

q -類似の Coq による形式化

アドバイザー：Jacques Garrigue 教授

学籍番号：322101289

氏名：中村 薫

February 14, 2023

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 Appendix

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 Appendix

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

はじめに

目的 : q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

形式化 人工言語に数学的な主張とその証明を翻訳し, 正しさを機械的に検証すること

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

形式化 人工言語に数学的な主張とその証明を翻訳し, 正しさを機械的に検証すること

Coq 形式化を行うためのソフトウェア

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

形式化 人工言語に数学的な主張とその証明を翻訳し, 正しさを機械的に検証すること

Coq 形式化を行うためのソフトウェア

形式化の意義：

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

形式化 人工言語に数学的な主張とその証明を翻訳し, 正しさを機械的に検証すること

Coq 形式化を行うためのソフトウェア

形式化の意義：

- 人間がチェックすることが難しい複雑な証明の正しさの保証

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

形式化 人工言語に数学的な主張とその証明を翻訳し, 正しさを機械的に検証すること

Coq 形式化を行うためのソフトウェア

形式化の意義：

- 人間がチェックすることが難しい複雑な証明の正しさの保証
- 証明付きプログラミング

はじめに

目的： q -類似の初等的な結果を Coq を用いて形式化する

q -類似 $q \rightarrow 1$ で通常の数学に戻るような諸概念の拡張

形式化 人工言語に数学的な主張とその証明を翻訳し, 正しさを機械的に検証すること

Coq 形式化を行うためのソフトウェア

形式化の意義：

- 人間がチェックすることが難しい複雑な証明の正しさの保証
- 証明付きプログラミング

本発表における q -類似の定義, 定理及びその証明は Victor Kac, Pokman Cheung の *Quantum Calculus* [2] によるものだが, その形式化を行ったという点において独自性がある. 形式化したコード全体は

<https://github.com/nakamurakaoru/q-analogue/tree/thesis> [5]
にある (q_analogu.v 1556 行, q_tool.v 331 行).

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 Appendix

q -類似の概要

q -類似：以下の 2 つの条件をみたす数学の諸概念の一般化

q -類似の概要

q -類似：以下の 2 つの条件をみたす数学の諸概念の一般化

- $q \rightarrow 1$ とすると通常の数学に一致する

q -類似の概要

q -類似：以下の 2 つの条件をみたす数学の諸概念の一般化

- $q \rightarrow 1$ とすると通常の数学に一致する
- 実数パラメータ q , 実数上の関数 f に対して

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

で定義される q -微分に対してうまく振る舞う

q -類似の概要

q -類似：以下の 2 つの条件をみたす数学の諸概念の一般化

- $q \rightarrow 1$ とすると通常の数学に一致する
- 実数パラメータ q , 実数上の関数 f に対して

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

で定義される q -微分に対してうまく振る舞う

q -類似を考える利点：あえてパラメータを増やすことで証明が簡単になる場合がある

q -類似の概要

q -類似：以下の 2 つの条件をみたす数学の諸概念の一般化

- $q \rightarrow 1$ とすると通常の数学に一致する
- 実数パラメータ q , 実数上の関数 f に対して

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

で定義される q -微分に対してうまく振る舞う

q -類似を考える利点：あえてパラメータを増やすことで証明が簡単になる場合がある

———— Jacobi の三重積 ([2] p35 Theorem 11.1) ————

$z, q \in \mathbb{R}, |q| < 1$ として,

$$\sum_{n=-\infty}^{\infty} q^{n^2} z^n = \prod_{n=1}^{\infty} (1 - q^{2n})(1 + q^{2n-1}z)(1 + q^{2n-1}z^{-1})$$

が成り立つ.

q -Taylor 展開

$f(x)$ を, N 次の実数係数多項式とする. 任意の $c \in \mathbb{R}$ に対し,

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

が成り立つ.

q -Taylor 展開

$f(x)$ を, N 次の実数係数多項式とする. 任意の $c \in \mathbb{R}$ に対し,

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

が成り立つ.

① D_q の定義

q -Taylor 展開

$f(x)$ を, N 次の実数係数多項式とする. 任意の $c \in \mathbb{R}$ に対し,

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

が成り立つ.

- ❶ D_q の定義
- ❷ $[n]$ の定義

q -Taylor 展開

$f(x)$ を, N 次の実数係数多項式とする. 任意の $c \in \mathbb{R}$ に対し,

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

が成り立つ.

- ❶ D_q の定義
- ❷ $[n]$ の定義
- ❸ $(x-c)_q^n$ の定義

q -差分, q -微分の定義をする. 以下, q を 1 でない実数とする.

q -差分, q -微分の定義をする. 以下, q を 1 でない実数とする.

Definition 2.1 ([2] p1 (1.1), p2 (1.5))

関数 $f : \mathbb{R} \rightarrow \mathbb{R}$ に対して, $f(x)$ の q 差分 $d_q f(x)$ を,

$$d_q f(x) := f(qx) - f(x)$$

と定める. 更に, $f(x)$ の q 微分 $D_q f(x)$ を,

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

と定める.

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

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

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

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

$$D_q x^n = \frac{(qx)^n - x^n}{(q - 1)x}$$

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

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

$$D_q x^n = \frac{(qx)^n - x^n}{(q-1)x} = \frac{q^n - 1}{q-1} x^{n-1}$$

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

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

$$D_q x^n = \frac{(qx)^n - x^n}{(q-1)x} = \frac{q^n - 1}{q-1} x^{n-1}$$

通常の微分では, $(x^n)' = nx^{n-1}$ となることと比較して, n の q -類似 $[n]$ を次のように定める ([2] p2 (1.9)).

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

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

$$D_q x^n = \frac{(qx)^n - x^n}{(q-1)x} = \frac{q^n - 1}{q-1} x^{n-1}$$

通常の変分では, $(x^n)' = nx^{n-1}$ となることと比較して, n の q -類似 $[n]$ を次のように定める ([2] p2 (1.9)).

$$[n] = \frac{q^n - 1}{q - 1}$$

自然数の q -類似

$f(x) = x^n$ ($n \in \mathbb{N}$) を定義に沿って q -微分する.

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

$$D_q x^n = \frac{(qx)^n - x^n}{(q-1)x} = \frac{q^n - 1}{q-1} x^{n-1}$$

通常の微分では, $(x^n)' = nx^{n-1}$ となることと比較して, n の q -類似 $[n]$ を次のように定める ([2] p2 (1.9)).

$$[n] = \frac{q^n - 1}{q - 1} (= 1 + q + q^2 + \cdots + q^{n-1})$$

$(x - a)^n$ の q -類似

Definition 2.2 ([2] p8 Definition (3.4))

$x, a \in \mathbb{R}, n \in \mathbb{N}$ に対して, $(x - a)^n$ の q -類似 $(x - a)_q^n$ を,

$$(x - a)_q^n = \begin{cases} 1 & \text{if } n = 0 \\ (x - a)(x - qa) \cdots (x - q^{n-1}a) & \text{if } n \geq 1 \end{cases}$$

と定義する.

$(x - a)^n$ の q -類似

Definition 2.2 ([2] p8 Definition (3.4))

$x, a \in \mathbb{R}, n \in \mathbb{N}$ に対して, $(x - a)^n$ の q -類似 $(x - a)_q^n$ を,

$$(x - a)_q^n = \begin{cases} 1 & \text{if } n = 0 \\ (x - a)(x - qa) \cdots (x - q^{n-1}a) & \text{if } n \geq 1 \end{cases}$$

と定義する.

Proposition 2.3

$n \in \mathbb{Z}_{>0}$ に対し,

$$D_q(x - a)_q^n = [n](x - a)_q^{n-1}$$

が成り立つ.

Definition 2.4 ([2] p7 (3.1))

$n \in \mathbb{N}$ について, 階乗の q -類似を以下のように定める.

$$[n]! := \begin{cases} 1 & (n = 0) \\ [n] \times [n-1] \times \cdots \times [1] & (n \geq 1) \end{cases}$$

Definition 2.4 ([2] p7 (3.1))

$n \in \mathbb{N}$ について, 階乗の q -類似を以下のように定める.

$$[n]! := \begin{cases} 1 & (n = 0) \\ [n] \times [n-1] \times \cdots \times [1] & (n \geq 1) \end{cases}$$

Theorem 2.5 ([2] p12 Theorem 4.1)

$f(x)$ を, N 次の実数係数多項式とする. 任意の $c \in \mathbb{R}$ に対し,

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

が成り立つ.

Gauss's binomial formula

Lemma 2.6 ([2] p15 Example (5.5))

$n \in \mathbb{Z}_{>0}$ について,

$$(x + a)_q^n = \sum_{j=0}^n \left[\begin{matrix} n \\ j \end{matrix} \right] q^{j(j-1)/2} a^j x^{n-j}$$

が成り立つ. この式は Gauss's binomial formula と呼ばれる.

Gauss's binomial formula

Lemma 2.6 ([2] p15 Example (5.5))

$n \in \mathbb{Z}_{>0}$ について,

$$(x + a)_q^n = \sum_{j=0}^n \left[\begin{matrix} n \\ j \end{matrix} \right] q^{j(j-1)/2} a^j x^{n-j}$$

が成り立つ. この式は Gauss's binomial formula と呼ばれる.

Definition 2.7 ([2] p12 (4.5))

$n \geq j$ をみたす $n, j \in \mathbb{N}$ について, 二項係数の q -類似を以下のように定める.

$$\left[\begin{matrix} n \\ j \end{matrix} \right] := \frac{[n]!}{[j]![n-j]!}$$

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 Appendix

Coq の使い方 コマンド

Coq に与える命令はコマンドとタクティックの 2 種類がある.

Coq の使い方 コマンド

Coq に与える命令はコマンドとタクティックの 2 種類がある.

Require Import ライブラリを読み込む.

Variable 特定の型を持つ変数を宣言する

Definition 新たに関数を定義する.

Fixpoint 再帰関数を定義する. 停止性が保証されていない関数を定義することはできない.

Lemma 補題を宣言する. **Lemma** の代わりに **Theorem**, **Corollary** 等でも同じ機能をもつ.

Proof/Qed **Proof** は **Lemma** の後に書いて補題の主張と証明を分ける. 証明を完了させて **Qed** を書くことで **Coq** に補題を登録することができ, 他の補題の証明に使えるようになる.

Coq の使い方 コマンド

Coq に与える命令はコマンドとタクティックの 2 種類がある.

Require Import ライブラリを読み込む.

Variable 特定の型を持つ変数を宣言する

Definition 新たに関数を定義する.

Fixpoint 再帰関数を定義する. 停止性が保証されていない関数を定義することはできない.

Lemma 補題を宣言する. **Lemma** の代わりに **Theorem**, **Corollary** 等でも同じ機能をもつ.

Proof/Qed **Proof** は **Lemma** の後に書いて補題の主張と証明を分ける. 証明を完了させて **Qed** を書くことで **Coq** に補題を登録することができ, 他の補題の証明に使えるようになる.

タクティックは **Proof...Qed** の間に使われる. よく使われるタクティックは **move**, **apply**, **rewrite** の 3 つ.

自然数 m, n について

$$m = 0 \implies n + m = n$$

自然数 m, n について

$$m = 0 \implies n + m = n$$

Lemma substitution $m\ n : m = 0 \rightarrow n + m = n.$

```
1 subgoal
```

```
m, n : nat
```

```
-----
```

```
m = 0 → n + m = n
```

Coq の使い方 代入計算

```
1 subgoal
```

```
m, n : nat
```

```
-----
```

```
m = 0 → n + m = n
```

タクティック : `move=> Hm`

```
1 subgoal
m, n : nat
Hm : m = 0
-----
n + m = 0
```

```
1 subgoal
m, n : nat
Hm : m = 0
-----
n + m = 0
```

タクティック : `rewrite Hm`

```
1 subgoal
m, n : nat
Hm : m = 0
-----
n + 0 = n
```

```
1 subgoal
m, n : nat
Hm : m = 0
-----
n + 0 = n
```

タクティック : `rewrite addn0`

Coq の使い方 代入計算

```
1 subgoal
m, n : nat
Hm : m = 0
-----
n = n
```

Coq の使い方 代入計算

```
1 subgoal
m, n : nat
Hm : m = 0
-----
n = n
```

タクティック : done

No more subgoals.

No more subgoals.

コマンド : Qed

Coq の使い方 代入計算

Lemma substitution m n : m = 0 \rightarrow n + m = n.

Proof.

```
move  $\Rightarrow$  Hm.  
rewrite Hm.  
rewrite addn0.  
by [].
```

Qed.

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 Appendix

本節の目標

本節の目標

Theorem `q_Taylorp n (f : {poly R}) c :`
 $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{size } f = n.+1 \rightarrow$
 $f = \sum_{(0 \leq i < n.+1)} ((Dq^{p'} \setminus^i f).[c] * : (\text{qbinom_pos_poly } c \ i / (\text{qfact } i)\%P)).$

Theorem `Gauss_binomial a n : (\forall n, \text{qfact } n \neq 0) \rightarrow`
 $\text{qbinom_pos_poly } (-a) \ n =$
 $\sum_{(0 \leq i < n.+1)} (\text{qbicoef } n \ i * q^{+ (i * (i - 1))./2 * a^{+ i}} * : 'X^{(n - i)}).$

本節の目標

Theorem `q_Taylorp` n ($f : \{\text{poly } R\}$) $c :$
 $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{size } f = n.+1 \rightarrow$
 $f = \sum_{(0 \leq i < n.+1)} ((Dq^{p'} \setminus^i f).[c] * : (\text{qbinom_pos_poly } c \ i / (\text{qfact } i)\%P)).$

Theorem `Gauss_binomial` a $n : (\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{qbinom_pos_poly } (-a) \ n =$
 $\sum_{(0 \leq i < n.+1)} (\text{qbicoef } n \ i * q^{+ (i * (i - 1))./2 * a^{+ i}} * : 'X^{(n - i)}).$

- q -微分の形式化

本節の目標

Theorem `q_Taylorp n (f : {poly R}) c :`
 $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{size } f = n.+1 \rightarrow$
 $f = \sum_{(0 \leq i < n.+1)} ((Dq^{p'} \setminus^i f).[c] * : (\text{qbinom_pos_poly } c \ i / (\text{qfact } i)\%P)).$

Theorem `Gauss_binomial a n : (\forall n, \text{qfact } n \neq 0) \rightarrow`
 $\text{qbinom_pos_poly } (-a) \ n =$
 $\sum_{(0 \leq i < n.+1)} (\text{qbicoef } n \ i * q^{+ (i * (i - 1))./2 * a^{+ i}} * : 'X^{(n - i)}).$

- q -微分の形式化
- $[n]$ の形式化

本節の目標

Theorem `q_Taylorp n (f : {poly R}) c :`
 $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{size } f = n.+1 \rightarrow$
 $f = \sum_{(0 \leq i < n.+1)} ((Dq^p' \setminus^i) f).[c] * : (\text{qbinom_pos_poly } c \ i / (\text{qfact } i)\%P).$

Theorem `Gauss_binomial a n :` $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{qbinom_pos_poly } (-a) \ n =$
 $\sum_{(0 \leq i < n.+1)} (\text{qbicoef } n \ i * q^{+ (i * (i - 1))./2 * a^{+ i}} * : 'X^{(n - i)}).$

- q -微分の形式化
- $[n]$ の形式化
- $(x - a)_q^n$ の形式化

本節の目標

Theorem `q_Taylorp n (f : {poly R}) c :`
 $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{size } f = n.+1 \rightarrow$
 $f = \sum_{(0 \leq i < n.+1)} ((Dq^p' \setminus^i) f).[c] * (\text{qbinom_pos_poly } c \ i / (\text{qfact } i)\%P).$

Theorem `Gauss_binomial a n :` $(\forall n, \text{qfact } n \neq 0) \rightarrow$
 $\text{qbinom_pos_poly } (-a) \ n =$
 $\sum_{(0 \leq i < n.+1)} (\text{qbicoef } n \ i * q^{+(i * (i - 1))./2 * a^{+ i}} * 'X^{(n - i)}).$

- q -微分の形式化
- $[n]$ の形式化
- $(x - a)_q^n$ の形式化
- 関数から多項式へ

$$d_q f(x) := f(qx) - f(x) \quad D_q f(x) := \frac{d_q f(x)}{d_q x}$$

$$d_q f(x) := f(qx) - f(x) \quad D_q f(x) := \frac{d_q f(x)}{d_q x}$$

Variables (R : rcfType) (q : R).

Hypothesis Hq : q - 1 ≠ 0.

Notation "f // g" := (fun x ⇒ f x / g x) (at level 40).

Definition dq (f : R → R) x := f (q * x) - f x.

Definition Dq f := dq f // dq id.

$[n]$ の形式化

$$[n] := \frac{q^n - 1}{q - 1} \quad D_q x^n = \frac{(qx)^n - x^n}{(q - 1)x} = \frac{q^n - 1}{q - 1} \cdot \frac{x^n}{x} = [n]x^{n-1}$$

$[n]$ の形式化

$$[n] := \frac{q^n - 1}{q - 1} \quad D_q x^n = \frac{(qx)^n - x^n}{(q - 1)x} = \frac{q^n - 1}{q - 1} \cdot \frac{x^n}{x} = [n]x^{n-1}$$

Definition `qnat n : R := (q ^ n - 1) / (q - 1).`

Lemma `Dq_pow n x : x ≠ 0 →`

`Dq (fun x ⇒ x ^ n) x = qnat n * x ^ (n - 1).`

Proof.

`move ⇒ Hx.`

`rewrite /Dq /dq /qnat.`

`rewrite -{4}(mul1r x) -mulrBl expfzM1 -add_div;`

`last by apply mulf_neq0.`

`rewrite [in x ^ n](_ : n = (n - 1) + 1) //; last by rewrite subrK.`

`rewrite expfzDr ?expr1z ?mulrA -?mulNr ?red_frac_r ?add_div //.`

`rewrite -{2}[x ^ (n - 1)]mul1r -mulrBl mulrC mulrA.`

`by rewrite [in (q - 1)^-1 * (q ^ n - 1)] mulrC.`

Qed.

$(x - a)_q^n$ の形式化

$$(x - a)_q^n := \begin{cases} 1 & \text{if } n = 0 \\ (x - a)(x - qa) \cdots (x - q^{n-1}a) & \text{if } n \geq 1 \end{cases}$$

$$D_q(x - a)_q^n = [n](x - a)_q^{n-1} \quad (n \in \mathbb{Z}_{>0})$$

$(x - a)_q^n$ の形式化

$$(x - a)_q^n := \begin{cases} 1 & \text{if } n = 0 \\ (x - a)(x - qa) \cdots (x - q^{n-1}a) & \text{if } n \geq 1 \end{cases}$$
$$D_q(x - a)_q^n = [n](x - a)_q^{n-1} \quad (n \in \mathbb{Z}_{>0})$$

```
Fixpoint qbinom_pos a n x :=  
  match n with  
  | 0 => 1  
  | n0.+1 => (qbinom_pos a n0 x) * (x - q ^ n0 * a)  
  end.
```

Theorem $D_q \text{qbinom_pos } a \ n \ x : x \neq 0 \rightarrow$
 $D_q (\text{qbinom_pos } a \ n.+1) \ x = \text{qnat } n.+1 * \text{qbinom_pos } a \ n \ x.$

関数から多項式へ

約分に条件が必要ない

約分に条件が必要ない

$x/x = 1$ を計算するとき

- 実数 $\dots x \neq 0$ が必要

約分に条件が必要ない

$x/x = 1$ を計算するとき

- 実数 $\dots x \neq 0$ が必要
- 多項式 $\dots x$ は単項式なので自動的に $x \neq 0$ (ゼロ多項式)

約分に条件が必要ない

$x/x = 1$ を計算するとき

- 実数 $\dots x \neq 0$ が必要
- 多項式 $\dots x$ は単項式なので自動的に $x \neq 0$ (ゼロ多項式)

→ 多項式で考えれば約分した後でも $x = 0$ での値が求められる.

約分に条件が必要ない

$x/x = 1$ を計算するとき

- 実数 $\dots x \neq 0$ が必要
- 多項式 $\dots x$ は単項式なので自動的に $x \neq 0$ (ゼロ多項式)

→ 多項式で考えれば約分した後でも $x = 0$ での値が求められる.

$$D_q(x - a)_q^n = [n](x - a)_q^{n-1}$$

という計算をした後でも 0 での値が求められる.

約分に条件が必要ない

$x/x = 1$ を計算するとき

- 実数 $\dots x \neq 0$ が必要
- 多項式 $\dots x$ は単項式なので自動的に $x \neq 0$ (ゼロ多項式)

→ 多項式で考えれば約分した後でも $x = 0$ での値が求められる.

$$D_q(x - a)_q^n = [n](x - a)_q^{n-1}$$

という計算をした後でも 0 での値が求められる.

→ Gauss's binomial formula の証明に必要

多項式での再定義

q -差分

Definition `scale_var (p : {poly R}):=`
`\poly_(i < size p) (q ^ i * p'_i).`

Definition `dqp p := scale_var p - p.`

多項式での再定義

q -差分

Definition `scale_var (p : {poly R}):=`
`\poly_(i < size p) (q ^ i * p'_i).`

Definition `dqp p := scale_var p - p.`

`scale_var` $\cdots p \mapsto (p \text{ の } i \text{ 次の係数を } q^i \text{ 倍した多項式})$

多項式での再定義

q -差分

Definition `scale_var (p : {poly R}):=`
`\poly_(i < size p) (q ^ i * p'_i).`

Definition `dqp p := scale_var p - p.`

`scale_var ... p` \mapsto (p の i 次の係数を q^i 倍した多項式)

q -微分

Definition `Dqp p := dqp p %/ dqp 'X.`

Definition `Dqp' (p : {poly R}) :=`
`\poly_(i < size p) (qnat (i.+1) * p'_{i.+1}).`

多項式での再定義

q -差分

Definition `scale_var (p : {poly R}):=`
`\poly_(i < size p) (q ^ i * p'_i).`

Definition `dqp p := scale_var p - p.`

`scale_var` $\cdots p \mapsto (p \text{ の } i \text{ 次の係数を } q^i \text{ 倍した多項式})$

q -微分

Definition `Dqp p := dqp p %/ dqp 'X.`

Definition `Dqp' (p : {poly R}) :=`
`\poly_(i < size p) (qnat (i.+1) * p'_{i.+1}).`

$(x - a)_q^n$

Fixpoint `qbinom_pos_poly a n :=`

`match n with`

`| 0 \Rightarrow 1`

`| n0.+1 \Rightarrow (qbinom_pos_poly a n0) * ('X - (q ^ n0 * a)%:P)`

`end.`

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

$$f(x) = \sum_{j=0}^N (D_q^j f)(c) \frac{(x-c)_q^j}{[j]!}$$

```
Fixpoint qfact n :=  
  match n with  
  | 0 => 1  
  | n0.+1 => qfact n0 * qnat n0.+1  
end.
```

```
Theorem q_Taylorp n (f : {poly R}) c :  
  (∀ n, qfact n ≠ 0) →  
  size f = n.+1 →  
  f = \sum_ (0 ≤ i < n.+1)  
    ((Dqp' ^ i) f).[c] *: (qbinom_pos_poly c i / (qfact i)%:P).
```

$$(x + a)_q^n = \sum_{j=0}^n \begin{bmatrix} n \\ j \end{bmatrix} q^{j(j-1)/2} a^j x^{n-j}$$

Gauss's binomial formula の形式化

$$(x + a)_q^n = \sum_{j=0}^n \begin{bmatrix} n \\ j \end{bmatrix} q^{j(j-1)/2} a^j x^{n-j}$$

Definition `qbicoef n j := qfact n / (qfact j * qfact (n - j)).`

Theorem `Gauss_binomial a n : (∀ n, qfact n ≠ 0) →
 qbinom_pos_poly (-a) n =
 \sum_(0 ≤ i < n.+1)
 (qbicoef n i * q ^+ (i * (i - 1))./2 * a ^+ i) *: 'X^(n - i).`

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 Appendix

現在開発中のライブラリ `mathcomp analysis` [4] の利用

現在開発中のライブラリ `mathcomp analysis` [4] の利用

- $q \rightarrow 1$ で通常の数学に戻ることの形式化

現在開発中のライブラリ `mathcomp analysis` [4] の利用

- $q \rightarrow 1$ で通常の数学に戻ることの形式化
- 無限和に関する形式化

現在開発中のライブラリ `mathcomp analysis` [4] の利用

- $q \rightarrow 1$ で通常の数学に戻ることの形式化
- 無限和に関する形式化
→ Gauss's binomial formula の拡張, q -指数関数, q -三角関数

- 1 はじめに
- 2 q -類似
- 3 Coq
- 4 q -類似の形式化
- 5 今後の展望
- 6 **Appendix**

Appendix q -微分の諸性質

線形性 ([2] p2)

$$D_q(af(x) + bg(x)) = aD_qf(x) + bD_qg(x)$$

積の微分法則 ([2] p3 (1.11), (1.12))

$$D_q(f(x)g(x)) = f(qx)D_qg(x) + g(x)D_qf(x)$$

$$D_q(f(x)g(x)) = f(x)D_qg(x) + g(qx)D_qf(x)$$

商の微分法則 ([2] p3 (1.13), (1.14))

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

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

Appendix $(x - a)_q^n$ の指数法則と負冪

Proposition 6.1 ([2] p8 (3.6))

$x, a \in \mathbb{R}, m, n \in \mathbb{Z}_{>0}$ について,

$$(x - a)_q^{m+n} = (x - a)_q^m (x - q^m a)_q^n$$

が成り立つ.

Definition 6.2 ([2] p9 (3.7))

$x, a \in \mathbb{R}, l \in \mathbb{Z}_{>0}$ とする. このとき,

$$(x - a)_q^{-l} := \frac{1}{(x - q^{-l}a)_q^l}$$

と定める.

Appendix q -指数関数, q -三角関数

q -指数関数 [2] 9

$$e_q^x := \sum_{j=0}^{\infty} \frac{x^j}{[j]!} \quad E_q^x := \sum_{j=0}^{\infty} q^{j(j-1)/2} \frac{x^j}{[j]!}$$
$$D_q e_q^x = e_q^x \quad D_q E_q^x = E_q^{qx}$$
$$e_q^x e_q^y = e_q^{x+y} \quad (\text{if } yx = qxy)$$
$$e_q^x E_q^{-x} = 1 \quad e_{1/q}^x = E_q^x$$

q -三角関数 [2] 10

$$\sin_q x := \frac{e_q^{ix} - e_q^{-ix}}{2i} \quad \cos_q x := \frac{e_q^{ix} + e_q^{-ix}}{2}$$
$$\text{Sin}_q x := \frac{E_q^{ix} - E_q^{-ix}}{2i} \quad \text{Cos}_q x := \frac{E_q^{ix} + E_q^{-ix}}{2}$$
$$\cos_q x \text{Cos}_q x + \sin_q x \text{Sin}_q x = 1$$

「命題 P, Q について, P かつ $P \implies Q$ であれば, Q が成り立つ」

「命題 P, Q について, P かつ $P \implies Q$ であれば, Q が成り立つ」

From mathcomp Require Import ssreflect.

Lemma modus_ponens (P Q : Prop) : P \wedge (P \rightarrow Q) \rightarrow Q.

Appendix モーダスポーネンス

「命題 P, Q について, P かつ $P \implies Q$ であれば, Q が成り立つ」

From mathcomp Require Import ssreflect.

Lemma modus_ponens (P Q : Prop) : P \wedge (P \rightarrow Q) \rightarrow Q.

Prop は Coq での命題全体を表す型, \wedge は「かつ」を表す

Appendix モーダスポーネンス

1 subgoal

P, Q : Prop

$P \wedge (P \rightarrow Q) \rightarrow Q$

Appendix モーダスポーネンス

1 subgoal

P, Q : Prop

$P \wedge (P \rightarrow Q) \rightarrow Q$

タクティック : move=> []

Appendix モーダスポーネンス

1 subgoal

P, Q : Prop

$P \rightarrow (P \rightarrow Q) \rightarrow Q$

Appendix モーダスポーネンス

1 subgoal

P, Q : Prop

$P \rightarrow (P \rightarrow Q) \rightarrow Q$

タクティック : move=> p

Appendix モーダスポーネンス

1 subgoal

P, Q : Prop

p : P

$(P \rightarrow Q) \rightarrow Q$

Appendix モーダスポーネンス

```
1 subgoal
P, Q : Prop
p : P
-----
(P → Q) → Q
```

タクティック : `move=> pq`

Appendix モーダスポーネンス

```
1 subgoal
P, Q : Prop
p : P
pq : P → Q
-----
Q
```

Appendix モーダスポーネンス

```
1 subgoal
P, Q : Prop
p : P
pq : P → Q
-----
Q
```

タクティック : `apply pq`

Appendix モーダスポーネンス

```
1 subgoal
P, Q : Prop
p : P
pq : P → Q
-----
P
```

Appendix モーダスポーネンス

```
1 subgoal
P, Q : Prop
p : P
pq : P → Q
-----
P
```

タクティック : done

No more subgoals.

Appendix モーダスポーネンス

No more subgoals.

コマンド : Qed

Appendix モーダスポーネンス

Lemma modus_ponens (P Q : **Prop**) : $P \wedge (P \rightarrow Q) \rightarrow Q$.

Proof.

move \Rightarrow [].

move \Rightarrow p.

move \Rightarrow pq.

apply pq.

done.

Qed.

標準ライブラリ [1] に加えて mathcomp [3] を用いる.

標準ライブラリ [1] に加えて `mathcomp` [3] を用いる.
`mathcomp` の型には階層構造があり, より一般の型の性質を引き継ぐ.

標準ライブラリ [1] に加えて `mathcomp` [3] を用いる.
`mathcomp` の型には階層構造があり, より一般の型の性質を引き継ぐ.
今回は実数として `rcfType`(Real Closed Field : 実閉体) を使う.

標準ライブラリ [1] に加えて `mathcomp` [3] を用いる.
`mathcomp` の型には階層構造があり, より一般の型の性質を引き継ぐ.
今回は実数として `rcfType`(Real Closed Field : 実閉体) を使う.

`eqType` \rightarrow `choiceType`

\rightarrow `zmodType` \rightarrow `ringType` \rightarrow `comRingType` \rightarrow `comUnitRingType`

\rightarrow `idomainType` \rightarrow `fieldType`

\rightarrow `numFieldType` \rightarrow `realFieldType` \rightarrow `rcfType`

標準ライブラリ [1] に加えて mathcomp [3] を用いる.
mathcomp の型には階層構造があり, より一般の型の性質を引き継ぐ.
今回は実数として rcfType(Real Closed Field : 実閉体) を使う.

eqType \rightarrow choiceType

\rightarrow zmodType \rightarrow ringType \rightarrow comRingType \rightarrow comUnitRingType

\rightarrow idomainType \rightarrow fieldType

\rightarrow numFieldType \rightarrow realFieldType \rightarrow rcfType

ringType, fieldType の性質が重要

Appendix Coq の多項式

`T: ringType` のとき, `{poly T} ... T` 係数多項式全体
`{poly R}` も `ringType` の構造を持っている

`\poly_(i < n) E(i)` 次数が $n-1$ 次以下, i 次の係数が $E(i)$ である多項式

`c%P` 定数 c のみからなる単項式

`'X` 変数 x のみからなる単項式

`p'_i` 多項式 p の i 次の係数

`size p` 多項式 p の次数 $+1$

`p.[x]` 多項式 p の x での値

Appendix Dqp の正当性

Definition $\text{Dqp } p := \text{dqp } p \% \text{dqp 'X}.$

$p \% p'$ は多項式 p を多項式 p' で割った商
→ 余りが 0 でない可能性があるため, q -微分の正しい形式化である保証がない

Lemma $\text{Dqp_ok } p : \text{dqp 'X} \% \text{dqp } p.$

Import `FracField`.

Local Notation $\text{tofrac} := (@\text{tofrac} [\text{idomainType } \text{of } \{\text{poly } R\}]).$

Local Notation $"x \%:F" := (\text{tofrac } x).$

Theorem $\text{Dqp_ok_frac } p : (\text{dqp } p)\%:F / (\text{dqp 'X})\%:F = (\text{Dqp } p)\%:F.$

$p' \% p \cdots p$ が p' で割り切れる

Appendix Dq と Dqp

Definition `ap_op_poly` ($D : (R \rightarrow R) \rightarrow (R \rightarrow R)$) ($p : \{\text{poly } R\}$) :=
D (fun (x : R) \Rightarrow p.[x]).

Notation "`D # p`" := (ap_op_poly D p) (at level 49).

Lemma `dqp_dqE` p x : (dqp p).[x] = (dq # p) x.

Lemma `Dqp_Dqp'E` p : Dqp p = Dqp' p.

Lemma `Dqp'_DqE` p x : $x \neq 0 \rightarrow$ (Dqp' p).[x] = (Dq # p) x.

Lemma `Dqp'_qbinom_poly` a n :
Dqp' (qbinom_pos_poly a n.+1) =
(qnat n.+1) *: (qbinom_pos_poly a n).



Coq Team, *The Coq Standard Library*,
<https://coq.inria.fr/distrib/current/stdlib/>, 2023.



Victor Kac, Pokman Cheung, *Quantum Calculus*, Springer, 2001.



Mathematical Components Team, *Mathematical Components*,
<https://github.com/math-comp/math-comp>, 2023.



Mathematical Components Team, *Mathematical Components
compliant Analysis Library*,
<https://github.com/math-comp/analysis>, 2023.



中村 薫, *q-analogue*,
<https://github.com/nakamurakaoru/q-analogue/tree/thesis>,
2023.