Dissertation

Algorithmen für q-holonome Funktionen und q-hypergeometrische Reihen

zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
(Dr. rer. nat.)
im Fachbereich Mathematik
der Universität Kassel

vorgelegt von

Torsten Sprenger

eingereicht bei

Prof. Dr. Wolfram Koepf

im

April 2009

Tag der mündlichen Prüfung
25. Juni 2009

Erstgutachter
Prof. Dr. Wolfram Koepf
Universität Kassel

Zweitgutachter
Prof. Dr. Tom H. Koornwinder
Universität Amsterdam

Inhaltsverzeichnis

Kapitel 1

Einführung

Die q-Analysis ist eine spezielle Diskretisierung der Analysis auf einem Gitter, welches eine geometrische Folge darstellt. Das Gitter besteht also für $k \in \mathbb{Z}$ aus Punkten der Form q^k bzgl. einer Basis q. Die q-Analysis findet insbesondere in der Quantenphysik eine breite Anwendung, ist aber auch in der Theorie der q-orthogonalen Polynome und speziellen Funktionen sowie in der Zahlentheorie von großer Bedeutung. Die betrachteten mathematischen Objekte aus der q-Welt wie z.B. q-Funktionen, q-Ableitungen, q-Integrale, q-Differentialgleichungen, q-Rekursionsgleichungen und q-Potenzreihen weisen meist eine recht komplizierte Struktur auf und es liegt daher nahe, sie mit Computeralgebrasystemen zu behandeln. Allerdings sind in aktuellen Computeralgebrasystemen wie beispielsweise Maple ([GLM08]), Mathematica ([Wol99]) oder auch MuPAD ([CO04]) bisher noch wenige bis gar keine q-Funktionen bzw. q-Funktionalitäten implementiert. Maple ist das einzige General Purpose Computeralgebrasystem, welches ein Package namens `QDifferenceEquations` *bereitstellt, das einige wenige q-Funktionen enthält und mit dem man beispielsweise gewisse q-Rekursionsgleichungen lösen kann. Da die q-Analysis immer mehr an Bedeutung gewinnt, herrscht also dringender Bedarf an neuen und effizienten q-Algorithmen und deren Implementierungen. Während zu einigen wichtigen klassischen Algorithmen (Gosper-, Zeilberger- und Petkovšek-Algorithmus) bereits q-Varianten existieren ([WZ92], [Koo93], [APP98], [BK99]), trifft dies auf Algorithmen für holonome Funktionen und hypergeometrische Reihen nur bedingt zu ([KK09]).*

In der vorliegenden Dissertation werden Algorithmen für q-holonome Funktionen und q-hypergeometrische Reihen vorgestellt. Alle Algorithmen sind in dem Maple-Package `qFPS` *implementiert. Wir geben einen kurzen Überblick über den Aufbau der Dissertation.*

In diesem Kapitel werden zunächst Grundlagen geschaffen, die zum weiteren Studium der Dissertation notwendig sind. Neben einigen Definitionen und Notationen werden die beiden Operatoren eingeführt, mit denen wir uns über die gesamte Arbeit beschäftigen werden, der q-Differential- und der q-Shiftoperator. Ferner präsentieren wir viele q-Funktionen, wie z.B. elementare q-Funktionen, q-Exponentialfunktionen, q-trigonometrische Funktionen, klassische q-orthogonale Polynome und die verallgemeinerte q-hypergeometrische Funktion.

Das zweite Kapitel beleuchtet, wie sich die eingeführten q-Funktionen unter den beiden Operationen, der q-Ableitung und dem q-Shift verhalten. Während die q-Ableitungen und die q-Shifts der elementaren q-Funktionen, der q-Exponentialfunktionen und der q-trigonometrischen Funktionen kanonisch sind und keine Wahl zulassen, hat man bei den klassischen q-orthogonalen Polynomen mehrere Möglichkeiten, die q-Ableitungen und q-Shifts darzustellen. Aus diesem Grund führen wir neue q-Ableitungs- und q-Shiftregeln ein, die wir aus den Strukturformeln der q-orthogonalen Polynome ([KS01]) ableiten und die für uns eine optimale Struktur aufweisen.

Nachdem wir im zweiten Kapitel q-Ableitungen und q-Shifts aller behandelten q-Funktionen bestimmt haben, können wir nun diese nutzen, um möglichst einfache, so genannte q-holonome q-Differential- und q-Rekursionsgleichungen aufzustellen. Der Algorithmus, den wir dazu im dritten Kapitel formulieren, basiert auf dem Algorithmus aus [KS94] und ist abhängig von der Darstellung der q-Ableitungen und der q-Shifts. Diese Darstellungen werden im zweiten Kapitel so gewählt, dass der Algorithmus eine optimale q-Differential- bzw. q-Rekursionsgleichung zu einer gegebenen q-Funktion bestimmen kann. Die q-Funktionen, welche eine q-holonome Differential- bzw. Rekursionsgleichung erfüllen, nennen wir q-holonom. Es werden alle q-holonomen Rekursionsgleichungen der in Kapitel 1

vorgestellten q-Funktionen elementar hergeleitet ([KRM07]). Für die verallgemeinerte q-hypergeometrische Funktion stellen wir eine neue explizite Formel für die q-holonome Rekursionsgleichung auf. In einem weiteren Abschnitt werden wir sehen, dass die Menge der q-holonomen Funktionen bzgl. $+$ und $\cdot$ einen kommutativen Ring bilden. Algorithmen, die zu zwei q-holonomen Funktionen $f(x)$ und $g(x)$ die q-holonome Differentialgleichung der Summe $f(x)+g(x)$, des Produkts $f(x)\cdot g(x)$ und unter gewissen Voraussetzungen der Komposition $(f\circ g)(x)$ bestimmen, wurden bereits in [KRM07] vorgestellt. Diese Algorithmen werden vereinfacht und effizienter gestaltet, indem wir q-holonome Rekursionsgleichungen an Stelle von q-holonomen Differentialgleichungen betrachten.

Im vierten Kapitel geht es um polynomiale, rationale und q-hypergeometrische Lösungen q-holonomer Rekursionsgleichungen. Um im nächsten Kapitel q-hypergeometrische Potenzreihenlösungen ermitteln zu können, benötigen wir einen effizienten Algorithmus, der im Stande ist, q-hypergeometrische Lösungen von komplizierten Rekursionen hoher Ordnung zu bestimmen. Nachdem zunächst einige nützliche Hilfsmittel zur Verfügung gestellt werden, die wir in diesem Kapitel benötigen, zeigen wir, wie man eine obere Gradschranke für polynomiale Lösungen einer q-holonomen Rekursionsgleichung und daraus auch alle polynomialen Lösungen bestimmt ([APP98],[BK99]). Um rationale Lösungen zu finden, muss man zunächst eine Nennerschranke dieser Lösungen ermitteln und kann dann polynomiale Lösungen einer modifizierten Rekursionsgleichung bestimmen, die zu rationalen Lösungen korrespondieren ([Abr95]). Zur Theorie q-hypergeometrischer Lösungen werden der q-Petkovšek-Algorithmus ([APP98]), eine q-Variante des Van-Hoeij-Algorithmus ([Hoe98a], [CH05]) und eine modifizierte Variante des q-Petkovšek-Algorithmus ([Hor08]) vorgestellt. In einem Laufzeitvergleich verschiedener Implementationen dieser Algorithmen werden wir sehen, dass die Prozedur `qHypergeomSolveRE` *aus dem Maple-Package* `qFPS` *mit dem modifizierten q-Petkovšek-Algorithmus die schnellste Prozedur aller Maple-Implementationen ist.*

Das fünfte Kapitel beschäftigt sich mit q-hypergeometrischen Potenzreihen. Wir verfolgen dazu die Philosophie von Kac/Cheung ([KC02]) und wählen als Basis der q-Potenzreihen nicht etwa die i.Allg. verwendeten q-Pochhammersymbole, sondern die im ersten Kapitel eingeführten q-Potenzen. Diese Wahl liegt u.a. begründet in der Tatsache, dass alle klassischen q-orthogonalen Polynome darstellbar sind als q-hypergeometrische Potenzreihe bzgl. dieser Basis. Wir legen zunächst die allgemeine Theorie dar und zeigen, wie man zu einem Rekursionsoperator einer Potenzreihe den so genannten induzierten Rekursionsoperator bestimmt, der die Koeffizienten der Reihe annulliert ([APR00]). Daraufhin wenden wir die Theorie auf den q-Fall an und können nun einen neuen Algorithmus formulieren, der zu einer q-holonomen Rekursionsgleichung einer q-hypergeometrischen Reihe mit nichttrivialem Entwicklungspunkt die entsprechende q-holonome Rekursionsgleichung für die Koeffizienten ermittelt. Ferner können wir einen neuen Algorithmus angeben, der umgekehrt zu einer q-holonomen Rekursionsgleichung für die Koeffizienten eine q-holonome Rekursionsgleichung der Reihe bestimmt. Mit diesem Algorithmus ist es uns möglich, ein weiteres Verfahren zu entwickeln, welches eine q-holonome Rekursionsgleichung für eine verallgemeinerte q-hypergeometrische Funktion aufstellt, die einer q-hypergeometrischen Potenzreihe entspricht. Mit Formulierung des q-Taylorsatzes haben wir alle Zutaten zusammen, um schließlich das Hauptergebnis dieser Arbeit, das q-Analogon des FPS-Algorithmus zu erhalten. Der FPS-Algorithmus aus [Koe92] bzw. [Koe06] bestimmt zu einer gegebenen holonomen Funktion die entsprechende hypergeometrische Reihe. Wir erweitern den Algorithmus dahingehend, dass sogar Linearkombinationen q-hypergeometrischer Potenzreihen bestimmt werden können. Im klassischen Fall stellt das Buch von [Han75] eine fast enzyklopädische Liste hypergeometrischer Potenzreihenentwicklungen dar. Während der FPS-Algorithmus in der Lage ist, [Han75] praktisch vollständig zu reproduzieren, indem die darstellenden Funktionen eingegeben werden, ist der q-FPS-Algorithmus durch Eingabe von q-Funktionen in der Lage, eine analoge Enzyklopädie für q-hypergeometrische Potenzreihen zu erzeugen. Im Abschnitt Anwendungen werden einige Beispiele zur Herleitung bekannter Identitäten gezeigt.

Integraler Bestandteil der Dissertation ist das Maple-Package `qFPS`*, welches im sechsten Kapitel vorgestellt wird und alle Algorithmen der Dissertation beinhaltet. Es ist das erste Package eines Computeralgebrasystems, welches die Möglichkeit besitzt, q-Funktionen zu q-differenzieren bzw. zu q-shiften, zu q-Funktionen q-holonome Differential- bzw. Rekursionsgleichungen aufzustellen, q-Rekursionsgleichungen gemäß Kapitel 5 zu konvertieren und zu q-Funktionen die zugehörige q-hypergeometrische Potenzreihe zu ermitteln. Die wichtigsten Prozeduren von* `qFPS` *werden beschrieben und es werden zahlreiche Beispiele zur Verwendung der Prozeduren präsentiert.*

Herzlich danken möchte ich Wolfram Koepf für die Betreuung der Dissertation und Peter Horn für die Hilfe bei LaTeX-Problemen.

1.1 Grundlagen und Notationen

In diesem Abschnitt werden wir Voraussetzungen, die im Folgenden gelten sollen und die überall in der Dissertation verwendet werden, festlegen.

1.1.1 Generalvoraussetzungen

Im Folgenden sei q eine Unbestimmte. Man kann sich unter q eine reelle Zahl mit $0 < q < 1$ vorstellen, aber in dieser Arbeit werden wir für q keine Zahlen einsetzen, sondern alle auftretenden Ausdrücke formal aus der algebraischen Sichtweise betrachten. Auch alle vorkommenden Funktionen werden wir lediglich als Funktionsterme behandeln. Potenzreihen sehen wir als formale Potenzreihen an und vernachlässigen dementsprechend das Konvergenzverhalten. Sei $\mathbb{K}$ ein beliebiger Körper der Charakteristik 0. Für alle $\mathbb{K}$ soll $\mathbb{K}^* = \mathbb{K} \setminus \{0\}$ sein. Mit $\mathbb{F}$ bezeichnen wir den Körper der rationalen Funktionen in q über $\mathbb{K}$, also $\mathbb{F} = \mathbb{K}(q)$. Dieser Körper ist unser Grundkörper, über dem wir die meisten Terme betrachten werden. Den algebraischen Abschluss von $\mathbb{F}$ bezeichnen wir mit $\overline{\mathbb{F}}$.

1.1.2 Zahlenbereiche

Wir verwenden folgende übliche Bezeichnungen für Mengen.

die Menge der natürlichen Zahlen	$\mathbb{N} = \{1, 2, 3, \ldots\}$
die Menge der negativen ganzen Zahlen	$-\mathbb{N} = \{-1, -2, -3, \ldots\}$
die Menge der nichtnegativen ganzen Zahlen	$\mathbb{N}_0 = \mathbb{N} \cup \{0\}$
die Menge der nichtpositiven ganzen Zahlen	$-\mathbb{N}_0 = -\mathbb{N} \cup \{0\}$
die Menge der ganzen Zahlen	$\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$
die Menge der rationalen Zahlen	$\mathbb{Q}$
die Menge der reellen Zahlen	$\mathbb{R}$
die Menge der komplexen Zahlen	$\mathbb{C}$

1.2 q-Operatoren

In der Dissertation werden wir im Wesentlichen zwei Operatoren betrachten, die wir an dieser Stelle vorstellen.

1.2.1 Der q-Differentialoperator D_q

Definition 1.1 Sei f eine Funktion in x. Den Operator D_q mit

$$D_q f(x) := \begin{cases} \frac{f(x)-f(qx)}{(1-q)x}, & x \neq 0 \\ \lim\limits_{t\to 0} D_q f(t), & x = 0 \end{cases}$$

nennt man den *q-Differential-* oder auch *Hahnoperator*. $D_q f(x)$ ist die *q-Ableitung* von $f(x)$. Höhere q-Ableitungen sind rekursiv definiert

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

Der Hahnoperator besitzt die folgenden Eigenschaften

Satz 1.2 Seien f und g Funktionen in x. Für den q-Differentialoperator D_q und beliebige Konstanten $a, b \in \mathbb{C}$ gelten die Regeln

$$D_q\left(af(x)+bg(x)\right) = aD_qf(x)+bD_qg(x) \qquad \text{(Linearität)} \tag{1.1}$$

$$D_q\left(f(x)\cdot g(x)\right) = f(qx)D_qg(x)+g(x)D_qf(x) \tag{1.2}$$

$$= f(x)D_qg(x)+g(qx)D_qf(x) \qquad \text{(Produktregeln)} \tag{1.3}$$

$$D_q\left(\frac{f(x)}{g(x)}\right) = \frac{g(x)D_qf(x)-f(x)D_qg(x)}{g(x)g(qx)} \tag{1.4}$$

$$= \frac{g(qx)D_qf(x)-f(qx)D_qg(x)}{g(x)g(qx)}. \qquad \text{(Quotientenregeln)} \tag{1.5}$$

Beweis Der q-Differentialoperator ist ein linearer Operator, denn es gilt

$$D_q\left(af(x)+bg(x)\right) = \frac{af(x)+bg(x)-\left(af(qx)+bg(qx)\right)}{(1-q)x} = \frac{a(f(x)-f(qx))+b(g(x)-g(qx))}{(1-q)x}$$
$$= a\frac{f(x)-f(qx)}{(1-q)x}+b\frac{g(x)-g(qx)}{(1-q)x} = aD_qf(x)+bD_qg(x).$$

Wir beweisen nun exemplarisch die erste Produktregel (1.2). Der Beweis der zweiten Regel verläuft analog.

$$D_q\left(f(x)\cdot g(x)\right) = \frac{f(x)g(x)-f(qx)g(qx)}{(1-q)x} = \frac{f(x)g(x)-f(qx)g(x)+f(qx)g(x)-f(qx)g(qx)}{(1-q)x}$$
$$= \frac{g(x)(f(x)-f(qx))+f(qx)(g(x)-g(qx))}{(1-q)x}$$
$$= g(x)\frac{f(x)-f(qx)}{(1-q)x}+f(qx)\frac{g(x)-g(qx)}{(1-q)x} = f(qx)D_qg(x)+g(x)D_qf(x)$$

Die erste Quotientenregel (1.4) erhalten wir durch q-Differentiation von $g(x)\frac{f(x)}{g(x)} = f(x)$ unter Verwendung der ersten Produktregel (1.2). Es ergibt sich

$$g(qx)D_q\left(\frac{f(x)}{g(x)}\right)+\frac{f(x)}{g(x)}D_qg(x) = D_qf(x)$$
$$g(qx)D_q\left(\frac{f(x)}{g(x)}\right) = D_qf(x)-\frac{f(x)}{g(x)}D_qg(x)$$
$$D_q\left(\frac{f(x)}{g(x)}\right) = \frac{g(x)D_qf(x)-f(x)D_qg(x)}{g(qx)g(x)}.$$

Anwendung der zweiten Produktregel (1.3) liefert die zweite Quotientenregel (1.5). □

Satz 1.3 Seien f und g Funktionen in x, wobei $g(qx) = q^bg(x)$ mit $b \in \mathbb{C}$ gelten soll. Für den q-Differentialoperator gilt dann die folgende *Kettenregel*

$$D_q\left(f\circ g\right)(x) = \left[D_{q^b}f(x)\right]_{x=g(x)}\cdot D_qg(x). \tag{1.6}$$

Beweis Es gilt mit $g(qx) = q^bg(x)$

$$D_q\left(f\circ g\right)(x) = \frac{(f\circ g)(x)-(f\circ g)(qx)}{(1-q)x}$$
$$= \frac{f(g(x))-f(g(qx))}{g(x)-g(qx)}\,\frac{g(x)-g(qx)}{(1-q)x}$$
$$= \frac{f(g(x))-f(q^bg(x))}{(1-q^b)g(x)}\,\frac{g(x)-g(qx)}{(1-q)x}$$
$$= \left[D_{q^b}f(x)\right]_{x=g(x)}\cdot D_qg(x).$$

□

Bemerkung 1.4 Wir betrachten Funktionen $g(x)$, die der Eigenschaft $g(qx) = q^b g(x)$ mit $b \in \mathbb{C}$ genügen, genauer und erhalten mit den Substitutionen $x \to q^j$ und $g(q^j) \to c_j$ für $j \in \mathbb{Z}$ (siehe auch Korollar 4.36)

$$\frac{g(qx)}{g(x)} = \frac{c_{j+1}}{c_j} = q^b \quad \Longrightarrow \quad g(x) = c_j = a\left(q^b\right)^j = a\left(q^j\right)^b = ax^b \quad \text{mit} \quad a \in \mathbb{C}.$$

Im Folgenden können wir uns unter $g(x)$ also alle Potenzfunktionen der Form ax^b vorstellen.

Betrachten wir nun das Vertauschungsverhalten des q-Differentialoperators bei der Multiplikation mit dem Operator, der einen Funktionsausdruck $f(x)$ mit x multipliziert.

Lemma 1.5 Für den q-Differentialoperator gilt die Kommutatorregel

$$1 + qxD_q = D_q x.$$

Beweis Sei f eine Funktion in x. Wenden wir $D_q x$ auf die Funktion $f(x)$ an, so erhalten wir mit der ersten Produktregel $(D_q x)f(x) = D_q\left(xf(x)\right) = qxD_q f(x) + f(x) = (1 + qxD_q)f(x)$. □

1.2.2 Der q-Shiftoperator ε_q

Definition 1.6 Sei f eine Funktion in x. Den Operator ε_q mit

$$\varepsilon_q f(x) = f(qx)$$

nennt man *q-Shiftoperator*. $\varepsilon_q f(x)$ ist der *q-Shift* von $f(x)$. Höhere q-Shifts sind rekursiv definiert

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

Der q-Shiftoperator besitzt die folgenden Eigenschaften

Satz 1.7 Seien f und g Funktionen in x. Für den q-Shiftoperator ε_q und beliebige Konstanten $a, b \in \mathbb{C}$ gelten die Regeln

$$\varepsilon_q\left(af(x) + bg(x)\right) = a\varepsilon_q f(x) + b\varepsilon_q g(x) \qquad \text{(Linearität)} \tag{1.7}$$

$$\varepsilon_q\left(f(x) \cdot g(x)\right) = \varepsilon_q f(x) \cdot \varepsilon_q g(x) \qquad \text{(Produktregel)} \tag{1.8}$$

$$\varepsilon_q\left(\frac{f(x)}{g(x)}\right) = \frac{\varepsilon_q f(x)}{\varepsilon_q g(x)} \qquad \text{(Quotientenregel)} \tag{1.9}$$

Beweis Der q-Shiftoperator ist ein linearer Operator, denn offenbar gilt

$$\varepsilon_q\left(af(x) + bg(x)\right) = af(qx) + bg(qx) = a\varepsilon_q f(x) + b\varepsilon_q g(x).$$

Die anderen beiden Formeln ergeben sich durch

$$\varepsilon_q\left(f(x) \cdot g(x)\right) = f(qx) \cdot g(qx) = \varepsilon_q f(x) \cdot \varepsilon_q g(x)$$

und

$$\varepsilon_q\left(\frac{f(x)}{g(x)}\right) = \frac{f(qx)}{g(qx)} = \frac{\varepsilon_q f(x)}{\varepsilon_q g(x)}.$$

□

Satz 1.8 Seien f und g Funktionen in x, wobei $g(qx) = q^b g(x)$ mit $b \in \mathbb{C}$ gelten soll. Für den q-Shiftoperator gilt dann folgende Regel für die verkettete Funktion $f \circ g$

$$\varepsilon_q\left(f \circ g\right)(x) = \left[\varepsilon_{q^b} f(x)\right]_{x=g(x)}. \tag{1.10}$$

Beweis Es gilt mit $g(qx) = q^b g(x)$

$$\varepsilon_q\left(f\circ g\right)(x) = \left(f\circ g\right)(qx) = f(g(qx)) = f(q^b g(x)) = \left[\varepsilon_{q^b} f(x)\right]_{x=g(x)}.$$

□

Lemma 1.9 Für den q-Shiftoperator gilt die Kommutatorregel

$$qx\varepsilon_q = \varepsilon_q x.$$

Beweis Sei f eine Funktion in x. Wenden wir $\varepsilon_q x$ auf die Funktion $f(x)$ an, so erhalten wir $(\varepsilon_q x)f(x) = \varepsilon_q\left(xf(x)\right) = qxf(qx) = qx\varepsilon_q f(x) = (qx\varepsilon_q)f(x)$. □

Satz 1.10 Für den q-Shiftoperator und den q-Differentialoperator gilt die Kommutatorregel

$$q\varepsilon_q D_q = D_q\varepsilon_q.$$

Beweis Sei f eine Funktion in x. Dann gilt

$$(q\varepsilon_q D_q)f(x) = q\varepsilon_q\frac{f(x)-f(qx)}{(1-q)x} = q\frac{f(qx)-f(q^2x)}{(1-q)qx} = \frac{f(qx)-f(q^2x)}{(1-q)x} = D_q f(qx) = (D_q\varepsilon_q)f(x).$$

□

Den nachstehenden Satz ([Ryd21], [Koo99]) können wir dazu verwenden, um höhere q-Ableitungen auf q-Shifts zurückzuführen und umgekehrt. Somit können wir jede Problemstellung, in der q-Ableitungen auftreten, auf q-Shifts übertragen. Aus der Sicht der Computeralgebra sind q-Shifts wesentlich einfacher und effizienter zu bestimmen und besitzen gegenüber dem q-Differentialoperator den Vorteil, dass sie die Eigenschaften (1.8) und (1.9) erfüllen. Alle in dieser Dissertation behandelten Algorithmen verwenden daher den q-Shiftoperator.

Satz 1.11 Sei $n \in \mathbb{N}_0$. Es existieren folgende Zusammenhänge zwischen höheren q-Ableitungen und höheren q-Shifts[1]

$$\varepsilon_q^n f(x) = \sum_{k=0}^{n}(-1)^k(1-q)^k\begin{bmatrix}n\\k\end{bmatrix}_q q^{\binom{k}{2}}x^k D_q^k f(x) \tag{1.11}$$

$$D_q^n f(x) = \frac{1}{(1-q)^n x^n}\sum_{k=0}^{n}(-1)^k\begin{bmatrix}n\\k\end{bmatrix}_q q^{\binom{k}{2}-(n-1)k}\varepsilon_q^k f(x). \tag{1.12}$$

Beweis Zum Beweis wenden wir bei beiden Gleichungen vollständige Induktion an. Betrachten wir zunächst die Gleichung (1.11). Wir erhalten jeweils $f(x)$ für $n = 0$ und die Gleichung $\varepsilon_q f(x) = f(x) - (1-q)xD_q f(x)$ für $n = 1$, welche nach Definition 1.1 richtig ist. Nehmen wir an Formel (1.11) gilt für ein $n \in \mathbb{N}_0$, dann gilt sie auch für $n+1$, denn

$$\begin{aligned}\varepsilon_q^{n+1} f(x) = \varepsilon_q\varepsilon_q^n f(x) &= \varepsilon_q\sum_{k=0}^{n}(-1)^k(1-q)^k\begin{bmatrix}n\\k\end{bmatrix}_q q^{\binom{k}{2}}x^k D_q^k f(x)\\ &= \sum_{k=0}^{n}(-1)^k(1-q)^k\begin{bmatrix}n\\k\end{bmatrix}_q q^{\binom{k}{2}}q^k x^k\varepsilon_q D_q^k f(x)\end{aligned}$$

[1] siehe Definition 1.19 für die auftretenden q-Binomialkoeffizienten $\left[\begin{smallmatrix}n\\k\end{smallmatrix}\right]_q$

Wegen $\varepsilon_q g(x) = g(x) - (1-q)xD_q g(x)$ folgt mit $g(x) = D_q^k f(x)$

$$
\begin{aligned}
&= \sum_{k=0}^{n} (-1)^k (1-q)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}} q^k x^k \left(D_q^k f(x) - (1-q)x D_q^{k+1} f(x) \right) \\
&= \sum_{k=0}^{n} (-1)^k (1-q)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}} q^k x^k D_q^k f(x) + \sum_{k=0}^{n} (-1)^{k+1} (1-q)^{k+1} \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k+1}{2}} x^{k+1} D_q^{k+1} f(x) \\
&= \sum_{k=0}^{n} (-1)^k (1-q)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}} q^k x^k D_q^k f(x) + \sum_{k=1}^{n+1} (-1)^k (1-q)^k \begin{bmatrix} n \\ k-1 \end{bmatrix}_q q^{\binom{k}{2}} x^k D_q^k f(x)
\end{aligned}
$$

Da sowohl $\begin{bmatrix} n \\ n+1 \end{bmatrix}_q = 0$, als auch $\begin{bmatrix} n \\ -1 \end{bmatrix}_q = 0$ für $n \in \mathbb{N}_0$ gilt, können wir bei beiden Summen den Summationsbereich erweitern und erhalten

$$
\begin{aligned}
&= \sum_{k=0}^{n+1} (-1)^k (1-q)^k \begin{bmatrix} n \\ k \end{bmatrix}_q q^{\binom{k}{2}} q^k x^k D_q^k f(x) + \sum_{k=0}^{n+1} (-1)^k (1-q)^k \begin{bmatrix} n \\ k-1 \end{bmatrix}_q q^{\binom{k}{2}} x^k D_q^k f(x) \\
&= \sum_{k=0}^{n+1} (-1)^k (1-q)^k \left(q^k \begin{bmatrix} n \\ k \end{bmatrix}_q + \begin{bmatrix} n \\ k-1 \end{bmatrix}_q \right) q^{\binom{k}{2}} x^k D_q^k f(x)
\end{aligned}
$$

Durch Anwenden der ersten q-Pascalgleichung (1.14) bekommt man schließlich

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

Wenden wir uns nun der zweiten Gleichung zu. Auch hier erhält man wieder jeweils $f(x)$ für $n = 0$ und die Gleichung $D_q f(x) = \frac{f(x) - \varepsilon_q f(x)}{(1-q)x}$ für $n = 1$. Gilt die Gleichung (1.12) für ein $n \in \mathbb{N}_0$, so folgt für $n+1$

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

Wegen $q^{\binom{k}{2} - nk} q^{n-k+1} = q^{\binom{k-1}{2} - n(k-1)}$ folgt

$$
= \frac{1}{(1-q)^{n+1} x^{n+1}} \sum_{k=0}^{n+1} (-1)^k \left(\begin{bmatrix} n \\ k \end{bmatrix}_q + q^{n-k+1} \begin{bmatrix} n \\ k-1 \end{bmatrix}_q \right) q^{\binom{k}{2} - nk} \varepsilon_q^k f(x)
$$

Anwendung der zweiten q-Pascalgleichung (1.14) liefert

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

und damit die Behauptung. □

1.3 q-Funktionen

In der klassischen Analysis betrachtet man Gitter, bei denen die **Differenz** zweier benachbarter Punkte konstant ist. Bezeichne h den Abstand dieser Punkte. Man definiert auf diesem Gitter

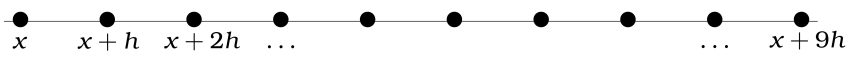

Abbildung 1.1 Äquidistantes Gitter mit $h > 0$

einen Differenzenquotienten $\frac{f(x+h)-f(x)}{x+h-x} = \frac{f(x+h)-f(x)}{h}$ und erhält beim Grenzübergang $h \to 0$ die Ableitung der Funktion $f(x)$, sofern $f(x)$ differenzierbar ist. Wir betrachten nun Funktionen auf einem *q-Gitter*, bei dem der **Quotient** zweier benachbarter Punkte konstant ist und mit q bezeichnet wird.

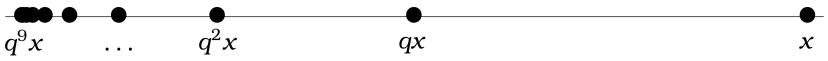

Abbildung 1.2 q-Gitter für $q < 1$

Der q-Differenzenquotient ist nun die q-Ableitung $D_q f(x) = \frac{f(x)-f(qx)}{x-qx} = \frac{f(x)-f(qx)}{(1-q)x}$. Funktionen, die auf einem q-Gitter „leben“, bezeichnen wir als q-Funktionen. In diesem Abschnitt werden wir nun einige q-Funktionen kennen lernen. Zu Funktionen, die man aus der Analysis kennt, werden entsprechende q-Verallgemeinerungen präsentiert. Bei zahlreichen q-Funktionen, wie z.B. bei den q-Exponentialfunktionen, gibt es nicht nur ein q-Analogon, sondern mehrere entsprechende q-Varianten, denen gemein ist, dass sich beim Grenzübergang $\lim_{q\uparrow 1}$ die bekannte klassische Funktion ergibt.

1.3.1 Elementare q-Funktionen

Wir führen zunächst einige elementare q-Funktionen ein.

Definition 1.12 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.

Definition 1.13 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 1.14 Die *q-Potenz* $(b \ominus a)_q^k$ ist für $a, b \in \mathbb{C}$ und $k \in \mathbb{N}_0$ definiert als

$$(b \ominus a)_q^k := \begin{cases} (b-a) \cdot (b-aq) \cdot \ldots \cdot (b-aq^{k-1}), & k \in \mathbb{N} \\ 1, & k = 0 \end{cases}$$

und lässt sich erweitern zu $(b \ominus a)_q^\infty := \prod_{i=0}^{\infty}(b-aq^i)$.

Definition 1.15 Das *q-Pochhammersymbol* $(a;q)_k$ ist für $a \in \mathbb{C}$ und $k \in \mathbb{Z}$ definiert als

$$(a;q)_k := \begin{cases} (1-a) \cdot (1-aq) \cdot \ldots \cdot (1-aq^{k-1}), & k \in \mathbb{N} \\ 1, & k = 0 \\ \frac{1}{(1-aq^{-1}) \cdot (1-aq^{-2}) \cdot \ldots \cdot (1-aq^k)}, & k \in -\mathbb{N}, \end{cases}$$

wobei im letzten Fall die Einschränkung $a \notin \{q, q^2, \ldots, q^{-k}\}$ gilt. Das q-Pochhammersymbol $(a;q)_\infty$ ist gegeben durch $(a;q)_\infty := \prod_{i=0}^{\infty}(1-aq^i)$. Mit Hilfe dieser Definition lässt sich das q-Pochhammersymbol fortsetzen zu

$$(a;q)_b := \frac{(a;q)_\infty}{(aq^b;q)_\infty}$$

mit $b \in \mathbb{C}$.

Es gilt offenbar

$$(a;q)_k = (1 \ominus a)_q^k \quad \text{und} \quad (b \ominus a)_q^k = b^k \left(\frac{a}{b};q\right)_k, \tag{1.13}$$

d.h. die q-Potenz kann man als Verallgemeinerung des q-Pochhammersymbols ansehen.

Definition 1.16 Die *q-Gammafunktion* $\Gamma_q(a)$ ist definiert als

$$\Gamma_q(a) := \frac{(q;q)_\infty}{(q^a;q)_\infty}(1-q)^{1-a}$$

für alle $a \in \mathbb{C} \setminus \{0, -1, -2, \ldots\}$.

Satz 1.17 Für die q-Gammafunktion gilt die *q-Fundamentalgleichung*

$$\Gamma_q(a+1) = [a]_q \Gamma_q(a).$$

Beweis Nach Definition gilt

$$\Gamma_q(a+1) = \frac{(q;q)_\infty}{(q^{a+1};q)_\infty}(1-q)^{-a} = \frac{(1-q^a)(q;q)_\infty}{(1-q)(q^a;q)_\infty}(1-q)^{-a+1} = [a]_q \Gamma_q(a).$$

□

Aus der q-Fundamentalgleichung folgt mit $\Gamma_q(1) = 1 = [0]_q!$ und vollständiger Induktion

Korollar 1.18 Die q-Gammafunktion interpoliert die q-Fakultät in den natürlichen Zahlen, d.h. es gilt

$$\Gamma_q(n+1) = [n]_q!$$

für $n \in \mathbb{N}_0$.

Definition 1.19 Seien $a, b \in \mathbb{C}$. Der *q-Binomialkoeffizient* ist definiert als

$$\begin{bmatrix} a \\ b \end{bmatrix}_q := \frac{\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+1)},$$

unter der Voraussetzung, dass $a \notin -\mathbb{N}$ ist. Für $a, b \in \mathbb{N}_0$ und $a \geq b$ ist

$$\begin{bmatrix} a \\ b \end{bmatrix}_q = \frac{[a]_q!}{[b]_q![a-b]_q!} = \frac{(q;q)_a}{(q;q)_b(q;q)_{a-b}}.$$

Die soeben definierten q-Fakultäten, q-Potenzen, q-Pochhammersymbole, q-Gammafunktionen und q-Binomialkoeffizienten können als Verallgemeinerung der gewöhnlichen Fakultäten, Potenzen, Pochhammersymbole, Gammafunktionen und Binomialkoeffizienten aufgefasst werden. Durch den Grenzübergang $\lim_{q\uparrow 1}$ bekommt man die bekannten Ausdrücke. Wir betrachten die Grenzübergänge für $k, n \in \mathbb{N}_0$ und $a, b \in \mathbb{C}$ im Einzelnen.

$$\begin{aligned}
\lim_{q\uparrow 1}[a]_q &= \lim_{q\uparrow 1}\frac{1-q^a}{1-q} = \lim_{q\uparrow 1} aq^{a-1} = a\\
\lim_{q\uparrow 1}[n]_q! &= \lim_{q\uparrow 1}[n]_q\cdot[n-1]_q\cdot\ldots\cdot[1]_q = n\cdot(n-1)\cdot\ldots\cdot 1 = n!\\
\lim_{q\uparrow 1}(b\ominus a)_q^k &= \lim_{q\uparrow 1}(b-a)\cdot(b-aq)\cdot\ldots\cdot(b-aq^{k-1}) = (b-a)^k\\
\lim_{q\uparrow 1}\frac{(q^a;q)_k}{(1-q)^k} &= \lim_{q\uparrow 1}\frac{(1-q^a)\cdot(1-q^{a+1})\cdot\ldots\cdot(1-q^{a+k-1})}{(1-q)\cdot(1-q)\cdot\ldots\cdot(1-q)}\\
&= \lim_{q\uparrow 1}[a]_q\cdot[a+1]_q\cdot\ldots\cdot[a+k-1]_q = a\cdot(a+1)\cdot\ldots\cdot(a+k-1) = (a)_k\\
\lim_{q\uparrow 1}\begin{bmatrix}a\\b\end{bmatrix}_q &= \lim_{q\uparrow 1}\frac{\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+1)} = \frac{\Gamma(a+1)}{\Gamma(b+1)\Gamma(a-b+1)} = \binom{a}{b}
\end{aligned}$$

Bei dem q-Pochhammersymbol betrachtet man nicht den Ausdruck $(a;q)_k$, sondern den Ausdruck $\frac{(q^a;q)_k}{(1-q)^k}$, um den Zusammenhang zum Pochhammersymbol herzustellen. Der Beweis der Gleichung $\lim_{q\uparrow 1}\Gamma_q(a) = \Gamma(a)$ ist nicht trivial. Man findet den Beweis und weitere Informationen zur Funktion $\Gamma_q(a)$ in [AAR01].

Satz 1.20 Die q-Binomialkoeffizienten erfüllen die *q-Pascalgleichungen*

$$\begin{bmatrix}a+1\\b\end{bmatrix}_q = q^b\begin{bmatrix}a\\b\end{bmatrix}_q + \begin{bmatrix}a\\b-1\end{bmatrix}_q \quad\text{und}\quad \begin{bmatrix}a+1\\b\end{bmatrix}_q = \begin{bmatrix}a\\b\end{bmatrix}_q + q^{a-b+1}\begin{bmatrix}a\\b-1\end{bmatrix}_q. \tag{1.14}$$

Beweis Unter mehrmaliger Verwendung der q-Fundamentalgleichung erhalten wir

$$\begin{aligned}
q^b\begin{bmatrix}a\\b\end{bmatrix}_q + \begin{bmatrix}a\\b-1\end{bmatrix}_q &= q^b\frac{\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+1)} + \frac{\Gamma_q(a+1)}{\Gamma_q(b)\Gamma_q(a-b+2)}\\
&= \frac{q^b[a-b+1]_q\Gamma_q(a+1) + [b]_q\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \frac{\left(q^b[a-b+1]_q + [b]_q\right)\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \frac{\left(q^b\left(1-q^{a-b+1}\right)+1-q^b\right)(1-q)^{-1}\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \frac{\left(q^b-q^{a+1}+1-q^b\right)(1-q)^{-1}\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \frac{(1-q^{a+1})(1-q)^{-1}\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \frac{[a+1]_q\Gamma_q(a+1)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \frac{\Gamma_q(a+2)}{\Gamma_q(b+1)\Gamma_q(a-b+2)}\\
&= \begin{bmatrix}a+1\\b\end{bmatrix}_q
\end{aligned}$$

und analog dazu die zweite Gleichung. □

Im q-Fall ist es also möglich, den q-Binomialkoeffizienten wiederum durch **einen** q-Binomialkoeffizienten darzustellen, indem man z.B. obige q-Pascalgleichungen gleichsetzt und nach einem q-Binomialkoeffizienten auflöst.

1.3.2 Die verallgemeinerte q-hypergeometrische Funktion

Definition 1.21 Die *verallgemeinerte q-hypergeometrische Funktion* ist nach [GR90] definiert als

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

Wir schreiben auch kurz

$$ {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \middle| q; x \right) := \sum_{k=0}^{\infty} \frac{(\boldsymbol{a};q)_k}{(\boldsymbol{b};q)_k} \frac{x^k}{(q;q)_k} \left((-1)^k q^{\binom{k}{2}} \right)^{1+s-r} $$

mit $\boldsymbol{a} = a_1, \ldots, a_r$ bzw. $\boldsymbol{b} = b_1, \ldots, b_s$ und $(\boldsymbol{a};q)_k = (a_1;q)_k \cdot \ldots \cdot (a_r;q)_k$ bzw. $(\boldsymbol{b};q)_k = (b_1;q)_k \cdot \ldots \cdot (b_s;q)_k$ oder auch ${}_r\phi_s$ bzw. ${}_r\phi_s(x)$, wenn die Parameter keine Rolle spielen.

Die verallgemeinerte q-hypergeometrische Funktion ${}_r\phi_s$ ist das q-Analogon der hypergeometrischen Funktion ${}_pF_q$. Die beiden Funktionen hängen wie folgt zusammen

$$ \lim_{q \uparrow 1} {}_r\phi_s\left(\begin{matrix} q^{a_1} \ldots q^{a_r} \\ q^{b_1} \ldots q^{b_s} \end{matrix} \middle| q; (q-1)^{1+s-r} x \right) = {}_rF_s\left(\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} \middle| x \right). $$

Die q-analoge Definition eines hypergeometrischen Terms folgt mit

Definition 1.22 Einen Term $f(x)$ nennt man *q-hypergeometrisch*, wenn

$$ \frac{\varepsilon_q f(x)}{f(x)} = \frac{f(qx)}{f(x)} = r(x) \in \mathbb{F}(x) $$

gilt, wobei $\mathbb{F} = \mathbb{K}(q)$ ist. Die rationale Funktion $r(x)$ nennt man *q-Zertifikat* von $f(x)$.

Alle elementaren q-Funktionen sind, wie wir später sehen werden, q-hypergeometrische Terme. Für Reihen definieren wir

Definition 1.23 Eine Potenzreihe $\sum_{j=0}^{\infty} c_j x^j$ nennt man *q-hypergeometrisch*, wenn der Koeffizient c_j q-hypergeometrisch ist, d.h. wenn

$$ \frac{c_{j+1}}{c_j} \in \mathbb{F}(q^j) $$

gilt.

1.3.3 q-Exponentialfunktionen

Wir führen nun einige q-Varianten der Exponentialfunktion ein.

Definition 1.24 Die *kleine q-Exponentialfunktion nach Gasper/Rahman* ([GR90]) ist gegeben durch

$$ \exp_q(x) := {}_1\phi_0\left(\begin{matrix} 0 \\ - \end{matrix} \middle| q; x \right) = \sum_{k=0}^{\infty} \frac{1}{(q;q)_k} x^k, $$

während die *kleine q-Exponentialfunktion nach Kac/Cheung* ([KC02]) durch

$$ \mathrm{e}_q(x) := {}_1\phi_0\left(\begin{matrix} 0 \\ - \end{matrix} \middle| q; (1-q)x \right) = \sum_{k=0}^{\infty} \frac{1}{[k]_q!} x^k $$

definiert ist.

Definition 1.25 Die *große q-Exponentialfunktion nach Gasper/Rahman* ([GR90]) ist gegeben durch

$$ \mathrm{Exp}_q(x) := {}_0\phi_0\left(\begin{matrix} - \\ - \end{matrix} \middle| q; -x \right) = \sum_{k=0}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_k} x^k, $$

während die *große q-Exponentialfunktion nach Kac/Cheung* ([KC02]) durch

$$ \mathrm{E}_q(x) := {}_0\phi_0\left(\begin{matrix} - \\ - \end{matrix} \middle| q; -(1-q)x \right) = \sum_{k=0}^{\infty} \frac{q^{\binom{k}{2}}}{[k]_q!} x^k $$

definiert ist.

1.3.4 q-Trigononometrische Funktionen

Die Definitionen der üblichen trigonometrischen Funktionen lassen sich auf die q-Analysis erweitern. Die folgenden Definitionen sind in [GR90] und [KC02] zu finden.

Definition 1.26 Die *kleine q-Sinusfunktion nach Gasper/Rahman* ist gegeben durch

$$\sin_q(x) := \frac{\exp_q(ix) - \exp_q(-ix)}{2i} = \sum_{k=0}^{\infty} \frac{(-1)^k}{(q;q)_{2k+1}} x^{2k+1},$$

während die *kleine q-Sinusfunktion nach Kac/Cheung* durch

$$\mathrm{s}_q(x) := \frac{\mathrm{e}_q(ix) - \mathrm{e}_q(-ix)}{2i} = \sum_{k=0}^{\infty} \frac{(-1)^k}{[2k+1]_q!} x^{2k+1}$$

definiert ist.

Definition 1.27 Die *große q-Sinusfunktion nach Gasper/Rahman* ist gegeben durch

$$\mathrm{Sin}_q(x) := \frac{\mathrm{Exp}_q(ix) - \mathrm{Exp}_q(-ix)}{2i} = \sum_{k=0}^{\infty} \frac{(-1)^k q^{k(2k+1)}}{(q;q)_{2k+1}} x^{2k+1},$$

während die *große q-Sinusfunktion nach Kac/Cheung* durch

$$\mathrm{S}_q(x) := \frac{\mathrm{E}_q(ix) - \mathrm{E}_q(-ix)}{2i} = \sum_{k=0}^{\infty} \frac{(-1)^k q^{k(2k+1)}}{[2k+1]_q!} x^{2k+1}$$

definiert ist.

Definition 1.28 Die *kleine q-Cosinusfunktion nach Gasper/Rahman* ist gegeben durch

$$\cos_q(x) := \frac{\exp_q(ix) + \exp_q(-ix)}{2} = \sum_{k=0}^{\infty} \frac{(-1)^k}{(q;q)_{2k}} x^{2k},$$

während die *kleine q-Cosinusfunktion nach Kac/Cheung* durch

$$\mathrm{c}_q(x) := \frac{\mathrm{e}_q(ix) + \mathrm{e}_q(-ix)}{2} = \sum_{k=0}^{\infty} \frac{(-1)^k}{[2k]_q!} x^{2k}$$

definiert ist.

Definition 1.29 Die *große q-Cosinusfunktion nach Gasper/Rahman* ist gegeben durch

$$\mathrm{Cos}_q(x) := \frac{\mathrm{Exp}_q(ix) + \mathrm{Exp}_q(-ix)}{2} = \sum_{k=0}^{\infty} \frac{(-1)^k q^{k(2k-1)}}{(q;q)_{2k}} x^{2k},$$

während die *große q-Cosinusfunktion nach Kac/Cheung* durch

$$\mathrm{C}_q(x) := \frac{\mathrm{E}_q(ix) + \mathrm{E}_q(-ix)}{2} = \sum_{k=0}^{\infty} \frac{(-1)^k q^{k(2k-1)}}{[2k]_q!} x^{2k}$$

definiert ist.

Wir betrachten auch hier den Grenzübergang $\lim_{q\uparrow 1}$. Es ergibt sich, wie man leicht überprüfen kann, für die kleinen und großen Gasper/Rahman-Funktionen $f_q(x)$

$$\lim_{q\uparrow 1} f_q((1-q)x) = f(x)$$

und für die kleinen und großen Kac/Cheung-Funktionen $\tilde{f}_q(x)$

$$\lim_{q\uparrow 1} \tilde{f}_q(x) = f(x),$$

wobei $f(x)$ jeweils die klassische Exponential-, Sinus- oder Cosinusfunktion bezeichnet.

1.3.5 Klassische q-orthogonale Polynome

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

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

der Gleichung

$$\sigma(x) D_q D_{q^{-1}} P_n(x) + \tau(x) D_q P_n(x) + \lambda_n P_n(x) = 0 \tag{1.15}$$

mit

$$\sigma(x) = ax^2+bx+c \quad \text{und} \quad \tau(x) = dx+e \quad \text{mit} \quad a,b,c,d,e \in \mathbb{F},\ \lambda_n \in \mathbb{F}(q^n), \quad abc \neq 0 \quad \text{und} \quad d \neq 0.$$

Die folgenden Definitionen der klassischen q-orthogonalen Polynome über die Darstellung als q-hypergeometrische Funktion sind in [KS98] zu finden.

Definition 1.30 Die klassischen q-orthogonalen Polynome der Hahnklasse sind

(a) die *großen q-Jacobipolynome* mit

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

Der Spezialfall $a = b = 1$ führt auf die *großen q-Legendrepolynome*

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

(b) die *q-Hahnpolynome* mit

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

(c) die *großen q-Laguerrepolynome* mit

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

(d) die *kleinen q-Jacobipolynome* mit

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

Der Spezialfall $a = b = 1$ führt auf die *kleinen q-Legendrepolynome*

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

(e) die *q-Meixnerpolynome* mit

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

(f) die *Quantum-q-Krawtchoukpolynome* mit

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

(g) die *q-Krawtchoukpolynome* mit

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

(h) die *affinen q-Krawtchoukpolynome* mit

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

(i) die *kleinen q-Laguerrepolynome* mit

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

(j) die *q-Laguerrepolynome* mit

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

(k) die *alternativen q-Charlierpolynome* mit

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

(l) die *q-Charlierpolynome* mit

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

(m) die *Al-Salam-Carlitz I-Polynome* mit

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

(n) die *Al-Salam-Carlitz II-Polynome* mit

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

(o) die *Stieltjes-Wigert-Polynome* mit

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

(p) die *diskreten q-Hermite I-Polynome* mit

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

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

(q) die *diskreten q-Hermite II-Polynome* mit

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

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

Es ist wohlbekannt ([KS98],[KS01]), dass alle klassischen q-orthogonalen Polynome $P_n(x)$ eine *Drei-Term-Rekursion* der Form

$$P_{n+1}(x) = (A_n x + B_n) P_n(x) - C_n P_{n-1}(x) \quad \text{mit} \quad A_n, B_n, C_n \in \mathbb{F}(q^n) \tag{1.16}$$

erfüllen.

Wir geben nun die q-Differentialgleichungen und den höchsten Koeffizienten der klassischen q-orthogonalen Polynome tabellarisch an. Die Tabelle 1.1 wurde wie folgt aus den ausmultiplizierten Darstellungen der jeweiligen q-Differentialgleichung in [KS98] extrahiert. Die q-Differentialgleichung (1.15) ist darstellbar als

$$B(x)\varepsilon_q P_n(x) - (B(x) + D(x))\, P_n(x) + D(x)\varepsilon_{q^{-1}} P_n(x) = -(q-1)^2 x^2 \lambda_n P_n(x) \tag{1.17}$$

mit $B(x) = \sigma(x) + (q-1)x\,\tau(x)$ und $D(x) = q\sigma(x)$. Umgekehrt erhält man aus Gleichung (1.17) die q-Differentialgleichung (1.15) mit $\sigma(x) = \frac{D(x)}{q}$ und $\tau(x) = \frac{qB(x)-D(x)}{q(q-1)x}$. Folglich kann man jeweils $\sigma(x)$, $\tau(x)$ und λ_n aus den Darstellungen (1.17) in [KS98] gewinnen. Den höchsten Koeffizienten k_n erhält man aus der q-hypergeometrischen Darstellung des q-orthogonalen Polynoms. Die Parameter aus Tabelle 1.1 benötigen wir im nächsten Kapitel.

q-orthogonales Polynom		a	b	c	d	e	λ_n	k_n
$\tilde{\mathrm{h}}_n(x;q)$	diskrete q-Hermite II	0	0	$q-1$	1	0	$-\frac{q^n-1}{q-1}$	1
$\mathrm{V}_n^{(a)}(x;q)$	Al-Salam-Carlitz II	0	0	$(q-1)a$	1	$-a-1$	$-\frac{q^n-1}{q-1}$	1
$\mathrm{C}_n(x;a;q)$	q-Charlier	0	$(q-1)a$	0	q	$-a-q$	$-\frac{q(q^n-1)}{q-1}$	$(-1)^n \frac{q^{n^2}}{a^n}$
$\mathrm{L}_n^{(\alpha)}(x;q)$	q-Laguerre	0	$q-1$	0	qq^α	$qq^\alpha-1$	$-\frac{qq^\alpha(q^n-1)}{q-1}$	$(-1)^n \frac{q^{n(n+\alpha)}}{(q;q)_n}$
$\mathrm{S}_n(x;q)$	Stieltjes-Wigert	0	$q-1$	0	q	-1	$-\frac{q(q^n-1)}{q-1}$	$(-1)^n \frac{q^{n^2}}{(q;q)_n}$
$\mathrm{M}_n(x;b,c;q)$	q-Meixner	0	$(q-1)c$	$q(1-q)bc$	q	$bcq-q-c$	$-\frac{q(q^n-1)}{q-1}$	$(-1)^n \frac{q^{n^2}}{c^n(bq;q)_n}$
$\mathrm{K}_n^{\mathrm{qtm}}(x;p,N;q)$	Quantum-q-Krawtchouk	0	$(q-1)q^N$	$(1-q)$	$-pqq^N$	$1-(1-pq)q^N$	$\frac{pqq^N(q^n-1)}{q-1}$	$\frac{p^n q^{n^2}}{\left(q^{-N};q\right)_n}$
$\mathrm{K}_n(x;p,N;q)$	q-Krawtchouk	$(1-q)q^N$	$(q-1)$	0	$(1+pq)q^N$	$-(1+pqq^N)$	$-\frac{qq^N(1+pq^n)(q^n-1))}{q^n(q-1)}$	$\frac{(-pq^n;q)_n}{\left(q^{-N};q\right)_n}$
$\mathrm{p}_n(x;a\|q)$	kleine q-Laguerre	$1-q$	$q-1$	0	1	$aq-1$	$-\frac{q^n-1}{q^{n-1}(q-1)}$	$(-1)^n \frac{q^{-\binom{n}{2}}}{(aq;q)_n}$
$\mathrm{K}_n(x;a;q)$	alternative q-Charlier	$1-q$	$q-1$	0	$1+aq$	-1	$-\frac{(1+aq^n)(q^n-1)}{q^{n-1}(q-1)}$	$(-1)^n \frac{q^{-\binom{n}{2}}}{(-aq^n;q)_n}$
$\mathrm{p}_n(x;a,b\|q)$	kleine q-Jacobi	$1-q$	$q-1$	0	$1-abq^2$	$aq-1$	$-\frac{(1-abqq^n)(q^n-1)}{q^{n-1}(q-1)}$	$(-1)^n \frac{q^{-\binom{n}{2}}(abqq^n;q)_n}{(aq;q)_n}$
$\mathrm{h}_n(x;q)$	diskrete q-Hermite I	$1-q$	0	$q-1$	1	0	$-\frac{(q^n-1)}{q^{n-1}(q-1)}$	1
$\mathrm{U}_n^{(a)}(x;q)$	Al-Salam-Carlitz I	$1-q$	$(q-1)(1+a)$	$(1-q)a$	1	$-a-1$	$-\frac{(q^n-1)}{q^{n-1}(q-1)}$	1
$\mathrm{P}_n(x;a,b;q)$	große q-Laguerre	$1-q$	$q(q-1)(a+b)$	$q^2(1-q)ab$	1	$q(abq-a-b)$	$-\frac{(q^n-1)}{q^{n-1}(q-1)}$	$\frac{1}{(aq;q)_n(bq;q)_n}$
$\mathrm{K}_n^{\mathrm{aff}}(x;p,N;q)$	affine q-Krawtchouk	$(1-q)q^N$	$(q-1)(1+pqq^N)$	$(1-q)pq$	q^N	$-(1+pq(q^N-1))$	$-\frac{qq^N(q^n-1))}{q^n(q-1)}$	$\frac{1}{(pq;q)_n\left(q^{-N};q\right)_n}$
$\mathrm{Q}_n(x;a,b;N\|q)$	q-Hahn	$(1-q)q^N$	$(q-1)(1+aqq^N)$	$q(1-q)a$	$(1-abq^2)q^N$	$aq(1+q^N(bq-1))-1$	$-\frac{q^N(1-abqq^n)(q^n-1)}{q^{n-1}(q-1)}$	$\frac{(abqq^n;q)_n}{(aq;q)_n\left(q^{-N};q\right)_n}$
$\mathrm{P}_n(x;a,b,c;q)$	große q-Jacobi	$(1-q)$	$q(q-1)(a+c)$	$q^2(1-q)ac$	$1-abq^2$	$q(aq(b+c)-a-c)$	$-\frac{(1-abqq^n)(q^n-1)}{q^{n-1}(q-1)}$	$\frac{(abqq^n;q)_n}{(aq;q)_n(cq;q)_n}$

Tabelle 1.1 q-Differentialgleichung und höchster Koeffizient klassischer q-orthogonaler Polynome

Kapitel 2

q-Ableitungen und q-Shifts von q-Funktionen

Die im vorigen Kapitel vorgestellten q-Funktionen kann man, wie klassische Funktionen auch, auf q-Differenzierbarkeit untersuchen. Da die q-Ableitung nichts anderes als der q-Differenzenquotient ist und kein Grenzwert betrachtet wird, ist jede q-Funktion q-differenzierbar, d.h. wir können für jede q-Funktion den entsprechenden q-Differenzenquotienten bestimmen. Unser Ziel in diesem Kapitel wird es sein, q-Ableitungen und q-Shifts für alle vorgestellten q-Funktionen herzuleiten, die eine für uns einfache Darstellung haben. Das soll bedeuten, die q-Ableitungen einer q-Funktion $f(x)$ sollen dargestellt werden als Linearkombination von bekannten linear unabhängigen q-Funktionen über $\mathbb{F}(x)$, vorrangig von der Ausgangsfunktion selbst. Dann können wir diese Darstellungen verwenden, um entsprechende einfache q-Differential- bzw. q-Rekursionsgleichungen für $f(x)$ aufzustellen.

2.1 q-Ableitungen und q-Shifts elementarer q-Funktionen

2.1.1 q-Ableitungen und q-Shifts bzgl. q

Die Tabelle 2.1 listet die q-Ableitungen und q-Shifts der elementaren q-Funktionen, der q-Exponentialfunktionen und q-trigonometrischen Funktionen auf. Für die Einträge der Tabelle gelte $k \in \mathbb{N}$ und $a \in \mathbb{C}$. Bei den ersten zehn Einträgen von Tabelle 2.1 handelt es sich um q-hypergeometrische Terme, bei denen die q-Ableitung bzw. der q-Shift gegeben ist durch

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

bzw.

$$\varepsilon_q f(x) = f(qx) = r(x) f(x).$$

mit $r(x) = \frac{f(qx)}{f(x)} \in \mathbb{F}(x)$. Die restlichen Einträge beweist man durch Anwendung des jeweiligen q-Operators auf die entsprechende Potenzreihe. Wir beweisen exemplarisch den neunten und zwölften Eintrag. Aus

$$\begin{aligned}
D_q \mathrm{Exp}_q(x) &= D_q \sum_{k=0}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_k} x^k = \sum_{k=0}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_k} D_q x^k = \sum_{k=1}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_k} [q]_k x^{k-1} \\
&= \sum_{k=1}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_k} \frac{1-q^k}{1-q} x^{k-1} = \sum_{k=1}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_{k-1}} \frac{1}{1-q} x^{k-1} = \frac{1}{1-q} \sum_{k=0}^{\infty} \frac{q^{\binom{k+1}{2}}}{(q;q)_k} x^k \\
&= \frac{1}{1-q} \sum_{k=0}^{\infty} \frac{q^{\binom{k}{2}+k}}{(q;q)_k} x^k = \frac{1}{1-q} \sum_{k=0}^{\infty} \frac{q^{\binom{k}{2}}}{(q;q)_k} (qx)^k = \frac{1}{1-q} \varepsilon_q \mathrm{Exp}_q(x)
\end{aligned}$$

q-Funktion $f(x)$	q-Ableitung $D_q f(x)$	q-Shift $\varepsilon_q f(x)$
x^k	$[k]_q\, x^{k-1}$ bzw. $\frac{[k]_q}{x} x^k$	$q^k x^k$
$(x \ominus a)_q^k$	$[k]_q\, (x \ominus a)_q^{k-1}$ bzw. $\frac{[k]_q}{x-aq^{k-1}} (x \ominus a)_q^k$	$\frac{q^k(x-aq^{-1})}{x-aq^{k-1}} (x \ominus a)_q^k$
$(a \ominus x)_q^k$	$\frac{[k]_q}{x-a} (a \ominus x)_q^k$	$\frac{a-xq^k}{a-x} (a \ominus x)_q^k$
$(a \ominus x)_q^\infty$	$\frac{x-a+1}{(q-1)(a-x)x} (a \ominus x)_q^\infty$	$\frac{1}{a-x} (a \ominus x)_q^\infty$
$(x;q)_k$	$\frac{[k]_q}{x-1} (x;q)_k$	$\frac{1-xq^k}{1-x} (x;q)_k$
$(x;q)_\infty$	$\frac{1}{(q-1)(1-x)} (x;q)_\infty$	$\frac{1}{1-x} (x;q)_\infty$
$\exp_q(x)$	$\frac{1}{1-q}\exp_q(x)$	$(1-x)\exp_q(x)$
$\mathrm{e}_q(x)$	$\mathrm{e}_q(x)$	$((q-1)x+1)\mathrm{e}_q(x)$
$\mathrm{Exp}_q(x)$	$\frac{1}{(1-q)(x+1)}\mathrm{Exp}_q(x)$	$\frac{1}{x+1}\mathrm{Exp}_q(x)$
$\mathrm{E}_q(x)$	$\frac{1}{(1-q)x+1}\mathrm{E}_q(x)$	$\frac{1}{(1-q)x+1}\mathrm{E}_q(x)$
$\sin_q(x)$	$\frac{1}{1-q}\cos_q(x)$	$-x\cos_q(x)+\sin_q(x)$
$\mathrm{s}_q(x)$	$\mathrm{c}_q(x)$	$(q-1)x\,\mathrm{c}_q(x)+\mathrm{s}_q(x)$
$\mathrm{Sin}_q(x)$	$\frac{1}{(1-q)(1+x^2)}\mathrm{Cos}_q(x)+\frac{x}{(1-q)(1+x^2)}\mathrm{Sin}_q(x)$	$-\frac{x}{1+x^2}\mathrm{Cos}_q(x)+\frac{1}{1+x^2}\mathrm{Sin}_q(x)$
$\mathrm{S}_q(x)$	$\frac{1}{(1-q)^2x^2+1}\mathrm{C}_q(x)+\frac{1-q}{(1-q)^2x^2+1}\mathrm{S}_q(x)$	$\frac{(q-1)x}{(1-q)^2x^2+1}\mathrm{C}_q(x)+\frac{1}{(1-q)^2x^2+1}\mathrm{S}_q(x)$
$\cos_q(x)$	$\frac{1}{q-1}\sin_q(x)$	$\cos_q(x)+x\sin_q(x)$
$\mathrm{c}_q(x)$	$-\mathrm{s}_q(x)$	$\mathrm{c}_q(x)+(1-q)x\,\mathrm{s}_q(x)$
$\mathrm{Cos}_q(x)$	$\frac{x}{(1-q)(1+x^2)}\mathrm{Cos}_q(x)+\frac{1}{(q-1)(1+x^2)}\mathrm{Sin}_q(x)$	$\frac{1}{1+x^2}\mathrm{Cos}_q(x)+\frac{x}{1+x^2}\mathrm{Sin}_q(x)$
$\mathrm{C}_q(x)$	$\frac{(1-q)x}{(1-q)^2x^2+1}\mathrm{C}_q(x)-\frac{1}{(1-q)^2x^2+1}\mathrm{S}_q(x)$	$\frac{1}{(1-q)^2x^2+1}\mathrm{C}_q(x)+\frac{(1-q)x}{(1-q)^2x^2+1}\mathrm{S}_q(x)$

Tabelle 2.1 q-Ableitungen und q-Shifts von q-Funktionen bzgl. q

und

$$\begin{aligned}(1+x)\varepsilon_q\mathrm{Exp}_q(x) &= \sum_{k=0}^{\infty}\frac{q^{\binom{k}{2}}}{(q;q)_k}(qx)^k + x\sum_{k=0}^{\infty}\frac{q^{\binom{k}{2}}}{(q;q)_k}(qx)^k\\ &= \sum_{k=0}^{\infty}\frac{q^{\binom{k+1}{2}}}{(q;q)_k}x^k + \sum_{k=0}^{\infty}\frac{q^{\binom{k+1}{2}}}{(q;q)_k}x^{k+1} = 1+\sum_{k=1}^{\infty}\frac{q^{\binom{k+1}{2}}}{(q;q)_k}x^k + \sum_{k=1}^{\infty}\frac{q^{\binom{k}{2}}}{(q;q)_{k-1}}x^k\\ &= 1+\sum_{k=1}^{\infty}\frac{q^{\binom{k+1}{2}}+q^{\binom{k}{2}}(1-q^k)}{(q;q)_k}x^k = 1+\sum_{k=1}^{\infty}\frac{q^{\binom{k+1}{2}}+q^{\binom{k}{2}}-q^{\binom{k+1}{2}}}{(q;q)_k}x^k\\ &= \sum_{k=0}^{\infty}\frac{q^{\binom{k}{2}}}{(q;q)_k}x^k = \mathrm{Exp}_q(x)\end{aligned}$$

ergibt sich die q-Ableitung und der q-Shift von $\mathrm{Exp}_q(x)$. Wie man sieht, sind im q-Fall also alle q-Exponentialfunktionen q-hypergeometrische Terme im Gegensatz zum klassischen Fall. Betrachten wir nun die kleine q-Sinusfunktion nach Kac/Cheung, so erhalten wir einerseits

$$\begin{aligned}D_q\mathrm{s}_q(x) &= D_q\sum_{k=0}^{\infty}\frac{(-1)^k}{[2k+1]_q!}x^{2k+1} = \sum_{k=0}^{\infty}\frac{(-1)^k}{[2k+1]_q!}D_q x^{2k+1}\\ &= \sum_{k=0}^{\infty}\frac{(-1)^k}{[2k+1]_q!}[2k+1]_q\,x^{2k} = \sum_{k=0}^{\infty}\frac{(-1)^k}{[2k]_q!}x^{2k} = \mathrm{c}_q(x)\end{aligned}$$

und andererseits

$$\begin{aligned}(q-1)x\mathrm{c}_q(x)+\mathrm{s}_q(x) &= (q-1)x\sum_{k=0}^{\infty}\frac{(-1)^k}{[2k]_q!}x^{2k}+\sum_{k=0}^{\infty}\frac{(-1)^k}{[2k+1]_q!}x^{2k+1}\\ &= \sum_{k=0}^{\infty}\frac{(-1)^k(q-1)}{[2k]_q!}x^{2k+1}+\sum_{k=0}^{\infty}\frac{(-1)^k}{[2k+1]_q!}x^{2k+1}\\ &= \sum_{k=0}^{\infty}\frac{(q-1)[2k+1]_q+1}{[2k+1]_q!}(-1)^k x^{2k+1}\\ &= \sum_{k=0}^{\infty}\frac{-(1-q^{2k+1})+1}{[2k+1]_q!}(-1)^k x^{2k+1}\\ &= \sum_{k=0}^{\infty}\frac{(-1)^k q^{2k+1}}{[2k+1]_q!}x^{2k+1}=\varepsilon_q\mathrm{s}_q(x).\end{aligned}$$

2.1.2 q-Ableitungen und q-Shifts bzgl. q^{-1}

Häufig ist nicht die q-Ableitung $D_q f(x)$ bzw. der q-Shift $\varepsilon_q f(x)$ einer q-Funktion $f(x)$ gefragt, sondern die q-Ableitung $D_{q^{-1}}f(x)$ bzw. der q-Shift $\varepsilon_{q^{-1}}f(x)$. Natürlich kann man diese q-Ableitungen und q-Shifts auch wieder wie im vorigen Abschnitt herleiten. Wir möchten aber nun entsprechende Formeln aus den bereits aufgestellten q-Shifts $\varepsilon_q f(x)$ ableiten. Nehmen wir an, $f(x)$ sei q-hyper-

q-Funktion $f(x)$	q-Ableitung $D_{q^{-1}}f(x)$	q-Shift $\varepsilon_{q^{-1}}f(x)$
x^k	$\frac{[k]_q}{q^{k-1}}x^{k-1}$	$\frac{1}{q^k}x^k$
$(x\ominus a)_q^k$	$\frac{[k]_q}{q^{k-1}(x-a)}(x\ominus a)_q^k$	$\frac{x-aq^k}{q^k(x-a)}(x\ominus a)_q^k$
$(a\ominus x)_q^k$	$\frac{[k]_q}{q^{k-1}x-a}(a\ominus x)_q^k$	$\frac{x-qa}{q^kx-qa}(a\ominus x)_q^k$
$(a\ominus x)_q^\infty$	$\frac{x-(a-1)q}{(q-1)x}(a\ominus x)_q^\infty$	$\frac{qa-x}{q}(a\ominus x)_q^\infty$
$(x;q)_k$	$\frac{[k]_q}{q^{k-1}x-1}(x;q)_k$	$\frac{x-q}{q^kx-q}(x;q)_k$
$(x;q)_\infty$	$\frac{1}{q-1}(x;q)_\infty$	$\frac{q-x}{q}(x;q)_\infty$
$\exp_q(x)$	$\frac{q}{(1-q)(q-x)}\exp_q(x)$	$\frac{q}{q-x}\exp_q(x)$
$\mathrm{e}_q(x)$	$\frac{q}{(q-1)x+q}\mathrm{e}_q(x)$	$\frac{q}{(q-1)x+q}\mathrm{e}_q(x)$
$\mathrm{Exp}_q(x)$	$\frac{1}{1-q}\mathrm{Exp}_q(x)$	$\frac{q+x}{q}\mathrm{Exp}_q(x)$
$\mathrm{E}_q(x)$	$\mathrm{E}_q(x)$	$\frac{(1-q)x+1}{q}\mathrm{E}_q(x)$
$\sin_q(x)$	$\frac{q^2}{(1-q)(q^2+x^2)}\cos_q(x)-\frac{qx}{(1-q)(q^2+x^2)}\sin_q(x)$	$\frac{qx}{q^2+x^2}\cos_q(x)+\frac{q^2}{q^2+x^2}\sin_q(x)$
$\mathrm{s}_q(x)$	$\frac{q^2}{(q-1)^2x^2+q^2}\mathrm{c}_q(x)+\frac{q(q-1)x}{(q-1)^2x^2+q^2}\mathrm{s}_q(x)$	$\frac{q(1-q)x}{(q-1)^2x^2+q^2}\mathrm{c}_q(x)+\frac{q^2}{(q-1)^2x^2+q^2}\mathrm{s}_q(x)$
$\mathrm{Sin}_q(x)$	$\frac{1}{1-q}\mathrm{Cos}_q(x)$	$\frac{x}{q}\mathrm{Cos}_q(x)+\mathrm{Sin}_q(x)$
$\mathrm{S}_q(x)$	$\mathrm{C}_q(x)$	$\frac{(1-q)x}{q}\mathrm{C}_q(x)+\mathrm{S}_q(x)$
$\cos_q(x)$	$\frac{qx}{(q-1)(q^2+x^2)}\cos_q(x)+\frac{q^2}{(q-1)(q^2+x^2)}\sin_q(x)$	$\frac{q^2}{q^2+x^2}\cos_q(x)-\frac{qx}{q^2+x^2}\sin_q(x)$
$\mathrm{c}_q(x)$	$\frac{q(q-1)x}{(q-1)^2x^2+q^2}\mathrm{c}_q(x)-\frac{q^2}{(q-1)^2x^2+q^2}\mathrm{s}_q(x)$	$\frac{q^2}{(q-1)^2x^2+q^2}\mathrm{c}_q(x)+\frac{q(q-1)x}{(q-1)^2x^2+q^2}\mathrm{s}_q(x)$
$\mathrm{Cos}_q(x)$	$\frac{1}{q-1}\mathrm{Sin}_q(x)$	$\mathrm{Cos}_q(x)-\frac{x}{q}\mathrm{Sin}_q(x)$
$\mathrm{C}_q(x)$	$-\mathrm{S}_q(x)$	$\mathrm{C}_q(x)+\frac{(q-1)x}{q}\mathrm{S}_q(x)$

Tabelle 2.2 q-Ableitungen und q-Shifts von q-Funktionen bzgl. q^{-1}

geometrisch, dann haben wir $\varepsilon_q f(x)=r(x)f(x)$ mit $r(x)\in\mathbb{F}(x)$ gegeben. Wenden wir nun den inversen q-Shiftoperator $\varepsilon_{q^{-1}}$ an, so erhalten wir nach Umstellen

$$\varepsilon_{q^{-1}}f(x)=\frac{1}{r(q^{-1}x)}f(x).$$

Folglich gilt für die q-Ableitung $D_{q^{-1}}f(x)$

$$D_{q^{-1}}f(x) = \frac{f(x) - \varepsilon_{q^{-1}}f(x)}{(1-q^{-1})x} = \underbrace{\frac{r(q^{-1}x) - 1}{r(q^{-1}x)(1-q^{-1})x}}_{\in \mathbb{F}(x)} f(x).$$

Diese Vorgehensweise liefert die ersten zehn Einträge der Tabelle 2.2. Die weiteren Einträge erhält man, indem man wie folgt vorgeht. Betrachten wir exemplarisch den q-Shift $\varepsilon_{q^{-1}}\sin_q(x)$. Wir erhalten aus den bereits bekannten Formeln $\varepsilon_q \sin_q(x) = -x\cos_q(x) + \sin_q(x)$ und $\varepsilon_q \cos_q(x) = \cos_q(x) + x\sin_q(x)$ durch Anwendung des inversen q-Shiftoperators $\varepsilon_{q^{-1}}$ die Formeln

$$\sin_q(x) = -\frac{x}{q}\varepsilon_{q^{-1}}\cos_q(x) + \varepsilon_{q^{-1}}\sin_q(x) \quad \text{und} \quad \cos_q(x) = \varepsilon_{q^{-1}}\cos_q(x) + \frac{x}{q}\varepsilon_{q^{-1}}\sin_q(x).$$

Nach Elimination von $\varepsilon_{q^{-1}}\cos_q(x)$ und Auflösen nach $\varepsilon_{q^{-1}}\sin_q(x)$ bekommen wir die Lösung

$$\varepsilon_{q^{-1}}\sin_q(x) = \frac{qx}{q^2+x^2}\cos_q(x) + \frac{q^2}{q^2+x^2}\sin_q(x).$$

Setzt man diese Gleichung nun in

$$D_{q^{-1}}\sin_q(x) = \frac{\sin_q(x) - \varepsilon_{q^{-1}}\sin_q(x)}{\left(1-\frac{1}{q}\right)x}$$

ein, so erhält man auch die q-Ableitung $D_{q^{-1}}\sin_q(x)$.

2.2 q-Ableitungs- und q-Shiftregeln klassischer q-orthogonaler Polynome

An dieser Stelle wollen wir eine allgemeine q-Ableitungs- und eine allgemeine q-Shiftregel für klassische q-orthogonale Polynome aufstellen. In [KS01] wurde gezeigt, dass klassische q-orthogonale Polynome die folgende *Strukturformel* erfüllen.

Satz 2.1 Sei $P_n(x)$ ein klassisches q-orthogonales Polynom, also ein Polynom n-ten Grades, welches die q-Differentialgleichung (1.15) erfüllt. Dann gibt es α_n, β_n und $\gamma_n \in \mathbb{F}(q^n)$, so dass

$$\sigma(x)D_{q^{-1}}P_n(x) = \alpha_n P_{n+1}(x) + \beta_n P_n(x) + \gamma_n P_{n-1}(x) \tag{2.1}$$

gilt.

Aus diesem Satz erhalten wir neue q-Shiftregeln, die den q-Shift eines klassischen q-orthogonalen Polynoms $P_n(x)$ als Linearkombination von $P_{n+i}(x)$ mit über $\mathbb{F}(q^n)(x)$ linear unabhängigen Summanden darstellt.

Korollar 2.2 Sei $P_n(x)$ ein klassisches q-orthogonales Polynom, also ein Polynom n-ten Grades, welches die q-Differentialgleichung (1.15) erfüllt. Dann gibt es $\bar{\alpha}_n, \bar{\beta}_n \in \mathbb{F}(q^n)(x)$ bzw. $\tilde{\alpha}_n, \tilde{\beta}_n \in \mathbb{F}(q^n)(x)$, so dass

$$\varepsilon_{q^{-1}}P_n(x) = \bar{\alpha}_n(x)P_n(x) + \bar{\beta}_n(x)P_{n-1}(x) \tag{2.2}$$

bzw.

$$\varepsilon_q P_n(x) = \tilde{\alpha}_n(x)P_n(x) + \tilde{\beta}_n(x)P_{n-1}(x) \tag{2.3}$$

gilt.

Beweis Aus der Strukturformel (2.1) und der q-Rekursionsgleichung (1.16) mit $A_n, B_n, C_n, \alpha_n, \beta_n, \gamma_n \in \mathbb{F}(q^n)$ folgt nach Umstellen von

$$\sigma(x)\frac{P_n(x) - \varepsilon_{q^{-1}}P_n(x)}{\left(1-\frac{1}{q}\right)x} = \alpha_n P_{n+1}(x) + \beta_n P_n(x) + \gamma_n P_{n-1}(x)$$

die q-Shiftregel

$$\begin{aligned}\varepsilon_{q^{-1}}P_n(x) &= \alpha_n\frac{(1-q)x}{q\sigma(x)}P_{n+1}(x) + \left(1+\beta_n\frac{(1-q)x}{q\sigma(x)}\right)P_n(x) + \gamma_n\frac{(1-q)x}{q\sigma(x)}P_{n-1}(x)\\ &= \alpha_n\frac{(1-q)x}{q\sigma(x)}\left((A_nx+B_n)P_n(x) - C_nP_{n-1}(x)\right) + \left(1+\beta_n\frac{(1-q)x}{q\sigma(x)}\right)P_n(x) + \gamma_n\frac{(1-q)x}{q\sigma(x)}P_{n-1}(x)\\ &= \underbrace{\left(\frac{(1-q)x}{q\sigma(x)}\left(\alpha_n(A_nx+B_n)+\beta_n\right)+1\right)}_{=:\bar{\alpha}_n(x)\in\mathbb{F}(q^n)(x)}P_n(x) + \underbrace{\left(\frac{(1-q)x}{q\sigma(x)}\left(\gamma_n-\alpha_nC_n\right)\right)}_{=:\bar{\beta}_n(x)\in\mathbb{F}(q^n)(x)}P_{n-1}(x).\end{aligned}$$

Drückt man die q-Differentialgleichung der klassischen q-orthogonalen Polynome (1.15) mit dem q-Shiftoperator aus[1], so erhält man

$$\frac{\sigma(x)-(1-q)\tau(x)}{(1-q)^2x^2}\varepsilon_qP_n(x) + \frac{\lambda_n(1-q)^2x^2-(1+q)\sigma(x)+(1-q)\tau(x)}{(1-q)^2x^2}P_n(x) + \frac{q\sigma(x)}{(1-q)^2x^2}\varepsilon_{q^{-1}}P_n(x) = 0.$$

Nach Umstellen dieser Gleichung nach $\varepsilon_qP_n(x)$ und Einsetzen der obigen q-Shiftregel für $\varepsilon_{q^{-1}}P_n(x)$, bekommt man die entsprechende q-Shiftregel (2.3) für $\varepsilon_qP_n(x)$. □

Mit dem folgenden Satz können wir allgemeine q-Shiftregeln für klassische q-orthogonale Polynome explizit durch Kenntnis der definierenden q-Differentialgleichung und des führenden Koeffizienten k_n angeben.

Satz 2.3 Die Koeffizienten der q-Shiftregel

$$\varepsilon_{q^{-1}}P_n(x) = \bar{\alpha}_n(x)P_n(x) + \bar{\beta}_n(x)P_{n-1}(x)$$

bzw.

$$\varepsilon_qP_n(x) = \tilde{\alpha}_n(x)P_n(x) + \tilde{\beta}_n(x)P_{n-1}(x)$$

der klassischen q-orthogonalen Polynome sind gegeben durch

$$\begin{aligned}\bar{\alpha}_n(x) =& \frac{1}{\left((-qd+d-a)\left(q^n\right)^2+q^2a\right)q^n\left(ax^2+bx+c\right)}\cdot\\ &\Big(\left((-qad-a(a-d))\left(q^n\right)^2+q^2a^2\right)x^2\\ &+\left(\left(-eq^2a+(ea-bd-ab)q-b(a-d)\right)\left(q^n\right)^2+\left(a(b+e)q^2+a(-e+b)q\right)q^n\right)x\\ &+(-qcd-c(a-d))\left(q^n\right)^3+cq^2q^na\Big)\end{aligned}$$

und

$$\begin{aligned}\bar{\beta}_n(x) =& \frac{k_n}{k_{n-1}}\frac{1}{\left(((1-q)d-a)\left(q^n\right)^2+q^3a\right)\left(((1-q)d-a)\left(q^n\right)^2+q^2a\right)^2}\frac{x}{\left(ax^2+bx+c\right)}\cdot\\ &\Big(\left(1-q^n\right)\left(\left(((1-q)d-a)q^n+q^2a\right)\left(c((q-1)d+a)^2\left(q^n\right)^4+qb((q-1)e+b)((q-1)d+a)\left(q^n\right)^3\right.\right.\\ &-q^2\left(-b^2d-2ca^2+2ab^2+qb^2d+2acd+2qaeb-2aeb-2qacd-2qae^2+q^2ae^2+ae^2\right)\left(q^n\right)^2\\ &\left.\left.+q^3ab((q-1)e+b)q^n-q^4ca^2\right)\right)\end{aligned}$$

bzw.

$$\begin{aligned}\tilde{\alpha}_n(x) =& \frac{1}{((q-1)d+a)(q^n)^2-q^2a}\frac{1}{((q-1)d+a)x^2+((q-1)e+b)x+c}\cdot\\ &\Big(\left(((q-1)d+a)^2\left(q^n\right)^3-aq^2((q-1)d+a)q^n\right)x^2+\left((qb-e+b+eq)((q-1)d+a)\left(q^n\right)^2\right.\\ &\left.-q(eq^2a+qab-eqa+bqd+ab-bd)q^n\right)x+c(((q-1)d+a)\left(q^n\right)^2-acq^2\Big)\end{aligned}$$

[1] siehe auch Gleichung (1.17)

und

$$\tilde{\beta}_n(x) = \frac{k_n}{k_{n-1}} \frac{1}{\left(((q-1)d+a)\left(q^n\right)^2 - q^3a\right)\left(((q-1)d+a)^2\left(q^n\right)^4 - 2q^2a((q-1)d+a)\left(q^n\right)^2 + q^4a^2\right)} \cdot \frac{q(q^n-1)x}{(((q-1)d+a)x^2 + ((q-1)e+b)x + c)} \cdot \Big(\left(((q-1)d+a)q^n - q^2a\right)\left(c((q-1)d+a)^2\left(q^n\right)^4 - bq((q-1)e+b)((q-1)d+a)\left(q^n\right)^3 - q^2(-ae^2q^2 + 2qae^2 - 2abeq - b^2dq + 2acqd + 2ca^2 - 2ab^2 + 2abe - ae^2 + b^2d - 2acd)\left(q^n\right)^2 - abq^3((q-1)e+b)q^n + q^4ca^2\right)\Big).$$

Beweis Sei $P_n(x) = k_n x^n + k_n' x^{n-1} + k_n'' x^{n-2} + \ldots$ ein klassisches q-orthogonales Polynom mit Bezeichnungen wie in (1.15) gegeben. Setzt man $P_n(x)$ in die q-Differentialgleichung (1.15) ein, so erhält man nach Vergleich des höchsten Koeffizienten

$$\hat{\eta}_n = -\left(a\,[n]_{q^{-1}}\,[n-1]_q + d\,[n]_q\right)$$

und nach Vergleich weiterer Koeffizienten Ausdrücke für k_n' und k_n'' als rationales Vielfaches von k_n. Diese Ausdrücke substituiert man nun in $P_n(x)$ und nach Einsetzen von $P_n(x)$ in die q-Rekursionsgleichung (1.16) erhält man wiederum durch Koeffizientenvergleich Formeln für A_n, B_n und C_n in Abhängigkeit von $a, b, c, d, e \in \mathbb{F}$, $\hat{\eta}_n \in \mathbb{F}(q^n)$ und $k_n \in \mathbb{F}$. Analog erhält man durch Einsetzen von $P_n(x)$ in die Strukturformel (2.1) Formeln für α_n, β_n und γ_n. Substituiert man nun diese Formeln in die letzte Gleichung im Beweis von Korollar 2.2, so erhält man obige Formeln für $\bar{\alpha}_n(x), \bar{\beta}_n(x)$ und nach Fortführung des Beweises von Korollar 2.2 auch $\tilde{\alpha}_n(x), \tilde{\beta}_n(x)$. □

In Tabelle 2.3 bzw. Tabelle 2.4 auf Seite 24 bzw. Seite 25 sind alle q-Shiftregeln für die klassischen q-orthogonalen Polynome aufgelistet. Diese Formeln wurden mit Hilfe von Tabelle 1.1 und den Formeln aus Satz 2.3 hergeleitet. Analog kann man entsprechende q-Ableitungsregeln für klassische q-orthogonale Polynome aufstellen.

Betrachten wir nun an Beispielen die implementierten Regeln in Maple

Maple-Sitzung 2.1 (q-Shift- und q-Ableitungsregeln)

q-Shiftregel für die q-Laguerrepolynome bzgl. q^{-1}

```
> qshift(qLaguerre(n,alpha,x,q),x,1/q);
```

$$\frac{\left(q^n q^\alpha - 1\right) qLaguerre\left(n-1,\alpha,x,q\right)}{q^n} + \frac{qLaguerre\left(n,\alpha,x,q\right)}{q^n}$$

q-Shiftregel für die q-Laguerrepolynome bzgl. q

```
> qshift(qLaguerre(n,alpha,x,q),x,q);
```

$$-\frac{\left(q^n q^\alpha - 1\right) qLaguerre\left(n-1,\alpha,x,q\right)}{q^n q^\alpha\left(x+1\right)} + \frac{\left(-1+q^n+q^n q^\alpha + q^\alpha x\left(q^n\right)^2\right) qLaguerre\left(n,\alpha,x,q\right)}{q^n q^\alpha\left(x+1\right)}$$

q-Ableitungsregel für die q-Laguerrepolynome bzgl. q^{-1}

```
> qdiff(qLaguerre(n,alpha,x,q),x,1/q);
```

$$\frac{q\left(-1+q^n\right) qLaguerre\left(n,\alpha,x,q\right)}{q^n x\left(q-1\right)} - \frac{\left(q^n q^\alpha - 1\right) qLaguerre\left(n-1,\alpha,x,q\right) q}{q^n x\left(q-1\right)}$$

q-Ableitungsregel für die q-Laguerrepolynome bzgl. q

```
> qdiff(qLaguerre(n,alpha,x,q),x,q);
```

$$\frac{\left(-1+q^n\right)\left(xq^n q^\alpha + 1\right) qLaguerre\left(n,\alpha,x,q\right)}{q^n q^\alpha\left(x+1\right)\left(q-1\right)x} - \frac{\left(q^n q^\alpha - 1\right) qLaguerre\left(n-1,\alpha,x,q\right)}{q^n q^\alpha\left(x+1\right)\left(q-1\right)x}$$

q-Shiftregel für die diskreten q-Hermite II-Polynome bzgl. q

```
> qshift(DiscreteqHermiteII(n,x,q),x,q);
```

$$\frac{\left(x^2 q^n + 1\right) DiscreteqHermiteII\left(n,x,q\right)}{x^2+1} + \frac{xq\left(-1+q^n\right) DiscreteqHermiteII\left(n-1,x,q\right)}{q^n\left(x^2+1\right)}$$

q-Ableitungsregel für die diskreten q-Hermite II-Polynome bzgl. q

```
> qdiff(DiscreteqHermiteII(n,x,q),x,q);
```

$$\frac{x\left(-1+q^n\right)\mathit{DiscreteqHermiteII}\left(n,x,q\right)}{\left(x^2+1\right)\left(q-1\right)}+\frac{q\left(-1+q^n\right)\mathit{DiscreteqHermiteII}\left(n-1,x,q\right)}{q^n\left(x^2+1\right)\left(q-1\right)}$$

Die in Maple implementierte q-Differentiationsprozedur `qdiff` bzw. q-Shiftprozedur `qshift` beinhaltet alle Regeln (1.2), (1.4) und (1.6) bzw. (1.8), (1.9) und (1.10) sowie alle in dieser Arbeit hergeleiteten q-Ableitungen und q-Shifts der q-Funktionen. Ferner sind beide Prozeduren rekursiv implementiert, so dass auch höhere q-Ableitungen und q-Shifts bestimmt werden können. Eine genaue Beschreibung der Prozeduren ist in Kapitel 6 zu finden. Dieses Werkzeug befähigt uns nun, mit q-orthogonalen Polynomen und weiteren q-Funktionen und deren q-Ableitungen und q-Shifts in einem Computeralgebrasystem zu arbeiten.

q-orthogonales Polynom		$\bar{\alpha}_n(x)$	$\bar{\beta}_n(x)$
$\tilde{\mathrm{h}}_n(x;q)$	diskrete q-Hermite II	1	$\frac{(1-q^n)x}{q^n}$
$\mathrm{V}_n^{(a)}(x;q)$	Al-Salam-Carlitz II	1	$\frac{(1-q^n)x}{q^n}$
$\mathrm{C}_n(x;a;q)$	q-Charlier	$\frac{1}{q^n}$	$\frac{(q^n-1)(q^n+a)}{aq^n}$
$\mathrm{L}_n^{(\alpha)}(x;q)$	q-Laguerre	$\frac{1}{q^n}$	$\frac{q^nq^\alpha-1}{q^n}$
$\mathrm{S}_n(x;q)$	Stieltjes-Wigert	$\frac{1}{q^n}$	$-\frac{1}{q^n}$
$\mathrm{M}_n(x;b,c;q)$	q-Meixner	$\frac{x-bqq^n}{q^n(x-bq)}$	$\frac{(q^n-1)(q^n+c)x}{cq^n(x-bq)}$
$\mathrm{K}_n^{\mathrm{qtm}}(x;p,N;q)$	Quantum-q-Krawtchouk	$\frac{xq^N-q^n}{q^n(xq^N-1)}$	$\frac{(1-q^n)(1-pq^n)q^Nx}{q^n(xq^N-1)}$
$\mathrm{K}_n(x;p,N;q)$	q-Krawtchouk	$\frac{(q^Nq+q^Np(q^n)^2)x-p(q^Nq+1)(q^n)^2-q(1-q^Np)q^n}{(q+(q^n)^2p)q^n(q^Nx-1)}$	$\frac{pq^n(1-q^n)(1+q^Nq^np)}{(q+(q^n)^2p)(xq^N-1)}$
$\mathrm{p}_n(x;a\vert q)$	kleine q-Laguerre	$\frac{x+a(q^n)^2-(a+1)q^n}{q^n(x-1)}$	$\frac{a(1-q^n)}{x-1}$
$\mathrm{K}_n(x;a;q)$	alternative q-Charlier	$\frac{(a(q^n)^2+q)x-q^n(aq^n+q))}{q^n(a(q^n)^2+q)(x-1)}$	$\frac{aq^n(1-q^n)}{(a(q^n)^2+q)(x-1)}$
$\mathrm{p}_n(x;a,b\vert q)$	kleine q-Jacobi	$\frac{(ab(q^n)^2-1)x-a(b+1)(q^n)^2+(a+1)q^n}{q^n(1-x)(1-ab(q^n)^2)}$	$-\frac{a(1-q^n)(1-bq^n)}{(1-x)(1-ab(q^n)^2)}$
$\mathrm{h}_n(x;q)$	diskrete q-Hermite I	$\frac{x^2-q^n}{q^n(x-1)(x+1)}$	$\frac{(q^n-1)x}{q(x-1)(x+1)}$
$\mathrm{U}_n^{(a)}(x;q)$	Al-Salam-Carlitz I	$\frac{x^2-(a+1)q^nx+aq^n}{q^n(x-1)(x-a)}$	$\frac{a(1-q^n)x}{q(x-1)(x-a)}$
$\mathrm{P}_n(x;a,b;q)$	große q-Laguerre	$\frac{x^2+(abq(q^n)^2-q(a+b+ab)q^n)x+abq^2q^n}{q^n(x-aq)(x-bq)}$	$\frac{abq(1-q^n)x}{(x-aq)(x-bq)}$
$\mathrm{K}_n^{\mathrm{aff}}(x;p,N;q)$	affine q-Krawtchouk	$\frac{-q^Nx^2+(-(q^n)^2p+(q^Npq+p+1)q^n)x-pq^nq}{q^n(1-q^Nx)(x-pq)}$	$\frac{(1-q^n)xp}{(q^Nx-1)(x-pq)}$
$\mathrm{Q}_n(x;a,b;N\vert q)$	q-Hahn	$\frac{q^N(ab(q^n)^2-1)x^2-(a(bqq^N(a+1)+b+1)(q^n)^2-(aqq^N(b+1)+a+1)q^n)x+a^2bq(q^n)^3-aqq^n}{q^n(1-ab(q^n)^2)(1-q^Nx)(x-aq)}$	$-\frac{a(1-q^n)(1-bq^n)(1-abq^{N+n+1})x}{(1-ab(q^n)^2)(1-q^Nx)(x-aq)}$
$\mathrm{P}_n(x;a,b,c;q)$	große q-Jacobi	$\frac{(ab(q^n)^2-1)x^2-(aq(b+c+ab+bc)(q^n)^2-q(a+c+ab+ac)q^n)x+acq^2q^n(ab(q^n)^2-1)}{q^n(ab(q^n)^2-1)(x-aq)(x-cq)}$	$\frac{aq(1-q^n)(1-bq^n)(abq^n-c)x}{(ab(q^n)^2-1)(x-aq)(x-cq)}$

Tabelle 2.3 q-Shiftregeln bzgl. $\varepsilon_{q^{-1}}P_n(x)$ für die klassischen q-orthogonalen Polynome

q-orthogonales Polynom		$\tilde{\alpha}_n(x)$	$\tilde{\beta}_n(x)$
$\tilde{\mathrm{h}}_n(x;q)$	diskrete q-Hermite II	$\frac{1+q^n x^2}{1+x^2}$	$\frac{q(q^n-1)x}{q^n(1+x^2)}$
$\mathrm{V}_n^{(a)}(x;q)$	Al-Salam-Carlitz II	$\frac{q^n x^2-(1+a)x+a}{(x-1)(x-a)}$	$\frac{aq(q^n-1)x}{q^n(x-1)(x-a)}$
$\mathrm{C}_n(x;a;q)$	q-Charlier	$\frac{(q^n)^2x+q^n(a-1)-a}{q^n(x-1)}$	$\frac{(1-q^n)(a+q^n)}{q^n(x-1)}$
$\mathrm{L}_n^{(\alpha)}(x;q)$	q-Laguerre	$\frac{q^\alpha(q^n)^2x+q^n(1+q^\alpha)-1}{q^n q^\alpha(x+1)}$	$\frac{1-q^n q^\alpha}{q^n q^\alpha(x+1)}$
$\mathrm{S}_n(x;q)$	Stieltjes-Wigert	$\frac{(q^n)^2x+q^n-1}{q^n x}$	$\frac{1}{q^n x}$
$\mathrm{M}_n(x;b,c;q)$	q-Meixner	$\frac{(q^n)^2x^2+((c+bc-1)q^n-c)x-bcq^n}{q^n(x-1)(x+bc)}$	$\frac{(1-q^n)(c+q^n)x}{q^n(x-1)(x+bc)}$
$\mathrm{K}_n^{\mathrm{qtm}}(x;p,N;q)$	Quantum-q-Krawtchouk	$\frac{pqq^N(q^n)^2x^2-\left(\left(1+q(1+p)q^N\right)q^n-q^N q\right)x+q^n}{q^n(x-1)(q^N qpx-1)}$	$\frac{q(1-q^n)(q^n p-1)q^N x}{q^n(x-1)(q^N qpx-1)}$
$\mathrm{K}_n(x;p,N;q)$	q-Krawtchouk	$\frac{(q^N p(q^n)^3+q^N q^n q)x+(1-q^N p)(q^n)^2-(1+q^N q)q^n}{q^N(x-1)(q+(q^n)^2 p)}$	$\frac{q^n(1-q^n)(q^N q^n p+1)}{q^N(x-1)(q+(q^n)^2 p)}$
$\mathrm{p}_n(x;a\|q)$	kleine q-Laguerre	q^n	$1-q^n$
$\mathrm{K}_n(x;a;q)$	alternative q-Charlier	$\frac{q^n((a(q^n)^2+q)x+q^n-1)}{(a(q^n)^2+q)x}$	$\frac{q^n(1-q^n)}{(a(q^n)^2+q)x}$
$\mathrm{p}_n(x;a,b\|q)$	kleine q-Jacobi	$\frac{q^n((ab^2q(q^n)^2-qb)x-b(1+a)q^n+b+1)}{(1-bqx)(1-ab(q^n)^2)}$	$\frac{(1-q^n)(1-bq^n)}{(1-bqx)(1-ab(q^n)^2)}$
$\mathrm{h}_n(x;q)$	diskrete q-Hermite I	1	$\left(q^n-1\right)x$
$\mathrm{U}_n^{(a)}(x;q)$	Al-Salam-Carlitz I	1	$\left(q^n-1\right)x$
$\mathrm{P}_n(x;a,b;q)$	große q-Laguerre	$\frac{1-q^n x}{1-x}$	$\frac{(q^n-1)x}{1-x}$
$\mathrm{K}_n^{\mathrm{aff}}(x;p,N;q)$	affine q-Krawtchouk	$\frac{1-q^n x}{1-x}$	$\frac{(q^n-1)x}{1-x}$
$\mathrm{Q}_n(x;a,b;N\|q)$	q-Hahn	$\frac{(ab^2q^{N+1}(q^n)^3-bq^{N+n+1})x^2-(b(abq^{N+1}+aq^{N+1}+a+1)(q^n)^2-((a+1)bq^{N+1}+b+1)q^n)x+ab(q^n)^2-1}{(ab(q^n)^2-1)(1-x)(1-bq^{N+1}x)}$	$\frac{(1-q^n)(1-bq^n)(1-abq^{N+n+1})x}{(ab(q^n)^2-1)(1-x)(1-bq^{N+1}x)}$
$\mathrm{P}_n(x;a,b,c;q)$	große q-Jacobi	$\frac{(ab^2(q^n)^3-bq^n)x^2-(b(ac+ab+a+c)(q^n)^2-(ab+bc+b+c)q^n)x+abc(q^n)^2-c}{(1-ab(q^n)^2)(1-x)(bx-c)}$	$\frac{(1-q^n)(1-bq^n)(c-abq^n)x}{(1-ab(q^n)^2)(1-x)(bx-c)}$

Tabelle 2.4 q-Shiftregeln bzgl. $\varepsilon_q P_n(x)$ für die klassischen q-orthogonalen Polynome

Kapitel 3

Algorithmen für q-holonome Funktionen und q-holonome Rekursionsgleichungen

Nachdem wir im letzten Kapitel einige nützliche Vorarbeit geleistet haben, können wir nun sämtliche aufgestellten q-Ableitungs- und q-Shiftregeln der q-Funktionen dazu verwenden, um algorithmisch zu einer möglichst einfachen q-Differential- bzw. q-Rekursionsgleichung für eine q-Funktion zu gelangen. Wir werden uns dann auf q-Funktionen spezialisieren, die solche einfachen q-Differential- bzw. q-Rekursionsgleichungen erfüllen und sehen, dass diese q-Funktionen einen kommutativen Ring bilden. Ferner lernen wir Algorithmen kennen, die es uns erlauben, mit diesen q-Funktionen bzw. deren q-Rekursionsgleichungen zu operieren, darunter fallen der Summen-, Produkt- und Kompositionsalgorithmus.

3.1 q-Holonome Funktionen

Zunächst definieren wir, was wir unter einer einfachen q-Differential- bzw. q-Rekursionsgleichung verstehen wollen.

Definition 3.1 Sei f eine Funktion in x. Unter einer *q-holonomen Differentialgleichung* für $f(x)$ versteht man eine Gleichung der Form

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

die linear und homogen ist und zudem polynomiale Koeffizienten $a_k(x)$ aus $\mathbb{F}[x]$ besitzt. Den Operator $\sum_{k=0}^{n} a_k(x) D_q^k$ nennen wir *q-holonomen Differentialoperator* und die Menge aller q-holonomen Differentialoperatoren bezeichnen wir mit $\mathbb{F}[x][D_q]$.

Die analoge Formulierung für q-Rekursionsgleichungen lautet

Definition 3.2 Sei f eine Funktion in x. Unter einer *q-holonomen Rekursionsgleichung* für $f(x)$ versteht man eine Rekursionsgleichung der Form

$$\sum_{k=0}^{n} a_k(x) \varepsilon_q^k f(x) = 0, \tag{3.1}$$

die linear und homogen ist und zudem polynomiale Koeffizienten $a_k(x)$ aus $\mathbb{F}[x]$ besitzt. Den Operator $\sum_{k=0}^{n} a_k(x) \varepsilon_q^k$ nennen wir *q-holonomen Rekursionsoperator*. Mit Hilfe der Substitutionen $x \to q^j$

und $f(q^j) \to c_j$ können wir für $j \in \mathbb{Z}$ die q-holonome Rekursionsgleichung auch schreiben als

$$\sum_{k=0}^{n} a_k(q^j) c_{j+k} = 0.$$

Wir sagen dann, dass die q-holonome Rekursionsgleichung in *additiver Form* vorliegt, während die obige Gleichung (3.1) die *multiplikative Form* besitzt. Die Menge aller q-holonomen q-Rekursionsoperatoren bezeichnen wir mit $\mathbb{F}[x][\varepsilon_q]$.

Die Mengen $\mathbb{F}[x][D_q]$ und $\mathbb{F}[x][\varepsilon_q]$ sind nichtkommutative Polynomringe, so genannte *Ore-Ringe*. Es gelten dort die Kommutatorregeln aus Lemma 1.5 und 1.9. Auch wenn wir nicht genauer auf die Operationen in Ore-Ringen wie z.B. Rechts- bzw. Links-Multiplikation und -Division eingehen wollen, so möchten wir an dieser Stelle darauf hinweisen, dass alle Operationen mit Orepolynomen (bzw. Rekursionsoperatoren) in dem Maple-Package `qFPS` als Prozeduren verfügbar sind. In dem Package `qFPS` sind alle Algorithmen dieser Dissertation implementiert. Wir setzen grundlegende Begriffe aus dem Ore-Setting, wie z.B. Rechtsteiler, im Folgenden voraus. Nachzulesen ist die Theorie nicht-kommutativer Polynome in [Ore33]. Betrachten wir nun die Operationen in `qFPS`, wobei aus Gründen der Konsistenz mit späteren Algorithmen die Operanden keine Operatoren sind, sondern q-Rekursionen. Die Rechtsdivision können wir später verwenden, um beispielsweise zu testen, ob eine Rekursionsgleichung einen gegebenen q-hypergeometrischen Term als Lösung besitzt oder nicht[1].

Maple-Sitzung 3.1 (Operationen von q-Rekursionsoperatoren)

```
> RE1:=Sq[x,x](F(x))-(1+q)*Sq[x](F(x))+q*(1+x^2)*F(x)=0;
```

$$RE1 := Sq_{x,x}(F(x)) - (1+q)\,Sq_x(F(x)) + q\left(1+x^2\right)F(x) = 0$$

```
> RE2:=(1+x)*Sq[x](F(x))-F(x)=0;
```

$$RE2 := (1+x)\,Sq_x(F(x)) - F(x) = 0$$

Rechtsdivision mit Rest liefert das rechte Quotientenpolynom und das rechte Restpolynom

```
> quoRE:=qDivideRE(RE1,RE2,F(x),polynomial='quo');
```

$$quoRE := \frac{Sq_x(F(x))}{1+qx} - \frac{q\,(x+1+qx)\,F(x)}{(1+qx)\,(x+1)}$$

```
> remRE:=qDivideRE(RE1,RE2,F(x),polynomial='rem');
```

$$remRE := \frac{qx^2\left(qx^2+x+1+q+qx\right)F(x)}{(1+qx)\,(x+1)}$$

```
> qDivideRE(RE1,RE2,F(x),polynomial='all');
```

$$\left[\frac{Sq_x(F(x))}{1+qx} - \frac{q\,(x+1+qx)\,F(x)}{(1+qx)\,(1+x)}, \frac{qx^2\left(q+1+x+qx+qx^2\right)F(x)}{(1+qx)\,(1+x)}\right]$$

Test

```
> collect(qMultiplyRE(quoRE,RE2,F(x))+remRE,{Sq,F(x)},normal);
```

$$Sq_{x,x}(F(x)) + (-1-q)\,Sq_x(F(x)) + q\left(1+x^2\right)F(x)$$

Das kleinste gemeinsame Rechtsvielfache der q-Rekursionen (Ausgabe unterdrückt)

```
> LCRMRE:=qLCM(RE1,RE2,F(x),direction=right):
```

Test mit dem größten gemeinsamen Linksteiler

```
> qGCD(LCRMRE,RE2,F(x),direction=left);
```

$$(1+x)\,Sq_x(F(x)) - F(x)$$

```
> qGCD(LCRMRE,RE1,F(x),direction=left);
```

$$Sq_{x,x}(F(x)) + (-1-q)\,Sq_x(F(x)) + q\left(1+x^2\right)F(x)$$

[1] Aus dem q-Zertifikat des q-hypergeometrischen Terms $f(x)$ kann man eine entsprechende q-holonome Rekursion aufstellen. Wenn der Rest bei der Rechtsdivision der gegebenen Rekursionsgleichung *RE* durch diese q-holonome Rekursion Null ist, so besitzt *RE* ebenfalls die q-hypergeometrische Lösung $f(x)$.

Definition 3.3 Wir nennen $f(x)$ eine *q-holonome Funktion*, wenn sie eine q-holonome Differentialgleichung erfüllt. Die Menge aller q-holonomen Funktionen bezeichnen wir mit $\mathcal{H}_q(\mathbb{F})$. Eine q-holonome Funktion $f(x)$ besitzt den *q-holonomen Grad n*, wenn n die kleinste Ordnung einer für $f(x)$ gültigen q-holonomen Differentialgleichung ist[2]. Mit $\text{holdeg}_x(f(x))$ bezeichnen wir den q-holonomen Grad von $f(x)$.

Bemerkung 3.4 Obige Definitionen können wir auch auf q-holonome Rekursionsgleichungen zurückführen, da man jede q-holonome Differentialgleichung der Ordnung n in eine q-holonome Rekursionsgleichung der Ordnung n konvertieren kann und umgekehrt (siehe Satz 1.11).

Satz 3.5 Die Menge aller q-holonomen Funktionen $\mathcal{H}_q(\mathbb{F})$ mit den Verknüpfungen $+$ und $\cdot$ bildet einen kommutativen Ring. Insbesondere gelten also für $f, g \in \mathcal{H}_q(\mathbb{F})$

$$f + g \in \mathcal{H}_q(\mathbb{F}) \quad \text{und} \quad f \cdot g \in \mathcal{H}_q(\mathbb{F}).$$

Für den q-holonomen Grad von $f+g$ gilt $\text{holdeg}_x((f+g)(x)) \leq \text{holdeg}_x(f(x)) + \text{holdeg}_x(g(x))$ und für den q-holonomen Grad von $f \cdot g$ gilt $\text{holdeg}_x((f \cdot g)(x)) \leq \text{holdeg}_x(f(x)) \cdot \text{holdeg}_x(g(x))$.

Beweis Wir beweisen zunächst die Aussage für die Summe. Seien also $f, g \in \mathcal{H}_q(\mathbb{F})$ gegeben vom Grad $\text{holdeg}_x(f(x)) = n$ und $\text{holdeg}_x(g(x)) = m$. Bezeichne $V_f = \langle f(x), \varepsilon_q f(x), \varepsilon_q^2 f(x), \ldots \rangle$ den Vektorraum über dem Körper der rationalen Funktionen $\mathbb{F}(x)$, der von den q-Shifts von $f(x)$ erzeugt wird und sei L der minimale q-Rekursionsoperator mit $Lf(x) = 0$. Dann ist die Menge $\{f(x), \varepsilon_q f(x), \varepsilon_q^2 f(x), \ldots, \varepsilon_q^{n-1} f(x)\}$ eine Basis von V_f, da wegen der Minimalität von n die Vektoren $f(x), \varepsilon_q f(x), \varepsilon_q^2 f(x), \ldots, \varepsilon_q^{n-1} f(x)$ linear unabhängig sind und jeder höhere q-Shift sich durch sukzessives Shiften von $Lf(x)$ als Linearkombination von $f(x), \varepsilon_q f(x), \varepsilon_q^2 f(x), \ldots, \varepsilon_q^{n-1} f(x)$ darstellen lässt. Analog gilt $\dim(V_g) = m$ für $V_g = \langle g(x), \varepsilon_q g(x), \varepsilon_q^2 g(x), \ldots \rangle$. Betrachten wir nun die Summe der beiden Vektorräume $V_f + V_g$, so gilt $\dim(V_f + V_g) \leq n + m$. Ferner sind die Funktionen

$$f(x) + g(x), \varepsilon_q\left(f(x) + g(x)\right), \varepsilon_q^2\left(f(x) + g(x)\right), \ldots, \varepsilon_q^k\left(f(x) + g(x)\right)$$

allesamt aus $V_f + V_g$, so dass $n+m+1$ dieser Funktionen ($k = n+m$) über $\mathbb{F}(x)$ linear abhängig sind. Das bedeutet aber, dass es eine q-holonome Rekursionsgleichung gibt, die als Lösung $f(x) + g(x)$ besitzt und deren Ordnung höchstens $n + m$ ist. Folglich gilt $f + g \in \mathcal{H}_q(\mathbb{F})$. Für die zweite Aussage betrachten wir das Tensorprodukt $V_f \otimes V_g$. Sei U ein $\mathbb{F}(x)$-Vektorraum und ϕ die bilineare Abbildung mit $\phi : V_f \times V_g \to U$, $\left(\varepsilon_q^i f(x), \varepsilon_q^j g(x)\right) \mapsto \left(\varepsilon_q^i f(x)\right) \cdot \left(\varepsilon_q^j g(x)\right)$. Da es sich bei der linearen Abbildung $\psi : V_f \otimes V_g \to U$, $\varepsilon_q^i f(x) \otimes \varepsilon_q^j g(x) \mapsto \left(\varepsilon_q^i f(x)\right) \cdot \left(\varepsilon_q^j g(x)\right)$ um einen Isomorphismus handelt, gilt $V_f \otimes V_g \cong U$. Betrachten wir nun die Vektoren

$$\varepsilon_q^i\left(f(x) \cdot g(x)\right) = \left(\varepsilon_q^i f(x)\right) \cdot \left(\varepsilon_q^i g(x)\right) \in U \quad \text{mit} \quad i = 0, \ldots, k,$$

so sind wegen $\dim(V_f \otimes V_g) = n \cdot m$ genau $n \cdot m + 1$ ($k = n \cdot m$) dieser Vektoren über $F(x)$ linear abhängig und demzufolge existiert eine q-holonome Rekursionsgleichung für $f(x) \cdot g(x)$ mit maximaler Ordnung $n \cdot m$. □

Bemerkung 3.6 Nicht alle Funktionen sind q-holonom, was man anhand des folgenden Beispiels, welches aus [KRM07] stammt, sieht. Nehmen wir z.B. die Funktion $f(x) = a^x$ mit $a > 0$ und $a \neq 1$, dann besitzt der Vektorraum

$$\langle f(x), \varepsilon_q f(x), \varepsilon_q^2 f(x), \ldots \rangle = \langle a^x, a^{qx}, a^{q^2 x}, \ldots \rangle$$

über dem Körper der rationalen Funktionen $\mathbb{F}(x)$ offenbar unendliche Dimension und folglich gibt es keine q-holonome Rekursionsgleichung für $f(x)$.

3.1.1 q-Holonome Rekursionsgleichungen elementarer q-Funktionen

Wir betrachten nun einige einfache q-Funktionen und untersuchen diese auf q-Holonomie. q-Hypergeometrische Terme $f(x)$ mit q-Zertifikat $r(x) = \frac{s(x)}{t(x)}$ mit $s(x), t(x) \in \mathbb{F}[x]$ sind immer q-

[2] Polynome sind trivialerweise immer q-holonom vom Grad 1. Aus diesem Grund sagen wir bei Polynomfamilien $P_k(x)$ mit $\deg_x(P_k(x)) = k$, sie seien q-holonom vom Grad n, wenn die Polynome eine q-holonome Differentialgleichung minimaler Ordnung n erfüllen, bei der die Grade der Koeffizientenpolynome nicht von k abhängig sind.

holonom mit $\text{holdeg}_x(f(x)) = 1$. Die zugehörige q-Rekursionsgleichung lautet

$$t(x)\varepsilon_q f(x) - s(x)f(x) = 0$$

und ergibt sich automatisch aus der Definition des q-Zertifikats. Folglich sind z.B. die q-Funktionen x^k, $(x \ominus a)_q^k$, $(a \ominus x)_q^k$, $(a \ominus x)_q^\infty$, $(x;q)_k$, $(x;q)_\infty$ und alle q-Exponentialfunktionen bzgl. x q-holonom.

$f(x)$	q-holonome Rekursionsgleichung für $f(x)$
x^k	$\varepsilon_q f(x) - q^k f(x) = 0$
$(x \ominus a)_q^k$	$(qx - aq^k)\varepsilon_q f(x) - q^k(qx - a)f(x) = 0$
$(a \ominus x)_q^k$	$(x - a)\varepsilon_q f(x) - (q^k x - a)f(x) = 0$
$(a \ominus x)_q^\infty$	$(x - a)\varepsilon_q f(x) + f(x) = 0$
$(x;q)_k$	$(x - 1)\varepsilon_q f(x) - (q^k x - 1)f(x) = 0$
$(x;q)_\infty$	$(x - 1)\varepsilon_q f(x) + f(x) = 0$
$\exp_q(x)$	$\varepsilon_q f(x) + (x - 1)f(x) = 0$
$\text{Exp}_q(x)$	$(x + 1)\varepsilon_q f(x) - f(x) = 0$
$\text{e}_q(x)$	$\varepsilon_q f(x) + ((1 - q)x - 1)f(x) = 0$
$\text{E}_q(x)$	$((q - 1)x - 1)\varepsilon_q f(x) + f(x) = 0$

Tabelle 3.1 q-Holonome Rekursionsgleichungen elementarer q-Funktionen

q-Holonome Funktionen sind aber nicht immer q-hypergeometrisch, wie wir im Folgenden sehen werden.

3.1.2 q-Holonome Rekursionsgleichungen q-trigonometrischer Funktionen

Die q-trigonometrischen Funktionen sind alle q-holonom. Sie erfüllen alle eine q-holonome Rekursionsgleichung der Ordnung 2 und besitzen den q-holonomen Grad 2.

$f(x)$	q-holonome Rekursionsgleichung für $f(x)$
$\sin_q(x)$ bzw. $\cos_q(x)$	$\varepsilon_q^2 f(x) - (q + 1)\varepsilon_q f(x) + q(x^2 + 1)f(x) = 0$
$\text{Sin}_q(x)$ bzw. $\text{Cos}_q(x)$	$(q^2x^2 + 1)\varepsilon_q^2 f(x) - (q + 1)\varepsilon_q f(x) + qf(x) = 0$
$\text{s}_q(x)$ bzw. $\text{c}_q(x)$	$\varepsilon_q^2 f(x) - (q + 1)\varepsilon_q f(x) + q((1 - q)^2x^2 + 1)f(x) = 0$
$\text{S}_q(x)$ bzw. $\text{C}_q(x)$	$(q^2(1 - q)^2x^2 + 1)\varepsilon_q^2 f(x) - (q + 1)\varepsilon_q f(x) + qf(x) = 0$

Tabelle 3.2 q-Holonome Rekursionsgleichungen q-trigonometrischer Funktionen

Wir leiten exemplarisch eine der q-holonomen Rekursionsgleichungen der Tabelle 3.2 her. Die Vorgehensweise werden wir im nächsten Abschnitt als Algorithmus formulieren, mit dem es uns möglich sein wird, q-holonome Rekursionsgleichungen für beliebige q-Funktionen aufzustellen, von denen wir wissen, wie sie sich unter dem q-Shift verhalten. Betrachten wir die q-trigonometrische Funktion $f(x) = \sin_q(x)$. Die kleine q-Sinusfunktion nach Gasper/Rahman erfüllt keine q-holonome Rekursionsgleichung erster Ordnung, da $\frac{\varepsilon_q f(x)}{f(x)} = 1 - x\frac{\cos_q(x)}{\sin_q(x)}$ aufgrund der unendlich vielen Polstellen keine rationale Funktion in x sein kann. Zusammen mit den q-Shifts

$$\varepsilon_q \sin_q(x) = \sin_q(x) - x\cos_q(x) \quad \text{und} \quad \varepsilon_q^2 \sin_q(x) = (1 - qx^2)\sin_q(x) - (1 + q)x\cos_q(x)$$

und dem Ansatz

$$\varepsilon_q^2 f(x) + A_1(x)\varepsilon_q f(x) + A_0(x)f(x) = 0$$

erhalten wir durch Umordnung des Ansatzes in eine Summe, deren Summanden linear unabhängig über $\mathbb{F}(x)$ sind

$$\left(A_1(x) + A_0(x) + (1 - qx^2)\right) \sin_q(x) - (x\,A_1(x) + (1+q)x) \cos_q(x) = 0$$

und somit durch Koeffizientenvergleich ein lineares Gleichungssystem über $\mathbb{F}(x)$ mit der Lösung $A_1(x) = -(q+1)$ und $A_0(x) = q(x^2+1)$. Setzen wir die Lösung in den Ansatz ein, so ergibt sich die q-holonome Rekursionsgleichung für $\sin_q(x)$ aus Tabelle 3.2.

3.1.3 q-Holonome Rekursionsgleichungen q-orthogonaler Polynome

Die klassischen q-orthogonalen Polynome sind definitionsgemäß q-holonom, da sie eine q-holonome Differential- und somit auch eine q-holonome Rekursionsgleichung zweiter Ordnung erfüllen. Wir listen einige Rekursionsgleichungen klassischer q-orthogonaler Polynome auf. Die Darstellungen erhält man durch Transformation der q-Differentialgleichungen aus Tabelle 1.1 in q-Rekursionsgleichungen unter Zuhilfenahme von (1.12).

$f(x)$	q-holonome Rekursionsgleichung für $f(x)$
$\tilde{\mathrm{h}}_n(x;q)$	$(q^2x^2+1)\varepsilon_q^2 f(x) - (q(q^{n+1}x^2+1)+1)\varepsilon_q f(x) + qf(x) = 0$
$\mathrm{V}_n^{(a)}(x;q)$	$(qx-a)(qx-1)\varepsilon_q^2 f(x) - (q^{n+2}x^2 - q(a+1)x + a(q+1))\varepsilon_q f(x) + aqf(x) = 0$
$\mathrm{C}_n(x;a;q)$	$(qx-1)\varepsilon_q^2 f(x) - (q^{n+1}x + a - 1)\varepsilon_q f(x) + af(x) = 0$
$\mathrm{L}_n^{(\alpha)}(x;q)$	$q^\alpha(qx+1)\varepsilon_q^2 f(x) - (q^\alpha(q^{n+1}x+1)+1)\varepsilon_q f(x) + f(x) = 0$
$\mathrm{S}_n(x;q)$	$qx\varepsilon_q^2 f(x) - (q^{n+1}x+1)\varepsilon_q f(x) + f(x) = 0$
$\mathrm{M}_n(x;b,c;q)$	$(qx-1)(qx+bc)\varepsilon_q^2 f(x) -$ $(q^{n+2}x^2 + q((b+1)c-1)x - bc(q+1))\varepsilon_q f(x) + cq(x-b)f(x) = 0$
$\mathrm{p}_n(x;a\|q)$	$aq^n\varepsilon_q^2 f(x) + (qx - q^n(a+1))\varepsilon_q f(x) - q^n(qx-1)f(x) = 0$
$\mathrm{K}_n(x;a;q)$	$aq^{n+1}x\varepsilon_q^2 f(x) - (q(aq^{2n}-1)x + q^n)\varepsilon_q f(x) - q^n(qx-1)f(x) = 0$
$\mathrm{p}_n(x;a,b\|q)$	$aq^n(bq^2x-1)\varepsilon_q^2 f(x) - (q(abqq^{2n}+1)x - q^n(a+1))\varepsilon_q f(x) - q^n(qx-1)f(x) = 0$
$\mathrm{h}_n(x;q)$	$q^n\varepsilon_q^2 f(x) + (q^3x^2 - q^n(q+1))\varepsilon_q f(x) - q^{n+1}(qx-1)(qx+1)f(x) = 0$
$\mathrm{U}_n^{(a)}(x;q)$	$aq^n\varepsilon_q^2 f(x) - (q^3x^2 - q^{n+2}(a+1)x + aq^n(q+1))\varepsilon_q f(x) + q^{n+1}(qx-1)(qx-a)f(x) = 0$
$\mathrm{P}_n(x;a,b;q)$	$abq^n(qx-1)\varepsilon_q^2 f(x) +$ $(qx^2 - q^{n+1}(a(b+1)+b)x + abq^n(q+1))\varepsilon_q f(x) - q^{n+1}(x-a)(x-b)f(x) = 0$
$\mathrm{Q}_n(x;a,b;N\|q)$	$aq^n(qx-1)(q^{N+2}bx-1)\varepsilon_q^2 f(x) - (q^{N+2}(abqq^{2n}+1)x^2 -$ $q^{n+1}(a(q^{N+1}(b+1)+1)+1)x + aq^n(q+1))\varepsilon_q f(x) + q^{n+1}(x-a)(q^{N+1}x-1)f(x) = 0$
$\mathrm{P}_n(x;a,b,c;q)$	$aq^n(qx-1)(qbx-c)\varepsilon_q^2 f(x) - (q(abqq^{2n}+1)x^2 -$ $q^{n+1}((a+1)c + a(b+1))x + acq^n(q+1))\varepsilon_q f(x) + q^{n+1}(x-a)(x-c)f(x) = 0$

Tabelle 3.3 q-Holonome Rekursionsgleichungen einiger klassischer q-orthogonaler Polynome

3.1.4 q-Holonome Rekursionsgleichungen verallgemeinerter q-hypergeometrischer Funktionen

Ziel dieses Abschnitts ist es, eine q-holonome Rekursionsgleichung für die verallgemeinerte q-hypergeometrische Funktion zu bestimmen. Dazu benötigen wir die so genannten q-Shiftrelationen, die Beziehungen zwischen der q-hypergeometrischen Funktion ${}_r\phi_s$ und derselben Funktion, bei der ein oberer (bzw. unterer) Parameter mit q multipliziert worden ist, herstellen. Die Bestimmung einer allgemeinen holonomen Rekursionsgleichung (im Nicht-q-Fall) kann man in [Roa96] nachlesen. Wir adaptieren die Beweisidee und übertragen sie auf den q-Fall. Dazu betrachten wir

zunächst das folgende Lemma, mit dem wir eine einfache q-Shiftregel (nach Umstellen nach ε_q) für die verallgemeinerte q-hypergeometrische Funktion aufstellen können. Diese Regel ist in `qshift` und die entsprechende q-Ableitungsregel in `qdiff` aus dem Package `qFPS` implementiert.

Lemma 3.7 Für die verallgemeinerte q-hypergeometrische Funktion gelten die beiden *q-Shiftrelationen*

$$\frac{1-a_i\varepsilon_q}{1-a_i}\,{}_r\phi_s\left(\begin{matrix}a_1,\ldots,a_i,\ldots,a_r\\ \boldsymbol{b}\end{matrix}\,\middle|\,q;x\right)={}_r\phi_s\left(\begin{matrix}a_1,\ldots,qa_i,\ldots,a_r\\ \boldsymbol{b}\end{matrix}\,\middle|\,q;x\right) \tag{3.2}$$

und

$$\frac{1-b_j\varepsilon_q}{1-b_j}\,{}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ b_1,\ldots,qb_j,\ldots,b_s\end{matrix}\,\middle|\,q;x\right)={}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ b_1,\ldots,b_j,\ldots,b_s\end{matrix}\,\middle|\,q;x\right) \tag{3.3}$$

für $i=1,\ldots,r$ und $j=1,\ldots,s$.

Beweis Es gilt

$$\begin{aligned}
&\frac{1-a_i\varepsilon_q}{1-a_i}\,{}_r\phi_s\left(\begin{matrix}a_1,\ldots,a_i,\ldots,a_r\\ \boldsymbol{b}\end{matrix}\,\middle|\,q;x\right)\\
&=\frac{1-a_i\varepsilon_q}{1-a_i}\sum_{k=0}^{\infty}\frac{(a_1;q)_k\cdot\ldots\cdot(a_i;q)_k\cdot\ldots\cdot(a_r;q)_k}{(\boldsymbol{b};q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&=(1-a_i\varepsilon_q)\sum_{k=0}^{\infty}\frac{(a_1;q)_k\cdot\ldots\cdot(qa_i;q)_{k-1}\cdot\ldots\cdot(a_r;q)_k}{(\boldsymbol{b};q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&=\sum_{k=0}^{\infty}(1-a_iq^k)\frac{(a_1;q)_k\cdot\ldots\cdot(qa_i;q)_{k-1}\cdot\ldots\cdot(a_r;q)_k}{(\boldsymbol{b};q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&=\sum_{k=0}^{\infty}\frac{(a_1;q)_k\cdot\ldots\cdot(qa_i;q)_k\cdot\ldots\cdot(a_r;q)_k}{(\boldsymbol{b};q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&={}_r\phi_s\left(\begin{matrix}a_1,\ldots,qa_i,\ldots,a_r\\ \boldsymbol{b}\end{matrix}\,\middle|\,q;x\right)
\end{aligned}$$

und

$$\begin{aligned}
&\frac{1-b_j\varepsilon_q}{1-b_j}\,{}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ b_1,\ldots,qb_j,\ldots,b_s\end{matrix}\,\middle|\,q;x\right)\\
&=\frac{1-b_j\varepsilon_q}{1-b_j}\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_k}{(b_1;q)_k\cdot\ldots\cdot(qb_j;q)_k\cdot\ldots\cdot(b_s;q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&=(1-b_j\varepsilon_q)\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_k}{(b_1;q)_k\cdot\ldots\cdot(b_j;q)_{k+1}\cdot\ldots\cdot(b_s;q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&=\sum_{k=0}^{\infty}(1-b_jq^k)\frac{(\boldsymbol{a};q)_k}{(b_1;q)_k\cdot\ldots\cdot(b_j;q)_{k+1}\cdot\ldots\cdot(b_s;q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&=\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_k}{(b_1;q)_k\cdot\ldots\cdot(b_j;q)_k\cdot\ldots\cdot(b_s;q)_k}\frac{x^k}{(q;q)_k}\left((-1)^kq^{\binom{k}{2}}\right)^{1+s-r}\\
&={}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ b_1,\ldots,b_j,\ldots,b_s;\end{matrix}\,\middle|\,q;x\right).
\end{aligned}$$

□

Definition 3.8 Sei

$$A_i:=\frac{1-a_i\varepsilon_q}{1-a_i}\qquad\text{und}\qquad B_j:=\frac{1-q^{-1}b_j\varepsilon_q}{1-q^{-1}b_j}.$$

A_i ist der Operator, der den oberen Parameter a_i einer verallgemeinerten q-hypergeometrischen Funktion mit q multipliziert und B_j ist der Operator, der den unteren Parameter b_j einer verallgemeinerten q-hypergeometrischen Funktion mit q^{-1} multipliziert.

Korollar 3.9 Es gelten die Gleichungen

$$\left(\prod_{i=1}^{r} A_i\right) {}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ \boldsymbol{b}\end{matrix}\,\middle|\, q;x\right) = {}_r\phi_s\left(\begin{matrix}q\boldsymbol{a}\\ \boldsymbol{b}\end{matrix}\,\middle|\, q;x\right) \tag{3.4}$$

und

$$\left(\prod_{j=1}^{s} B_j\right) {}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ \boldsymbol{b}\end{matrix}\,\middle|\, q;x\right) = {}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ q^{-1}\boldsymbol{b}\end{matrix}\,\middle|\, q;x\right). \tag{3.5}$$

Beweis Die Behauptung folgt durch r- bzw. s-maliges Anwenden ($i = 1,\dots,r$ bzw. $j = 1,\dots,s$) von (3.2) bzw. (3.3), wobei man im zweiten Fall b_j jeweils durch $q^{-1}b_j$ ersetzt. □

Satz 3.10 Es gilt die folgende Beziehung zwischen den aufwärts q-geshifteten q-hypergeometrischen Funktionen und den abwärts q-geshifteten q-hypergeometrischen Funktionen

$$(-1)^r(1-\varepsilon_q)\varepsilon_q^{r-m}\,{}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ q^{-1}\boldsymbol{b}\end{matrix}\,\middle|\, q;x\right) = (-1)^{s+1}q^{r-m}x\varepsilon_q^{1+s-m}\frac{\prod_{i=1}^{r}(1-a_i)}{\prod_{j=1}^{s}(1-q^{-1}b_j)}\,{}_r\phi_s\left(\begin{matrix}q\boldsymbol{a}\\ \boldsymbol{b}\end{matrix}\,\middle|\, q;x\right), \tag{3.6}$$

wobei $m = \min(r, s+1)$ ist.

Beweis

$$\begin{aligned}
&(-1)^r(1-\varepsilon_q)\varepsilon_q^{r-m}\,{}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ q^{-1}\boldsymbol{b}\end{matrix}\,\middle|\, q;x\right)\\
&= (-1)^r(1-\varepsilon_q)\varepsilon_q^{r-m}\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_k}{(q^{-1}\boldsymbol{b};q)_k}\frac{x^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^r(1-\varepsilon_q)\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_k}{(q^{-1}\boldsymbol{b};q)_k}\frac{q^{(r-m)k}x^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^r\sum_{k=0}^{\infty}(1-q^k)\frac{(\boldsymbol{a};q)_k}{(q^{-1}\boldsymbol{b};q)_k}\frac{q^{(r-m)k}x^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^r\sum_{k=1}^{\infty}(1-q^k)\frac{(\boldsymbol{a};q)_k}{(q^{-1}\boldsymbol{b};q)_k}\frac{q^{(r-m)k}x^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^r\sum_{k=0}^{\infty}(1-q^{k+1})\frac{(\boldsymbol{a};q)_{k+1}}{(q^{-1}\boldsymbol{b};q)_{k+1}}\frac{q^{(r-m)(k+1)}x^{k+1}}{(q;q)_{k+1}}\left((-1)^{k+1} q^{\binom{k+1}{2}}\right)^{1+s-r}\\
&= (-1)^r q^{r-m}x\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_{k+1}}{(q^{-1}\boldsymbol{b};q)_{k+1}}\frac{q^{(r-m)k}x^k}{(q;q)_k}\left((-1)^{k+1} q^{\binom{k}{2}+k}\right)^{1+s-r}\\
&= q^{r-m}x\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_{k+1}}{(q^{-1}\boldsymbol{b};q)_{k+1}}\frac{x^k}{(q;q)_k}(-1)^r\left(q^k\right)^{r-m}(-1)^{1+s-r}\left(q^k\right)^{1+s-r}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^{s+1}q^{r-m}x\sum_{k=0}^{\infty}\frac{(\boldsymbol{a};q)_{k+1}}{(q^{-1}\boldsymbol{b};q)_{k+1}}\frac{q^{(1+s-m)k}x^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^{s+1}\,q^{r-m}x\sum_{k=0}^{\infty}\frac{\prod_{i=1}^{r}(1-a_i)}{\prod_{j=1}^{s}(1-q^{-1}b_j)}\frac{(q\boldsymbol{a};q)_k}{(\boldsymbol{b};q)_k}\frac{(q^{1+s-m}x)^k}{(q;q)_k}\left((-1)^k q^{\binom{k}{2}}\right)^{1+s-r}\\
&= (-1)^{s+1}q^{r-m}x\varepsilon_q^{1+s-m}\frac{\prod_{i=1}^{r}(1-a_i)}{\prod_{j=1}^{s}(1-q^{-1}b_j)}\,{}_r\phi_s\left(\begin{matrix}q\boldsymbol{a}\\ \boldsymbol{b}\end{matrix}\,\middle|\, q;x\right)
\end{aligned}$$

□

Mit Hilfe von Satz 3.10 können wir nun eine explizite Formel für die q-holonome Rekursionsgleichung der verallgemeinerten q-hypergeometrischen Funktion angeben.

Satz 3.11 Die verallgemeinerte q-hypergeometrische Funktion

$$ {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) $$

erfüllt die q-holonome Rekursionsgleichung

$$ \left((-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} \left(\prod_{i=1}^{r} (1 - a_i \varepsilon_q) \right) + (-1)^r (\varepsilon_q - 1) \varepsilon_q^{r-m} \left(\prod_{j=1}^{s} (1 - q^{-1} b_j \varepsilon_q) \right) \right) f(x) = 0 \quad (3.7) $$

der Ordnung $\max(r, s+1)$, wobei $m = \min(r, s+1)$ ist.

Beweis Satz 3.10 besagt, dass

$$ (-1)^r (1 - \varepsilon_q) \varepsilon_q^{r-m} {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ q^{-1}\boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) = (-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} \frac{\prod_{i=1}^{r} (1 - a_i)}{\prod_{j=1}^{s} (1 - q^{-1} b_j)} {}_r\phi_s\left(\begin{matrix} q\boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) $$

gilt. Daraus folgt durch Einsetzen der q-Shiftrelationen (3.4) und (3.5)

$$ \begin{aligned} &(-1)^r (1 - \varepsilon_q) \varepsilon_q^{r-m} \left(\prod_{j=1}^{s} B_j \right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) \\ &= (-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} \frac{\prod_{i=1}^{r} (1 - a_i)}{\prod_{j=1}^{s} (1 - q^{-1} b_j)} \left(\prod_{i=1}^{r} A_i \right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right). \end{aligned} $$

Nach Definition der Operatoren A_i und B_j erhält man

$$ \begin{aligned} &(-1)^r (1 - \varepsilon_q) \varepsilon_q^{r-m} \left(\prod_{j=1}^{s} \frac{1 - q^{-1} b_j \varepsilon_q}{1 - q^{-1} b_j} \right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) \\ &= (-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} \frac{\prod_{i=1}^{r} (1 - a_i)}{\prod_{j=1}^{s} (1 - q^{-1} b_j)} \left(\prod_{i=1}^{r} \frac{1 - a_i \varepsilon_q}{1 - a_i} \right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) \end{aligned} $$

und somit durch Umstellen der Gleichung und anschließendem Kürzen

$$ \left((-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} \left(\prod_{i=1}^{r} (1 - a_i \varepsilon_q) \right) + (-1)^r (\varepsilon_q - 1) \varepsilon_q^{r-m} \left(\prod_{j=1}^{s} (1 - q^{-1} b_j \varepsilon_q) \right) \right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x \right) = 0. $$

Multiplizieren wir den q-Rekursionsoperator der linken Seite aus, so ergibt sich

$$ \begin{aligned} &(-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} (-1)^r \left(\prod_{i=1}^{r} a_i \right) \varepsilon_q^r + (-1)^r \varepsilon_q^{r-m+1} (-1)^s \left(\prod_{j=1}^{s} q^{-1} b_j \right) \varepsilon_q^s + \ldots + \\ &(-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} + (-1)^r (-1) \varepsilon_q^{r-m} \\ &= (-1)^{1+s+r} \left(q^{r-m} x \left(\prod_{i=1}^{r} a_i \right) - q^{-s} \left(\prod_{j=1}^{s} b_j \right) \right) \varepsilon_q^{1+s+r-m} + \ldots + \\ &(-1)^{s+1} q^{r-m} x \varepsilon_q^{1+s-m} + (-1)^{r+1} \varepsilon_q^{r-m}. \end{aligned} $$

Der höchste Koeffizient kann nur dann verschwinden, wenn mindestens einer der oberen und einer der unteren Parameter der q-hypergeometrischen Funktion Null wird. Dieser Fall ist aber ausgeschlossen, da wir annehmen können, dass kein oberer Parameter identisch einem der unteren Parameter ist. Gelte nun $r > s+1$, dann ist $(-1)^{s+1} q^{r-s-1} x$ das nichtverschwindende Absolutglied und somit beträgt die Ordnung der q-holonomen Rekursionsgleichung $1 + s + r - m = r$. Für $r < s+1$ erhält man wegen dem Absolutglied $(-1)^{r+1}$ die Ordnung $1 + s + r - m = s + 1$. Der Fall $r = s + 1$ liefert $(-1)^r (x - 1)$ als Fußkoeffizienten der Rekursion und somit die Ordnung r. Zusammenfassend ist die Ordnung der q-holonomen Rekursionsgleichung also gegeben durch das Maximum der beiden Zahlen r und $s + 1$. □

Als Nebenprodukt des vorigen Satzes formulieren wir

Satz 3.12 Es existieren $\alpha_k(x), \beta_k(x) \in \mathbb{F}[x]$, so dass die beiden *q-Nachbarschaftsrelationen*

$$\left(\sum_{k=0}^{d} \alpha_k(x) A_i^k\right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x\right) = 0 \quad \text{und} \quad \left(\sum_{k=0}^{d} \beta_k(x) B_j^k\right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x\right) = 0$$

für $i = 1, \ldots, r$ und $j = 1, \ldots, s$ gültig sind, wobei $d = \max(r, s+1)$ und

$$A_i^k = \frac{1 - q^{k-1} a_i \varepsilon_q}{1 - q^{k-1} a_i} \cdot \ldots \cdot \frac{1 - q a_i \varepsilon_q}{1 - q a_i} \cdot \frac{1 - a_i \varepsilon_q}{1 - a_i} \quad \text{mit} \quad a_i \notin \{1, q^{-1}, q^{-2}, \ldots, q^{-(k-1)}\}$$

und

$$B_j^k = \frac{1 - q^{-k} b_j \varepsilon_q}{1 - q^{-k} b_j} \cdot \ldots \cdot \frac{1 - q^{-2} b_j \varepsilon_q}{1 - q^{-2} b_j} \cdot \frac{1 - q^{-1} b_j \varepsilon_q}{1 - q^{-1} b_j} \quad \text{mit} \quad b_j \notin \{q, q^2, q^3, \ldots, q^k\}.$$

Ferner gilt

$$A_i^{-1} = -\sum_{k=0}^{d-1} \frac{\alpha_{k+1}(x)}{\alpha_0(x)} A_i^k \quad \text{bzw.} \quad B_j^{-1} = -\sum_{k=0}^{d-1} \frac{\beta_{k+1}(x)}{\beta_0(x)} B_j^k,$$

für alle i mit $a_i \neq q^{-1} b_j$ für alle $j = 1, \ldots, s$ bzw. für alle j mit $b_j \neq q a_i$ für alle $i = 1, \ldots, r$.

Beweis Sei im Folgenden $m = \min(r, s+1)$. Substituieren wir $\varepsilon_q = \frac{a_i - 1}{a_i} A_i + \frac{1}{a_i}$ in den zugehörigen Operator der q-holonomen Rekursionsgleichung (3.7), so erhalten wir ein Polynom in A_i mit

$$\left(\sum_{k=0}^{d} \alpha_k(x) A_i^k\right) {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; x\right) = 0 \quad \text{und} \quad \alpha_k \in \mathbb{F}[x]. \tag{3.8}$$

Indem wir diese Gleichung nach dem Fußkoeffizienten umstellen und durch $\alpha_0(x)$ teilen, können wir den Identitätsoperator auch schreiben als

$$\mathrm{id} = -\sum_{k=1}^{d} \frac{\alpha_k(x)}{\alpha_0(x)} A_i^k.$$

Multiplizieren wir diesen Operator von rechts mit A_i^{-1}, bekommen wir eine Formel der Form

$$A_i^{-1} = -\sum_{k=0}^{d-1} \frac{\alpha_{k+1}(x)}{\alpha_0(x)} A_i^k.$$

Für $\tau \in \mathbb{F}$ gilt

$$1 - \tau\varepsilon_q = 1 - \tau\left(\frac{a_i - 1}{a_i} A_i + \frac{1}{a_i}\right) = \left(\frac{1 - a_i}{a_i}\tau\right) A_i + \left(1 - \frac{\tau}{a_i}\right).$$

Wir möchten nun das Absolutglied $\alpha_0(x)$ des Polynoms aus (3.8) bestimmen. Der erste Summand von (3.7) liefert keinen Beitrag, da dieser wegen $1 - a_i\varepsilon_q = (1 - a_i) A_i$ nur aus A_i-Potenzen besteht. Folglich gilt

$$\alpha_0(x) = (-1)^r \left(\frac{1}{a_i} - 1\right) a_i^{m-r} \prod_{j=1}^{s} \left(1 - \frac{q^{-1} b_j}{a_i}\right) = (-1)^{r+1} \frac{a_i - 1}{a_i^{1+s+r-m} q^s} \prod_{j=1}^{s} (q a_i - b_j).$$

Der Operator A_i^{-1} ist demnach für alle i definiert, für die $a_i \neq q^{-1} b_j$ für alle $j = 1, \ldots, s$ und für die $a_i \notin \{1, q^{-1}, q^{-2}, \ldots, q^{-(d-2)}\}$ gilt (siehe Nenner von A_i^{d-1}). Analog geht man für B_j und B_j^{-1} vor. Für $\beta_0(x)$ erhält man wegen

$$1 - \tau\varepsilon_q = 1 - \tau\left(\frac{q^{-1} b_j - 1}{q^{-1} b_j} B_j + \frac{1}{q^{-1} b_j}\right) = \left(\frac{1 - q^{-1} b_j}{q^{-1} b_j}\tau\right) B_j + \left(1 - \frac{q\tau}{b_j}\right)$$

die Formel

$$\beta_0(x) = (-1)^{s+1} q^{r-m} x \left(\frac{q}{b_j}\right)^{1+s-m} \prod_{i=1}^{r} \left(1 - \frac{q a_i}{b_j}\right) = (-1)^{s+1} q^{1+s+r-2m} b_j^{m-r-s-1} x \prod_{i=1}^{r} (b_j - q a_i).$$

B_j^{-1} ist somit definiert, wenn $b_j \neq q a_i$ für alle $i = 1, \ldots, r$ und $b_j \notin \{q, q^2, q^3, \ldots, q^{d-1}\}$. □

Wir werden später in Abschnitt 5.1.2 auch q-holonome Rekursionsgleichungen für Funktionen ${}_r\phi_s(x)$ bestimmen können, bei denen ausschließlich bzw. zusätzlich einer der oberen Parameter die Variable x enthält.

3.2 Algorithmen zur Bestimmung q-holonomer Rekursionsgleichungen

Wir haben im letzten Kapitel q-Ableitungen und q-Shifts für alle in dieser Arbeit erwähnten q-Funktionen bestimmt. Mit diesen Informationen können wir nun einen allgemeinen Algorithmus formulieren, der aus Kenntnis der q-Shifts $\varepsilon_q f(x), \varepsilon_q^2 f(x), \varepsilon_q^3 f(x), \ldots$ einer beliebigen q-Funktion $f(x)$ eine q-holonome Rekursionsgleichung bzw. aus Kenntnis der q-Ableitungen $D_q f(x), D_q^2 f(x), D_q^3 f(x), \ldots$ eine q-holonome Differentialgleichung minimaler Ordnung bestimmt. Alle q-holonome Rekursionsgleichungen der letzten Abschnitte (bis auf die q-holonome Rekursionsgleichung der verallgemeinerten q-hypergeometrischen Funktion, bei der man die Rekursion explizit angeben kann) können wir so algorithmisch herleiten. Ferner sind wir mit dem Algorithmus in der Lage, zu beliebigen Summen und Produkten q-holonomer Funktionen die zugehörige q-holonome Rekursionsgleichung minimaler Ordnung zu bestimmen. Der angesprochene Algorithmus ist das q-Analogon des Algorithmus aus [Koe92] bzw. [KS94].

3.2.1 Algorithmus zur Bestimmung q-holonomer Rekursionsgleichungen aus q-Shifts

Der nachstehende Algorithmus bestimmt aus einer q-holonomen Funktion eine q-holonome Rekursionsgleichung minimaler Ordnung. Wir schließen im Folgenden rationale Funktionen als Eingabe aus[3].

Algorithmus 3.1 Bestimmung einer q-holonomen Rekursionsgleichung für eine Funktion $f(x)$ aus deren q-Shifts (`qHolonomicRE`)

Eingabe : Eine Funktion $f(x)$ und eine obere Schranke *MAXREORDER* für die Ordnung der q-holonomen Rekursionsgleichung

Ausgabe : Eine q-holonome Rekursionsgleichung für $f(x)$ minimaler Ordnung (falls eine existiert)

begin
  $n \leftarrow 1$
  while $n \leq$ *MAXREORDER* **do**
    Ansatz $\leftarrow \varepsilon_q^n F(x) + A_{n-1}(x)\varepsilon_q^{n-1} F(x) + \ldots + A_0(x)F(x) = 0$
    RE $\leftarrow \varepsilon_q^n f(x) + A_{n-1}(x)\varepsilon_q^{n-1} f(x) + \ldots + A_0(x)f(x)$
    Zerlege *RE* in eine Summe linear unabhängiger Terme über $\mathbb{F}(x)$.
    if *Anzahl der linear unabhängigen Terme* $= n$ **then**
      Betrachte die Koeffizienten der linear unabhängigen Terme und setze diese Null.
      Löse das resultierende lineare Gleichungssystem mit n Gleichungen nach den n Unbekannten $A_0(x), \ldots, A_{n-1}(x)$ über $\mathbb{F}(x)$.
      Setze die Lösung in *Ansatz* ein und multipliziere mit $\operatorname{lcm}(\operatorname{denom}(A_0(x)), \ldots, \operatorname{denom}(A_{n-1}(x)))$.
      return *die so erhaltene q-holonome Rekursionsgleichung*
    end
    $n \leftarrow n + 1$
  end
  return *„Es existiert keine q-holonome Rekursion mit Ordnung ≤ MAXREORDER.“*
end

[3] Rationale Funktionen und insbesondere Polynome sind q-hypergeometrisch und somit q-holonom vom Grad 1. Die zugehörige q-holonome Rekursionsgleichung lässt sich leicht aufstellen.

Beweis Sei $f(x)$ eine Funktion in x. Betrachten wir zunächst $n = 1$. Dann gibt es mindestens einen linear unabhängigen Term in $RE = \varepsilon_q f(x) + A_0(x)f(x)$ über $\mathbb{F}(x)$, nämlich $f(x)$ selbst. Gibt es keinen weiteren linear unabhängigen Term, so lässt sich die Gleichung $RE = 0$ eindeutig lösen. Die Lösung lautet $A_0(x) = -\frac{\varepsilon_q f(x)}{f(x)}$. Andernfalls gibt es keine Lösung des linearen Gleichungssystems und wir erhöhen n um 1. Angenommen wir sind im n-ten Schritt des obigen Algorithmus und die Summe $\varepsilon_q^k f(x) + A_{k-1}(x)\varepsilon_q^{k-1} f(x) + \ldots + A_0(x)f(x)$ besaß für alle $k < n$ mehr als k linear unabhängige Terme über $\mathbb{F}(x)$ (und somit auch keine Lösung des durch Koeffizientenvergleich erhaltenen linearen Gleichungssystems), dann gibt es wiederum zwei mögliche Fälle für $RE = \varepsilon_q^n f(x) + A_{n-1}(x)\varepsilon_q^{n-1} f(x) + \ldots + A_0(x)f(x)$

1. Die Anzahl der linear unabhängigen Terme von RE über $\mathbb{F}(x)$ ist genau n.
2. Die Anzahl der linear unabhängigen Terme von RE über $\mathbb{F}(x)$ ist größer als n.

Im ersten Fall gibt es eine eindeutige Lösung des linearen Gleichungssystems, denn gäbe es zwei Lösungen

$$\varepsilon_q^n f(x) + B_{n-1}(x)\varepsilon_q^{n-1} f(x) + \ldots + B_0(x)f(x) = 0$$

und

$$\varepsilon_q^n f(x) + C_{n-1}(x)\varepsilon_q^{n-1} f(x) + \ldots + C_0(x)f(x) = 0,$$

so würde die Differenz

$$(B_{n-1}(x) - C_{n-1}(x))\varepsilon_q^{n-1} f(x) + \ldots + (B_0(x) - C_0(x))f(x) = 0 \tag{3.9}$$

eine q-holonome Rekursionsgleichung der Ordnung $n-1$ darstellen. Das ist aber nicht möglich, da keine Lösung des linearen Gleichungssystems für $k < n$ existiert (n ist minimal). Folglich müssen die Koeffizienten von (3.9) verschwinden und die obigen Rekursionsgleichungen gleich sein ($B_i(x) = C_i(x)$ für $i = 0, \ldots, n-1$). Im zweiten Fall haben wir mehr Gleichungen als Unbekannte im auftretenden linearen Gleichungssystem. Somit kann es keine Lösung geben und wir inkrementieren n. Die obere Schranke *MAXREORDER* ist notwendig, falls die Eingabe eine nicht q-holonome Funktion ist, da in diesem Fall der Algorithmus nicht abbrechen würde. □

Den Algorithmus kann man analog für q-holonome Differentialgleichungen formulieren.

Bemerkung 3.13 Der schwierigste Teil des Algorithmus ist in Zeile 6 zu finden. An dieser Stelle muss man RE in eine Summe linear unabhängiger Terme über $\mathbb{F}(x)$ zerlegen. Aus diesem Grund haben wir alle Darstellungen von q-Shifts im vorangegangenen Kapitel so gewählt, dass diese (und somit auch Darstellungen höherer q-Shifts) Linearkombinationen von bekannten linear unabhängigen q-Funktionen über $\mathbb{F}(x)$ sind. Dadurch kann man Zeile 6 einfach realisieren. Allerdings ergibt sich bei den klassischen q-orthogonalen Polynomen $P_n(x)$ ein Problem. Wie kann man dort die lineare Abhängigkeit überprüfen? Die Antwort liegt in der für jedes klassische q-orthogonale Polynom existierende Drei-Term-Rekursion (1.16). Durch sukzessive Anwendung der q-Shiftregel (2.3) des entsprechenden q-orthogonalen Polynoms erhält man nach einmaliger Wiederholung der Schleife des Algorithmus folgende Zerlegung für RE

$$\varepsilon_q^2 P_n(x) + A_1(x)\varepsilon_q P_n(x) + A_0(x)P_n(x) = \tilde{A}_0(x)P_n(x) + \tilde{A}_1(x)P_{n-1}(x) + \tilde{A}_2(x)P_{n-2}(x) \tag{3.10}$$

mit $\tilde{A}_i(x) \in \mathbb{F}(x)$. Verwendet man nun die Drei-Term-Rekursion von $P_n(x)$, so kann man den ersten Summanden von (3.10) durch $P_{n-1}(x)$ und $P_{n-2}(x)$ ausdrücken, so dass sich die Anzahl der linear unabhängigen Termen über $\mathbb{F}(x)$ auf zwei Terme reduziert. Folglich erhält man auf diesem Weg die q-holonome Rekursionsgleichung zweiter Ordnung für $P_n(x)$. Diese Vorgehensweise ist in einer zusätzlichen Prozedur implementiert, die im Algorithmus aufgerufen wird. Die Minimalität der Ordnung der q-holonomen Rekursionsgleichung ist nur dann gewährleistet, wenn das Computeralgebrasystem auch alle linearen Abhängigkeiten erkennt. Für alle in dieser Dissertation behandelten q-Funktionen haben wir sowohl die q-Shifts als auch die q-Shiftregeln der q-orthogonalen Polynome so gewählt, dass die Rekursionen minimale Ordnung besitzen.

Wir sehen nun die Prozeduren `qHolonomicRE` und `qHolonomicDE` aus dem Package `qFPS` in Aktion, die zu einer q-holonomen Funktion die zugehörige q-holonome Rekursionsgleichung und Differentialgleichung bestimmen.

Maple-Sitzung 3.2 (q-Holonome Rekursions- und Differentialgleichungen)

q-Holonome Rekursionsgleichungen

```
> qHolonomicRE(qexp(x^2,q),F(x));
```

$$-(-1+x)(x+1)\left(-1+qx^2\right)F(x)+Sq_x(F(x))=0$$

```
> qHolonomicRE(qsin(x,q)+qcos(x,q),F(x));
```

$$q\left(1+x^2\right)F(x)+(-q-1)\,Sq_x(F(x))+\left(Sq_{x,x}\right)(F(x))=0$$

```
> qHolonomicRE(qexp(x,q)*qsin(x,q),F(x));
```

$$q(-1+x)\left(1+x^2\right)(-1+qx)F(x)+(1+q)(-1+qx)\,Sq_x(F(x))+\left(Sq_{x,x}\right)(F(x))=0$$

```
> qHolonomicRE(qExp(x,q)*qLaguerre(n,alpha,x,q),F(x));
```

$$F(x)-(x+1)\left(qxq^nq^\alpha+1+q^\alpha\right)Sq_x(F(x))+(x+1)\,q^\alpha(qx+1)^2\left(Sq_{x,x}\right)(F(x))=0$$

```
> qHolonomicRE(qphihypergeom([a,b],[c],x,q),F(x));
```

$$q(-1+x)F(x)+(-xqb-xqa+q+c)\,Sq_x(F(x))+(abxq-c)\left(Sq_{x,x}\right)(F(x))=0$$

Trotz einfacher Eingaben können die Rekursionsgleichungen sehr groß werden. Die folgende Rekursion füllt mehrere DIN A4-Seiten.

```
> qHolonomicRE((qexp(x^2,q)+qcos(x,q))*qsin(x,q),F(x),MAXREORDER=5):
```

q-Holonome Rekursionsgleichungen mit q-Shifts bzgl. q^{-1}

```
> qHolonomicRE(qsin(x,q),F(x),var=1/q);
```

$$F(x)\,q^3-q^3(1+q)\,Sq_x(F(x))+\left(q^4+x^2\right)\left(Sq_{x,x}\right)(F(x))=0$$

```
> qHolonomicRE(qexp(x,q)*qSin(x,q),F(x),var=1/q);
```

$$\left(x^2+q^2\right)F(x)-q(1+q)(q-x)\,Sq_x(F(x))+\left(q^2-x\right)(q-x)\left(Sq_{x,x}\right)(F(x))=0$$

q-Holonome Differentialgleichungen

```
> qHolonomicDE(qsin(x,q),F(x));
```

$$F(x)+(q-1)^2\left(Dq_{x,x}\right)(F(x))=0$$

```
> qHolonomicDE(DiscreteqHermiteI(n,x,q),F(x));
```

$$-(-1+q^n)\,q^2F(x)+x(q-1)\,q^2Dq_x(F(x))+q^n(q-1)^2\left(Dq_{x,x}\right)(F(x))=0$$

Die Prozedur `qHolonomicRE` wird in Kapitel 6 ausführlich beschrieben.

3.2.2 Der Summen-, Produkt- und Kompositionsalgorithmus

In [KRM07] wurden Algorithmen vorgestellt, mit denen man aus zwei q-holonomen Differentialgleichungen für die Funktionen $f(x)$ und $g(x)$ eine q-holonome Differentialgleichung für $f(x)+g(x)$ und $f(x)\cdot g(x)$ bestimmen kann. Ein weiterer Algorithmus bestimmt aus einer q-holonomen Differentialgleichung für $f(x)$ eine q-holonome Differentialgleichung für $(f\circ g)(x)$, sofern $g(qx)=q^bg(x)$ mit $b\in\mathbb{N}$ gilt. Diese Algorithmen werden nun für q-holonome Rekursionsgleichungen präsentiert. Durch Betrachtung des q-Shiftoperators und Ausnutzung dessen Eigenschaften sind diese Algorithmen wesentlich effizienter und einfacher. Besonders drückt sich das beim Produktalgorithmus aus, da man dort nicht die q-Leibnizregel für den q-Hahnoperator verwenden muss, sondern die Linearität bzgl. der Multiplikation benutzt.

Wir betrachten zunächst den Summenalgorithmus und beschreiben das Verfahren in

Algorithmus 3.2 Bestimmung einer q-holonomen Rekursionsgleichung, die als Lösung die Summe der Lösungen zweier q-holonomer Rekursionsgleichungen besitzt (`qSumRE`)

Eingabe : Zwei q-holonome Rekursionsgleichungen $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ der Ordnung n für $f(x)$ und $\sum_{k=0}^{m} b_k(x)\varepsilon_q^k g(x) = 0$ der Ordnung m für $g(x)$

Ausgabe : Eine q-holonome Rekursionsgleichung, die gültig für die Summe $f(x) + g(x)$ ist und höchstens die Ordnung $n + m$ besitzt

```
1  begin
2      for i ← max(n, m) to n + m do
3          Ansatz ← ∑_{k=0}^{i} A_k(x) ε_q^k F(x) = 0
4          RE ← ∑_{k=0}^{i} A_k(x) ε_q^k (f(x) + g(x))
5          RE ← Reduziere RE durch rekursive Anwendung von ε_q^n f(x) → − ∑_{k=0}^{n−1} (a_k(x)/a_n(x)) ε_q^k f(x)
6          RE ← Reduziere RE durch rekursive Anwendung von ε_q^m g(x) → − ∑_{k=0}^{m−1} (b_k(x)/b_m(x)) ε_q^k g(x)
7          Setze die Koeffizienten von f(x), ..., ε_q^{n−1} f(x) und g(x), ..., ε_q^{m−1} g(x) von RE Null.
8          Löse das resultierende lineare Gleichungssystem mit n + m Gleichungen nach den
           i + 1 Unbekannten A_0(x), ..., A_i(x) über F(x).
9          if eine Lösung existiert then
10             Setze die Lösung in Ansatz ein und multipliziere mit
               lcm(denom(A_0(x)), ..., denom(A_i(x)))
11             return die so erhaltene q-holonome Rekursionsgleichung
12         end
13     end
14 end
```

Beweis Zunächst einmal ist zu bemerken, dass man aus den beiden gegebenen q-holonomen Rekursionsgleichungen keine Rekursion kleinerer Ordnung erhalten kann. Folglich setzen wir bei der Ordnung $i = \max(m, n)$ an. Aus den beiden gegebenen q-holonomen Rekursionsgleichungen erhält man nach Umstellen die Gleichungen

$$\varepsilon_q^n f(x) = -\sum_{k=0}^{n-1} \frac{a_k(x)}{a_n(x)} \varepsilon_q^k f(x) \quad \text{und} \quad \varepsilon_q^m g(x) = -\sum_{k=0}^{m-1} \frac{b_k(x)}{b_n(x)} \varepsilon_q^k g(x).$$

Wendet man diese beiden Gleichungen rekursiv auf

$$\begin{aligned} L\left(f(x) + g(x)\right) := \sum_{k=0}^{i} A_k(x)\varepsilon_q^k \left(f(x) + g(x)\right) &= \sum_{k=0}^{i} A_k(x) \left(\varepsilon_q^k f(x) + \varepsilon_q^k g(x)\right) \\ &= \sum_{k=0}^{i} A_k(x)\varepsilon_q^k f(x) + \sum_{k=0}^{i} A_k(x)\varepsilon_q^k g(x) \end{aligned}$$

an, so kann man jeden Ausdruck der Form $\varepsilon_q^r f(x)$ mit $r \geq n$ bzw. $\varepsilon_q^s g(x)$ mit $s \geq m$ als Linearkombination von $f(x), \ldots, \varepsilon_q^{n-1} f(x)$ bzw. $g(x), \ldots, \varepsilon_q^{m-1} g(x)$ darstellen und erhält

$$L\left(f(x) + g(x)\right) = \sum_{k=0}^{n-1} B_k(x)\varepsilon_q^k f(x) + \sum_{k=0}^{m-1} C_k(x)\varepsilon_q^k g(x)$$

mit $B_k(x), C_k(x) \in \mathbb{F}(x)$. Betrachtet man nun $L\left(f(x) + g(x)\right) = 0$, so erhält man durch Koeffizientenvergleich ein homogenes lineares Gleichungssystem mit $n + m$ Gleichungen und $i + 1$ Unbekannten $A_0(x), \ldots, A_i(x)$ über $\mathbb{F}(x)$. Dieses Gleichungssystem besitzt spätestens für $i = n+m$ eine nichttriviale Lösung, so dass die Ordnung der gesuchten Rekursionsgleichung höchstens $n + m$ beträgt. □

Bemerkung 3.14 Der Summenalgorithmus bestimmt i.Allg. keine q-holonome Rekursionsgleichungen minimaler Ordnung (siehe nächste Maple-Sitzung). Evtl. lineare Abhängigkeiten der Funktionen $f(x)$ und $g(x)$ gehen verloren. Für q-holonome Differentialgleichungen funktioniert der Algorithmus 3.2 analog. Man verwendet an Stelle des q-Shiftoperators den Hahnoperator. Die Reduktionen, die durchgeführt werden, sind nichts anderes als Divisionen mit Rest durch den zugehörigen q-Rekursionsoperator der gegebenen q-holonomen Rekursionsgleichung.

Algorithmus 3.3 Bestimmung einer q-holonomen Rekursionsgleichung, die als Lösung das Produkt der Lösungen zweier q-holonomer Rekursionsgleichungen besitzt (`qProductRE`)

Eingabe : Zwei q-holonome Rekursionsgleichungen $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ der Ordnung n für $f(x)$ und $\sum_{k=0}^{m} b_k(x)\varepsilon_q^k g(x) = 0$ der Ordnung m für $g(x)$

Ausgabe : Eine q-holonome Rekursionsgleichung, die gültig für das Produkt $f(x)\cdot g(x)$ ist und höchstens die Ordnung $n\cdot m$ besitzt

begin
  for $i \leftarrow \max(n,m)$ **to** $n\cdot m$ **do**
    Ansatz $\leftarrow \sum_{k=0}^{i} A_k(x)\varepsilon_q^k F(x) = 0$
    RE $\leftarrow \sum_{k=0}^{i} A_k(x)\varepsilon_q^k(f(x)\cdot g(x))$
    RE $\leftarrow$ Reduziere *RE* durch rekursive Anwendung von $\varepsilon_q^n f(x) \rightarrow -\sum_{k=0}^{n-1} \frac{a_k(x)}{a_n(x)}\varepsilon_q^k f(x)$
    RE $\leftarrow$ Reduziere *RE* durch rekursive Anwendung von $\varepsilon_q^m g(x) \rightarrow -\sum_{k=0}^{m-1} \frac{b_k(x)}{b_m(x)}\varepsilon_q^k g(x)$
    Setze die Koeffizienten von $\varepsilon_q^r f(x)\cdot\varepsilon_q^s g(x)$ für $r = 0,\ldots,n-1$ und $s = 0,\ldots,m-1$ von *RE* Null.
    Löse das resultierende lineare Gleichungssystem mit $n\cdot m$ Gleichungen nach den $i+1$ Unbekannten $A_0(x),\ldots,A_i(x)$ über $\mathbb{F}(x)$.
    if *eine Lösung existiert* **then**
      Setze die Lösung in *Ansatz* ein und multipliziere mit $\operatorname{lcm}(\operatorname{denom}(A_0(x)),\ldots,\operatorname{denom}(A_i(x)))$
      return *die so erhaltene q-holonome Rekursionsgleichung*
    end
  end
end

Beweis Der Beweis verläuft analog zum Beweis von Algorithmus 3.2. Man betrachtet nun allerdings den Operator

$$L\left(f(x)\cdot g(x)\right) := \sum_{k=0}^{i} A_k(x)\varepsilon_q^k\left(f(x)\cdot g(x)\right) = \sum_{k=0}^{i} A_k(x)\left(\varepsilon_q^k f(x)\cdot\varepsilon_q^k g(x)\right),$$

welchen man auf

$$L\left(f(x)\cdot g(x)\right) = \sum_{r=0}^{n-1}\sum_{s=0}^{m-1} B_{rs}(x)\left(\varepsilon_q^r f(x)\cdot\varepsilon_q^s g(x)\right)$$

mit $B_{rs}(x)\in\mathbb{F}(x)$ reduzieren kann. Das homogene lineare Gleichungssystem, welches wieder aus Koeffizientenvergleich von

$$L\left(f(x)\cdot g(x)\right) = 0$$

entsteht, besitzt nun $n\cdot m$ Gleichungen und $i+1$ Unbekannte $A_0(x),\ldots,A_i(x)$ über $\mathbb{F}(x)$. Folglich erhält man eine nichttriviale Lösung spätestens bei $i = n\cdot m$. □

Bemerkung 3.15 Der obige Algorithmus bestimmt für zwei monische q-Rekursionsoperatoren L_1 und L_2 aus $\mathbb{F}[x][\varepsilon_q]$ das so genannte *symmetrische Produkt* $L_1 \circledS L_2$ von L_1 und L_2. Das symmetrische

Produkt $L_1 \circledS L_2$ ist der monische q-Rekursionsoperator L minimaler Ordnung, für den $L(f(x) \cdot g(x)) = 0$ gilt, wobei $f, g \in \mathcal{H}_q(\mathbb{F})$ mit $L_1 f(x) = 0$ und $L_2 g(x) = 0$. Betrachtet man q-holonome Differentialgleichungen, so muss man die auftretenden q-Ableitungen $D_q^k(f(x) \cdot g(x))$ zunächst mit der q-Leibnizregel des Hahnoperators ausmultiplizieren, was zusätzlichen Aufwand bedeutet. An dieser Stelle ist es also wesentlich effizienter mit dem q-Shiftoperator zu arbeiten.

Abschließend stellen wir den Kompositionsalgorithmus vor.

Algorithmus 3.4 Bestimmung einer q-holonomen Rekursionsgleichung, die als Lösung die Komposition einer Lösung einer q-holonomen Rekursionsgleichung mit einer Funktion $g(x)$ mit $g(qx) = q^b g(x)$ besitzt (`qCompositionRE`)

Eingabe : Eine q-holonome Rekursionsgleichung $\sum_{k=0}^{n} a_k(x) \varepsilon_q^k f(x) = 0$ der Ordnung n für $f(x)$ und eine Funktion $g(x)$ mit $g(qx) = q^b g(x)$ für ein $b \in \mathbb{N}$

Ausgabe : Eine q-holonome Rekursionsgleichung, die für die Verkettung $(f \circ g)(x)$ gültig ist und die Ordnung n besitzt

begin
- *Ansatz* $\leftarrow \sum_{k=0}^{n} \tilde{A}_k(x) \varepsilon_q^k F(x) = 0$
- *RE* $\leftarrow \sum_{k=0}^{n} A_k(x) \varepsilon_q^{bk} f(x)$
- *RE* $\leftarrow$ Reduziere *RE* durch rekursive Anwendung von $\varepsilon_q^n f(x) \rightarrow -\sum_{k=0}^{n-1} \frac{a_k(x)}{a_n(x)} \varepsilon_q^k f(x)$
- Setze die Koeffizienten von $f(x), \ldots, \varepsilon_q^{n-1} f(x)$ von *RE* Null.
- Löse das resultierende lineare Gleichungssystem mit n Gleichungen nach den $n+1$ Unbekannten $A_0(x), \ldots, A_n(x)$ über $\mathbb{F}(x)$.
- Substituiere $x \rightarrow g(x)$ in den Lösungen $A_k(x)$ und bilde somit die Lösungen $\tilde{A}_k(x) = A_k(g(x))$ für $k = 0, \ldots, n$.
- Setze die Lösungen $\tilde{A}_k(x)$ in *Ansatz* ein und multipliziere mit $\operatorname{lcm}(\operatorname{denom}(\tilde{A}_0(x)), \ldots, \operatorname{denom}(\tilde{A}_n(x)))$.
- **return** *die so erhaltene q-holonome Rekursionsgleichung*

end

Beweis Nach Satz 1.8 gilt $\varepsilon_q (f \circ g)(x) = \left[\varepsilon_{q^b} f(x)\right]_{x=g(x)}$ und somit für $k \in \mathbb{N}$

$$\varepsilon_q^k (f \circ g)(x) = \varepsilon_{q^k} (f \circ g)(x) = \left[\varepsilon_{(q^k)^b} f(x)\right]_{x=g(x)} = \left[\varepsilon_{q^{bk}} f(x)\right]_{x=g(x)} = \left[\varepsilon_q^{bk} f(x)\right]_{x=g(x)}.$$

Ausgehend von

$$L((f \circ g)(x)) := \sum_{k=0}^{n} \tilde{A}_k(x) \varepsilon_q^k (f \circ g)(x) = \left[\sum_{k=0}^{n} A_k(x) \varepsilon_q^{bk} f(x)\right]_{x=g(x)}$$

mit $A_k(g(x)) = \tilde{A}_k(x)$ erhält man durch rekursive Anwendung der aus der gegebenen q-holonomen Rekursion gewonnenen Regel

$$\varepsilon_q^n f(x) = -\sum_{k=0}^{n-1} \frac{a_k(x)}{a_n(x)} \varepsilon_q^k f(x)$$

die Darstellung

$$L((f \circ g)(x)) = \left[\sum_{k=0}^{n-1} B_k(x) \varepsilon_q^k f(x)\right]_{x=g(x)}$$

mit $B_k(x) \in \mathbb{F}(x)$. Setzt man nun die Summe der letzten Gleichung Null und führt Koeffizientenvergleich bzgl. $f(x), \ldots, \varepsilon_q^{n-1} f(x)$ durch, bekommt man ein homogenes lineares Gleichungssystem

mit n Gleichungen und $n+1$ Unbekannten $A_0(x),\dots,A_n(x)$ über $\mathbb{F}(x)$. Da dieses Gleichungssystem mehr Unbekannte als Gleichungen hat, besitzt es eine nichttriviale Lösung. Indem man $\tilde{A}_k(x) = A_k(g(x))$ in

$$\sum_{k=0}^{n} \tilde{A}_k(x)\varepsilon_q^k F(x) = 0$$

einsetzt und anschließend mit $\operatorname{lcm}(\operatorname{denom}(\tilde{A}_0(x)),\dots,\operatorname{denom}(\tilde{A}_n(x)))$ multipliziert, erhält man die q-holonome Rekursionsgleichung für die Komposition $(f\circ g)(x)$ der Ordnung n. □

Im Folgenden betrachten wir die implementierten Algorithmen in `qFPS`. Eine genaue Beschreibung der Prozeduren ist in Kapitel 6 zu finden.

Maple-Sitzung 3.3 (q-Holonome Algebra)

```
> RE1:=Sq[x,x](F(x))-(1+q)*Sq[x](F(x))+q*(1+x^2)*F(x)=0;
```

$$RE1 := Sq_{x,x}(F(x)) - (1+q)\,Sq_x(F(x)) + q\left(1+x^2\right)F(x) = 0$$

```
> RE2:=(1+x)*Sq[x](F(x))-F(x)=0;
```

$$RE2 := (1+x)\,Sq_x(F(x)) - F(x) = 0$$

Summe der q-Rekursionen

```
> qSumRE(RE1,RE2,F(x));
```

$$\begin{aligned}&-q^3\left(1+x^2\right)\left(x^2q^3+qx+xq^2+1+q\right)F(x)+q\Big(1+2q^2+2q+2xq^2+x+q^4x^2+qx+2x^2q^2+2x^2q^3\\&+qx^2+q^5x^2+q^4x^4+x^4q^3+x^5q^5+q^3+q^4x\ +q^5x^4+q^4x^3+2x^3q^3+x^3q^2+3q^3x+q^5x^3\Big)Sq_x(F(x))\\&+\Big(-1-2q^4x-3q^3x-2x^2q^3-q^4x^2-2qx-2xq^2-2x^2q^2-q^5x^2-qx^2-x^3q^3-q^4x^3-x-2q\\&-2q^2-q^3\Big)\left(Sq_{x,x}\right)(F(x))\ +\left(xq^2+1\right)\left(q+qx^2+qx+x+1\right)\left(Sq_{x,x,x}\right)(F(x))=0\end{aligned}$$

Produkt der q-Rekursionen

```
> qProductRE(RE1,RE2,F(x));
```

$$(1+x)(1+qx)\,Sq_{x,x}(F(x)) - (1+x)(1+q)\,Sq_x(F(x)) + q\left(1+x^2\right)F(x) = 0$$

Komposition der ersten q-Rekursion mit ax^2

```
> qCompositionRE(RE1,F(x),a*x^2);
```

$$Sq_{x,x}(F(x)) + \left(1+q^2\right)\left(q^3a^2x^4-1\right)Sq_x(F(x)) + q^2\left(a^2x^4+1\right)\left(q^2a^2x^4+1\right)F(x) = 0$$

Der Summenalgorithmus liefert i.Allg. keine Rekursionsgleichung minimaler Ordnung

```
> RE3:=qHolonomicRE(x,F(x));
```

$$RE3 := Sq_x(F(x)) - qF(x) = 0$$

```
> RE4:=qHolonomicRE(1/x,F(x));
```

$$RE4 := qSq_x(F(x)) - F(x) = 0$$

```
> qSumRE(RE3,RE4,F(x));
```

$$q\left(Sq_{x,x}\right)(F(x)) + \left(-q^2-1\right)Sq_x(F(x)) + qF(x) = 0$$

```
> qHolonomicRE(x+1/x,F(x));
```

$$q\left(1+x^2\right)Sq_x(F(x)) + \left(-1-q^2x^2\right)F(x) = 0$$

Ein Mathematica-Package mit ähnlichen Funktionalitäten wurde in [KK09] präsentiert.

Kapitel 4

Lösen von q-holonomen Rekursionsgleichungen

In diesem Kapitel widmen wir uns den Lösungen von q-holonomen Rekursionsgleichungen. Wie wir bereits gesehen haben, können q-holonome Rekursionsgleichungen in additiver und in multiplikativer Form vorliegen und wir können diese Darstellungen durch einfache Substitution ineinander überführen. Wir betrachten an dieser Stelle q-holonome Rekursionsgleichungen in multiplikativer Form und interessieren uns für polynomiale, rationale und q-hypergeometrische Lösungen dieser Rekursion. Bei den q-hypergeometrischen Lösungen werden wir uns dann mit der Bestimmung eines q-Zertifikats zufrieden geben. Ein q-Zertifikat entspricht einer einfachen q-hypergeometrischen Rekursionsgleichung erster Ordnung und diese Rekursion kann man, indem man sie in die additive Darstellung bringt, lösen und damit die q-hypergeometrische Lösung explizit angeben.

4.1 Vorbereitungen zur Lösungstheorie q-holonomer Rekursionsgleichungen

Bevor wir auf die Lösungstheorie der q-holonomen Rekursionsgleichungen eingehen, werden wir zunächst einige nützliche Hilfsmittel zur Verfügung stellen, die wir in diesem Kapitel benötigen.

4.1.1 Das q-Newton-Polygon

Definition 4.1 Sei R ein Ring. Eine Abbildung $v : R \to \mathbb{R} \cup \{\infty\}$ heißt eine *Bewertung auf R*, wenn für alle $\alpha, \beta \in R$ die folgenden Axiome gelten

(a) $v(0) = \infty$

(b) $v(\alpha \cdot \beta) = v(\alpha) + v(\beta)$

(c) $v(\alpha + \beta) \geq \min\{v(\alpha), v(\beta)\}$

(d) $v(\alpha) \neq v(\beta) \;\Rightarrow\; v(\alpha + \beta) = \min\{v(\alpha), v(\beta)\}$.

Wir nennen v eine *echte Bewertung*, wenn zusätzlich gilt

(e) $v(\alpha) = \infty \;\Rightarrow\; \alpha = 0$.

Bemerkung 4.2 Wir werden im Folgenden die zwei echten Bewertungen $v_{\deg}$ und v_{ldeg} auf dem Polynomring $\mathbb{F}[x]$ betrachten. Diese sind für $a(x) \in \mathbb{F}[x]$ definiert als

$$v_{\deg}(a(x)) = -\deg_x(a(x)) \tag{4.1}$$

und

$$v_{\mathrm{ldeg}}(a(x)) = \mathrm{ldeg}_x(a(x)), \tag{4.2}$$

wobei $\deg_x(a(x))$ der *Grad* und $\mathrm{ldeg}_x(a(x))$ die *x-adische Bewertung* des Polynoms $a(x)$ sein soll.

In der Dissertation [Hor08] wurde das q-Newton-Polygon ausführlich behandelt. Wir schreiben im nächsten Abschnitt zusammenfassend die wichtigsten Definitionen und Sätze auf.

Definition 4.3 Sei

$$L = \sum_{k=0}^{n} a_k(x)\varepsilon_q^k \in \mathbb{F}[x][\varepsilon_q]$$

ein q-holonomer Rekursionsoperator. Dann ist das zugehörige *q-Newtonpolygon bzgl. der Bewertung* v definiert als

$$N_v(L) := \text{Konvexe Hülle von } \bigcup_{k=0}^{n} \{(k, y) \mid y \geq v(a_k(x))\} \subseteq \mathbb{R}^2.$$

Beispiel 4.4 Sei die Bewertung $v_{\deg}$ und

$$L = \varepsilon_q^5 + (q^2x^2 - 1)\varepsilon_q^4 + x\,\varepsilon_q^3 - (x^4 + qx + 1)\varepsilon_q^2 + q^2x^4\varepsilon_q - (x^3 + q)$$

gegeben. Dann sind die Eckpunkte des q-Newton-Polygons $N_{v_{\deg}}(L)$ gegeben durch

$$\{(0,-3), (1,-4), (2,-4), (4,-2), (5,0)\}.$$

Der zum Term $x\,\varepsilon_q^3$ gehörige Punkt $(3,-1)$ liegt im Inneren des q-Newton-Polygons und ist demzufolge kein Eckpunkt.

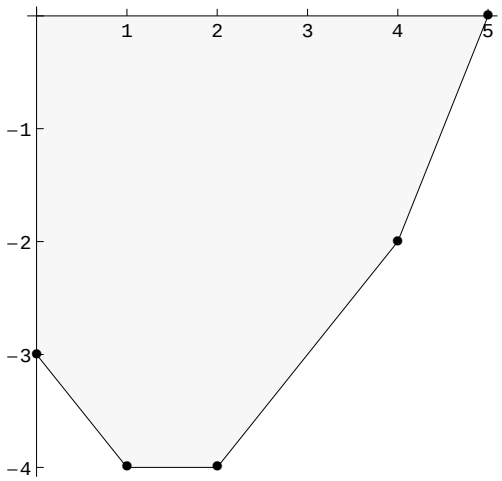

Abbildung 4.1 Das q-Newton-Polygon $N_{v_{\deg}}(L)$

Im Folgenden betrachten wir die Steigungen der q-Newton-Polygone näher. Abschnitte des q-Newton-Polygons mit Steigung $w \in \mathbb{Q}$ bezeichnen wir als *Kante* w.

Definition 4.5 Das *charakteristische Polynom zur Kante* $w \in \mathbb{Q}$ von $N_v(L)$ ist

$$\mathrm{P}_{L,v,w}(T) := \sum_{\substack{k,i \\ (k,v(\alpha_{ki}x^i)) \text{ liegt auf Kante } w}} q^{-v(x)\frac{k(k-1)}{2}w}\alpha_{ki}T^{k-k_0}$$

mit $L = \sum_{k=0}^{n}\sum_{i=0}^{m_k}\alpha_{ki}x^i\varepsilon_q^k \in \mathbb{F}[x][\varepsilon_q]$, wobei $k_0 \in \mathbb{N}_0$ so gewählt ist, dass $T \nmid \mathrm{P}_{L,v,w}(T)$.

Beispiel 4.6 Greifen wir Beispiel 4.4 auf und betrachten die charakteristischen Polynome zu allen auftretenden Kanten bzgl. der Bewertung $v_{\deg}$, so erhalten wir

$$\begin{aligned} \mathrm{P}_{L,v_{\deg},-1}(T) &= q^2T - 1 \\ \mathrm{P}_{L,v_{\deg},0}(T) &= -T + q^2 \\ \mathrm{P}_{L,v_{\deg},1}(T) &= q^8T^2 - q \\ \mathrm{P}_{L,v_{\deg},2}(T) &= q^{20}T + q^{14}. \end{aligned}$$

Um das q-Newton-Polygon $N_v(L)$ eines gegebenen Rekursionsoperators L bzgl. einer Bewertung v mitsamt der charakteristischen Polynome zu bestimmen, verwendet man

Algorithmus 4.1 Bestimmung des q-Newton-Polygons einer q-holonomen Rekursionsgleichung und die dazugehörigen charakteristischen Polynome (`qNewtonPolygon`)

Eingabe : Eine q-holonome Rekursionsgleichung $Lf(x) := \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ der Ordnung n und eine Bewertung v

Ausgabe : Das q-Newton-Polygon als Liste von Eckpunkten und die charakteristischen Polynome zu den Kanten als Liste von Paaren $(w, \mathrm{P}_{L,v,w}(T))$ mit w als Kante

```
1  begin
2      NP ← ∅
3      charpols ← ∅
4      i ← 0
5      while i ≠ n do
6          if a_i(x) = 0 then next i
7          NP = NP ∪ {(i, v(a_i(x)))}
8          slope ← ∞
9          for j ← i + 1 to n do
10             if a_j(x) = 0 then next j
11             s ← (v(a_j(x)) − v(a_i(x))) / (j − i)
12             if s < slope then
13                 slope ← s
14                 charpol ← q^{−v(x) i(i−1)/2 s} · (Koeff. von a_i(x) mit Bewertung v(a_i(x)))
15             end
16             if s = slope then
17                 charpol ← charpol + q^{−v(x) j(j−1)/2 s} · (Koeff. von a_j(x) mit Bewertung v(a_j(x))) · T^{j−i}
18                 k ← j
19             end
20         end
21         charpols = charpols ∪ {(slope, charpol)}
22         i ← k
23     end
24     NP = NP ∪ {(n, v(a_n(x)))}
25     return NP, charpols
26 end
```

Beweis Der Algorithmus geht folgendermaßen vor. Zunächst wird in Zeile 7 der erste Eckpunkt $(i, v(a_i(x)))$ des q-Newton-Polygons bestimmt. Daraufhin werden in der Schleife ab Zeile 9 alle

Steigungen von diesem Punkt zu anderen Punkten $(j, v(a_j(x)))$ für $j = i+1, \ldots, n$ berechnet und unter diesen wird die kleinste Steigung (beachte Konvexität) und der am weitesten rechts gelegene Punkt mit dieser Steigung ermittelt. Die Position wird in der Variablen k und die Steigung der Kante in *slope* gespeichert. Ferner wird während des Schleifendurchgangs das charakteristische Polynom *charpol* zur Kante *slope* entwickelt. Dazu wird *charpol* mit dem Koeffizienten von $a_i(x)$ mit Bewertung $v(a_i(x))$ initialisiert (Zeile 14). In den Zeilen 16 bis 19 werden weitere Summanden des charakteristischen Polynoms hinzugenommen, falls der aktuell betrachtete Punkt auf der Kante *slope* liegt. Die charakteristischen Polynome werden in *charpols* zusammen mit der Kante *slope* gespeichert. Der Punkt $(k, v(a_k(x)))$ ist der nächste Eckpunkt des q-Newton-Polygons. Aus diesem Grund wird i auf k gesetzt und der Vorgang wiederholt. Ist $k = n$, so wird der Iterationsvorgang abgebrochen und als letzter Punkt des q-Newton-Polygons wird $(n, v(a_n(x)))$ zu der Menge der Eckpunkte des q-Newton-Polygons *NP* hinzugefügt. □

Der zentrale Satz dieses Kapitels aus [Hor08], den wir an vielen Stellen nutzen werden, ist

Satz 4.7 Sei $f(x)$ eine q-hypergeometrische Lösung von $Lf(x) = 0$ mit q-Zertifikat $r(x) = c\frac{s(x)}{t(x)}$, wobei $L = \sum_{k=0}^{n} a_k(x)\varepsilon_q^k \in \mathbb{F}[x][\varepsilon_q]$, $c \in \mathbb{F}^*$, $s(x), t(x) \in \mathbb{F}[x]$, $\gcd(s(x), t(x)) = 1$ und $s(x), t(x)$ bzgl. der gegebenen Bewertung v normiert[1]. Dann besitzt das q-Newton-Polygon $N_v(L)$ eine Kante der Steigung $w = -(v(s(x)) - v(t(x)))$. Ferner gilt

$$\mathrm{P}_{L,v,w}(c) = 0.$$

Beweis Sei also $f(x)$ eine q-hypergeometrische Lösung mit obigen Voraussetzungen gegeben. Die erste Teilaussage, die besagt, dass das q-Newton-Polygon eine Kante der Steigung $w = -(v(s(x)) - v(t(x)))$ besitzt, ist in [Hor08] nachzulesen und verwendet u.a. Legendre-Transformationen von Bewertungspolygonen. Wir beweisen die zweite Teilaussage. Man erhält mit $\varepsilon_q f(x) = f(x)r(x)$ durch Induktion $\varepsilon_q^k f(x) = f(x)\prod_{i=0}^{k-1} \varepsilon_q^i r(x)$ für $k \in \mathbb{N}$. Folglich gilt

$$Lf(x) = \left(\sum_{k=0}^{n} a_k(x) \prod_{i=0}^{k-1} \varepsilon_q^i r(x)\right) f(x) = \left(\sum_{k=0}^{n} a_k(x) c^k \prod_{i=0}^{k-1} \varepsilon_q^i \frac{s(x)}{t(x)}\right) f(x) = 0$$

bzw. nach Multiplikation des linken Faktors mit $\prod_{i=0}^{n-1} \varepsilon_q^i t(x)$

$$\sum_{k=0}^{n} a_k(x) c^k \prod_{i=0}^{k-1} \varepsilon_q^i\,(s(x)) \prod_{i=k}^{n-1} \varepsilon_q^i\,(t(x)) = 0.$$

Definiere nun

$$\mathrm{P}(T) := \sum_{k=0}^{n} \underbrace{\left(a_k(x) \prod_{i=0}^{k-1} \varepsilon_q^i\,(s(x)) \prod_{i=k}^{n-1} \varepsilon_q^i\,(t(x))\right)}_{=:b_k(x)} T^k,$$

so gilt offenbar $P(c) = 0$.

Wir betrachten nun das Polynom $\mathrm{P}(T)$ als Polynom in x und führen Koeffizientenvergleich durch. Dazu selektieren wir die Polynome $b_k(x)$, deren Bewertung minimal ist, aus, denn deren Leitkoeffizient trägt zum höchsten Koeffizienten ($v_{\deg}$) bzw. deren Fußkoeffizient trägt zum niedrigsten Koeffizienten (v_{ldeg}) von $P(T)$ bzgl. x bei. Jeder Koeffizient ist ein Polynom in T und besitzt c als Nullstelle. Die Bewertung von $b_k(x)$ ist gegeben durch

$$\begin{aligned} v(b_k(x)) &= v\left(a_k(x) \prod_{i=0}^{k-1} \varepsilon_q^i\,(s(x)) \prod_{i=k}^{n-1} \varepsilon_q^i\,(t(x))\right) \\ &= v(a_k(x)) + \sum_{i=0}^{k-1} v(s(x)) + \sum_{i=k}^{n-1} v(t(x)) \\ &= v(a_k(x)) + k\,v(s(x)) + (n-k)v(t(x)) \\ &= v(a_k(x)) - k\,w + n\,v(t(x)), \end{aligned}$$

[1] Für $v = v_{\deg}$ gelte $\mathrm{lcoeff}_x(s(x)) = \mathrm{lcoeff}_x(t(x)) = 1$ und für $v = v_{\mathrm{ldeg}}$ gelte $\mathrm{tcoeff}_x(s(x)) = \mathrm{tcoeff}_x(t(x)) = 1$.

wobei wir hier nur Bewertungen ($v_{\deg}$ und $v_{\ldeg}$) verwenden, die mit dem q-Shift verträglich sind, d.h. für die $v(\varepsilon_q g(x)) = v(g(x))$ gilt, sonst könnten wir die zweite Gleichung nicht folgern. Der Term $n\,v(t(x))$ ist nicht abhängig von k und kann vernachlässigt werden. Die Bewertung $v(b_k(x))$ wird minimal, wenn der dazugehörige Punkt $(k, v(a_k(x))$ auf der Kante w des q-Newton-Polygons $N_v(L)$ liegt, denn es gilt zum Einen

$$\frac{v(a_{k_{\min}}(x)) - v(a_i(x))}{k_{\min} - i} < w \quad \Longrightarrow \quad v(a_{k_{\min}}(x)) - k_{\min} w < v(a_i(x)) - i\,w$$

für alle $i < k_{\min}$ und zum Anderen

$$\frac{v(a_{k_{\max}}(x)) - v(a_i(x))}{k_{\max} - i} > w \quad \Longrightarrow \quad v(a_{k_{\max}}(x)) - k_{\max} w < v(a_i(x)) - i\,w$$

für alle $i > k_{\max}$, wobei $k_{\min}$ den kleinsten Index und $k_{\max}$ den größten Index bezeichne, für den $(k, v(a_k(x)))$ auf der Kante w liegt. Für solche i mit $k_{\min} < i < k_{\max}$, bei denen $(i, v(a_i(x)))$ nicht auf der Kante w liegt, gelten offenbar ebenfalls obige Abschätzungen und somit ist die Bewertung dort nicht minimal.

Wähle im Folgenden $v = v_{\deg}$, also $w = \deg_x(s(x)) - \deg_x(t(x))$. Betrachten wir nun den Leitkoeffizienten von $b_k(x)$, dann bekommen wir

$$\begin{aligned}
\operatorname{lcoeff}_x(b_k(x)) &= \operatorname{lcoeff}_x(a_k(x)) \prod_{i=0}^{k-1} q^{i\deg_x(s(x))} \prod_{i=k}^{n-1} q^{i\deg_x(t(x))} \\
&= \operatorname{lcoeff}_x(a_k(x)) q^{\frac{k(k-1)}{2}\deg_x(s(x))} q^{\left(\frac{n(n-1)}{2} - \frac{k(k-1)}{2}\right)\deg_x(t(x))} \\
&= \operatorname{lcoeff}_x(a_k(x)) q^{\frac{k(k-1)}{2}(\deg_x(s(x)) - \deg_x(t(x)))} q^{\frac{n(n-1)}{2}\deg_x(t(x))} \\
&= \operatorname{lcoeff}_x(a_k(x)) q^{\frac{k(k-1)}{2} w} q^{\frac{n(n-1)}{2}\deg_x(t(x))}.
\end{aligned}$$

Der Faktor $q^{\frac{n(n-1)}{2}\deg_x(t(x))}$ ist unabhängig von k und kann somit gekürzt werden. Der Fall $v = v_{\ldeg}$ verläuft analog durch Betrachtung des niedrigsten Koeffizienten von $b_k(x)$. Man erhält dann mit $w = -(\ldeg_x(s(x)) - \ldeg_x(t(x)))$

$$\operatorname{tcoeff}_x(b_k(x)) = \operatorname{tcoeff}_x(a_k(x)) q^{-\frac{k(k-1)}{2} w} q^{\frac{n(n-1)}{2}\ldeg_x(t(x))}.$$

Wiederum kann der konstante Faktor $q^{\frac{n(n-1)}{2}\ldeg_x(t(x))}$ bzgl. k vernachlässigt werden.

Nach obigen Betrachtungen gilt also zusammenfassend

$$\sum_{\substack{k,i \\ (k, v(\alpha_{ki} x^i)) \text{ liegt auf Kante } w}} q^{-v(x)\frac{k(k-1)}{2} w} \alpha_{ki} c^k = \mathrm{P}_{L,v,w}(c) c^{k_0} = 0$$

mit α_{ki} und k_0 aus Definition 4.5, also die Behauptung. □

4.1.2 Lokale Typen q-hypergeometrischer Terme

Wir möchten im Folgenden Informationen über q-hypergeometrische Terme liefern, die uns später befähigen, Lösungstypen von q-holonomen Rekursionsgleichungen (wie in [CH05] für den klassischen Fall) anzugeben. Dazu benötigen wir einige Definitionen.

Definition 4.8 Seien $\alpha, \beta \in \overline{\mathbb{F}}$. Wir führen folgende *$\varepsilon_q$-Relation* ein

$$\alpha \sim_{\varepsilon_q} \beta \quad :\Longleftrightarrow \quad \alpha = q^k \beta \text{ für ein } k \in \mathbb{Z}$$

Die ε_q-Relation ist offenbar eine Äquivalenzrelation, so dass wir die *ε_q-Äquivalenzklasse* von $\alpha \in \overline{\mathbb{F}}$ mit

$$q^{\mathbb{Z}}\alpha := \left\{ q^k \alpha \;\middle|\; k \in \mathbb{Z} \right\}$$

bezeichnen. Wir nennen α und β ε_q-*äquivalent*, wenn $\alpha \sim_{\varepsilon_q} \beta$ gilt. Ferner ist $\overline{\mathbb{F}}/\sim_{\varepsilon_q}$ die Menge aller ε_q-Äquivalenzklassen. Die Funktion

$$\log_{\varepsilon_q}^{\alpha} : \ q^{\mathbb{Z}}\alpha \ \to \ \mathbb{Z}, \quad q^k\alpha \ \mapsto \ k$$

gibt die *q-Shiftdifferenz* zu α an.

Definition 4.9 Sei p ein Polynom aus $\mathbb{F}[x]$ und $\alpha \in \overline{\mathbb{F}}$. Dann bezeichne

$$\mathfrak{Z}_p^{\alpha} := \left\{\beta \in \overline{\mathbb{F}} \ \mid \ p(\beta) = 0 \ \text{ und } \ \beta \in q^{\mathbb{Z}}\alpha\right\}$$

die *Nullstellenmenge von $p(x)$ bzgl. α*.

Definition 4.10 Wir sagen

$$\alpha \geq_{\varepsilon_q} \beta \quad :\Longleftrightarrow \quad \alpha = q^k\beta \ \text{ für ein } \ k \in \mathbb{N}_0$$

für $\beta \in q^{\mathbb{Z}}\alpha$ und bezeichnen die Ordnungsrelation $\geq_{\varepsilon_q}$ als ε_q-*Ordnung*. Mit dieser ε_q-Ordnung definiert man das ε_q-*Maximum* bzw. ε_q-*Minimum* einer nichtleeren, endlichen Menge $M \in q^{\mathbb{Z}}\alpha$ und bezeichnet es als $\max_{\varepsilon_q}(M)$ bzw. $\min_{\varepsilon_q}(M)$.

Lemma 4.11 Sei

$$r(x) = \frac{s(x)}{t(x)} \quad \text{mit} \quad s(x), t(x) \in \mathbb{F}[x]. \tag{4.3}$$

Die rationale Funktion $r(x)$ kann man wie folgt als formale Potenzreihe um 0 und als formale Potenzreihe im Unendlichen ($x = \frac{1}{y}$ und Entwicklung um Entwicklungspunkt $y = 0$) ausdrücken. Es gilt

$$r(x) = c_0 x^{v_0}\left(1 + \mathcal{O}(x^1)\right) \tag{4.4}$$

und

$$r\left(\frac{1}{y}\right) = c_\infty y^{v_\infty}\left(1 + \mathcal{O}(y^1)\right) \tag{4.5}$$

mit[2]

$$c_0 = \frac{\operatorname{tcoeff}_x(s(x))}{\operatorname{tcoeff}_x(t(x))} \in \mathbb{F}^* \quad \text{und} \quad v_0 = \operatorname{ldeg}_x(s(x)) - \operatorname{ldeg}_x(t(x)) \in \mathbb{Z}$$

und

$$c_\infty = \frac{\operatorname{lcoeff}_x(s(x))}{\operatorname{lcoeff}_x(t(x))} \in \mathbb{F}^* \quad \text{und} \quad v_\infty = -\left(\deg_x(s(x)) - \deg_x(t(x))\right) \in \mathbb{Z}.$$

Beweis Wir entwickeln die zu $r(x)$ zugehörige Potenzreihendarstellung. Dazu gelten obige Formeln. Wir erhalten einerseits

$$r\left(x\right) = \frac{s(x)}{t(x)} = x^{v_0}\frac{x^{-\operatorname{ldeg}_x(s(x))}}{x^{-\operatorname{ldeg}_x(t(x))}}\frac{s(x)}{t(x)} = x^{v_0}\left(c_0 + \mathcal{O}\left(x^1\right)\right) = c_0 x^{v_0}\left(1 + \mathcal{O}\left(x^1\right)\right)$$

und andererseits mit $x = \frac{1}{y}$ und Polynomen $\tilde{s}(y), \tilde{t}(y) \in \mathbb{F}[y]$[3]

$$r\left(\frac{1}{y}\right) = \frac{s(x)}{t(x)} = x^{-v_\infty}\frac{x^{-\deg_x(s(x))}}{x^{-\deg_x(t(x))}}\frac{s(x)}{t(x)} = y^{v_\infty}\frac{\tilde{s}(y)}{\tilde{t}(y)} = y^{v_\infty}\left(c_\infty + \mathcal{O}\left(y^1\right)\right) = c_\infty y^{v_\infty}\left(1 + \mathcal{O}\left(y^1\right)\right).$$

□

[2]Wir können auch schreiben $v_0 = v_{\operatorname{ldeg}}(s(x)) - v_{\operatorname{ldeg}}(t(x))$ bzw. $v_\infty = v_{\deg}(s(x)) - v_{\deg}(t(x))$.
[3]Das Polynom $\tilde{s}(y)$ bzw. $\tilde{t}(y)$ entsteht aus $s(x)$ bzw. $t(x)$ durch Umkehrung der Reihenfolge der Koeffizienten.

Definition 4.12 Ein q-hypergeometrischer Term $f(x)$ besitzt den *lokalen Typ in 0*

$$\mathrm{ltyp}_0(f(x)) := (c_0, v_0)\,,$$

wenn das q-Zertifikat $r(x)$ von $f(x)$ die Darstellung

$$r(x) = c_0 x^{v_0}\left(1 + \mathcal{O}(x^1)\right)$$

besitzt, wobei $c_0 \in \mathbb{F}^*$ und $v_0 \in \mathbb{Z}$ sind. Ferner sagen wir $f(x)$ besitzt den *lokalen Typ im Unendlichen*

$$\mathrm{ltyp}_\infty(f(x)) := (c_\infty, v_\infty)\,,$$

wenn $r(x)$ die Darstellung

$$r(x) = r\left(\frac{1}{y}\right) = c_\infty y^{v_\infty}\left(1 + \mathcal{O}(y^1)\right)$$

besitzt, wobei $c_\infty \in \mathbb{F}^*$ und $v_\infty \in \mathbb{Z}$ sind.

Beispiel 4.13 Sei

$$r(x) = \frac{(1+qx)(1-qx)}{x(1+2x)^2(1-q^2x)}$$

das q-Zertifikat des q-hypergeometrischen Terms $f(x)$. Für die lokalen Typen ergeben sich einerseits mit $c_0 = \frac{1\cdot 1}{1\cdot 1\cdot 1} = 1$ und $v_0 = (0+0) - (1+0+0)) = 0 - 1 = -1$ in 0

$$\mathrm{ltyp}_0(f(x)) = (1, -1)$$

und andererseits mit $c_\infty = \frac{q\cdot(-q)}{1\cdot 2^2\cdot(-q^2)} = \frac{1}{4}$ und $v_\infty = -((1+1) - (1+2+1)) = -(2-4) = 2$ im Unendlichen

$$\mathrm{ltyp}_\infty(f(x)) = \left(\frac{1}{4}, 2\right).$$

Definition 4.14 Ein q-hypergeometrischer Term $f(x)$ mit q-Zertifikat $r(x)$ besitzt den *lokalen Typ an der Stelle* $p \in \overline{\mathbb{F}}^*$

$$\mathrm{ltyp}_p(f(x)) := \sum_{\tilde{p}\in 3_s^p} \mathrm{mult}(\tilde{p}) - \sum_{\tilde{p}\in 3_t^p} \mathrm{mult}(\tilde{p})$$

mit $r(x), s(x), t(x)$ aus Darstellung (4.3), wobei $\mathrm{mult}(\tilde{p})$ die Vielfachheit der Nullstelle $\tilde{p}$ bezeichnet.

Beispiel 4.15 Führen wir nun Beispiel 4.13 fort, so erhalten wir die lokalen Typen

$$\mathrm{ltyp}_{q^{-1}}(f(x)) = 1 - 1 = 0,\ \mathrm{ltyp}_{-q^{-1}}(f(x)) = 1 \text{ und } \mathrm{ltyp}_{-\frac{1}{2}}(f(x)) = -2.$$

Die lokalen Typen an anderen Stellen (Stellen, die nicht ε_q-äquivalent zu den Nullstellen der Zähler- bzw. Nennerpolynome sind) sind 0.

Die folgenden q-Fuchs-Relationen ([CH05]) sind ein wichtiges Hilfsmittel, um später im Kapitel die Menge der Kandidaten für q-hypergeometrische Lösungen einzuschränken.

Satz 4.16 Es gelten die so genannten *q-Fuchs-Relationen*

$$v_0 + v_\infty + \sum_{q^{\mathbb{Z}}p \in \left(\overline{\mathbb{F}}/\sim_{\varepsilon_q}\right)^*} \mathrm{ltyp}_p(f(x)) = 0 \tag{4.6}$$

und

$$c_0 \sim_{\varepsilon_q} c_\infty \prod_{q^{\mathbb{Z}}p \in \left(\overline{\mathbb{F}}/\sim_{\varepsilon_q}\right)^*} (-p)^{\mathrm{ltyp}_p(f(x))} \tag{4.7}$$

für einen q-hypergeometrischen Term $f(x)$.

Beweis Beide Relationen beweist man durch Angabe zweier verschiedener Wege zu addieren bzw. zu multiplizieren. Wir zeigen zunächst (4.6) und verwenden die Notationen aus Lemma 4.11, wobei $r(x) = \frac{s(x)}{t(x)}$ das q-Zertifikat von $f(x)$ sei. Dann gilt offenbar

$$v_0 + \sum_{q^{\mathbb{Z}}p \in \left(\overline{\mathbb{F}}/\sim_{\varepsilon_q}\right)^*} \mathrm{ltyp}_p(f(x)) = \deg_x(s(x)) - \deg_x(t(x)) = -v_\infty.$$

Die zweite q-Fuchs-Relation (4.7) folgt mit dem Satz von Vieta aus

$$\underbrace{\frac{\mathrm{lcoeff}_x(s(x))}{\mathrm{lcoeff}_x(t(x))}}_{c_\infty} \prod_{q^{\mathbb{Z}}p \in \left(\overline{\mathbb{F}}/\sim_{\varepsilon_q}\right)^*} (-p)^{\mathrm{ltyp}_p(f(x))} \quad \sim_{\varepsilon_q} \quad \frac{\mathrm{tcoeff}_x(s(x))}{\mathrm{tcoeff}_x(t(x))} = c_0.$$

Es gilt nicht unbedingt Gleichheit, da der linke Ausdruck nur bis auf q-Shifts eindeutig ist (die Darstellung ist abhängig von der Repräsentantenwahl von $q^{\mathbb{Z}}p$). □

Definition 4.17 Zwei q-hypergeometrische Terme $f(x)$ und $g(x)$ sind *vom selben Typ*, wenn

$$\frac{f(x)}{g(x)} \in \mathbb{F}(x)$$

gilt.

Definition 4.17 liegt begründet in

Satz 4.18 Zwei q-hypergeometrische Terme $f(x)$ und $g(x)$ sind genau dann vom selben Typ, wenn die lokalen Typen von $f(x)$ und $g(x)$ an jeder Stelle, in 0 und im Unendlichen übereinstimmen[4].

Beweis Sei $r(x)$ das q-Zertifikat von $f(x)$ und $\tilde{r}(x)$ das q-Zertifikat von $g(x)$. Ferner seien $f(x)$ und $g(x)$ vom selben Typ, d.h. es gelte

$$\frac{f(x)}{g(x)} = h(x) \in \mathbb{F}(x).$$

Dann gilt offenbar

$$\frac{\varepsilon_q h(x)}{h(x)} = \frac{r(x)}{\tilde{r}(x)} \qquad \text{bzw.} \qquad r(x) = \frac{\varepsilon_q h(x)}{h(x)} \tilde{r}(x).$$

Anhand der letzten Gleichung sieht man, dass die lokalen Typen von $f(x)$ und $g(x)$ an jeder Stelle, in 0 und im Unendlichen gleich sein müssen, denn der rationale Ausdruck $\frac{\varepsilon_q h(x)}{h(x)}$ führt zu keiner Veränderung der lokalen Typen von $g(x)$, da dort der lokale Typ $\mathrm{ltyp}_p(h(x))$ an jeder Stelle p Null ist und ferner

$$\mathrm{ltyp}_\infty(h(x)) = \left(q^{\deg_x(\mathrm{numer}(h(x))) - \deg_x(\mathrm{denom}(h(x)))}, 0\right)$$

und

$$\mathrm{ltyp}_0(h(x)) = (q^{\mathrm{ldeg}_x(\mathrm{numer}(h(x))) - \mathrm{ldeg}_x(\mathrm{denom}(h(x)))}, 0)$$

gilt. Ausnahmen sind also die Größen c_0 und c_∞, bei der die Aussage nur bis auf q-Shifts richtig ist. Stimmen die lokalen Typen umgekehrt überein, so ist der Quotient der q-Zertifikate $\frac{r(x)}{\tilde{r}(x)}$ nach Definition der lokalen Typen das q-Zertifikat einer rationalen Funktion. Folglich sind die q-hypergeometrischen Terme auch vom selben Typ. □

4.2 Polynomiale Lösungen q-holonomer Rekursionsgleichungen

Wir beschäftigen uns zu allererst mit polynomialen Lösungen von q-holonomen Rekursionen. Die folgenden Algorithmen sind in [APP98] bzw. [Böi98] zu finden.

[4] Bei den lokalen Typen in 0 und im Unendlichen müssen die jeweiligen Größen c_0 und c_∞ nur bis auf q-Shifts übereinstimmen.

4.2.1 Obere Gradschranke polynomialer Lösungen

Wir möchten zunächst eine obere Gradschranke für alle möglichen polynomialen Lösungen einer q-holonomen Rekursionsgleichung ermitteln.

Algorithmus 4.2 Bestimmung einer oberen Gradschranke aller polynomialen Lösungen einer q-holonomen Rekursionsgleichung (`qPolUpperBound`)

Eingabe : Eine q-holonome Rekursion $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ mit $a_k(x) = \sum_{i=0}^{m_k} \alpha_{ki} x^i$

Ausgabe : Obere Gradschranke N der polynomialen Lösungen $f(x)$

```
1  begin
2      m ← max{deg_x(a_k(x)) | k = 0, ..., n}
3      p(x) ← Σ_{k=0}^{n} α_{km} x^k
4      s ← min{k ∈ ℕ_0 | α_{km} ≠ 0}
5      t ← max{i ∈ ℕ_0 | q^i teilt α_{sm}}
6      for i ← t downto 0 do
7          if p(q^t) = 0 then
8              return t
9          end
10     end
11     return -1
12 end
```

Beweis Sei $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ mit $a_k(x) = \sum_{i=0}^{m} \alpha_{ki} x^i$ für $k = 1, \ldots, n$ gegeben, wobei m das Maximum der Grade der Koeffizientenpolynome der q-holonomen Rekursion ist. O.B.d.A. gelte $\alpha_{ki} \in \mathbb{K}[q]$. Ferner sei $f(x) = \sum_{j=0}^{r} \beta_j x^j$ mit $\beta_r \neq 0$ eine polynomiale Lösung der q-holonomen Rekursionsgleichung. Dann gilt

$$\begin{aligned}
\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) &= 0 \\
\sum_{k=0}^{n}\sum_{i=0}^{m}\sum_{j=0}^{r} \alpha_{ki}\beta_j q^{kj} x^{i+j} &= 0 \\
\left(\sum_{k=0}^{n} \alpha_{km}\beta_r q^{kr}\right) x^{m+r} + \ldots = 0 \quad &\Big| \text{ Koeffizientenvgl. bzgl. höchstem Koeffizienten} \\
\beta_r \sum_{k=0}^{n} \alpha_{km} \left(q^r\right)^k &= 0.
\end{aligned}$$

Sei $p(x) = \sum_{k=0}^{n} \alpha_{km} x^k$. Dann muss wegen $\beta_r \neq 0$ und der letzten Gleichung q^r eine Nullstelle von $p(x)$ sein. Wir suchen nun also nach dem größtem $l \in \mathbb{N}_0$, so dass $p(q^l) = 0$ gilt. Sei dazu s der Index des niedrigsten von Null verschiedenen Koeffizients, also $s = \min\{k \in \mathbb{N}_0 \mid \alpha_{km} \neq 0\}$ und t der höchste Exponent von q, so dass q^t den Koeffizienten α_{sm} teilt, also $t = \max\{i \in \mathbb{N}_0 \mid q^i \text{ teilt } \alpha_{sm}\}$. Dann folgt aus $p(q^l) = 0$

$$\begin{aligned}
\alpha_{nm} q^{ln} + \ldots + \alpha_{s+1m} q^{l(s+1)} + \alpha_{sm} q^{ls} &= 0 \\
q^{ls}\left(\alpha_{nm} q^{l(n-s)} + \ldots + \alpha_{s+1m} q^{l} + \alpha_{sm}\right) &= 0 \quad \Big| \ \alpha_{sm} = \tilde{\alpha}_{sm} q^t \text{ mit } \tilde{\alpha}_{sm} \in \mathbb{K}[q] \\
\alpha_{nm} q^{l(n-s)} + \ldots + \alpha_{s+1m} q^{l} + \tilde{\alpha}_{sm} q^{t} &= 0 \\
\left(\alpha_{nm} q^{l(n-s-1)} + \ldots + \alpha_{s+1m}\right) q^{l} + \tilde{\alpha}_{sm} q^{t} &= 0.
\end{aligned}$$

Da das Absolutglied von $\tilde{\alpha}_{sm}$ von Null verschieden ist (t ist maximal), folgt schließlich $l \leq t$. Die obere Gradschranke N ist also gegeben durch $N = \max\{l \in \{0, 1, 2, \ldots, t\} \mid p(q^l) = 0\}$. □

Alternativ können wir mit den eingeführten Notationen für das q-Newton-Polygon und den dazugehörigen charakteristischen Polynomen die Gradschranke auch mit Hilfe des folgenden Satzes bestimmen.

Satz 4.19 Sei $f(x)$ eine polynomiale Lösung der q-holonomen Rekursionsgleichung

$$Lf(x) := \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0,$$

dann gilt

$$\mathrm{P}_{L,v_{\deg},0}\left(q^{\deg_x(f(x))}\right) = 0.$$

Beweis Sei $f(x)$ eine polynomiale Lösung von $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ und L der zugehörige Rekursionsoperator. Dann gilt für das q-Zertifikat von $f(x) = \sum_{j=0}^{N} \beta_j x^j$ mit $N = \deg_x(f(x))$ und $x = \frac{1}{y}$

$$r(x) = \frac{\varepsilon_q f(x)}{f(x)} = q^N \frac{x^N + \frac{\beta_{N-1}q^{-1}}{\beta_N}x^{N-1} + \dots}{x^N + \frac{\beta_{N-1}}{\beta_N}x^{N-1} + \dots} = q^N y^0 \left(1 + \mathcal{O}(y^1)\right).$$

Es folgt somit mit Satz 4.7 $\mathrm{P}_{L,v_{\deg},0}\left(q^N\right) = 0$. □

Anders notiert können wir also auch alle Nullstellen des charakteristischen Polynoms $\mathrm{P}_{L,v_{\deg},0}$ der Form q^k mit $k \in \mathbb{N}_0$ betrachten und unter diesen den größten Exponenten als obere Gradschranke verwenden.

4.2.2 Polynomiale Lösungen q-holonomer Rekursionsgleichungen

Nun können wir alle polynomialen Lösungen einer q-holonomen Rekursionsgleichung bestimmen.

Algorithmus 4.3 Bestimmung aller polynomialen Lösungen einer q-holonomen Rekursionsgleichung (`qPolynomialSolveRE`)

Eingabe : Eine q-holonome Rekursion $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$

Ausgabe : Alle polynomialen Lösungen $f(x)$ der q-holonomen Rekursion

```
1  begin
2      N ← obere Gradschranke aus Algorithmus 4.2
3      if N < 0 then
4          return „Es existieren keine polynomialen Lösungen.“
5      end
6      f(x) ← Σ_{j=0}^{N} β_j x^j
7      Setze f(x) in die q-holonome Rekursionsgleichung ein und betrachte die linke Seite der
       Gleichung als Polynom in x.
8      Löse das durch Koeffizientenvergleich erhaltene lineare Gleichungssystem in den N + 1
       Unbekannten β_0, ..., β_N über 𝔽 .
9      if eine Lösung existiert then
10         Setze die Lösung in f(x) ein.
11         return f(x)
12     end
13     return „Es existieren keine polynomialen Lösungen.“
14 end
```

In [ABP95] wird eine andere Methode zum Bestimmen von polynomialen Lösungen präsentiert, die in einigen Fällen effizienter als das in Algorithmus 4.3 beschriebene Verfahren ist, da ein kleineres lineares Gleichungssystem betrachtet wird.

Wir wollen im Folgenden alle Lösungen der in der nächsten Maple-Sitzung auftretenden q-holonomen Rekursionsgleichungen bestimmen. Die polynomialen Lösungen werden an dieser Stelle ermittelt. Die zweite q-holonome Rekursion besitzt offenbar keine polynomialen Lösungen, während die erste Rekursion ax mit $a \in \mathbb{F}$ als Lösung besitzt.

Maple-Sitzung 4.1 (Polynomiale Lösungen q-holonomer Rekursionsgleichungen)

```
> RE1:=q*Sq[x,x](F(x))-(1+q^2)*Sq[x](F(x))+q*F(x)=0;
```

$$RE1 := \left(Sq_{x,x}\right)(F(x))\,q - \left(1+q^2\right)Sq_x(F(x)) + qF(x) = 0$$

```
> RE2:=q*(q*x-1)*Sq[x,x](F(x))+(1-q^3*x^2)*Sq[x](F(x))-x*(1-q^2*x)*F(x)=0;
```

$$RE2 := q(qx-1)\left(Sq_{x,x}\right)(F(x)) + \left(1-x^2q^3\right)Sq_x(F(x)) - x\left(1-xq^2\right)F(x) = 0$$

```
> qPolynomialSolveRE(RE1,F(x));
```

$$a_0\,x$$

```
> qPolynomialSolveRE(RE2,F(x));
Error, (in qPolynomialSolveRE) there exists no polynomial solution
```

4.3 Rationale Lösungen q-holonomer Rekursionsgleichungen

In diesem Abschnitt sind wir an allen rationalen Lösungen einer q-holonomen Rekursionsgleichung ([Abr95]) interessiert.

4.3.1 Die q-Dispersionsmenge

Wir betrachten im Folgenden den Nenner einer rationalen Lösung genauer und stellen daraufhin Beziehungen zu der q-Dispersionsmenge her.

Lemma 4.20 Sei $f(x)$ eine rationale Lösung der q-holonomen Rekursionsgleichung

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$$

der Ordnung n mit $a_k \in \mathbb{F}[x]$ und sei $x_0 \neq 0$ ein Pol von $f(x)$ der Ordnung m. Dann gilt

(a) Wenn $qx_0, \ldots, q^n x_0$ keine Pole von $f(x)$ sind, dann ist $(x-x_0)^m$ ein Teiler von $a_0(x)$.

(b) Wenn $q^{-1}x_0, \ldots, q^{-n}x_0$ keine Pole von $f(x)$ sind, dann ist $(x-x_0)^m$ ein Teiler von $a_n(q^{-n}x)$.

(c) Wenn $qx_0, \ldots, q^n x_0$ und $q^{-1}x_0, \ldots, q^{-n}x_0$ keine Pole von $f(x)$ sind, dann ist $(x-x_0)^m$ ein Teiler von $\gcd(a_0(x), a_n(q^{-n}x))$.

Beweis Wir stellen die q-holonome Rekursionsgleichung wie folgt um

$$a_0(x)f(x) = -\left(a_1(x)\varepsilon_q f(x) + \ldots + a_n(x)\varepsilon_q^n f(x)\right)$$

und betrachten die rechte Seite dieser Gleichung. Sie besitzt nach Voraussetzung von (a) keinen Pol in x_0. Folglich hat auch die linke Seite keinen Pol in x_0. Das kann allerdings nur richtig sein, wenn $(x-x_0)^m$ ein Teiler von $a_0(x)$ ist, da x_0 ein Pol von $f(x)$ der Ordnung m ist. Die Aussage (b) erfolgt analog durch Betrachtung der Gleichung

$$a_n(q^{-n}x)f(x) = -\left(a_0(q^{-n}x)\varepsilon_q^{-n} f(x) + \ldots + a_{n-1}(q^{-n}x)\varepsilon_q^{-1} f(x)\right).$$

Die Aussage (c) folgt direkt aus den beiden vorigen Aussagen (a) und (b). □

Bemerkung 4.21 Lemma 4.20 ist nicht für $x_0 = 0$ gültig. Die Zahl 0 spielt im q-Fall eine besondere Rolle, da $\gcd(x - x_0, \varepsilon_q x - x_0) \neq 1$ mit $x_0 \in \mathbb{F}$ nur dann gilt, wenn $x_0 = 0$ ist. Wir nennen 0 aus diesem Grund eine *spezielle Singularität*. Im Folgenden werden wir zunächst rationale Lösungen von q-holonomen Rekursionsgleichungen betrachten, bei denen keine spezielle Singularität, also kein Faktor der Form x^{-j} mit $j \in \mathbb{N}$, auftritt.

Lemma 4.22 Sei $v(x)$ der Nenner einer rationalen Lösung $f(x)$ der q-holonomen Rekursionsgleichung

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$$

der Ordnung n mit $a_k \in \mathbb{F}[x]$ und $\alpha \in \overline{\mathbb{F}}$. Ferner sei $\mathfrak{Z}_v^\alpha \neq \emptyset$ (siehe Definition 4.9). Dann gilt

(a) $\max_{\varepsilon_q}\mathfrak{Z}_v^\alpha \in \mathfrak{Z}_{a_0}^\alpha$

(b) $\min_{\varepsilon_q}\mathfrak{Z}_v^\alpha \in \mathfrak{Z}_{\varepsilon_q^{-n}a_n}^\alpha$

(c) $\mathfrak{Z}_v^\alpha \subseteq \left\{\min_{\varepsilon_q}\mathfrak{Z}_{\varepsilon_q^{-n}a_n}^\alpha, \ldots, \max_{\varepsilon_q}\mathfrak{Z}_{a_0}^\alpha\right\}$[5].

Beweis

(a) Es gibt keinen ε_q-größeren Pol von $f(x)$ bzgl. α (bzw. keine ε_q-größere Nullstelle von $v(x)$ aus $\mathfrak{Z}_v^\alpha$) als $\max_{\varepsilon_q}\mathfrak{Z}_v^\alpha$. Nach Lemma 4.20 (a) ist dann $\max_{\varepsilon_q}\mathfrak{Z}_v^\alpha$ eine Nullstelle von $a_0(x)$, also $\max_{\varepsilon_q}\mathfrak{Z}_v^\alpha \in \mathfrak{Z}_{a_0}^\alpha$.

(b) analog zu (a) mit Lemma 4.20 (b)

(c) Es gilt offenbar mit (a) und (b) $\max_{\varepsilon_q}\mathfrak{Z}_v^\alpha \leq \max_{\varepsilon_q}\mathfrak{Z}_{a_0}^\alpha$ und $\min_{\varepsilon_q}\mathfrak{Z}_v^\alpha \geq \min_{\varepsilon_q}\mathfrak{Z}_{\varepsilon_q^{-n}a_n}^\alpha$. Folglich gilt $\mathfrak{Z}_v^\alpha \subseteq \left\{\min_{\varepsilon_q}\mathfrak{Z}_{\varepsilon_q^{-n}a_n}^\alpha, \ldots, \max_{\varepsilon_q}\mathfrak{Z}_{a_0}^\alpha\right\}$.

□

Bei Summationsalgorithmen, wie z.B. dem Gosper-Algorithmus ([Koe98]), taucht der Begriff der Dispersionsmenge auf. Wir definieren nun das entsprechende q-Analogon und können mit der q-Dispersion z.B. „messen“, ob zwei Polynome gemeinsame ε_q-äquivalente Nullstellen über ihrem Zerfällungskörper besitzen und wenn ja, wie weit diese maximal auseinander liegen[6].

Definition 4.23 Die *q-Dispersionsmenge* zweier Polynome $p \in \mathbb{F}[x]$ und $r \in \mathbb{F}[x]$ ist definiert als

$$\mathrm{Disp}_q(p(x), r(x)) := \left\{j \in \mathbb{N}_0 \;\middle|\; \gcd(p(x), \varepsilon_q^j r(x)) \neq 1\right\}. \tag{4.8}$$

Mit der *q-Dispersion* von $p(x)$ und $r(x)$ bezeichnen wir die Zahl

$$\mathrm{disp}_q(p(x), r(x)) := \max \mathrm{Disp}_q(p(x), r(x)).^7 \tag{4.9}$$

Bemerkung 4.24 Sind beide Polynome $p(x)$ und $r(x)$ durch x teilbar, so würde nach obiger Definition die q-Dispersionsmenge unendlich sein. Deshalb legt man in diesem Fall

$$\mathrm{Disp}_q(p(x), r(x)) := \mathrm{Disp}_q(\bar{p}(x), \bar{r}(x))$$

fest mit $p(x) = \bar{p}(x) \cdot x^k$ und $r(x) = \bar{r}(x) \cdot x^l$ und maximalen $k \in \mathbb{N}_0$ und $l \in \mathbb{N}_0$.

Offensichtlich gilt

[5] Die rechte Menge ist folgendermaßen zu verstehen. Die Elemente werden beginnend mit dem linken Element bzgl. der ε_q-Ordnung sukzessive inkrementiert, also mit q multipliziert, bis das rechte Element erreicht ist. Ist die obere Grenze ε_q-kleiner als die untere, so ist die Menge leer.

[6] Die Abbildung, die zwei Polynomen $p(x)$ und $r(x)$ ihre q-Dispersion (q-Dispersionsmenge) zuordnet, ist nicht kommutativ. Die obige Aussage kann man überprüfen, indem man sowohl die q-Dispersion (q-Dispersionsmenge) von $p(x)$ und $r(x)$ als auch die q-Dispersion (q-Dispersionsmenge) von $r(x)$ und $p(x)$ bestimmt.

[7] Wenn die q-Dispersionsmenge leer ist, so definiert man $\mathrm{disp}_q(p(x), r(x)) := -\infty$.

Korollar 4.25 Für die q-Dispersion N zweier Polynome $p(x)$ und $r(x)$ gilt

$$N = \max\left(\left\{ \log_{\varepsilon_q}^{\alpha}\left(\max_{\varepsilon_q} \mathfrak{Z}_r^{\alpha}\right) - \log_{\varepsilon_q}^{\alpha}\left(\min_{\varepsilon_q} \mathfrak{Z}_p^{\alpha}\right) \;\middle|\; \alpha \in \overline{\mathbb{F}} \text{ mit } \mathfrak{Z}_p^{\alpha} \neq \emptyset \text{ und } \mathfrak{Z}_r^{\alpha} \neq \emptyset \right\} \cap \mathbb{N}_0\right).$$

Wir wollen nun einen Algorithmus angeben, der die q-Dispersionsmenge zweier Polynome $p(x)$ und $r(x)$ bestimmt. Der folgende Algorithmus ist eine Modifikation des Algorithmus aus [MW94] bzw. [Koe06].

Algorithmus 4.4 Bestimmung der q-Dispersionsmenge zweier Polynome (`qDispersion`)

Eingabe : Zwei Polynome $p \in \mathbb{F}[x]$ und $r \in \mathbb{F}[x]$

Ausgabe : Die q-Dispersionsmenge $\mathrm{Disp}_q(p(x), r(x))$

```
1  begin
2      DS ← ∅
3      Bestimme p̄(x) mit p(x) = p̄(x) · x^k und maximalem k ∈ ℕ_0
4      Bestimme r̄(x) mit r(x) = r̄(x) · x^l und maximalem l ∈ ℕ_0
5      Faktorisiere p̄(x) und r̄(x) über 𝔽
6      for jeden irreduziblen Faktor p̃(x) von p(x) do
7          for jeden irreduziblen Faktor r̃(x) von r(x) do
8              n ← deg_x(p̃(x))
9              m ← deg_x(r̃(x))
10             if n = m then
11                 a ← coeff_x(p̃(x), n)
12                 b ← coeff_x(p̃(x), 0)
13                 c ← coeff_x(r̃(x), n)
14                 d ← coeff_x(r̃(x), 0)
15                 if (q^n)^j = ad/bc mit j ∈ ℕ_0 then
16                     if cq^{jn} p̃(x) = a r̃(q^j x) then
17                         DS ← DS ∪ {j}
18                     end
19                 end
20             end
21         end
22     end
23     return DS
24 end
```

Beweis In der ersten Zeile des Algorithmus wird die spätere q-Dispersionsmenge mit der leeren Menge initialisiert. Daraufhin wird Bemerkung 4.24 angewendet. Wir berechnen also nun die q-Dispersionsmenge von $\bar{p}(x)$ und $\bar{r}(x)$. Wir zeigen zunächst, dass die q-Dispersionsmenge zweier irreduzibler Polynome $\tilde{p}(x)$ und $\tilde{r}(x)$ durch die Zeilen in dem Rumpf der **for**-Schleifen bestimmt werden. Sei $j \in \mathbb{N}_0$ die q-Dispersion der beiden Polynome $\tilde{p}(x)$ und $\tilde{r}(x)$, es gelte also $\gcd(\tilde{p}(x), \varepsilon_q^j \tilde{r}(x)) \neq 1$. Dann müssen aufgrund der Irreduzibilität die Grade der beiden Polynome übereinstimmen, welches die erste **if**-Abfrage begründet. Die beiden Polynome $\tilde{p}(x)$ und $\varepsilon_q^j \tilde{r}(x)$ müssen sogar assoziiert zueinander sein, d.h. es muss gelten

$$\frac{c}{a} q^{jn} \tilde{p}(x) = \varepsilon_q^j \tilde{r}(x) \tag{4.10}$$

mit den im Algorithmus angegebenen Belegungen. Für die Absolutglieder gilt dann

$$\frac{c}{a} q^{jn} b = d \qquad \text{bzw.} \qquad (q^n)^j = \frac{ad}{bc}.$$

Die Koeffizienten b und c sind nicht 0, da zum einen $\tilde{p}(x)$ ein nichttriviales Absolutglied besitzt ($x \nmid \bar{p}(x)$) und zum anderen n der Grad von $\tilde{r}(x)$ ist. Folglich ist die q-Dispersionsmenge leer, wenn es kein $j \in \mathbb{N}_0$ mit $(q^n)^j = \frac{ad}{bc}$ gibt. Aus diesem Grund überprüft man zunächst, ob ein solches $j \in \mathbb{N}_0$ existiert, bevor man versucht die Gleichheit (4.10) nachzuweisen. Um die q-Dispersionsmenge der beiden Polynome $p(x)$ und $r(x)$ zu bestimmen, werden diese Schritte für jedes Paar irreduzibler Faktoren durchgeführt und die erhaltenen Werte in der Menge DS gesammelt. □

4.3.2 Nennerschranke rationaler Lösungen

An dieser Stelle möchten wir einen Algorithmus vorstellen, der uns eine Nennerschranke rationaler Lösungen von q-holonomen Rekursionsgleichungen liefert. Definieren wir zunächst, was wir genau unter einer Nennerschranke verstehen wollen.

Definition 4.26 Sei $f(x)$ eine rationale Lösung einer q-holonomen Rekursionsgleichung. Eine *Nennerschranke* von $f(x)$ ist ein Polynom $v(x)$ aus $\mathbb{F}[x]$, so dass $f(x)v(x) \in \mathbb{F}[x]$ gilt.

Mit dem Wissen aus dem vorigen Abschnitt können wir nun eine Nennerschranke für eine rationale Lösung (ohne spezielle Singularität) einer q-holonomen Rekursionsgleichung unter Modifikation des Satzes des klassischen Falls aus [Wei01] angeben.

Satz 4.27 Sei

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0 \tag{4.11}$$

eine q-holonome Rekursionsgleichung und $N := \mathrm{disp}_q(a_n(q^{-n}x), a_0(x))$. Dann ist

$$v(x) := \gcd\left(\prod_{i=0}^{N} \varepsilon_q^{-(n+i)} a_n(x), \prod_{i=0}^{N} \varepsilon_q^i a_0(x)\right) \tag{4.12}$$

eine Nennerschranke aller rationalen Lösungen $f(x) = \frac{s(x)}{t(x)}$ der q-holonomen Rekursionsgleichung mit $s, t \in \mathbb{F}[x]$ und $x \nmid t(x)$.

Beweis Wir können die q-holonome Rekursionsgleichung (4.11) wie folgt umstellen

$$\varepsilon_q^n f(x) = -\frac{a_0(x)f(x) + a_1(x)\varepsilon_q f(x) + \ldots + a_{n-1}(x)\varepsilon_q^{n-1} f(x)}{a_n(x)}.$$

Wenden wir nun die Operatoren $\varepsilon_q^{-1}, \varepsilon_q^{-2}, \ldots, \varepsilon_q^{-N}$ auf diese Gleichung an, so erhalten wir

$$\begin{aligned}
\varepsilon_q^{n-1} f(x) &= -\frac{a_0(q^{-1}x)\varepsilon_q^{-1} f(x) + a_1(q^{-1}x)f(x) + \ldots + a_{n-1}(q^{-1}x)\varepsilon_q^{n-2} f(x)}{a_n(q^{-1}x)} \\
\varepsilon_q^{n-2} f(x) &= -\frac{a_0(q^{-2}x)\varepsilon_q^{-2} f(x) + a_1(q^{-2}x)\varepsilon_q^{-1} f(x) + \ldots + a_{n-1}(q^{-2}x)\varepsilon_q^{n-3} f(x)}{a_n(q^{-2}x)} \\
&\vdots \\
\varepsilon_q^{n-N} f(x) &= -\frac{a_0(q^{-N}x)\varepsilon_q^{-N} f(x) + a_1(q^{-N}x)\varepsilon_q^{-N+1} f(x) + \ldots + a_{n-1}(q^{-N}x)\varepsilon_q^{n-N-1} f(x)}{a_n(q^{-N}x)}.
\end{aligned}$$

Die rechten Seiten dieser Gleichungen setzt man nun in die erste Gleichung ein und erhält

$$\varepsilon_q^n f(x) = \frac{\sum_{k=0}^{n-1} \tilde{a}_k(x)\varepsilon_q^{k-N} f(x)}{\prod_{i=0}^{N} a_n(q^{-i}x)} \quad \text{mit} \quad \tilde{a}_k \in \mathbb{F}[x].$$

Nach Definition der q-Dispersion N sind die Pole von $\varepsilon_q^n f(x)$ verschieden von den Polen von $\varepsilon_q^{n-1-N} f(x), \ldots, \varepsilon_q^{-N} f(x)$ (sie sind mehr als N q-Shifts voneinander entfernt). Folglich sind die Pole von $f(x)$ bestimmt durch das Produkt der $a_n(q^{-i}x)$ mit $i = 0, \ldots, N$. Sei nun also $v(x)$ eine

Nennerschranke einer rationalen Lösung $f(x) = \frac{s(x)}{t(x)}$ mit obigen Voraussetzungen, dann gilt

$$\varepsilon_q^n v(x) \mid \prod_{i=0}^{N} a_n(q^{-i}x) \quad \text{bzw.} \quad v(x) \mid \prod_{i=0}^{N} a_n(q^{-(n+i)}x).$$

Analog zeigt man nach Umstellen der Rekursionsgleichung (4.11) nach $f(x)$ und durch Anwenden der Operatoren $\varepsilon_q, \ldots, \varepsilon_q^N$

$$v(x) \mid \prod_{i=0}^{N} a_0(q^i x)$$

und somit folgt die Behauptung. □

Der Fall, bei dem das Nennerpolynom einer rationalen Lösung $f(x)$ durch x teilbar ist, muss gesondert betrachtet werden, da wir die Argumentationen des Beweises von Satz 4.27 hier nicht anwenden können. Grund dafür ist, dass $f(x)$ eine spezielle Singularität bei 0 besitzt und Aussagen, wie z.B. Lemma 4.20, dann nicht mehr gültig sind. Für den Fall, dass x^j mit $0 \neq j \in \mathbb{Z}$ ein Faktor von $f(x)$ ist, formulieren wir

Satz 4.28 Sei $f(x)$ eine rationale Lösung der q-holonomen Rekursion $Lf(x) := \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ der Form $f(x) = x^j \frac{s(x)}{t(x)}$ mit $j \in \mathbb{Z}$, $s(x), t(x) \in \mathbb{F}[x]$, $\gcd(s(x), t(x)) = 1$, $x \nmid s(x)$ und $x \nmid t(x)$. Dann gilt

$$\mathrm{P}_{L, v_{\mathrm{ldeg}}, 0}\left(q^j\right) = 0.$$

Beweis Wir betrachten das q-Zertifikat $r(x)$ von $f(x)$ und erhalten

$$r(x) = \frac{\varepsilon_q f(x)}{f(x)} = q^j \frac{s(qx)}{t(qx)} \frac{t(x)}{s(x)} = q^j x^0 \left(1 + \mathcal{O}\left(x^1\right)\right).$$

Mit Satz 4.7 folgt dann die Behauptung. □

Mit Algorithmus 4.5 aus [Abr95] können wir die Nennerschranke $v(x)$ aus Satz 4.27 bestimmen.

Algorithmus 4.5 Bestimmung einer Nennerschranke aller rationalen Lösungen einer q-holonomen Rekursionsgleichung (`qUniversalDenominator`)

Eingabe : Eine q-holonome Rekursion $Lf(x) := \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$

Ausgabe : Eine Nennerschranke aller rationalen Lösungen $f(x)$ der q-holonomen Rekursion

begin
- $g(x) \leftarrow \gcd(\varepsilon_q^{-n} a_n(x), \ldots, \varepsilon_q^{-1} a_1(x), a_0(x))$
- $p(x) \leftarrow \frac{\varepsilon_q^{-n} a_n(x)}{g(x)}$
- $r(x) \leftarrow \frac{a_0(x)}{g(x)}$
- $DS \leftarrow \mathrm{Disp}_q(p(x), r(x))$ unter Verwendung von Algorithmus 4.4
- **for** $i \in DS$ *(in absteigender Reihenfolge)* **do**
  - $d_i(x) \leftarrow \gcd(p(x), \varepsilon_q^i r(x))$
  - $p(x) \leftarrow \frac{p(x)}{d_i(x)}$
  - $r(x) \leftarrow \frac{r(x)}{\varepsilon_q^{-i} d_i(x)}$
- **end**
- $j \leftarrow \max\{0\} \cup \left\{m \in \mathbb{N} \mid \mathrm{P}_{L, v_{\mathrm{ldeg}}, 0}\left(q^{-m}\right) = 0\right\}$
- **return** $x^j g(x) \prod_{i \in DS} \prod_{k=0}^{i} \varepsilon_q^{-k} d_i(x)$

end

Beweis Zum Beweis verwenden wir die gff-Notation (greatest factorial factorization) von Paule ([Pau95]) in der q-Variante ([PS95]) und übertragen den Beweis aus [Wei01] in den q-Fall. Definiere also $\mathrm{gff}_q(x^j; p_0(x), p_1(x), \ldots, p_m(x)) := x^j \prod_{i=0}^{m} \prod_{k=0}^{i} \varepsilon_q^{-k} p_i(x)$ mit q-normierten[8] Polynomen $p_i \in \mathbb{F}[x]$ und $j \in \mathbb{N}_0$. Wir setzen dabei voraus, dass in der Darstellung ausschließlich die „längsten Ketten“ auftreten[9]. Man kann dann jedes Polynom aus $\mathbb{F}[x]$ eindeutig als $c \cdot \mathrm{gff}_q(x^j; p_0(x), p_1(x), \ldots, p_m(x))$ mit geeigneten q-normierten $p_i \in \mathbb{F}[x], j \in \mathbb{N}_0$ und $c \in \mathbb{F}$ darstellen. Mit obigem $g(x)$ erhält man

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = \sum_{k=0}^{n} \varepsilon_q^k \left(\left(\varepsilon_q^{-k} a_k(x)\right) f(x)\right) = \sum_{k=0}^{n} \tilde{a}_k(x)\varepsilon_q^k \left(g(x) f(x)\right) \quad \text{mit} \quad \tilde{a}_k(x) = \frac{a_k(x)}{\varepsilon_q^k g(x)} \in \mathbb{F}[x].$$

In der Folge wird nun also die Nennerschranke von $g(x)f(x)$ bestimmt. Da wir an einer Nennerschranke für $f(x)$ interessiert sind, müssen wir am Ende die Nennerschranke mit dem Polynom $g(x)$ multiplizieren[10]. O.B.d.A. sei im Folgenden $g(x) = 1$. Um die Richtigkeit des Algorithmus zu beweisen, müssen wir zeigen, dass $v(x) = \mathrm{gff}_q(1; d_0(x), \ldots, d_N(x))$ gilt, wobei $N := \mathrm{disp}_q(p(x), r(x))$ und $v(x)$ wie in (4.12) sei. Dabei ist zu bemerken, dass wir im Beweis beginnend mit N in absteigender Reihenfolge alle $d_i(x)$ wie oben bestimmen. Im Algorithmus hingegen lassen wir Schritte, bei denen nur der triviale größte gemeinsame Teiler in Zeile 7 existiert, weg. Wir können (4.12) als

$$v(x) = \gcd(\mathrm{gff}_q(1; 1, \ldots, 1, \underbrace{\varepsilon_q^{-n} a_n(x)}_{N+1\text{-te Pos.}}), \mathrm{gff}_q(1; 1, \ldots, 1, \underbrace{\varepsilon_q^{N} a_0(x)}_{N+1\text{-te Pos.}}))$$

in gff_q-Notation schreiben[11]. Den in den Zeilen 7-9 bestimmten Polynomen geben wir für jeden Iterationsschritt einen Namen. Sei $p_{i-1}(x) = p_i(x)/d_i(x)$ und $r_{i-1}(x) = r_i(x)/\varepsilon_q^{-i} d_i(x)$. Zuvor wird $d_i(x)$ durch $d_i(x) = \gcd(p_i(x), \varepsilon_q^i r_i(x))$ bestimmt. Die Anfangswerte sind $p_N(x) = \varepsilon_q^{-n} a_n(x)$ und $r_N(x) = a_0(x)$. Es ergibt sich im ersten Schritt

$$\begin{aligned}
v(x) &= \gcd\left(\mathrm{gff}_q\left(1; 1, \ldots, 1, p_N(x)\right), \mathrm{gff}_q\left(1; 1, \ldots, 1, \varepsilon_q^N r_N(x)\right)\right) \\
&= \gcd\left(\mathrm{gff}_q\left(1; 1, \ldots, 1, d_N(x)\frac{p_N(x)}{d_N(x)}\right), \mathrm{gff}_q\left(1; 1, \ldots, 1, d_N(x)\frac{\varepsilon_q^N r_N(x)}{d_N(x)}\right)\right) \\
&= \gcd\left(\mathrm{gff}_q\left(1; 1, \ldots, 1, d_N(x)p_{N-1}(x)\right), \mathrm{gff}_q\left(1; 1, \ldots, 1, d_N(x)\varepsilon_q^N r_{N-1}(x)\right)\right) \\
&= \gcd\left(\mathrm{gff}_q\left(1; 1, \ldots, 1, d_N(x)p_{N-1}(x)\right), \mathrm{gff}_q\left(1; 1, \ldots, \varepsilon_q^{N-1} r_{N-1}(x), d_N(x)\right)\right) \\
&= \gcd\left(\mathrm{gff}_q\left(1; 1, \ldots, p_{N-1}(x), d_N(x)\right), \mathrm{gff}_q\left(1; 1, \ldots, \varepsilon_q^{N-1} r_{N-1}(x), d_N(x)\right)\right).
\end{aligned}$$

Die zweitletzte Gleichung erhält man durch folgende Überlegungen. Nach Definition von N und wegen $\gcd(p_{N-1}(x), \varepsilon_q^N r_{N-1}(x)) = 1$ gilt $\gcd(\varepsilon_q^{-k} p_{N-1}(x), \varepsilon_q^N r_{N-1}(x)) = 1$ für alle $k \in \mathbb{N}$. Folglich ist auch

$$\gcd\left(\prod_{k=0}^{N} \varepsilon_q^{-k} p_{N-1}(x), \varepsilon_q^N r_{N-1}(x)\right) = 1$$

und damit ist das Polynom $\varepsilon_q^N r_{N-1}(x)$ überflüssig in der zweiten gff_q-Darstellung der dritten Zeile. Wir streichen es aus dem Produkt

$$\prod_{k=0}^{N} \varepsilon_q^{-k}\left(\varepsilon_q^N r_{N-1}(x)\right) = \varepsilon_q^N r_{N-1}(x) \prod_{k=0}^{N-1} \varepsilon_q^{-k}\left(\varepsilon_q^{N-1} r_{N-1}(x)\right)$$

heraus und übrig bleibt ein Produkt mit N Faktoren, welches somit als Argument $\varepsilon_q^{N-1} r_{N-1}(x)$ an die N-te Position der gff_q-Darstellung tritt. Auf die gleiche Art und Weise erhalten wir die letzte Gleichung. Vollständige Induktion liefert schließlich $v(x) = \mathrm{gff}_q(1; d_0(x), \ldots, d_N(x))$. Den Faktor x^j der Nennerschranke bestimmen wir über Satz 4.28 und wählen dazu das maximale $j \in \mathbb{N}$ mit

[8] Ein Polynom $p(x)$ ist q-normiert, wenn $p(0) = 1$ gilt.

[9] Damit ist gemeint, dass man jeweils $\prod_{k=0}^{i} \varepsilon_q^{-k} p_i(x)$ mit maximalem i wählt und den Faktor nicht weiter in $\prod_{k=0}^{j} \varepsilon_q^{-k} p_i(x) \prod_{k=0}^{i-j-1} \varepsilon_q^{-k} p_i(q^{-(j+1)}x)$ mit $j < i$ zerlegt.

[10] Dies ist eine Nennerschranke für den Fall, dass die Lösung keine spezielle Singularität besitzt.

[11] Wir vernachlässigen im Folgenden die Voraussetzung, dass die auftretenden Polynome q-normiert sein müssen und damit auch den konstanten Vorfaktor c des Polynoms. Dadurch ändert sich der größte gemeinsame Teiler nicht.

$P_{L,v_{\deg},0}(q^{-j}) = 0$. Existiert kein $j \in \mathbb{N}$ mit dieser Eigenschaft, so ist das Nennerpolynom der Lösung nicht durch x teilbar und wir multiplizieren 1 zur Schranke. □

Eine andere Nennerschranke $v(x)$ einer rationalen Lösung $f(x)$ nach Mark van Hoeij ist in [Hoe98b] zu finden. Die Schranke ist dort eine rationale Funktion. Das hat den Vorteil, dass in einigen Fällen der Grad von $f(x)v(x) \in \mathbb{F}[x]$ verringert werden kann.

4.3.3 Rationale Lösungen q-holonomer Rekursionsgleichungen

Wir können nun den Algorithmus zur Bestimmung aller rationalen Lösungen q-holonomer Rekursionsgleichungen formulieren ([Abr95]).

Algorithmus 4.6 Bestimmung aller rationalen Lösungen einer q-holonomen Rekursionsgleichung (`qRationalSolveRE`)

Eingabe : Eine q-holonome Rekursion $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$

Ausgabe : Alle rationalen Lösungen $f(x)$ der q-holonomen Rekursion

```
1  begin
2      v(x) ← Nennerschranke aus Algorithmus 4.5
3      RE ← Σ_{k=0}^{n} a_k(x)/(ε_q^k v(x)) ε_q^k f(x) = 0
4      RE ← Multipliziere RE mit lcm(v(x), ε_q v(x), ..., ε_q^n v(x))
5      u(x) ← Bestimme alle polynomialen Lösungen von RE mit Algorithmus 4.3
6      if eine Lösung u(x) existiert then
7          return u(x)/v(x)
8      else
9          return „Es existieren keine rationalen Lösungen.“
10     end
11 end
```

Beweis Sei $f(x)$ eine rationale Lösung der q-holonomen Rekursionsgleichung

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0. \tag{4.13}$$

Da $v(x) \in \mathbb{F}[x]$ eine Nennerschranke von $f(x)$ ist, gilt $u(x) := f(x)v(x) \in \mathbb{F}[x]$. Betrachten wir nun die Rekursionsgleichung

$$\sum_{k=0}^{n} \frac{a_k(x)}{\varepsilon_q^k v(x)}\varepsilon_q^k g(x) = 0, \tag{4.14}$$

so ist $g(x) = u(x)$ eine polynomiale Lösung dieser Rekursion, denn es gilt

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = \sum_{k=0}^{n} a_k(x)\varepsilon_q^k \left(\frac{u(x)}{v(x)}\right) = \sum_{k=0}^{n} \frac{a_k(x)}{\varepsilon_q^k v(x)}\varepsilon_q^k u(x) = 0.$$

Umgekehrt existiert zu jeder polynomialen Lösung $u(x)$ der Rekursionsgleichung (4.14) mit $v(x) \in \mathbb{F}[x]$ eine rationale Lösung $f(x) := \frac{u(x)}{v(x)}$ der Rekursionsgleichung (4.13). Da man mit Algorithmus 4.3 alle polynomialen Lösungen $u(x)$ von (4.13) erhält, erhält man somit auch alle rationalen Lösungen von (4.14), namentlich $\frac{u(x)}{v(x)}$. □

Wir führen die letzte Maple-Sitzung von Seite 53 fort und bestimmen nun alle rationalen Lösungen der betrachteten q-holonomen Rekursionsgleichungen.

Maple-Sitzung 4.2 (Rationale Lösungen q-holonomer Rekursionsgleichungen)

```
> qRationalSolveRE(RE1,F(x));
```

$$\frac{a_0 + a_1 x^2}{x}$$

```
> qRationalSolveRE(RE2,F(x));
```

$$\frac{a_0}{x}$$

4.4 q-Hypergeometrische Lösungen q-holonomer Rekursionsgleichungen

Nachdem wir uns mit polynomialen und rationalen Lösungen von q-holonomen Rekursionsgleichungen beschäftigt haben, widmen wir uns in diesem Abschnitt den q-hypergeometrischen Lösungen.

4.4.1 Lokale Typen q-hypergeometrischer Lösungen

Wir betrachten zunächst die Struktur q-hypergeometrischer Lösungen von q-holonomen Rekursionsgleichungen genauer und untersuchen, welche Informationen uns der zugehörige Rekursionsoperator über die lokalen Typen der Lösung liefert.

Satz 4.29 Sei

$$L = \sum_{k=0}^{n} a_k(x)\varepsilon_q^k \in \mathbb{F}[x][\varepsilon_q].$$

Ferner sei $Lf(x) = 0$ die zugehörige q-holonome Rekursionsgleichung und $f(x)$ eine q-hypergeometrische Lösung mit q-Zertifikat $r(x) = \frac{s(x)}{t(x)}$, also $s(x), t(x) \in \mathbb{F}[x]$ und $\gcd(s(x), t(x)) = 1$. Dann gelten die Teilbarkeitsrelationen

$$s(x) \mid a_0(x) \qquad \text{und} \qquad t(x) \mid \varepsilon_q^{-(n-1)}\left(a_n(x)\right).$$

Beweis Sei $r(x) = \frac{s(x)}{t(x)}$ das q-Zertifikat der Lösung $f(x)$ mit obigen Eigenschaften. Dann ist $t(x)\varepsilon_q - s(x)$ ein polynomialer Rechtsfaktor von L. Nach Ausmultiplizieren mit dem dazugehörigen Linksfaktor $\sum_{k=0}^{n-1} b_k(x)\varepsilon_q^k$ erhält man

$$\underbrace{b_{n-1}(x)\varepsilon_q^{n-1}\left(t(x)\right)}_{a_n(x)}\varepsilon_q^n + \ldots \underbrace{-b_0(x)s(x)}_{a_0}$$

und damit nach Koeffizientenvergleich mit L und $(n-1)$-maligem Shiften beim höchsten Koeffizienten die Behauptung. □

Das folgende Korollar gibt uns untere und obere Schranken für die lokalen Typen einer q-hypergeometrischen Lösung an den endlichen Stellen.

Korollar 4.30 Seien die Voraussetzungen des vorigen Satzes gegeben. Dann gilt für den lokalen Typ an der Stelle $p \in \overline{\mathbb{F}}^*$ mit den Bezeichnungen aus Definition 4.9 und 4.14

$$-\sum_{\tilde{p} \in \mathfrak{Z}^p_{\varepsilon_q^{-(n-1)} a_n}} \operatorname{mult}(\tilde{p}) \le \operatorname{ltyp}_p(f(x)) \le \sum_{\tilde{p} \in \mathfrak{Z}^p_{a_0}} \operatorname{mult}(\tilde{p}).$$

Beweis Den lokalen Typ an der Stelle $p \in \overline{\mathbb{F}}^*$

$$\operatorname{ltyp}_p(f(x)) = \sum_{\tilde{p} \in \mathfrak{Z}^p_s} \operatorname{mult}(\tilde{p}) - \sum_{\tilde{p} \in \mathfrak{Z}^p_t} \operatorname{mult}(\tilde{p})$$

kann man mit Satz 4.29 nach oben abschätzen durch

$$\sum_{\tilde{p}\in\mathfrak{Z}_{a_0}^p} \operatorname{mult}(\tilde{p}),$$

denn $\operatorname{ltyp}_p(f(x))$ wird maximal, wenn alle Nullstellen von $a_0(x)$ aus $q^{\mathbb{Z}}p$ auch Nullstellen von $s(x)$ sind und das Polynom $t(x)$ keine Nullstellen aus $q^{\mathbb{Z}}p$ besitzt. Die untere Schranke erhält man analog. □

Wie kommt man nun mit Hilfe der charakteristischen Polynome an Kandidaten für $\operatorname{ltyp}_0(f(x))$ und $\operatorname{ltyp}_\infty(f(x))$? Antwort gibt uns das folgende Korollar von Satz 4.7.

Korollar 4.31 Für eine q-hypergeometrische Lösung $f(x)$ von $Lf(x) := \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ gilt

$$\operatorname{ltyp}_\infty(f(x)) \in \left\{(\tilde{c}_\infty, \tilde{v}_\infty) \mid N_{v_{\deg}}(L) \text{ besitzt Kante mit Steigung } -\tilde{v}_\infty \text{ und } \mathrm{P}_{L,v_{\deg},-\tilde{v}_\infty}(\tilde{c}_\infty) = 0\right\}$$

und

$$\operatorname{ltyp}_0(f(x)) \in \left\{(\tilde{c}_0, \tilde{v}_0) \mid N_{v_{\mathrm{ldeg}}}(L) \text{ besitzt Kante mit Steigung } -\tilde{v}_0 \text{ und } \mathrm{P}_{L,v_{\mathrm{ldeg}},-\tilde{v}_0}(\tilde{c}_0) = 0\right\}.$$

Beweis Wir zeigen die Aussage bzgl. der Bewertung $v_{\deg}$. Sei $r(x) = c\frac{s(x)}{t(x)}$ das q-Zertifikat einer q-hypergeometrischen Lösung von $Lf(x) = 0$ mit normierten und gekürzten Polynomen $s(x), t(x) \in \mathbb{F}[x]$ und $c \in \mathbb{F}^*$. Dann gibt es nach Satz 4.7 eine Kante des q-Newton-Polygons $N_{v_{\deg}}(L)$ mit Steigung $w = -(v_{\deg}(s(x)) - v_{\deg}(t(x)))$. Wiederum nach Satz 4.7 ist c eine Nullstelle des charakteristischen Polynoms $\mathrm{P}_{L,v_{\deg},w}(T)$. Es gilt offenbar nach Definition des lokalen Typs im Unendlichen $v_\infty = -w$ und $c_\infty = c$. Aus diesem Grund ist die Kandidatenmenge für $\operatorname{ltyp}_\infty(f(x))$ gegeben durch die Paare $(\tilde{c}_\infty, \tilde{v}_\infty)$, wobei $-\tilde{v}_\infty$ alle Kantensteigungen des q-Newton-Polygons durchläuft und $\tilde{c}_\infty$ alle Nullstellen der charakteristischen Polynome dieser Steigungen. Der Beweis bzgl. der Bewertung v_{ldeg} verläuft analog. □

Definition 4.32 Wir bezeichnen die *Kandidatenmengen für den lokalen Typ* im Unendlichen bzw. in 0 einer q-hypergeometrischen Lösung $f(x)$ einer q-holonomen Rekursionsgleichung aus Korollar 4.31 mit $\overline{\operatorname{ltyp}}_\infty(f(x))$ bzw. $\overline{\operatorname{ltyp}}_0(f(x))$. Ferner schreiben wir für die Kandidatenmenge für den lokalen Typ an der Stelle $p \in \overline{\mathbb{F}}^*$ von $f(x)$ (die Teilmenge von $\mathbb{Z}$, die beschränkt ist durch die Grenzen aus Korollar 4.30) auch $\overline{\operatorname{ltyp}}_p(f(x))$.

Beispiel 4.33 Wir führen Beispiel 4.6 fort und erhalten mit Satz 4.7 für den lokalen Typ im Unendlichen einer q-hypergeometrischen Lösung $f(x)$ von L folgende Kandidaten

$$\overline{\operatorname{ltyp}}_\infty(f(x)) = \left\{(q^{-2}, 1), (q^2, 0), (q^{-\frac{7}{2}}, -1), (-q^{-\frac{7}{2}}, -1), (-q^{-6}, -2)\right\}.$$

4.4.2 q-Holonome Rekursionsgleichungen vom q-hypergeometrischen Typ

Das Lösen von q-holonomen Rekursionsgleichungen erster Ordnung ist besonders einfach, da der Quotient der Koeffizientenpolynome der Rekursion im Wesentlichen dem q-Zertifikat der q-hypergeometrischen Lösung entspricht. Wie wir in diesem Abschnitt sehen werden, können wir aus dem q-Zertifikat die entsprechende q-hypergeometrische Lösung ermitteln. Wir können aber auch andere Lösungstypen von Rekursionen höherer Ordnung m, bei denen lediglich zwei Summanden auftreten, bestimmen. Aus diesem Grund betrachten wir zunächst

Definition 4.34 Eine q-holonome Rekursionsgleichung ist vom *q-hypergeometrischen Typ*, wenn die Rekursion die Form

$$t(x)\varepsilon_q^m f(x) - s(x)f(x) = 0 \quad \text{oder} \quad \frac{\varepsilon_q^m f(x)}{f(x)} = \frac{s(x)}{t(x)}$$

bzw.

$$a_m(q^j)c_{j+m} + a_0(q^j)c_j = 0 \quad \text{oder} \quad \frac{c_{j+m}}{c_j} = -\frac{a_0(q^j)}{a_m(q^j)}$$

mit $s(x), t(x) \in \mathbb{F}[x]$ bzw. $a_0, a_m \in \mathbb{F}[q^j]$ und $m \in \mathbb{N}$ besitzt. Den Term $f(x)$ bzw. c_j nennt man dann auch m-q-hypergeometrisch bzw. vom q-hypergeometrischen Typ *mit Symmetriezahl* m.

Wir sehen nun, wie man alle m-q-hypergeometrischen Lösungen einer q-holonomen Rekursionsgleichung vom q-hypergeometrischen Typ bestimmt.

Satz 4.35 Sei

$$a_m(q^j)c_{j+m} + a_0(q^j)c_j = 0$$

eine q-holonome Rekursionsgleichung vom q-hypergeometrischen Typ in additiver Form mit $a_0, a_m \in \mathbb{F}[q^j]$ und $m \in \mathbb{N}$. Die allgemeine Lösung der Rekursion ist gegeben durch

$$c_{mj+k} = \frac{\gamma^j \left(q^\lambda\right)^{kj+m\binom{j}{2}} \prod_{i=1}^{u} \beta_i^{j-s_k} \left(-\frac{\alpha_i}{\beta_i} q^{n_i(ms_k+k)}; q^{n_i m}\right)_{j-s_k}}{\gamma^{s_k} \left(q^\lambda\right)^{ks_k+m\binom{s_k}{2}} \prod_{i=1}^{v} \hat{\beta}_i^{j-s_k} \left(-\frac{\hat{\alpha}_i}{\hat{\beta}_i} q^{\hat{n}_i(ms_k+k)}; q^{\hat{n}_i m}\right)_{j-s_k}} c_{ms_k+k} \tag{4.15}$$

mit

$$s_k := \min\left\{ j_0 \in \mathbb{N}_0 \;\middle|\; \hat{\alpha}_i \left(q^{mj+k}\right)^{\hat{n}_i} + \hat{\beta}_i \neq 0 \text{ für alle } i = 1, \ldots, v \text{ und } j \geq j_0 \right\}$$

für $k = 0, \ldots, m-1$ und $j \geq s_k$, wobei

$$\frac{c_{j+m}}{c_j} = -\frac{a_0(q^j)}{a_m(q^j)} = \frac{\prod_{i=1}^{u} \left(\alpha_i \left(q^j\right)^{n_i} + \beta_i\right)}{\prod_{i=1}^{v} \left(\hat{\alpha}_i \left(q^j\right)^{\hat{n}_i} + \hat{\beta}_i\right)} \gamma \left(q^j\right)^\lambda$$

mit $u, v \in \mathbb{N}_0$, $n_i, \hat{n}_i \in \mathbb{N}$, $\lambda \in \mathbb{Z}$ und $\alpha_i, \hat{\alpha}_i, \beta_i, \hat{\beta}_i, \gamma \in \overline{\mathbb{F}}^*$. Bezeichne dabei $\overline{\mathbb{F}}$ einen geeigneten Erweiterungskörper von $\mathbb{F}$.

Beweis Da $\frac{c_{j+m}}{c_j} = -\frac{a_0(q^j)}{a_m(q^j)} \in \mathbb{F}(q^j)$ gilt, gibt es in einem geeigneten Erweiterungskörper $\overline{\mathbb{F}}$ von $\mathbb{F}$ eine Faktorisierung der Zähler- und Nennerpolynome der Form

$$\frac{c_{j+m}}{c_j} = -\frac{a_0(q^j)}{a_m(q^j)} = \frac{\prod_{i=1}^{u} \left(\alpha_i \left(q^j\right)^{n_i} + \beta_i\right)}{\prod_{i=1}^{v} \left(\hat{\alpha}_i \left(q^j\right)^{\hat{n}_i} + \hat{\beta}_i\right)} \gamma \left(q^j\right)^\lambda \tag{4.16}$$

mit $u, v \in \mathbb{N}_0$, $n_i, \hat{n}_i \in \mathbb{N}$, $\lambda \in \mathbb{Z}$ und $\alpha_i, \hat{\alpha}_i, \beta_i, \hat{\beta}_i, \gamma \in \overline{\mathbb{F}}^*$. Betrachten wir nun die zwei verschiedenen Arten von Faktoren, die in dem obigen Quotienten auftreten, gesondert. Seien dazu $k, s, j \in \mathbb{N}_0$ mit $k < m$ und $s < j$.

- $\frac{c_{j+m}}{c_j} = \alpha \left(q^j\right)^n + \beta$ mit $n \in \mathbb{N}$ und $\alpha, \beta \in \overline{\mathbb{F}}^*$

$$\begin{aligned}
c_{j+m} &= \left(\alpha \left(q^j\right)^n + \beta\right) c_j && \Big| \; j \to m(j-1)+k \\
c_{mj+k} &= \left(\alpha \left(q^{m(j-1)+k}\right)^n + \beta\right) c_{m(j-1)+k} && \Big| \; (j-s-1)\text{-malige Anwendung} \\
c_{mj+k} &= \prod_{i=0}^{j-s-1} \left(\alpha \left(q^{m(i+s)+k}\right)^n + \beta\right) c_{ms+k} \\
c_{mj+k} &= \textstyle\prod_{i=0}^{j-s-1} \beta \left(1 + \frac{\alpha}{\beta} q^{n(ms+k)} \left(q^{nm}\right)^i\right) c_{ms+k} = \beta^{j-s} \left(-\frac{\alpha}{\beta} q^{n(ms+k)}; q^{nm}\right)_{j-s} c_{ms+k}
\end{aligned}$$

- $\frac{c_{j+m}}{c_j} = \gamma \left(q^j\right)^\lambda$ mit $\gamma \in \overline{\mathbb{F}}^*$ und $\lambda \in \mathbb{Z}$

$$\begin{aligned}
c_{j+m} &= \gamma \left(q^j\right)^\lambda c_j && \Big| \; j \to m(j-1)+k \\
c_{mj+k} &= \gamma \left(q^{m(j-1)+k}\right)^\lambda c_{m(j-1)+k} \\
c_{mj+k} &= \gamma q^{\lambda k} \left(q^{\lambda m}\right)^{j-1} c_{m(j-1)+k} && \Big| \; (j-s-1)\text{-malige Anwendung} \\
c_{mj+k} &= \left(\gamma q^{\lambda k}\right)^{j-s} \left(q^{\lambda m}\right)^{\sum_{i=0}^{j-s-1}(j-1-i)} c_{ms+k} = \left(\gamma q^{\lambda k}\right)^{j-s} \left(q^{\lambda m}\right)^{\sum_{i=0}^{j-1} i - \sum_{i=0}^{s-1} i} c_{ms+k} \\
c_{mj+k} &= \left(\gamma q^{\lambda k}\right)^{j-s} \left(q^{\lambda m}\right)^{\binom{j}{2}-\binom{s}{2}} c_{ms+k} = \frac{\gamma^j \left(q^\lambda\right)^{kj+m\binom{j}{2}}}{\gamma^s \left(q^\lambda\right)^{ks+m\binom{s}{2}}} c_{ms+k}
\end{aligned}$$

Aus diesen beiden Spezialfällen lässt sich nun die Lösung für den allgemeinen Fall (4.16) konstruieren. Dabei ist darauf zu achten, dass die Faktoren im Nenner von (4.16) für einige $j \in \mathbb{N}_0$ verschwinden können. Aus diesem Grund erzeugen wir eine Lösung, die erst ab einem gewissen Index $s_k \in \mathbb{N}_0$ gültig ist. Dieser Index s_k berechnet sich für $k = 0, \ldots, m-1$ durch

$$s_k := \min \left\{ j_0 \in \mathbb{N}_0 \;\middle|\; \hat{\alpha}_i \left(q^{mj+k}\right)^{\hat{n}_i} + \hat{\beta}_i \neq 0 \text{ für alle } i = 1, \ldots, v \text{ und } j \geq j_0 \right\}.$$

Schließlich erhalten wir damit die Lösung (4.15) für $j \geq s_k$ und $k = 0, \ldots, m-1$. □

Da häufig der Fall eintritt, bei dem die Parameter die Größen $m = 1$, $k = 0$ und $s_0 = 0$ besitzen, bei dem also c_j ein q-hypergeometrischer Term ist, formulieren wir diesen Fall nochmals gesondert.

Korollar 4.36 Die Lösung der q-holonomen Rekursionsgleichung vom q-hypergeometrischen Typ

$$a_1(q^j)c_{j+1} + a_0(q^j)c_j = 0$$

ist gegeben durch

$$c_j = \gamma^j \left(q^{\hat{n}}\right)^{\binom{j}{2}} \frac{\prod_{i=1}^{u} \beta_i^j \left(-\frac{\alpha_i}{\beta_i}; q^{n_i}\right)_j}{\prod_{i=1}^{v} \hat{\beta}_i^j \left(-\frac{\hat{\alpha}_i}{\hat{\beta}_i}; q^{\hat{n}_i}\right)_j} c_0,$$

wobei

$$\frac{c_{j+1}}{c_j} = -\frac{a_0(q^j)}{a_1(q^j)} = \frac{\prod_{i=1}^{u} \left(\alpha_i \left(q^j\right)^{n_i} + \beta_i\right)}{\prod_{i=1}^{v} \left(\hat{\alpha}_i \left(q^j\right)^{\hat{n}_i} + \hat{\beta}_i\right)} \gamma \left(q^j\right)^{\hat{n}}$$

gelte.

Wenn wir also einen q-hypergeometrischen Term $f(x) \in \mathbb{F}(x)$ mit Zertifikat $r(x)$ besitzen, dann erhalten wir über einen geeigneten Erweiterungskörper von $\mathbb{F}$ und den Substitutionen $x \to q^j$ und $f(q^j) \to c_j$ die Darstellung

$$r(x) = \frac{f(qx)}{f(x)} = \frac{c_{j+1}}{c_j} = \frac{\prod_{i=1}^{u} \left(\alpha_i \left(q^j\right)^{n_i} + \beta_i\right)}{\prod_{i=1}^{v} \left(\hat{\alpha}_i \left(q^j\right)^{\hat{n}_i} + \hat{\beta}_i\right)} \gamma \left(q^j\right)^{\hat{n}}$$

und können so mit Korollar 4.36 und einem Startwert c_0 den q-hypergeometrischen Term explizit angeben.

Mit Hilfe von Satz 4.35 haben wir eine Lösungsformel für eine q-holonome Rekursionsgleichung vom q-hypergeometrischen Typ angeben können. Nun möchten wir uns dem Fall allgemeiner q-holonomer Rekursionsgleichungen widmen.

4.4.3 Der q-Petkovšek-Algorithmus

Wir halten zunächst fest, dass wir jede rationale Funktion über $\mathbb{F}$ in die folgende Normalform bringen können.

Satz 4.37 Sei $r \in \mathbb{F}(x)^*$ gegeben. Dann besitzt $r(x)$ die *Normalform*

$$r(x) = z \frac{\alpha(qx)}{\alpha(x)} \frac{\beta(x)}{\gamma(x)}$$

mit $z \in \mathbb{F}^*$ und $\alpha, \beta, \gamma \in \mathbb{F}[x]$ normiert und den Eigenschaften

$$\begin{aligned} \gcd(\beta(x), \varepsilon_q^k \gamma(x)) &= 1 \quad \text{für alle } k \in \mathbb{N} \\ \gcd(\beta(x), \alpha(x)) &= 1 \\ \gcd(\gamma(x), \varepsilon_q \alpha(x)) &= 1 \\ \alpha(0) &\neq 0. \end{aligned}$$

Den Beweis des Satzes kann man in [APP98] nachlesen. An dieser Stelle präsentieren wir nun den q-Petkovšek-Algorithmus aus [APP98].

Algorithmus 4.7 Bestimmung aller q-hypergeometrischen Lösungen einer q-holonomen Rekursionsgleichung (`qHypergeomSolveRE` mit `method=qPetkovsek`)

Eingabe : Eine q-holonome Rekursion $Lf(x) := \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$

Ausgabe : Alle q-Zertifikate q-hypergeometrischer Lösungen $f(x)$ der Rekursion

```
1  begin
2      C ← ∅
3      for jeden normierten Faktor β(x) von a_0(x) do
4          for jeden normierten Faktor γ(x) von ε_q^{-(n-1)} a_n(x) do
5              w ← -(υ_deg(β(x)) - υ_deg(γ(x)))
6              for jede Lösung z von P_{L,υ_deg,w}(z) = 0 do
7                  Bestimme polynomiale Lösungen α(x) von
                   Σ_{k=0}^{n} ( z^k a_k(x) Π_{i=0}^{k-1} ε_q^i β(x) Π_{i=k}^{n-1} ε_q^i γ(x) ) ε_q^k α(x) = 0 mit Algorithmus 4.3
8                  if eine Lösung existiert then
9                      Füge r(x) ← z α(qx)/α(x) β(x)/γ(x) zu C hinzu
10                 end
11             end
12         end
13     end
14     return C
15 end
```

Beweis Sei $r(x)$ das q-Zertifikat einer q-hypergeometrischen Lösung $f(x)$. Dann können wir $r(x)$ in Normalform schreiben

$$r(x) = z\frac{\alpha(qx)}{\alpha(x)}\frac{\beta(x)}{\gamma(x)}$$

wie in Satz 4.37. Mit Satz 4.29 folgt

$$\beta(x) \mid a_0(x) \qquad \text{und} \qquad \gamma(x) \mid \varepsilon_q^{-(n-1)} a_n(x).$$

Führen wir den Beweis von Satz 4.7 mit der Normalform durch, so erhalten wir

$$\sum_{k=0}^{n}\left(z^k a_k(x)\prod_{i=0}^{k-1}\varepsilon_q^i\beta(x)\prod_{i=k}^{n-1}\varepsilon_q^i\gamma(x)\right)\varepsilon_q^k\alpha(x) = 0. \tag{4.17}$$

Also durchläuft man im Algorithmus alle möglichen normierten Teiler von $a_0(x)$ und $\varepsilon_q^{-(n-1)}a_n(x)$ und sämtliche Nullstellen von $\mathrm{P}_{L,\upsilon_{\deg},w}$ mit $w = -(\upsilon_{\deg}(\beta(x)) - \upsilon_{\deg}(\gamma(x)))$ (siehe Satz 4.7) und stellt gemäß letzter Gleichung eine q-holonome Rekursionsgleichung auf. Die polynomialen Lösungen dieser Rekursion liefern das $\alpha(x)$ der Normalform und führen schließlich zu q-hypergeometrischen Lösungen, sofern welche existieren. □

Bemerkung 4.38 Wir können im Algorithmus 4.7 auch auf die Schreibweise des charakteristischen Polynoms $\mathrm{P}_{L,\upsilon_{\deg},w}(z)$ verzichten. Schreibe dazu Gleichung (4.17) als

$$\sum_{k=0}^{n}\left(z^k\sum_j u_{k,j}x^j\right)\varepsilon_q^k\alpha(x) = 0 \quad \text{mit} \quad u_{k,j}\in\mathbb{F}.$$

Dann ist z eine Lösung von $\sum_{k=0}^{n} u_{k,0}z^k = 0$. Also betrachtet man in Zeile 6 von Algorithmus 4.7, wie in [APP98] beschrieben, alle Nullstellen des Polynoms $\sum_{k=0}^{n} u_{k,0}z^k$. Das liefert den Original-q-

Petkovšek-Algorithmus aus [APP98]. Mit dem Konzept der charakteristischen Polynome des q-Newtonpolygons lässt sich dieser Schritt allerdings kurz, wie in Algorithmus 4.7 dargestellt, beschreiben, und das q-Newtonpolygon wird für die verbesserte und effizientere Version in Algorithmus 4.9 ohnehin benötigt.

Der q-Petkovšek-Algorithmus ist bei q-Rekursionsgleichungen, deren Leit- und/oder Fußkoeffizient mehrere Faktoren besitzen, sehr ineffizient. Das Problem bei diesem Algorithmus ist dann, dass man zu viele Kombinationen von Teilern $\beta(x)$ und $\gamma(x)$ betrachten muss und zu jeder Kombination eine Rekursionsgleichung zu lösen versucht. Die Laufzeit des Algorithmus ist deswegen exponentiell in der Anzahl der Faktoren. Aus diesem Grund wenden wir uns nun zunächst einem anderen Algorithmus zu, der dasselbe Problem behandelt, aber im klassischen Fall wesentlich effizienter ist.

4.4.4 Der q-Van-Hoeij-Algorithmus

Wir geben nun eine q-Variante des Van-Hoeij-Algorithmus ([Hoe98a],[CH05],[Hor08]) an.

Algorithmus 4.8 Bestimmung aller q-hypergeometrischen Lösungen einer q-holonomen Rekursionsgleichung (`qHypergeomSolveRE` mit `method=qVanHoeij`)

Eingabe : Eine q-holonome Rekursion $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$

Ausgabe : Alle q-Zertifikate q-hypergeometrischer Lösungen $f(x)$ der Rekursion

1 **begin**
2 $C \leftarrow \emptyset$
3 Bestimme alle Kandidatenmengen $\overline{\text{ltyp}}_\infty(f(x)), \overline{\text{ltyp}}_0(f(x))$ und $\overline{\text{ltyp}}_p(f(x))$ für alle $p \in \overline{\mathbb{F}}^*$
4 $\tilde{C} \leftarrow \left\{ \tilde{c}_\infty x^{\tilde{v}_0} \prod_{q^{\mathbb{Z}} p \in \left(\overline{\mathbb{F}}/\sim_{\varepsilon_q}\right)^*} (x-p)^{\tilde{v}_p} \;\middle|\; \tilde{v}_p \in \overline{\text{ltyp}}_p(f(x)), (\tilde{c}_\infty, *) \in \overline{\text{ltyp}}_\infty(f(x)), (*, \tilde{v}_0) \in \overline{\text{ltyp}}_0(f(x)) \right\}$
5 **for** *jedes* $\tilde{r}(x) \in \tilde{C}$ **do**
6 **if** *mind. eine der beiden q-Fuchs-Relationen* (4.6) *und* (4.7) *bzgl.* $\tilde{r}(x)$ *nicht erfüllt* **then**
7 **next**
8 **end**
9 $RE1 \leftarrow \varepsilon_q f(x) - \frac{1}{\tilde{r}(x)} f(x) = 0$
10 $RE2 \leftarrow \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$
11 $RE \leftarrow$ Bilde Produktrekursion von $RE1$ und $RE2$ mit Algorithmus 3.3
12 Bestimme alle rationalen Lösungen $s(x)$ von RE mit Algorithmus 4.6
13 **if** *eine Lösung existiert* **then**
14 Füge $\frac{\varepsilon_q s(x)}{s(x)} \tilde{r}(x)$ zu C hinzu
15 **end**
16 **end**
17 **return** C
18 **end**

Beweis Sei $f(x)$ eine q-hypergeometrische Lösung der gegebenen q-holonomen Rekursionsgleichung mit q-Zertifikat $r(x)$. Ferner sei $\tilde{r}(x) \in \tilde{C}$, also $\tilde{r}(x)$ ein q-Zertifikat eines q-hypergeometrischen Terms $g(x)$. Dann ist $g(x)$ nach Konstruktion vom selben Typ wie $f(x)$ (siehe Satz 4.18). Sind beide q-Fuchs-Relationen für $g(x)$ gültig, so könnte $\tilde{r}(x)$ zu einem q-Zertifikat einer q-hypergeometrischen Lösung beitragen, andernfalls betrachten wir den nächsten Kandidaten aus der Menge $\tilde{C}$. Die in der elften Zeile bestimmte Produktrekursion RE hat nach Konstruktion die rationale Lösung

$s(x) = \frac{1}{g(x)} \cdot f(x) = \frac{f(x)}{g(x)}$, falls überhaupt eine Lösung existiert. Bilden wir nun $\frac{\varepsilon_q s(x)}{s(x)} \tilde{r}(x)$, so erhalten wir

$$\frac{\varepsilon_q s(x)}{s(x)} \tilde{r}(x) = \frac{g(x)\varepsilon_q f(x)}{f(x)\varepsilon_q g(x)} \frac{\varepsilon_q g(x)}{g(x)} = \frac{\varepsilon_q f(x)}{f(x)} = r(x),$$

also das gewünschte q-Zertifikat der q-hypergeometrischen Lösung $f(x)$. □

Bemerkung 4.39 In Zeile 3 von Algorithmus 4.8 muss man für jedes $p \in \overline{\mathbb{F}}^*$ (unendlich viele) die Kandidatenmenge $\overline{\text{ltyp}}_p(f(x))$ bestimmen. Zur Durchführung betrachtet man an dieser Stelle lediglich Nullstellen $p \in \overline{\mathbb{F}}^*$ von $a_0(x)$ und $a_n(x)$ (also endlich viele), da nur diese zu nichttrivialen lokalen Typen einer q-hypergeometrischen Lösung $f(x)$ führen. Der lokale Typ an anderen endlichen Stellen $p \in \overline{\mathbb{F}}^*$ von $f(x)$ ist 0 und man setzt demzufolge $\overline{\text{ltyp}}_p(f(x)) = \{0\}$. Folglich ist in Zeile 4 das Produkt über alle ε_q-Äquivalenzklassen (siehe Definition 4.8) ein endliches Produkt.

4.4.5 Der modifizierte q-Petkovšek-Algorithmus

Wir betrachten nun noch einmal den q-Petkovšek-Algorithmus und versuchen möglichst viele Kombinationen von Faktoren $\beta(x)$ und $\gamma(x)$ von vornherein auszuschließen. Wir greifen dazu die Idee aus [Hor08] auf und verwenden erneut das q-Newton-Polygon.

Satz 4.40 Sei

$$r(x) = z \frac{\alpha(qx)}{\alpha(x)} \frac{\beta(x)}{\gamma(x)}$$

ein q-Zertifikat einer q-hypergeometrischen Lösung $f(x)$ in Normalform der q-holonomen Rekursionsgleichung

$$\sum_{k=0}^{n} a_k(x) \varepsilon_q^k f(x) = 0$$

mit $z \in \mathbb{F}^*$ und α, β und $\gamma \in \mathbb{F}[x]$ normiert wie in Satz 4.37. Dann gilt

$$\text{ltyp}_0(f(x)) = (c_0, v_0) = \left(z \frac{\text{tcoeff}_x(\beta(x))}{\text{tcoeff}_x(\gamma(x))}, \text{ldeg}_x(\beta(x)) - \text{ldeg}_x(\gamma(x)) \right)$$

und

$$\text{ltyp}_\infty(f(x)) = (c_\infty, v_\infty) = \left(z q^{\deg_x(\alpha(x))}, -(\deg_x(\beta(x)) - \deg_x(\gamma(x))) \right).$$

Ferner gilt

$$\frac{\text{tcoeff}_x(\beta(x))}{\text{tcoeff}_x(\gamma(x))} \sim_{\varepsilon_q} \prod_{q^{\mathbb{Z}} p \in \left(\overline{\mathbb{F}}/\sim_{\varepsilon_q}\right)^*} (-p)^{\text{ltyp}_p(f(x))}.$$

Beweis Man verwende die Formeln aus Lemma 4.11 zur Bestimmung des lokalen Typs eines q-hypergeometrischen Terms und berücksichtigt, dass alle auftretenden Polynome normiert sind. Die letzte Äquivalenz ergibt sich durch Anwendung der zweiten q-Fuchs-Relation und Kürzen von z. □

Man beachte, dass c_0, v_0 und v_∞ nicht von dem später zu bestimmenden Polynom $\alpha(x)$ abhängen. Lediglich der Grad des Polynoms $\alpha(x)$ fließt in die Größe c_∞. Beim modifizierten q-Petkovšek-Algorithmus, der in Algorithmus 4.9 beschrieben wird, filtern wir alle Polynome $\beta(x)$ und $\gamma(x)$ gemäß Korollar 4.31 und Satz 4.40 heraus, die nicht mehr Teil eines q-Zertifikats einer q-hypergeometrischen Lösung sein können. Dadurch wird der Algorithmus, im Vergleich zum klassischen Petkovšek-Algorithmus, wesentlich effizienter, da zum Einen die q-holonome Rekursionsgleichung, die große Koeffizienten aufweisen kann, nicht aufgestellt und zum Anderen die polynomiale Lösungsmethode nicht eingesetzt werden muss. Bei Rekursionsgleichungen, deren Leit- und Fußkoeffizient viele Faktoren besitzen, kommt der Performanzzuwachs besonders stark zum Ausdruck (siehe auch den Laufzeitvergleich im nächsten Abschnitt).

Algorithmus 4.9 Bestimmung aller q-hypergeometrischen Lösungen einer q-holonomen Rekursionsgleichung (modifiziert) (`qHypergeomSolveRE` mit `method=modqPetkovsek`)

Eingabe : Eine q-holonome Rekursion $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$

Ausgabe : Alle q-Zertifikate q-hypergeometrischer Lösungen $f(x)$ der Rekursion

begin
  $C \leftarrow \emptyset$
  for *jeden normierten Faktor* $\beta(x)$ *von* $a_0(x)$ **do**
    for *jeden normierten Faktor* $\gamma(x)$ *von* $\varepsilon_q^{-(n-1)} a_n(x)$ **do**
      $v_0 \leftarrow v_{\mathrm{ldeg}}(\beta(x)) - v_{\mathrm{ldeg}}(\gamma(x))$
      $v_\infty \leftarrow v_{\mathrm{deg}}(\beta(x)) - v_{\mathrm{deg}}(\gamma(x))$
      if $(*, v_0) \notin \overline{\mathrm{ltyp}}_0(f(x))$ *oder* $(*, v_\infty) \notin \overline{\mathrm{ltyp}}_\infty(f(x))$ **then**
        next
      end
      $c_0 \leftarrow \frac{\mathrm{tcoeff}_x(\beta(x))}{\mathrm{tcoeff}_x(\gamma(x))}$
      $c_\infty \leftarrow \frac{\mathrm{lcoeff}_x(\beta(x))}{\mathrm{lcoeff}_x(\gamma(x))}$
      if $(q^k c_0, v_0) \notin \overline{\mathrm{ltyp}}_0(f(x))\ \forall k \in \mathbb{Z}$ *oder* $(q^j c_\infty, v_\infty) \notin \overline{\mathrm{ltyp}}_\infty(f(x))\ \forall j \in \mathbb{Z}$ **then**
        next
      end
      for *jede Lösung* z *von* $\mathrm{P}_{L, v_{\mathrm{deg}}, -v_\infty}(z) = 0$ **do**
        Bestimme polynomiale Lösungen $\alpha(x)$ von
        $\sum_{k=0}^{n} \left(z^k a_k(x) \prod_{i=0}^{k-1} \varepsilon_q^i \beta(x) \prod_{i=k}^{n-1} \varepsilon_q^i \gamma(x) \right) \varepsilon_q^k \alpha(x) = 0$ mit Algorithmus 4.3
        if *eine Lösung existiert* **then**
          Füge $r(x) \leftarrow z \frac{\alpha(qx)}{\alpha(x)} \frac{\beta(x)}{\gamma(x)}$ zu C hinzu
        end
      end
    end
  end
  return C
end

Schließlich bestimmen wir, nachdem wir bei den letzten beiden Maple-Sitzungen von Seite 53 und 60 die polynomialen und rationalen Lösungen zweier q-holonomer Rekursionsgleichungen berechnet haben, nun die q-hypergeometrischen Lösungen. Für die erste Rekursion gibt es keine weiteren Lösungen als die polynomiale Lösung ax bzw. die rationale Lösung $ax + bx^{-1}$. Bei der zweiten Rekursion können wir, neben der rationalen Lösung ax^{-1}, eine q-hypergeometrische Lösung mit q-Zertifikat x finden. Siehe Kapitel 6 für eine Beschreibung der Prozeduren `qHypergeomSolveRE` und `qREtoqRE`. Letztere transformiert q-holonome Rekursionen in multiplikativer Form in additive Form und umgekehrt.

Maple-Sitzung 4.3 (q-Hypergeometrische Lösungen q-holonomer Rekursionsgleichungen)

```
> qHypergeomSolveRE(RE1,F(x));
```

$$[q, q^{-1}]$$

```
> RE:=qREtoqRE(RE1,F(x),A(k));
```

$$RE := -qA(k) + \left(1 + q^2\right) A(k+1) - qA(k+2) = 0$$

```
> qHypergeomSolveRE(RE,A(k));
```

$$a_0 q^k + \frac{a_1}{q^k}$$

```
> qHypergeomSolveRE(RE2,F(x));
```

$$[q^{-1}, x]$$

```
> RE:=qREtoqRE(RE2,F(x),A(k));
```

$$RE := q^k\left(-1+q^k q^2\right)A(k) + \left(1-\left(q^k\right)^2 q^3\right)A(k+1) + q\left(q\,q^k-1\right)A(k+2) = 0$$

```
> qHypergeomSolveRE(RE,A(k));
```

$$\frac{a_0}{q^k} + a_1 q^{\frac{1}{2}k(k-1)}$$

4.5 Laufzeitvergleich

Wir vergleichen unsere Algorithmen aus `qFPS` mit entsprechenden Funktionalitäten aus dem internen Maple-Package `QDifferenceEquations` und dem Maple-Package `qsum` ([BK99]). Dazu verwenden wir Operatoren der Ordnung 3, deren Leit- bzw. Fußkoeffizienten den Grad $\lfloor \frac{m+1}{2} \rfloor$ bzw. $\lfloor \frac{m}{2} \rfloor + 1$ besitzen, wobei $m \in \mathbb{N}_0$ gelte und deren q-hypergeometrische Lösung das q-Zertifikat x besitzt. Je größer die Anzahl der Faktoren des Fuß- und Leitkoeffizienten des q-Rekursionsopera-

	QDifferenceEquations	qsum	qFPS		
m	QHypergeometricSolution	qrecsolve	qHypergeomSolveRE mit method=		
			qPetkovsek	qVanHoeij	modqPetkovsek
0	000.160	000.077	000.083	000.383	000.064
1	000.087	000.047	000.053	000.327	000.064
2	000.096	000.050	000.099	000.116	000.060
3	000.405	000.117	000.137	000.228	000.073
4	000.614	000.233	000.209	000.156	000.091
5	000.808	000.524	000.365	004.723	000.128
6	000.900	000.928	000.742	000.604	000.130
7	001.397	001.403	001.211	003.070	000.239
8	002.359	004.078	003.166	002.304	000.215
9	003.715	010.542	006.899	090.627	000.644
10	004.201	017.851	015.004	011.110	000.519
11	005.539	036.566	020.816	086.444	001.846
12	010.897	164.357	088.682	122.732	001.436
13	014.683	607.680	252.462	483.771	006.647
14	016.792	609.486	703.804	1220.991	004.938

Tabelle 4.1 Laufzeitvergleich verschiedener Maple-Prozeduren zur Lösung q-holonomer Rekursionen bzgl. Operator (4.18)

tors, desto länger ist die Rechenzeit, da die Komplexität exponentiell in der Anzahl der Faktoren wächst. Aus diesem Grund betrachten wir die folgenden Operatoren in Abhängigkeit von m. Die Komplexität ist auch abhängig von der Struktur der Faktoren, z.B. macht es einen Unterschied, ob die Nullstellen der Faktoren paarweise ε_q-äquivalent sind oder nicht. In Tabelle 4.1 wurde der Laufzeitvergleich bzgl. des Operators

$$\left(\left(\prod_{i=1}^{\lfloor \frac{m+1}{2} \rfloor}(x+iq^i)\right)\varepsilon_q^2 + \prod_{i=1}^{\lfloor \frac{m}{2} \rfloor}(x-iq^i)\right)\cdot(\varepsilon_q - x)\,, \tag{4.18}$$

durchgeführt, dessen Koeffizienten Nullstellen besitzen, die nicht paarweise ε_q-äquivalent zueinander sind[12]. Betrachten wir den Operator

$$\left(\left(\prod_{i=1}^{\lfloor\frac{m+1}{2}\rfloor}(x-q^i)\right)\varepsilon_q^2+\prod_{i=1}^{\lfloor\frac{m}{2}\rfloor}(x-iq^i)\right)\cdot(\varepsilon_q-x)\,, \tag{4.19}$$

bei dem alle Faktoren des Leitkoeffizienten paarweise eine nichttriviale q-Dispersion besitzen, so ergibt sich Tabelle 4.2.

	QDifferenceEquations	qsum	qFPS		
m	QHypergeometricSolution	qrecsolve	qHypergeomSolveRE mit method=		
			qPetkovsek	qVanHoeij	modqPetkovsek
0	000.169	000.075	000.082	000.341	000.065
1	000.099	000.046	000.055	000.334	000.065
2	000.105	000.051	000.098	000.115	000.059
3	000.387	000.047	000.136	000.289	000.074
4	000.606	000.076	000.222	000.118	000.093
5	000.787	000.195	000.364	000.141	000.113
6	000.829	000.181	000.752	000.173	000.109
7	001.643	000.652	001.550	000.197	000.230
8	002.325	000.747	003.200	000.269	000.194
9	003.273	002.717	006.773	000.303	000.568
10	003.417	003.334	014.342	000.363	000.468
11	006.029	013.270	033.573	000.480	001.774
12	007.970	018.587	081.127	000.501	001.267
13	011.567	083.961	232.316	000.683	006.241
14	012.209	143.046	628.259	001.019	004.676

Tabelle 4.2 Laufzeitvergleich verschiedener Maple-Prozeduren zur Lösung q-holonomer Rekursionen bzgl. Operator (4.19)

Als Letztes betrachten wir den Operator

$$\left(\left(\prod_{i=1}^{\lfloor\frac{m+1}{2}\rfloor}\left(x-iq^{i-2}\right)\right)\varepsilon_q^2+\prod_{i=1}^{\lfloor\frac{m}{2}\rfloor}(x-iq^i)\right)\cdot(\varepsilon_q-x)\,, \tag{4.20}$$

bei dem alle Nullstellen des Leitkoeffizienten ε_q-äquivalent sind zu Nullstellen des Fußkoeffizienten und erhalten Tabelle 4.3.

Die Prozedur aus dem Package `QDifferenceEquations` ist nicht fehlerfrei implementiert[13]. Es gibt viele Rekursionsgleichungen, bei denen die Prozedur **keine** q-hypergeometrische Lösung findet, obwohl eine existiert. Eine q-holonome Rekursionsgleichung, bei der z.B. kein q-Zertifikat ermittelt wird, obwohl es derer sogar zwei gibt, ist in dem Beispiel der Verwendung von `qHypergeomSolveRE` in Kapitel 6 auf Seite 114 zu finden. Folglich ist anzunehmen, dass auch nicht alle Kombinationen, die auf Lösungen führen könnten, betrachtet werden. Diese Tatsache ist bei den gemessenen Zeiten von `QHypergeometricSolution` zu berücksichtigen. Die in `qFPS` enthaltene Prozedur `qHypergeomSolveRE` mit `method=modqPetkovsek` schneidet in allen drei Vergleichen sehr gut ab und unterstreicht die Voreinstellung des optionalen Parameters `method` auf diese Methode. Die Komplexität der q-Van-Hoeij-Methode ist stark von der Struktur der Rekursionsgleichung abhängig und erzielt nur bei ausgewählten Beispielen die schnellsten Zeiten. Die

[12] Der Laufzeitvergleich wurde auf einem MacBook Pro mit 2.33GHz Intel Core 2 Duo-Prozessor und 3GB RAM durchgeführt. Die Zeiten wurden in Sekunden gemessen.

[13] Maple-Version 12.01

gebildete Produktrekursion wird oftmals sehr groß und das Bestimmen der rationalen Lösungen dieser Rekursion dauert häufig sehr lang.

	QDifferenceEquations	qsum	qFPS		
m	QHypergeometricSolution	qrecsolve	qHypergeomSolveRE mit method=		
			qPetkovsek	qVanHoeij	modqPetkovsek
0	000.100	000.046	000.055	000.330	000.063
1	000.089	000.048	000.055	000.286	000.064
2	000.102	000.049	000.081	000.114	000.059
3	000.483	000.150	000.123	000.412	000.079
4	000.694	000.182	000.221	000.166	000.095
5	000.959	000.494	000.375	000.580	000.147
6	001.161	000.887	000.712	000.215	000.137
7	002.143	002.036	001.435	001.109	000.279
8	002.687	001.233	002.336	002.968	000.228
9	004.567	010.013	006.690	001.860	000.710
10	005.102	018.701	014.509	000.501	000.552
11	008.683	063.317	034.041	002.323	002.031
12	012.113	142.247	085.343	155.077	017.548
13	017.290	818.392	193.064	2070.761	030.317

Tabelle 4.3 Laufzeitvergleich verschiedener Maple-Prozeduren zur Lösung q-holonomer Rekursionen bzgl. Operator (4.20)

Während im klassischen Fall der Van-Hoeij-Algorithmus mit Abstand der schnellste Algorithmus zur Bestimmung hypergeometrischer Lösungen ist (der modifizierte Petkovšek-Algorithmus ist dort wesentlich langsamer), ist die Situation im q-Fall anders (siehe [Hor08]).

Die Operatoren, die wir an dieser Stelle betrachtet haben, treten in der Praxis selten auf. Trotzdem lässt sich auch beim Lösen komplizierter q-holonomer Rekursionen, die beim q-FPS-Algorithmus im nächsten Kapitel vorkommen, ein deutlicher Geschwindigkeitsvorteil bei `qHypergeomSolveRE` im Vergleich zu den anderen Prozeduren feststellen.

Kapitel 5

Algorithmen für q-hypergeometrische Potenzreihen

In diesem Kapitel geht es um die Bestimmung der Koeffizienten einer formalen Potenzreihe aus einer q-holonomen Rekursionsgleichung der Reihe. Wir stellen einen Algorithmus auf, mit dem wir in der Lage sind, zu jeder q-holonomen Rekursionsgleichung einer Potenzreihe bzgl. einer Polynombasis eine q-holonome Rekursionsgleichung für die Koeffizienten der Reihe zu bestimmen. Wenn diese Rekursionsgleichung eine q-hypergeometrische Lösung besitzt, so können wir diese mit Hilfe der Algorithmen des vorigen Kapitels finden. Ferner präsentieren wir einen Algorithmus, der umgekehrt eine q-holonome Rekursionsgleichung für die Koeffizienten in eine q-holonome Rekursionsgleichung für die Reihe konvertiert. Damit wird es uns u.a. möglich sein, algorithmisch eine q-holonome Rekursionsgleichung für die verallgemeinerte q-hypergeometrische Funktion zu bestimmen.

5.1 Algorithmen zur Bestimmung q-hypergeometrischer Potenzreihen

Wir betrachten zunächst die grundlegende Theorie aus [APR00], die wir in den folgenden Abschnitten benötigen und wenden diese auf den q-Fall an.

Definition 5.1 Bezeichne $\mathfrak{B} = \{P_j(x) \mid j \in \mathbb{N}_0\}$ eine Teilmenge von $\mathbb{F}[x]$ mit den Eigenschaften

(a) $\deg_x(P_j(x)) = j$ für alle $j \in \mathbb{N}_0$

(b) $P_j(x) \mid P_k(x)$ für alle $0 \leq j < k$ mit $j, k \in \mathbb{N}_0$

und $\mathfrak{L}(\mathbb{F}[x])$ die Menge aller linearen Operatoren von $\mathbb{F}[x]$. Wir sagen, $\mathfrak{B}$ ist *kompatibel zu* dem Operator $L \in \mathfrak{L}(\mathbb{F}[x])$, wenn es $i_0, i_1 \in \mathbb{N}_0$ und $a_{i,j} \in \mathbb{F}$ gibt mit

$$LP_j(x) = \sum_{i=-i_0}^{i_1} a_{i,j} P_{j+i}(x) \quad \text{mit} \quad P_k(x) = 0, \quad \text{wenn} \quad k < 0.$$

Ferner bezeichnen wir die Menge aller linearen Operatoren aus $\mathfrak{L}(\mathbb{F}[x])$, die zu einem gegebenen $\mathfrak{B}$ kompatibel sind, mit $\mathfrak{L}_{\mathfrak{B}}$.

Bemerkung 5.2 Die Menge $\mathfrak{B}$ ist offenbar wegen Eigenschaft (a) eine Basis von $\mathbb{F}[x]$. Wie man leicht verifizieren kann, sind $\mathbb{F}[x]$ und $\mathfrak{L}(\mathbb{F}[x])$ $\mathbb{F}$-Algebren[1]. In der Praxis werden wir nur kompatible Basen betrachten, bei denen $a_{i,j} = a_i(q^j)$ ist, d.h. bei denen

$$LP_j(x) = \sum_{i=-i_0}^{i_1} a_i(q^j) P_{j+i}(x)$$

[1] Sei $(\mathbb{K}, +, \odot)$ ein Körper. Eine (assoziative) $\mathbb{K}$-Algebra ist ein Ring $(V, +, \circ)$, bei dem zusätzlich $(V, +, \cdot)$ ein $\mathbb{K}$-Vektorraum ist und $(a \cdot v) \circ (b \cdot w) = (a \odot b) \cdot (v \circ w)$ für $a, b \in \mathbb{K}$ und $v, w \in V$ gilt.

mit $a_i \in \mathbb{F}(q^j)$ gilt.

Beispiel 5.3 Sei $\mathfrak{B}$ beliebig und $L \in \mathfrak{L}(\mathbb{F}[x])$ mit $LP_j(x) := xP_j(x)$ für $P_j(x) \in \mathfrak{B}$. Dann gilt $L \in \mathfrak{L}_{\mathfrak{B}}$, d.h. $\mathfrak{B}$ ist zu L kompatibel. Wir betrachten zum Beweis $LP_j(x)$. Da $\mathfrak{B}$ eine Basis von $\mathbb{F}[x]$ ist, gibt es $\tilde{a}_{i,j} \in \mathbb{F}$ mit

$$xP_j(x) = \sum_{i=0}^{j+1} \tilde{a}_{i,j} P_i(x). \tag{5.1}$$

Die linke Seite und auch die beiden Summanden $\tilde{a}_{j,j}P_j(x)$ und $\tilde{a}_{j+1,j}P_{j+1}(x)$ der rechten Seite sind wegen (b) durch $P_j(x)$ teilbar. Somit ist auch $Q(x) := \sum_{i=0}^{j-1} \tilde{a}_{i,j}P_i(x)$ durch $P_j(x)$ teilbar. Da der Grad von $Q(x)$ aber höchstens $j-1$ ist, muss $Q(x)$ das Nullpolynom sein. Das liefert die Darstellung

$$LP_j(x) = \tilde{a}_{j,j}P_j(x) + \tilde{a}_{j+1,j}P_{j+1}(x) = \sum_{i=0}^{1} a_{i,j}P_{j+i}(x) \quad \text{mit} \quad a_{i,j} = \tilde{a}_{j+i,j}.$$

L \ $P_j^a(x)$	$(x \ominus a)_q^j$	x^j	$(a \ominus x)_q^j$	$(x;q)_j$
x	$\left(\varepsilon + aq^j\right)(x \ominus a)_q^j$	εx^j	$\left(-\frac{1}{q^j}\varepsilon + \frac{a}{q^j}\right)(a \ominus x)_q^j$	$\left(-\frac{1}{q^j}\varepsilon + \frac{1}{q^j}\right)(x;q)_j$
D_q	$[j]_q\, \varepsilon^{-1}(x \ominus a)_q^j$	$[j]_q\, \varepsilon^{-1}x^j$	-	-
ε_q	$\left(aq^{j-1}(q^j-1)\varepsilon^{-1} + q^j\right)(x \ominus a)_q^j$	$q^j x^j$	-	-
$D_{q^{-1}}$	-	-	$-[j]_q\, \varepsilon^{-1}(a \ominus x)_q^j$	$-[j]_q\, \varepsilon^{-1}(x;q)_j$
$\varepsilon_{q^{-1}}$	-	-	$\left(\frac{a(q^j-1)}{q^j}\varepsilon^{-1} + \frac{1}{q^j}\right)(a \ominus x)_q^j$	$\left(\frac{q^j-1}{q^j}\varepsilon^{-1} + \frac{1}{q^j}\right)(x;q)_j$

Tabelle 5.1 Kompatible q-Operatoren L für Polynombasen $\mathfrak{B} = \{P_j^a(x) \mid j \in \mathbb{N}_0, a \in \mathbb{F}\}$

Bemerkung 5.4 Die Polynome $P_j^a(x)$ der Tabelle 5.1 erfüllen offenbar die Eigenschaften (a) und (b) von Definition 5.1. Die dritte bzw. die letzte Spalte ergibt sich jeweils durch die vorherige Spalte durch die Substitution $a = 0$ bzw. $a = 1$. Einträge, die mit einem Strich versehen sind, deuten an, dass es keine Darstellung gemäß Gleichung (5.1) gibt. Die Darstellungen von $LP_j^a(x)$ erhält man leicht, indem man $LP_j^a(x)$ als Polynom in $P_i^a(x)$ schreibt und dazu sukzessive die Koeffizienten (beginnend mit dem höchsten Koeffizienten) bestimmt. Die Resultate wurden in Operatorschreibweise (siehe auch Definition 5.11) transformiert, so dass alle Ausdrücke die Gestalt $\sum_i a_i(q^j)\varepsilon^i P_j^a(x)$ mit $a_i \in \mathbb{F}(q^j)$ besitzen.

Definition 5.5 Bezeichne

$$\mathfrak{B}^a := \left\{(x \ominus a)_q^j \mid j \in \mathbb{N}_0\right\}$$

die *q-Potenz-Basis* und

$$\mathfrak{B}_a := \left\{(a \ominus x)_q^j \mid j \in \mathbb{N}_0\right\}$$

die *q-Pochhammer-Basis*. Die Wahl der Namen liegt begründet in der Tatsache, dass man für $a = 0$ bzw. $a = 1$ die speziellen Basen

$$\mathfrak{B}^0 = \left\{x^j \mid j \in \mathbb{N}_0\right\} \quad \text{bzw.} \quad \mathfrak{B}_1 = \left\{(x;q)_j \mid j \in \mathbb{N}_0\right\}$$

erhält.

Bemerkung 5.6 Man beachte, dass man die beiden Basen mit Hilfe der Regel

$$(a \ominus x)_{q^{-1}}^j = (-1)^j q^{\binom{j}{2}}(x \ominus a)_q^j$$

ineinander überführen kann. Aus diesem Grund könnte man sich im Folgenden auf eine Basis beschränken. Wir werden allerdings bei allen weiteren Algorithmen beide Basen anwenden. Der Grund, weshalb wir diese Basen wählen, ist, dass zum Einen

$$\lim_{q\uparrow 1}(x \ominus a)_q^j = (x-a)^j \quad \text{bzw.} \quad \lim_{q\uparrow 1}(a \ominus x)_q^j = (a-x)^j$$

gilt und zum Anderen diese q-Potenzen sich unter der q-Ableitung so verhalten, wie die Potenzen unter der klassischen Ableitung. Wie wir später zeigen werden, sind diese Basen die einzigen Basen, die die Voraussetzungen des q-Taylorsatzes 5.20 erfüllen. Somit sind $\mathfrak{B}^a$ und $\mathfrak{B}_a$ q-Verallgemeinerungen der Potenzbasen. Ferner besitzen alle klassischen q-orthogonalen Polynome q-Potenzreihendarstellungen bzgl. einer dieser Basen. Durch die Identitäten

$$(x \ominus a)_q^j = x^j \left(\frac{a}{x}; q\right)_j \quad \text{bzw.} \quad (a \ominus x)_q^j = a^j \left(\frac{x}{a}; q\right)_j$$

können wir $\mathfrak{B}^a$ und $\mathfrak{B}_a$ auch durch q-Pochhammersymbole darstellen (siehe auch S. 8*ff*).

Zusammen mit obigen Definitionen und Tabelle 5.1 erhalten wir für die Basen, die im Folgenden für uns von Interesse sind,

Korollar 5.7 Sei $a \in \mathbb{F}$. Es gilt $x, D_q, \varepsilon_q \in \mathfrak{L}_{\mathfrak{B}^a}$ und $x, D_{q^{-1}}, \varepsilon_{q^{-1}} \in \mathfrak{L}_{\mathfrak{B}_a}$.

Die Frage, ob die Summe und die Verkettung zweier zu $\mathfrak{B}$ kompatibler Operatoren wieder kompatibel zu $\mathfrak{B}$ sind, beantwortet

Satz 5.8 $\mathfrak{L}_{\mathfrak{B}}$ ist eine $\mathbb{F}$-Algebra.

Beweis Seien $\lambda_1, \lambda_2 \in \mathbb{F}$ und $L_1, L_2 \in \mathfrak{L}_{\mathfrak{B}}$. Dann existieren $a_{i,j} \in \mathbb{F}$ und $b_{i,j} \in \mathbb{F}$ mit

$$L_1 P_j(x) = \sum_{i=-i_0}^{i_1} a_{i,j} P_{j+i}(x) \quad \text{und} \quad L_2 P_j(x) = \sum_{i=-\tilde{\imath}_0}^{\tilde{\imath}_1} b_{i,j} P_{j+i}(x).$$

Setze $I_0 := \max\{i_0, \tilde{\imath}_0\}$ und $I_1 := \max\{i_1, \tilde{\imath}_1\}$, dann gilt auch

$$(\lambda_1 L_1 + \lambda_2 L_2)\, P_j(x) = \sum_{i=-I_0}^{I_1} (\lambda_1 a_{i,j} + \lambda_2 b_{i,j}) P_{j+i}(x),$$

also ist $\mathfrak{B}$ kompatibel zu $\lambda_1 L_1 + \lambda_2 L_2$. Ferner gilt

$$(L_2 L_1)\, P_j(x) = \sum_{i=-i_0}^{i_1} a_{i,j} L_2 P_{j+i}(x) = \sum_{i=-i_0}^{i_1} \sum_{k=-\tilde{\imath}_0}^{\tilde{\imath}_1} a_{i,j} b_{k,j+i} P_{j+i+k}(x) = \sum_{i=-i_0-\tilde{\imath}_0}^{i_1+\tilde{\imath}_1} c_{i,j} P_{j+i}(x) \tag{5.2}$$

mit $c_{i,j} = \sum_k a_{k,j} b_{i-k,j+k}$, also gilt $L_2 L_1 \in \mathfrak{L}_{\mathfrak{B}}$, wobei $a_{i,j}$ und $b_{i,j}$ Null sind, falls i außerhalb der entsprechenden Grenzen verläuft. □

Wir können $\mathbb{F}[x]$ in die folgende Menge formaler Reihen einbetten.

Definition 5.9 Bezeichne $\mathbb{F}[[\mathfrak{B}]]$ die Menge aller formalen Reihen bzgl. der Basis $\mathfrak{B}$, also

$$\mathbb{F}[[\mathfrak{B}]] = \left\{ \sum_{j=0}^{\infty} c_j P_j(x) \;\middle|\; c_j \in \mathbb{F}, P_j(x) \in \mathfrak{B} \right\}.$$

Satz 5.10 $\mathbb{F}[[\mathfrak{B}]]$ ist mit der Multiplikation

$$\left(\sum_{j=0}^{\infty} a_j P_j(x)\right) \cdot \left(\sum_{j=0}^{\infty} b_j P_j(x)\right) = \sum_{j=0}^{\infty} \left(\sum_{m,n} a_m b_n l_j(P_m(x) P_n(x))\right) P_j(x)$$

eine $\mathbb{F}$-Algebra, wobei

$$l_j : \mathbb{F}[x] \mapsto \mathbb{F}, \quad P_i(x) \to \delta_{ij}$$

für jedes $j \in \mathbb{N}_0$.

Beweis Wir zeigen an dieser Stelle lediglich, dass die Multiplikation wohldefiniert ist. Dazu betrachten wir einen Summanden des Resultats beim formalen Multiplizieren von zwei Reihen bzgl.

einer Basis $\mathfrak{B}$.

$$\begin{aligned} a_m P_m(x) b_n P_n(x) &= a_m b_n P_m(x) P_n(x) \\ &= a_m b_n \sum_{k=0}^{m+n} c_k P_k(x) \text{ mit geeigneten } c_k \in \mathbb{F} \\ &= a_m b_n \sum_{k=\max\{m,n\}}^{m+n} c_k P_k(x) \end{aligned}$$

Die letzte Gleichung erhalten wir aufgrund der Eigenschaft (b) von $\mathfrak{B}$. Anwendung von l_j liefert einen Summanden des Koeffizienten von $P_j(x)$

$$a_m b_n l_j(P_m(x)P_n(x)) = a_m b_n \sum_{k=\max\{m,n\}}^{m+n} c_k l_j(P_k(x)) = \begin{cases} a_m b_n c_j, & \max\{m,n\} \le j \le m+n \\ 0, & \text{sonst.} \end{cases}$$

□

Wir erweitern nun den zur Basis $\mathfrak{B}$ kompatiblen Operator L aus Definition 5.1 zu

$$\begin{aligned} L \sum_{j=0}^{\infty} c_j P_j(x) &:= \sum_{j=0}^{\infty} c_j L P_j(x) = \sum_{j=0}^{\infty} \sum_{i=-i_0}^{i_1} a_{i,j} c_j P_{j+i}(x) \\ &= \sum_{j=0}^{\infty} \sum_{i=-i_0}^{i_1} a_{i,j-i} c_{j-i} P_j(x) = \sum_{j=0}^{\infty} \sum_{i=-i_1}^{i_0} a_{-i,j+i} c_{j+i} P_j(x) \end{aligned} \tag{5.3}$$

mit $c_j = 0$, wenn $j < 0$ und können nun definieren

Definition 5.11 Sei $\mathfrak{B} = \{P_j(x) \mid j \in \mathbb{N}_0\}$ und $L \in \mathfrak{L}_{\mathfrak{B}}$, dann bezeichnen wir den durch L *induzierten Rekursionsoperator* mit

$$R_{\mathfrak{B}} L := \sum_{i=-i_1}^{i_0} a_{-i,j+i} \varepsilon^i, \tag{5.4}$$

wobei ε der *Vorwärtsshift* ist, der durch $\varepsilon j := j + 1$ definiert wird und $i_0, i_1, a_{i,j}$ aus Definition 5.1 sind. Ferner sei $\sigma : \mathbb{F}[[\mathfrak{B}]] \to \mathbb{F}^{\mathbb{Z}}$ die Abbildung, die einer formalen q-Reihe $\sum_{j=0}^{\infty} c_j P_j(x) \in \mathbb{F}[[\mathfrak{B}]]$ ihre Koeffizientenfolge $\langle c_j \rangle_{j \in \mathbb{Z}}$ zuordnet, wobei $c_j = 0$, wenn $j < 0$. Die Menge aller Rekursionsoperatoren bzgl. ε bezeichnen wir mit $\mathfrak{R}$.

Satz 5.12 Sei $\mathfrak{B} = \{P_j(x) \mid j \in \mathbb{N}_0\}$. Für $f(x) = \sum_{j=0}^{\infty} c_j P_j(x) \in \mathbb{F}[[\mathfrak{B}]]$ und jeden Operator $L \in \mathfrak{L}_{\mathfrak{B}}$ gilt

$$\sigma L f(x) = (R_{\mathfrak{B}} L)\, \sigma f(x),$$

d.h. das Diagramm

$$\begin{array}{ccc} \mathbb{F}[[\mathfrak{B}]] & \xrightarrow{\ L\ } & \mathbb{F}[[\mathfrak{B}]] \\ \Big\downarrow{\scriptstyle \sigma} & & \Big\downarrow{\scriptstyle \sigma} \\ \mathbb{F}^{\mathbb{Z}} & \xrightarrow[R_{\mathfrak{B}} L]{} & \mathbb{F}^{\mathbb{Z}} \end{array}$$

kommutiert.

Beweis Sei also $f(x) = \sum_{j=0}^{\infty} c_j P_j(x) \in \mathbb{F}[[\mathfrak{B}]]$. Dann gilt unter obigen Voraussetzungen

$$\sigma L f(x) \overset{(5.3)}{=} \left\langle \sum_{i=-i_1}^{i_0} a_{-i,j+i} \varepsilon^i c_j \right\rangle_{j \in \mathbb{Z}} \overset{(5.4)}{=} (R_{\mathfrak{B}} L)\, \sigma f(x).$$

□

Satz 5.13 Die Transformation $\mathfrak{L}_{\mathfrak{B}} \to \mathfrak{R},\ L \mapsto R_{\mathfrak{B}}L$ ist ein Isomorphismus von $\mathbb{F}$-Algebren.

Beweis Die Menge aller Rekursionsoperatoren $\mathfrak{R}$ ist offenbar eine $\mathbb{F}$-Algebra. Zeigen wir nun, dass die obige Abbildung ein $\mathbb{F}$-Algebra-Isomorphismus ist.

(a) Es gilt offenbar $R_{\mathfrak{B}}(\lambda_1 L_1 + \lambda_2 L_2) = \lambda_1 R_{\mathfrak{B}}(L_1) + \lambda_2 R_{\mathfrak{B}}(L_2)$.

(b) Einerseits gilt mit (5.4) und (5.2) die Gleichung

$$R_{\mathfrak{B}}(L_2 L_1) = \sum_i \sum_k a_{k,j+i} b_{-i-k,j+i+k} \varepsilon^i,$$

andererseits erhalten wir

$$\begin{aligned}(R_{\mathfrak{B}}L_2)\,(R_{\mathfrak{B}}L_1) &= \left(\sum_{k=-\tilde{\imath}_1}^{\tilde{\imath}_0} b_{-k,j+k}\varepsilon^k\right)\left(\sum_{i=-i_1}^{i_0} a_{-i,j+i}\varepsilon^i\right) = \sum_{k=-\tilde{\imath}_1}^{\tilde{\imath}_0}\sum_{i=-i_1}^{i_0} a_{-i,j+i+k} b_{-k,j+k}\varepsilon^{i+k} \\ &= \sum_i \sum_k a_{-k,j+i} b_{-i+k,j+i-k}\varepsilon^i \overset{k\to -k}{=} \sum_i \sum_k a_{k,j+i} b_{-i-k,j+i+k}\varepsilon^i.\end{aligned}$$

Also folgt $R_{\mathfrak{B}}(L_2L_1) = (R_{\mathfrak{B}}L_2)\,(R_{\mathfrak{B}}L_1)$.

(c) Um zu zeigen, dass $R_{\mathfrak{B}}L$ ein Isomorphismus ist, geben wir einfach die Umkehrabbildung an. Sei $\tilde{L} \in \mathfrak{R}$ mit $\tilde{L} = \sum_{i=-i_0}^{i_1} a_{i,j}\varepsilon^i$ gegeben und $\tilde{R}_{\mathfrak{B}} : \mathfrak{R} \to \mathfrak{L}_{\mathfrak{B}},\ \tilde{L} \mapsto L$, wobei L der Operator sei, für den $LP_j(x) = \sum_{i=-i_1}^{i_0} a_{-i,j-i}P_{j+i}(x)$ gilt mit $P_j(x) = 0$, wenn $j < 0$. Dann ist, wie man leicht sieht, $\tilde{R}_{\mathfrak{B}}$ die Umkehrabbildung von $R_{\mathfrak{B}}$.

□

In [APR00] wird gezeigt, wie man Satz 5.13 dazu verwenden kann, um Rekursionsoperatoren zu faktorisieren. Man betrachtet zu gegebenem Operator L den induzierten Rekursionsoperator $R_{\mathfrak{B}}L$ und faktorisiert, wenn möglich, diesen Operator. Bei erfolgreicher Faktorisierung erhält man durch Anwendung der Umkehrabbildung eine Faktorisierung für den Ausgangsoperator L.

5.1.1 Konversion zwischen q-Rekursionsgleichungen einer q-Reihe und deren Koeffizienten

Wir wenden nun die Sätze aus dem vorigen Abschnitt auf die von uns betrachteten q-holonomen Rekursions- und Differentialoperatoren an. In Korollar 5.7 haben wir bereits Basen gefunden, die zum Hahn- und q-Shiftoperator sowie zu dem Operator, der einen Ausdruck mit x multipliziert, kompatibel sind und die wir an dieser Stelle erneut betrachten.

Satz 5.14 Die q-Potenz-Basis $\mathfrak{B}^a$ ist kompatibel zu jedem q-holonomen Rekursionsoperator $L \in \mathbb{F}[x][\varepsilon_q]$ bzw. zu jedem q-holonomen Differentialoperator $L \in \mathbb{F}[x][D_q]$ und die q-Pochhammer-Basis $\mathfrak{B}_a$ ist kompatibel zu jedem q-holonomen Rekursionsoperator $L \in \mathbb{F}[x][\varepsilon_{q^{-1}}]$ bzw. zu jedem q-holonomen Differentialoperator $L \in \mathbb{F}[x][D_{q^{-1}}]$.

Beweis Mit $L_1P_j(x) := xP_j(x)$ und $L_2P_j(x) := \varepsilon_q P_j(x)$ lässt sich $L \in \mathbb{F}[x][\varepsilon_q]$ schreiben als

$$L = \sum_{k=0}^{n}\sum_{i=0}^{m_k} \alpha_{ki} x^i \varepsilon_q^k = \sum_{k=0}^{n}\sum_{i=0}^{m_k} \alpha_{ki} L_1^i L_2^k$$

mit $\alpha_{ki} \in \mathbb{F}$. Da $L_1, L_2 \in \mathfrak{L}_{\mathfrak{B}^a}$ nach Korollar 5.7 und $\mathfrak{L}_{\mathfrak{B}^a}$ nach Satz 5.8 eine $\mathbb{F}$-Algebra ist, ist L auch in $\mathfrak{L}_{\mathfrak{B}^a}$. Der Rest verläuft analog. □

Es ist anzumerken, dass man jede q-holonome Rekursionsgleichung bzgl. ε_q in eine q-holonome Rekursionsgleichung bzgl. $\varepsilon_{q^{-1}}$ durch einfaches q-Shiften transformieren kann. Aus diesem Grund gehen wir im Folgenden immer von einem Operator $L \in \mathbb{F}[x][\varepsilon_q]$ aus.

Satz 5.15 Sei $L \in \mathbb{F}[x][\varepsilon_q]$, also L ein q-holonomer Rekursionsoperator, $\mathfrak{B} = \{P_j(x) \mid j \in \mathbb{N}_0\}$ kompatibel zu L mit

$$LP_j(x) = \sum_{i=-i_0}^{i_1} a_i(q^j)P_{j+i}(x),$$

wobei $a_i \in \mathbb{F}(q^j)$ gelte. Dann erfüllt die formale Reihe

$$f(x) = \sum_{j=0}^{\infty} c_j P_j(x) \in \mathbb{F}[[\mathfrak{B}]]$$

die q-holonome Rekursionsgleichung $Lf(x) = 0$ genau dann, wenn die Koeffizienten c_j die q-holonome Rekursionsgleichung[2]

$$(R_{\mathfrak{B}} L)\, c_j = \sum_{i=-i_1}^{i_0} a_{-i}(q^{j+i})\varepsilon^i c_j =: \sum_{i=-i_1}^{i_0} \tilde{a}_{-i}(q^j)\varepsilon^i c_j = 0$$

erfüllen.

Beweis Zunächst ist zu bemerken, dass der induzierte Rekursionsoperator Koeffizienten aus $\mathbb{F}(q^j)$ besitzt und somit zu einem q-holonomen Rekursionsoperator korrespondiert. Das ergibt sich aus Tabelle 5.1 und der Tatsache, dass das Produkt zweier Operatoren mit rationalen Koeffizienten in q^j auch wieder Koeffizienten aus $\mathbb{F}(q^j)$ besitzt. Den Beweis der Äquivalenz liefert (5.3). Der Ausdruck $L\sum_{j=0}^{\infty} c_j P_j(x)$ ist genau dann 0, wenn $\sum_{i=-i_1}^{i_0} a_{-i}(q^{j+i})\varepsilon^i c_j = 0$ gilt. □

Im klassischen Fall geht man in der Regel folgendermaßen vor, um eine holonome Rekursionsgleichung für die Koeffizienten c_j von $f(x) = \sum_{j=0}^{\infty} c_j(x-a)^j$ zu erhalten. Man stellt zunächst für $f(x+a)$ eine holonome Differentialgleichung auf. Diese konvertiert man in eine holonome Rekursionsgleichung für die Koeffizienten der Reihe $f(x+a)$ bzgl. x^j, welche offenbar eine Rekursion für c_j darstellt. Folglich beschränkt man sich auf Konversionsalgorithmen bzgl. der Basis $\mathfrak{B} = \{x^j \mid j \in \mathbb{N}_0\}$. Im q-Fall funktioniert diese Vorgehensweise nicht, da $(x + a \ominus a)_q^j \neq (x \ominus 0)_q^j = x^j$ für $a \neq 0$ gilt und es offensichtlich auch keine lineare Transformation $t_a(x) \in \mathbb{F}[x]$ gibt, die der Eigenschaft $(t_a(x) \ominus a)_q^j = x^j$ für alle $j \in \mathbb{N}_0$ genügt. Selbst wenn es eine Transformation gäbe, so würde man schon im ersten Schritt bei der Bestimmung der q-holonomen Rekursion scheitern, da wir ja im q-Fall keine allgemeingültige Kettenregel haben. Aus diesem Grund lassen wir im Folgenden den Entwicklungspunkt a mit in den Algorithmus einfließen.

Wie bestimmt man nun zu einer gegebenen q-holonomen Rekursionsgleichung für eine formale q-Reihe die zugehörige q-holonome Rekursionsgleichung für die Koeffizienten? Nehmen wir an, $f(x)$ habe die Darstellung

$$f(x) = \sum_{j=0}^{\infty} c_j P_j^a(x) \quad \text{mit} \quad c_j = 0 \quad \text{für} \quad j < 0$$

und erfülle die q-holonome Rekursionsgleichung $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$ mit Namen *RE*. Ferner sei L der zugehörige Rekursionsoperator und es gelten die beiden folgenden Identitäten bzgl. der betrachteten Basis $\mathfrak{B} = \left\{P_j^a(x) \mid j \in \mathbb{N}_0\right\}$, die mit den Operatoren ε_q und x des zugehörigen q-holonomen Rekursionsoperators der Gleichung *RE* kompatibel sind

$$\varepsilon_q P_j^a(x) = \sum_{s=-i_0}^{i_1} a_s(q^j)P_{j+s}^a(x) \quad \text{und} \quad xP_j^a(x) = \sum_{t=-\tilde{i}_0}^{\tilde{i}_1} b_t(q^j)P_{j+t}^a(x) \quad \text{mit} \quad a_s, b_t \in \mathbb{F}(q^j).$$

Aus diesen beiden Identitäten ergeben sich die beiden Gleichungen

$$\varepsilon_q f(x) = \sum_{j=0}^{\infty} c_j \varepsilon_q P_j^a(x) = \sum_{j=0}^{\infty} \sum_{s=-i_1}^{i_0} a_{-s}(q^{j+s})c_{j+s}P_j^a(x)$$

$$xf(x) = \sum_{j=0}^{\infty} c_j x P_j^a(x) = \sum_{j=0}^{\infty} \sum_{t=-\tilde{i}_1}^{\tilde{i}_0} b_{-t}(q^{j+t})c_{j+t}P_j^a(x).$$

[2]Diese Rekursionsgleichung hat im Allgemeinen rationale Koeffizienten in q^j und muss noch mit dem kleinsten gemeinsamen Vielfachen der Nennerpolynome der Koeffizienten multipliziert werden!

Wir bringen die Rekursionsgleichung *RE* in expandierte Form

$$\sum_{k=0}^{n}\sum_{i=0}^{m_k}\alpha_{ki}x^i\varepsilon_q^k f(x)=0 \quad \text{mit} \quad a_k(x)=\sum_{i=0}^{m_k}\alpha_{ki}x^i \quad \text{und} \quad \alpha_{ki}\in\mathbb{F}$$

und können nun alle Summanden der Form $\alpha_{ki}x^i\varepsilon_q^k f(x)$ durch iterative Anwendung der letzten beiden Gleichungen ausdrücken als

$$\sum_{j=0}^{\infty}\beta_{ki}(c_j)P_j^a(x), \tag{5.5}$$

wobei sich $\beta_{ki}(c_j)$ wie folgt bestimmen lässt. Wir betrachten zunächst die q-Shifts und erhalten mit Hilfe der ersten Gleichung die Rekursion

$$\begin{aligned} d_{ki}^{(0)}(c_j) &= \alpha_{ki}c_j \\ d_{ki}^{(l)}(c_j) &= \sum_{s=-i_1}^{i_0} a_{-s}(q^{j+s})\varepsilon^s d_{ki}^{(l-1)}(c_j) \quad \text{für} \quad l=1,\ldots,k. \end{aligned} \tag{5.6}$$

Die zweite Gleichung hilft uns bei der Elimination der Potenzen von x. Es ergibt sich die folgende Rekursion

$$\begin{aligned} e_{ki}^{(0)}(c_j) &= d_{ki}^{(k)}(c_j) \\ e_{ki}^{(l)}(c_j) &= \sum_{t=-\tilde{i}_1}^{\tilde{i}_0} b_{-t}(q^{j+t})\varepsilon^t e_{ki}^{(l-1)}(c_j) \quad \text{für} \quad l=1,\ldots,i. \end{aligned} \tag{5.7}$$

Es gilt $\beta_{ki}(c_j)=e_{ki}^{(i)}(c_j)$ und multiplizieren wir $\beta_{ki}(c_j)$ aus, so lässt sich der Ausdruck offenbar darstellen als

$$\beta_{ki}(c_j)=\sum_{r=-I_0}^{I_1}p_{kir}(q^j)\varepsilon^r c_j \tag{5.8}$$

mit $p_{kir}\in\mathbb{F}(q^j)$ und $I_0,I_1\in\mathbb{N}_0$. Wir können nun jeden Summanden der q-holonomen Rekursionsgleichung ersetzen durch (5.5) mit Gleichung (5.8) und erhalten durch Koeffizientenvergleich bzgl. $P_j^a(x)$

$$\begin{aligned} \sum_{k=0}^{n}\sum_{i=0}^{m_k}\alpha_{ki}x^i\varepsilon_q^k f(x)&=0 \\ \sum_{k=0}^{n}\sum_{i=0}^{m_k}\sum_{j=0}^{\infty}\beta_{ki}(c_j)P_j^a(x)&=0 \\ \sum_{j=0}^{\infty}\left(\sum_{k=0}^{n}\sum_{i=0}^{m_k}\beta_{ki}(c_j)\right)P_j^a(x)&=0 \\ \sum_{k=0}^{n}\sum_{i=0}^{m_k}\beta_{ki}(c_j)&=0 \\ \underbrace{\sum_{r=-I_0}^{I_1}\sum_{k=0}^{n}\sum_{i=0}^{m_k}p_{kir}(q^j)\varepsilon^r c_j}_{=R_{\mathfrak{B}}L}&=0. \end{aligned}$$

Indem wir die letzte Gleichung mit dem kleinsten gemeinsamen Vielfachen der Nennerpolynome der rationalen Funktionen $p_{kir}(q^j)$ multiplizieren, gelangen wir zu einer q-holonomen Rekursionsgleichung für c_j. Wir halten die Vorgehensweise fest in

Algorithmus 5.1 Bestimmung einer q-holonomen Rekursionsgleichung für die Koeffizienten einer formalen q-Reihe (bzgl. Basis $\mathfrak{B}^a$ bzw. $\mathfrak{B}_a$) aus einer q-holonomen Rekursionsgleichung der q-Reihe (`qREtoRE` mit `base=qpower` bzw. `base=qpochhammer` und `expansionpt=a`)

Eingabe : Eine q-holonome Rekursionsgleichung $\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$, eine Basis $\mathfrak{B}$ und einen Entwicklungspunkt a

Ausgabe : Eine q-holonome Rekursionsgleichung für c_j, wobei $f(x) = \sum_{j=0}^{\infty} c_j P_j^a(x)$

1 **begin**
2 $RE \leftarrow \sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0$
3 $RE \leftarrow$ Multipliziere die linke Seite von RE aus
4 Bestimme $\beta_{ki}(c_j)$ für jeden Summanden $\alpha_{ki}x^i\varepsilon_q^k f(x)$ von RE induktiv bzgl. Basis $\mathfrak{B}$ mit den Formeln (5.6) und (5.7), die aus den entsprechenden Einträgen der Tabelle 5.1 resultieren
5 $RE \leftarrow$ Ersetze $\alpha_{ki}x^i\varepsilon_q^k f(x)$ in RE durch $\beta_{ki}(c_j)$ aus (5.8) und multipliziere mit kleinsten gemeinsamen Vielfachen der Nenner
6 $RE \leftarrow$ Shifte RE, so dass lediglich positive Shifts auftreten
7 **return** RE
8 **end**

Verwendet man die Basis $\mathfrak{B}_a$, so muss man die Rekursionsgleichung RE in Zeile 2 zunächst so shiften, dass lediglich negative q-Shifts, also nur Terme der Form $\varepsilon_{q^{-1}}^k f(x)$ mit $k \geq 0$ auftreten. Das geschieht durch die einfache Substitution $x \to q^{-n}x$. Für q-Differentialgleichungen kann man den Algorithmus durch Austauschen des Operators ε_q und Anwendung von D_q gemäß Tabelle 5.1 anpassen. In `qFPS` existiert dazu die Prozedur `qDEtoRE`, die die gleichen optionalen Parameter wie `qREtoRE` besitzt.

Für die Basen $\mathfrak{B}^0$ und $\mathfrak{B}_0$ können wir explizite Formeln für die Substitution in Zeile 4 des Algorithmus 5.1 angeben.

Satz 5.16 Sei f eine q-holonome Funktion in x, welche die q-holonome Rekursionsgleichung

$$\sum_{k=0}^{n} a_k(x)\varepsilon_q^k f(x) = 0 \quad \text{mit} \quad a_k \in \mathbb{F}[x]$$

erfüllt und es gelte $f(x) = \sum_{j=0}^{\infty} c_j x^j$ bzw. $f(x) = \sum_{j=0}^{\infty} c_j(-1)^j q^{\binom{j}{2}} x^j$ mit $c_j = 0$, wenn $j < 0$. Ist

$$\sum_{k=0}^{n}\sum_{i=0}^{m_k} \alpha_{ki}x^i\varepsilon_q^k f(x) = 0 \quad \text{mit} \quad a_k(x) = \sum_{i=0}^{m_k} \alpha_{ki}x^i \quad \text{und} \quad \alpha_{ki} \in \mathbb{F}$$

die expandierte Form der Rekursion, so liefert die formale Substitution

$$x^i\varepsilon_q^k f(x) \to q^{-ik}\left(q^j\right)^k \varepsilon^{-i}c_j \quad \text{bzw.} \quad x^i\varepsilon_q^k f(x) \to (-1)^i q^{\binom{i+1}{2}} q^{-ik}\left(q^j\right)^{k-n-i} \varepsilon^{-i}c_j \tag{5.9}$$

eine q-holonome Rekursionsgleichung für c_j. Bei q-holonomen Differentialgleichungen verwendet man die Substitution

$$x^i D_q^k f(x) \to \frac{(q^{j-i+1};q)_k}{(1-q)^k}\varepsilon^{k-i}c_j \quad \text{bzw.} \quad x^i D_{q^{-1}}^k f(x) \to (-1)^i q^{\binom{i+1}{2}}\left(q^j\right)^{-i}\frac{(q^{j-i+1};q)_k}{(1-q)^k}\varepsilon^{k-i}c_j. \tag{5.10}$$

Beweis Wir beweisen die Aussage für die Basis $\mathfrak{B}^0$. Der Beweis für die Basis $\mathfrak{B}_0$ verläuft analog, wobei man im ersten Schritt bei der q-holonomen Rekursionsgleichung zunächst den Operator $\varepsilon_q^{-n} = \varepsilon_{q^{-1}}^n$ anwenden muss. Nehmen wir an, $f(x)$ habe die Darstellung

$$f(x) = \sum_{j=0}^{\infty} c_j x^j \quad \text{mit} \quad c_j = 0 \quad \text{für} \quad j < 0.$$

Dann erhalten wir durch Einsetzen von $a_k(x) = \sum_{i=0}^{m_k} \alpha_{ki} x^i$ und $f(x) = \sum_{j=0}^{\infty} c_j x^j$ in die gegebene q-holonome Rekursionsgleichung

$$\begin{aligned}
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} x^i \varepsilon_q^k f(x) &= 0 \\
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} x^i \sum_{j=0}^{\infty} c_j \left(q^j\right)^k x^j &= 0 \\
\sum_{j=0}^{\infty} \left(\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \left(q^j\right)^k c_j \right) x^{j+i} &= 0 \\
\sum_{j=0}^{\infty} \underbrace{\left(\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} q^{-ik} \left(q^j\right)^k \left(\varepsilon^{-i} c_j\right) \right)}_{=0} x^j &= 0.
\end{aligned}$$

Gehen wir nun von einer q-holonomen Differentialgleichung aus, so ergibt sich

$$\begin{aligned}
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} x^i D_q^k f(x) &= 0 \\
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} x^i \sum_{j=0}^{\infty} c_j \left(\prod_{l=0}^{k-1} [j-l]_q \right) x^{j-k} &= 0 \\
\sum_{j=0}^{\infty} \left(\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \left(\prod_{l=0}^{k-1} [j-l]_q \right) c_j \right) x^{j+i-k} &= 0 \\
\sum_{j=0}^{\infty} \left(\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \left(\prod_{l=1}^{k} [j-i+l]_q \right) \left(\varepsilon^{k-i} c_j\right) \right) x^j &= 0 \\
\sum_{j=0}^{\infty} \underbrace{\left(\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \frac{(q^{j-i+1};q)_k}{(1-q)^k} \left(\varepsilon^{k-i} c_j\right) \right)}_{=0} x^j &= 0.
\end{aligned}$$

□

Fassen wir nun den Spezialfall aus Satz 5.16 in den folgenden Algorithmen zusammen.

Algorithmus 5.2 Bestimmung einer q-holonomen Rekursionsgleichung für die Koeffizienten einer formalen q-Reihe (bzgl. $\mathfrak{B}^0$ bzw. $\mathfrak{B}_0$) aus einer q-holonomen Rekursionsgleichung (`qREtoRE` mit `base=qpower` bzw. `base=qpochhammer` und `expansionpt=0`)

Eingabe : Eine q-holonome Rekursionsgleichung $\sum_{k=0}^{n} a_k(x) \varepsilon_q^k f(x) = 0$

Ausgabe : Eine q-holonome Rekursionsgleichung für c_j, wobei $f(x) = \sum_{j=0}^{\infty} c_j x^j$ bzw. $f(x) = \sum_{j=0}^{\infty} c_j (-1)^j q^{\binom{j}{2}} x^j$

begin
- $RE \leftarrow \sum_{k=0}^{n} a_k(x) \varepsilon_q^k f(x) = 0$
- $RE \leftarrow$ Multipliziere die linke Seite von RE aus
- $RE \leftarrow$ Ersetze $x^i \varepsilon_q^k f(x)$ in RE mit der entsprechenden Formel (5.9) und multipliziere mit kleinsten gemeinsamen Vielfachen der Nenner
- $RE \leftarrow$ Shifte RE um das Maximum der Grade der $a_k(x)$ nach rechts
- **return** RE

end

Algorithmus 5.3 Bestimmung einer q-holonomen Rekursionsgleichung für die Koeffizienten einer formalen q-Reihe (bzgl. $\mathfrak{B}^0$ bzw. $\mathfrak{B}_0$) aus einer q-holonomen Differentialgleichung (`qDEtoRE` mit `base=qpower` bzw. `base=qpochhammer` und `expansionpt=0`)

Eingabe : Eine q-holonome Differentialgleichung für $f(x)$ bzgl. D_q bzw. $D_{q^{-1}}$

Ausgabe : Eine q-holonome Rekursionsgleichung für c_j, wobei $f(x) = \sum_{j=0}^{\infty} c_j x^j$ bzw. $f(x) = \sum_{j=0}^{\infty} c_j (-1)^j q^{\binom{j}{2}} x^j$

1 **begin**
2 $DE \leftarrow$ q-holonome Differentialgleichung der Eingabe
3 $DE \leftarrow$ Multipliziere die linke Seite von DE aus
4 $RE \leftarrow$ Ersetze $x^i D_q^k f(x)$ bzw. $x^i D_{q^{-1}}^k f(x)$ in DE mit der entsprechenden Formel (5.10) und multipliziere mit kleinsten gemeinsamen Vielfachen der Nenner
5 $RE \leftarrow$ Shifte RE um das Minimum von $\{k - i \mid x^i D_q^k f(x) \text{ in } DE\}$ nach links
6 **return** RE
7 **end**

Beweis siehe Beweis von Satz 5.16. Die jeweilige Verschiebung in Zeile 5 hat zur Folge, dass die entstehende Rekursionsgleichung nur positive Shifts aufweist und bei c_j anfängt. □

Bemerkung 5.17 Wir können die Ordnung und den maximalen Grad der Koeffizientenpolynome einer induzierten Rekursionsgleichung für die Basen $\mathfrak{B}^a$ und $\mathfrak{B}_a$ aus der Struktur der gegebenen q-holonomen Rekursionsgleichung $\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} x^i \varepsilon_q^k f(x) = 0$ ermitteln. Sei dazu $\alpha_{0,i} \neq 0$ für ein $i = 0, \ldots, m_k$ und $\alpha_{k,0} \neq 0$ für ein $k = 0, \ldots, n$. Wir erhalten die folgende Tabelle

Basis	Zahl	$a \neq 0$	$a = 0$
$\mathfrak{B}^a$	Ordnung	$\max\{i \mid \alpha_{ki} \neq 0\} + \max\{k \mid \alpha_{ki} \neq 0\}$	$\max\{i \mid \alpha_{ki} \neq 0\}$
	Grad	$\max\{i + 2k \mid \alpha_{ki} \neq 0\}$	$\max\{k \mid \alpha_{ki} \neq 0\}$
$\mathfrak{B}_a$	Ordnung	$\max\{i \mid \alpha_{ki} \neq 0\} + \max\{k \mid \alpha_{ki} \neq 0\}$	$\max\{i \mid \alpha_{ki} \neq 0\}$
	Grad	$\max\{n + i - k \mid \alpha_{ki} \neq 0\}$	$\max\{n + i - k \mid \alpha_{ki} \neq 0\}$

Tabelle 5.2 Ordnung und maximaler Grad der Koeffizientenpolynome des induzierten Rekursionsoperators

Beweis Wir beweisen exemplarisch zwei Einträge der Tabelle. Betrachten wir zunächst die Ordnung des induzierten Rekursionsoperators $R_{\mathfrak{B}^a} L$ für den Fall $a \neq 0$. Dem Algorithmus 5.1 entnimmt man, dass eine einmalige Anwendung des q-Shiftoperators eine Inkrementierung des Index von c_j nach sich zieht. Bei der Multiplikation mit x hingegen wird der Index von c_j dekrementiert und c_j wird so zu c_{j-1}. Bei beiden Operationen wird aber auch ein neuer c_j-Term erzeugt. Die k-malige Anwendung von ε_q liefert also einen Ausdruck, bei dem ein Term mit c_{j+k} und ein Term mit c_j (und weitere Terme dazwischen) vorkommen und die Multiplikation von x^i erzeugt u.a. einen c_{j-i}-Term und einen c_j-Term. Dadurch bleiben c_j's, die durch ε_q erhöht worden sind, erhalten und der maximale negative Shift c_{j-i} wird erreicht. Es ergibt sich also die Ordnung $\max\{i \mid \alpha_{ki} \neq 0\} + \max\{k \mid \alpha_{ki} \neq 0\}$. Der maximale Grad der Koeffizientenpolynome in q^j ist $\max\{i + 2k \mid \alpha_{ki} \neq 0\}$, da Anwendung von ε_q den Grad um 2 und Anwendung von x den Grad um 1 erhöht. □

Bestimmen wir einige induzierte Rekursionsgleichungen mit Maple.

Maple-Sitzung 5.1 (Konversionen q-holonomer Rekursionsgleichungen I)

```
> RE:=qHolonomicRE(qsin(x,q),F(x));
```

$$RE := q\left(1+x^2\right) F(x) + (-1-q)\, Sq_x(F(x)) + \left(Sq_{x,x}\right)(F(x)) = 0$$

```
> RE1:=qREtoRE(RE,F(x),c(j));
```

$$RE1 := c(j) + \left(q^j q^2 - 1\right)\left(q^j q - 1\right) c(j+2) = 0$$

```
> RE2:=qREtoRE(RE,F(x),c(j),base=qpochhammer);
```

$$RE2 := c(j) + q\left(q^j\right)^2\left(q^j q^2 - 1\right)\left(q^j q - 1\right) c(j+2) = 0$$

```
> RE3:=qREtoRE(RE,F(x),c(j),expansionpt=a);
```

$$RE3 := c(j) + aq^j q(1+q)\, c(j+1) + \left(a^2\left(q^j\right)^2 q^4 + \left(q^j\right)^2 q^3 - q^j q^2 - q^j q + 1\right) c(j+2)$$
$$+aq^j q(1+q)\left(q^j q^3 - 1\right)\left(q^j q^2 - 1\right) c(j+3) + a^2\left(q^j\right)^2 q^4\left(q^j q^4 - 1\right)\left(q^j q^3 - 1\right) c(j+4) = 0$$

```
> qREtoRE(RE,F(x),c(j),base=qpochhammer,expansionpt=a);
```

$$q^{10} c(j) + (1+q)\, aq^6\left(q^j q^3 - q^2 - 1\right) c(j+1) + q^3 + \left(-q^9\left(q^j\right)^3 - q^{10}\left(q^j\right)^3 + q^{11}\left(q^j\right)^4 + 2a^2 q^2\right.$$
$$\left.+a^2 q^3 + a^2 + a^2 q + a^2 q^4 + q^8\left(q^j\right)^2 - a^2 q^j q^3 - 2a^2 q^j q^4 - 2q^5 a^2 q^j - q^6 a^2 q^j + q^7 a^2\left(q^j\right)^2\right) c(j+2)$$
$$-aq(1+q)\left(q^j q^3 - 1\right)\left(q^{10}\left(q^j\right)^3 - q^8\left(q^j\right)^2 + a^2 q^j q^4 - a^2 q^2 - a^2\right) c(j+3)$$
$$+a^2\left(q^j q^4 - 1\right)\left(q^j q^3 - 1\right)\left(q^8\left(q^j\right)^2 + a^2\right) c(j+4) = 0$$

Wie erhalten wir nun umgekehrt aus einer q-holonomen Rekursionsgleichung für die Koeffizienten einer Potenzreihe $f(x)$ eine q-holonome Rekursionsgleichung für $f(x)$? Um dieses Problem zu lösen, geben wir an, wie die Operatoren ε^{-1} und q^j auf die q-Potenz- und die q-Pochhammer-Basis wirken. Das Ergebnis dieser Operationen muss ein Ausdruck der Form $\sum_k a_k(x)\varepsilon_q^k P_j^a(x)$ mit $a_k \in \mathbb{F}(x)$ sein. Für die Basis $\mathfrak{B}^a$ gelingt es uns in einem Fall nicht, eine Darstellung gemäß obiger Summe zu finden, wohingegen wir mit der folgenden Vorgehensweise für alle anderen Fälle die Einträge von Tabelle 5.3 erhalten. Man wählt den Ansatz

$$\left(a_{-1}(x)\varepsilon_q^{-1} + a_0(x) + a_1(x)\varepsilon_q\right) P_j^a(x) = L P_j^a(x) \tag{5.11}$$

mit $L = q^j$ oder $L = \varepsilon^{-1}$ und möglichst kleinen $a_{-1}(x), a_0(x), a_1(x) \in \mathbb{F}(x)$, in dem Sinne, dass Zähler- und Nennerpolynome, deren Koeffizienten Unbestimmte aus $\mathbb{F}$ sind, kleinen Grad besitzen. Wir können nun Gleichung (5.11) nach Umstellen und Anwendung der Operatoren auch schreiben als $r(x)P_j^a(x) = 0$, wobei $r(x) \in \mathbb{F}(q^j)(x)$. Durch Koeffizientenvergleich des Zählerpolynoms bzgl. x und q^j erhält man ein lineares Gleichungssystem, dessen Lösung nach Einsetzen in den Ansatz die Darstellungen von Tabelle 5.3 liefert.

L \ $P_j^a(x)$	$(x \ominus a)_q^j$	x^j	$(a \ominus x)_q^j$	$(x;q)_j$
q^j	$\frac{1}{qx-a}\varepsilon_q \varepsilon\,(x \ominus a)_q^j$	$\varepsilon_q x^j$	$\left(\frac{x-a}{x}\varepsilon_q + \frac{a}{x}\right)(a \ominus x)_q^j$	$\left(\frac{x-1}{x}\varepsilon_q + \frac{1}{x}\right)(x;q)_j$
ε^{-1}	$\left(\frac{a}{(qx-a)x}\varepsilon_q + \frac{1}{x}\right)(x \ominus a)_q^j$	$\frac{1}{x}x^j$	$\frac{q}{qa-x}\varepsilon_q^{-1}(a \ominus x)_q^j$	$\frac{q}{q-x}\varepsilon_q^{-1}(x;q)_j$

Tabelle 5.3 Identitäten zur Formulierung der Umkehrung von Algorithmus 5.1

Der folgende Algorithmus bestimmt die q-holonome Rekursionsgleichung einer formalen q-Reihe bzgl. Basis $\mathfrak{B}_a$ mit Hilfe der Darstellungen aus Tabelle 5.3. Der Algorithmus ist eine Adaption des Algorithmus 5.1.

Für $\mathfrak{B} = \left\{P_j^a(x) \mid j \in \mathbb{N}_0\right\}$ gelte

$$\varepsilon^{-1} P_j^a(x) = \sum_{s=-i_0}^{i_1} a_s(x)\varepsilon_q^s P_j^a(x) \quad \text{und} \quad q^j P_j^a(x) = \sum_{t=-\tilde{i}_0}^{\tilde{i}_1} b_t(x)\varepsilon_q^t P_j^a(x) \quad \text{mit} \quad a_s, b_t \in \mathbb{F}(x).$$

Aus diesen beiden Gleichungen folgen

$$\sum_{j=0}^{\infty}(\varepsilon c_j)\,P_j^a(x)=\sum_{j=0}^{\infty}c_j\varepsilon^{-1}P_j^a(x)=\sum_{j=0}^{\infty}\sum_{s=-i_0}^{i_1}a_s(x)\varepsilon_q^s c_j P_j^a(x)=\sum_{s=-i_0}^{i_1}a_s(x)\varepsilon_q^s f(x)$$
$$\sum_{j=0}^{\infty}q^j c_j P_j^a(x)=\sum_{j=0}^{\infty}\sum_{t=-\tilde{i}_0}^{\tilde{i}_1}b_t(x)\varepsilon_q^t c_j P_j^a(x)=\sum_{t=-\tilde{i}_0}^{\tilde{i}_1}b_t(x)\varepsilon_q^t f(x).$$

Wir expandieren die Rekursionsgleichung

$$\sum_{k=0}^{n}\sum_{i=0}^{m_k}\alpha_{ki}\left(q^j\right)^i\varepsilon^k c_j=0\quad\text{mit}\quad a_k(q^j)=\sum_{i=0}^{m_k}\alpha_{ki}(q^j)^i\quad\text{und}\quad\alpha_{ki}\in\mathbb{F}\tag{5.12}$$

und können nun alle Reihen $\sum_{j=0}^{\infty}\alpha_{ki}(q^j)^i\left(\varepsilon^k c_j\right)P_j^a(x)$ mit Hilfe der letzten beiden Gleichungen wie folgt ausdrücken durch $\beta_{ki}(f(x))$. Wir führen die Iterationen

$$\begin{aligned}d_{ki}^{(0)}(f(x))&=\alpha_{ki}f(x)\\ d_{ki}^{(l)}(f(x))&=\sum_{s=-i_0}^{i_1}a_s(x)\varepsilon_q^s d_{ki}^{(l-1)}(f(x))\quad\text{für}\quad l=1,\dots,k\end{aligned}\tag{5.13}$$

und

$$\begin{aligned}e_{ki}^{(0)}(f(x))&=d_{ki}^{(k)}(f(x))\\ e_{ki}^{(l)}(f(x))&=\sum_{t=-\tilde{i}_0}^{\tilde{i}_1}b_t(x)\varepsilon_q^t e_{ki}^{(l-1)}(f(x))\quad\text{für}\quad l=1,\dots,i\end{aligned}\tag{5.14}$$

aus und erhalten so $\beta_{ki}(f(x))=e_{ki}^{(i)}(f(x))$. Somit können wir die q-holonome Rekursionsgleichung für $f(x)$ ausgeben als

$$\sum_{k=0}^{n}\sum_{i=0}^{m_k}\beta_{ki}(f(x))=\sum_{j=0}^{\infty}\underbrace{\sum_{k=0}^{n}\sum_{i=0}^{m_k}\alpha_{ki}\left(q^j\right)^i\left(\varepsilon^k c_j\right)}_{=0}P_j^a(x)=0.\tag{5.15}$$

Die Vorgehensweise fassen wir für den Fall $\mathfrak{B}=\mathfrak{B}_a$ zusammen in

Algorithmus 5.4 Bestimmung einer q-holonomen Rekursionsgleichung einer formalen q-Reihe (bzgl. Basis $\mathfrak{B}_a$) aus einer q-holonomen Rekursionsgleichung der Koeffizienten der q-Reihe (`REtoqRE` mit `base=qpochhammer` und `expansionpt=a`)

Eingabe : Eine q-holonome Rekursionsgleichung $\sum_{k=0}^{n}a_k(q^j)\varepsilon^k c_j=0$ und einen Entwicklungspunkt a

Ausgabe : Eine q-holonome Rekursionsgleichung für $f(x)$, wobei $f(x)=\sum_{j=0}^{\infty}c_j\,(a\ominus x)_q^j$

begin
- $RE\leftarrow\sum_{k=0}^{n}a_k(q^j)\varepsilon^k c_j=0$
- $RE\leftarrow$ Multipliziere die linke Seite von RE aus
- Bestimme $\beta_{ki}(f(x))$ für jede Reihe $\sum_{j=0}^{\infty}\alpha_{ki}(q^j)^i\left(\varepsilon^k c_j\right)(a\ominus x)_q^j$ induktiv mit den Formeln (5.13) und (5.14), die aus den entsprechenden Einträgen der Tabelle 5.3 resultieren
- $RE\leftarrow$ Ersetze $\alpha_{ki}(q^j)^i\varepsilon^k c_j$ in RE durch $\beta_{ki}(f(x))$ und multipliziere mit kleinsten gemeinsamen Vielfachen der Nenner
- $RE\leftarrow$ Shifte RE, so dass lediglich positive q-Shifts auftreten
- **return** RE

end

Für die q-Potenz-Basis können wir den obigen Algorithmus in dieser Form nicht anwenden, da wir keine Gleichung $q^j\,(x\ominus a)_q^j = \sum_k a_k(x)\varepsilon_q^k\,(x\ominus a)_q^j$ mit $a_k \in \mathbb{F}(x)$ haben, sondern $q^j\,(x\ominus a)_q^j = \frac{1}{qx-a}\varepsilon_q\varepsilon\,(x\ominus a)_q^j$. Wir erhalten in diesem Fall für $i,k\in\mathbb{N}_0$

$$\begin{aligned}\sum_{j=0}^{\infty}\left(q^j\right)^i\left(\varepsilon^k c_j\right)(x\ominus a)_q^j &= \sum_{j=-1}^{\infty}\left(q^j\right)^{i-1}\left(\varepsilon^k c_j\right)\,q^j\,(x\ominus a)_q^j\\ &= \sum_{j=-1}^{\infty}\left(q^j\right)^{i-1}\left(\varepsilon^k c_j\right)\frac{1}{qx-a}\varepsilon_q\varepsilon\,(x\ominus a)_q^j\\ &= q^{-(i-1)}\frac{1}{qx-a}\varepsilon_q\sum_{j=0}^{\infty}\left(q^j\right)^{i-1}\left(\varepsilon^{k-1}c_j\right)(x\ominus a)_q^j\\ &= \ldots = (-1)^i q^{-\binom{i}{2}}\frac{1}{(a\ominus qx)_q^i}\varepsilon_q^i\sum_{j=0}^{\infty}\left(\varepsilon^{k-i}c_j\right)(x\ominus a)_q^j\,.\end{aligned}$$

Auf diese Art und Weise eliminieren wir sämtliche Potenzen $(q^j)^i$ in der Rekursionsgleichung. Um lediglich nichtnegative Shifts zu erhalten, müssen wir gewährleisten, dass $k-i\in\mathbb{N}_0$ für $i,k\in\mathbb{N}_0$ gilt. Das erreichen wir dadurch, dass wir die Ausgangsrekursion (5.12) gemäß $j\to j-\min\{k-m_k\in -\mathbb{N}_0 \mid k=0,\ldots,n\}$ shiften. Die positiven Shifts können wir dann mit Hilfe des entsprechenden Eintrags der Tabelle 5.3, ähnlich wie im obigen Algorithmus, eliminieren. Wir passen die Iterationen des Algorithmus 5.4 unserer jetzigen Situation an und erhalten

$$\begin{aligned}d_{ki}^{(0)}(f(x)) &= \alpha_{ki}f(x)\\ d_{ki}^{(l)}(f(x)) &= q^{-(i-l)}\frac{1}{qx-a}\varepsilon_q d_{ki}^{(l-1)}(f(x)) \quad \text{für} \quad l=1,\ldots,i\end{aligned} \tag{5.16}$$

und

$$\begin{aligned}e_{ki}^{(0)}(f(x)) &= d_{ki}^{(i)}(f(x))\\ e_{ki}^{(l)}(f(x)) &= e_{ki}^{(l-1)}\left(\left(\frac{a}{(qx-a)x}\varepsilon_q+\frac{1}{x}\right)f(x)\right) \quad \text{für} \quad l=1,\ldots,k-i.\end{aligned} \tag{5.17}$$

Somit gilt mit $\beta_{ki}(f(x)) = e_{ki}^{(k-i)}(f(x))$ und $P_j^a(x) = (x\ominus a)_q^j$ wiederum Gleichung (5.15).

Algorithmus 5.5 Bestimmung einer q-holonomen Rekursionsgleichung einer formalen q-Reihe (bzgl. Basis $\mathfrak{B}^a$) aus einer q-holonomen Rekursionsgleichung der Koeffizienten der q-Reihe (`REtoqRE` mit `base=qpower` und `expansionpt=a`)

Eingabe : Eine q-holonome Rekursionsgleichung $\sum_{k=0}^n a_k(q^j)\varepsilon^k c_j = 0$ und einen Entwicklungspunkt a

Ausgabe : Eine q-holonome Rekursionsgleichung für $f(x)$, wobei $f(x)=\sum_{j=0}^{\infty} c_j\,(x\ominus a)_q^j$

begin
- $RE \leftarrow \sum_{k=0}^n a_k(q^j)\varepsilon^k c_j = 0$
- $RE \leftarrow$ Führe die Substitution $j\to j-\min\{k-m_k\in-\mathbb{N}_0 \mid k=0,\ldots,n\}$ bzgl. RE durch
- $RE \leftarrow$ Multipliziere die linke Seite von RE aus
- Bestimme $\beta_{ki}(f(x))$ für jede Reihe $\sum_{j=0}^{\infty}\alpha_{ki}(q^j)^i\left(\varepsilon^k c_j\right)(x\ominus a)_q^j$ induktiv mit den Formeln (5.16) und (5.17)
- $RE \leftarrow$ Ersetze $\alpha_{ki}(q^j)^i\varepsilon^k c_j$ in RE durch $\beta_{ki}(f(x))$ und multipliziere mit kleinsten gemeinsamen Vielfachen der Nenner
- **return** RE

end

Für die Basen $\mathfrak{B}^0$ und $\mathfrak{B}_0$ können wir erneut direkte Substitutionsregeln angeben.

Satz 5.18 Sei f eine Funktion in x mit der q-hypergeometrischen Reihenentwicklung

$$f(x) = \sum_{j=0}^{\infty} c_j x^j \quad \text{bzw.} \quad f(x) = \sum_{j=0}^{\infty} c_j (-1)^j q^{\binom{j}{2}} x^j$$

mit $\frac{c_{j+1}}{c_j} \in \mathbb{F}(q^j)$. Ferner erfüllen die Koeffizienten eine q-holonome Rekursionsgleichung der Form

$$\sum_{k=0}^{n} a_k(q^j) \varepsilon^k c_j = 0.$$

Dann erhält man durch die formale Substitution

$$\left(q^j\right)^i \varepsilon^k c_j \to q^{-ki} x^{n-k} \varepsilon_q^i f(x) \quad \text{bzw.} \quad \left(q^j\right)^i \varepsilon^k c_j \to (-1)^k q^{\binom{k+1}{2}} q^{-ki} x^{-k} \varepsilon_q^{i-k} \tag{5.18}$$

eine q-holonome Rekursionsgleichung für $f(x)$.

Beweis Seien die Voraussetzungen des Satzes gegeben. Wir beweisen die Aussage für die Basis $\mathfrak{B}^0$. Dazu bringen wir die q-holonome Rekursionsgleichung der Koeffizienten zunächst mit $a_k(q^j) = \sum_{i=0}^{m_k} \alpha_{ki} \left(q^j\right)^i$ in die expandierte Form und verwenden dann die Formel $\sum_j \left(q^j\right)^i c_j x^j = \varepsilon_q^i f(x)$

$$\begin{aligned}
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \left(q^j\right)^i \varepsilon^k c_j &= 0 \\
\sum_{j=0}^{\infty} \sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \left(q^j\right)^i \left(\varepsilon^k c_j\right) x^j &= 0 \\
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} \sum_{j=0}^{\infty} \left(q^j\right)^i \left(\varepsilon^k c_j\right) x^j &= 0 \\
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} q^{-ki} x^{-k} \sum_{j=0}^{\infty} \left(q^{j+k}\right)^i \left(\varepsilon^k c_j\right) x^{j+k} &= 0 \quad \Big| \cdot x^n \\
\sum_{k=0}^{n} \sum_{i=0}^{m_k} \alpha_{ki} q^{-ki} x^{n-k} \varepsilon_q^i f(x) &= 0.
\end{aligned}$$

Analog erhält man die Aussage für die Basis $\mathfrak{B}_0$. □

Algorithmus 5.6 Bestimmung einer q-holonomen Rekursionsgleichung für eine q-Reihe (bzgl. $\mathfrak{B}^0$ bzw. $\mathfrak{B}_0$) aus einer q-holonomen Rekursionsgleichung der Koeffizienten (`REtoqRE` mit `base=qpower` bzw. `base=qpochhammer` und `expansionpt=0`)

Eingabe : Eine q-holonome Rekursionsgleichung $\sum_{k=0}^{n} a_k(q^j) \varepsilon^k c_j = 0$

Ausgabe : Eine q-holonome Rekursionsgleichung für $f(x)$, wobei $f(x) = \sum_{j=0}^{\infty} c_j x^j$ bzw. $f(x) = \sum_{j=0}^{\infty} c_j (-1)^j q^{\binom{j}{2}} x^j$

begin
- *RE* ← $\sum_{k=0}^{n} a_k(q^j) \varepsilon^k c_j = 0$
- *RE* ← Multipliziere die linke Seite von *RE* aus
- *RE* ← Ersetze $\left(q^j\right)^i \varepsilon^k c_j$ in *RE* mit der entsprechenden Formel (5.18) und multipliziere mit kleinsten gemeinsamen Vielfachen der Nenner
- *RE* ← Shifte *RE*, so dass lediglich positive q-Shifts auftreten
- **return** *RE*

end

Während wir mit Algorithmus 5.6 die Umkehrabbildung von $R_{\mathfrak{B}^0}$ bzw. $R_{\mathfrak{B}_0}$ als Transformation $\mathbb{F}[x][\varepsilon_q] \to \mathbb{F}[q^j][\varepsilon]$ (Algorithmus 5.2) formulieren konnten[3], bilden die Algorithmen 5.4 und 5.5 keine Umkehrabbildungen von $R_{\mathfrak{B}_a}$ und $R_{\mathfrak{B}^a}$. Für die Fälle $a \neq 0$ bei den q-Potenz- und q-Pochhammerbasen benötigen wir aber für das weitere Vorgehen auch keine Umkehrabbildungen. Unsere Algorithmen machen es vielmehr möglich, zu **jeder** q-holonomen Rekursionsgleichung einer Reihe eine korrespondierende Rekursionsgleichung für die Koeffizienten und zu **jeder** q-holonomen Rekursionsgleichung für die Koeffizienten eine korrespondierende Rekursionsgleichung der Reihe zu bestimmen. Wissen wir aber, dass ein gegebener Rekursionsoperator einem induzierten Rekursionsoperator $R_{\mathfrak{B}}L$ entspricht, so können wir auch den Ausgangsoperator L angeben. Wir deuten an, wie man diesen algorithmisch erhalten und in diesem Fall die Ordnung der q-holonomen Rekursionsgleichung für die Reihe möglichst niedrig halten kann. Sei n die Ordnung des Operators $R_{\mathfrak{B}}L$ und m der maximale Grad der Koeffizientenpolynome bzgl. q^j. Zunächst macht man den Ansatz $L = \sum_{k=0}^{\tilde{n}} \sum_{i=0}^{\tilde{m}} \alpha_{ki} x^i \varepsilon_q^k$ mit unbekannten $\alpha_{ki} \in \mathbb{F}$ unter Berücksichtigung von Bemerkung 5.17, d.h. es gelte $\tilde{n} + \tilde{m} = n$ und es treten z.B. bei $\mathfrak{B}^a$ im Ansatz nur Terme $\alpha_{ki} x^i \varepsilon_q^k$ auf, für die $i + 2k \leq m$ gilt. Für jeden dieser Ansätze (für $\tilde{n}$ und $\tilde{m}$ gibt es i.Allg. mehrere Möglichkeiten) führt man Algorithmus 5.1 aus und führt nun Koeffizientenvergleich bzgl. $x^i \varepsilon_q^k$ mit dem Operator $R_{\mathfrak{B}}L$ durch. Durch Lösen des zustande kommenden linearen Gleichungssystems erhält man dann bei einem der Ansätze eine nichttriviale Lösung und somit den Operator L.

Wir führen die letzte Maple-Sitzung von Seite 80 fort.

Maple-Sitzung 5.2 (Konversionen q-holonomer Rekursionsgleichungen II)

```
> REtoqRE(RE1,c(j),F(x));
```

$$q\left(1+x^2\right)F(x) + (-1-q)\,Sq_x\left(F(x)\right) + \left(Sq_{x,x}\right)\left(F(x)\right) = 0$$

```
> REtoqRE(RE2,c(j),F(x),base=qpochhammer);
```

$$q\left(1+x^2\right)F(x) + (-1-q)\,Sq_x\left(F(x)\right) + \left(Sq_{x,x}\right)\left(F(x)\right) = 0$$

```
> REtoqRE(RE3,c(j),F(x),expansionpt=a);
```

$$\begin{aligned} &a^2\left(Sq_{x,x,x,x}\right)\left(F(x)\right) - a\,(1+q)\left(-q^2x+2\,a\right)\left(Sq_{x,x,x}\right)\left(F(x)\right) + \\ &\left(a^2 - 2\,aq^2x + q^4x^2 + 4\,a^2q - 3\,aq^3x + q^5x^2a^2 - aq^4x + a^2q^2\right)\left(Sq_{x,x}\right)\left(F(x)\right) - \\ &q\,(1+q)\left(-q^2x+a\right)\left(aq^2x^2 - qx + 2\,a\right)Sq_x\left(F(x)\right) + q^2\left(1+x^2\right)\left(-q^2x+a\right)(-qx+a)\,F(x) = 0 \end{aligned}$$

5.1.2 q-Holonome Rekursionsgleichungen verallgemeinerter q-hypergeometrischer Funktionen

Nachdem wir in Abschnitt 3.1.4 bereits eine explizite Formel für die q-holonome Rekursionsgleichung der verallgemeinerten q-hypergeometrischen Funktion

$${}_r\phi_s\left(\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} \middle| q; x\right)$$

mit $a_i, b_j \in \mathbb{F}$ aufstellen konnten, können wir nun einen Algorithmus formulieren, der uns auch befähigt, q-holonome Rekursionsgleichungen für ${}_r\phi_s$ zu bestimmen, bei denen x zusätzlich oder ausschließlich als oberer Parameter auftritt. Dazu führen wir die verallgemeinerte q-hypergeometrische Funktion auf q-hypergeometrische Potenzreihen der Form $\sum_j c_j P_j^a(x)$ mit $P_j^a(x) \in \mathfrak{B}^a$ bzw. $P_j^a(x) \in \mathfrak{B}_a$ zurück.

Es gelten offenbar die Identitäten

$$x^j \left(\frac{a}{x}; q\right)_j = (x \ominus a)_q^j \quad \text{und} \quad (ax; q)_j = a^j \left(\frac{1}{a} \ominus x\right)_q^j \quad \text{für} \quad a \neq 0. \tag{5.19}$$

Mit diesen Identitäten können wir folgenden Algorithmus angeben

[3] zum Beweis betrachte die Substitutionsformeln (5.9) und (5.18)

Algorithmus 5.7 Bestimmung einer q-holonomen Rekursionsgleichung für die verallgemeinerte q-hypergeometrische Funktion ${}_r\phi_s$ (`qHolonomicRE` bei `qphihypergeom`)

Eingabe : Eine verallgemeinerte q-hypergeometrische Funktion ${}_r\phi_s(x)$

Ausgabe : Eine q-holonome Rekursionsgleichung für ${}_r\phi_s(x)$

begin
  switch ${}_r\phi_s(\boldsymbol{a}, \boldsymbol{b} \mid q; X)$ **do**
    case *x im Argument X und nicht in oberen Parametern* $\boldsymbol{a}$ *enthalten*
      $c_j \leftarrow \frac{(\boldsymbol{a};q)_j}{(\boldsymbol{b};q)_j} \frac{X^j}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} \frac{1}{x^j}$, *base* $\leftarrow \mathfrak{B}^0$
    end
    case *x im Argument X und einmal im Nenner eines oberen Parameters a enthalten*
      $c_j \leftarrow \frac{(\boldsymbol{a};q)_j}{(\boldsymbol{b};q)_j} \frac{X^j}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} \frac{1}{x^j} (a;q)_j^{-1}$, *base* $\leftarrow \mathfrak{B}^{ax}$
    end
    case *x nicht im Argument X und einmal im Zähler eines oberen Parameters a enthalten*
      $c_j \leftarrow \frac{(\boldsymbol{a};q)_j}{(\boldsymbol{b};q)_j} \frac{X^j}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} \left(\frac{a}{x}\right)^j (a;q)_j^{-1}$, *base* $\leftarrow \mathfrak{B}_{\frac{x}{a}}$
    end
    otherwise
      return *„Es konnte keine q-holonome Rekursionsgleichung gefunden werden.“*
    end
  end
  $RE \leftarrow$ Stelle aus dem Termverhältnis $\frac{c_{j+1}}{c_j}$ die q-holonome Rekursion erster Ordnung für c_j auf
  $RE \leftarrow$ Konvertiere RE in eine q-holonome Rekursion für die Reihe bzgl. Basis *base* mit Algorithmus 5.4 bzw. 5.5
  return RE
end

Beweis Wir beweisen die Korrektheit der drei obigen Fälle bei der Bestimmung von c_j, der Basis und des Entwicklungspunktes e für

$$ {}_r\phi_s(x) = {}_r\phi_s\left(\begin{matrix}\boldsymbol{a}\\ \boldsymbol{b}\end{matrix} \,\middle|\, q; X\right) = \sum_{j=0}^{\infty} \frac{(\boldsymbol{a};q)_j}{(\boldsymbol{b};q)_j} \frac{X^j}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} = \sum_{j=0}^{\infty} c_j P_j^e(x). $$

Verwende dazu obige Formeln (5.19) und bezeichne mit $(\tilde{\boldsymbol{a}};q)_j$ das um das q-Pochhammersymbol des Parameters a gekürzte Produkt der oberen q-Pochhammersymbole. Dann gilt

(I) ${}_r\phi_s(x) = \sum_{j=0}^{\infty} \frac{(\boldsymbol{a};q)_j}{(\boldsymbol{b};q)_j} \frac{1}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} \left(\frac{X}{x}\right)^j x^j = \sum_{j=0}^{\infty} c_j x^j$

(II) ${}_r\phi_s(x) = \sum_{j=0}^{\infty} \frac{(\tilde{\boldsymbol{a}};q)_j}{(\boldsymbol{b};q)_j} \frac{1}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} \left(\frac{X}{x}\right)^j (x \ominus ax)_q^j = \sum_{j=0}^{\infty} c_j (x \ominus ax)_q^j$

(III) ${}_r\phi_s(x) = \sum_{j=0}^{\infty} \frac{(\tilde{\boldsymbol{a}};q)_j}{(\boldsymbol{b};q)_j} \frac{X^j}{(q;q)_j} \left((-1)^j q^{\binom{j}{2}}\right)^{1+s-r} \left(\frac{a}{x}\right)^j \left(\frac{x}{a} \ominus x\right)_q^j = \sum_{j=0}^{\infty} c_j \left(\frac{x}{a} \ominus x\right)_q^j.$

Alle c_j erfüllen eine q-holonome Rekursionsgleichung erster Ordnung, da c_j ein q-hypergeometrischer Term ist. Somit erhalten wir unter Verwendung des Konversionsalgorithmus 5.4 bzw. 5.5 die gesuchte q-holonome Rekursionsgleichung für ${}_r\phi_s(x)$. □

Um eine q-holonome Rekursionsgleichung für ${}_r\phi_s(\frac{1}{x})$ zu bestimmen, führt man zunächst die Substitution $y = \frac{1}{x}$ durch und bestimmt mit Algorithmus 5.7 eine Rekursionsgleichung der Form

$\sum_{k=0}^{n} a_k(y)\varepsilon_q^k f(y) = 0$. Wegen $\varepsilon_q^k f(y) = \varepsilon_{q^{-1}}^k f(\frac{1}{x}) = \varepsilon_q^{-k} f(\frac{1}{x})$ können wir diese Rekursion in eine q-holonome Rekursion bzgl. x überführen. Um das zu erreichen, muss man die Rekursion mit dem kleinsten gemeinsamen Vielfachen der Nennerpolynome der $a_k(\frac{1}{x})$ multiplizieren und um die Ordnung der Rekursion shiften. Dieses Vorgehen ist ebenfalls in `qHolonomicRE` implementiert.

Maple-Sitzung 5.3 (q-Holonome Rekursionsgleichungen verallg. q-hypergeometrischer Funktionen)

```
> qHolonomicRE(qphihypergeom([1/x],[],x,q),F(x));
```

$$(x-1)F(x) + Sq_x(F(x)) = 0$$

```
> qHolonomicRE(qCharlier(n,a,x,q),F(x));
```

$$aF(x) + (-q^n qx - a + 1)\,Sq_x(F(x)) + (qx-1)\left(Sq_{x,x}\right)(F(x)) = 0$$

```
> infolevel[qHolonomicRE]:=4:
> qHolonomicRE(qphihypergeom([1/q^n,x],[0],-q^n*q/a,q),F(x));
qphihypergeomRE: q-holonomic recurrence equation for coefficients
```

$$\left(q^n - q^k\right) q\,A(k) - \left(-1 + q^k q\right) a\,A(k+1) = 0$$

```
qphihypergeomRE: base=qpochhammer with expansionpt
```

$$1$$

$$-aF(x) + (q^n qx - 1 + a)\,Sq_x(F(x)) + (-qx+1)\left(Sq_{x,x}\right)(F(x)) = 0$$

5.1.3 Der q-Taylorsatz

Wir suchen im Folgenden nach formalen Potenzreihen, die eine vorgegebene q-holonome Rekursionsgleichung erfüllen. Dazu benötigen wir eine q-Version des Satzes von Taylor, formulieren aber zunächst einen verallgemeinerten Taylorsatz, ähnlich wie in [KC02].

Satz 5.19 Sei $f(x)$ eine beliebige Funktion, L ein linearer Operator und $a \in \mathbb{F}$ der Entwicklungspunkt. Ferner sei $\mathfrak{B} = \left\{P_j^a(x) \in \mathbb{F}[x] \mid j \in \mathbb{N}_0 \text{ und } \deg_x(P_j^a(x)) = j\right\}$ gegeben mit den Eigenschaften

(a) $P_0^a(a) = 1$ und $P_j^a(a) = 0$ für alle $j \geq 1$

(b) $LP_j^a(x) = \lambda_j P_{j-1}^a(x)$ mit $\lambda_j \in \mathbb{F}$ für alle $j \geq 1$ und $L(1) = 0$,

dann ist

$$f(x) = \sum_{j=0}^{\infty} \frac{[L^j f(x)]_{x=a}}{\prod_{i=1}^{j} \lambda_i} P_j^a(x) \tag{5.20}$$

die zu $f(x)$ gehörige *Taylorreihe* in $\mathbb{F}[[\mathfrak{B}]]$.

Beweis Sei $f(x) = \sum_{j=0}^{\infty} c_j P_j^a(x)$. Bestimmen wir die k-te q-Ableitung von $f(x)$, so erhalten wir

$$\begin{aligned} L^k f(x) &= \sum_{j=0}^{\infty} L^k\left(c_j P_j^a(x)\right) = \sum_{j=0}^{\infty} c_j L^{k-1} LP_j^a(x) \\ &= \sum_{j=1}^{\infty} \lambda_j c_j L^{k-1} P_{j-1}^a(x) = \ldots = \sum_{j=k}^{\infty} \lambda_j \cdot \ldots \cdot \lambda_{j-k+1} c_j P_{j-k}^a(x). \end{aligned}$$

Werten wir die Gleichung an der Stelle a aus, so ergibt sich nach Umstellen

$$c_k = \frac{\left[L^k f(x)\right]_{x=a}}{\lambda_k \cdot \ldots \cdot \lambda_{k-k+1} P_0^a(a)} = \frac{\left[L^k f(x)\right]_{x=a}}{\prod_{i=1}^{k} \lambda_i}.$$

□

Korollar 5.20 Für $L = D_q$ erhalten wir mit $\lambda_j = [j]_q$ die *q-Taylorreihe*

$$f(x) = \sum_{j=0}^{\infty} \frac{\left[D_q^j f(x)\right]_{x=a}}{[j]_q!} (x \ominus a)_q^j$$

und für $L = D_{q^{-1}}$ erhalten wir mit $\hat{\eta}_j = -[j]_q$

$$f(x) = \sum_{j=0}^{\infty} (-1)^j \frac{\left[D^j_{q^{-1}} f(x)\right]_{x=a}}{[j]_q!} (a \ominus x)^j_q .$$

Beweis $\mathfrak{B}^a$ und $\mathfrak{B}_a$ erfüllen offenbar beide die Voraussetzungen des verallgemeinerten Taylorsatzes. Bei $\mathfrak{B}^a$ ist Eigenschaft (b) bzgl. D_q und bei $\mathfrak{B}_a$ bzgl. $D_{q^{-1}}$ gegeben. Siehe dazu auch Tabelle 5.1, deren Einträge die Aussage liefern. Es sind aber auch die einzigen Basen[4], die die Voraussetzungen erfüllen. Wir betrachten zum Beweis $\mathfrak{B}^a$ und legen $\hat{\eta}_j = [j]_q$ fest. Dann ist $P^a_0(x) = 1$ wegen (a). Für $P^a_1(x) = \alpha x + \beta$ erhalten wir $P^a_1(a) = \alpha a + \beta = 0$ aus (a) und $D_q P^a_1(x) = \alpha = 1$ aus (b). Also ist $P^a_1(x) = x - a$. Vollständige Induktion nach j liefert $P^a_j(x) = (x \ominus a)^j_q$. □

5.1.4 Der q-FPS-Algorithmus

Wir können an dieser Stelle alle Algorithmen, die wir bisher vorgestellt haben, nutzen, um zu einer q-holonomen Funktion die zugehörige q-hypergeometrische Potenzreihenentwicklung zu bestimmen und formulieren damit das q-Analogon zum FPS-Algorithmus aus [Koe92] bzw. [Koe06]. Wir gehen sogar noch einen Schritt weiter und erweitern den FPS-Algorithmus dahingehend, dass dieser Linearkombinationen von q-hypergeometrischen Reihen bestimmt. Der q-FPS-Algorithmus fußt auf dem folgenden beschriebenen Basis-Algorithmus.

Algorithmus 5.8 Bestimmung einer q-hypergeometrischen Potenzreihe (bzgl. $\mathfrak{B}^a$ bzw. $\mathfrak{B}_a$) für eine q-holonome Funktion (`convert/qFPS` mit `base=qpower` bzw. `base=qpochhammer` und `expansionpt=a`)

Eingabe : Eine q-holonome Funktion $f(x)$, eine Basis $\mathfrak{B}$ und einen Entwicklungspunkt a

Ausgabe : Eine q-hypergeometrische Potenzreihe $\sum_{j=0}^{\infty} c_j P^a_j(x)$ für $f(x)$

1 **begin**
2 $qRE \leftarrow$ Bestimme eine q-holonome Rekursionsgleichung für $f(x)$ mit Algorithmus 3.1
3 $RE \leftarrow$ Konvertiere qRE in eine q-holonome Rekursionsgleichung für c_j bzgl. Basis $\mathfrak{B}$ und Entwicklungspunkt a mit Algorithmus 5.1
4 Bestimme alle m-q-hypergeometrischen Lösungen von RE mit Satz 4.35 (sofern RE vom q-hypergeometrischen Typ ist) bzw. alle q-hypergeometrischen Lösungen von RE mit dem modifizierten q-Petkovšek-Algorithmus (Algorithmus 4.9)
5 Bestimme mit hinreichend vielen Anfangswerten die q-Reihe mit Hilfe der vorigen Lösung
6 **return** *die daraus entstehende Potenzreihe*
7 **end**

Dieser Algorithmus bestimmt u.a. zu einer q-holonomen Funktion die entsprechende q-hypergeometrische Potenzreihe bzgl. einer gegebenen Basis, wenn eine solche Reihe existiert. Voraussetzung dafür ist, dass der verwendete Algorithmus in Zeile 2 alle linearen Abhängigkeiten der Eingabe erkennt, was in der Praxis schwierig sein kann (siehe auch Algorithmus 3.1). Nach Satz 5.15 ist eine Potenzreihe genau dann eine Lösung der q-holonomen Rekursionsgleichung qRE, wenn die Koeffizienten der Reihe die induzierte Rekursion RE erfüllen. Alle q-hypergeometrischen Lösungen von RE können wir aber mit unseren Algorithmen aus Kapitel 4 bestimmen. In Zeile 5 wird schließlich mit Hilfe von Anfangswerten die Potenzreihenlösung bestimmt. Diese Vorgehensweise betrachten wir im Folgenden genauer.

Bekanntlich ist durch eine Rekursionsgleichung n-ter Ordung für c_j und n Anfangswerten die Folge c_j, abgesehen von evtl. Singularitäten, eindeutig bestimmt. Diese Angaben bilden zusammen

[4] Eindeutigkeit bis auf einen Vorfaktor, der abhängig von j ist

eine Normalform. Wie bestimmt man aber in unserem Fall die Anfangswerte? Gehen wir von der Darstellung $f(x) = \sum_{j=0}^{\infty} c_j P_j^a(x)$ aus, dann gilt nach Satz 5.19 (für explizite Darstellung bzgl. $\mathfrak{B}^a$ bzw. $\mathfrak{B}_a$ siehe Satz 5.20) für $j \in \mathbb{N}_0$ und $i \in \mathbb{N}$

$$c_j = \frac{[L^j f(x)]_{x=a}}{\prod_{i=1}^{j} \lambda_i} \quad \text{mit} \quad \lambda_i = \frac{LP_i^a(x)}{P_{i-1}^a(x)}. \tag{5.21}$$

Folglich muss man die Werte $[L^j f(x)]_{x=a}$ kennen, um die Koeffizienten c_j der Potenzreihe zu bestimmen. Liefert die Anwendung des Satzes 4.35 eine m-q-hypergeometrische Lösung der Form $\{c_{mj}, c_{mj+1}, \ldots, c_{mj+m-1}\}$, so können wir $f(x)$ unter Bestimmung der Anfangswerte c_{ms_k+k} (siehe Satz 4.35 für s_k) mit Hilfe von (5.21) für $k = 0 \ldots m-1$ angeben als

$$\sum_{j=0}^{\infty} c_{mj} P_{mj}^a(x) + \sum_{j=0}^{\infty} c_{mj+1} P_{mj+1}^a(x) + \ldots + \sum_{j=0}^{\infty} c_{mj+m-1} P_{mj+m-1}^a(x).$$

Erhalten wir keine m-q-hypergeometrischen Lösungen, dann führen wir den modifizierten q-Petkovšek-Algorithmus aus. Dieser liefert uns eine Basis q-hypergeometrischer Lösungen, sofern welche existieren. Sei nun

$$\left\{c_j^{(1)}, c_j^{(2)}, \ldots, c_j^{(l)}\right\}$$

eine Basis q-hypergeometrischer Lösungen einer q-holonomen Rekursion für die Koeffizienten c_j der Ordnung n mit $l \leq n$. Dann stellen wir mit Hilfe der ersten n Anfangswerte ein lineares Gleichungssystem in den Unbekannten $t_k \in \mathbb{F}$ auf mit

$$\sum_{k=1}^{l} t_k c_r^{(k)} = \frac{[L^r f(x)]_{x=a}}{\prod_{i=1}^{r} \lambda_i} \quad \text{mit} \quad r = 0, \ldots, n-1$$

und lösen dies. Existiert eine Lösung, so erhalten wir die q-Potenzreihenentwicklung

$$f(x) = \sum_{j=0}^{\infty} \left(\sum_{k=1}^{l} t_k c_j^{(k)} \right) P_j^a(x),$$

die i.Allg. nicht mehr q-hypergeometrisch ist. Diese Methode setzt voraus, dass die Anfangswerte für $r = 0, \ldots, n-1$ definiert sind und nicht alle verschwinden. Das können wir durch folgende Vorgehensweise immer erreichen. Bestimmen wir direkt aus der Rekursionsgleichung den Index m, für den c_j für alle $j \geq m$ definiert ist und nicht verschwindet. Sei also $\sum_{k=0}^{n} a_k(q^j) \varepsilon^k c_j = 0$ gegeben mit $a_k \in \mathbb{F}[q^j]$ und c_j eine q-hypergeometrische Lösung. Nach Satz 4.29 gilt dann für das q-Zertifikat $r(q^j) := \frac{c_{j+1}}{c_j} = \frac{s(q^j)}{t(q^j)}$ mit $s, t \in \mathbb{F}[q^j]$

$$s(q^j) \mid a_0(q^j) \qquad \text{und} \qquad t(q^j) \mid a_n(q^{j-n+1}).$$

Sei nun $M := \{i \in \mathbb{N}_0 \mid q^i \text{ ist Nullstelle von } a_0(q^j) a_n(q^{j-n+1})\}$. Dann sind offenbar alle $i \in \mathbb{N}_0$ in M enthalten, für die das q-Zertifikat $r(q^j)$ Nullstellen und Pole der Form q^i besitzt. Somit ist $m = 1 + \max M \cup \{-1\}$. Wir bilden aus diesem Grund $\tilde{r}(q^j) := r(q^{j+m})$ und erhalten aus diesem q-Zertifikat einen q-hypergeometrischen Term $\tilde{c}_j$, der für alle $j \geq 0$ definiert ist.

In `qFPS` ist Algorithmus 5.8 in der Prozedur `'convert/qFPS'` implementiert. Betrachten wir dazu die folgende Maple-Sitzung.

Maple-Sitzung 5.4 (Bestimmung formaler q-Potenzreihen)

Endliche Summen

```
> PS:=convert(x^n,qFPS,x,expansionpt=1);
```

$$PS := \sum_{k=0}^{\infty} \frac{(-1)^k q^{-1/2\,k(-2\,n+k-1)} \, qpochhammer\left(q^{-n}, q, k\right) qpower\left(x, 1, q, k\right)}{qpochhammer\left(q, q, k\right)}$$

```
> convert(PS,qbinomial);
```

$$\sum_{k=0}^{\infty} qbinomial\left(n, k, q\right) qpower\left(x, 1, q, k\right)$$

```
> convert(qLaguerre(n,alpha,x,q),qFPS,x,base=qpochhammer,expansionpt=-1);
```

$$\sum_{k=0}^{\infty} \frac{(-1)^k q^{k(1+n+\alpha)} qpochhammer\left(q^{-n},q,k\right) qpower\left(-1,x,q,k\right)}{qpochhammer\left(q,q,n\right) qpochhammer\left(q,q,k\right)}$$

```
> convert(DiscreteqHermiteII(n,x,q),qFPS,x,base=qpochhammer,expansionpt=-I);
```

$$\sum_{k=0}^{\infty} \frac{(-1)^n \mathrm{e}^{1/2\, i\pi\,(n+k)} q^{-1/2\, n^2+1/2\, n+kn-1/2\, k^2+1/2\, k} qpochhammer\left(q^{-n},q,k\right) qpower\left(-i,x,q,k\right)}{qpochhammer\left(q,q,k\right)}$$

q-Hypergeometrische Potenzreihen

```
> convert(qexp(x,q),qFPS,base=qpochhammer,expansionpt=a,var=1/q);
```

$$\sum_{k=0}^{\infty} \frac{qexp\left(a,q\right) q^{-k} qpower\left(a,x,q^{-1},k\right)}{qpochhammer\quad\left(q^{-1},q^{-1},k\right)}$$

```
> convert(qsin(x,q)+qSin(x,1/q),qFPS);
```

$$\sum_{k=0}^{\infty} -\frac{(-1)^k x^{2\,k+1}\left(q^{2\,k+1}-1\right)}{qpochhammer\left(q,q,2\,k+1\right)}$$

```
> convert(qsin(x,q)+x*qcos(x,q),qFPS);
```

$$\sum_{k=0}^{\infty} -\frac{(-1)^k x^{2\,k+1}\left(-2+q^{2\,k+1}\right)}{qpochhammer\left(q,q,2\,k+1\right)}$$

Beweis von Identitäten

```
> PS:=convert(qphihypergeom([b/a],[b],a*x,q)/qpochhammer(x,q,infinity),qFPS,x);
```

$$PS := \sum_{k=0}^{\infty} \frac{qpochhammer\left(a,q,k\right) x^k}{qpochhammer\left(q,q,k\right) qpochhammer\left(b,q,k\right)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\left(\left[a,0\right],\left[b\right],x,q\right)$$

```
> convert(Sinq(x,q)-sinq(x,1/q),qFPS);
```

$$0$$

Linearkombinationen q-hypergeometrischer Potenzreihen

```
> convert(qsin(x,q),qFPS,expansionpt=a);
```

$$\sum_{k=0}^{\infty} \frac{(-1)^k qsin\left(a,q\right) qpower\left(x,a,q,2\,k\right)}{qpochhammer\left(q,q,2\,k\right)} + \sum_{k=0}^{\infty} \frac{(-1)^k qcos\left(a,q\right) qpower\left(x,a,q,2\,k+1\right)}{qpochhammer\left(q,q,2\,k+1\right)}$$

```
> convert(qcos(x,q)*qsin(x,1/q),qFPS,x);
```

$$\sum_{k=0}^{\infty} \frac{1}{4} \frac{\left(2\, qpochhammer\left(q,q,2\,k+1\right)-\left(1+q^{2\,k+1}\right)qpochhammer\left(-1,q,2\,k+1\right)\right)\,(-1)^k x^{2\,k+1}}{qpochhammer\left(q,q,2\,k+1\right)}$$

```
> convert(sinq(x,q)*qExp(x,q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{1}{2}\, i\, x^k \left(-qpochhammer\left(\frac{-i}{q-1},q,k\right)+(-1)^k qpochhammer\left(\frac{i}{q-1},q,k\right)\right) \mathrm{e}^{1/2\, ik\pi}\left(qfactorial\left(k,q\right)\right)^{-1}$$

```
> convert((a*x+b)*qcos(x,q)+(c*x+d)*qsin(x,q),qFPS,x);
```

$$\sum_{k=0}^{\infty} \frac{x^{2\,k}(-1)^k\left(b-c+q^{2\,k}c\right)}{qpochhammer\left(q,q,2\,k\right)} + \sum_{k=0}^{\infty} -\frac{(-1)^k x^{2\,k+1}\left(-a+q^{2\,k+1}a-d\right)}{qpochhammer\left(q,q,2\,k+1\right)}$$

Produkte von Polynomen und q-hypergeometrischen Reihen sind auch wieder q-hypergeometrische Reihen

```
> convert((a*x^2+b*x+c)*qexp(x,q),qFPS,x,termwise);
```

$$c+\frac{(-c+bq-b)\,x}{q-1} + \sum_{k=0}^{\infty} -\frac{\left(-c+bq^{2+k}-b-aq^{3+2\,k}+aq^{2+k}+q^{k+1}a-a\right)x^{2+k}}{\left(q^{k+1}-1\right)\left(q^{2+k}-1\right)qpochhammer\left(q,q,k\right)}$$

Weitere Beispiele folgen im Abschnitt 5.2. Für eine Beschreibung der sehr mächtigen Prozedur `'convert/qFPS'` siehe Seite 115. Der Algorithmus der Prozedur `'convert/qphihypergeom'`,

die in der Maple-Sitzung verwendet wurde, wird im nächsten Abschnitt formuliert.

5.1.5 Konversion einer q-Reihe in die Darstellung der verallgemeinerten q-hypergeometrischen Funktion

Um eine formale q-hypergeometrische Potenzreihe in die Darstellung einer verallgemeinerten q-hypergeometrischen Funktion zu bringen, kann man folgenden Algorithmus verwenden.

Algorithmus 5.9 Bestimmung der Darstellung einer q-hypergeometrischen Potenzreihe als verallgemeinerte q-hypergeometrische Funktion (`'convert/qphihypergeom'`)

Eingabe : Eine q-hypergeometrische Potenzreihe $\sum_{j=0}^{\infty} c_j P_j^a(x)$

Ausgabe : Die Darstellung der Reihe als verallgemeinerte q-hypergeometrische Funktion

begin
- $upper \leftarrow \emptyset$, $lower \leftarrow \emptyset$
- $\tilde{r}(q^j) \leftarrow \frac{c_{j+1}}{c_j} \frac{P_{j+1}^a(x)}{P_j^a(x)}$
- $\tilde{s}(q^j) \leftarrow \text{numer}(\tilde{r}(q^j))$
- $\tilde{t}(q^j) \leftarrow \text{denom}(\tilde{r}(q^j))$
- $\tilde{c}_0 \leftarrow \frac{\text{tcoeff}_{q^j}(\tilde{s}(q^j))}{\text{tcoeff}_{q^j}(\tilde{t}(q^j))}$
- $v_0 \leftarrow v_{\text{ldeg}}(\tilde{s}(q^j)) - v_{\text{ldeg}}(\tilde{t}(q^j))$
- $\hat{s}(q^j) \leftarrow$ Faktorisiere numer $\left(\frac{\tilde{r}(q^j)}{(q^j)^{v_0}}\right)$ über einem geeigneten Erweiterungskörper
- $\hat{t}(q^j) \leftarrow$ Faktorisiere denom $\left(\frac{\tilde{r}(q^j)}{(q^j)^{v_0}}\right)$ über einem geeigneten Erweiterungskörper
- $m \leftarrow \deg_{q^j}(\hat{s}(q^j))$
- $n \leftarrow \deg_{q^j}(\hat{t}(q^j))$
- **if** $(1 - qq^j)$ *ist kein Teiler von* $\hat{t}(q^j)$ **then**
  - $r \leftarrow m + 1$, $s \leftarrow n$, $\hat{s}(q^j) \leftarrow (1 - qq^j)\hat{s}(q^j)$
- **else**
  - $r \leftarrow m$, $s \leftarrow n - 1$, $\hat{t}(q^j) \leftarrow \frac{\hat{t}(q^j)}{1-qq^j}$
- **end**
- **for** *jeden Linearfaktor* $(1 - \alpha q^j)$ *von* $\hat{s}(q^j)$ **do**
  - $upper \leftarrow upper \cup \{\alpha\}$
- **end**
- **for** *jeden Linearfaktor* $(1 - \beta q^j)$ *von* $\hat{t}(q^j)$ **do**
  - $lower \leftarrow lower \cup \{\beta\}$
- **end**
- **if** $v_0 < 1 + s - r$ **then**
  - $r \leftarrow 1 + s - v_0$, $upper \leftarrow$ Füge $1 + s - r - v_0$ Nullen zu *upper* hinzu
- **else**
  - $s \leftarrow v_0 + r - 1$, $lower \leftarrow$ Füge $v_0 + r - s - 1$ Nullen zu *lower* hinzu
- **end**
- **return** $c_0\, {}_r\phi_s\left(\begin{matrix} upper \\ lower \end{matrix} \middle| q; \frac{\tilde{c}_0}{(-1)^{v_0}}\right)$

end

Beweis Es gelte

$$\sum_{j=0}^{\infty} c_j P_j^a(x) = \sum_{j=0}^{\infty} C_j = c_0 \, {}_r\phi_s\left(\begin{matrix} \boldsymbol{a} \\ \boldsymbol{b} \end{matrix} \,\middle|\, q; \tilde{x} \right)$$

mit noch unbestimmten $r, s \in \mathbb{N}_0$, $\tilde{x}$, oberen Parametern a_i und unteren Parametern b_j, sowie mit bekanntem q-hypergeometrischem C_j. Dann ergibt sich das Termverhältnis

$$\tilde{r}(q^j) := \frac{C_{j+1}}{C_j} = (-1)^{1+s-r} \left(q^j\right)^{1+s-r} \frac{(1-a_1q^j)\cdot\ldots\cdot(1-a_rq^j)}{(1-b_1q^j)\cdot\ldots\cdot(1-b_sq^j)} \frac{\tilde{x}}{(1-qq^j)} \in \overline{\mathbb{F}}(x)(q^j).$$

Ferner sei $\tilde{r}(q^j) = \frac{\tilde{s}(q^j)}{\tilde{t}(q^j)}$ mit $\tilde{s}, \tilde{t} \in \overline{\mathbb{F}}(x)[q^j]$. Dann ist offenbar

$$\mathrm{ltyp}_0(C_j) = (\tilde{c}_0, v_0) = \left(\frac{\mathrm{tcoeff}_{q^j}(\tilde{s}(q^j))}{\mathrm{tcoeff}_{q^j}(\tilde{t}(q^j))}, v_{\mathrm{ldeg}}(\tilde{s}(q^j)) - v_{\mathrm{ldeg}}(\tilde{t}(q^j)) \right) = \left((-1)^{1+s-r}\,\tilde{x}, 1+s-r \right),$$

d.h. es gilt $\tilde{x} = \frac{\tilde{c}_0}{(-1)^{v_0}}$. Außerdem erfüllen die Zahlen r und s die Gleichung $1+s-r = v_0$. Sei nun

$$\tilde{r}(q^j) = \left(q^j\right)^{v_0} \frac{\hat{s}(q^j)}{\hat{t}(q^j)} \quad \text{mit} \quad \hat{s}, \hat{t} \in \overline{\mathbb{F}}(x)[q^j],$$

wobei beide Polynome über $\overline{\mathbb{F}}(x)$ wie oben komplett in Linearfaktoren zerfallen. In den Zeilen 12-16 wird der Faktor $(1-qq^j)$ aus dem Nenner von $\hat{t}(q^j)$ gestrichen (falls vorhanden) oder zum Zähler hinzugefügt (falls nicht vorhanden), so dass in beiden Polynomen nur noch Linearfaktoren auftreten, die einem oberen oder unteren Parameter entsprechen. Die Zahlen r bzw. s, die später die Anzahlen der oberen Parameter bzw. unteren Parameter angeben, werden zudem angepasst. In den darauffolgenden Zeilen werden die Parameter aus den Linearfaktoren gemäß Korollar 4.36 abgelesen. Gilt $v_0 \neq 1+s-r$, so müssen den oberen bzw. unteren Parametern noch Nullen hinzugefügt werden, die jeweils einem Faktor $1 = (0;q)_j$ bei C_j entsprechen. Diese Anpassung findet in den Zeilen 23-27 statt. □

Maple-Sitzung 5.5 (Konversion in verallgemeinerte q-hypergeometrische Funktion)

```
> PS:=convert(qexp(x,q),qFPS);
```

$$PS := \sum_{k=0}^{\infty} \frac{x^k}{qpochhammer\,(q,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\,([0],[\,],x,q)$$

```
> PS:=convert(qCharlier(n,a,x,q),qFPS,x);
```

$$PS := \sum_{k=0}^{\infty} q^{1/2\,k(2\,n+k+1)}\, qpochhammer\left(q^{-n},q,k\right) a^{-k}\, qpochhammer\left(-\frac{q}{a},q,n\right) x^k \left(qpochhammer\,(q,q,k)\right)^{-1} \left(qpochhammer\left(-\frac{q}{a},q,k\right)\right)^{-1}$$

```
> convert(PS,qphihypergeom);
```

$$qpochhammer\left(-\frac{q}{a},q,n\right) qphihypergeom\left(\left[q^{-n}\right],\left[-\frac{q}{a}\right],-\frac{q^n q x}{a},q\right)$$

```
> PS:=convert(qCharlier(n,a,x,q),qFPS,x,base=qpochhammer,expansionpt=1);
```

$$PS := \sum_{k=0}^{\infty} \frac{(-1)^k\, q^{k(1+n)}\, qpochhammer\left(q^{-n},q,k\right) a^{-k}\, qpochhammer\,(x,q,k)}{qpochhammer\,(q,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\left(\left[q^{-n},x\right],[0],-\frac{q^n q}{a},q\right)$$

5.2 Anwendungen

In diesem Abschnitt betrachten wir einige Anwendungsbeispiele von `qFPS`. Der q-Zeilberger-Algorithmus ([Koo93],[Böi98]) ist in der Lage, durch Eingabe eines q-hypergeometrischen Terms $F(n,k)$ in n und k eine q-holonome Rekursionsgleichung für $s_n = \sum_k F(n,k)$ zu bestimmen, wobei $F(n,k) \neq 0$ nur für endlich viele Summanden gelte. Aus dieser Rekursionsgleichung kann man in vielen Fällen durch Anwendung des q-Petkovšek-Algorithmus eine geschlossene Form für s_n bestimmen. Für einige der folgenden Anwendungsbeispiele kann z.B. der q-Zeilberger-Algorithmus die betrachteten Identitäten mit Kenntnis von $F(n,k)$ herleiten. Mit unserem q-FPS-Algorithmus können wir u.a. nun umgekehrt aus s_n den Summanden $F(n,k)$ bestimmen, sofern s_n eine q-Potenzreihenentwicklung bzgl. einer der Basen $\mathfrak{B}^a$ und $\mathfrak{B}_a$ besitzt. Wir sind aber nicht nur auf endliche Summen beschränkt, sondern können den Algorithmus zum eigentlichen Zweck, dem Bestimmen von q-hypergeometrischen Potenzreihen verwenden, um z.B. Transformationsformeln oder q-hypergeometrische Reihendarstellungen von q-Taylorkoeffizienten erzeugender Funktionen herzuleiten. Die folgenden Formeln, die wir mit Maple beweisen werden, stammen aus [GR90] bzw. [KS98].

5.2.1 Das q-Binomial-Theorem

Das *q-Binomial-Theorem* ist gegeben durch

$$ {}_1\phi_0\left(\begin{matrix} a \\ - \end{matrix} \middle| q; x \right) = \sum_{j=0}^{\infty} \frac{(a;q)_j}{(q;q)_j} x^j = \frac{(ax;q)_\infty}{(x;q)_\infty} \quad \text{für} \quad |x| < 1 \quad \text{und} \quad |q| < 1. $$

Da wir uns nur für formale Potenzreihen interessieren, werden wir bei den folgenden Identitäten die Voraussetzungen zur Konvergenz vernachlässigen. Wir leiten das q-Binomial-Theorem und weitere alternative Formeln des q-Binomial-Theorems, die man z.B. in [Gas96] nachschlagen kann, mit `qFPS` her.

Maple-Sitzung 5.6 (Das q-Binomial-Theorem)

```
> PS:=convert(qpochhammer(a*x,q,infinity)/qpochhammer(x,q,infinity),qFPS,x);
```

$$PS := \sum_{k=0}^{\infty} \frac{qpochhammer(a,q,k)\,x^k}{qpochhammer(q,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom([a],[],x,q)$$

Umgekehrt

```
> convert(qphihypergeom([a],[],x,q),qFPS,x);
```

$$\sum_{k=0}^{\infty} \frac{qpochhammer(a,q,k)\,x^k}{qpochhammer(q,q,k)}$$

Alternative Formeln des q-Binomial-Theorems

```
> convert(qpochhammer(x/q,1/q,infinity)/qpochhammer(a*x/q,1/q,infinity),
> qFPS,x);
```

$$\sum_{k=0}^{\infty} \frac{qpochhammer(a,q,k)\,x^k}{qpochhammer(q,q,k)}$$

```
> convert(convert(qpochhammer(x*q^(-n),q,n),qFPS,x),qbinomial);
```

$$\sum_{k=0}^{\infty} (-1)^k\, qbinomial(n,k,q)\, q^{1/2\,k(-2\,n+k-1)} x^k$$

```
> convert(convert(y^n*qpochhammer(-x*q^(-n)/y,q,n),qFPS,x),qbinomial);
```

$$\sum_{k=0}^{\infty} qbinomial(n,k,q)\, q^{1/2\,k(-2\,n+k-1)} x^k y^{n-k}$$

```
> convert(convert(qpower(x,-y,q,n),qFPS,x),qbinomial);
```

$$\sum_{k=0}^{\infty} qbinomial\left(n,k,q\right) q^{1/2\,(k-n+1)(-n+k)} x^k y^{n-k}$$

5.2.2 Summations- und Transformationsformeln

Wir beweisen an dieser Stelle einige Summationsformeln, darunter

$$ {}_1\phi_1\left(\begin{matrix} a \\ c \end{matrix} \middle| q; \frac{c}{a}\right) = \frac{\left(\frac{c}{a};q\right)_\infty}{\left(c;q\right)_\infty}$$

$$ {}_2\phi_1\left(\begin{matrix} a,b \\ c \end{matrix} \middle| q; \frac{c}{ab}\right) = \frac{\left(\frac{c}{a};q\right)_\infty \left(\frac{c}{b};q\right)_\infty}{\left(c;q\right)_\infty \left(\frac{c}{ab};q\right)_\infty} \qquad (q\text{-Gauß I})$$

$$ {}_2\phi_1\left(\begin{matrix} a,b \\ c \end{matrix} \middle| q; q\right) = \frac{\left(\frac{a}{c};q^{-1}\right)_\infty \left(\frac{b}{c};q^{-1}\right)_\infty}{\left(\frac{1}{c};q^{-1}\right)_\infty \left(\frac{ab}{c};q^{-1}\right)_\infty} \qquad (q\text{-Gauß II})$$

$$ {}_2\phi_1\left(\begin{matrix} q^{-n},b \\ c \end{matrix} \middle| q; \frac{cq^n}{b}\right) = \frac{\left(\frac{c}{b};q\right)_n}{\left(c;q\right)_n} \qquad (q\text{-Chu-Vandermonde I})$$

$$ {}_2\phi_1\left(\begin{matrix} q^{-n},b \\ c \end{matrix} \middle| q; q\right) = \frac{\left(\frac{c}{b};q\right)_n}{\left(c;q\right)_n} b^n \qquad (q\text{-Chu-Vandermonde II})$$

mit $a,b,c \in \mathbb{F}$ und $n \in \mathbb{N}_0$.

Maple-Sitzung 5.7 (Summationsformeln)

Summationsformel für ${}_1\phi_1$

```
> term:=qpochhammer(c/a,q,infinity)/qpochhammer(c,q,infinity);
```

$$term := qpochhammer\left(\frac{c}{a},q,\infty\right)\left(qpochhammer\left(c,q,\infty\right)\right)^{-1}$$

```
> PS:=subs(a=1/a,convert(subs(a=1/a,term),qFPS,a,expansionpt=1));
```

$$PS := \sum_{k=0}^{\infty} \frac{(-1)^k q^{1/2\,(k-1)k} c^k qpower\left(a^{-1},1,q,k\right)}{qpochhammer\left(q,q,k\right) qpochhammer\left(c,q,k\right)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\left(\left[a\right],\left[c\right],\frac{c}{a},q\right)$$

q-Gauß

```
> term:=qpochhammer(c/a,q,infinity)*qpochhammer(c/b,q,infinity)/
> qpochhammer(c,q,infinity)/qpochhammer(c/a/b,q,infinity);
```

$$term := qpochhammer\left(\frac{c}{a},q,\infty\right) qpochhammer\left(\frac{c}{b},q,\infty\right)\left(qpochhammer\left(c,q,\infty\right)\right)^{-1} \left(qpochhammer\left(\frac{c}{ab},q,\infty\right)\right)^{-1}$$

```
> PS:=subs(b=1/b,convert(subs(b=1/b,term),qFPS,b,expansionpt=1));
```

$$PS := \sum_{k=0}^{\infty} \frac{c^k qpochhammer\left(a,q,k\right) a^{-k} qpower\left(b^{-1},1,q,k\right)}{qpochhammer\left(q,q,k\right) qpochhammer\left(c,q,k\right)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\left(\left[a,b\right],\left[c\right],\frac{c}{ab},q\right)$$

```
> term:=qpochhammer(a/c,1/q,infinity)*qpochhammer(b/c,1/q,infinity)/
> qpochhammer(1/c,1/q,infinity)/qpochhammer(a*b/c,1/q,infinity);
```

$$term := qpochhammer\left(\frac{a}{c},q^{-1},\infty\right) qpochhammer\left(\frac{b}{c},q^{-1},\infty\right)\left(qpochhammer\left(c^{-1},q^{-1},\infty\right)\right)^{-1} \left(qpochhammer\left(\frac{ab}{c},q^{-1},\infty\right)\right)^{-1}$$

```
> PS:=convert(term,qFPS,b,base=qpochhammer,expansionpt=1);
```

$$PS := \sum_{k=0}^{\infty} \frac{q^k\, qpochhammer(a,q,k)\, qpochhammer(b,q,k)}{qpochhammer(q,q,k)\, qpochhammer(c,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom([a,b],[c],q,q)$$

q-Chu-Vandermonde

```
> term:=qpochhammer(c/b,q,n)/qpochhammer(c,q,n);
```

$$term := qpochhammer\left(\frac{c}{b},q,n\right)(qpochhammer(c,q,n))^{-1}$$

```
> PS:=subs(b=1/b,convert(subs(b=1/b,term),qFPS,b,expansionpt=1));
```

$$PS := \sum_{k=0}^{\infty} \frac{q^{nk}\, qpochhammer\left(q^{-n},q,k\right) c^k\, qpower\left(b^{-1},1,q,k\right)}{qpochhammer(q,q,k)\, qpochhammer(c,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\left(\left[q^{-n},b\right],[c],\frac{cq^n}{b},q\right)$$

```
> term:=qpochhammer(c/b,q,n)/qpochhammer(c,q,n)*b^n;
```

$$term := qpochhammer\left(\frac{c}{b},q,n\right) b^n (qpochhammer(c,q,n))^{-1}$$

```
> PS:=convert(term,qFPS,b,base=qpochhammer,expansionpt=1);
```

$$PS := \sum_{k=0}^{\infty} \frac{qpochhammer\left(q^{-n},q,k\right) q^k\, qpochhammer(b,q,k)}{qpochhammer(q,q,k) \quad qpochhammer(c,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$qphihypergeom\left(\left[q^{-n},b\right],[c],q,q\right)$$

Ferner können wir viele bekannte Transformationsformeln überprüfen. Exemplarisch zeigen wir

$$ {}_2\phi_1\left(\begin{matrix}a,b\\c\end{matrix}\,\middle|\,q;x\right) = \frac{(ax;q)_\infty (b;q)_\infty}{(c;q)_\infty (x;q)_\infty}\, {}_2\phi_1\left(\begin{matrix}\frac{c}{b},x\\ax\end{matrix}\,\middle|\,q;b\right) \qquad (q\text{-Heine I})$$

$$ {}_2\phi_1\left(\begin{matrix}a,b\\c\end{matrix}\,\middle|\,q;x\right) = \frac{\left(\frac{ab}{c}x;q\right)_\infty}{(x;q)_\infty}\, {}_2\phi_1\left(\begin{matrix}\frac{c}{a},\frac{c}{b}\\c\end{matrix}\,\middle|\,q;\frac{abx}{c}\right) \qquad (q\text{-Heine II})$$

$$ {}_2\phi_1\left(\begin{matrix}a,b\\c\end{matrix}\,\middle|\,q;x\right) = \frac{(ax;q)_\infty}{(x;q)_\infty}\, {}_2\phi_2\left(\begin{matrix}a,\frac{c}{b}\\c,ax\end{matrix}\,\middle|\,q;bx\right) \qquad (q\text{-Pfaff-Kummer})$$

$$ {}_3\phi_2\left(\begin{matrix}q^{-n},a,0\\b,c\end{matrix}\,\middle|\,q;q\right) = \frac{\left(\frac{q}{b};q\right)_\infty}{\left(\frac{q}{b}q^{-n};q\right)_\infty}\, {}_2\phi_1\left(\begin{matrix}q^{-n},\frac{c}{a}\\c\end{matrix}\,\middle|\,q;\frac{aq}{b}\right) \qquad (q\text{-Jackson})$$

mit $a,b,c,x \in \mathbb{F}$ und $n \in \mathbb{N}_0$. Bei den Beweisen der ersten drei Formeln bringen wir die q-Pochhammersymbole zunächst auf die linke Seite.

Maple-Sitzung 5.8 (Transformationsformeln)

q-Heine

```
> PS:=convert(qpochhammer(c,q,infinity)*qpochhammer(x,q,infinity)/
> qpochhammer(b,q,infinity)/qpochhammer(a*x,q,infinity)*
> qphihypergeom([a,b],[c],x,q),qFPS,b,expansionpt=c,initialcheck=false);
```

$$PS := \sum_{k=0}^{\infty} \frac{qpochhammer(x,q,\infty)\, qphihypergeom([a,c],[c],x,q)\, qpochhammer(x,q,k)\, qpower(b,c,q,k)}{qpochhammer(ax,q,\infty)\, qpochhammer(q,q,k)\, qpochhammer(ax,q,k)}$$

```
> convert(subs(qphihypergeom([a,c],[c],x,q)=qpochhammer(a*x,q,infinity)/
> qpochhammer(x,q,infinity),PS),qphihypergeom);
```

$$qphihypergeom\left(\left[\frac{c}{b},x\right],[ax],b,q\right)$$

```
> PS:=convert(qpochhammer(x,q,infinity)/qpochhammer(a*b*x/c,q,infinity)*
> qphihypergeom([a,b],[c],x,q),qFPS,b,expansionpt=c,initialcheck=false);
```

$$PS := \sum_{k=0}^{\infty} qpochhammer(x,q,\infty)\, qphihypergeom([a,c],[c],x,q)\, a^k qpochhammer\left(\frac{c}{a},q,k\right) x^k c^{-k}$$
$$qpower(b,c,q,k)\,(qpochhammer(ax,q,\infty))^{-1}\,(qpochhammer(q,q,k))^{-1}\,(qpochhammer(c,q,k))^{-1}$$

```
> convert(subs(qphihypergeom([a,c],[c],x,q)=qpochhammer(a*x,q,infinity)/
> qpochhammer(x,q,infinity),PS),qphihypergeom);
```

$$qphihypergeom\left(\left[\frac{c}{a},\frac{c}{b}\right],[c],\frac{abx}{c},q\right)$$

q-Pfaff-Kummer

```
> PS:=convert(qpochhammer(x,q,infinity)/qpochhammer(a*x,q,infinity)*
> qphihypergeom([a,b],[c],x,q),qFPS,b,expansionpt=c,initialcheck=false);
```

$$PS := \sum_{k=0}^{\infty} qpochhammer(x,q,\infty)\, qphihypergeom([a,c],[c],x,q)\ (-1)^k q^{1/2\,k(k-1)} x^k qpochhammer(a,q,k)$$
$$qpower(b,c,q,k)\ (qpochhammer(ax,q,\infty))^{-1}\,(qpochhammer(q,q,k))^{-1}\,(qpochhammer(c,q,k))^{-1}$$
$$(qpochhammer(ax,q,k))^{-1}$$

```
> convert(subs(qphihypergeom([a,c],[c],x,q)=qpochhammer(a*x,q,infinity)/
> qpochhammer(x,q,infinity),PS),qphihypergeom);
```

$$qphihypergeom\left(\left[a,\frac{c}{b}\right],[c,ax],bx,q\right)$$

q-Jackson

```
> PS:=convert(qpochhammer(q/b,q,infinity)/qpochhammer(q/b/q^n,q,infinity)*
> qphihypergeom([1/q^n,c/a],[c],a*q/b,q),qFPS,a,base=qpochhammer,expansionpt=1,
> initialcheck=false);
```

$$PS := \sum_{k=0}^{\infty} qpochhammer\left(\frac{q}{b},q,\infty\right) qphihypergeom\left([q^{-n},c],[c],\frac{q}{b},q\right)\quad qpochhammer\left(q^{-n},q,k\right) q^k$$
$$qpochhammer(a,q,k)\left(qpochhammer\left(\frac{q}{q^n b},q,\infty\right)\right)^{-1}(qpochhammer(q,q,k))^{-1}$$
$$(qpochhammer(c,q,k))^{-1}\,(qpochhammer(b,q,k))^{-1}$$

```
> convert(subs(b=1/b,qpochhammer(q/b,q,infinity)*
> qphihypergeom([1/q^n,c],[c],q/b,q)/qpochhammer(q/b/q^n,q,infinity)),qFPS,b);
```

$$1$$

```
> convert(subs(qphihypergeom([1/q^n,c],[c],q/b,q)=
> qpochhammer(q/b/q^n,q,infinity)/qpochhammer(q/b,q,infinity),PS),
> qphihypergeom);
```

$$qphihypergeom\left(\left[q^{-n},a,0\right],[b,c],q,q\right)$$

5.2.3 Erzeugende Funktionen

In [Koe03] wird die folgende Methode verwendet, um zu einer erzeugenden Funktion

$$w(x,t) = \sum_{i=0}^{\infty} C_i(x) t^i$$

die hypergeometrische Potenzreihendarstellung

$$C_i(x) = \sum_{j=0}^{\infty} c_j^{(i)} x^j \quad \text{mit} \quad \frac{c_{j+1}^{(i)}}{c_j^{(i)}} \in \mathbb{K}(j)$$

zu bestimmen. Man wendet zunächst auf $w(x,t)$ den FPS-Algorithmus bzgl. x an und danach auf das Ergebnis wiederum FPS bzgl. t. Man erhält so durch Vertauschung der Summation (mit zusätzlicher Substitution) eine innere Summe, die die Reihendarstellung von $C_i(x)$ darstellt. Sei $w(x,t)$ wie oben gegeben. Wir wollen nun die q-hypergeometrische Darstellung von

$$C_i(x) = \sum_{j=0}^{\infty} c_j^{(i)} P_j^a(x) \quad \text{mit} \quad \frac{c_{j+1}^{(i)}}{c_j^{(i)}} \in \mathbb{F}(q^j)$$

bzgl. der Basis $\mathfrak{B} = \left\{P_j^a(x) \mid j \in \mathbb{N}_0\right\}$ bestimmen. Betrachten wir dazu die folgende Sitzung, bei der wir exemplarisch die beiden Darstellungen

$$\frac{(t^2;q^2)_\infty}{(xt;q)_\infty} = \frac{(t;q)_\infty\,(-t;q)_\infty}{(xt;q)_\infty} = \sum_{n=0}^{\infty} \frac{\mathrm{h}_n(x;q)}{(q;q)_n}\,t^n \qquad \text{(diskrete } q\text{-Hermite I)}$$

$$\frac{1}{(t;q)_\infty}\,{}_1\phi_1\left(\begin{matrix}-x\\0\end{matrix}\,\middle|\,q;q^{\alpha+1}t\right) = \sum_{n=0}^{\infty} \mathrm{L}_n^{(\alpha)}(x;q)\,t^n \qquad (q\text{-Laguerre})$$

herleiten.

Maple-Sitzung 5.9 (Erzeugende Funktionen)

diskrete q-Hermite I

```
> PS1:=convert(qpochhammer(t,q,infinity)/qpochhammer(x*t,q,infinity),
> qFPS,x,expansionpt=1);
```

$$PS1 := \sum_{k=0}^{\infty} \frac{t^k\,qpower(x,1,q,k)}{qpochhammer(q,q,k)}$$

```
> PS2:=convert(qpochhammer(-t,q,infinity),qFPS,t,j);
```

$$PS2 := \sum_{j=0}^{\infty} \frac{q^{1/2\,j(j-1)}\,t^j}{qpochhammer(q,q,j)}$$

als Doppelsumme

```
> S:=Sum(Sum(simplify(op(1,PS1)*subs(j=n-k,op(1,PS2)),power),k=0..n),
> n=0..infinity);
```

$$S := \sum_{n=0}^{\infty}\left(\sum_{k=0}^{n} \frac{t^n\,qpower(x,1,q,k)\,q^{1/2\,(k-n)(-n+k+1)}}{qpochhammer(q,q,k)\,qpochhammer(q,q,n-k)}\right)$$

innere Summe

```
> PS:=Sum(op([1,1],S)/t^n,k=0..infinity);
```

$$PS := \sum_{k=0}^{\infty} \frac{qpower(x,1,q,k)\,q^{1/2\,(k-n)(-n+k+1)}}{qpochhammer(q,q,k)\,qpochhammer(q,q,n-k)}$$

```
> PS:=subs(qpochhammer(q,q,n-k)=qpochhammer(q,q,n)/qpochhammer(q^(-n),q,k)*
> (-1)^k*q^(binomial(k,2)-n*k),PS);
```

$$PS := \sum_{k=0}^{\infty} \frac{qpower(x,1,q,k)\,q^{1/2\,(k-n)(-n+k+1)}\,qpochhammer\left(q^{-n},q,k\right)}{qpochhammer(q,q,k)\,qpochhammer(q,q,n)\,(-1)^k\,q^{\binom{k}{2}-nk}}$$

```
> convert(PS,qphihypergeom);
```

$$\frac{q^{1/2\,n(n-1)}\,qphihypergeom\left([q^{-n},x^{-1}],[0],-xq,q\right)}{qpochhammer(q,q,n)}$$

```
> PS:=convert(DiscreteqHermiteI(n,x,q)/qpochhammer(q,q,n),qFPS,x,
> expansionpt=1);
```

$$PS := \sum_{k=0}^{\infty} \frac{(-1)^k\,q^{1/2\,n^2-1/2\,n+k}\,qpochhammer\left(q^{-n},q,k\right)\,qpower(x,1,q,k)}{qpochhammer(q,q,n)\,qpochhammer(q,q,k)}$$

```
> convert(PS,qphihypergeom);
```

$$\frac{q^{1/2\,n(n-1)}\,qphihypergeom\left([q^{-n},x^{-1}],[0],-xq,q\right)}{qpochhammer(q,q,n)}$$

q-Laguerre

```
> PS1:=convert(qphihypergeom([-x],[0],q^(alpha+1)*t,q),qFPS,x);
```

$$PS1 := \sum_{k=0}^{\infty} \frac{(-1)^k\,q^{k(k+\alpha)}\,t^k\,qphihypergeom\left([0],[0],q^{\alpha+1}t,q\right)x^k}{qpochhammer(q,q,k)\,qpochhammer(q^{\alpha+1}t,q,k)}$$

```
> PS2:=convert(qphihypergeom([0],[0],q^(alpha+1)*t,q)/
> qpochhammer(q^(alpha+1)*t,q,k)/qpochhammer(t,q,infinity),qFPS,t,j);
```

$$PS2 := \sum_{j=0}^{\infty} \frac{qpochhammer\left(q^{\alpha+1+k},q,j\right)\,t^j}{qpochhammer(q,q,j)}$$

als Doppelsumme

```
> S:=Sum(Sum(simplify(op(1,PS1)/qphihypergeom([0],[0],q^(alpha+1)*t,q)*
> *qpochhammer(q^(alpha+1)*t,q,k)*subs(j=n-k,op(1,PS2)),power),k=0..n),
> n=0..infinity);
```

$$S := \sum_{n=0}^{\infty}\left(\sum_{k=0}^{n}\frac{(-1)^k\, q^{k(k+\alpha)}\, t^n x^k\, qpochhammer\left(q^{\alpha+1+k}, q, n-k\right)}{qpochhammer\left(q, q, k\right)\, qpochhammer\left(q, q, n-k\right)}\right)$$

innere Summe

```
> PS:=Sum(op([1,1],S)/t^n,k=0..infinity);
```

$$PS := \sum_{k=0}^{\infty}\frac{(-1)^k\, q^{k(k+\alpha)}\, x^k\, qpochhammer\left(q^{\alpha+1+k}, q, n-k\right)}{qpochhammer\left(q, q, k\right)\, qpochhammer\left(q, q, n-k\right)}$$

```
> PS:=subs(qpochhammer(q,q,n-k)=qpochhammer(q,q,n)/qpochhammer(q^(-n),q,k)*
> (-1)^k*q^(binomial(k,2)-n*k),qpochhammer(q^(alpha+1+k),q,n-k)=
> qpochhammer(q^(alpha+1),q,n)/qpochhammer(q^(alpha+1),q,k),PS);
```

$$PS := \sum_{k=0}^{\infty}\frac{q^{k(k+\alpha)}\, x^k\, qpochhammer\left(q^{\alpha+1}, q, n\right)\qquad qpochhammer\left(q^{-n}, q, k\right)}{qpochhammer\left(q, q, k\right)\, qpochhammer\left(q^{\alpha+1}, q, k\right)\, qpochhammer\left(q, q, n\right)\, q^{\binom{k}{2}-nk}}$$

```
> convert(PS,qphihypergeom);
```

$$\frac{qpochhammer\left(q^{\alpha+1}, q, n\right)\, qphihypergeom\left([q^{-n}], [q^{\alpha}q], -q^{\alpha}qq^{n}x, q\right)}{qpochhammer\left(q, q, n\right)}$$

```
> PS:=convert(qLaguerre(n,alpha,x,q),qFPS,x);
```

$$PS := \sum_{k=0}^{\infty}\frac{q^{1/2\,k(k+1+2\,n+2\,\alpha)}\, qpochhammer\left(q^{-n}, q, k\right)\, qpochhammer\left(q^{\alpha+1}, q, n\right)\, x^k}{qpochhammer\left(q, q, k\right)\, qpochhammer\left(q^{\alpha+1}, q, k\right)\, qpochhammer\left(q, q, n\right)}$$

```
> convert(PS,qphihypergeom);
```

$$\frac{qpochhammer\left(q^{\alpha+1}, q, n\right)\, qphihypergeom\left([q^{-n}], [q^{\alpha}q], -q^{\alpha}qq^{n}x, q\right)}{qpochhammer\left(q, q, n\right)}$$

Bei beiden Beweisen verwenden wir die Identität

$$(q;q)_{n-k} = (-1)^k q^{\binom{k}{2}-nk}\frac{(q;q)_n}{(q^{-n};q)_k},$$

damit die innere Summe endlich wird und die natürlichen Grenzen 0 und n sind.

5.2.4 Verschiedene Identitäten

Wir können weitere einfache Identitäten mit `qFPS` zeigen, wie z.B. die verschiedenen Darstellungen der q-Exponentialfunktionen

$$\begin{aligned}\exp_q(x) = \mathrm{e}_q\left(\frac{x}{1-q}\right) &= \frac{1}{(x;q)_\infty} = {}_1\phi_0\left(\begin{matrix}0\\-\end{matrix}\,\middle|\, q;x\right) = \sum_{j=0}^{\infty}\frac{x^k}{(q;q)_k}\\ &= \exp_q(a)\,{}_1\phi_0\left(\begin{matrix}\frac{a}{x}\\-\end{matrix}\,\middle|\, q;x\right) = \exp_q(a)\,{}_2\phi_1\left(\begin{matrix}\frac{x}{a},0\\\frac{q}{a}\end{matrix}\,\middle|\, q;q\right)\\ &= \exp_q(a)\,{}_1\phi_1\left(\begin{matrix}\frac{a}{x}\\\frac{a}{q}\end{matrix}\,\middle|\, q^{-1};xq^{-1}\right) = \exp_q(a)\,{}_1\phi_0\left(\begin{matrix}\frac{x}{a}\\-\end{matrix}\,\middle|\, q^{-1};aq^{-1}\right)\end{aligned}$$

bzw.

$$\begin{aligned}\mathrm{Exp}_q(x) = \mathrm{E}_q\left(\frac{x}{1-q}\right) &= (-x;q)_\infty = {}_0\phi_0\left(\begin{matrix}-\\-\end{matrix}\,\middle|\, q;-x\right) = \sum_{j=0}^{\infty}\frac{q^{\binom{k}{2}}x^k}{(q;q)_k}\\ &= \mathrm{Exp}_q(a)\,{}_1\phi_1\left(\begin{matrix}\frac{a}{x}\\-a\end{matrix}\,\middle|\, q;-x\right) = \mathrm{Exp}_q(a)\,{}_1\phi_0\left(\begin{matrix}\frac{x}{a}\\-\end{matrix}\,\middle|\, q;-a\right)\\ &= \mathrm{Exp}_q(a)\,{}_1\phi_0\left(\begin{matrix}\frac{a}{x}\\-\end{matrix}\,\middle|\, q^{-1};-xq^{-1}\right) = \mathrm{Exp}_q(a)\,{}_2\phi_1\left(\begin{matrix}\frac{x}{a},0\\-\frac{1}{a}\end{matrix}\,\middle|\, q^{-1};q^{-1}\right)\end{aligned}$$

mit $a \in \mathbb{F}^*$ sowie

$$\exp_q(x)\,\mathrm{Exp}_q(-x) = 1 \quad \text{bzw.} \quad \mathrm{e}_q(x)\,\mathrm{E}_q(-x) = 1$$

und die *q-Additionstheoreme*

$$\sin_q(x)\,\mathrm{Sin}_q(x) + \cos_q(x)\,\mathrm{Cos}_q(x) = 1 \quad \text{bzw.} \quad \mathrm{s}_q(x)\,\mathrm{S}_q(x) + \mathrm{c}_q(x)\,\mathrm{C}_q(x) = 1$$

und

$$\sin_q(x)\,\mathrm{Cos}_q(x) - \mathrm{Sin}_q(x)\cos_q(x) = 0 \quad \text{bzw.} \quad \mathrm{s}_q(x)\,\mathrm{C}_q(x) - \mathrm{S}_q(x)\,\mathrm{c}_q(x) = 0.$$

Maple-Sitzung 5.10 (Verschiedene Identitäten)

Identitäten für q-Exponentialfunktionen

```
> convert(qexp(x,q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{x^k}{qpochhammer(q,q,k)}$$

```
> convert(expq(x/(1-q),q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{x^k}{qpochhammer(q,q,k)}$$

```
> convert(1/qpochhammer(x,q,infinity),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{x^k}{qpochhammer(q,q,k)}$$

```
> convert(qphihypergeom([0],[],x,q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{x^k}{qpochhammer(q,q,k)}$$

```
> convert(convert(qexp(x,q),qFPS,expansionpt=a),qphihypergeom);
```

$$qexp(a,q)\,qphihypergeom\left(\left[\frac{a}{x}\right],[\,],x,q\right)$$

```
> convert(convert(qexp(x,q),qFPS,base=qpochhammer,expansionpt=a),
> qphihypergeom);
```

$$qexp(a,q)\,qphihypergeom\left(\left[\frac{x}{a},0\right],\left[\frac{q}{a}\right],q,q\right)$$

```
> convert(convert(qexp(x,q),qFPS,expansionpt=a),qphihypergeom,var=1/q);
```

$$qexp(a,q)\,qphihypergeom\left(\left[\frac{x}{a}\right],[\,],\frac{a}{q},q^{-1}\right)$$

```
> convert(convert(qexp(x,q),qFPS,base=qpochhammer,expansionpt=a),
> qphihypergeom,var=1/q);
```

$$qexp(a,q)\,qphihypergeom\left(\left[\frac{a}{x}\right],\left[\frac{a}{q}\right],\frac{x}{q},q^{-1}\right)$$

```
> convert(qExp(x,q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{q^{1/2\,k(k-1)}x^k}{qpochhammer(q,q,k)}$$

```
> convert(Expq(x/(1-q),q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{q^{1/2\,k(k-1)}x^k}{qpochhammer(q,q,k)}$$

```
> convert(qpochhammer(-x,q,infinity),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{q^{1/2\,k(k-1)}x^k}{qpochhammer(q,q,k)}$$

```
> convert(qphihypergeom([],[],-x,q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{q^{1/2\,k(k-1)}x^k}{qpochhammer\left(q,q,k\right)}$$

```
> convert(convert(qExp(x,q),qFPS,expansionpt=a),qphihypergeom);
```

$$qExp\left(a,q\right)\ qphihypergeom\left(\left[\frac{a}{x}\right],\left[-a\right],-x,q\right)$$

```
> convert(convert(qExp(x,q),qFPS,base=qpochhammer,expansionpt=a),
> qphihypergeom);
```

$$qExp\left(a,q\right)\ qphihypergeom\left(\left[\frac{x}{a}\right],\left[\,\right],-a,q\right)$$

```
> convert(convert(qExp(x,q),qFPS,expansionpt=a),qphihypergeom,var=1/q);
```

$$qExp\left(a,q\right)\ qphihypergeom\left(\left[\frac{x}{a},0\right],\left[-a^{-1}\right],q^{-1},q^{-1}\right)$$

```
> convert(convert(qExp(x,q),qFPS,base=qpochhammer,expansionpt=a),
> qphihypergeom,var=1/q);
```

$$qExp\left(a,q\right)\ qphihypergeom\left(\left[\frac{a}{x}\right],\left[\,\right],-\frac{x}{q},q^{-1}\right)$$

```
> convert(qexp(x,q)*qExp(-x,q),qFPS);
```

$$1$$

```
> convert(expq(x,q)*Expq(-x,q),qFPS);
```

$$1$$

q-Additionstheoreme

```
> convert(qsin(x,q)*qSin(x,q)+qcos(x,q)*qCos(x,q),qFPS);
```

$$1$$

```
> convert(sinq(x,q)*Sinq(x,q)+cosq(x,q)*Cosq(x,q),qFPS);
```

$$1$$

```
> convert(qsin(x,q)*qCos(x,q)-qSin(x,q)*qcos(x,q),qFPS);
```

$$0$$

```
> convert(sinq(x,q)*Cosq(x,q)-Sinq(x,q)*cosq(x,q),qFPS);
```

$$0$$

Kapitel 6

Das Maple-Package qFPS

In diesem Kapitel werden wir einige Prozeduren aus dem umfangreichen qFPS*-Package vorstellen und zahlreiche Beispiele und Erläuterungen zur Anwendung des Packages liefern.*

6.1 Installation und Einführung

Das Maple-Package qFPS umfasst alle Algorithmen, die in dieser Arbeit behandelt werden. Man lädt qFPS, indem man zunächst die Datei *qFPS.mpl* auf der Festplatte speichert. Nachdem man Maple[1] geöffnet hat, liest man diese Datei mit dem Befehl `read "/path/qFPS.mpl":` auf UNIX-Systemen bzw. `read "C:/path/qFPS.mpl":` auf Windows-Systemen ein, wobei `path` den Pfad bezeichnet, wo die Datei liegt. Danach lädt man das Package mit der Befehlszeile `with(qFPS):` und kann nun auf alle Prozeduren des Packages zugreifen.

6.2 Die q-Funktionen in qFPS

Tabelle 6.1 listet alle von qFPS unterstützten q-Funktionen mit Namen und Argumenten auf. Die verallgemeinerte q-hypergeometrische Funktion

$$ {}_r\phi_s\left(\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} \middle| q; x \right) $$

wird in qFPS aufgerufen mit `qphihypergeom([a1,...,ar],[b1,...,bs],x,q)`. Man beachte, dass die q-Funktionen in qFPS immer mit x und einem darauffolgenden q enden (bis auf q-Potenzen und q-Pochhammersymbole) und dies aus Konsistenzgründen bei ${}_r\phi_s$ beibehalten worden ist. Bei der Maple-Notation der verallgemeinerten q-hypergeometrischen Funktion muss man also aufpassen, dass man nicht wie gewohnt x auf q folgen lässt. Für jede q-Funktion sind jeweils zwei q-Shiftregeln (bzgl. ε_q und $\varepsilon_{q^{-1}}$) und zwei q-Ableitungsregeln (bzgl. D_q und $D_{q^{-1}}$) implementiert. Diese sind durch die Prozeduren `qshift` und `qdiff` (siehe nächster Abschnitt) abrufbar. Ferner sind einige Funktionswerte klassischer q-orthogonaler Polynome an Stellen wie 0 oder 1 verfügbar. Bei den q-orthogonalen Polynomen erhält man durch Angabe spezifischer Parameter **keine** explizite Darstellung des Polynoms. Die Prozedurnamen werden dort lediglich zum q-Shiften, q-Differenzieren, Testen der linearen Abhängigkeit und Bestimmen der Anfangswerte verwendet, also zu Operationen, die für den q-FPS-Algorithmus hinreichend sind. Andere implementierte q-Funktionen sind

$$ [n]_q, [n]_q! \quad \text{und} \quad \begin{bmatrix} n \\ k \end{bmatrix}_q, $$

die mit `qbrackets(n,q)`,`qfactorial(n,q)` und `qbinomial(n,k,q)` aufgerufen werden können. Es besteht ferner die Möglichkeit das qFPS-Package um weitere q-Funktionen zu erweitern.

[1] Das qFPS-Package setzt Maple-Version 11 oder höher voraus.

q-Funktion	Aufruf	q-Funktion	Aufruf
$(x \ominus a)_q^k$	`qpower(x,a,q,k)`	$(x \ominus a)_q^\infty$	`qpower(x,a,q,infinity)`
$(x;q)_k$	`qpochhammer(x,q,k)`	$(x;q)_\infty$	`qpochhammer(x,q,infinity)`
$\exp_q(x)$	`qexp(x,q)`	$e_q(x)$	`expq(x,q)`
$\mathrm{Exp}_q(x)$	`qExp(x,q)`	$E_q(x)$	`Expq(x,q)`
$\sin_q(x)$	`qsin(x,q)`	$s_q(x)$	`sinq(x,q)`
$\mathrm{Sin}_q(x)$	`qSin(x,q)`	$S_q(x)$	`Sinq(x,q)`
$\cos_q(x)$	`qcos(x,q)`	$c_q(x)$	`cosq(x,q)`
$\mathrm{Cos}_q(x)$	`qCos(x,q)`	$C_q(x)$	`Cosq(x,q)`
$U_n^{(a)}(x;q)$	`AlSalamCarlitzI(n,a,x,q)`	$V_n^{(a)}(x;q)$	`AlSalamCarlitzII(n,a,x,q)`
$C_n(x;a;q)$	`qCharlier(n,a,x,q)`	$K_n(x;a;q)$	`AlternativeqCharlier(n,a,x,q)`
$h_n(x;q)$	`DiscreteqHermiteI(n,x,q)`	$\tilde{h}_n(x;q)$	`DiscreteqHermiteII(n,x,q)`
$p_n(x;a,b\|q)$	`LittleqJacobi(n,a,b,x,q)`	$P_n(x;a,b,c;q)$	`BigqJacobi(n,a,b,c,x,q)`
$p_n(x\|q)$	`LittleqLegendre(n,x,q)`	$P_n(x;c;q)$	`BigqLegendre(n,c,x,q)`
$p_n(x;a\|q)$	`LittleqLaguerre(n,a,x,q)`	$P_n(x;a,b;q)$	`BigqLaguerre(n,a,b,x,q)`
$L_n^{(\alpha)}(x;q)$	`qLaguerre(n,alpha,x,q)`	$S_n(x;q)$	`StieltjesWigert(n,x,q)`
$\mathcal{Q}_n(x;a,b;N\|q)$	`qHahn(n,a,b,N,x,q)`	$M_n(x;b,c;q)$	`qMeixner(n,b,c,x,q)`
$K_n(x;p,N;q)$	`qKrawtchouk(n,p,N,x,q)`	$K_n^{\mathrm{aff}}(x;p,N;q)$	`AffineqKrawtchouk(n,p,N,x,q)`
$K_n^{\mathrm{qtm}}(x;p,N;q)$	`QuantumKrawtchouk(n,p,N,x,q)`		

Tabelle 6.1 q-Funktionen und ihre Namen und Argumente in qFPS

Dazu stehen die Prozeduren `addqdiff`, `addqshift` und `addqrec` zur Verfügung, die dem Package q-Ableitungs- und q-Shiftregeln und evtl. q-Rekursionsgleichungen zum Testen der linearen Abhängigkeit (siehe Bemerkung 3.13) hinzufügen. Die Verwendung dieser Prozeduren wird neben vielen weiteren Demonstrationen in dem Maple-Worksheet *qFPS-Demo.mw* illustriert.

6.3 Die Prozeduren in qFPS

Wir geben in diesem Abschnitt zahlreiche Informationen über die verfügbaren Prozeduren im Package qFPS an, wie z.B. Syntax, Eingabe und Ausgabe der Prozedur, optionale Parameter, Erläuterung der Vorgehensweise und Beispiele zur Verwendung. Es werden nicht alle Prozeduren behandelt. Bei vielen Prozeduren gibt es mehrere Varianten. In solchen Fällen wird nur eine Prozedur beschrieben. Die anderen Prozeduren, bei denen die Syntax der Eingabeparameter und die Vorgehensweise identisch bzw. ähnlich sind, werden dann unter dem Punkt *Ähnliche Prozeduren* referenziert. Beispielsweise gibt es meist zu einer Prozedur, die eine q-holonome Rekursionsgleichung bestimmt, eine entsprechende DE-Variante, bei der eine q-holonome Differentialgleichung ausgegeben wird. Als Beispiel sei hier die Prozedur `qHolonomicRE` genannt.

6.3.1 Prozeduren zum q-Differenzieren und q-Shiften

`qdiff` – bestimmt die q-Ableitung eines Ausdrucks

Aufruf

```
qdiff(term,x,q)
qdiff(term,[x1,...,xn],q)
qdiff(term,[x$n],q)
```

Parameter

`term`	algebraischer Ausdruck, auf den man den Hahnoperator anwendet
`x,x1,...,xn`	Variablen, auf denen der Hahnoperator operiert
`q`	Variable, die den Parameter q in der q-Ableitung bezeichnet
`n`	nichtnegative ganze Zahl, die die Ordnung der q-Ableitung angibt

Optionen

`explicit` *boolean* Wenn `explicit=true`, dann wird der Hahnoperator auch auf nicht definierte Funktionen `f(x)` angewendet (resultiert in `(f(x)-f(q*x))/(1-q)/x`, andernfalls wird die Operatorschreibweise mit dem Hahnoperator `Dq` verwendet. Die Voreinstellung ist `explicit=false`.

Erklärung

- Die Prozedur bestimmt für einen algebraischen Ausdruck `term` die zugehörige q-Ableitung. Gibt es keine implementierte Regel für den Ausdruck, dann wird die q-Ableitung in Operatorschreibweise ausgegeben. Die Eingabe `term` kann auch wieder Hahnoperatoren beinhalten.
- Höhere q-Ableitungen werden rekursiv bestimmt. Die Zahl `n` gibt dabei die Ordnung der q-Ableitung an.

Beispiele

Einfache q-Ableitungen

```
> qdiff(qexp(x,q),x,q);
```

$$-\frac{qexp(x,q)}{q-1}$$

```
> qdiff(f(x),x,q);
```

$$Dq_x(f(x))$$

```
> qdiff(f(x),x,q,explicit);
```

$$-\frac{-f(xq)+f(x)}{(q-1)x}$$

Höhere q-Ableitungen

```
> qdiff(qexp(x,q),[x$3],q);
```

$$-\frac{qexp(x,q)}{(q-1)^3}$$

```
> qdiff(qexp(x*y,q),[x,y],q);
```

$$\frac{(-q+1+xyq)\,qexp(xy,q)}{(q-1)^2}$$

```
> qdiff(f(x,y),[x,y],q);
```

$$\left(Dq_{x,y}\right)(f(x,y))$$

```
> qdiff(f(x,y),[x,y],q,explicit);
```

$$\frac{f(x,y)}{(q-1)^2xy}-\frac{f(x,yq)}{(q-1)^2xy}-\frac{f(xq,y)}{(q-1)^2xy}+\frac{f(xq,yq)}{(q-1)^2xy}$$

Algorithmen

Tabelle 2.1 (S. 18), Tabelle 2.2 (S. 19)

`qshift` – bestimmt den q-Shift eines Ausdrucks

Aufruf

```
qshift(term,x,q)
qshift(term,[x1,...,xn],q)
qshift(term,[x$n],q)
```

Parameter

`term`	algebraischer Ausdruck, auf den man den q-Shift anwendet
`x,x1,...,xn`	Variablen, auf denen der q-Shift operiert
`q`	Variable, die den Parameter q im q-Shift bezeichnet
`n`	nichtnegative ganze Zahl, die die Ordnung des q-Shifts angibt

Optionen

`explicit`	*boolean*	Wenn `explicit=true`, dann wird der q-Shift auch auf nicht definierte Funktionen `f(x)` angewendet (resultiert in `f(q*x)`), andernfalls wird die Operatorschreibweise mit dem q-Shiftoperator `Sq` verwendet. Die Voreinstellung ist `explicit=false`.

Erklärung

- Die Prozedur bestimmt für einen algebraischen Ausdruck `term` den zugehörigen q-Shift. Gibt es keine implementierte Regel für den Ausdruck, dann wird der q-Shift in Operatorschreibweise ausgegeben. Die Eingabe `term` kann auch wieder q-Shiftoperatoren beinhalten.
- Höhere q-Shifts werden rekursiv bestimmt. Die Zahl `n` gibt die Ordnung des q-Shifts an.

Beispiele

Einfache q-Shifts

```
> qshift(qexp(x,q),x,q);
```

$$(-x+1)\,qexp(x,q)$$

```
> qshift(f(x),x,q);
```

$$Sq_x(f(x))$$

```
> qshift(f(x),x,q,explicit);
```

$$f(xq)$$

Höhere q-Shifts

```
> qshift(qexp(x,q),[x$3],q);
```

$$-\left(xq^2-1\right)(xq-1)(x-1)\,qexp(x,q)$$

```
> qshift(qexp(x*y,q),[x,y],q);
```

$$(xyq-1)(xy-1)\,qexp(xy,q)$$

```
> qshift(f(x,y),[x,x,y],q);
```

$$\left(Sq_{x,x,y}\right)(f(x,y))$$

```
> qshift(f(x,y),[x,x,y],q,explicit);
```

$$f\left(xq^2,yq\right)$$

Inverse q-Shifts

```
> qshift(qexp(x,q),x,1/q,explicit);
```

$$\frac{q\,qexp(x,q)}{q-x}$$

```
> qshift(f(x),x,1/q,explicit);
```

$$f\left(\frac{x}{q}\right)$$

Algorithmen

Tabelle 2.1 (S. 18), Tabelle 2.2 (S. 19), Tabelle 2.3 (S. 24), Tabelle 2.4 (S. 25)

6.3.2 Prozeduren zur Bestimmung q-holonomer Differential- und Rekursionsgleichungen

qHolonomicRE – bestimmt eine q-holonome Rekursionsgleichung aus den q-Shifts eines Ausdrucks

Aufruf

`qHolonomicRE(term,Fx)`

Parameter

`term` algebraischer Ausdruck

`Fx` Bezeichnung der Funktion, die die Rekursionsgleichung erfüllt

Optionen

`var`	*algebraic*	Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.
`MAXREORDER`	*posint*	Obere Schranke, für die Ordnung der q-holonomen Rekursionsgleichung. Diese Schranke wird benötigt, falls der gegebene Ausdruck `term` nicht q-holonom ist, denn sonst würde die Prozedur nicht terminieren. Wird keine q-holonome Rekursionsgleichung der Ordnung $\leq$ `MAXREORDER` gefunden, so kann man `MAXREORDER` erhöhen und die Prozedur erneut ausführen. Voreingestellt ist der Wert 4.
`minsol`	*boolean*	Der boolesche Parameter `minsol` gibt an, ob die Prozedur lediglich die q-holonome Rekursionsgleichung minimaler Ordnung ausgibt (`true` voreingestellt) oder ob alle gefundenen Rekursionen bis zur Ordnung $\leq$ `MAXREORDER` ausgegeben werden (`false`).
`explicit`	*boolean*	Wenn `explicit=true`, dann treten die q-Shifts in der resultierenden Rekursionsgleichung in der Form `F(x),F(q*x),F(q^2*x),...` (sofern `Fx=F(x)`) auf, andernfalls wird die Operatorschreibweise mit dem q-Shiftoperator `Sq` verwendet. Die Voreinstellung ist `explicit=false`.

Erklärung

- Die Prozedur bestimmt zu einem q-holonomen Ausdruck `term` die zugehörige q-holonome Rekursionsgleichung minimaler Ordnung. Ist `term` nicht q-holonom, so gibt die Prozedur einen Hinweis aus, dass bis zur vorgegebenen Ordung `MAXREORDER` keine q-holonome Rekursionsgleichung gefunden worden ist. Der Algorithmus kann also eine nicht q-holonome Funktion `term` nicht als solche erkennen.
- Die Prozedur verwendet die Prozedur `qshift`. Gibt es keine q-Shiftregel für einen Ausdruck `term`, so kann der Algorithmus auch keine q-holonome Rekursionsgleichung finden.

Beispiele

```
> qHolonomicRE(qexp(x,q),F(x));
```

$$(-1+x)\,F(x)+Sq_x\,(F(x))=0$$

```
> qHolonomicRE(qexp(x,p),F(x),var=p,MAXREORDER=2,minsol=false,explicit);
```

$$[(-1+x)\,F(x)+F(px)=0,-(-1+x)\,(px-2)\,F(x)+F(px)+F\left(p^2x\right)\ =0]$$

Algorithmen

Algorithmus 3.1 (S. 36), Satz 3.11 (S. 34), Tabelle 2.3 (S. 24), Tabelle 2.4 (S. 25),

Algorithmus 5.7 (S. 86)

Ähnliche Prozeduren

`qHolonomicDE`

6.3.3 Prozeduren der q-holonomen Algebra

qSumRE – bestimmt zu zwei q-holonomen Rekursionen die Rekursion der Summe

Aufruf

`qSumRE(RE1,RE2,Fx)`

Parameter

`RE1` erste q-holonome Rekursionsgleichung

`RE2` zweite q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die Rekursion der Summe erfüllt

Optionen

`var`	*algebraic*	Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.
`explicit`	*boolean*	Wenn `explicit=true`, dann treten die q-Shifts in der resultierenden Rekursionsgleichung in der Form `F(x),F(q*x),F(q^2*x),...` (sofern `Fx=F(x)`) auf, andernfalls wird die Operatorschreibweise mit dem q-Shiftoperator `Sq` verwendet. Die Voreinstellung ist `explicit=false`.

Erklärung

- Die Eingaben der Prozedur sind zwei q-holonome Rekursionsgleichungen `RE1` und `RE2` in `Fx`, wobei `Fx` ein Ausdruck der Form `F(x)` darstellt. Erfüllt die q-holonome Funktion $f(x)$ die Rekursion `RE1` und $g(x)$ die Rekursion `RE2`, dann ist das Ergebnis eine q-holonome Rekursionsgleichung für $f(x)+g(x)$.

Beispiele

```
> RE1:=qHolonomicRE(qexp(x,q),F(x));
```

$$RE1 := (-1+x)\,F(x) + Sq_x\,(F(x)) = 0$$

```
> RE2:=qHolonomicRE(qExp(x,q),F(x));
```

$$RE2 := -F(x) + (x+1)\,Sq_x\,(F(x)) = 0$$

```
> qSumRE(RE1,RE2,F(x));
```

$$-q^2\,(-1+x)\,F(x) + \left(-1+q^2x^2-q^2\right) Sq_x\,(F(x)) + (xq+1)\left(Sq_{x,x}\right)(F(x)) = 0$$

```
> qSumRE(RE1,RE2,F(x),explicit);
```

$$-q^2\,(-1+x)\,F(x) + (xq+1)\,F\left(q^2x\right) + \left(-1+q^2x^2-q^2\right) F(xq) = 0$$

Algorithmen

Algorithmus 3.2 (S. 39)

Ähnliche Prozeduren

`qSumDE`

qProductRE – bestimmt zu zwei q-holonomen Rekursionen die Rekursion des Produkts

Aufruf

`qProductRE(RE1,RE2,Fx)`

Parameter

`RE1` erste q-holonome Rekursionsgleichung

`RE2` zweite q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die Rekursion des Produkts erfüllt

Optionen

var	*algebraic*	Der optionale Parameter var steht für den Parameter q im q-Fall. Standardmäßig ist var=q.
explicit	*boolean*	Wenn explicit=true, dann treten die q-Shifts in der resultierenden Rekursionsgleichung in der Form F(x),F(q*x),F(q^2*x),... (sofern Fx=F(x)) auf, andernfalls wird die Operatorschreibweise mit dem q-Shiftoperator Sq verwendet. Die Voreinstellung ist explicit=false.

Erklärung

- Die Eingaben der Prozedur sind zwei q-holonome Rekursionsgleichungen RE1 und RE2 in Fx, wobei Fx ein Ausdruck der Form F(x) darstellt. Erfüllt die q-holonome Funktion $f(x)$ die Rekursion RE1 und $g(x)$ die Rekursion RE2, dann ist das Ergebnis eine q-holonome Rekursionsgleichung für $f(x) \cdot g(x)$.

Beispiele

```
> RE1:=qHolonomicRE(qexp(x,q),F(x));
```

$$RE1 := (-1+x)\,F(x) + Sq_x(F(x)) = 0$$

```
> RE2:=qHolonomicRE(qExp(x,q),F(x));
```

$$RE2 := -F(x) + (x+1)\,Sq_x(F(x)) = 0$$

```
> qProductRE(RE1,RE2,F(x));
```

$$(-1+x)\,F(x) + (x+1)\,Sq_x(F(x)) = 0$$

```
> qProductRE(RE1,RE2,F(x),explicit);
```

$$(-1+x)\,F(x) + (x+1)\,F(xq) = 0$$

Algorithmen

Algorithmus 3.3 (S. 40)

Ähnliche Prozeduren

qProductDE

---

qCompositionRE – bestimmt zu einer q-holonomen Rekursion und zu einer Potenzfunktion die Rekursion der Komposition

Aufruf

```
qCompositionRE(RE,Fx,term)
```

Parameter

RE	q-holonome Rekursionsgleichung
Fx	Bezeichnung der Funktion, die die Rekursion der Komposition erfüllt
term	algebraischer Ausdruck

Optionen

var	*algebraic*	Der optionale Parameter var steht für den Parameter q im q-Fall. Standardmäßig ist var=q.
explicit	*boolean*	Wenn explicit=true, dann treten die q-Shifts in der resultierenden Rekursionsgleichung in der Form F(x),F(q*x),F(q^2*x),... (sofern Fx=F(x)) auf, andernfalls wird die Operatorschreibweise mit dem q-Shiftoperator Sq verwendet. Die Voreinstellung ist explicit=false.

Erklärung

- Die Eingabe der Prozedur ist zum Einen eine q-holonome Rekursionsgleichung `RE` in `Fx`, wobei `Fx` ein Ausdruck der Form `F(x)` darstellt und zum Anderen ein Term $g(x)$, für den $g(qx) = q^b g(x)$ für ein $n \in \mathbb{N}$ gilt. Erfüllt die q-holonome Funktion $f(x)$ die Rekursion `RE`, dann ist das Ergebnis eine q-holonome Rekursionsgleichung für $(f \circ g)(x)$.

Beispiele

```
> RE:=qHolonomicRE(qexp(x,q),F(x));
```

$$RE := (-1+x)\,F(x) + Sq_x(F(x)) = 0$$

```
> term:=a*x^3;
```

$$term := ax^3$$

```
> qCompositionRE(RE,F(x),term);
```

$$\left(-1+ax^3\right)\left(ax^3q-1\right)\left(q^2ax^3-1\right)F(x) + Sq_x(F(x)) = 0$$

```
> qCompositionRE(RE,F(x),term,explicit);
```

$$\left(-1+ax^3\right)\left(ax^3q-1\right)\left(q^2ax^3-1\right)F(x) + F(xq) = 0$$

Algorithmen

Algorithmus 3.4 (S. 41)

Ähnliche Prozeduren

`qCompositionDE`

6.3.4 Konversionsprozeduren

`qREtoqRE` – transformiert q-Rekursionsgleichungen von additiver Form in multiplikative Form und umgekehrt

Aufruf

`qREtoqRE(RE,Ak,Fx)`

Parameter

`RE`	q-holonome Rekursionsgleichung
`Ak`	Bezeichnung der Funktion, die die Eingaberekursion `RE` erfüllt
`Fx`	Bezeichnung der Funktion, die die Ausgaberekursion erfüllt

Optionen

`var`	*algebraic*	Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.
`explicit`	*boolean*	Wenn `explicit=true` und die Eingaberekursion in additiver Form vorliegt, dann treten die q-Shifts in der resultierenden Rekursionsgleichung in der Form `F(x),F(q*x),F(q^2*x),...` (sofern `Fx=F(x)`) auf, andernfalls wird die Operatorschreibweise mit dem q-Shiftoperator `Sq` verwendet. Die Voreinstellung ist `explicit=false`.

Erklärung

- Die Eingabe der Prozedur ist eine q-holonome Rekursionsgleichung `RE` in `Ak`, wobei `Ak` ein Ausdruck der Form `A(k)` darstellt. Die Ausgabe ist eine q-holonome Rekursionsgleichung in `Fx`, wobei `Fx` eine Eingabe gemäß `F(x)` ist.
- Gibt man eine Rekursion in additiver Form ein, so wird eine Rekursion in multiplikativer Form ausgegeben. Umgekehrt wird bei Eingabe einer Rekursion in multiplikativer Form die entsprechende Rekursion in additiver Form bestimmt.

Beispiele

```
> RE:=(q^2*x-1)*Sq[x,x](F(x))+x*q*(q^2*x+q*x-1)*Sq[x](F(x))+
> q^2*x^3*F(x)=0;
```

$$RE := \left(q^2x-1\right)\left(Sq_{x,x}\right)\left(F\left(x\right)\right)+xq\left(q^2x+qx-1\right)Sq_x\left(F\left(x\right)\right)+q^2x^3F\left(x\right)=0$$

```
> RE:=qREtoqRE(RE,F(x),A(k));
```

$$RE := \left(q^2q^k-1\right)A\left(k+2\right)+q^kq\left(q^2q^k+qq^k-1\right)A\left(k+1\right)+q^2\left(q^k\right)^3A\left(k\right)=0$$

```
> qREtoqRE(RE,A(k),F(x));
```

$$\left(q^2x-1\right)\left(Sq_{x,x}\right)\left(F\left(x\right)\right)+xq\left(q^2x+qx-1\right)Sq_x\left(F\left(x\right)\right)+q^2x^3F\left(x\right)=0$$

```
> qREtoqRE(RE,A(k),F(x),explicit);
```

$$\left(q^2x-1\right)F\left(q^2x\right)+xq\left(q^2x+qx-1\right)F\left(qx\right)+q^2x^3F\left(x\right)=0$$

Algorithmen

Substitutionen $x \leftrightarrow q^k$ und $F(x) \leftrightarrow A(k)$ werden durchgeführt

Ähnliche Prozeduren

qREtoqDE, qDEtoqRE

---

qREtoqDE – transformiert q-holonome Rekursionsgleichungen in q-holonome Differentialgleichungen

Aufruf

```
qREtoqDE(RE,Fx)
```

Parameter

RE	q-holonome Rekursionsgleichung
Fx	Bezeichnung der Funktion, die die Eingaberekursion RE erfüllt

Optionen

var	*algebraic*	Der optionale Parameter var steht für den Parameter q im q-Fall. Standardmäßig ist var=q.

Erklärung

- Die Eingabe der Prozedur ist eine q-holonome Rekursionsgleichung RE in Fx, wobei Fx ein Ausdruck der Form F(x) darstellt.
- Die Prozedur konvertiert die gegebene q-holonome Rekursionsgleichung in eine q-holonome Differentialgleichung.

Beispiele

```
> RE:=Sq[x,x](F(x))-(q+1)*Sq[x](F(x))+q*(1+(q-1)^2*x^2)*F(x)=0;
```

$$RE := \left(Sq_{x,x}\right)\left(F\left(x\right)\right)-\left(q+1\right)Sq_x\left(F\left(x\right)\right)+q\left(1+\left(q-1\right)^2x^2\right)F\left(x\right)=0$$

```
> RE:=qREtoqDE(RE,F(x));
```

$$RE := \left(Dq_{x,x}\right)\left(F\left(x\right)\right)+F\left(x\right)=0$$

Algorithmen

Satz 1.11 (S. 6)

Ähnliche Prozeduren

qDEtoqRE

qREtoRE – transformiert eine q-holonome Rekursionsgleichung einer q-Potenzreihe in eine q-holonome Rekursionsgleichung für deren Koeffizienten

Aufruf

```
qREtoRE(RE,Fx,Ak)
```

Parameter

RE	q-holonome Rekursionsgleichung
Fx	Bezeichnung der Funktion, die die Eingaberekursion RE erfüllt
Ak	Bezeichnung der Funktion, die die Ausgaberekursion erfüllt

Optionen

var	*algebraic*	Der optionale Parameter var steht für den Parameter q im q-Fall. Standardmäßig ist var=q.
base		Der optionale Parameter base gibt die Basis an, bzgl. derer eine q-holonome Rekursionsgleichung gefunden werden soll. Es kann entweder die q-Potenz- (qpower) oder die q-Pochhammer-Basis (qpochhammer) gewählt werden. Voreingestellt ist base=qpower.
expansionpt	*algebraic*	Mit dieser Option kann man den Entwicklungspunkt angeben. Standardmäßig ist expansionpt=0.

Erklärung

- Die Eingabe der Prozedur ist eine q-holonome Rekursionsgleichung RE in Fx, wobei Fx ein Ausdruck der Form F(x) darstellt.
- Die Prozedur konvertiert die gegebene q-holonome Rekursionsgleichung einer q-Potenzreihe $\sum_{j=0}^{\infty} c_j P_j^a(x)$ in eine q-holonome Rekursion für c_j, wobei $P_j^a(x)$, je nach Wahl von base, entweder aus $\mathfrak{B}^a$ oder aus $\mathfrak{B}_a$ stammt. Der Parameter a bezeichne hierbei den Entwicklungspunkt expansionpt.

Beispiele

```
> RE:=Sq[x](F(x))-x*F(x)=0;
```

$$RE := Sq_x(F(x)) - xF(x) = 0$$

```
> qREtoRE(RE,F(x),A(k));
```

$$q^k qA(k+1) - A(k) = 0$$

```
> qREtoRE(RE,F(x),A(k),expansionpt=a);
```

$$-A(k) - q^k q(-1+a)A(k+1) + q^k qa\left(q^k q^2 - 1\right)A(k+2) = 0$$

```
> qREtoRE(RE,F(x),A(k),base=qpochhammer,expansionpt=a);
```

$$q^3 A(k) + q\left(q^3\left(q^k\right)^2 + aq^k q^2 - qa - a\right)A(k+1) - a^2\left(q^k q^2 - 1\right)A(k+2) = 0$$

Algorithmen

Algorithmus 5.1 (S. 78), Algorithmus 5.2 (S. 79)

Ähnliche Prozeduren

qDEtoRE, REtoqRE, REtoqDE

6.3.5 Prozeduren zur Lösung q-holonomer Rekursionsgleichungen

qPolUpperBound – bestimmt eine Gradschranke aller polynomialen Lösungen einer q-holonomen Rekursionsgleichung

Aufruf

`qPolUpperBound(RE,Fx)`

Parameter

`RE` q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die eingegebene Rekursion erfüllt

Optionen

`var` *algebraic* Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Rekursionsgleichung eine Gradschranke aller polynomialen Lösungen.

Beispiele

```
> RE:=Sq[x,x](F(x))-(q+1)*Sq[x](F(x))+q*F(x)=0;
```

$$RE := \left(Sq_{x,x}\right)(F(x)) - (q+1)\,Sq_x(F(x)) + qF(x) = 0$$

```
> qPolUpperBound(RE,F(x));
```

$$1$$

Algorithmen

Algorithmus 4.2 (S. 51)

`qPolynomialSolveRE` – bestimmt alle polynomialen Lösungen einer q-holonomen Rekursionsgleichung

Aufruf

`qPolynomialSolveRE(RE,Fx)`

Parameter

`RE` q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die eingegebene Rekursion erfüllt

Optionen

`var` *algebraic* Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.

`output` Werte für diesen Parameter können `gensol` und `onesol` sein. Bei `gensol` wird eine allgemeine (parameterabhängige) Lösung ausgegeben, wohingegen bei `onesol` eine Lösung ausgegeben wird. Die Voreinstellung ist `output=gensol`.

`genvar` *symbol* Mit der Option `genvar` kann man die Variable angeben, die für die Parameter bei `gensol` verwendet werden soll.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Rekursionsgleichung alle polynomialen Lösungen.

Beispiele

```
> RE:=Sq[x,x](F(x))-(q+1)*Sq[x](F(x))+q*F(x)=0;
```

$$RE := \left(Sq_{x,x}\right)(F(x)) - (q+1)\,Sq_x(F(x)) + qF(x) = 0$$

```
> qPolynomialSolveRE(RE,F(x));
```

$$a_0 + a_1 x$$

```
> qPolynomialSolveRE(RE,F(x),output=onesol);
```

$$1+x$$

Algorithmen

Algorithmus 4.3 (S. 52)

---

qUniversalDenominator – bestimmt eine Nennerschranke aller rationalen Lösungen einer q-holonomen Rekursionsgleichung

Aufruf

```
qUniversalDenominator(RE,Fx)
```

Parameter

`RE` q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die eingegebene Rekursion erfüllt

Optionen

`var` *algebraic* Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Rekursionsgleichung eine Nennerschranke aller rationalen Lösungen.

Beispiele

```
> RE:=q^2*Sq[x,x](F(x))-(q+1)*q*Sq[x](F(x))+q*F(x)=0;
```

$$RE := q^2\left(Sq_{x,x}\right)(F(x)) - (q+1)\,qSq_x(F(x)) + qF(x) = 0$$

```
> qUniversalDenominator(RE,F(x));
```

$$x$$

Algorithmen

Algorithmus 4.5 (S. 57)

---

qRationalSolveRE – bestimmt alle rationalen Lösungen einer q-holonomen Rekursionsgleichung

Aufruf

```
qRationalSolveRE(RE,Fx)
```

Parameter

`RE` q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die eingegebene Rekursion erfüllt

Optionen

`var` *algebraic* Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.

`output` Werte für diesen Parameter können `gensol` und `onesol` sein. Bei `gensol` wird eine allgemeine (parameterabhängige) Lösung ausgegeben, wohingegen bei `onesol` eine Lösung ausgegeben wird. Die Voreinstellung ist `output=gensol`.

`genvar` *symbol* Mit der Option `genvar` kann man die Variable angeben, die für die Parameter bei `gensol` verwendet werden soll.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Rekursionsgleichung alle rationalen Lösungen.

Beispiele

```
> RE:=q^2*Sq[x,x](F(x))-(q+1)*q*Sq[x](F(x))+q*F(x)=0;
```

$$RE := q^2\left(Sq_{x,x}\right)(F(x)) - (q+1)\,qSq_x(F(x)) + qF(x) = 0$$

```
> qRationalSolveRE(RE,F(x));
```

$$\frac{a_0 + a_1 x}{x}$$

```
> qRationalSolveRE(RE,F(x),output=onesol);
```

$$\frac{1+x}{x}$$

Algorithmen

Algorithmus 4.4 (S. 55), Algorithmus 4.6 (S. 59)

---

qHypergeomTypeSolveRE – bestimmt alle m-q-hypergeometrischen Lösungen einer q-holonomen Rekursionsgleichung vom q-hypergeometrischen Typ mit Symmetriezahl m ($m \in \mathbb{N}$)

Aufruf

`qHypergeomTypeSolveRE(RE,Fx)`

Parameter

`RE`	q-holonome Rekursionsgleichung
`Fx`	Bezeichnung der Funktion, die die eingegebene Rekursion erfüllt

Optionen

`var`	*algebraic*	Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.
`certificate`	*boolean*	Wenn `certificate=true` ist, wird lediglich das zu der m-q-hypergeometrischen Lösung zugehörige q-Zertifikat ausgegeben. Voreingestellt ist `certificate=false`. Wird die Rekursionsgleichung in multiplikativer Form eingegeben, so wird immer (unabhängig von der Einstellung von `certificate`) das q-Zertifikat ausgegeben.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Rekursionsgleichung vom q-hypergeometrischen Typ mit Symmetriezahl m, alle m-q-hypergeometrischen Lösungen.

Beispiele

```
> RE:=Sq[x,x](F(x))+(1-q)*x*F(x)=0;
```

$$RE := \left(Sq_{x,x}\right)(F(x)) + (1-q)\,xF(x) = 0$$

```
> qHypergeomTypeSolveRE(RE,F(x));
```

$$x(-1+q)$$

```
> qHypergeomTypeSolveRE(RE,F(x),certificate);
```

$$x(-1+q)$$

```
> RE:=qREtoqRE(RE,F(x),A(k));
```

$$RE := -q^k(-1+q)\,A(k) + A(k+2) = 0$$

```
> qHypergeomTypeSolveRE(RE,A(k));
```

$$\left[q^{2\binom{k}{2}}(-1+q)^k A(0),\, q^k q^{2\binom{k}{2}}(-1+q)^k A(1)\right]$$

```
> qHypergeomTypeSolveRE(RE,A(k),certificate);
```

$$q^k(-1+q)$$

Algorithmen

Satz 4.35 (S. 62)

---

qHypergeomSolveRE – bestimmt alle q-hypergeometrischen Lösungen einer q-holonomen Rekursionsgleichung

Aufruf

```
qHypergeomSolveRE(RE,Fx)
```

Parameter

`RE` q-holonome Rekursionsgleichung

`Fx` Bezeichnung der Funktion, die die eingegebene Rekursion erfüllt

Optionen

`var`	*algebraic*	Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.
`certificate`	*boolean*	Wenn `certificate=true` ist, wird lediglich das zu der q-hypergeometrischen Lösung zugehörige q-Zertifikat ausgegeben. Voreingestellt ist `certificate=false`. Wird die Rekursionsgleichung in multiplikativer Form eingegeben, so wird immer (unabhängig von der Einstellung von `certificate`) das q-Zertifikat ausgegeben.
`output`		Werte für diesen Parameter können `gensol`, `onesol` und `basis` sein. Bei `gensol` wird eine allgemeine (parameterabhängige) Lösung, bei `onesol` eine Lösung und bei `basis` eine Basis q-hypergeometrischer Lösungen ausgegeben. Die Einstellung ist nur wirksam bei Rekursionen in additiver Form. Die Voreinstellung ist `output=gensol`.
`genvar`	*symbol*	Mit der Option `genvar` kann man die Variable angeben, die für die Parameter bei `gensol` verwendet werden soll.
`method`		Mit `method` kann man das zugrundeliegende Verfahren festlegen. Wählbar sind `qPetkovsek`, `modqPetkovsek` und `qVanHoeij`. Standardmäßig ist `method=modqPetkovsek`.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Rekursionsgleichung, alle q-hypergeometrischen Lösungen.

Beispiele

```
> RE:=(q^2*x-1)*Sq[x,x](F(x))+x*q*(q^2*x+q*x-1)*Sq[x](F(x))+
> q^2*x^3*F(x)=0;
```

$$RE := \left(q^2x-1\right)\left(Sq_{x,x}\right)(F(x)) + xq\left(q^2x+qx-1\right)Sq_x(F(x)) + q^2x^3F(x) = 0$$

```
> qHypergeomSolveRE(RE,F(x));
```

$$[-x, -\frac{x^2q}{qx-1}]$$

```
> qHypergeomSolveRE(RE,F(x),certificate);
```

$$[-x, -\frac{x^2q}{qx-1}]$$

```
> RE:=qREtoqRE(RE,F(x),A(k));
```

$$RE := \left(q^2 q^k - 1\right) A(k+2) + q^k q \left(q^2 q^k + q q^k - 1\right) A(k+1) + q^2 \left(q^k\right)^3 A(k) = 0$$

```
> qHypergeomSolveRE(RE,A(k));
```

$$a_0 (-1)^k q^{\frac{1}{2}k(k-1)} + a_1 \frac{q^{k^2}}{qpochhammer(q, q, k)}$$

Algorithmen

Algorithmus 4.7 (S. 64), Algorithmus 4.8 (S. 65), Algorithmus 4.9 (S. 67)

6.3.6 Prozeduren für q-hypergeometrische Potenzreihen

`convert/qFPS` – bestimmt zu einer q-holonomen Funktion die entsprechende q-hypergeometrische Potenzreihe, sofern eine existiert

Aufruf

```
convert(term,qFPS)
`convert/qFPS`(term)
```

Parameter

`term` eine q-holonome Funktion

Optionen

`var`	*algebraic*	Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.
`base`		Der optionale Parameter `base` gibt die Basis an, bzgl. derer eine q-hypergeometrische Potenzreihe gefunden werden soll. Es kann entweder die q-Potenz- (`qpower`) oder die q-Pochhammer-Basis (`qpochhammer`) gewählt werden. Voreingestellt ist `base=qpower`.
`expansionpt`	*algebraic*	Mit dieser Option kann man den Entwicklungspunkt angeben. Standardmäßig ist `expansionpt=0`.
`termwise`	*boolean*	Der optionale Parameter `termwise` gibt an, ob die Eingabe zunächst ausmultipliziert und dann die Potenzreihenentwicklung faktorweise bzw. summandenweise bestimmt wird (`termwise=true`) oder ob die Eingabe als Ganzes betrachtet und der Algorithmus einmal angewendet wird (`termwise=false`). Voreingestellt ist `termwise=false`. Bei `termwise=true` werden am Ende die Teilergebnisse zu einer Reihe zusammengefasst, soweit dies möglich ist.
`initialcheck`	*boolean*	Der optionale Parameter `initialcheck` gibt an, ob die berechneten Anfangswerte der Koeffizienten überprüft werden oder nicht. Es kann sein, dass höhere q-Ableitungen (beispielsweise bei `qphihypergeom`) nicht an bestimmten Stellen ausgewertet und somit auch nicht mit den q-hypergeometrischen Lösungen verglichen werden können. Aus diesem Grund kann man mittels `initialcheck=false` diesen Test unterbinden und erhält u.U. eine Lösung, die allerdings nicht ausreichend verifiziert worden ist oder eine Lösung, bei der noch unbestimmte Konstanten auftreten. Standardmäßig ist `initialcheck=true`.
`MAXREORDER`	*posint*	Obere Schranke, für die Ordnung der q-holonomen Rekursionsgleichung bei Algorithmus 3.1. Voreingestellt ist der Wert 4.

Erklärung

- Die Prozedur bestimmt zu einer q-holonomen Funktion eine Darstellung als Linearkombination q-hypergeometrischer Potenzreihen, sofern eine existiert.
- Falls mehrere Unbekannte in der Eingabe `term` auftreten, so muss man die Variable, nach der man die Potenzreihe entwickeln möchte, zusätzlich angeben. Das macht man, indem man den Argumenten die Bezeichnung der Variablen als weiteren optionalen Parameter hinzufügt.
- Man kann den Summationsindex wählen, indem man zunächst die Variable, bzgl. derer man die Potenzreihe entwickelt, und danach als weiteres Argument die Variable für den Summationsindex angibt.

Beispiele

```
> convert(qexp(x,q),qFPS);
```

$$\sum_{k=0}^{\infty} \frac{x^k}{qpochhammer\left(q,q,k\right)}$$

```
> convert(qexp(x,q),qFPS,base=qpochhammer,expansionpt=a,var=1/q);
```

$$\sum_{k=0}^{\infty} \frac{qexp\left(a,q\right) q^{-k} qpower\left(a,x,q^{-1},k\right)}{qpochhammer\quad\left(q^{-1},q^{-1},k\right)}$$

```
> convert(qexp(x,q),qFPS,base=qpochhammer,expansionpt=a);
```

$$\sum_{k=0}^{\infty} qexp\left(a,q\right) q^{k} a^{-k} qpower\left(a,x,q,k\right) \left(qpochhammer\left(\frac{q}{a},q,k\right)\right)^{-1} \left(qpochhammer\left(q,q,k\right)\right)^{-1}$$

```
> convert(qsin(x,q),qFPS,expansionpt=a);
```

$$\sum_{k=0}^{\infty} \frac{\left(-1\right)^{k} qsin\left(a,q\right) qpower\left(x,a,q,2\,k\right)}{qpochhammer\left(q,q,2\,k\right)} + \sum_{k=0}^{\infty} \frac{\left(-1\right)^{k} qcos\left(a,q\right) qpower\left(x,a,q,2\,k+1\right)}{qpochhammer\left(q,q,2\,k+1\right)}$$

```
> convert(qLaguerre(n,alpha,x,q),qFPS,x,j);
```

$$\sum_{j=0}^{\infty} \frac{q^{1/2\,j\left(j+2\,n+2\,\alpha+1\right)} qpochhammer\left(q^{-n},q,j\right)\ qpochhammer\left(q^{\alpha+1},q,n\right) x^{j}}{qpochhammer\left(q,q,j\right) qpochhammer\left(q^{\alpha+1},q,j\right) qpochhammer\left(q,q,n\right)}$$

```
> convert(qCharlier(n,a,x,q),qFPS,x,base=qpochhammer,expansionpt=1);
```

$$\sum_{k=0}^{\infty} \frac{\left(-1\right)^{k} q^{k\left(n+1\right)} qpochhammer\left(q^{-n},q,k\right) a^{-k} qpochhammer\left(x,q,k\right)}{qpochhammer\left(q,q,k\right)}$$

Algorithmen

Algorithmus 5.8 (S. 88)

(Algorithmus 3.1 (S. 36), Algorithmus 5.1 (S. 78), Algorithmus 4.9 (S. 67))

`'convert/qphihypergeom'` – bestimmt zu einer q-hypergeometrischen Potenzreihe die Darstellung als verallgemeinerte q-hypergeometrische Funktion ${}_r\phi_s(x)$

Aufruf

`convert(term,qphihypergeom)`

`'convert/qphihypergeom'(term)`

Parameter

`term` eine q-hypergeometrische Potenzreihe

Optionen

`var` *algebraic* Der optionale Parameter `var` steht für den Parameter q im q-Fall. Standardmäßig ist `var=q`.

Erklärung

- Die Prozedur bestimmt zu einer q-hypergeometrischen Potenzreihe die entsprechende Darstellung als verallgemeinerte q-hypergeometrische Funktion.

Beispiele

```
> PS:=convert(qLaguerre(n,alpha,x,q),qFPS,x);
```

$$PS := \sum_{k=0}^{\infty} \frac{q^{1/2\,k(k+2\,n+2\,\alpha+1)}\,qpochhammer\left(q^{-n},q,k\right)\,qpochhammer\left(q^{\alpha+1},q,n\right)\,x^k}{qpochhammer\left(q,q,k\right)\,qpochhammer\left(q^{\alpha+1},q,k\right)\,qpochhammer\left(q,q,n\right)}$$

```
> convert(PS,qphihypergeom);
```

$$\frac{qpochhammer\left(q^{\alpha+1},q,n\right)\,qphihypergeom\left(\left[q^{-n}\right],\left[qq^{\alpha}\right],-qxq^{n}q^{\alpha},q\right)}{qpochhammer\left(q,q,n\right)}$$

```
> PS:=convert(qphihypergeom([a,b],[c],x,1/q),qFPS,x);
```

$$PS := \sum_{k=0}^{\infty} \frac{a^k\,qpochhammer\left(a^{-1},q,k\right)\,b^k\,qpochhammer\left(b^{-1},q,k\right)\,q^k c^{-k} x^k}{qpochhammer\left(c^{-1},q,k\right)\,qpochhammer\left(q,q,k\right)}$$

```
> convert(PS,qphihypergeom,var=1/q);
```

$$qphihypergeom\left(\left[a,b\right],\left[c\right],x,q^{-1}\right)$$

Algorithmen

Algorithmus 5.9 (S. 91)

6.3.7 Weitere Informationen

Um genauere Informationen zum Ablauf der Algorithmen zu erhalten, verwendet man den Befehl `infolevel`. Wir betrachten ein Beispiel anhand der Prozedur `'convert/qFPS'`.

Maple-Sitzung 6.1 (Interne Informationen zum Programmablauf)

```
> infolevel[qFPS]:=4:
> convert(1/x^3*qsin(x^3,q),qFPS);
convert/qFPS: divide by 1/x^3
```

$$qsin\left(x^3,q\right)$$

```
convert/qFPS: substitute x = x^(1/3)
```

$$qsin\left(x,q\right)$$

```
convert/qFPS: q-holonomic recurrence equation of order 2
```

$$q\left(1+x^2\right)F\left(x\right)+\left(-1-q\right)Sq_x\left(F\left(x\right)\right)+\left(Sq_{x,x}\right)\left(F\left(x\right)\right)=0$$

```
convert/qFPS: q-holonomic recurrence equation for series coefficients w.r.t. qpower of order 2
```

$$A\left(k\right)+\left(q^kq^2-1\right)\left(qq^k-1\right)A\left(k+2\right)=0$$

```
convert/qFPS: solution of recurrence equation for coefficients w.r.t. qpower
```

$$[0,\frac{\left(-1\right)^k x^{2\,k+1}}{qpochhammer\left(q,q,2\,k+1\right)}]$$

```
convert/qFPS: resubstitute x = x^3
```

$$\sum_{k=0}^{\infty}\frac{\left(-1\right)^k\left(x^3\right)^{2\,k+1}}{qpochhammer\left(q,q,2\,k+1\right)}$$

```
convert/qFPS: multiply by 1/x^3
```

$$\sum_{k=0}^{\infty}\frac{\left(-1\right)^k\left(x^3\right)^{2\,k}}{qpochhammer\left(q,q,2\,k+1\right)}$$

$$\sum_{k=0}^{\infty}\frac{\left(-1\right)^k x^{6k}}{qpochhammer\left(q,q,2\,k+1\right)}$$

Erhöht man die Zahl `infolevel[qFPS]`, so erhält man weitere Informationen, wie z.B. Informationen über die Ermittlung der Anfangswerte. Man kann `infolevel` auch in Verbindung mit `qHolonomicRE` und `qHypergeomSolveRE` an Stelle von `qFPS` verwenden.

In diesem Kapitel wurden nicht alle Prozeduren von `qFPS` vorgestellt. Für einen Überblick über weitere Funktionalitäten und Beispiele dient das zur Dissertation beiliegende Maple-Worksheet *qFPS-Demo.mw*. Die Beispiele, die in diesem Kapitel zur Erläuterung der Funktionsweise der Prozeduren verwendet worden sind, sind in *Dissertation-qFPS-Beispiele.mw* und die Maple-Sitzungen in *Dissertation-Sitzungen.mw* zu finden. Ferner kann man unter der Datei *Dissertation-Beispiele.mw* alle in der Dissertation gezeigten Beispiele, sowie einige der Tabellen aufrufen. Für die Anwendung der Algorithmen bei den klassischen q-orthogonalen Polynomen siehe *qFPS-q-Orthogonale Polynome-Demo.mw*.

Ausblick

In dieser Dissertation haben wir u.a. das q-Analogon des FPS-Algorithmus kennen gelernt. Wir sind dabei von einer q-holonomen Funktion ausgegangen und haben zu dieser Funktion die dazugehörige q-hypergeometrische Potenzreihe bestimmt. In [Roa96] wurde gezeigt, wie man im Nicht-q-Fall den umgekehrten Weg gehen kann. Beim Roach-Algorithmus wird eine hypergeometrische Reihe vorgegeben und es wird versucht, diese Reihe als Summe spezieller Funktionen auszudrücken. Man kann so z.B. die Identität

$$ {}_2F_1\left(\begin{matrix} -\frac{3}{2}, -\frac{1}{2} \\ \frac{1}{2} \end{matrix} \,\middle|\, x \right) = \frac{2+x}{2}\sqrt{1-x} + \frac{3\sqrt{x}}{2}\arcsin(\sqrt{x}) $$

herleiten, indem man, ausgehend von der bekannten Gleichung

$$ \sqrt{x}\,{}_2F_1\left(\begin{matrix} \frac{1}{2}, \frac{1}{2} \\ \frac{3}{2} \end{matrix} \,\middle|\, x \right) = \arcsin(\sqrt{x}), $$

mehrfach Operationen anwendet, die einen oberen bzw. unteren Parameter um 1 erhöhen oder erniedrigen. Dieser Algorithmus existiert für den q-Fall noch nicht. Allerdings haben wir bereits in der vorliegenden Dissertation Vorarbeit für einen q-Roach-Algorithmus geleistet, indem wir die dafür notwendigen Operatoren A_i,B_j (S. 32) und deren Inversen A_i^{-1} und B_j^{-1} (S. 35), sowie die q-Shift- bzw. q-Nachbarschaftsrelationen (S. 32 bzw. S. 35) hergeleitet haben. Die Formulierung eines vollständigen q-Roach-Algorithmus steht noch aus und würde die Algorithmen dieser Dissertation komplettieren.

Unsere vorgestellten Algorithmen für q-holonome Funktionen und q-hypergeometrische Potenzreihen basieren auf einem linearen q-Gitter. Alle Algorithmen sind übertragbar auf ein *quadratisches q-Gitter* ([Sus03]) mit

$$ x(s) = c_1 q^{-s} + c_2 q^s + c_3. $$

Die mathematischen Objekte, die wir dann behandeln, sind keine q-Differential- bzw. q-Shiftoperatoren D_q bzw. ε_q, klassische q-orthogonale Polynome und q-holonome Rekursionsgleichungen mehr, sondern Operatoren („companion operators“) $\mathbb{D}_x$ (als Spezialfall der *Askey-Wilson-Operator* $\frac{\delta}{\delta x}$) und $\mathbb{S}_x$, stetige („continuous“) klassische q-orthogonale Polynome und q-holonome Dividierte-Differenzengleichungen ([Fou06]). Die Algorithmen dieser Dissertation können dahingehend modifiziert werden. Die Theorie zur Lösung q-holonomer Dividierte-Differenzengleichungen wird in [FKKM09] behandelt. Mit [IS03] existieren q-Taylorsätze auf einem quadratischen q-Gitter bzgl. der Basen mit den Polynomen

- $\psi_j^a(x) = \left(ae^{i\partial}; q\right)_j \left(ae^{-i\partial}; q\right)_j$
- $\phi_j(x) = q^{\frac{j(j/2-1)}{2}} \left(1+e^{2i\partial}\right) \left(-q^{2-j}e^{2i\partial}; q^2\right)_{j-1} e^{-ij\partial}$
- $\varphi_j(x) = \left(q^{\frac{1}{4}}e^{i\partial}; q^{\frac{1}{2}}\right)_j \left(q^{\frac{1}{4}}e^{-i\partial}; q^{\frac{1}{2}}\right)_j$

in $x = cos(\partial)$ mit entsprechend kompatiblen Operatoren. Die stetigen q-orthogonalen Polynome sind allesamt darstellbar als q-hypergeometrische Potenzreihe bzgl. einer der obigen Polynombasen. Ein FPS-Algorithmus auf einem quadratischen q-Gitter würde auf natürliche Weise folgen.

Literaturverzeichnis

[AAR01] Andrews, George E. ; Askey, Richard ; Roy, Ranjan: *Special Functions*. Cambridge : Cambridge University Press, 2001 10

[ABP95] Abramov, Sergei A. ; Bronstein, Manuel ; Petkovšek, Marko: On polynomial solutions of linear operator equations. In: *ISSAC 1995 Proceedings*, 1995, S. 290–296 53

[Abr95] Abramov, Sergei: Rational solutions to linear difference and q-difference equations with polynomial coefficients. In: *Programmirovanie* (1995), Nr. 6, S. 3–11 2, 53, 57, 59

[APP98] Abramov, Sergei ; Paule, Peter ; Petkovšek, Marko: q-Hypergeometric solutions of q-difference equations. In: *Discrete Math.* 180 (1998), S. 3–22 1, 2, 50, 64, 65

[APR00] Abramov, Sergei ; Petkovšek, Marko ; Ryabenko, Anna: Special formal series solutions of linear operator equations. In: *Discrete Math.* (2000), S. 3–25 2, 71, 75

[BK99] Böing, Harald ; Koepf, Wolfram: Algorithms for q-hypergeometric summation in computer algebra. In: *J. Symbolic Comput.* 28 (1999), S. 777–799 1, 2, 68

[Böi98] Böing, Harald: *Theorie und Anwendungen zur q-hypergeometrischen Summation*, Freie Universität Berlin, Diplomarbeit, 1998 50, 93

[CH05] Cluzeau, Thomas ; Hoeij, Mark V.: Computing hypergeometric solutions of linear recurrence equations. In: *Appl. Algebra Engrg. Comm. Comput.* 17 (2005), S. 83–115 2, 47, 49, 65

[CO04] Creutzig, Christopher ; Oevel, Walter: *MuPAD Tutorial*. Springer, 2004 1

[FKKM09] Foupouagnigni, Mama ; Koepf, Wolfram ; Kenfack Nangho, Maurice ; Mboutngam, Salifou: *On solutions of holonomic divided-difference equations on nonuniform lattices*. 2009. – noch nicht veröffentlicht 119

[Fou06] Foupouagnigni, Mama: *On difference and differential equations for modifications of classical orthogonal polynomials*. Habilitationsschrift, Fachbereich Mathematik der Universität Kassel, 2006 119

[Gas96] Gasper, George: Elementary derivations of summations and transformation formulas for q-series. In: *ArXiv Mathematics e-prints* (1996) 93

[GLM08] Geddes, Keith ; Labahn, George ; Monagan, Michael: *Maple 12 Advanced Programming Guide*. Maplesoft, 2008 1

[GR90] Gasper, George ; Rahman, Mizan: *Encyclopedia of Mathematics and its Applications*. Bd. 35: *Basic Hypergeometric Series*. Cambridge : Cambridge University Press, 1990 11, 12, 93

[Han75] Hansen, Eldon R.: *Table of Series and Products*. Prentice Hall, 1975 2

[Hoe98a] Hoeij, Mark V.: Finite singularities and hypergeometric solutions of linear recurrence equations. In: *J. Pure Appl. Algebra* 139 (1998), S. 109–131 2, 65

[Hoe98b] Hoeij, Mark V.: Rational solutions of linear difference equations. In: *ISSAC 1998 Proceedings*, 1998, S. 120–123 59

[Hor08] Horn, Peter: *Faktorisierung in Schief-Polynomringen*, Universität Kassel, Diss., 2008. http://nbn-resolving.de/urn/resolver.pl?urn=urn:nbn:de:hebis:34-2009030226513. - URN urn:nbn:de:hebis:34-2009030226513 2, 44, 46, 65, 66, 70

[IS03] Ismail, Mourad E. ; Stanton, Dennis: Applications of q-Taylor theorems. In: *J. Comput. Appl. Math.* (2003), Nr. 153, S. 259–272 119

[KC02] Kac, Victor ; Cheung, Pokman: *Quantum Calculus*. New York : Springer, 2002 2, 11, 12, 87

[KK09] Kauers, Manuel ; Koutschan, Christoph: A Mathematica package for q-holonomic sequences and power series. In: *Ramanujan J.* 19 (2009), S. 137–150 1, 42

[Koe92] Koepf, Wolfram: Power series in computer algebra. In: *J. Symbolic Comput.* 13 (1992), S. 581–603 2, 36, 88

[Koe98] Koepf, Wolfram: *Hypergeometric Summation, an Algorithmic Approach to Summation and Special Function Identities*. Braunschweig-Wiesbaden : Vieweg, 1998 (Advanced Lectures in Mathematics) 54

[Koe03] Koepf, Wolfram: Power series, Bieberbach conjecture and the de Branges and Weinstein functions. In: *ISSAC 2003 Proceedings*, ACM, 2003, S. 169–175 96

[Koe06] Koepf, Wolfram: *Computeralgebra - Eine algorithmisch orientierte Einführung*. Berlin : Springer, 2006 2, 55, 88

[Koo93] Koornwinder, Tom H.: On Zeilberger's algorithm and its q-analogue. In: *J. Comput. Appl. Math.* (1993), Nr. 48, S. 91–111 1, 93

[Koo99] Koornwinder, Tom H.: Some simple applications and variants of the q-binomial formula / University of Amsterdam. 1999. – Forschungsbericht 6

[KRM07] Koepf, Wolfram ; Rajković, Predrag M. ; Marinković, Sladjana D.: Properties of q-holonomic functions. In: *J. Difference Equ. Appl.* 13 (2007), Nr. 7, S. 621–638 2, 29, 38

[KS94] Koepf, Wolfram ; Schmersau, Dieter: Spaces of functions satisfying simple differential equations / Konrad-Zuse-Zentrum Berlin. 1994. – Forschungsbericht 1, 36

[KS98] Koekoek, Roelof ; Swarttouw, René F.: *The Askey-scheme of Hypergeometric Orthogonal Polynomials and its q-Analogue*. Delft University of Technology, 1998 (98-17) 13, 14, 15, 93

[KS01] Koepf, Wolfram ; Schmersau, Dieter: On a structure formula for classical q-orthogonal polynomials. In: *J. Comput. Appl. Math.* (2001), Nr. 136, S. 99–107 1, 14, 20

[MW94] Man, Yiu-Kwong ; Wright, Francis J.: Fast polynomial dispersion computation and its application to indefinite summation. In: *ISSAC 1994 Proceedings*, 1994, S. 175–180 55

[Ore33] Ore, Oystein: Theory of non-commutative polynomials. In: *The Annals of Mathematics* 34 (1933), S. 480–508 28

[Pau95] Paule, Peter: Greatest factorial factorization and symbolic summation. In: *J. Symbolic Comput.* 20 (1995), S. 235–268 58

[PS95] Paule, Peter ; Strehl, Volker: Symbolic summation - some recent developments. In: *Computer Algebra in Science and Engineering*, World Scientific, 1995, S. 138–162 58

[Roa96] Roach, Kelly: Hypergeometric function representations. In: *ISSAC 1996 Proceedings*, 1996, S. 301–308 31, 119

[Ryd21] Ryde, Folke: *A contribution to the theory of linear homogeneous geometric difference equations (q-difference equations)*, Lund, Diss., 1921 6

[Sus03] Suslov, Sergei K.: *Developments in Mathematics*. Bd. 9: *An Introduction to Basic Fourier Series*. Dordrecht/Boston/London : Kluwer Academic Publishers, 2003 119

[Wei01] Weixlbaumer, Christian: *Solutions of difference equations with polynomial coefficients*, RISC Linz, J. Kepler University Linz, Diplomarbeit, 2001 56, 58

[Wol99] Wolfram, Stephen ; Fourth (Hrsg.): *The Mathematica Book*. Wolfram Media und Cambridge University Press, 1999 1

[WZ92] Wilf, Herbert S. ; Zeilberger, Doron: An algorithmic proof theory for hypergeometric (ordinary and q-) multisum/integral identities. In: *Invent. Math.* (1992), Nr. 108, S. 575–633 1

Index

Verzeichnis der Sitzungen

Verzeichnis der Algorithmen

Tabellenverzeichnis

Abbildungsverzeichnis

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Dissertation selbständig und ohne unerlaubte Hilfe 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. Kein Teil dieser Arbeit ist in einem anderen Promotions- oder Habilitationsverfahren verwendet worden.

Kassel, den 21. April 2009