n[w(x-n;\alpha+n,\beta+n,N-n)]$ mit $w(x;\alpha,\beta,N) := \binom{\alpha+x}{x}\binom{\beta+N-x}{N-x}$

Tabelle 4.5: Die Rodriguesformeln der klassischen diskreten orthogonalen Polynome in Δ-Schreibweise

Die Rodriguesformeln in Δ-Schreibweise wurden aus der in [KLS(2010)] verwandten ∇-Schreibweise durch n-fachen Shift der Variablen x gewonnen, vergleiche Satz 2.13. Außerdem ist die w-Funktion der Hahn-Polynome wie folgt

definiert:

$$w(x;\alpha,\beta,N) := \binom{\alpha+x}{x}\binom{\beta+N-x}{N-x}.$$

Mit den Notationen aus [Aigner(2006)] können wir Δ darstellen als

$$\Delta = E - I \tag{4.30}$$

mit E als dem Translationsoperator mit Schrittweite 1 und der Identität $I = E^0$. E nennen wir auch Shift-Operator.
Da diese Operatoren linear sind und kommutativ bezüglich der Hintereinanderausführung, vergleiche [Aigner(2006)], ergibt sich nach dem Binomialsatz

$$(E-I)^n = \sum_{k=0}^{n} \binom{n}{k} E^k (-I)^{n-k} \tag{4.31}$$

und damit

$$\Delta^n h_n(x) = \sum_{k=0}^{n} (-1)^{n-k} \binom{n}{k} h_n(x+k). \tag{4.32}$$

Also lässt sich die Rodriguesformel (4.29) als Summe darstellen:

$$f_n(x) = g_n(x)\Delta^n h_n(x) = \sum_{k=0}^{n} g_n(x)(-1)^{n-k} \binom{n}{k} h_n(x+k). \tag{4.33}$$

Da die Theorie des Vorwärtsdifferenzenoperators Δ und des Rückwärtsdifferenzenoperators ∇ gleichwertig sind, kann man die diskrete Rodriguesformel auch schreiben als

$$f_n(x) = g_n(x)\nabla^n \tilde{h}_n(x). \tag{4.34}$$

Auf Grund des Binomialsatzes für Operatoren angewandt auf $\nabla = (I - E^{-1})$ gilt

$$\nabla^n \tilde{h}_n(x) = \sum_{k=0}^{n} (-1)^k \binom{n}{k} \tilde{h}_n(x-k) \tag{4.35}$$

und (4.34) lässt sich wie folgt in eine Summe umwandeln:

$$f_n(x) = \sum_{k=0}^{n} g_n(x)(-1)^k \binom{n}{k} \tilde{h}_n(x-k). \tag{4.36}$$

Auf die Summendarstellungen (4.33) und (4.36) sowohl bzgl. Δ als auch bzgl. ∇ kann nun die Mapleprozedur `sumrecursion` aus [Koepf(2014)] angewandt werden.
Wir wollen an dieser Stelle die Mapleprozedur `rodriguesDeltarec` angeben, die diese Vorgehensweise für den Δ-Operator implementiert und Drei-Term-Rekursionen für diskrete Funktionenfamilien erzeugt, deren Rodriguesformel bekannt ist.

```
rodriguesDeltarec:=proc(g,h,x,sn)
  local S,n,k,rec;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  rec:=sumrecursion(g*binomial(n,k)*(-1)^(n-k)*subs(x=x+k,h),k,
  S(n));
end: #rodriguesDeltarec
```

Als Testbeispielklasse für diese Prozedur wählen wir die Meixnerpolynome. Sie besitzen die hypergeometrische Summendarstellung

$$M_n(x) = \sum_{k=0}^{n} \frac{(-n)_k\,(-x)_k}{(\beta)_k\,k!}\left(1-\frac{1}{c}\right)^k \tag{4.37}$$

und ihre Rodriguesformel lautet in Δ-Schreibweise:

$$M_n(x) = \frac{x!}{(\beta)_x \cdot c^x}\Delta^n\left[\frac{(\beta+n)_{x-n}\cdot c^{x-n}}{(x-n)!}\right]. \tag{4.38}$$

Daraus berechnen wir nun folgende Drei-Term-Rekursion:

```
> rodriguesDeltarec(x!/pochhammer(beta,x)/c^x,
> pochhammer(beta+n,x-n)*c^(x-n)/(x-n)!,x,M(n));
```

$$c\,(\beta+n+1)\,\mathrm{M}(n+2) - (c\,\beta + c\,n + x\,c + c + n - x + 1)\,\mathrm{M}(n+1) + (n+1)\,\mathrm{M}(n) = 0$$

Wir überprüfen Formel (4.38) mit Hilfe der Mapleprozedur `sumrecursion` aus [Koepf(2014)] und wenden diese Prozedur auf die hypergeometrische Summendarstellung (4.37) an. Wir wollen schauen, ob wir damit dieselbe Rekursionsgleichung erzeugen können.

```
> sumrecursion(x!/(pochhammer(beta,x)*c^x)*
> hyperterm([-n,-x],[beta],1-1/c,k),k,M(n,x));
```

$$c\,(\beta+n+1)\,\mathrm{M}(n+2)-(c\,\beta+c\,n+x\,c+c+n-x+1)\,\mathrm{M}(n+1)\\+(n+1)\,\mathrm{M}(n)=0$$

Nachdem wir zwei Anfangswerte sowohl aus der Rodriguesdarstellung (4.38) als auch aus der hypergeometrischen Summendarstellung (4.37) berechnet haben, stellen wir fest, dass diese gleich sind und für beide Vorgehensweisen $M_0(x) = 1$ sowie $M_1(x) = \frac{\beta c+xc-x}{\beta c}$ ergeben. Damit ist die Rodriguesformel (4.38) der Meixner-Polynome verifiziert.
Analog arbeitet die Prozedur `rodriguesNablarec`, die Drei-Term-Rekursionen aus der Rodriguesformel in ∇-Schreibweise generiert. Zusätzlich haben wir noch die Prozedur `DeltaNorm` geschrieben, die zur Normierung der von den Prozeduren `rodriguesDeltarec` und `rodriguesNablarec` ausgegebenen Rekursionen dient und die Rekursionsgleichungen in der besonderen Form von Bemerkung 2.27 ausgibt, wie sie auch in [KLS(2010)] dargestellt werden. Wir geben den Code der Mapleprozedur `DeltaNorm` an.

```
DeltaNorm:=proc(rec,x,sn)
  local S,n,k,result;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  result:=lhs(rec)-rhs(rec);
  result:=subs(n=n-1,result);
  result:=collect(-expand(result-coeff(result,x,0)),S,factor)=
  coeff(result,x,0);
end: #DeltaNorm
```

Der Code der hier nicht explizit angegebenen Mapleprozeduren ist auf dem der Arbeit beigefügten Datenträger dokumentiert.
Der Vollständigkeit halber geben wir in Tabelle 4.6 auf Seite 63 die durch Hintereinanderausführung von `rodriguesDeltarec` und `DeltaNorm` erzeugten Drei-Term-Rekursionen der CDOPS an.

4.2.2 Holonome Differenzengleichungen aus Rodriguesformeln im diskreten Fall

Wir wollen nun diskrete Funktionenfamilien $(f_n(x))_n$, die durch eine Rodriguesformel gegeben sind, auf eine andere Art und Weise identifizieren. Dafür werden wir im Folgenden holonome Differenzengleichungen aus Rodriguesformeln erzeugen und mit den aus der hypergeometrischen Summendarstellung erzeugten abgleichen. Schließlich werden wir noch entsprechend viele Anfangswerte vergleichen.
Wir werden uns in der Hauptsache mit denjenigen Rodriguesformeln beschäftigen, die den Vorwärtsdifferenzenoperator Δ verwenden. Ausführungen mit Hilfe des Rückwärtsdifferenzenoperators ∇ sind ebenfalls möglich und wurden ebenfalls von uns implementiert und getestet.
Um holonome Differenzengleichungen aus der Rodriguesformel

$$f_n(x) = g_n(x)\Delta^n h_n(x)$$

zu generieren, wandeln wir diese Gleichung zunächst wieder wie in Abschnitt 4.2.1 in eine Summendarstellung um:

$$f_n(x) = \sum_{k=0}^{n} g_n(x)(-1)^{n-k}\binom{n}{k}h_n(x+k). \tag{4.39}$$

Hierauf können wir genauso wie im vorhergehenden Kapitel den diskreten Zeilberger-Algorithmus bzgl. der Variablen x anwenden, vergleiche dazu Kapitel 3.1.2 und die Mapleprozedur `sumrecursion` aus [Koepf(2014)].
Der Unterschied besteht darin, dass nun die Rollen von n und x vertauscht sind und wir eine Rekursionsgleichung in der Variablen x erhalten. Diese nennen wir auch Differenzengleichung.
Diese Überlegungen haben wir in der Mapleprozedur `rodriguesDeltadiffeq` umgesetzt. Man beachte, dass im Vergleich zur Prozedur `rodriguesDeltarec` die Rollen von n und x getauscht wurden. Wir geben den Code von `rodriguesDeltadiffeq` an.

```
rodriguesDeltadiffeq:=proc(g,h,n,sx)
  local S,x,k,diffeq;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
```

```
  else
    x:=op(1,sx);
  end if;
  diffeq:=sumrecursion(g*(-1)^(n-k)*binomial(n,k)*subs(x=x+k,h),
  k,S(x));
end: #rodriguesDeltadiffeq
```

Wir testen die Prozedur wieder am Beispiel der Meixnerpolynome. Aus der Rodriguesdarstellung (4.38) erhalten wir folgende Drei-Term-Differenzengleichung:

```
> rodriguesDeltadiffeq(x!/pochhammer(beta,x)/c^x,
> pochhammer(beta+n,x-n)*c^(x-n)/(x-n)!,n,M(x));
```

$$(\beta + x + 1)\, c\, \mathrm{M}(x+2) - (c\,\beta + c\,n + c\,x + c - n + x + 1)\, \mathrm{M}(x+1) + (x+1)\, \mathrm{M}(x) = 0$$

Um Darstellung (4.38) zu verifizieren, hoffen wir mit der Mapleprozedur `sumrecursion` aus [Koepf(2014)] angewandt auf die hypergeometrische Reihe (4.37) dieselbe Differenzengleichung erzeugen zu können. Maple bestätigt dies:

```
> sumrecursion(hyperterm([-n,-x],[beta],1-1/c,k),k,M(x));
```

$$(\beta + x + 1)\, c\, \mathrm{M}(x+2) - (c\,\beta + c\,n + c\,x + c - n + x + 1)\, \mathrm{M}(x+1) + (x+1)\, \mathrm{M}(x) = 0$$

Die erzeugten Differenzengleichungen stimmen also überein. Um Darstellung (4.38) vollends zu überprüfen, schauen wir, ob die Anfangswerte $M_n(0)$ sowie $M_n(1)$ jeweils aus der hypergeometrischen Reihe (4.37) und der Rodriguesdarstellung (4.38) berechnet übereinstimmen. Zunächst wandeln wir die Rodriguesformel (4.38) mit Hilfe von (4.39) in eine Summenschreibweise um und erhalten

$$M_n(x) = \frac{x!}{(\beta)_x \cdot c^x} \sum_{k=0}^{n} (-1)^{n-k} \binom{n}{k} \frac{(\beta+n)_{x+k-n} \cdot c^{x+k-n}}{(x+k-n)!}. \tag{4.40}$$

1. Für den Abgleich von $M_n(0)$ berechnen wir diesen Wert aus der hypergeometrischen Reihe (4.37) und sehen, dass sich auf diese Weise $M_n(0) = 1$ ergibt. Für die Berechnung von $M_n(0)$ aus der Rodriguesdarstellung müssen wir uns die Frage stellen, wie eine hypergeometrische Lösung einer Rekursionsgleichung von mindestens zweiter Ordnung gefunden werden

kann. Mit dieser Problemstellung befasst sich der Petkovšek-van-Hoeij-Algorithmus ([Petkovšek(1992), vanHoeij(1998)]), der in [Koepf(2014), Kapitel 9] dokumentiert wurde. Er kann in Maple unter `LREtools[hypergeomsols]` abgerufen werden. Wir werden ihn gleich verwenden.

Wir setzen nun $x = 0$ in (4.40) ein und wenden auf diese Summe den diskreten Zeilberger-Algorithmus an, der in Abschnitt 3.1.2 dokumentiert wurde. Wir verwenden dabei wieder die Prozedur `sumrecursion` aus [Koepf(2014)]. Maple gibt uns folgende Rekursionsgleichung zweiter Ordnung zurück:

```
> RE1:=sumrecursion((-1)^(n-k)*binomial(n,k)
> *pochhammer(beta+n,k-n)*c^(k-n)/(k-n)!,k,M(n));
```

$$c\,(\beta+n+1)\,\mathrm{M}(n+2)-(\beta\,c+c\,n+c+n+1)\,\mathrm{M}(n+1)+(n+1)\,\mathrm{M}(n)=0$$

Diese Rekursionsgleichung können wir wie gesagt mit dem Petkovšek-van-Hoeij-Algorithmus unter Verwendung von `LREtools[hypergeomsols]` lösen:

```
> LREtools[hypergeomsols](RE1,M(n),{},output=basis);
```

$$[1]$$

Wir erhalten als Lösung dieser Rekursionsgleichung 1 oder Vielfache davon. Um sicherzustellen, dass $M_n(0)$ tatsächlich den Wert 1 annimmt, berechnen wir, da es sich um eine Rekursionsgleichung zweiter Ordnung handelt, noch die beiden Werte $M_0(0)$ und $M_1(0)$. Wir setzen $x = 0$ in (4.40) ein und lassen Maple für uns rechnen:

```
> M[0](0):=sum((-1)^(-k)*binomial(0,k)*
> pochhammer(beta,k)*c^k/k!,k=0..0);
```

$$1$$

```
> M[1](0):=sum((-1)^(1-k)*binomial(1,k)*
> pochhammer(beta+1,k-1)*c^(k-1)/(k-1)!,k=0..1);
```

$$1$$

Da die beiden Anfangswerte ebenfalls konstant 1 sind, gilt auch für die Berechnung aus der Rodriguesdarstellung $M_n(0) = 1$. Der erste Anfangswert $M_n(0)$ für die eigentliche Identifizierung der Rodriguesformel (4.38) ist also für beide Darstellungen gezeigt.

2. Widmen wir uns nun dem zweiten Anfangswert $M_n(1)$ für die Identifizierung der Rodriguesformel der Meixner-Polynome. Berechnen wir diesen Wert aus der hypergeometrischen Reihe (4.37), erhalten wir unter Verwendung von Maple

```
> M[n](1):=normal(1+(-n)*(-1)/beta*(1-1/c));
```

$$M_n(1) := \frac{\beta\, c + c\, n - n}{\beta\, c}$$

Für die Berechnung von $M_n(1)$ aus der Rodriguesdarstellung setzen wir $x = 1$ in (4.38) ein und wenden auf die so entstandene Summe den diskreten Zeilberger-Algorithmus an. Wir benutzen wieder `sumrecursion` aus [Koepf(2014)]. Die Berechnung in Maple liefert uns folgende Rekursionsgleichung zweiter Ordnung:

```
> RE2:=sumrecursion(1/beta/c*(-1)^(n-k)*binomial(n,k)
> *pochhammer(beta+n,1+k-n)*c^(1+k-n)/(1+k-n)!,k,M(n));
```

$$c\,(\beta+n+1)\,\mathrm{M}(n+2)-(\beta\, c+c\, n+2\, c+n)\,\mathrm{M}(n+1)+(n+1)\,\mathrm{M}(n)=0$$

Wir verwenden erneut den Petkovšek-van-Hoeij-Algorithmus und lassen uns eine hypergeometrische Lösung dieser Rekursionsgleichung ausgeben:

```
> LREtools[hypergeomsols](RE2,M(n),{},output=basis);
```

$$[\beta\, c + c\, n - n]$$

Wir erhalten als Lösung dieser Rekursionsgleichung $\beta\, c + c\, n - n$ oder Vielfache dieses Ausdrucks. Insbesondere löst $\frac{\beta\, c+c\, n-n}{\beta\, c}$ die Rekursionsgleichung. Um $M_n(1) = \frac{\beta\, c+c\, n-n}{\beta\, c}$ zu verifizieren, überprüfen wir die beiden Anfangswerte $M_0(1)$ sowie $M_1(1)$ in Maple:

```
> M[0](1):=1/beta/c*sum((-1)^(-k)*binomial(0,k)*
> pochhammer(beta,1+k)*c^(1+k)/(1+k)!,k=0..0);
```

$$1$$

```
> M[1](1):=1/beta/c*sum((-1)^(1-k)*binomial(1,k)*
> pochhammer(beta+1,k)*c^k/k!,k=0..1);
```

$$\frac{-1 + (\beta + 1)\, c}{\beta\, c}$$

Damit haben wir in der Tat $M_n(1) = \frac{\beta\, c+c\, n-n}{\beta\, c}$ aus der Rodriguesformel berechnet. Dieses Ergebnis hatten wir auch aus der hypergeometrischen Reihe erhalten. Als Zwischenresultat halten wir fest, dass der Wert $M_n(1)$ für beide Berechnungsweisen übereinstimmt. Wir haben also auch den zweiten Anfangswert zur Differenzengleichung verifiziert.

CDOPS	Drei-Term-Rekursion
Charlier	$-x\,C\,(n) = a\,C\,(n+1) - (n+a)\,C\,(n) + n\,C\,(n-1)$
Meixner	$(c-1)\,x\,M\,(n) = c\,(n+\beta)\,M\,(n+1) - (\beta c + (n-1)\,c + c + n)\,M\,(n) + n\,M\,(n-1)$
Krawtchouk	$x\,K\,(n) = -p\,(N-n)\,K\,(n+1) + (pN - 2\,p\,(n-1) + n - 2\,p)\,K\,(n) + n\,(p-1)\,K\,(n-1)$
Hahn	$-(2\,n+\alpha+\beta+2)\,(2\,n+\alpha+\beta+1)\,(2\,n+\alpha+\beta)\,x\,Q\,(n) = (n+\alpha+1)\,(n+\alpha+\beta+1)\,(2\,n+\alpha+\beta)\,(N-n)\,\mathrm{Q}(n+1)$ $-(2n+\alpha+\beta+1)(N\alpha^2+N\alpha\beta+2N\alpha n+2N\beta n+2Nn^2-\alpha^2 n-\alpha n^2+\beta^2 n+\beta n^2+N\alpha+N\beta+2Nn-\alpha n+\beta n)Q\,(n)$ $+n\,(n+\beta)\,(2\,n+\alpha+\beta+2)\,(N+n+\alpha+\beta+1)\,Q\,(n-1)$

Tabelle 4.6: Die Drei-Term-Rekursionen der klassischen diskreten orthogonalen Polynome

CDOPS	Drei-Term-Differenzengleichung
Charlier	$-n\,C\,(x) = a\,C\,(x+1) - (x+a)\,C\,(x) + x\,C\,(x-1)$
Meixner	$n\,(c-1)\,M\,(x) = c\,(x+\beta)\,M\,(x+1) - (\beta c + (x-1)\,c + c + x)\,M\,(x) + x\,M\,(x-1)$
Krawtchouk	$n\,K\,(x) = -p\,(N-x)\,K\,(x+1) + (pN - 2\,p\,(x-1) - 2\,p + x)\,K\,(x) + (p-1)\,x\,K\,(x-1)$
Hahn	$-n\,(n+\alpha+\beta+1)\,Q\,(x) = (x+\alpha+1)\,(N-x)\,Q\,(x+1)$ $-\left(N\alpha + 2\,N(x-1) - \alpha\,(x-1) + \beta\,(x-1) - 2\,(x-1)^2 + 3\,N - \alpha + \beta - 4\,x + 2\right) Q\,(x)$ $+(N-x+\beta+1)\,x\,Q\,(x-1)$

Tabelle 4.7: Die Drei-Term-Differenzengleichungen der klassischen diskreten orthogonalen Polynome

Insgesamt haben wir, da die aus der hypergeometrischen Reihe (4.37) und aus der Rodriguesdarstellung (4.38) erzeugten Drei-Term-Differenzengleichungen und die beiden Anfangswerte $M_n(0)$ sowie $M_n(1)$ übereinstimmen, gezeigt, dass Darstellung (4.38) tatsächlich die Rodriguesdarstellung der Meixnerpolynome ist (Zeilbergers Paradigma).
Analog zur Mapleprozedur `rodriguesDeltadiffeq` arbeitet die Prozedur `rodriguesNabladiffeq`, die Differenzengleichungen aus der Rodriguesformel in ∇-Schreibweise generiert, vergleiche (4.34) in Abschnitt 4.2.1. Der Code der hier nicht explizit angegebenen Mapleprozedur ist auf dem der Arbeit beiliegenden Datenträger erfasst. In Tabelle 4.7 auf Seite 63 haben wir die durch Hintereinanderausführung von `rodriguesDeltadiffeq` und `DeltaNorm` erzeugten Drei-Term-Differenzengleichungen der CDOPS aufgelistet. In der Prozedur `DeltaNorm` mussten im Vergleich zur Ausführung in Verbindung mit `rodriguesDeltarec` wieder die Rollen von n und x getauscht werden.

4.3 q-Rekursions- und q-Differenzengleichungen aus q-Rodriguesformeln

Wir werden uns nun der Frage zuwenden, wie wir q-diskrete Funktionenfamilien identifizieren können. Wir werden erneut nach Zeilbergers Paradigma vorgehen. Wir werden diese Funktionenfamilien identifizieren, indem wir sowohl aus der q-hypergeometrischen Reihe als auch aus der q-Rodriguesdarstellung zum einen q-Rekursionsgleichungen, zum anderen q-Differenzengleichungen erzeugen, vergleichen und gegebenenfalls noch entsprechend viele Anfangswerte berechnen. Bei der Erzeugung dieser Gleichungen wird uns der q-Zeilberger-Algorithmus dienlich sein. Um diesen anwenden zu können, müssen wir die q-Rodriguesformeln für q-Funktionenfamilien zunächst in Summenschreibweise überführen. Wir testen die Algorithmen beispielhaft an der Klasse der q-orthogonalen Polynome.
Wir betrachten q-Rodriguesformeln der Gestalt

$$f_n(x) = g_n(x) D_q^n h_n(x) \tag{4.41}$$

$$f_n(x) = g_n(x) D_{q^{-1}}^n h_n(x) \tag{4.42}$$

$$f_n(q^{-x}) = g_n(x) \nabla_q^n h_n(x) \tag{4.43}$$

$$f_n(q^{x}) = g_n(x) \nabla_{q+}^n h_n(x), \tag{4.44}$$

wobei $\nabla_q := \frac{\nabla}{\nabla q^{-x}}$ sowie $\nabla_{q+} := \frac{\nabla}{\nabla q^{x}}$. In den q-Rodriguesformeln der q-Funktionenfamilien, bei denen x in der Basis erscheint, steht der D_q- bzw. $D_{q^{-1}}$-Operator. Die q-Rodriguesformel der q-Funktionenfamilien, bei denen x im Exponenten von q erscheint, wird mit dem ∇_q bzw. ∇_{q+}-Operator gebildet.

4.3.1 q-Rekursionsgleichungen aus q-Rodriguesformeln

In diesem Abschnitt werden wir q-Rekursionsgleichungen aus den angegebenen q-Rodriguesformeln erzeugen.
Wir geben in Tabelle 4.8 diejenigen klassischen q-orthogonalen Polynome an, die laut [KLS(2010)] einer q-Rodriguesformel der Form (4.41) genügen. Darstellung (4.41) lässt sich nun leicht mit folgendem Satz, erstmalig erwähnt in [KRM(2007), Seite 2], in eine Summe umwandeln:

q-OPS	Notation	q-Rodriguesdarstellung	w
Große q-Jacobi	$\mathrm{P}_n\left(x;a,b,c;q\right)$	$\frac{a^n c^n q^{n(n+1)}(1-q)^n}{w(x;a,b,c;q)(aq,cq;q)_n} D_q^n[w(x;aq^n,bq^n,cq^n;q)]$	$w(x;a,b,c;q) := \frac{\left(a^{-1}x,c^{-1}x;q\right)_\infty}{(x,bc^{-1}x;q)_\infty}$
Große q-Legendre	$\mathrm{P}_n\left(x;c;q\right)$	$\frac{c^n q^{n(n+1)}(1-q)^n}{(q,cq;q)_n} D_q^n[(q^{-n}x, c^{-1}q^{-n}x;q)_n]$	-
Große q-Legendre	$\mathrm{P}_n\left(x;c;q\right)$	$\frac{(1-q)^n}{(q,cq;q)_n} D_q^n[(qx^{-1}, cqx^{-1};q)_n\, x^{2n}]$	-
Große q-Laguerre	$\mathrm{P}_n\left(x;a,b;q\right)$	$\frac{a^n b^n q^{n(n+1)}(1-q)^n}{w(x;a,b;q)(aq,bq;q)_n} D_q^n[w(x;aq^n,bq^n;q)]$	$w(x;a,b;q) := \frac{\left(a^{-1}x,b^{-1}x;q\right)_\infty}{(x;q)_\infty}$
q-Laguerre	$\mathrm{L}_n^{(\alpha)}(x;q)$	$\frac{(1-q)^n}{(q;q)_n w(x;\alpha;q)} D_q^n[w(x;\alpha+n;q)]$	$w(x;\alpha;q) := \frac{x^\alpha}{(-x;q)_\infty}$
Stieltjes-Wigert	$\mathrm{S}_n\left(x;q\right)$	$\frac{q^{\binom{n+1}{2}}(1-q)^n}{w(x;q)(q;q)_n}((D_q)^n w)(q^n x;q)$	$w(x;q) := \frac{1}{(-x,-qx^{-1};q)_\infty}$
Al-Salam-Carlitz II	$\mathrm{V}_n^{(a)}(x;q)$	$\frac{a^n(q-1)^n q^{-\binom{n}{2}}}{w(x;a;q)} D_q^n[w(x;a;q)]$	$w(x;a;q) := \frac{1}{(x,a^{-1}x;q)_\infty}$
Diskrete q-Hermite II	$\tilde{\mathrm{h}}_n\left(x;q\right)$	$\frac{(q-1)^n q^{-\binom{n}{2}}}{w(x;q)} D_q^n[w(x;q)]$	$w(x;q) := \frac{1}{(-x^2;q^2)_\infty}$

Tabelle 4.8: Die q-Rodriguesformeln der q-orthogonalen Polynome, die Gleichung (4.41) genügen

Satz 4.1 *Sei $n \in \mathbb{N}_0$ und f eine auf dem Gitter $\{q^k,\ k \in \mathbb{Z}\}$ definierte Funktion. Dann gilt die folgende q-Gleichung:*

$$D_q^n f(x) = \frac{1}{(1-q)^n x^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^k x). \tag{4.45}$$

Beweis. Der Beweis kann bei [Sprenger(2009), Seite 7] nachgelesen werden. □
Erfüllt eine Funktionenfamilie also die q-Rodriguesformel (4.41), so lässt sich diese Darstellung mit Hilfe von (4.45) in folgende Summe überführen:

$$f_n(x) = \frac{g_n(x)}{(1-q)^n x^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} h_n(q^k x).$$

Hieraus kann dann der q-Zeilberger-Algorithmus eine q-Rekursionsgleichung generieren. Wir haben den q-Zeilberger-Algorithmus in Abschnitt 3.3.2 eingeführt. Erstmals vorgestellt und umgesetzt hatte ihn [Koornwinder(1993)]. Verallgemeinert haben ihn dann [BK(1999)] in der Mapleprozedur `qsumrecursion`. Diese Prozedur wird ebenfalls in [Koepf(2014)] verwendet. Wir haben sie genutzt, um für den Fall von Gleichung (4.41) q-Rekursionsgleichungen aus q-Rodriguesformeln zu generieren. Wir haben dies in der Mapleprozedur `rodriguesDqrec` umgesetzt.

```
rodriguesDqrec:=proc(g,h,q,x,sn)
  local S,n,k,rec;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  rec:=qsumrecursion(g/(1-q)^n/x^n*(-1)^k*qbinomial(n,k,q)*
  q^(binomial(k,2)-(n-1)*k)*subs(x=q^k*x,h),q,k,S(n),
  recursion=up);
end: #rodriguesDqrec
```

Wir testen die Prozedur am Beispiel der Diskreten q-Hermite II-Polynome. Diese Polynomfamilie besitzt die q-hypergeometrische Summendarstellung

$$\tilde{h}_n(x;q) = x^n {}_2\phi_1\left(\begin{matrix} q^{-n}, q^{-n+1} \\ 0 \end{matrix} \middle| q^2; -\frac{q^2}{x^2} \right) = x^n \sum_{k=0}^{n} \frac{(q^{-n};q^2)_k \, (q^{-n+1};q^2)_k}{(0;q^2)_k \, (q^2;q^2)_k} \left(-\frac{q^2}{x^2}\right)^k. \tag{4.46}$$

Wir wollen nun überprüfen, ob die in [KLS(2010), Seite 551] angegebene q-Rodriguesformel für die q-Hermite II-Polynome

$$\tilde{h}_n(x;q) = \frac{(q-1)^n q^{-\binom{n}{2}}}{w(x;q)} D_q^n[w(x;q)], \tag{4.47}$$

richtig ist, wobei $w(x;q) := \frac{1}{(-x^2;q^2)_\infty}$. Die Mapleprozedur `rodriguesDqrec` liefert folgende q-Rekursionsgleichung:

```
> wqhermite2:=(x,q)->1/qpochhammer(-x^2,q^2,infinity);
```

$$(x,\, q) \to \frac{1}{\mathrm{qpochhammer}(-x^2,\, q^2,\, \infty)}$$

```
> rodriguesDqrec(1/wqhermite2(x,q)*(q-1)^n*
> q^(-binomial(n,2)),wqhermite2(x,q),q,x,h(n));
```

$$-q^{(2\,n+1)}\,\mathrm{h}(n+2) + q^{(2\,n+1)}\,x\,\mathrm{h}(n+1) + (q^{(n+1)}-1)\,\mathrm{h}(n) = 0$$

Diese q-Rekursionsgleichung erhalten wir auch aus der q-hypergeometrischen Reihe (4.46):

```
> qsumrecursion(x^n*qphihyperterm(
> [q^(-n),q^(-n+1)],[0],q^2,-q^2/x^2,k),q,k,h(n),
> recursion=up);
```

$$-q^{(2\,n+1)}\,\mathrm{h}(n+2) + q^{(2\,n+1)}\,x\,\mathrm{h}(n+1) + (q^{(n+1)}-1)\,\mathrm{h}(n) = 0$$

Wenn wir nun noch die beiden Anfangswerte $\tilde{h}_0(x;q)$ und $\tilde{h}_1(x;q)$ sowohl für die q-Rodriguesdarstellung (4.47) als auch für die hypergeometrische Summendarstellung (4.46) überprüfen, haben wir die q-Rodriguesformel (4.47) verifiziert. Aus (4.47) ergibt sich

$$\tilde{h}_0(x;q) = \frac{w(x;q)}{w(x;q)} = 1$$

und wenn wir (4.46) betrachten, sehen wir, dass

$$\tilde{h}_0(x;q) = \sum_{k=0}^{0} \frac{(1;q^2)_k\,(q;q^2)_k}{(0;q^2)_k\,(q^2;q^2)_k}\left(-\frac{q^2}{x^2}\right)^k = 1.$$

Auf Grund der Definition des q-Pochhammersymbols ergeben die auftretenden q-Pochhammersymbole den Wert 1. Nun berechnen wir $\tilde{h}_1(x;q)$ aus (4.47):

$$\begin{aligned}\tilde{h}_1(x;q) &= \frac{(q-1)}{w(x;q)}\frac{w(x;q)-w(qx;q)}{(1-q)x} \\ &= -\left[\frac{1}{x} - \frac{w(qx;q)}{xw(x;q)}\right].\end{aligned}$$

Eine Nebenrechnung in Maple zeigt

```
> qsimpcomb(-(1/x-wqhermite2(q*x,q)/x/wqhermite2(x,q)));
```

$$x$$

und somit gilt $\tilde{h}_1(x;q) = x$. Wir berechnen $\tilde{h}_1(x;q)$ zum Vergleich aus der hypergeometrischen Summendarstellung (4.46):

$$\tilde{h}_1(x;q) = x \sum_{k=0}^{1} \frac{(q^{-1};q^2)_k \, (1;q^2)_k}{(0;q^2)_k \, (q^2;q^2)_k} \left(-\frac{q^2}{x^2} \right)^k .$$

Da das q-Pochhammersymbol $(1;q^2)_k$ für $k \geq 1$ den Wert 0 annimmt, bleibt nur der Summand für $k = 0$ übrig. Wir sehen auch auf diesem Wege, dass $\tilde{h}_1(x;q) = x$.

Damit haben wir gezeigt, dass die Anfangswerte $\tilde{h}_0(x;q)$ und $\tilde{h}_1(x;q)$ für beide Darstellungen übereinstimmen. Also ist die q-Rodriguesformel (4.47) für die q-Hermite II-Polynome bewiesen.

Betrachten wir nun ein weiteres Beispiel, das uns deutlich macht, dass unsere Algorithmen wertvolle Werkzeuge sind, um Rodriguesformeln zu überprüfen. Laut [KLS(2010)] besitzt die Familie der Stieltjes-Wigert-Polynome, gegeben durch

$$\begin{aligned} S_n(x;q) &= \frac{1}{(q;q)_n} {}_1\phi_1 \left(\begin{matrix} q^{-n} \\ 0 \end{matrix} \middle| q; -q^{n+1}x \right) \\ &= \frac{1}{(q;q)_n} \sum_{k=0}^{n} \frac{(q^{-n};q)_k}{(0;q)_k} \frac{(-q^{n+1}x)^k}{(q;q)_k} \left((-1)^k q^{\binom{k}{2}} \right), \end{aligned} \tag{4.48}$$

die q-Rodriguesdarstellung

$$S_n(x;q) = \frac{q^n(1-q)^n}{w(x;q)\,(q;q)_n} ((D_q)^n w)(q^n x;q), \tag{4.49}$$

wobei $w(x;q) := \frac{1}{(-x,-qx^{-1};q)_\infty}$.

Aus (4.48) erhalten wir mit Hilfe der Mapleprozedur `qsumrecursion` aus [Koepf(2014)] die q-Rekursionsgleichung

```
> qsumrecursion(1/qpochhammer(q,q,n)*qphihyperterm(
> [q^(-n)],[0],q,-q^(n+1)*x,k),q,k,S(n),recursion=up);
```

$$(-q^{(n+2)} + 1)\,\mathrm{S}(n+2) + (x\,q^{(2\,n+3)} + q^{(n+2)} - q - 1)\,\mathrm{S}(n+1) + q\,\mathrm{S}(n) = 0.$$

Wandeln wir (4.49) gemäß Satz 4.1 in die Summe

$$S_n(x;q) = \frac{q^n}{w(x;q)\,(q;q)_n\,(q^n x)^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^{n+k}x)$$

um, wobei wir x durch $q^n x$ ersetzt haben, können wir auf diese Summe die Mapleprozedur `qsumrecursion` anwenden und erhalten folgende q-Rekursionsgleichung:

```
> wstieltjes:=(x,q)->
> 1/qpochhammer(-x,q,infinity)/qpochhammer(-q/x,q,infinity);
```

$$(x,\, q) \to \frac{1}{\mathrm{qpochhammer}(-x,\, q,\, \infty)\,\mathrm{qpochhammer}(-\frac{q}{x},\, q,\, \infty)}$$

```
> qsumrecursion(q^n/wstieltjes(x,q)/qpochhammer(q,q,n)/
> (q^n*x)^n*(-1)^k*qbinomial(n,k,q)*q^(binomial(k,2)-(n-1)*k)
> *subs(x=q^k*q^n*x,wstieltjes(x,q)),q,k,S(n),recursion=up);
```

$$-q^{(1+2\,n)}\,(q^{(n+2)}-1)\,\mathrm{S}(n+2)+q^n\,(x\,q^{(2\,n+3)}+q^{(n+2)}-q-1)\,\mathrm{S}(n+1)+q\,\mathrm{S}(n)=0$$

Interessanterweise erhalten wir eine andere q-Rekursionsgleichung als die aus der q-hypergeometrischen Reihe erzeugte. Das heißt, (4.49) kann nicht die richtige q-Rodriguesformel der Stieltjes-Wigert-Polynome sein. Durch Abgleich der Werte für $S_n(x;q)$ für niedrige n berechnet aus (4.48) als auch aus (4.49) haben wir schließlich die korrekte q-Rodriguesdarstellung gefunden.

Satz 4.2 *Die Stieltjes-Wigert-Polynome erfüllen die q-Rodriguesformel*

$$S_n(x;q) = \frac{q^{\binom{n+1}{2}}(1-q)^n}{w(x;q)\,(q;q)_n}((D_q)^n w)(q^n x;q), \tag{4.50}$$

wobei $w(x;q) := \frac{1}{(-x,-qx^{-1};q)_\infty}$.

Beweis. Wir schauen, ob wir mit der Mapleprozedur `qsumrecursion` (siehe [Koepf(2014)]) aus (4.50) dieselbe q-Rekursionsgleichung erzeugen können wie unter Verwendung von `qsumrecursion` aus der q-hypergeometrischen Summendarstellung (4.48). Dafür wandeln wir Gleichung (4.50) wieder mit Satz 4.1 in eine Summe um:

$$S_n(x;q) = \frac{q^{\binom{n+1}{2}}}{w(x;q)\,(q;q)_n\,(q^n x)^n}\sum_{k=0}^{n}(-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} w(q^{n+k}x;q)$$

und wenden hierauf `qsumrecursion` an:

```
> qsumrecursion(q^binomial(n+1,2)/wstieltjes(x,q)/
> qpochhammer(q,q,n)/(q^n*x)^n*(-1)^k*qbinomial(n,k,q)*
> q^(binomial(k,2)-(n-1)*k)*subs(x=q^k*q^n*x,wstieltjes(x,q)),
> q,k,S(n),recursion=up);
```

$$(-q^{(n+2)}+1)\,\mathrm{S}(n+2)+(x\,q^{(2\,n+3)}+q^{(n+2)}-q-1)\,\mathrm{S}(n+1)+q\,\mathrm{S}(n)=0$$

Diese q-Rekursionsgleichung hatten wir auch aus der q-hypergeometrischen Reihe erzeugt. Überprüfen wir noch die beiden Anfangswerte $S_0(x)$ und $S_1(x)$ sowohl für Darstellung (4.48) als auch für Darstellung (4.50), haben wir die q-Rodriguesformel (4.50) der Stieltjes-Wigert-Polynome verifiziert.
(4.50) liefert uns

$$S_0(x) = \frac{w(x;q)}{w(x;q)} = 1$$

und wenn wir uns (4.48) anschauen, erhalten wir ebenfalls $S_0(x;q) = 1$. Nun berechnen wir $S_1(x)$ aus (4.50):

$$\begin{aligned} S_1(x) &= \frac{q(1-q)}{w(x;q)\,(q;q)_1}\,\frac{w(qx;q) - w(q^2x,q)}{(1-q)qx} \\ &= \frac{w(qx;q) - w(q^2x,q)}{xw(x;q)\,(q;q)_1} \end{aligned}$$

Wir schauen uns folgende Nebenrechnung in Maple an

```
> qsimpcomb((wstieltjes(q*x,q)-wstieltjes(q^2*x,q))/
> x/wstieltjes(x,q)/qpochhammer(q,q,1));
```

$$\frac{q\,x-1}{-1+q}$$

und sehen, dass $S_1(x) = \frac{qx-1}{q-1}$. Diesen Wert erhalten wir auch aus (4.48):

$$S_1(x) = \frac{1}{(q;q)_1}\sum_{k=0}^{1}\frac{(q^{-1};q)_k}{(0;q)_k}\frac{(-q^2x)^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)$$

Wir ziehen wieder Maple zu Rate:

```
> qsimpcomb(1/qpochhammer(q,q,1)*add(subs(n=1,
qphihyperterm([q^(-n)],[0],q,-q^(n+1)*x,k)),k=0..1));
```

$$\frac{q\,x-1}{-1+q}$$

und sehen, dass auch auf diesem Wege $S_1(x) = \frac{qx-1}{q-1}$.
Da die aus den beiden Darstellungen erzeugten q-Rekursionsgleichungen zweiter Ordnung und zwei Anfangswerte übereinstimmen, haben wir bewiesen, dass Darstellung (4.50) die q-Rodriguesformel der Stieltjes-Wigert-Polynome ist. □

In Tabelle 4.9 auf Seite 72 sind die q-Rekursionsgleichungen derjenigen klassischen q-orthogonalen Polynome dargestellt, die einer q-Rodriguesformel der Form (4.41) genügen. Wir führen sie in der Form von Bemerkung 2.27 an, wie

q-OPS	q-Drei-Term-Rekursion[a]
Große q-Jacobi	$-\left(q^{2n}ab-1\right)\left(q^{2n+1}ab-1\right)\left(q^{2n+2}ab-1\right)(x-1)\,P(n)$ $=\left(q^{n+1}c-1\right)\left(q^{n+1}ab-1\right)\left(q^{2n}ab-1\right)\left(q^{n+1}a-1\right)P(n+1)$ $-\left(\left(q^{n+1}c-1\right)\left(q^{n+1}ab-1\right)\left(q^{2n}ab-1\right)\left(q^{n+1}a-1\right)-\left(q^{n}ab-c\right)\left(q^{n}-1\right)\left(q^{2n+2}ab-1\right)a\left(q^{n}b-1\right)q^{n+1}\right)P(n)$ $-\left(q^{n}ab-c\right)\left(q^{n}-1\right)\left(q^{2n+2}ab-1\right)a\left(q^{n}b-1\right)q^{n+1}P(n-1)$
Große q-Legendre	$-\left(q^{n}+1\right)\left(q^{n+1}+1\right)\left(q^{2n+1}-1\right)(x-1)\,P(n)=\left(q^{n+1}c-1\right)\left(q^{n+1}-1\right)\left(q^{n}+1\right)P(n+1)$ $-\left(\left(q^{n+1}c-1\right)\left(q^{n+1}-1\right)\left(q^{n}+1\right)-\left(q^{n+1}+1\right)\left(q^{n}-1\right)\left(q^{n}-c\right)q^{n+1}\right)P(n)$ $-\left(q^{n+1}+1\right)\left(q^{n}-1\right)\left(q^{n}-c\right)q^{n+1}P(n-1)$
Große q-Laguerre	$(x-1)\,P(n)=\left(q^{n+1}a-1\right)\left(q^{n+1}b-1\right)P(n+1)$ $-\left(\left(q^{n+1}a-1\right)\left(q^{n+1}b-1\right)+q^{n+1}ab\left(q^{n}-1\right)\right)P(n)+q^{n+1}ab\left(q^{n}-1\right)P(n-1)$
q-Laguerre	$L(n)\,q^{\alpha+2n+1}x=\left(q^{n+1}-1\right)L(n+1)-\left(\left(q^{n+1}-1\right)+\left(q^{\alpha+n+1}-q\right)\right)L(n)+\left(q^{\alpha+n+1}-q\right)L(n-1)$
Stieltjes-Wigert	$S(n)\,q^{2n+1}x=\left(q^{n+1}-1\right)S(n+1)$ $-\left(\left(q^{n+1}-1\right)-q\right)S(n)-qS(n-1)$
Al-Salam-Carlitz II	$\left(q^{2n}x+aq^{n+1}-q^{2n}-aq^{n}-aq-q^{n}\right)V(n)=q^{2n}V(n+1)$ $-\left(q^{2n}-a\left(q^{n+1}-q\right)\right)V(n)-a\left(q^{n+1}-q\right)V(n-1)$
Diskrete q-Hermite II	$\left(q^{2n}x-q^{2n}+q^{n+1}-q\right)h(n)=q^{2n}h(n+1)$ $-\left(q^{2n}-\left(q^{n+1}-q\right)\right)h(n)-\left(q^{n+1}-q\right)h(n-1)$

Tabelle 4.9: Die Drei-Term-Rekursionen der q-orthogonalen Polynome, die Gleichung (4.41) genügen

[a] In den Mapleausgaben wurden q-Potenzen manuell zusammengefasst und Variablen nach Möglichkeit wie in [KLS(2010)] sortiert.

sie auch in [KLS(2010)] notiert worden sind. Dafür haben wir insbesondere die Normierungsprozedur `DqNorm` geschrieben und schließlich die Mapleprozeduren `rodriguesDqrec` und `DqNorm` nacheinander ausgeführt. Wir geben an dieser Stelle den Code der Prozedur `DqNorm` an.

```
DqNorm:=proc(rec,q,sn)
  local S,n,Q,an,cn,left,right,result;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  result:=lhs(rec)-rhs(rec);
  result:=subs(n=n-1,result);
  result:=`power/subs`(q^n=Q,result);
  an:=coeff(result,S(n+1));
  cn:=coeff(result,S(n-1));
  right:=an*S(n+1)-(an+cn)*S(n)+cn*S(n-1);
  left:=(-(result-right));
  left:=factor(left);
  result:=subs(Q=q^n,left=right);
end proc: #DqNorm
```

Wir geben in Tabelle 4.10 auf Seite 74 diejenigen klassischen q-orthogonalen Polynome an, die laut [KLS(2010)] einer q-Rodriguesformel der Form (4.42) genügen. Diese q-Rodriguesformeln können mit Hilfe von Satz 4.3 in eine Summe umgeformt werden.

q-OPS	Notation	q-Rodriguesdarstellung	w
Kleine q-Jacobi	$\mathrm{p}_n\left(x;a,b\vert q\right)$	$\frac{q^{n\alpha+\binom{n}{2}}(1-q)^n}{w(x;\alpha,\beta\vert q)(q^{\alpha+1};q)_n} D^n_{q^{-1}}[w(x;\alpha+n,\beta+n\vert q)]$	$w(x;\alpha,\beta\vert q) := \frac{(qx;q)_\infty}{\left(q^{\beta+1}x;q\right)_\infty} x^\alpha$
Kleine q-Legendre	$\mathrm{p}_n\left(x\vert q\right)$	$\frac{q^{\binom{n}{2}}(1-q)^n}{(q;q)_n} D^n_{q^{-1}}[(qx;q)_n\, x^n]$	-
Kleine q-Laguerre	$\mathrm{p}_n\left(x;a\vert q\right)$	$\frac{q^{n\alpha+\binom{n}{2}}(1-q)^n}{w(x;\alpha\vert q)(q^{\alpha+1};q)_n} D^n_{q^{-1}}[w(x;\alpha+n\vert q)]$	$w(x;\alpha\vert q) := (qx;q)_\infty\, x^\alpha$
Al-Salam-Carlitz I	$\mathrm{U}^{(a)}_n(x;q)$	$\frac{a^n q^{\frac{1}{2}n(n-3)}(1-q)^n}{w(x;a;q)} D^n_{q^{-1}}[w(x;a;q)]$	$w(x;a;q) := (qx, a^{-1}qx;q)_\infty$
Diskrete q-Hermite I	$\mathrm{h}_n\left(x;q\right)$	$\frac{(q-1)^n q^{\frac{1}{2}n(n-3)}}{w(x;q)} D^n_{q^{-1}}[w(x;q)]$	$w(x;q) := (qx, -qx;q)_\infty$

Tabelle 4.10: Die q-Rodriguesformeln der q-orthogonalen Polynome, die Gleichung (4.42) genügen

Satz 4.3 *Sei $n \in \mathbb{N}_0$ und f eine auf dem Gitter $\{q^k,\ k \in \mathbb{Z}\}$ definierte Funktion. Dann gilt die folgende q-Gleichung:*

$$D_{q^{-1}}^n f(x) = \frac{q^{\binom{n+1}{2}}}{(1-q)^n x^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^{k-n}x). \tag{4.51}$$

Beweis. Wir beweisen die Aussage durch vollständige Induktion nach n. Man sieht leicht, dass die Aussage für $n = 0$ erfüllt ist. Sei nun $n \geq 0$. Angenommen, es gelte (4.51) für dieses n. Dann ist auch folgende Gleichung erfüllt:[2]

$$\begin{aligned}
D_{q^{-1}}^{n+1} f(x) &= \frac{q^{\binom{n+1}{2}}}{(1-q)^n} D_{q^{-1}} \left(x^{-n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^{k-n}x) \right) \\
&= \frac{q^{\binom{n+1}{2}}}{(1-q)^n} \frac{q}{(q-1)x} \left(x^{-n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^{k-n}x) \right. \\
&\qquad \left. - q^n x^{-n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^{k-n-1}x) \right) \\
&= \frac{(-1)q^{\binom{n+1}{2}} q^{n+1}}{(1-q)^{n+1} x^{n+1}} \left(\sum_{k=1}^{n+1} (-1)^{k+1} \begin{bmatrix} n \\ k-1 \end{bmatrix}_q q^{\binom{k-1}{2}-(n-1)(k-1)-n} f(q^{k-n-1}x) \right. \\
&\qquad \left. - \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} f(q^{k-n-1}x) \right) \\
&= \frac{q^{\binom{n+2}{2}}}{(1-q)^{n+1} x^{n+1}} \left(\sum_{k=1}^{n+1} (-1)^k \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{k-1} \end{bmatrix}_{\boldsymbol{q}} q^{\binom{k-1}{2}-(n-1)(k-1)-n} f(q^{k-n-1}x) \right. \\
&\qquad \left. + \sum_{k=0}^{n} (-1)^k \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{k} \end{bmatrix}_{\boldsymbol{q}} \boldsymbol{q}^{\binom{k}{2}-nk+\boldsymbol{k}} f(q^{k-n-1}x) \right) \\
&= \frac{q^{\binom{n+2}{2}}}{(1-q)^{n+1} x^{n+1}} \sum_{k=0}^{n+1} (-1)^k \begin{bmatrix} \boldsymbol{n+1} \\ \boldsymbol{k} \end{bmatrix}_{\boldsymbol{q}} q^{\binom{k}{2}-nk} f(q^{k-n-1}x)
\end{aligned}$$

Damit ist die Aussage bewiesen. □

Also können q-Rodriguesformeln der Gestalt (4.42) wie folgt als Summe dargestellt werden:

$$f_n(x) = \frac{g_n(x) q^{\binom{n+1}{2}}}{(1-q)^n x^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} h_n(q^{k-n}x).$$

[2] Im letzten Schritt wird die q-Pascalregel (2.3) benutzt. Außerdem verwenden wir im Beweis, dass $\binom{n}{2} + n = \binom{n+1}{2}$.

Aus dieser Summenformel werden wir dann mit dem q-Zeilberger-Algorithmus q-Rekursionsgleichungen generieren. Dies wurde in der Mapleprozedur `rodriguesDqinversrec` umgesetzt, die auf die Prozedur `qsumrecursion` zurückgreift. In letzterer hatten [BK(1999)] den q-Zeilberger-Algorithmus implementiert. Wir geben an dieser Stelle den Code der Mapleprozedur `rodriguesDqinversrec` an.

```
rodriguesDqinversrec:=proc(g,h,q,x,sn)
  local S,n,k,rec;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  rec:=qsumrecursion(g*q^(binomial(n+1,2))/(1-q)^n/x^n*(-1)^k*
  qbinomial(n,k,q)*q^(binomial(k,2)-(n-1)*k)*
  subs(x=q^(k-n)*x,h),q,k,S(n),recursion=up);
end: #rodriguesDqinversrec
```

Wir testen die Prozedur am Beispiel der Diskreten q-Hermite I-Polynome. Diese Polynomfamilie besitzt die q-hypergeometrische Summendarstellung

$$h_n(x;q) = q^{\binom{n}{2}}{}_2\phi_1\left(\begin{matrix} q^{-n}, x^{-1} \\ 0 \end{matrix} \,\middle|\, q; -qx\right) = q^{\binom{n}{2}} \sum_{k=0}^{n} \frac{(q^{-n};q)_k\,(x^{-1};q)_k}{(0;q)_k\,(q;q)_k}(-qx)^k. \tag{4.52}$$

Wir wollen einmal mit unseren Algorithmen überprüfen, ob die in [KLS(2010), Seite 548] angegebene q-Rodriguesformel für die q-Hermite I-Polynome

$$h_n(x;q) = \frac{(q-1)^n q^{\frac{1}{2}n(n-3)}}{w(x;q)} D^n_{q^{-1}}[w(x;q)], \tag{4.53}$$

wobei $w(x;q) := (qx, -qx; q)_\infty$, tatsächlich stimmt.
Die Mapleprozedur `rodriguesDqinversrec` liefert folgende q-Rekursionsgleichung:

```
> wqhermite1:=(x,q)->qpochhammer(q*x,q,infinity)*
> qpochhammer(-q*x,q,infinity);
```

$$(x,\, q) \to \text{qpochhammer}(q\,x,\, q,\, \infty)\,\text{qpochhammer}(-q\,x,\, q,\, \infty)$$

```
> rodriguesDqinversrec(1/wqhermite1(x,q)*(q-1)^n*
> q^(n*(n-3)/2),wqhermite1(x,q),q,x,h(n));
```

$$-\mathrm{h}(n+2) + x\,\mathrm{h}(n+1) + (q^{(n+1)} - 1)\,q^n\,\mathrm{h}(n) = 0$$

Diese q-Rekursionsgleichung erhalten wir ebenso, wenn wir `qsumrecursion` aus [Koepf(2014)] auf die hypergeometrische Reihe (4.52) anwenden:

```
> qsumrecursion(q^binomial(n,2)*
> qphihyperterm([q^(-n),x^(-1)],[0],q,-q*x,k),q,k,h(n),
> recursion=up);
```

$$-\mathrm{h}(n+2) + x\,\mathrm{h}(n+1) + (q^{(n+1)} - 1)\,q^n\,\mathrm{h}(n) = 0$$

Wir überprüfen nun noch die beiden Anfangswerte $h_0(x;q)$ und $h_1(x;q)$ sowohl für die q-Rodriguesdarstellung (4.53) als auch für die q-hypergeometrische Summendarstellung (4.52). Aus (4.53) ergibt sich

$$h_0(x;q) = \frac{w(x;q)}{w(x;q)} = 1$$

und aus der Darstellung (4.52) erhalten wir:

$$h_0(x;q) = \sum_{k=0}^{0} \frac{(1;q)_k\,(x^{-1};q)_k}{(0;q)_k\,(q;q)_k}(-qx)^k = 1.$$

Wenn wir $h_1(x;q)$ aus (4.53) berechnen, erhalten wir:

$$h_1(x;q) = \frac{1-q^{-1}}{w(x;q)}\frac{w(x;q) - w(xq^{-1};q)}{(1-q^{-1})x} = \frac{1}{x} - \frac{w(xq^{-1};q)}{xw(x;q)}$$

Eine kleine Rechnung in Maple zeigt

```
> qsimpcomb(1/x-wqhermite1(x*q^(-1),q)/x/wqhermite1(x,q));
```

$$x$$

und somit gilt $h_1(x;q) = x$. Zum Abgleich berechnen wir nun $h_1(x;q)$ aus der q-hypergeometrischen Summendarstellung (4.52):

$$h_1(x;q) = q^{\binom{1}{2}} \sum_{k=0}^{1} \frac{(q^{-1};q)_k\,(x^{-1};q)_k}{(0;q)_k\,(q;q)_k}(-qx)^k.$$

Maple gibt uns dann folgenden Wert zurück:

```
> qsimpcomb(q^binomial(1,2)*add(subs(n=1,
> qphihyperterm([q^(-n),x^(-1)],[0],q,-q*x,k)),k=0..1));
```

$$x$$

Wir erhalten also denselben Wert wie bei der Berechnung gemäß (4.53). Insgesamt halten wir fest, dass die Anfangswerte $h_0(x;q)$ und $h_1(x;q)$ für beide Darstellungen übereinstimmen und damit ist (4.53) verifiziert.

In Tabelle 4.11 auf Seite 79 sind die q-Rekursionsgleichungen derjenigen klassischen q-orthogonalen Polynome aufgelistet, die einer q-Rodriguesformel der Form (4.42) genügen. Wir haben sie in der Form von Bemerkung 2.27 angegeben, wie sie auch [KLS(2010)] angeführt haben. Dafür haben wir die Maplepro-zeduren `rodriguesDqinversrec` und `DqNorm` nacheinander ausgeführt. Diese Prozeduren sind auf dem der Arbeit beiliegenden Datenträger dokumentiert. In Tabelle 4.12 auf Seite 80 geben wir sodann diejenigen klassischen q-orthogonalen Polynome an, die den q-Rodriguesformeln (4.43) und (4.44) genügen.

q-OPS	q-Drei-Term-Rekursion
Kleine q-Jacobi	$-x\left(q^{\alpha+\beta+2n}-1\right)\left(q^{\alpha+\beta+2n+1}-1\right)\left(q^{\alpha+\beta+2n+2}-1\right)p(n)$
	$=\left(q^{\alpha+n+1}-1\right)\left(q^{\alpha+\beta+n+1}-1\right)q^n\left(q^{\alpha+\beta+2n}-1\right)p(n+1)$
	$-\left(\left(q^{\alpha+n+1}-1\right)\left(q^{\alpha+\beta+n+1}-1\right)q^n\left(q^{\alpha+\beta+2n}-1\right)+\left(q^n-1\right)\left(q^{\alpha+\beta+2n+2}-1\right)q^{\alpha+n}\left(q^{\beta+n}-1\right)\right)p(n)$
	$+\left(q^n-1\right)\left(q^{\alpha+\beta+2n+2}-1\right)q^{\alpha+n}\left(q^{\beta+n}-1\right)p(n-1)$
Kleine q-Legendre	$-x\left(q^n+1\right)\left(q^{n+1}+1\right)\left(q^{2n+1}-1\right)p(n)=\left(q^n+1\right)q^n\left(q^{n+1}-1\right)p(n+1)$
	$-\left(\left(q^n+1\right)q^n\left(q^{n+1}-1\right)+\left(q^n-1\right)q^n\left(q^{n+1}+1\right)\right)p(n)$
	$+\left(q^n-1\right)q^n\left(q^{n+1}+1\right)p(n-1)$
Kleine q-Laguerre	$xp(n)=\left(q^{\alpha+n+1}-1\right)q^np(n+1)-\left(\left(q^{\alpha+n+1}-1\right)q^n+\left(q^n-1\right)q^{\alpha+n}\right)p(n)+\left(q^n-1\right)q^{\alpha+n}p(n-1)$
Al-Salam-Carlitz I	$-\left(aq^{2n-1}+aq^n-aq^{n-1}+q^n-x+1\right)U(n)=U(n+1)$
	$-\left(q+aq^{n-1}\left(q^n-1\right)\right)U(n)+aq^{n-1}\left(q^n-1\right)U(n-1)$
Diskrete q-Hermite I	$\left(q^{2n-1}+x-q^{n-1}-1\right)h(n)=h(n+1)-\left(1-q^{n-1}\left(q^n-1\right)\right)h(n)-q^{n-1}\left(q^n-1\right)h(n-1)$

Tabelle 4.11: Die Drei-Term-Rekursionen der q-orthogonalen Polynome, die Gleichung (4.42) genügen

q-OPS	Notation	q-Rodriguesdarstellung	w
q-Charlier	$\mathrm{C}_n\left(q^{-x};a;q\right)$	$\frac{(1-q)^n}{w(x;a;q)}\nabla_q^n[w(x;aq^{-n};q)]$	$w(x;a;q) := \frac{a^x q^{\binom{x+1}{2}}}{(q;q)_x}$
q-Meixner	$\mathrm{M}_n\left(q^{-x};b,c;q\right)$	$\frac{(1-q)^n}{w(x;b,c;q))}\nabla_q^n[w(x;bq^n,cq^{-n};q)]$	$w(x;b,c;q) := \frac{(bq;q)_x}{(q,-bcq;q)_x}c^x q^{\binom{x+1}{2}}$
q-Krawtchouk	$\mathrm{K}_n\left(q^{-x};p,N;q\right)$	$\frac{(1-q)^n}{w(x;p,N;q)}\nabla_q^n[w(x;pq^{2n},N-n;q)]$	$w(x;p,N;q) := \frac{\left(q^{-N};q\right)_x}{(q;q)_x}(-\frac{q}{p})^x$
q-Hahn	$\mathrm{Q}_n\left(q^{-x};\alpha,\beta;N\vert q\right)$	$\frac{(1-q)^n}{w(x;\alpha,\beta,N\vert q)}\nabla_q^n[w(x;\alpha q^n,\beta q^n,N-n\vert q)]$	$w(x;\alpha,\beta,N\vert q) := \frac{\left(\alpha q,q^{-N};q\right)_x}{(q,\beta^{-1}q^{-N};q)_x}(\alpha\beta)^{-x}$
Qtm. q-Krawtchouk	$\mathrm{K}_n^{\mathrm{qtm}}\left(q^{-x};p,N;q\right)$	$\frac{(1-q)^n}{w(x;p,N;q)}\nabla_q^n[w(x;pq^n,N-n;q)]$	$w(x;p,N;q) := \frac{\left(q^{-N};q\right)_x}{(q,p^{-1}q^{-N};q)_x}(-p)^{-x}q^{\binom{x+1}{2}}$
Affine q-Krawtchouk	$\mathrm{K}_n^{\mathrm{aff}}\left(q^{-x};p,N;q\right)$	$\frac{(-1)^n q^{-Nn+\binom{n}{2}}(1-q)^n}{(q^{-N};q)_n w(x;p,N;q)}\nabla_q^n[w(x;pq^n,N-n;q)]$	$w(x;p,N;q) := \frac{(pq;q)_x}{(q;q)_x(q;q)_{N-x}}p^{-x}$
q-Bessel	$\mathrm{y}_n\left(q^x;a;q\right)$	$\frac{a^n(1-q)^n q^{n(n-1)}}{w(x;a;q)}\nabla_{q+}^n[w(x;aq^{2n};q)]$	$w(x;a;q) := \frac{a^x q^{\binom{x}{2}}}{(q;q)_x}$

Tabelle 4.12: Die q-Rodriguesformeln der q-orthogonalen Polynome, die den Gleichungen (4.43) und (4.44) genügen

Die q-Rodriguesformeln aus Tabelle 4.12 auf Seite 80 können mit Hilfe der folgenden Sätze in eine Summe umgeformt werden:

Satz 4.4 *Sei $n \in \mathbb{N}_0$ und f eine auf dem Gitter $\{q^k,\ k \in \mathbb{Z}\}$ definierte Funktion. Dann gilt die folgende q-Gleichung:*

$$\nabla_q^n f(x) = \frac{1}{(1-q)^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{nx+\binom{n-k}{2}-\binom{n}{2}} f(x-k). \tag{4.54}$$

Beweis. Wir beweisen die Aussage durch vollständige Induktion nach n. Man sieht leicht, dass die Aussage für $n = 0$ erfüllt ist. Sei nun $n \geq 0$. Angenommen, es gelte (4.54) für dieses n. Dann isyt auch folgende Gleichung erfüllt:[3]

$$\begin{aligned}
\nabla_q^{n+1} f(x) &= \frac{1}{(1-q)^n} \nabla_q \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{nx+\binom{n-k}{2}-\binom{n}{2}} f(x-k) \right) \\
&= \frac{q^x}{(1-q)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{nx+\binom{n-k}{2}-\binom{n}{2}} f(x-k) \right. \\
&\qquad \left. - \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{nx-n+\binom{n-k}{2}-\binom{n}{2}} f(x-1-k) \right) \\
&= \frac{1}{(1-q)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{(n+1)x+\binom{n-k}{2}-\binom{n}{2}} f(x-k) \right. \\
&\quad \left. - \sum_{k=1}^{n+1} (-1)^{k-1} \begin{bmatrix} n \\ k-1 \end{bmatrix}_q q^{nx-n+x+\binom{n+1-k}{2}-\binom{n}{2}} f(x-k) \right) \\
&= \frac{1}{(1-q)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^k q^{(n+1)x+\binom{n-k}{2}-k-\binom{n+1}{2}+n} f(x-k) \right. \\
&\quad \left. - \sum_{k=1}^{n+1} (-1)^{k-1} \begin{bmatrix} n \\ k-1 \end{bmatrix}_q q^{(n+1)x+\binom{n+1-k}{2}-\binom{n+1}{2}} f(x-k) \right) \\
&= \frac{1}{(1-q)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{k} \end{bmatrix}_{\boldsymbol{q}} \boldsymbol{q^k} q^{(n+1)x+\binom{n-k}{2}+(n-k)-\binom{n+1}{2}} f(x-k) \right. \\
&\quad \left. + \sum_{k=1}^{n+1} (-1)^k \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{k-1} \end{bmatrix}_{\boldsymbol{q}} q^{(n+1)x+\binom{n+1-k}{2}-\binom{n+1}{2}} f(x-k) \right) \\
&= \frac{1}{(1-q)^{n+1}} \sum_{k=0}^{n+1} (-1)^k \begin{bmatrix} \boldsymbol{n+1} \\ \boldsymbol{k} \end{bmatrix}_{\boldsymbol{q}} q^{(n+1)x+\binom{n+1-k}{2}-\binom{n+1}{2}} f(x-k)
\end{aligned}$$

[3] Im letzten Schritt verwenden wir die q-Pascalregel (2.3). Außerdem wird im Beweis benutzt, dass $\binom{n}{2} + n = \binom{n+1}{2}$.

Damit ist die Aussage bewiesen. □

Satz 4.5 *Sei $n \in \mathbb{N}_0$ und f eine auf dem Gitter $\{q^k,\ k \in \mathbb{Z}\}$ definierte Funktion. Dann gilt die folgende q-Gleichung:*

$$\nabla_{q+}^n f(x) = \frac{1}{(q-1)^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+n-nx} f(x-k). \tag{4.55}$$

Beweis. Wir beweisen die Aussage durch vollständige Induktion nach n. Man sieht leicht, dass die Aussage für $n = 0$ erfüllt ist. Sei nun $n \geq 0$. Angenommen, es gelte (4.55) für dieses n. Dann ist auch folgende Gleichung erfüllt:[4]

$$\begin{aligned}
\nabla_{q+}^{n+1} f(x) &= \frac{1}{(q-1)^n} \nabla_{q+} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+n-nx} f(x-k) \right) \\
&= \frac{q^{-x+1}}{(q-1)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+n-nx} f(x-k) \right. \\
&\qquad \left. - \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+n-nx+n} f(x-1-k) \right) \\
&= \frac{1}{(q-1)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+(n+1)-(n+1)x} f(x-k) \right. \\
&\quad \left. - \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+(n+1)-(n+1)x+n} f(x-1-k) \right) \\
&= \frac{1}{(q-1)^{n+1}} \left(\sum_{k=0}^{n} (-1)^k \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{k} \end{bmatrix}_{\boldsymbol{q}} q^{\binom{k}{2}+(n+1)-(n+1)x} f(x-k) \right. \\
&\quad \left. + \sum_{k=1}^{n+1} (-1)^k \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{k-1} \end{bmatrix}_{\boldsymbol{q}} \boldsymbol{q^{n-k+1}} q^{\binom{k-1}{2}+(k-1)+(n+1)-(n+1)x} f(x-k) \right) \\
&= \frac{1}{(q-1)^{n+1}} \sum_{k=0}^{n+1} (-1)^k \begin{bmatrix} \boldsymbol{n+1} \\ \boldsymbol{k} \end{bmatrix}_{\boldsymbol{q}} q^{\binom{k}{2}+(n+1)-(n+1)x} f(x-k)
\end{aligned}$$

Damit ist die Aussage bewiesen. □

[4] Im letzten Schritt verwenden wir die q-Pascalregel (2.4). Außerdem wird im Beweis benutzt, dass $\binom{n}{2} + n = \binom{n+1}{2}$.

Mit Hilfe von Satz 4.4 können wir somit die q-Rodriguesformel (4.43) in folgende Summendarstellung überführen:

$$f_n(q^{-x}) = \frac{g_n(x)}{(1-q)^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{nx+\binom{n-k}{2}-\binom{n}{2}} h_n(x-k) \tag{4.56}$$

Wenden wir den q-Zeilberger-Algorithmus auf diese Summe an, erhalten wir die gewünschte q-Rekursionsgleichung. Explizit verwenden wir wieder die Mapleprozedur `qsumrecursion`, siehe [BK(1999)] und [Koepf(2014)], die den q-Zeilberger-Algorithmus umsetzt. Die folgende Prozedur generiert nun eine q-Rekursionsgleichung aus einer q-Rodriguesformel der Gestalt (4.43) entsprechend unseren Überlegungen.

```
rodriguesNablaqrec:=proc(g,h,q,x,sn)
  local S,n,k,rec;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  rec:=qsumrecursion(g/(1-q)^n*(-1)^k*qbinomial(n,k,q)*
  q^(n*x+binomial(n-k,2)-binomial(n,2))*subs(x=x-k,h),q,k,S(n),
  recursion=up);
end: #rodriguesNablaqrec
```

Wir überprüfen die Funktionsweise dieser Mapleprozedur einmal an Hand der q-Meixner-Polynome. Deren q-hypergeometrische Summendarstellung lautet

$$\mathrm{M}_n\left(q^{-x};b,c;q\right) = {}_2\phi_1\left(\left.\begin{matrix} q^{-n}, q^{-x} \\ bq \end{matrix}\right| q; -\frac{q^{n+1}}{c}\right) = \sum_{k=0}^{n} \frac{(q^{-n};q)_k (q^{-x};q)_k}{(bq;q)_k (q;q)_k} \left(-\frac{q^{n+1}}{c}\right)^k. \tag{4.57}$$

Wir werden sogleich sehen, ob die in [KLS(2010), Seite 490] angegebene q-Rodriguesformel für die q-Meixner-Polynome

$$\mathrm{M}_n\left(q^{-x};b,c;q\right) = \frac{(1-q)^n}{w(x;b,c;q)} \nabla_q^n [w(x;bq^n,cq^{-n};q)] \tag{4.58}$$

mit $w(x;b,c;q) := \frac{(bq;q)_x}{(q,-bcq;q)_x} c^x q^{\binom{x+1}{2}}$ korrekt ist.
Die Mapleprozedur `rodriguesNablaqrec` erzeugt uns daraus folgende q-Rekursionsgleichung:

```
> wqmeixner:=(x,b,c,q)->qpochhammer(b*q,q,x)/
> qpochhammer(q,q,x)/qpochhammer(-b*c*q,q,x)*
> c^x*q^binomial(x+1,2);
```

$$(x,\, b,\, c,\, q) \to \frac{\mathrm{qpochhammer}(q\,b,\, q,\, x)\, c^{x}\, q^{\mathrm{binomial}(x+1,\,2)}}{\mathrm{qpochhammer}(q,\, q,\, x)\,\mathrm{qpochhammer}(-b\,c\,q,\, q,\, x)}$$

```
> rodriguesNablaqrec((1-q)^n/wqmeixner(x,b,c,q),
> wqmeixner(x,b*q^n,c*q^(-n),q),q,x,M(n));
```

$$\begin{aligned}&(q^{(2+n)}\,b-1)\,c\,q^{x}\,\mathrm{M}(n+2)+\\&(-c\,q^{(2+n+x)}\,b-q^{(2\,n+3)}-c\,q^{(2+n+x)}+q^{(2+n+x)}+c\,q^{(x+1)}+c\,q^{x})\\&\mathrm{M}(n+1)+(q^{(1+n)}+c)\,(q^{(1+n)}-1)\,q^{(x+1)}\,\mathrm{M}(n)=0\end{aligned}$$

Wir erhalten diese Ausgabe ebenso, wenn wir die Mapleprozedur `qsumrecursion` auf die q-hypergeometrische Reihe (4.57) anwenden:

```
> qsumrecursion(qphihyperterm([q^(-n),q^(-x)],[b*q],
> q,k,M(n),recursion=up);
```

$$\begin{aligned}&(q^{(2+n)}\,b-1)\,c\,q^{x}\,\mathrm{M}(n+2)+\\&(-c\,q^{(2+n+x)}\,b-q^{(2\,n+3)}-c\,q^{(2+n+x)}+q^{(2+n+x)}+c\,q^{(x+1)}+c\,q^{x})\\&\mathrm{M}(n+1)+(q^{(1+n)}+c)\,(q^{(1+n)}-1)\,q^{(x+1)}\,\mathrm{M}(n)=0\end{aligned}$$

Stimmen noch die beiden Anfangswerte $\mathrm{M}_0\left(q^{-x};b,c;q\right)$ und $\mathrm{M}_1\left(q^{-x};b,c;q\right)$ sowohl aus Gleichung (4.57) als auch aus der q-Rodriguesformel (4.58) berechnet überein, haben wir die q-Rodriguesdarstellung (4.58) der q-Meixner-Polynome verifiziert.
Wir beginnen mit der Berechnung von $\mathrm{M}_0\left(q^{-x};b,c;q\right)$ aus der q-Rodriguesdarstellung (4.58). Es zeigt sich, dass

$$\mathrm{M}_0\left(q^{-x};b,c;q\right)=\frac{w(x;b,c;q)}{w(x;b,c;q)}=1.$$

Andererseits sehen wir leicht, dass in der q-hypergeometrischen Reihe (4.57) der q-Meixner-Polynome für $n=0$ nur der erste Summand übrig bleibt und auch auf diesem Wege

$$\mathrm{M}_0\left(q^{-x};b,c;q\right)=1$$

gilt. Berechnen wir nun $\mathrm{M}_1\left(q^{-x};b,c;q\right)$ aus der q-Rodriguesformel (4.58), erhalten wir

$$\mathrm{M}_1\left(q^{-x};b,c;q\right)=\frac{1-q}{w(x;b,c;q)}\nabla_q[w(x;bq,cq^{-1};q)]$$

und Maple gibt schließlich folgenden Wert aus:

```
> Nablaq:=proc(f,x,q)
> (f-subs(x=x-1,f))/(q^(-x)-q^(-x+1));
> end proc:
> simplify(qsimpcomb((1-q)/wqmeixner(x,b,c,q)*
> Nablaq(wqmeixner(x,b*q,c*
> q^(-1),q),x,q)));
```

$$\frac{b\,c\,q-q-c+q^{(1-x)}}{c\,(b\,q-1)}$$

Doch auch Darstellung (4.57) liefert uns diesen Wert, wie uns folgende Mapleausgabe verdeutlicht:

```
> simplify(qsimpcomb(add(subs(n=1,qphihyperterm([q^(-n),
> q^(-x)],[b*q],q
> ,-q^(n+1)/c,k)),k=0..1)));
```

$$\frac{b\,c\,q-q-c+q^{(1-x)}}{c\,(b\,q-1)}$$

Da sowohl die aus den Darstellungen (4.57) und (4.58) generierten q-Rekursionsgleichungen zweiter Ordnung als auch die beiden daraus berechneten Anfangswerte übereinstimmen, ist Darstellung (4.58) tatsächlich die q-Rodriguesformel der q-Meixner-Polynome.
Wir wenden uns an dieser Stelle dem noch verbleibenden Fall zu, in dem eine q-Funktionenfamilie einer q-Rodriguesformel der Gestalt

$$f_n(q^x) = g_n(x)\nabla_{q+}^n h_n(x) \tag{4.59}$$

genügt. In [KLS(2010)] wird nur die q-Rodriguesformel der q-Bessel-Polynome mit Hilfe des Operators ∇_{q+} angegeben, wie Tabelle 4.12 zeigt. Wir werden an dieser Stelle auch für diese Klasse einen Algorithmus angeben, der aus einer q-Rodriguesformel der Form (4.59) bzw. (4.44) eine q-Rekursionsgleichung erzeugt. In Satz 4.5 hatten wir bereits gezeigt, wie man den in Gleichung (4.59) vorkommenden n-fach angewandten Operator ∇_{q+} (angewandt auf eine Funktion) in eine Summe umwandelt. Damit können wir die q-Rodriguesdarstellung (4.59) als folgende Summe schreiben:

$$f_n(q^x) = \frac{g_n(x)}{(q-1)^n}\sum_{k=0}^{n}(-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+n-nx} h_n(x-k). \tag{4.60}$$

Auf die so bestimmte Summendarstellung kann dann der q-Zeilberger-Algorithmus, also die Prozedur `qsumrecursion`, angewandt werden. Dieses Vorgehen ist in der Mapleprozedur `rodriguesNablaqrec2` zusammengefasst, deren Code wir hier angeben.

```
rodriguesNablaqrec2:=proc(g,h,q,x,sn)
  local S,n,k,rec;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  rec:=qsumrecursion(g/(q-1)^n*(-1)^k*qbinomial(n,k,q)*
  q^(binomial(k,2)+n-n*x)*subs(x=x-k,h),q,k,S(n),recursion=up);
end: #rodriguesNablaqrec2
```

Wir werden diese Mapleprozedur am Beispiel der q-Bessel-Polynome testen und überprüfen einmal deren in [KLS(2010), Seite 528] angegebene q-Rodriguesdarstellung. Diese q-orthogonale Polynomfamilie besitzt die q-hypergeometrische Reihe

$$\mathrm{y}_n(q^x;a;q) = {}_2\phi_1\left(\begin{matrix} q^{-n}, -aq^n \\ 0 \end{matrix} \,\middle|\, q;q^{x+1}\right) = \sum_{k=0}^{n} \frac{\left(q^{-n};q\right)_k \left(-aq^n;q\right)_k}{\left(0;q\right)_k \left(q;q\right)_k} \left(q^{x+1}\right)^k \tag{4.61}$$

und [KLS(2010)] zufolge lautet ihre q-Rodriguesdarstellung

$$\mathrm{y}_n\left(q^x;a;q\right) = \frac{a^n(1-q)^n q^{n(n-1)}}{w(x;a;q)} \nabla_{q+}^n [w(x;aq^{2n};q)], \tag{4.62}$$

wobei $w(x;a;q) := \frac{a^x q^{\binom{x}{2}}}{(q;q)_x}$. Wenn wir die Mapleprozedur `rodriguesNablaqrec2` hierauf anwenden, erhalten wir folgende q-Rekursionsgleichung

```
> wqbessel:=(x,a,q)->a^x*q^binomial(x,2)/qpochhammer(q,q,x);
```

$$(x,\, a,\, q) \to \frac{a^x\, q^{\mathrm{binomial}(x,\, 2)}}{\mathrm{qpochhammer}(q,\, q,\, x)}$$

```
> rodriguesNablaqrec2(a^n*(1-q)^n*q^(n*(n-1)),
> wqbessel(x,a*q^(2*n),q),q,x,y(n));
```

$$-q^{(n+1)}\left(q^{(1+2\,n)}\,a+1\right)\left(q^{(n+1)}\,a+1\right)\mathrm{y}(n+2)-\left(q^{(2\,n+2)}\,a+1\right)\left(q^{(4\,n+4+x)}\,a^2\right.$$
$$+q^{(3\,n+3)}\,a+q^{(x+2\,n+3)}\,a-q^{(2\,n+2)}\,a+q^{(1+2\,n+x)}\,a-q^{(1+2\,n)}\,a$$
$$\left.-q^{(n+1)}+q^x\right)\mathrm{y}(n+1)+q^{(1+2\,n)}\,a\left(q^{(n+1)}-1\right)\left(q^{(2\,n+3)}\,a+1\right)\mathrm{y}(n)=0,$$

die wir ebenso erhalten, wenn wir die Mapleprozedur `qsumrecursion` auf die hypergeometrische Reihe (4.61) anwenden:

```
> qsumrecursion(qphihyperterm([q^(-n),-a*q^n],[0],q,q^(x+1),k),
> q,k,y(n),recursion=up);
```

$$-q^{(n+1)}\left(q^{(1+2\,n)}\,a+1\right)\left(q^{(n+1)}\,a+1\right)\mathrm{y}(n+2)-\left(q^{(2\,n+2)}\,a+1\right)\left(q^{(4\,n+4+x)}\,a^2\right.$$
$$+q^{(3\,n+3)}\,a+q^{(x+2\,n+3)}\,a-q^{(2\,n+2)}\,a+q^{(1+2\,n+x)}\,a-q^{(1+2\,n)}\,a$$
$$\left.-q^{(n+1)}+q^x\right)\mathrm{y}(n+1)+q^{(1+2\,n)}\,a\left(q^{(n+1)}-1\right)\left(q^{(2\,n+3)}\,a+1\right)\mathrm{y}(n)=0$$

Falls nun noch die beiden Anfangswerte $\mathrm{y}_0\left(q^x;a;q\right)$ und $\mathrm{y}_1\left(q^x;a;q\right)$ sowohl für (4.62) als auch (4.61) übereinstimmen, haben wir bewiesen, dass (4.62) die q-Rodriguesdarstellung der q-Bessel-Polynome ist.
Wir fangen damit an, $\mathrm{y}_0\left(q^x;a;q\right)$ gemäß der q-Rodriguesdarstellung (4.62) zu berechnen und stellen fest, dass

$$\mathrm{y}_0\left(q^x;a;q\right)=\frac{w(x;a;q)}{w(x;a;q)}=1.$$

Aus der q-hypergeometrischen Reihe berechnet besteht die Summe aus nur einem Summanden und es ergibt sich ebenfalls der Wert $\mathrm{y}_0\left(q^x;a;q\right)=1$.
Bei der Berechnung von $\mathrm{y}_1\left(q^x;a;q\right)$ aus (4.62) bemerken wir, dass

$$\mathrm{y}_1\left(q^x;a;q\right)=\frac{a(1-q)}{w(x;a;q)}\nabla_{q+}[w(x;aq^{2n};q)] \tag{4.63}$$

und Maple berechnet für uns

```
> Nablaq2:=proc(f,x,q)
> (f-subs(x=x-1,f))/(q^(x)-q^(x-1));
> end proc:

> simplify(qsimpcomb(a*(1-q)/wqbessel(x,a,q)*
> Nablaq2(wqbessel(x,a*q^2,q),x,q)));
```

$$-q^{(x+1)}\,a-q^x+1$$

Denselben Wert erhalten wir gemäß (4.61), wie uns folgende Mapleausgabe zeigt:

```
> combine(qsimpcomb(add(subs(n=1,qphihyperterm(
> [q^(-n),-a*q^(n)],[0],q,q^(x+1),k)),k=0..1)),power);
```

$$-q^{(x+1)}\,a-q^x+1$$

Da sowohl die aus (4.62) und (4.61) erzeugten q-Rekursionsgleichungen zweiter Ordnung als auch zwei Anfangswerte übereinstimmen, ist die q-Rodriguesdarstellung (4.62) der q-Bessel-Polynome verifiziert.
In Tabelle 4.13 auf Seite 89 sind die q-Rekursionsgleichungen derjenigen klassischen q-orthogonalen Polynome aufgeführt, die einer q-Rodriguesformel der Form (4.43) oder (4.44) genügen. Wir haben sie in der Form von Bemerkung 2.27 angegeben, wie sie auch [KLS(2010)] dargestellt haben. Wir haben für diese Normierung die Mapleprozedur `NablaqNorm` geschrieben. Dabei haben wir die Prozeduren `rodriguesNablaqrec` bzw. `rodriguesNablaqrec2` sowie `NablaqNorm` nacheinander auf die q-Rodriguesdarstellungen dieser q-orthogonalen Polynomfamilien angewandt. Wir geben an dieser Stelle den Code der Mapleprozedur `NablaqNorm` an.

```
NablaqNorm:=proc(rec,q,sn)
  local S,n,X,K,J,an,cn,left,right,result;
  if type(sn,function) then
    S:=op(0,sn); n:=op(1,sn);
  else
    n:=op(1,sn);
  end if;
  result:=lhs(rec)-rhs(rec);
  result:=subs(n=n-1,result);
  result:=`power/subs`(q^x=1/X,result);
  result:=collect(numer(normal(result)),S,factor);
  an:=coeff(result,S(n+1));
  cn:=coeff(result,S(n-1));
  right:=an*S(n+1)-(an+cn)*S(n)+cn*S(n-1);
  left:=-(result-right);
  left:=`power/subs`(q^N=J,left);
  left:=`power/subs`(q^n=K,left);
  left:=factor(left);
  left:=subs({X=q^(-x),K=q^n,J=q^N},left);
  right:=subs(X=q^(-x),right);
  result:=combine(left,power)=map(f->combine(f,power),right);
end proc: #NablaqNorm
```

q-OPS	q-Drei-Term-Rekursion
q-Charlier	$q^{2n+1}\left(q^{-x}-1\right)C\left(n\right)=-aC\left(n+1\right)-\left(-a+q\left(q^{n}-1\right)\left(a+q^{n}\right)\right)C\left(n\right)+q\left(q^{n}-1\right)\left(a+q^{n}\right)C\left(n-1\right)$
q-Meixner	$q^{2n+1}\left(q^{-x}-1\right)M\left(n\right)=c\left(bq^{n+1}-1\right)M\left(n+1\right)$
	$-\left(c\left(bq^{n+1}-1\right)+q\left(q^{n}-1\right)\left(c+q^{n}\right)\right)M\left(n\right)+q\left(q^{n}-1\right)\left(c+q^{n}\right)M\left(n-1\right)$
q-Krawtchouk	$q^{N}\left(pq^{2n-1}+1\right)\left(pq^{2n}+1\right)\left(pq^{2n+1}+1\right)\left(q^{-x}-1\right)K\left(n\right)=\left(q^{N}-q^{n}\right)\left(pq^{n}+1\right)\left(pq^{2n-1}+1\right)K\left(n+1\right)$
	$-\left(\left(q^{N}-q^{n}\right)\left(pq^{n}+1\right)\left(pq^{2n-1}+1\right)+pq^{2n-1}\left(pq^{n+N}+1\right)\left(q^{n}-1\right)\left(pq^{2n+1}+1\right)\right)K\left(n\right)$
	$+pq^{2n-1}\left(pq^{n+N}+1\right)\left(q^{n}-1\right)\left(pq^{2n+1}+1\right)K\left(n-1\right)$
q-Hahn	$q^{N}\left(\alpha\beta q^{2n}-1\right)\left(\alpha\beta q^{2n+1}-1\right)\left(\alpha\beta q^{2n+2}-1\right)\left(q^{-x}-1\right)Q\left(n\right)$
	$=\left(q^{N}-q^{n}\right)\left(\alpha q^{n+1}-1\right)\left(\alpha\beta q^{n+1}-1\right)\left(\alpha\beta q^{2n}-1\right)Q\left(n+1\right)$
	$-\left(\left(q^{N}-q^{n}\right)\left(\alpha q^{n+1}-1\right)\left(\alpha\beta q^{n+1}-1\right)\left(\alpha\beta q^{2n}-1\right)+\alpha q^{n}\left(q^{n}-1\right)\left(\alpha\beta q^{n+N+1}-1\right)\left(\beta q^{n}-1\right)\left(\alpha\beta q^{2n+2}-1\right)\right)Q\left(n\right)$
	$+\alpha q^{n}\left(q^{n}-1\right)\left(\alpha\beta q^{n+N+1}-1\right)\left(\beta q^{n}-1\right)\left(\alpha\beta q^{2n+2}-1\right)Q\left(n-1\right)$
Qtm.	$pq^{2n+N+1}\left(q^{-x}-1\right)K\left(n\right)$
q-Krawtchouk	$=\left(q^{N}-q^{n}\right)K\left(n+1\right)-\left(-q^{n}+q^{N}+q^{N+1}\left(q^{n}-1\right)\left(pq^{n}-1\right)\right)K\left(n\right)+q^{N+1}\left(q^{n}-1\right)\left(pq^{n}-1\right)K\left(n-1\right)$
Affine	$q^{N}\left(q^{-x}-1\right)K\left(n\right)$
q-Krawtchouk	$=-\left(pq^{n+1}-1\right)\left(q^{N}-q^{n}\right)K\left(n+1\right)-\left(-\left(pq^{n+1}-1\right)\left(q^{N}-q^{n}\right)+pq^{n}\left(q^{n}-1\right)\right)K\left(n\right)+pq^{n}\left(q^{n}-1\right)K\left(n-1\right)$
q-Bessel	$\left(aq^{2n-1}+1\right)\left(aq^{2n}+1\right)\left(aq^{2n+1}+1\right)y\left(n\right)=-\left(aq^{n}+1\right)q^{-x}q^{n}\left(aq^{2n-1}+1\right)y\left(n+1\right)$
	$-\left(-\left(aq^{n}+1\right)q^{n}\left(aq^{2n-1}+1\right)q^{-x}+\left(aq^{2n-1}\right)\left(q^{n}-1\right)\left(aq^{2n+1}+1\right)q^{-x}\right)y\left(n\right)$
	$+aq^{2n-1}\left(q^{n}-1\right)\left(aq^{2n+1}+1\right)q^{-x}y\left(n-1\right)$

Tabelle 4.13: Die Drei-Term-Rekursionen der q-orthogonalen Polynome, die den Gleichungen (4.43) und (4.44) genügen

4.3.2 q-Differenzengleichungen aus q-Rodriguesformeln

In diesem Abschnitt werden wir q-Differenzengleichungen aus den angegebenen q-Rodriguesformeln erzeugen. Wir beginnen mit denjenigen q-Funktionenfamilien, deren q-Rodriguesformel in der Gestalt (4.41) angegeben ist. Wir hatten in Tabelle 4.8 auf Seite 66 klassische q-orthogonale Polynomfamilien aufgelistet, deren q-Rodriguesformel in [KLS(2010)] auf diese Weise angegeben war. Die q-Rodriguesformel

$$f_n(x) = g_n(x) D_q^n h_n(x)$$

lässt sich mit Satz 4.1 in die Summe

$$f_n(x) = \frac{g_n(x)}{(1-q)^n x^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} h_n(q^k x)$$

umformen, worauf dann der q-Zeilberger-Algorithmus angewandt werden kann. Wir führen den q-Zeilberger-Algorithmus in Form der Prozedur `qsumdiffeq` aus [Koepf(2014)] aus, die q-Differenzengleichungen erzeugt. Bei Eingabe der Werte für g_n und h_n sowie anderer Standardparameter gibt uns die Mapleprozedur `rodriguesDqdiffeq` die gesuchte q-Differenzengleichung zurück. Wir geben an dieser Stelle ihren Code an.

```
rodriguesDqdiffeq:=proc(g,h,q,n,sx)
  local S,x,k,diffeq;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  diffeq:=qsumdiffeq(g/(1-q)^n/x^n*(-1)^k*qbinomial(n,k,q)*
  q^(binomial(k,2)-(n-1)*k)*subs(x=q^k*x,h),q,k,S(x),
  evalqdiff=true);
end: #rodriguesDqdiffeq
```

Wir verzichten für diese Beispielklasse darauf, die Funktionsweise dieser Prozedur an einem Beispiel zu testen, werden dies aber für den Fall ausführlich

zeigen, in dem die q-Rodriguesformel folgende Form hat:

$$f_n(x) = g_n(x) D_{q^{-1}}^n h_n(x).$$

In Tabelle 4.14 auf Seite 92 geben wir die q-Differenzengleichungen derjenigen q-orthogonalen Polynome an, für die in der Literatur eine q-Rodriguesformel der Gestalt (4.41) angeführt wird. Dafür haben wir die mit `rodriguesDqdiffeq` erzeugten q-Differenzengleichungen noch wie in [KLS(2010)] normiert, wobei wir uns nach Bemerkung 2.27 mit entsprechender Vertauschung der Variablen n und x gerichtet haben. Diese Normierung wurde mit Hilfe der Mapleprozedur `DqNorm2` vorgenommen.

```
DqNorm2:=proc(diffeq,q,sx)
  local S,x,Q,ax,cx,left,right,result;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  result:=lhs(diffeq)-rhs(diffeq);
  result:=subs(x=x/q,result);
  result:=`power/subs`(q^n=Q,result);
  ax:=coeff(result,S(q*x));
  cx:=coeff(result,S(x/q));
  right:=ax*S(q*x)-(ax+cx)*S(x)+cx*S(x/q);
  left:=(-(result-right));
  left:=factor(left);
  left:=subs(Q=q^n,left);
  right:=subs(Q=q^n,right);
  result:=combine(left,power)=map(f->combine(f,power),right);
end proc: #DqNorm2
```

Wir bemerken außerdem, dass die q-Differenzengleichung für die Stieltjes-Wigert-Polynome aus der korrigierten Fassung der q-Rodriguesformel entsprechend Satz 4.2 generiert wurde.

q-OPS	q-Drei-Term-Differenzengleichung
Große q-Jacobi	$-$[a]
Große q-Legendre	$\left(q^n-1\right)\left(q^{n+1}-1\right)x^2P\left(x\right)=q^{n+1}\left(x-1\right)\left(x-c\right)P\left(qx\right)$ $-\left(q^{n+1}\left(x-1\right)\left(x-c\right)+q^{n+2}\left(\frac{x}{q}-1\right)\left(\frac{x}{q}-c\right)\right)P\left(x\right)+q^{n+2}\left(\frac{x}{q}-1\right)\left(\frac{x}{q}-c\right)P\left(\frac{x}{q}\right)$
Große q-Laguerre	$\left(q^n-1\right)x^2P\left(x\right)=abq^{n+1}\left(x-1\right)P\left(qx\right)$ $-\left(abq^{n+1}\left(x-1\right)-q^n\left(x-aq\right)\left(x-bq\right)\right)P\left(x\right)-q^n\left(x-aq\right)\left(x-bq\right)P\left(\frac{x}{q}\right)$
q-Laguerre	$q^\alpha\left(q^n-1\right)xL\left(x\right)=q^\alpha\left(x+1\right)L\left(qx\right)-\left(q^\alpha\left(x+1\right)+1\right)L\left(x\right)+L\left(\frac{x}{q}\right)$
Stieltjes-Wigert	$x\left(q^n-1\right)S\left(x\right)=xS\left(qx\right)-\left(x+1\right)S\left(x\right)+S\left(\frac{x}{q}\right)$
Al-Salam-Carlitz II	$\left(q^n-1\right)x^2V\left(x\right)=\left(1-x\right)\left(a-x\right)V\left(qx\right)-\left(\left(1-x\right)\left(a-x\right)+aq\right)V\left(x\right)+aqV\left(\frac{x}{q}\right)$
Diskrete q-Hermite II	$\left(q^n-1\right)x^2h\left(x\right)=\left(x^2+1\right)h\left(qx\right)-\left(x^2+q+1\right)h\left(x\right)+qh\left(\frac{x}{q}\right)$

Tabelle 4.14: Die Drei-Term-Differenzengleichungen der q-orthogonalen Polynome, die Gleichung (4.41) genügen

[a] Maple konnte auch nach mehreren Stunden kein Ergebnis ausgeben.

Beschäftigen wir uns nun mit dem Fall, dass die q-Rodriguesformel einer q-Funktionenfamilie in der Form

$$f_n(x) = g_n(x) D_{q^{-1}}^n h_n(x) \tag{4.64}$$

vorliegt. q-orthogonale Polynomfamilien solcher Art hatten wir in Tabelle 4.10 auf Seite 74 aufgeführt. Wir können eine q-Rodriguesformel wie in (4.64) mit Hilfe von Satz 4.3 in folgende Summe umwandeln:

$$f_n(x) = \frac{g_n(x) q^{\binom{n+1}{2}}}{(1-q)^n x^n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} h_n(q^{k-n}x).$$

Auf diese findet dann der q-Zeilberger-Algorithmus Anwendung. Wir benutzen dafür wieder die Mapleprozedur `qsumdiffeq` aus [Koepf(2014)]. Zusammengefasst erzeugt nun die Mapleprozedur `rodriguesDqinversdiffeq` bei Kenntnis einer q-Rodriguesformel der Form (4.64) (bzw. (4.42)) eine q-Differenzengleichung für diese q-Funktionenfamilie. Wir geben hier ihren Code an.

```
rodriguesDqinversdiffeq:=proc(g,h,q,n,sx)
  local S,x,k,diffeq;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  diffeq:=qsumdiffeq(g*q^(binomial(n+1,2))/(1-q)^n/x^n*(-1)^k*
  qbinomial(n,k,q)*q^(binomial(k,2)-(n-1)*k)*subs(x=q^(k-n)*x,h),
  q,k,S(x),evalqdiff=true);
end: #rodriguesDqinversdiffeq
```

Wir testen die Funktionsweise dieser Mapleprozedur am Beispiel der Kleinen q-Legendre-Polynome. Deren q-hypergeometrische Summendarstellung lautet

$$\mathrm{p}_n(x|q) = {}_2\phi_1\left(\begin{matrix} q^{-n}, q^{n+1} \\ q \end{matrix} \middle| q; qx \right) = \sum_{k=0}^{n} \frac{(q^{-n};q)_k (q^{n+1};q)_k}{(q;q)_k (q;q)_k} (qx)^k. \tag{4.65}$$

Wir werden nun mit Hilfe der Mapleprozedur `rodriguesDqinversdiffeq` überprüfen, ob die in [KLS(2010), Seite 488] angegebene q-Rodriguesformel

$$\mathrm{p}_n(x|q) = \frac{q^{\binom{n}{2}}(1-q)^n}{(q;q)_n} \left(D_{q^{-1}}\right)^n [(qx;q)_n x^n] \tag{4.66}$$

der Kleinen q-Legendre-Polynome richtig ist. Dazu erzeugen wir aus (4.66) mittels `rodriguesDqinversdiffeq` folgende q-Differenzengleichung:

```
> rodriguesDqinversdiffeq(q^binomial(n,2)*(1-q)^n/
> qpochhammer(q,q,n),qpochhammer(q*x,q,n)*x^n,q,n,p(x));
```

$$-(q\,x-1)\,q^n\,\mathrm{p}(x)+((q^n)^2\,q^2\,x+q\,x-2\,q^n)\,\mathrm{p}(q\,x)-(q^2\,x-1)\,q^n\,\mathrm{p}(q^2\,x)=0$$

Diese q-Differenzengleichung können wir unter Verwendung der Mapleprozedur `qsumdiffeq`, die in [Koepf(2014), Kapitel 10] dokumentiert ist, auch aus der q-hypergeometrischen Reihe (4.65) erzeugen, wie uns folgende Mapleausgabe zeigt:

```
> qsumdiffeq(qphihyperterm([q^(-n),q^(n+1)],[q],q,q*x,k),
> q,k,p(x),evalqdiff=true);
```

$$-(q\,x-1)\,q^n\,\mathrm{p}(x)+((q^n)^2\,q^2\,x+q\,x-2\,q^n)\,\mathrm{p}(q\,x)-(q^2\,x-1)\,q^n\,\mathrm{p}(q^2\,x)=0$$

Wir stellen fest, dass die aus (4.66) erzeugte q-Differenzengleichung mit der aus (4.65) erzeugten übereinstimmt. Also haben wir die Darstellung (4.66) dann als q-Rodriguesformel der Kleinen q-Legendre-Polynome identifiziert, wenn auch noch zwei Anfangswerte jeweils berechnet gemäß (4.65) bzw. (4.66) übereinstimmen, da es sich bei der erzeugten q-Differenzengleichung um eine zweiter Ordnung handelt. Wir berechnen zuerst $\mathrm{p}_n(0|q)$ aus (4.65) und sehen sogleich, dass $\mathrm{p}_n(0|q)=1$, da nur der Summand für $k=0$ stehen bleibt. Um $\mathrm{p}_n(0|q)$ aus der q-Rodriguesformel (4.66) zu zeigen, wandeln wir (4.66) zunächst mit Hilfe von Satz 4.3 in eine Summe um und erhalten

$$\mathrm{p}_n(x|q)=\frac{q^{n^2}}{(q;q)_n}\sum_{k=0}^{n}(-1)^k\begin{bmatrix}n\\k\end{bmatrix}_q q^{\binom{k}{2}-(n-1)k}\left(q^{k+1-n}x;q\right)_n q^{nk-n^2}$$

Setzen wir nun $x=0$ ein, ergibt sich folgende Summe:

$$\mathrm{p}_n(0|q)=\frac{q^{n^2}}{(q;q)_n}\sum_{k=0}^{n}(-1)^k\begin{bmatrix}n\\k\end{bmatrix}_q q^{\binom{k}{2}-(n-1)k}q^{nk-n^2} \tag{4.67}$$

Auf diese Summe wenden wir dann den q-Zeilberger-Algorithmus in Form der Mapleprozedur `qsumrecursion` aus [Koepf(2014)] an:

```
> qsumrecursion(q^(n^2)/qpochhammer(q,q,n)*(-1)^k*
> qbinomial(n,k,q)*q^(binomial(k,2)-(n-1)*k)*q^(n*k-n^2),
> q,k,p(n));
```

$$\mathrm{p}(n)-\mathrm{p}(n-1)=0$$

Die so erhaltene Rekursion gibt uns zu verstehen, dass $\mathrm{p}_n(0|q)$ für alle n konstant ist. Wir lassen Maple deshalb noch den Wert für $n = 0$ ausrechnen:

```
> qsimpcomb(sum(subs(n=0,q^(n^2)/qpochhammer(q,q,n)*(-1)^k*
> qbinomial(n,k,q)*q^(binomial(k,2)-(n-1)*k)*q^(n*k-n^2)),
> k=0..0));
```

$$1$$

Somit erhalten wir aus (4.66) $\mathrm{p}_n(0|q)$=1.
Gehen wir nun zur Berechnung von $\mathrm{p}_n(1|q)$ über. Aus der q-hypergeometrischen Reihe (4.65) ergibt sich

$$\mathrm{p}_n(1|q) = \sum_{k=0}^{n} \frac{(q^{-n};q)_k (q^{n+1};q)_k}{(q;q)_k (q;q)_k} q^k. \tag{4.68}$$

Aus der q-Rodriguesdarstellung (4.66) ergibt sich nach Anwendung von Satz 4.3

$$\mathrm{p}_n(1|q) = \frac{q^{n^2}}{(q;q)_n} \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}-(n-1)k} \left(q^{k+1-n};q\right)_n q^{nk-n^2}. \tag{4.69}$$

Wir werden nun erneut mit Zeilbergers Paradigma argumentieren und die Gleichheit dieser beiden Summen zeigen, indem wir daraus dieselben q-Rekursionsgleichungen für $\mathrm{p}_n(1|q)$ erzeugen und entsprechend viele Anfangswerte aus beiden Summen berechnen. Wir beginnen mit (4.68) und erzeugen daraus mit dem q-Zeilberger-Algorithmus, explizit mit der Mapleprozedur `qsumrecursion` aus [Koepf(2014)], folgende q-Rekursionsgleichung erster Ordnung

```
> RE1:=qsumrecursion(qpochhammer(q^(-n),q,k)*
> qpochhammer(q^(n+1),q,k)*q^k/qpochhammer(q,q,k)/
> qpochhammer(q,q,k),q,k,p(n));
```

$$-\mathrm{p}(n) - q^n \mathrm{p}(n-1) = 0$$

Nun möchten wir diese q-Rekursionsgleichung auch aus der Summe (4.69) generieren. Wir gehen wie gehabt vor und Maple erzeugt uns

```
> RE2:=qsumrecursion(q^(n^2)/qpochhammer(q,q,n)*(-1)^k*
> qbinomial(n,k,q)*q^(binomial(k,2)-(n-1)*k)*
> qpochhammer(q^(k+1-n),q,n)*q^(n*k-n^2),q,k,p(n));
```

$$-\mathrm{p}(n) - q^n \mathrm{p}(n-1) = 0$$

Wir haben in der Tat dieselbe q-Rekursionsgleichung erhalten. Nun bleibt nur noch zu prüfen, ob $\mathrm{p}_0(1|q)$ jeweils aus den beiden Summen berechnet denselben Wert ergibt. Wir sehen gleich, dass wir sowohl aus (4.68) als auch aus

(4.69) $\mathrm{p}_0\,(1|q) = 1$ erhalten. Damit haben wir auch den zweiten Anfangswert $\mathrm{p}_n\,(1|q)$ der q-Differenzengleichung für beide Darstellungen überprüft und insgesamt gezeigt, dass es sich bei Darstellung (4.66) um die q-Rodriguesformel der Kleinen q-Legendre-Polynome handelt.
In Tabelle 4.15 auf Seite 100 die q-Differenzengleichungen derjenigen q-orthogonalen Polynome angeführt, für die wir in der Literatur eine q-Rodriguesformel der Form (4.64) (bzw. (4.42)) gefunden haben. Dazu haben wir die mit `rodriguesDqinversdiffeq` erzeugten q-Differenzengleichungen noch mit der Normierungsprozedur `DqNorm2` wie in [KLS(2010)] normiert.

Wir wollen nun nochmals q-Funktionenfamilien identifizieren, von denen eine q-Rodriguesformel der Gestalt

$$f_n(q^{-x}) = g_n(x)\nabla_q^n h_n(x) \tag{4.70}$$

bekannt ist. Wir werden an Stelle von q-Rekursionsgleichungen q-Differenzengleichungen erzeugen.
Wir hatten in Tabelle 4.12 q-orthogonale Polynome gelistet, die in [KLS(2010)] in dieser Form angegeben wurden. Mit Hilfe von Satz 4.4 konnten wir diese Darstellung in die Summe

$$f_n(q^{-x}) = \frac{g_n(x)}{(1-q)^n}\sum_{k=0}^{n}(-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{nx+\binom{n-k}{2}-\binom{n}{2}} h_n(x-k)$$

umwandeln. Auf eine solche Summe können wir nun auch den q-Zeilberger-Algorithmus anwenden, um eine q-Differenzengleichung zu erzeugen. Verglichen mit derjenigen q-Differenzengleichung, die aus der q-hypergeometrischen Reihe der q-Funktionenfamilie hervorgeht, und nach Abgleich entsprechend vieler Anfangswerte ist die q-Rodriguesformel auch für diesen Fall verifiziert und die entsprechende q-Funktionenfamilie wurde erkannt.
Da bei q-orthogonalen Polynomen, deren q-Rodriguesformel mit Hilfe des ∇_q-Operators entsprechend der Form (4.70) (bzw. (4.43)) dargestellt ist, x im Exponenten von q auftritt, werden wir die Mapleprozedur `qsumrecursion` aus [Koepf(2014)] anwenden, die mit einer solchen Darstellung arbeiten kann. Wir haben eine Mapleprozedur geschrieben, die bei Kenntnis einer q-Rodriguesformel der Gestalt (4.70) (bzw. (4.43)) eine q-Differenzengleichung ausgibt.
Wir geben ihren Code an.

```
rodriguesNablaqdiffeq:=proc(g,h,q,n,sx)
  local S,x,k,diffeq;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  diffeq:=qsumrecursion(g/(1-q)^n*(-1)^k*qbinomial(n,k,q)*
  q^(n*x+binomial(n-k,2)-binomial(n,2))*subs(x=x-k,h),q,k,S(x),
  recursion=up);
end: #rodriguesNablaqdiffeq
```

Mit der Prozedur `rodriguesNablaqdiffeq` können wir nun eine q-Differenzengleichung aus einer q-Rodriguesformel der Gestalt (4.70) erzeugen. Wir können sie daraufhin mit derjenigen q-Differenzengleichung vergleichen, die wir mit der Mapleprozedur `qsumrecursion` aus der q-hypergeometrischen Reihe der zu testenden q-Funktionenfamilie erzeugt haben. Wenn die q-Differenzengleichungen übereinstimmen, überprüfen wir noch entsprechend viele Anfangswerte. Damit ist die q-Rodriguesformel dann bewiesen und die q-Funktionenfamilie wurde identifiziert.
Wir werden kein Beispiel angeben, um die Funktionalität von `rodriguesNablaqdiffeq` konkret zu demonstrieren, da die Beispiele dieser Klasse dem Beispiel der Kleinen q-Legendre-Polynome für die Mapleprozedur `rodriguesDqinversdiffeq` entsprechen.
Wir geben der Vollständigkeit halber in Tabelle 4.16 auf Seite 101 die q-Differenzengleichungen derjenigen q-orthogonalen Polynome an, für die wir in der Literatur, etwa in [KLS(2010)], eine q-Rodriguesformel der Form (4.70) (bzw. (4.43)) finden können.
Wir haben sie erzeugt, indem wir `rodriguesNablaqdiffeq` und eine Normierungsprozedur angewandt haben, die die q-Differenzengleichung in der Form von Bemerkung 2.27 mit entsprechender Vertauschung der Variablen n und x ausgibt.
Im Folgenden geben wir diese Normierungsprozedur, die wir `NablaqNorm2` genannt haben, an.

```
NablaqNorm2:=proc(diffeq,q,sx)
  local S,x,X,K,J,ax,cx,right,left,result;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  result:=lhs(diffeq)-rhs(diffeq);
  result:=subs(x=x-1,result);
  result:=`power/subs`(q^x=1/X,result);
  result:=collect(numer(normal(result)),S,factor);
  ax:=coeff(result,S(x+1));
  cx:=coeff(result,S(x-1));
  right:=ax*S(x+1)-(ax+cx)*S(x)+cx*S(x-1);
  left:=-(result-right);
  left:=`power/subs`(q^N=J,left);
  left:=`power/subs`(q^n=K,left);
  left:=factor(left);
  left:=subs({X=q^(-x),K=q^n,J=q^N},left);
  right:=subs(X=q^(-x),right);
  result:=combine(left,power)=map(f->combine(f,power),right);
end proc: #NablaqNorm2
```

Wir betrachten nun noch q-Funktionenfamilien, deren q-Rodriguesformel in der Form (4.44), also in der Form

$$f_n(q^x) = g_n(x)\nabla_{q+}^n h_n(x) \tag{4.71}$$

angegeben ist, und identifizieren diese, indem wir auch für diese q-Funktionenfamilien eine q-Differenzengleichung sowohl aus der q-Rodriguesformel als auch der q-hypergeometrischen Reihe erzeugen. Wir hatten in Tabelle 4.12 auf Seite 80 klassische q-orthogonale Polynome angeführt, deren q-Rodriguesformel in [KLS(2010)] in dieser Form angegeben wird. Die q-Rodriguesformel (4.71) lässt sich mit Satz 4.5 in die Summe

$$f_n(q^x) = \frac{g_n(x)}{(q-1)^n}\sum_{k=0}^{n}(-1)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}+n-nx} h_n(x-k). \tag{4.72}$$

überführen, worauf dann der q-Zeilberger-Algorithmus angewandt werden kann. Wir führen den q-Zeilberger-Algorithmus in der Form von `qsumrecursion` aus, da x im Exponenten von q erscheint. Insgesamt erzeugt nun die Mapleprozedur `rodriguesNablaqdiffeq2` bei Kenntnis einer q-Rodriguesformel der Form (4.71) (bzw. (4.44)) die gesuchte q-Differenzengleichung. Wir geben an dieser Stelle ihren Code an.

```
rodriguesNablaqdiffeq2:=proc(g,h,q,n,sx)
  local S,x,k,diffeq;
  if type(sx,function) then
    S:=op(0,sx); x:=op(1,sx);
  else
    x:=op(1,sx);
  end if;
  diffeq:=qsumrecursion(g/(q-1)^n*(-1)^k*qbinomial(n,k,q)*
  q^(binomial(k,2)+n-n*x)*subs(x=x-k,h),q,k,S(x),recursion=up);
end: #rodriguesNablaqdiffeq2
```

Wir geben kein Beispiel für diese Funktionenklasse an. Der Vollständigkeit halber geben wir in Tabelle 4.16 auf Seite 101 die q-Differenzengleichung der q-Bessel-Polynome an, die die einzige klassische q-orthogonale Polynomfamilie ist, für die in [KLS(2010)] eine q-Rodriguesformel der Form (4.44) angegeben wird. Wir haben die angegebene q-Differenzengleichung erhalten, indem wir die Mapleprozeduren `rodriguesNablaqdiffeq2` und `NablaqNorm2` nacheinander ausgeführt haben. Damit haben wir auch für die letzte der vier q-Rodriguesformeln (4.41)-(4.44) gezeigt, wie man aus ihr eine q-Differenzengleichung generiert.

q-OPS	q-Drei-Term-Differenzengleichung
Kleine q-Jacobi	$\left(q^n-1\right)\left(q^{n+1}q^\alpha q^\beta-1\right)xp\left(x\right)=$
	$q^nq^\alpha\left(q^\beta qx-1\right)p\left(qx\right)-\left(q^nq^\alpha\left(q^\beta qx-1\right)+q^n\left(x-1\right)\right)p\left(x\right)+q^n\left(x-1\right)p\left(\frac{x}{q}\right)$
Kleine q-Legendre	$\left(q^n-1\right)\left(q^{n+1}-1\right)xp\left(x\right)=$
	$q^n\left(qx-1\right)p\left(qx\right)-\left(q^n\left(qx-1\right)+q^n\left(x-1\right)\right)p\left(x\right)+q^n\left(x-1\right)p\left(\frac{x}{q}\right)$
Kleine q-Laguerre	$\left(q^n-1\right)xp\left(x\right)=q^nq^\alpha p\left(qx\right)-\left(q^nq^\alpha+q^n\left(1-x\right)\right)p\left(x\right)+q^n\left(1-x\right)p\left(\frac{x}{q}\right)$
Al-Salam-Carlitz I	$-\left(q^n-1\right)x^2U\left(x\right)=$
	$aq^{n-1}U\left(qx\right)-\left(aq^{n-1}+q^n\left(1-x\right)\left(a-x\right)\right)U\left(x\right)+q^n\left(1-x\right)\left(a-x\right)U\left(\frac{x}{q}\right)$
Diskrete q-Hermite I	$-q\left(q^n-1\right)x^2h\left(x\right)=$
	$-q^nh\left(qx\right)-\left(-q^n+q^{n+1}\left(x-1\right)\left(x+1\right)\right)h\left(x\right)+q^{n+1}\left(x-1\right)\left(x+1\right)h\left(\frac{x}{q}\right)$

Tabelle 4.15: Die Drei-Term-Differenzengleichungen der q-orthogonalen Polynome, die Gleichung (4.42) genügen

q-OPS	q-Drei-Term-Differenzengleichung
q-Charlier	$-\left(q^n-1\right)q^{-x}C\left(x\right)=-a\,C\left(x+1\right)-\left(-a+1-q^{-x}\right)C\left(x\right)+\left(1-q^{-x}\right)C\left(x-1\right)$
q-Meixner	$-\left(q^n-1\right)q^{-2x}M\left(x\right)=$
	$-c\left(q^{-x}-bq\right)M\left(x+1\right)-\left(-c\left(q^{-x}-bq\right)-\left(q^{-x}-1\right)\left(q^{-x}+bc\right)\right)M\left(x\right)-\left(q^{-x}-1\right)\left(q^{-x}+bc\right)M\left(x-1\right)$
q-Krawtchouk	$-\left(q^n-1\right)\left(pq^n+1\right)q^{-x+N}K\left(x\right)=$
	$q^n\left(q^{-x+N}-1\right)K\left(x+1\right)-\left(q^n\left(q^{-x+N}-1\right)-pq^{n+N}\left(q^{-x}-1\right)\right)K\left(x\right)-pq^{n+N}\left(q^{-x}-1\right)K\left(x-1\right)$
q-Hahn	$-\left(q^n-1\right)\left(\alpha\,\beta\,q^{n+1}-1\right)q^{-2x+N}Q\left(x\right)=-q^n\left(q^{-x}-\alpha\,q\right)\left(q^{-x+N}-1\right)Q\left(x+1\right)$
	$-\left(-q^n\left(q^{-x}-\alpha q\right)\left(q^{-x+N}-1\right)-\alpha q^n\left(q^{-x}-1\right)\left(\beta q^{-x+N+1}-1\right)\right)Q\left(x\right)-\alpha q^n\left(q^{-x}-1\right)\left(\beta q^{-x+N+1}-1\right)Q\left(x-1\right)$
Qtm.	$p\left(q^n-1\right)q^{-2x+N+1}K\left(x\right)=-q\left(q^{-x+N}-1\right)K\left(x+1\right)$
q-Krawtchouk	$-\left(-q\left(q^{-x+N}-1\right)+\left(q^{-x}-1\right)\left(pq^{-x+N+1}-1\right)\right)K\left(x\right)+\left(q^{-x}-1\right)\left(pq^{-x+N+1}-1\right)K\left(x-1\right)$
Affine	$-\left(q^n-1\right)q^{-2x+N}K\left(x\right)=q^n\left(q^{-x}-pq\right)\left(q^{-x+N}-1\right)K\left(x+1\right)$
q-Krawtchouk	$-\left(q^n\left(q^{-x}-pq\right)\left(q^{-x+N}-1\right)-pq^n\left(q^{-x}-1\right)\right)K\left(x\right)-pq^n\left(q^{-x}-1\right)K\left(x-1\right)$
q-Bessel	$-\left(q^n-1\right)\left(aq^n+1\right)y\left(X\right)=-aq^n y\left(X+1\right)-\left(-aq^n-q^n\left(q^{-X}-1\right)\right)y\left(X\right)-q^n\left(q^{-X}-1\right)y\left(X-1\right)$[a]

Tabelle 4.16: Die Drei-Term-Differenzengleichung der q-orthogonalen Polynome, die den Gleichungen (4.43) und (4.44) genügen

[a] wobei $X=q^x$

Kapitel 5

Ausblick

Es stellt sich dem interessierten Forscher die Frage, ob man nicht auch aus einer erzeugenden Funktion

$$F(z) = \sum_{n=0}^{\infty} a_n P_n(x) z^n,$$

die alle Informationen über eine analytische Polynomfamilie $P_n(x)$ enthält, Rekursions- und Differentialgleichungen erzeugen kann.
Almkvist und Zeilberger ([AZ(1990)]) sind dieser Frage für den stetigen Fall nachgegangen. Wir hatten uns in Abschnitt 4.1.1 dieser Arbeit in Erinnerung gerufen, dass eine analytische Funktion $f(x)$ in einer komplexen Variable x auf einem einfach zusammenhängenden Gebiet $D \subset \mathbb{C}$ unendlich oft differenzierbar ist. Für ihre Ableitungen gilt dann nach der Cauchyschen Integralformel

$$f^{(n)}(x) = \frac{n!}{2\pi i} \int_\gamma \frac{f(t)}{(t-x)^{n+1}} dt. \tag{5.1}$$

Diese Formel haben Almkvist und Zeilberger gemeinsam mit dem Satz von Taylor genutzt, um aus $F(z)$ eine Integralschreibweise für die Koeffizienten $P_n(x)$ zu generieren:

$$P_n(x) = \frac{1}{a_n} \frac{F^(n)(0)}{n!} = \frac{1}{a_n} \frac{1}{2\pi i} \int_\gamma \frac{F(t)}{(t-x)^{n+1}} dt$$

Auf diese Integralschreibweise konnte dann der stetige Zeilberger-Algorithmus angewandt werden. Wolfram Koepf hat dieses Vorgehen in [Koepf(2014)] in den Prozeduren `GFrecursion` und `GFdiffeq` implementiert, wobei er auf seine Prozeduren `intrecursion` und `intdiffeq`, die den stetigen Zeilberger-Algorithmus ausführen, zurückgreift.

Dieses Problem nun auch für diskrete Funktionenfamilien zu lösen, ist sicher von Interesse. Dabei müsste die Variable z in der erzeugenden Funktion vermutlich in einer anderen Basisschreibweise, etwa den fallenden Faktoriellen, auftauchen. Darüber hinaus lohnt es sich, die Fragestellung ebenso für den q-Fall zu untersuchen, wobei eine geeignete q-erzeugende Funktion der gegebenen Polynomfamilie $P_n(x)$ (davon gibt es dann mehrere) mit einer passenden q-Taylorformel (auch hiervon gibt es eine ganze Reihe) in eine Summe umgeformt werden muss. Dies bleibt dem interessierten Forscher für die Zukunft.

Literaturverzeichnis

[Aigner(2006)] M. Aigner. Diskrete Mathematik. Vieweg, Wiesbaden, 6. Auflage 2006.

[AZ(1990)] G. Almkvist und D. Zeilberger. The method of differentiating under the integral sign. *J. Symbolic Comput.*, 10:571–591, 1990.

[Bateman(1931)] H. Bateman. The k-function, a particular case of the confluent hypergeometric function. *Trans. Amer. Math. Soc.*, 33:817–831, 1931.

[BS(1976)] H. Behnke und F. Sommer. *Theorie der analytischen Funktionen einer komplexen Veränderlichen*. Springer, Berlin, 1976.

[Bochner(1929)] S. Bochner. Über Sturm-Liouvillesche Polynomsysteme. *Mat. Z.*, 29:730–736, 1929.

[Böing(1998)] H. Böing. *Theorie und Algorithmen zur q-hypergeometrischen Summation*. Diplomarbeit, Freie Universität Berlin, 1998.

[BK(1999)] H. Böing und W. Koepf. Algorithms for q-hypergeometric summation in computer algebra. *J. Symbolic Comput.*, 28:777–799, 1999.

[BD(2011)] J. Borwein und K. Devlin. *Experimentelle Mathematik – Eine beispielorientierte Einführung*. Spektrum, Heidelberg, 2011.

[Cusick(1989)] T. W. Cusick. Recurrences for Sums of powers of binomial coefficients. *J. Combin. Theory Ser. A*, 52:77–83, 1989.

[Franel(1894)] J. Franel. *L'Intermédiare des Mathématiciens*, Volume 1:45–47, 1894.

[Franel(1895)] J. Franel. *L'Intermédiare des Mathématiciens*, Volume 2:33–35, 1895.

[GR(1990)] G. Gasper und M. Rahman. *Basic Hypergeometric Series*, Volume 35 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1990.

[Gosper(1978)] R. W. J. Gosper. Decision procedure for indefinite hypergeometric summation. *Proc. Natl. Acad. Sci. USA*, 75:40–42, 1978.

[GKP(1994)] R. L. Graham, D. E. Knuth und O. Patashnik. *Concrete Mathematics, 2nd edn. A Foundation for Computer Science*. Addison-Wesley, Reading Massachussets, 1994.

[Hahn(1949)] W. Hahn. Über Orthogonalpolynome, die q-Differenzengleichungen genügen. *Math. Nachr.*, 2:4–34, 1949.

[KC(2002)] V. Kac und P. Cheung. *Quantum Calculus*. Springer, New York, 2002.

[KLS(2010)] R. Koekoek, P. A. Lesky und R. F. Swarrtouw. *Hypergeometric Orthogonal Polynomials and Their q-Analogues*. Springer, Berlin, 2010.

[Koepf(1992)] W. Koepf. Power series in computer algebra. *J. Symbolic Comput.*, 13:581–603, 2007.

[Koepf(2006)] W. Koepf. *Computeralgebra – Eine algorithmisch orientierte Einführung*. Springer, Berlin, 2006.

[Koepf(2014)] W. Koepf. *Hypergeometric Summation, an Algorithmic Approach to Summation and Special Function Identities*. Advanced Lectures in Mathematics. Springer, London, 2014; `http://www.hypergeometric-summation.org/`.

[KRM(2007)] W. Koepf, P. M. Rajković und S. D. Marinković. Properties of q-holonomic functions. *J. Difference Equ. Appl.*, 13(7):621–638, 2007.

[Koornwinder(1993)] T. H. Koornwinder. On Zeilberger's algorithm and its q-analogue. *J. Comput. Appl. Math.*, (48):91–111, 1993.

[Lesky(2005)] P. A. Lesky. *Eine Charakterisierung der klassischen kontinuierlichen-, diskreten- und q-Orthogonalpolynome.* Shaker, Aachen, 2005.

[NU(1988)] A. F. Nikiforov und V. B. Uvarov. *Special Functions of Mathematical Physics.* Birkhäuser, Basel, Boston, 1988.

[NSU(1991)] A. F. Nikiforov, S. K. Suslov und V. B. Uvarov. *Classical Orthogonal Polynomials of a Discrete Variable.* Springer, Berlin, 1991.

[OLBC(2010)] F. Olver, D. Lozier, R. Boisvert und C. Clark. *NIST Handbook of Mathematical Functions.* Cambridge University Press, New York, 2010; siehe `http://dlmf.nist.gov/`.

[Perlstadt(1987)] M. A. Perlstadt. Some recurrences for sums of powers of binomial coefficients. *J. Number Theory*, 27:304–309, 1987.

[Petkovšek(1992)] M. Petkovšek. Hypergeometric solutions of linear recurrences with polynomial coefficients. *J. Symbolic Comput.*, 14:243–264, 1992.

[Sprenger(2009)] T. Sprenger. *Algorithmen für q-holonome Funktionen und q-hypergeometrische Reihen.* PhD thesis, Universität Kassel, 2009.

[vanHoeij(1998)] M. van Hoeij. Finite singularities and hypergeometric solutions of linear recurrence equations. *J. Pure Appl. Algebra*, 139:109–131, 1998.

[Walter(2000)] W. Walter. Gewöhnliche Differentialgleichungen. Springer, Berlin, 7. Auflage 2000.

[WZ(1992)] H. S. Wilf und D. Zeilberger. An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities. *Invent. Math.*, 108:575–633, 1992.

[Zeilberger(1990)] D. Zeilberger. A fast algorithm for proving terminating hypergeometric identities. *Discrete Math.*, 80:207–211, 1990.

[Zeilberger(1991)] D. Zeilberger. The method of creative telescoping. *J. Symbolic Comput.*, 11:195–204, 1991.

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Dissertation selbstständig, ohne unerlaubte Hilfe Dritter angefertigt und andere als die in der Dissertation angegebenen Hilfsmittel nicht benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen sind, habe ich als solche kenntlich gemacht. Dritte waren an der inhaltlich-materiellen Erstellung der Dissertation nicht beteiligt; insbesondere habe ich hierfür nicht die Hilfe eines Promotionsberaters in Anspruch genommen. Kein Teil dieser Arbeit ist in einem anderen Promotions- oder Habilitationsverfahren verwendet worden.