HYPERG

Maple Package

User's Reference Manual

Version 1.0

Bruno Gauthier
Bruno.Gauthier@univ-mlv.fr

Université Marne-la-Vallée

Institut Gaspard Monge

June 1999

Table of contents

Copyright notice

Abstract

HYPERG is a *Maple* package for the manipulation of generalized hypergeometric series denoted by the usual symbol $F(a_1, a_2, \ldots; b_1, b_2, \ldots; x)$. It also provides tools for automatic proofs of identities, but they are just tools: every step in a series of manipulations has to be controlled by the user. The goal of this package is to make the algorithms in this area better known, and to help you enter the wonderful world of hypergeometric series.
Many features of this package are due to C. Krattenthaler and his *Mathematica* package `HYP` [15]. Many thanks to C. Krattenthaler, P. Paule, M. Petkovšek, A. Tefera and D. Zeilberger for their helpful comments.

"There must be many universities to-day, where 95 per cent, if not 100 per cent, of the functions studied by physics, engineering, and even mathematics students, are covered by the single symbol $F(a, b; c; x)$."

— W. W. Sawyer [27]

Part I

Installation

This part is devoted to the installation of HYPERG. This package is developped at the University of Marne-la-Vallée, it is available on the *World Wide Web* at the URL `http://www-igm.univ-mlv.fr/~gauthier/`.

For the moment, this program only works under *Maple V* Release 2 or later.

0.1 Installing HYPERG on a UNIX/LINUX system

You must have the following files:

`README`	
`HYPERG`	The code file
`HYPERG.tst`	The test file
`HYPERG.ps.gz`	This manual (gzipped postscript-file)

To compile the package, you must move the `HYPERG` file in your maple packages directory (for example: `/usr/people/joe/maple_pack`) then type the UNIX command:

```
maple -s -q < HYPERG
```

Then, you have to tell *Maple* where to find the compiled file `HYPERG.m`. You must add the two following lines to the `.mapleinit` file in your HOME directory, or create such a file if you don't yet have one:

- For *Maple V* Release **1**:

```
HYPERGLib:=`/usr/people/joe/maple_pack`:
_liblist:=[HYPERGLib]:
```

- For *Maple V* Release **2**, **3** or **4**:

```
HYPERGLib:=`/usr/people/joe/maple_pack`:
libname:=libname, HYPERGLib:
```

0.2 Using the package

You can use `HYPERG` by running *Maple V* and then loading the package by the *Maple* command `with(HYPERG);`. To get help, you can type `?HYPERG`, and to know the loaded version of `HYPERG`, you can type `` `HYPERG/Version`; ``.

Set the global variable `'HYPERG/Verbose'` to a value greater than 0 to display information about the execution of some procedures (the greater the value, the more informations displayed). Its default value is 0 (for no information). You can for instance set it to 1 to trace your mathematical work.

0.3 Contact

Bruno Gauthier

Université de Marne-la-Vallée,

Institut Gaspard Monge,

Cité Descartes,

5, Bd Descartes, Champs-sur-Marne,

F-77454 Marne-la-Vallée, CEDEX 2,

Email: `Bruno.Gauthier@univ-mlv.fr`

URL: `http://www-igm.univ-mlv.fr/~gauthier`

HYPERG's main page: `http://www-igm.univ-mlv.fr/~gauthier/HYPERG.html`

Part II

Description of the package

Chapter 1

Introduction

This is a *Maple* package for the manipulation and identification of hypergeometric series and identities. The package provides several tools (about seventy functions) for:

- transforming sums into hypergeometric notation ;
- manipulating factorial expressions ;
- applying summation and transformation formulas ;
- applying contiguous relations ;
- computing formal limits of hypergeometric expressions ;
- manipulating linear recurrence equations ;
- finding polynomial, rational and hypergeometric solutions of recurrences ;
- using the Gosper and Zeilberger algorithms.

You can also manipulate basic hypergeometric series (q-series), see part **V**.

```
$ maple
    |\^/|     Maple V Release 3 (Univ-Marne La Vallee)
._|\|   |/|_. Copyright (c) 1981-1994 by Waterloo Maple Software and the
 \  MAPLE  /  University of Waterloo. All rights reserved. Maple and Maple V
 <____ ____>  are registered trademarks of Waterloo Maple Software.
      |       Type ? for help.
> with(HYPERG);

[AddParam, BaseSplit, CheckRec, Ext1, Ext2, FirstTerms, GenQRec, GenRec,

    Gosper, Homog, HypContig, HypContigList, HypConverg, HypDiff, HypEval,

    HypOrder, HypPermBoth, HypPermLow, HypPermUp, HypSimplify, HypSolQRec,

    HypSolRec, HypSum, HypSumList, HypSumPrint, HypToRec, HypToVHyp, HypTransf,

    HypTransfList, HypTransfPrint, HypType, HypergToRec, Inv, IsHYP, IsHomog,

    IsHyperg, IsQBIN, IsQHYP, IsQRF, IsRF, IsVHYP, IsWHYP, Lim, Linear1,
```

```
Linear2, MapApply, MapList, Neg1, Neg2, PolySolQRec, PolySolRec, Prove,
QBinEval, QHypEval, QHypOrder, QHypToWHyp, QHypType, QRfEval, RatioSolQRec,
RatioSolRec, RecOrder, RfEval, ShiftRec, SimplifyRec, Split, SubsRec,
SumToHyp, SumToRec, SummandToRec, Time, Trans, VHypToHyp, WHypToQHyp]
```

The following chapters (organized by topic) describe all the features of **HYPERG** package.

Chapter 2

Background

This chapter has two goals:

- introduce hypergeometric terms (definition, notation) ;
- present basic `HYPERG` functions.

The aim is the manipulation of hypergeometric expressions.

```
HYP & RF: IsHYP, IsRF, HypEval, RfEval
IsHyperg, HypergToRec
AddParam, FirstTerms, HypConverg, HypDiff
HypPermBoth, HypPermLow, HypPermUp, HypSimplify
VHYP: HypType, HypOrder, IsVHYP, HypToVHyp, VHypToHyp
MapList, MapApply
RecOrder, SubsRec, CheckRec, SimplifyRec, ShiftRec, IsHomog, Homog
```

2.1 Basics

The basic objects are:

- the Gamma function `GAMMA(k)`: $\Gamma(k) = (k-1)!$;
- the binomial coefficient `binomial(n,k)`: $\binom{n}{k}$, which satisfies the well-known recurrence relation:
$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} ;$$
- the rising factorial (also known as *Pochhammer* symbol) `RF[a,k]`:
$$(a)_k := \begin{cases} a\,(a+1)\ldots(a+k-1) & k = 1, 2, \ldots, \\ 1 & k = 0, \\ \left[(a-1)\,(a-2)\ldots(a+k)\right]^{-1} & k = -1, -2, \ldots \qquad a \neq 1, 2, \ldots, -k. \end{cases}$$

  The usual identities for this Pochhammer symbol are:
$$\frac{(a)_n}{(a)_m} = (a+m)_{n-m} \ , \qquad (a)_n = \frac{\Gamma(a+n)}{\Gamma(a)} \quad \text{if } a+n \neq 0, -1, -2, \ldots ;$$

- and also the hypergeometric series `HYP[[a_1,...,a_r],[b_1,...,b_s],z]`:

$$ {}_rF_s \left[\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} ; z \right] \ , $$

  (see notation 1 page 19).

Definition 1 *A function $F(k)$ is **rational** in k if there exist two polynomials u and v relatively prime, such that:*

$$ F(k) = \frac{u(k)}{v(k)} \ . \tag{2.1} $$

Definition 2 *A series $\sum_{k\geq 0} t_k$ is **geometric** if the ratio of two consecutive terms is constant:*

$$ \frac{t_{k+1}}{t_k} = \boldsymbol{constant} \ . \tag{2.2} $$

*The k^{th} term of a geometric series is of the form $c\,x^k$ where c and x are constants (independent of the summation index k). Therefore a general **geometric** series looks like:*

$$ \sum_{k\geq 0} c\,x^k \ . \tag{2.3} $$

Definition 3 *A series $\sum_{k\geq 0} t_k$ is **hypergeometric** if the ratio of two consecutive terms is a rational function of the summation index k:*

$$ \frac{t_{k+1}}{t_k} = \frac{p(k)}{q(k)} \ , \tag{2.4} $$

where p and q are polynomials in k.

Example 1 *2^n, $n!$ and $\binom{2\,n}{n}$ are hypergeometric terms in n, but $\cos n$ is not.*

The `IsHyperg` function allows to test if a term is hypergeometric:

```
> IsHyperg( binomial(n,k), n );

                              true

> IsHyperg( binomial(n/2,k), n );

                              false

> IsHyperg( binomial(n,k), k );

                              true

> IsHyperg( 2^p/p!, p );

                              true

> IsHyperg( a+2^n, n );

                              false

> IsHyperg( (-1)^n * RF[n,3] * 2^n / (n+b), n );

                              true
```

Obviously, a hypergeometric term t_k satisfy a first-order linear homogeneous recurrence relation:

$$p(k)\, t_k - q(k)\, t_{k+1} = 0 \ ,$$

where p and q are polynomials in k.

```
> HypergToRec( 2^n, U(n) );

                          2 U(n) - U(n + 1) = 0

> HypergToRec( binomial(2*k,k), T(k) );

                    (4 k + 2) T(k) - (k + 1) T(k + 1) = 0

> HypergToRec( binomial(n,k)*2^n*(-1)^n*RF[n,4], U(n) );

             - 2 (n + 4) (n + 1) U(n) - (n + 1 - k) n U(n + 1) = 0

> HypergToRec( cos(z), V(z) );
Error, (in HypergToRec) cos(z), is not a hypergeometric term
```

Definition 4 *Two hypergeometric terms U_k and V_k are similar ($U_k \sim V_k$) if $\frac{U_k}{V_k}$ is a rational function of k.*

Notation 1 *The standard form of a (generalized) hypergeometric series is*

$$ {}_rF_s \left[\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} ; z \right] := \sum_{k=0}^{\infty} \frac{(a_1)_k \ldots (a_r)_k}{(b_1)_k \ldots (b_s)_k} \frac{z^k}{k!} \ , \tag{2.5}$$

where z is called "evaluation point", and r, s are respectively the number of "upper" and "lower" parameters of the series. This series is defined only if $\forall i = 1, \ldots, s, \quad b_i \notin \{0, -1, -2, \ldots\}$.

The basic functions are `IsHYP`, `IsRF` and `HypEval`, `RfEval`. The last two functions are rules that transform a `HYP` (respectively a `RF`) into a sum (respectively factorial expression).

```
> IsRF( RF[4,3] );

                                   true

> IsHYP( HYP[[a,2,c],[d,-3],z] );

       warning HYP: denominator values must not be nonpositive integers

                                   true
```

The second lower parameter of this hypergeometric series is a negative integer, so its evaluation can be impossible (a division by 0 can occur under certain conditions).

```
> IsHYP( HYP[[a,b,c],[d,e],z] );

                                   true
```

```
> RfEval( RF[a,3] );

                              a (a + 1) (a + 2)

> RfEval( RF[4,3] );

                                     120

> RfEval( RF[a,b] );

                                 GAMMA(a + b)
                                 ------------
                                   GAMMA(a)

> RfEval( RF[a,-3] );

                                      1
                           -----------------------
                           (a - 3) (a - 2) (a - 1)

> H  := HYP[[a,b,c],[d,e],z]:
> He := HypEval( H );

                      infinity
                       -----                             k
                        \      RF[a, k] RF[b, k] RF[c, k] z
                 He :=   )     -----------------------------
                        /           RF[d, k] RF[e, k] k!
                       -----
                       k = 0

> RfEval( He );

      infinity
       -----                                                           k
        \      GAMMA(a + k) GAMMA(b + k) GAMMA(c + k) GAMMA(d) GAMMA(e) z
         )     -----------------------------------------------------------
        /        GAMMA(a) GAMMA(b) GAMMA(c) GAMMA(d + k) GAMMA(e + k) k!
       -----
       k = 0
```

To manipulate hypergeometric series, `AddParam` allows to add a parameter to upper and lower parameters:

```
> AddParam( HYP[[a,b],[d],1], c );

                           c, must be an integer > 0.

                           HYP[[c, a, b], [c, d], 1]
```

To simplify expressions involving hypergeometric series, you can use the function `HypSimplify`:

```
> HypSimplify( HYP[[a,b,c],[c],z]+HYP[[a,b,d],[d],z] );

                             2 HYP[[a, b], [], z]

> HypSimplify( HYP[[a,-3,v],[b,-3,c],z] );

       warning HYP: denominator values must not be nonpositive integers

                             HYP[[a, v], [b, c], z]
```

The functions HypPermLow, HypPermUp and HypPermBoth allow to permute parameters in hypergeometric series, the usage is HypPermXX(shyp,p) with:

- shyp, a hypergeometric series (in standard notation);
- p, a list of positive numbers forming a permutation.

The effect is that the new (upper or/and lower) parameter at position i is the old parameter from position p[i].

```
> HypPermLow( HYP[[a,b,d,c],[e,f,g],z], [2,3,1] );

                        HYP[[a, b, d, c], [f, g, e], z]

> HypPermBoth( HYP[[a,b,c],[d,e],z], [2,1] );

                          HYP[[a, c, b], [e, d], z]
```

The HypConverg function tells if a series is convergent (see [29, section 2.2]).

```
> HypConverg( HYP[[a,b,c],[d,4],1] );
# d+4-a-b-c must be greater than 0

                                    true
```

2.2 Differentiation

Let $D = d/dz$ be the usual differential operator with respect to z.

$$\begin{aligned}
D\ {}_rF_s\left[\begin{matrix}a_1,\ldots,a_r\\ b_1,\ldots,b_s\end{matrix};z\right] &= D\sum_{k=0}^{\infty}\frac{(a_1)_k\ldots(a_r)_k}{(b_1)_k\ldots(b_s)_k}\,\frac{z^k}{k!}\ ,\\
&= \sum_{k=1}^{\infty}\frac{(a_1)_k\ldots(a_r)_k}{(b_1)_k\ldots(b_s)_k}\,k\,\frac{z^{k-1}}{k!}\ ,\\
&= \sum_{k=0}^{\infty}\frac{(a_1)_{k+1}\ldots(a_r)_{k+1}}{(b_1)_{k+1}\ldots(b_s)_{k+1}}\,k+1\,\frac{z^k}{(k+1)!}\ ,\\
&= \frac{a_1\ \ldots\ a_r}{b_1\ \ldots\ b_s}\sum_{k=0}^{\infty}\frac{(a_1+1)_k\ldots(a_r+1)_k}{(b_1+1)_k\ldots(b_s+1)_k}\,\frac{z^k}{k!}\ ,\\
&= \frac{a_1\ \ldots\ a_r}{b_1\ \ldots\ b_s}\,{}_rF_s\left[\begin{matrix}a_1+1,\ldots,a_r+1\\ b_1+1,\ldots,b_s+1\end{matrix};z\right]\ .
\end{aligned}$$

```
> HypDiff( HYP[[a,b,c],[d,e],z], z );

             a b c HYP[[a + 1, b + 1, c + 1], [d + 1, e + 1], z]
             ---------------------------------------------------
                                    d e
```

2.3 Others hypergeometric series

Definition 5 *The hypergeometric series*

$$ {}_{r+1}F_r \left[\begin{matrix} a_1, a_2, \ldots, a_{r+1} \\ b_1, \ldots, b_r \end{matrix} ; z \right] \tag{2.6} $$

*is called **well-poised** if its parameters satisfy the relation*

$$ 1 + a_1 = a_2 + b_1 = a_3 + b_2 = \ldots = a_{r+1} + b_r \ . $$

Definition 6 *The series (2.6) is called **nearly-poised** if all but one of the above pairs of parameters (regarding* 1 *as the first lower parameter) have the same sum. We distinguish two cases:*

- *a series is called a **nearly-poised series of the first kind** if*

$$ 1 + a_1 \neq a_2 + b_1 = a_3 + b_2 = \ldots = a_{r+1} + b_r \ , $$

- *and it is called a **nearly-poised series of the second kind** if*

$$ 1 + a_1 = a_2 + b_1 = a_3 + b_2 = \ldots \neq a_{r+1} + b_r \ . $$

Definition 7 *If the series (2.6) is well-poised and* $a_2 = 1 + a_1/2$*, then it is called a **very-well-poised** series.*

This first function prints the type of a hypergeometric series:

```
> # very-well poised order
> HypType( HYP[[-n,b,1+a/2,a],[a+1-b,a/2,a+1+n],z] );

                     Very-well poised hypergeometric series

> # well-poised order
> HypType( HYP[[a,b,c,1],[a,a+1-c,a+1-b],z] );

                       Well-poised hypergeometric series

> # nearly-poised order
> HypType( HYP[[x+1,-n-1/2,-n],[x-2*n,3/2],1] );

                Nearly-poised (first kind) hypergeometric series

> # very-well poised order
> HypType( VHYP[a,[b,2+c,d],z] );

                     Very-well poised hypergeometric series
```

The following function tries to put the parameters of a hypergeometric series into the *well-poised*, *very-well-poised*, *nearly-poised* order, repectively, whatever is possible.

```
> # well-poised order
> HypOrder( HYP[[a,b,c,1],[a,a+1-c,a+1-b],z] );

                HYP[[a, b, c, 1], [a + 1 - b, a + 1 - c, a], z]

> # very-well-poised order
> HypOrder( HYP[[-n,b,1+a/2,a],[a+1-b,a/2,a+1+n],z] );

         HYP[[a, 1 + 1/2 a, - n, b], [1/2 a, a + 1 + n, a + 1 - b], z]

> # nearly-poised order
> HypOrder( HYP[[x+1,-n-1/2,-n],[x-2*n,3/2],1] );

                HYP[[- n, - n - 1/2, x + 1], [3/2, x - 2 n], 1]
```

The object `VHYP[a,[list],z]` represents a HYPergeometric series in Very-well-poised order: precisely, `VHYP[a,[b,c],z]` corresponds to `HYP[[a,1+a/2,b,c],[a/2,1+a-b,1+a-c],z]`. You can imagine what are the features of the following functions: `IsVHYP`, `VHypToHyp`, `HypToVHyp`, and `HypEval`.

```
> V := VHYP[-3,[2,4,-3],z];

                         V := VHYP[-3, [2, 4, -3], z]

# The evaluation of this hypergeometric series in very-well-poised order
# can  be  impossible  because  a lower parameter  of  the corresponding
# hypergeometric series is a negative integer.
> IsVHYP( V );

       warning VHYP: denominator values must not be nonpositive integers

                                        true
> VHypToHyp( V );

       warning VHYP: denominator values must not be nonpositive integers

                HYP[[-3, -1/2, 2, 4, -3], [-3/2, -4, -6, 1], z]

> HV := HYP[[-n,b,1+a/2,a],[a+1-b,a/2,a+1+n],z]:
> VH := HypToVHyp( HV );

                          VH := VHYP[a, [- n, b], z]

> HypEval( HV );

           infinity
            -----                                                k
             \      RF[a, k] RF[1 + 1/2 a, k] RF[- n, k] RF[b, k] z
              )     -----------------------------------------------
             /     RF[1/2 a, k] RF[a + 1 + n, k] RF[a + 1 - b, k] k!
            -----
            k = 0
```

2.4 Manipulation of hypergeometric expressions

One of the most important things in manipulating hypergeometric expressions is that you want to apply rules/functions to specific subexpressions of a (sometimes huge) expression. In this package, you can do this by using `MapList` and `MapApply`.

The function `MapList` provides a set of subexpressions (with their respective positions in the complete expression):

```
> expr1 := RF[a,3] * HYP[[a,b,c],[d,e],z] * RF[b,-2]:
> MapList( expr1 );

     {[RF[a, 3], [1]], [HYP[[a, b, c], [d, e], z], [2]], [RF[b, -2], [3]]}
```

and the function `MapApply` allow to select subexpressions that you want to apply a function (here, the first and the third subexpressions):

```
> MapApply( RfEval, {[1],[3]}, expr1 );

                  a (a + 1) (a + 2) HYP[[a, b, c], [d, e], z]
                  -------------------------------------------
                                 (b - 2) (b - 1)
```

You can also apply function which need several parameters:

```
> expr2 := HYP[[a,b],[e],1] + HYP[[a,c],[d],1]:
> MapList( expr2 );

            {[HYP[[a, b], [e], 1], [1]], [HYP[[a, c], [d], 1], [2]]}

> # apply Vandermonde's theorem (S2101) to the second term
> res := MapApply( HypSum, {[2]}, expr2, 1 );

                           c, must be an integer <= 0

                                                  RF[d - a, - c]
                  res := HYP[[a, b], [e], 1] + --------------
                                                    RF[d, - c]
```

Moreover, you can choose the depth in the analysis expression:

```
> MapList( res, 2 );

                                                                   1
 {[HYP[[a, b], [e], 1], [1]], [[RF[d - a, - c], [2, 1]], [----------, [2, 2]]]}
                                                              RF[d, - c]

> MapApply( RfEval, {[2,2]}, res );

                                        RF[d - a, - c] GAMMA(d)
                 HYP[[a, b], [e], 1] + -----------------------
                                             GAMMA(d - c)
```

2.5 Manipulation of recurrence relations

The most important tools for automatic proofs of identities are recurrence relations (see chapters 10 and 23).

Definition 8 *The sequence U **satisfy** a p-order linear recurrence relation with polynomial coefficients if there exist $a_0(n), a_1(n), \ldots, a_p(n)$ and $b(n) \in \mathbb{K}[n]$ such that:*

$$a_p(n)\, U(n+p) + \cdots + a_1(n)\, U(n+1) + a_0(n)\, U(n) = b(n) \ .$$

*The **leading** coefficient is the highest no-zero coefficient ($a_p(n)$) and the **trailing** coefficient is the lowest non-zero coefficient ($a_0(n)$).*
This recurrence equation is called:

- ***monic** if the leading coefficient is 1 ;*
- ***homogeneous** if $b(n) = 0$.*

By convention, a recurrence is therefore an expression of the form `sum(p[i]*u(n+i),i=0..p)=b` where the `p[i]`'s and `b` are polynomials in `n`. The expression `sum(p[i]*u(n+i),i=0..p)-b` has the same meaning because it is understood to be equal to zero.

Let us introduce a few functions working on these objects:

The `IsHomog` function tests if a recurrence relation is homogeneous and the `Homog` function homogenizes recurrence relations:

```
> IsHomog( 2*(2*k+1)*(k+1)*T(k+1)-3*(3*k+1)*(3*k+2)*T(k)=0, T(k) );

                                     true

> IsHomog( (n+2)*U(n+1)-n*U(n)=n, U(n) );

                                    false

> Homog( (n+2)*U(n+1)-n*U(n)=n, U(n) );

                                     2
         n (n + 1) U(n) - 2 (n + 1)  U(n + 1) + n (n + 3) U(n + 2) = 0
```

The `RecOrder`, `SubsRec` and `CheckRec` functions respectively computes the order of a recurrence, substitutes a sequence by an expression in a recurrence relation, and checks if an expression satisfy a recurrence:

```
> RecOrder( U(n+3)+U(n+2)+n*U(n-1)=3, U(n) );

                                      4

> SubsRec( (n+2)*U(n+1)-n*U(n)=n, U(n), 2^n );

                                  n      n
                                n 2  + 4 2  = n

> CheckRec( (n+2)*U(n+1)-n*U(n)=n, U(n), 2^n );
```

```
                                        false

> CheckRec( C(n)-2*C(n+1)+C(n+2)=0, C(n), n+alpha );

                                         true
```

The SimplifyRec and ShiftRec functions respectively simplifies and shifts a recurrence:

```
> rec := SimplifyRec( U(n)+U(n+2)+n*U(n)=3*n, U(n) );
  # combines equal recurrence terms and factors the polynomial coefficients

                      rec := (n + 1) U(n) + U(n + 2) = 3 n

> rec := ShiftRec( rec, U(n) );
  # default shift is one step forward

                  rec := (n + 2) U(n + 1) + U(n + 3) = 3 n + 3

> ShiftRec( rec, U(n), -3 );

                       (n - 1) U(n - 2) + U(n) = 3 n - 6
```

Chapter 3

Transformation of sums into hypergeometric notation

It is important to recognize when a given series is hypergeometric, because the general theory of hypergeometric functions is very powerful. So, what we want to do is to show how one may identify a given hypergeometric series with a particular ${}_rF_s$ with *real* or *complex* parameters (see notation 1 and definition 3).

`SumToHyp`

We describe here the process to convert summations to standard ${}_rF_s$ hypergeometric forms.

Algorithm 1 Given a series $\sum_k t_k$;

- the first step is to shift the summation index k so that the sum starts at $k=0$ with a nonzero term. The term corresponding to $k=0$ must be extracted as a common factor, so that the first term of the sum is 1;
- the second step is the simplification of the ratio t_{k+1}/t_k to bring it into the form $p(k)/q(k)$ where p,q are polynomials. **If it doesn't succeed, the series is not hypergeometric**;
- the third step gives parameters of the hypergeometric series: put the ratio $p(k)/q(k)$ in the following (completely factored) form:

$$\frac{t_{k+1}}{t_k} = \frac{p(k)}{q(k)} = \frac{(k+a_1)\,(k+a_2)\cdots(k+a_r)}{(k+b_1)\,(k+b_2)\cdots(k+b_s)\,(k+1)}\,z\ , \tag{3.1}$$

  where z is a constant.
  If the factor $k+1$ in the denominator weren't here, put it in, and compensate by inserting an extra factor of $k+1$ in the numerator.
  Notice that all the coefficients of k, in numerator and denominator, must be equal to 1;
- the final step is to identify upper, lower parameters, and evaluation point of the hypergeometric series. It gives the hypergeometric expression:

$$\sum_{k\geq 0} t_k = t_0 \times {}_rF_s\left[\begin{matrix} a_1,\ldots,a_r \\ b_1,\ldots,b_s\end{matrix};z\right]\ .$$

□

Example 2 *In the exponential series,* $e^x = \sum_{k\neq 0} \frac{x^k}{k!}$*, the initial term is 1, and the ratio of the* $(k+1)^{st}$ *term to the* k^{th} *is* $x/(k+1)$*. So,*

$$e^x = {}_0F_0\left[\begin{matrix}-\\-\end{matrix};x\right] .$$

Example 3

$$J_p(x) = \sum_{k=0}^{\infty} \frac{(-1)^k\,(\frac{x}{2})^{2\,k+p}}{k!\,(k+p)!} .$$

The ratio of consecutive terms is:

$$\begin{aligned}\frac{t_{k+1}}{t_k} &= \frac{(-1)^{k+1}\,(\frac{x}{2})^{2\,k+2+p}}{(k+1)!\,(k+p+1)!}\,\frac{k!\,(k+p)!}{(-1)^k\,(\frac{x}{2})^{2\,k+p}} ,\\ &= \frac{-(\frac{x^2}{4})}{(k+1)\,(k+p+1)} .\end{aligned}$$

We note that $t_0 = \frac{(\frac{-x}{2})^p}{p!} \neq 1$*, whereas the standardized hypergeometric series begins with a term equal to* 1*. Finally, the hypergeometric standard form of the Bessel function is:*

$$J_p(x) = \frac{(\frac{x}{2})^p}{p!}\,{}_0F_1\left[\begin{matrix}-\\p+1\end{matrix};\frac{x^2}{4}\right] .$$

Example 4

$$f(n) = \sum_{k=0}^{\infty}(-1)^k \binom{2\,n}{k}\binom{2\,k}{k}\binom{4\,n-2\,k}{2\,n-k} .$$

The ratio of consecutive terms is:

$$\begin{aligned}\frac{t_{k+1}}{t_k} &= \frac{(-1)^{k+1}\,\binom{2\,n}{k+1}\,\binom{2\,k+2}{k+1}\,\binom{4\,n-2\,k-2}{2\,n-k-1}}{(-1)^k\,\binom{2\,n}{k}\,\binom{2\,k}{k}\,\binom{4\,n-2\,k}{2\,n-k}} ,\\ &= \frac{(k+\frac{1}{2})\,(k-2\,n)\,(k-2\,n)}{(k+1)\,(k+1)\,(k-2\,n+\frac{1}{2})} .\end{aligned}$$

The first term of the sum is not 1*, but is instead* $\binom{4\,n}{2\,n}$*, hence we determined completely the initial sum:*

$$f(n) = \binom{4\,n}{2\,n}\,{}_3F_2\left[\begin{matrix}-2n,-2n,\frac{1}{2}\\1,-2\,n+\frac{1}{2}\end{matrix};1\right] .$$

The function `SumToHyp` transforms a series `Sum(expr, k=lower..upper)` into hypergeometric notation `HYP[[a,b,...],[c,...],z]` if possible. The upper bound must be ∞.

It also applies the `HypOrder` function to put parameters of the hypergeometric series in well-poised, very-well-poised or nearly-poised order (see definitions 5, 7 and 6).

```
> SumToHyp( Sum(x^k/k!,k) );

                         HYP[[], [], x]

> SumToHyp( Sum(1/(2*k-1)/(2*k+1)!,k=0..infinity) );
```

```
                        - HYP[[-1/2], [3/2, 1/2], 1/4]

> # Hypergeometric series with complex parameters
> SumToHyp( Sum(4*k^2+1,k) );

             HYP[[1 - 1/2 I, 1 + 1/2 I, 1], [- 1/2 I, 1/2 I], 1]

> SumToHyp( Sum(binomial(M,k)*binomial(N,R-k),k) );

                binomial(N, R) HYP[[- M, - R], [N - R + 1], 1]

> SumToHyp( Sum(2^k/(k!)^2,k=0..infinity) );

                               HYP[[], [1], 2]

> s1 := Sum( RF[a,k]*RF[b,k]*RF[c,k]*RF[d,k]/RF[e,k]/RF[f,k]*(z^2)^k/k!, k );

                    -----                                       2 k
                      \    RF[a, k] RF[b, k] RF[c, k] RF[d, k] (z )
             s1 :=     )   -----------------------------------------
                      /                RF[e, k] RF[f, k] k!
                    -----
                      k

> SumToHyp( s1 );

                                                        2
                        HYP[[d, c, b, a], [e, f], z ]

> `HYPERG/Verbose` := 1:
> s2 := Sum(RF[k+2,1]*RF[-n,k+1]*RF[a,k+1]/RF[b,k+1]/RF[c,k+1]/RF[1,k+1]*z^k,k=0..infinity):
> SumToHyp( s2 );

 Ratio of two consecutive terms: -(k+3)*z*(n-k-1)*(a+k+1)/(k+2)^2/(b+k+1)/(c+k+1)

 Free ok index summation: -2*n*a/b/c

 Upper parameters: [1, 3, -n+1, a+1]

 Lower parameters: [2, 2, b+1, c+1]

             n a HYP[[3, - n + 1, a + 1, 1], [2, 2, c + 1, b + 1], z]
         - 2 --------------------------------------------------------
                                          b c

> SumToHyp( Sum(k^2+k!,k) );

 Ratio of two consecutive terms: (k+1)*(k+1+GAMMA(k+1))/k/(k+GAMMA(k))

 Free of index summation: 1

Error, (in SumToHyp) Not a hypergeometric series
```

Chapter 4

Rules for manipulating factorial expressions

`BaseSplit, Ext1, Ext2, Inv, Linear1, Linear2, Neg1, Neg2, Trans, Split`

These rules allow you to handle any factorial expression (Gamma function and Rising Factorial):

- **BaseSplit**:

$$
\begin{aligned}
(a)_n &= m^n \prod_{k=0}^{m-1} \left(\frac{a+k}{m}\right)_{n/m} , \\
\Gamma(a) &= m^{a-1/2} \, (2\,\pi)^{(1-m)/2} \prod_{k=0}^{m-1} \Gamma((a+k)/m) \; .
\end{aligned}
$$

- **Ext1**:

$$
(a)_n = \begin{cases} \dfrac{\Gamma(a+n)}{\Gamma(a)} & \text{if } m \text{ is infinity,} \\ \dfrac{(a)_{m+n}}{(a+n)_m} & \text{if } m \text{ is an integer.} \end{cases}
$$

- **Ext2**:

$$
\begin{aligned}
(a)_n &= \frac{(a-m)_{m+n}}{(a-m)_m} , \\
\Gamma(a) &= \Gamma(a-m)\,(a-m)_m \; ,
\end{aligned}
$$

  where m is an integer.

- **Inv**: Reflection formula for the Γ-function

$$
\Gamma(z) = \frac{\pi}{\sin(\pi\, z)} \, \frac{1}{\Gamma(1-z)} \; ,
$$

- **Linear1**:

$$\begin{aligned} (a)_n &= a\,(a+1)_{n-1}\ , \\ \Gamma(a) &= (a-1)\,\Gamma(a-1)\ . \end{aligned}$$

- **Linear2**:

$$(a)_n = (a+n-1)\,(a)_{n-1}\ ,$$

- **Neg1**:

$$(a)_n = \frac{1}{(a+n)_{-n}}\ ,$$

- **Neg2**:

$$(a)_n = \frac{(-1)^n}{(1-a)_{-n}}\ ,$$

- **Trans**:

$$(a)_n = (-1)^n\,(1-n-a)_n\ ,$$

- **Split**:

$$\begin{aligned} (a)_n &= (a)_m\,(a+m)_{n-m}\ , \\ \Gamma(a) &= \frac{\Gamma(a+m)}{(a)_m}\ , \end{aligned}$$

  where m is an integer.

```
> BaseSplit( GAMMA(2*c), 3 );

           (2 c - 1/2)
          3            GAMMA(2/3 c) GAMMA(2/3 c + 1/3) GAMMA(2/3 c + 2/3)
      1/2 ---------------------------------------------------------------
                                      Pi

> rf := RF[a,n]:
> Ext1( rf, 1/2 );
Error, (in HYPERG/Ext1/heart) The second argument is not an integer, 1/2
> Ext1( rf, 3 );

                                  RF[a, n + 3]
                                  ------------
                                  RF[a + n, 3]

> Neg1( 3/RF[a,n] );

                                3 RF[a + n, - n]

> Split( a*GAMMA(a)*RF[a,b]*GAMMA(b), m );

                  a GAMMA(a + m) GAMMA(b + m) RF[a + m, b - m]
                  --------------------------------------------
                                  RF[b, m]
```

Chapter 5

Summations for hypergeometric series

`HypSum, HypSumList, HypSumPrint`

This package contains several summation formulas:

```
- S1001,
- S2101, S2103, S2104, S2105, S2106, S2131, S2132,
- S3201, S3202, S3204, S3231, S3232, S3233, S3234, S3235,
  S3261, S3291,
- S4306, S4307, S4331, S4332,
- S5431, S5432,
- S6531, S6532,
- S7631, S7632, S7691.
```

The numbering of each formula is `S<up><dw><nb>` where:

- `up` is the number of upper parameters ;
- `dw` is the number of lower parameters ;
- `nb` allows to distinguish the formulas applying to hypergeometric series with equal numbers of upper and lower parameters:
  - `nb` is within the range 01-60 if the summation is a one-term summation ;
  - `nb` is within the range 61-120 if the summation is a two- or more-term summation.

The usage of the function `HypSum` is: `HypSum(shyp,nb)` with:

- `shyp`, a hypergeometric series (in standard notation) ;
- `nb`, the numbering of the formula (see above) ;
- the fact that the parameters of hypergeometric series are symmetric implies that a given formula may have several ways of applying, according to the position of the parameters. It is possible to modify the position of the parameters with the functions `HypPermUp` and `HypPermLow` (see chapter **2**). [Let us remark that some formulas are totally symmetric and the position of parameters doesn't modify the result].

Last, most of the formulas need conditions on parameters. If this is the case, those conditions are displayed before the result of the summation formula. It is up to the user to check that the summation is valid.

This is the list of all summation formulas:

5.1 ${}_1F_0$ hypergeometric series

❑ **S1001** ([29], Appendix(III.1)) **The binomial theorem**

$$ {}_1F_0\left[\begin{matrix} a \\ - \end{matrix}; z\right] \longrightarrow (1-z)^{-a} $$

5.2 ${}_2F_1$ hypergeometric series

❑ **S2101** ([29], (1.7.7), Appendix(III.4)) **Vandermonde's theorem**

$$ {}_2F_1\left[\begin{matrix} a, -n \\ c \end{matrix}; 1\right] \longrightarrow \frac{(c-a)_n}{(c)_n} $$

where n is a nonnegative integer.

❑ **S2103** ([29], (1.7.6), Appendix(III.3)) **Gauss's theorem**

$$ {}_2F_1\left[\begin{matrix} a, b \\ c \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(c)\,\Gamma(c-a-b)}{\Gamma(c-a)\,\Gamma(-b+c)} $$

❑ **S2104** ([29], (1.7.1.6) corrected, (2.3.2.9), Appendix(III.5)) **Kummer's theorem**

$$ {}_2F_1\left[\begin{matrix} a, b \\ 1+a-b \end{matrix}; -1\right] \longrightarrow \frac{\Gamma(1+\frac{a}{2})\,\Gamma(1+a-b)}{\Gamma(1+a)\,\Gamma(1+\frac{a}{2}-b)} $$

❑ **S2105** ([7], Ex 1.6(i), $q \to 1$)

$$ {}_2F_1\left[\begin{matrix} \frac{-n}{2}, \frac{1-n}{2} \\ \frac{1}{2}+b \end{matrix}; 1\right] \longrightarrow \frac{2^n\,(b)_n}{(2\,b)_n} $$

where n is a nonnegative integer.

❑ **S2106** ([29], (1.5.21))

$$ {}_2F_1\left[\begin{matrix} 2\,a, a+1 \\ a \end{matrix}; z\right] \longrightarrow \frac{1+z}{(1-z)^{2a+1}} $$

where the input series is in very-well-poised order.

❑ **S2131** ([29], (1.7.1.8), Appendix(III.7)) **Bailey's theorem**

$$ {}_2F_1\left[\begin{matrix} a, 1-a \\ b \end{matrix}; \frac{1}{2}\right] \longrightarrow \frac{\Gamma(\frac{b}{2})\,\Gamma(\frac{1+b}{2})}{\Gamma(\frac{a+b}{2})\,\Gamma(\frac{1-a+b}{2})} $$

❑ **S2132** ([29], (1.7.1.9), Appendix(III.6)) **Gauss's second theorem**

$$ {}_2F_1\left[\begin{matrix} 2\,a, 2\,b \\ \frac{1}{2}+a+b \end{matrix}; \frac{1}{2}\right] \longrightarrow \frac{\Gamma(\frac{1}{2})\,\Gamma(\frac{1}{2}+a+b)}{\Gamma(\frac{1}{2}+a)\,\Gamma(\frac{1}{2}+b)} $$

5.3 ${}_3F_2$ hypergeometric series

❑ **S3201** ([29], (2.3.1.3), Appendix(III.2)) **Saalschütz's theorem**

$$ {}_3F_2\left[\begin{matrix} a, b, -n \\ c, 1+a+b-c-n \end{matrix}; 1\right] \longrightarrow \frac{(c-a)_n\,(c-b)_n}{(c)_n\,(c-a-b)_n} $$

where n is a nonnegative integer.

❑ **S3202** ([29], (2.3.3.5), Appendix(III.8) terminated in the first variable)

$$ {}_3F_2\left[\begin{matrix} -2\,n, b, c \\ 1-b-2\,n, 1-c-2\,n \end{matrix}; 1\right] \longrightarrow \frac{(1)_{2\,n}\,(b)_n\,(c)_n\,(b+c)_{2\,n}}{(1)_n\,(b)_{2\,n}\,(c)_{2\,n}\,(b+c)_n} $$

where n is a nonnegative integer.

❑ **S3204** ([7], Ex. 3.9, $q \to 1$)

$$ {}_3F_2\left[\begin{matrix} a, 1+\frac{\lambda}{2}, b \\ \frac{\lambda}{2}, 1-b+\lambda \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(\lambda)\,\Gamma(1-a+\lambda)\,\Gamma(-a-2\,b+\lambda)\,\Gamma(1-b+\lambda)}{\Gamma(1+\lambda)\,\Gamma(-a+\lambda)\,\Gamma(-2\,b+\lambda)\,\Gamma(1-a-b+\lambda)} $$

❑ **S3231** ([29], (2.3.3.5), Appendix(III.8)) **Dixon's theorem**

$$ {}_3F_2\left[\begin{matrix} a, b, c \\ 1+a-b, 1+a-c \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(1+a-b)\,\Gamma(1+a-c)\,\Gamma(1+\frac{a}{2})\,\Gamma(1+\frac{a}{2}-b-c)}{\Gamma(1+a)\,\Gamma(1+\frac{a}{2}-b)\,\Gamma(1+\frac{a}{2}-c)\,\Gamma(1+a-b-c)} $$

❑ **S3232** ([29], (2.3.3.6), Appendix(III.9)) **Dixon's theorem**

$$ {}_3F_2\left[\begin{matrix} a, b, -n \\ 1+a-b, 1+a+n \end{matrix}; 1\right] \longrightarrow \frac{(1+a)_n\,(1+\frac{a}{2}-b)_n}{(1+\frac{a}{2})_n\,(1+a-b)_n} $$

where n is a nonnegative integer.

❑ **S3233** ([29], (2.3.3.13), Appendix(III.23)) **Watson's theorem**

$$ {}_3F_2\left[\begin{matrix} a, b, c \\ \frac{1+a+b}{2}, 2\,c \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(\frac{1}{2})\,\Gamma(\frac{1}{2}+c)\,\Gamma(\frac{1+a+b}{2})\,\Gamma(\frac{1-a-b}{2}+c)}{\Gamma(\frac{1+a}{2})\,\Gamma(\frac{1+b}{2})\,\Gamma(\frac{1-a}{2}+c)\,\Gamma(\frac{1-b}{2}+c)} $$

❑ **S3234** ([29], (2.3.3.14), Appendix(III.24) corrected) **Whipple's theorem**

$$ {}_3F_2\left[\begin{matrix} a, b, c \\ d, e \end{matrix}; 1\right] \longrightarrow \pi\, 2^{1-2\,c}\,\frac{\Gamma(d)\,\Gamma(e)}{\Gamma(\frac{a+e}{2})\,\Gamma(\frac{a+d}{2})\,\Gamma(\frac{b+e}{2})\,\Gamma(\frac{b+d}{2})} $$

provided that $a+b=1$ and that $d+e=1+2\,c$.

❑ **S3235** ([29], (2.4.2.5), Appendix(III.16))

$$ {}_3F_2\left[\begin{matrix} a, b, -n \\ 1+a-b, 1+2\,b-n \end{matrix}; 1\right] \longrightarrow \frac{(1+\frac{a}{2}-b)_n\,(a-2\,b)_n\,(-b)_n}{(\frac{a}{2}-b)_n\,(1+a-b)_n\,(-2\,b)_n} $$

where n is a nonnegative integer.

❑ **S3261** ([29], (2.4.4.4), Appendix(III.31)) **Saalschütz's theorem, in the non-terminating form**

$$ {}_3F_2\left[\begin{matrix} a, b, c \\ d, 1+a+b+c-d \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(1+a-d)\,\Gamma(1+b-d)\,\Gamma(1+c-d)\,\Gamma(1+a+b+c-d)}{\Gamma(1-d)\,\Gamma(1+b+c-d)\,\Gamma(1+a+c-d)\,\Gamma(1+a+b-d)} $$

$$ -\frac{\Gamma(d-1)\,\Gamma(1+a-d)\,\Gamma(1+b-d)\,\Gamma(1+c-d)\,\Gamma(1+a+b+c-d)}{\Gamma(1+d)\,\Gamma(a)\,\Gamma(b)\,\Gamma(c)\,\Gamma(2+a+b+c-2*d)}\,{}_3F_2\left[\begin{matrix} 1+a-d, 1+b-d, 1+c-d \\ 2-d, 2+a+b+c-2\,d \end{matrix}; 1\right] $$

❑ **S3291** ([7], Ex. 2.9, $q \to 1$)

$$
{}_3F_2\left[\begin{matrix} a, b, 1+a-2\,c \\ 1+a-c, 2\,b \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(1/2)\,\Gamma(\frac{1+a}{2}-b)\,\Gamma(1+a-c)\,\Gamma(1+\frac{a}{2}-b-c)}{\Gamma(\frac{1+a}{2})\,\Gamma(\frac{1}{2}-b)\,\Gamma(1+\frac{a}{2}-c)\,\Gamma(1+a-b-c)}
$$

$$
-{}_3F_2\left[\begin{matrix} 1-b, 1+a-2\,b, 2+a-2\,b-2\,c \\ 2+a-2\,b-c, 2-2\,b \end{matrix}; 1\right]
$$

$$
\times \frac{\Gamma(\frac{1+a}{2}-b)\,\Gamma(1+\frac{a}{2}-b)\,\Gamma(-\frac{1}{2}+b)\,\Gamma(1+a-c)\,\Gamma(1+\frac{a}{2}-b-c)\,\Gamma(\frac{3}{2}+\frac{a}{2}-b-c)}{\Gamma(\frac{1+a}{2})\,\Gamma(\frac{a}{2})\,\Gamma(\frac{1}{2}-b)\,\Gamma(\frac{1+a}{2}-c)\,\Gamma(1+\frac{a}{2}-c)\,\Gamma(2+a-2\,b-c)}
$$

5.4 $_4F_3$ hypergeometric series

❑ **S4306** ([29], (2.4.2.6), Appendix(III.17))

$$
{}_4F_3\left[\begin{matrix} a, 1+\frac{a}{2}, b, -n \\ \frac{a}{2}, 1+a-b, 1+2\,b-n \end{matrix}; 1\right] \longrightarrow \frac{(a-2\,b)_n\,(-b)_n}{(1+a-b)_n\,(-2\,b)_n}
$$

where n is a nonnegative integer.

❑ **S4307** ([29], (2.3.4.6), Appendix(III.10)) **Dixon's theorem**

$$
{}_4F_3\left[\begin{matrix} a, 1+\frac{a}{2}, b, c \\ \frac{a}{2}, 1+a-b, 1+a-c \end{matrix}; -1\right] \longrightarrow \frac{\Gamma(1+a-b)\,\Gamma(1+a-c)}{\Gamma(1+a)\,\Gamma(1+a-b-c)}
$$

where the input series is in very-well-poised order.

❑ **S4331** ([29], Appendix(III.22))

$$
{}_4F_3\left[\begin{matrix} a, 1+\frac{a}{2}, b, c \\ \frac{a}{2}, 1+a-b, 1+a-c \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(1+a-b)\,\Gamma(1+a-c)\,\Gamma(\frac{1}{2}+\frac{a}{2})\,\Gamma(\frac{1}{2}+\frac{a}{2}-b-c)}{\Gamma(1+a)\,\Gamma(1+a-b-c)\,\Gamma(\frac{1}{2}+\frac{a}{2}-b)\,\Gamma(\frac{1}{2}+\frac{a}{2}-c)}
$$

where the input series is in very-well-poised order.

❑ **S4332** ([29], (2.4.2.7), Appendix(III.18))

$$
{}_4F_3\left[\begin{matrix} a, 1+\frac{a}{2}, b, -n \\ \frac{a}{2}, 1+a-b, 2+2\,b-n \end{matrix}; 1\right] \longrightarrow \frac{(\frac{1}{2}+\frac{a}{2}-b)_n\,(-1+a-2\,b)_n\,(-1-b)_n}{(-\frac{1}{2}+\frac{a}{2}-b)_n\,(1+a-b)_n\,(-1-2\,b)_n}
$$

where n is a nonnegative integer.

5.5 $_5F_4$ hypergeometric series

❑ **S5431** ([29], (2.3.4.5), Appendix(III.12))

$$
{}_5F_4\left[\begin{matrix} a, \frac{a}{2}+1, b, c, d \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(1+a-b)\,\Gamma(1+a-c)\,\Gamma(1+a-d)\,\Gamma(1+a-b-c-d)}{\Gamma(a+1)\,\Gamma(1+a-b-c)\,\Gamma(1+a-b-d)\,\Gamma(1+a-c-d)}
$$

where the input series is in very-well-poised order.

❑ **S5432** ([29], (2.3.4.6), Appendix(III.13))

$$
{}_5F_4\left[\begin{matrix} a, \frac{a}{2}+1, b, c, -n \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a+n \end{matrix}; 1\right] \longrightarrow \frac{(a+1)_n\,(1+a-b-c)_n}{(1+a-b)_n\,(1+a-c)_n}
$$

where the input series is in very-well-poised order and n is a nonnegative integer,.

5.6 ${}_6F_5$ hypergeometric series

❑ **S6531**

$$
{}_6F_5\left[\begin{matrix} -\frac{1}{2}+\frac{a}{2}+\frac{b}{2}+c, \frac{3}{4}+\frac{a}{4}+\frac{b}{4}+\frac{c}{2}, a, b, c, \frac{1}{2}+\frac{a}{2}+\frac{b}{2}-c \\ -\frac{1}{4}+\frac{a}{4}+\frac{b}{4}+\frac{c}{2}, \frac{1}{2}-\frac{a}{2}+\frac{b}{2}+c, \frac{1}{2}+\frac{a}{2}-\frac{b}{2}+c, \frac{1}{2}+\frac{a}{2}+\frac{b}{2}, 2\,c \end{matrix}; -1\right]
$$
$$
\longrightarrow \frac{\Gamma(\frac{1}{2})\,\Gamma(\frac{1}{2}+\frac{a}{2}+\frac{b}{2})\,\Gamma(\frac{1}{2}+c)\,\Gamma(\frac{1}{2}+\frac{a}{2}-\frac{b}{2}+c)\,\Gamma(\frac{1}{2}-\frac{a}{2}+\frac{b}{2}+c)}{\Gamma(\frac{1}{2}+\frac{a}{2})\,\Gamma(\frac{1}{2}+\frac{b}{2})\,\Gamma(\frac{1}{2}-\frac{a}{2}+c)\,\Gamma(\frac{1}{2}-\frac{b}{2}+c)\,\Gamma(\frac{1}{2}+\frac{a}{2}+\frac{b}{2}+c)}
$$

where the input series is in very-well-poised order.

❑ **S6532**

$$
{}_6F_5\left[\begin{matrix} a, \frac{a}{2}+1, b, 1-b, d, 1-d \\ \frac{a}{2}, 1+a-b, a+b, 1+a-d, a+d \end{matrix}; -1\right]
$$
$$
\longrightarrow 2^{2\,b}\,\frac{\Gamma(1+a-b)\,\Gamma(a+b)\,\Gamma(1+a-d)\,\Gamma(1+\frac{a}{2}+\frac{b}{2}-\frac{d}{2})\,\Gamma(\frac{1}{2}+\frac{a}{2}+\frac{b}{2}+\frac{d}{2})\,\Gamma(a+d)}{\Gamma(a)\,\Gamma(1+a)\,\Gamma(1+a+b-d)\,\Gamma(1+\frac{a}{2}-\frac{b}{2}-\frac{d}{2})\,\Gamma(\frac{1}{2}+\frac{a}{2}-\frac{b}{2}+\frac{d}{2})\,\Gamma(a+b+d)}
$$

where the input series is in very-well-poised order.

5.7 ${}_7F_6$ hypergeometric series

❑ **S7631** ([29], (2.3.4.4), Appendix(III.14)) **Dougall's theorem**

$$
{}_7F_6\left[\begin{matrix} a, \frac{a}{2}+1, b, c, d, 1+2\,a-b-c-d+n, -n \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, -a+b+c+d-n, 1+a+n \end{matrix}; 1\right]
$$
$$
\longrightarrow \frac{(a+1)_n\,(1+a-b-c)_n\,(1+a-b-d)_n\,(1+a-c-d)_n}{(1+a-b)_n\,(1+a-c)_n\,(1+a-d)_n\,(1+a-b-c-d)_n}
$$

where the input series is in very-well-poised order and n is a nonnegative integer.

❑ **S7632** ([29], (2.4.1.5), Appendix(III.19))

$$
{}_7F_6\left[\begin{matrix} a, \frac{a}{2}+1, b, \frac{1}{2}+b, a-2\,b, 1+2\,a-2\,b+n, -n \\ \frac{a}{2}, 1+a-b, \frac{1}{2}+a-b, 1+2\,b, -a+2\,b-n, 1+a+n \end{matrix}; 1\right]
$$
$$
\longrightarrow \frac{(1+a)_n\,(1+2\,a-4\,b)_n}{(1+a-2\,b)_n\,(1+2\,a-2\,b)_n}
$$

where the input series is in very-well-poised order, and n is a nonnegative integer.

❑ **S7691** ([29], (4.2.3.8), Appendix(III.32))

$$
{}_7F_6\left[\begin{matrix} a, \frac{a}{2}+1, b, c, d, e, 1+2\,a-b-c-d-e \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, -a+b+c+d+e \end{matrix}; 1\right]
$$
$$
\begin{aligned}
\longrightarrow\ & \frac{\Gamma(1+a-c)\,\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(-a+b+c+d+e)}{\Gamma(1+a)\,\Gamma(-a+b)\,\Gamma(1+a-c-d)\,\Gamma(1+a-c-e)} \\
& \times \frac{\Gamma(-a+b+c)\,\Gamma(-a+b+d)\,\Gamma(-a+b+e)\,\Gamma(1+a-c-d-e)}{\Gamma(-a+b+d+e)\,\Gamma(1+a-d-e)\,\Gamma(-a+b+c+e)\,\Gamma(-a+b+c+d)} \\
-\ & \frac{\Gamma(a-b)\,\Gamma(1+a-c)\,\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(-a+b+c+d+e)}{\Gamma(1+a)\,\Gamma(c)\,\Gamma(d)\,\Gamma(e)\,\Gamma(1+2\,a-b-c-d-e)} \\
& \times \frac{\Gamma(-a+b+c)\,\Gamma(-a+b+d)\,\Gamma(-a+b+e)\,\Gamma(1+a-c-d-e)\,\Gamma(1-a+2\,b)}{\Gamma(-a+b)\,\Gamma(1+b-c)\,\Gamma(1+b-d)\,\Gamma(1+b-e)\,\Gamma(-2\,a+2\,b+c+d+e)} \\
& \times {}_7V_6\left[-a+2\,b; b, -a+b+c, -a+b+d, -a+b+e, 1+a-c-d-e; 1\right]
\end{aligned}
$$

where the input series is in very-well-poised order.

```
> # S1001
> HypSum( HYP[[a],[],z], 1 );

                                        (- a)
                                 (1 - z)

> # S2101
> HypSum( HYP[[a,-n],[c],z], 1 );

                             z, must be equal to, 1

                          - n, must be an integer <= 0

                                 RF[c - a, n]
                                 ------------
                                   RF[c, n]

> # S2101
> HypSum( HYP[[b,a],[c],z], 1 );

                             z, must be equal to, 1

                           a, must be an integer <= 0

                                RF[c - b, - a]
                                --------------
                                  RF[c, - a]
```

To make your work easier, the function HypSumList returns a list of summation formulas (S1001-S7691) which can be applied either directly or after permutation of parameters.

```
> HypSumList( HYP[[a,b],[d],1] );

  [[S2101], [HypPermUp, [2, 1], S2101], [S2103], [HypPermUp, [2, 1], S2103],

   [S2105], [HypPermUp, [2, 1], S2105]]
```

Last, the function HypSumPrint allows to display all the formulas on the screen (as identities) and to work with them. The usage is: HypSumPrint(shyp,nb) where shyp is a hypergeometric series and nb is the numbering of the formula.

```
> HypSumPrint( HYP[[a,b],[c],z], 1 );

                             z, must be equal to, 1

                           b, must be an integer <= 0

                                            RF[c - a, - b]
                    HYP[[a, b], [c], z] = --------------
                                              RF[c, - b]
```

```
> HypSumPrint( HYP[[a,b,c],[4,e],1], 31 );

                         1 + a - b, must be equal to, 4

                         1 + a - c, must be equal to, e

   HYP[[a, b, c], [4, e], 1] =

       GAMMA(1 + a - b) GAMMA(1 + a - c) GAMMA(1/2 a + 1, 1 + 1/2 a - b - c)
    ---------------------------------------------------------------------------
    GAMMA(a + 1) GAMMA(1 + 1/2 a - b) GAMMA(1 + 1/2 a - c) GAMMA(1 + a - b - c)
```

Chapter 6

Transformations for hypergeometric series

`HypTransf, HypTransfList, HypTransfPrint`

This package contains several transformation formulas:

```
- T1101,
- T2103, T2104, T2106, T2107, T2110, T2111, T2112, T2131, T2133, T2134, T2135,
  T2136, T2137, T2138, T2139, T2140, T2141,
  T2163, T2191, T2192,
- T3204, T3205, T3206, T3207, T3231, T3234, T3235, T3236, T3237, T3239, T3240,
  T3261, T3262, T3263, T3264, T3267, T3268,
- T4301, T4302, T4303, T4304, T4306, T4309, T4310, T4312, T4331, T4332,
- T4362, T4391,
- T5401, T5402, T5403,
- T6501, T6531, T6532, T6533,
- T7631, T7632, T7633, T7634, T7635, T7636, T7637,
- T7691, T7692, T7693, T7694,
- T8731, T8732.
```

and several others will be available soon.

The comments for the numbering of the summation formulas (see page 32) also apply here for transformation formulas.
The usage of the function `HypTransf` is: `HypTransf(shyp,nb)` with:

- `shyp`, a hypergeometric series (in standard notation) ;
- `nb`, the numbering of the formula ;
- the fact that the parameters of hypergeometric series are symmetric implies that a given formula may have several ways of applying, according to the position of the parameters. It is possible to modify the position of the parameters with the functions `HypPermUp` and `HypPermLow` (see chapter **2**). [Let us remark that some formulas are totally symmetric and the position of parameters doesn't modify the result].

Last, most of the formulas need conditions on parameters. If this is the case, those conditions are displayed before the result of the transformation formula. It is up to the user to check that the summation is valid.

This is the list of all transformation formulas:

6.1 ${}_1F_1$ hypergeometric series

❑ **T1101** ([11], (10.7)) **Kummer's transformation**

$$ {}_1F_1\left[\begin{matrix}a\\b\end{matrix};-z\right] \longrightarrow e^{-z}\,{}_1F_1\left[\begin{matrix}b-a\\b\end{matrix};z\right] $$

6.2 ${}_2F_1$ hypergeometric series

❑ **T2103** ([29], (1.3.15))

$$ {}_2F_1\left[\begin{matrix}a,b\\c\end{matrix};z\right] \longrightarrow (1-z)^{c-a-b}\,{}_2F_1\left[\begin{matrix}c-a,c-b\\c\end{matrix};z\right] $$

❑ **T2104** ([29], (1.7.1.3))

$$ {}_2F_1\left[\begin{matrix}a,b\\c\end{matrix};z\right] \longrightarrow (1-z)^{-a}\,{}_2F_1\left[\begin{matrix}a,c-b\\c\end{matrix};-\frac{z}{1-z}\right] $$

❑ **T2106** ([29], (1.7.1.3))

$$ {}_2F_1\left[\begin{matrix}a,-n\\c\end{matrix};z\right] \longrightarrow z^n\,\frac{(c-a)_n}{(c)_n}\,{}_2F_1\left[\begin{matrix}-n,1-c-n\\1+a-c-n\end{matrix};-\frac{1-z}{z}\right] $$

where n is a nonnegative integer.

❑ **T2107** ([29], (1.8.10))

$$ {}_2F_1\left[\begin{matrix}a,-n\\c\end{matrix};z\right] \longrightarrow \frac{(c-a)_n}{(c)_n}\,{}_2F_1\left[\begin{matrix}-n,a\\1+a-c-n\end{matrix};1-z\right] $$

where n is a nonnegative integer.

❑ **T2110**

$$ {}_2F_1\left[\begin{matrix}a,b\\1+a-b\end{matrix};z\right] \longrightarrow (1+z)^{-a}\,{}_2F_1\left[\begin{matrix}\frac{a}{2},\frac{1}{2}+\frac{a}{2}\\1+a-b\end{matrix};\frac{4\,z}{(1+z)^2}\right] $$

❑ **T2111** ([29], (2.3.2.1)) **Kummer's quadratic transformation**

$$ {}_2F_1\left[\begin{matrix}a,b\\1+a-b\end{matrix};z\right] \longrightarrow (1-z)^{-a}\,{}_2F_1\left[\begin{matrix}\frac{a}{2},\frac{1}{2}+\frac{a}{2}+b\\1+a-b\end{matrix};\frac{-4\,z}{(1-z)^2}\right] $$

❑ **T2112**

$$ {}_2F_1\left[\begin{matrix}a,\frac{1}{2}+a\\\frac{1}{2}+b\end{matrix};z^2\right] \longrightarrow (1-z)^{-2\,a}\,{}_2F_1\left[\begin{matrix}2\,a,b\\2\,b\end{matrix};\frac{2\,z}{-1+z}\right] $$

❑ **T2131** ([29], (1.8.10))

$$ {}_2F_1\left[\begin{matrix}a,-n\\c\end{matrix};z\right] \longrightarrow (1-z)^n\,\frac{(a)_n}{(c)_n}\,{}_2F_1\left[\begin{matrix}-n,c-a\\1-a-n\end{matrix};\frac{1}{1-z}\right] $$

where n is a nonnegative integer.

❑ **T2133**

$${}_2F_1\left[\begin{matrix}a,b\\ \frac{1}{2}+a+b\end{matrix};\frac{z^2}{-1+z^2}\right]\longrightarrow\frac{(1-z)^a}{(1+z)^a}\,{}_2F_1\left[\begin{matrix}2\,a,a+b\\ 2\,a+2\,b\end{matrix};\frac{2\,z}{1+z}\right]$$

❑ **T2134**

$${}_2F_1\left[\begin{matrix}a,b\\ \frac{1}{2}+\frac{a}{2}+\frac{b}{2}\end{matrix};z\right]\longrightarrow{}_2F_1\left[\begin{matrix}\frac{a}{2},\frac{b}{2}\\ \frac{1}{2}+\frac{a}{2}+\frac{b}{2}\end{matrix};4\,z\,(1-z)\right]$$

❑ **T2135**

$${}_2F_1\left[\begin{matrix}a,b\\ \frac{1}{2}+a+b\end{matrix};z\right]\longrightarrow{}_2F_1\left[\begin{matrix}2\,a,2\,b\\ \frac{1}{2}+a+b\end{matrix};\frac{1-\sqrt{1-z}}{2}\right]$$

❑ **T2136**

$${}_2F_1\left[\begin{matrix}a,b\\ 2\,b\end{matrix};z\right]\longrightarrow\left(\frac{2}{2-z}\right)^a\,{}_2F_1\left[\begin{matrix}\frac{a}{2},\frac{1}{2}+\frac{a}{2}\\ \frac{1}{2}+b\end{matrix};\frac{z^2}{(z-2)^2}\right]$$

❑ **T2137**

$${}_2F_1\left[\begin{matrix}a,b\\ 2\,b\end{matrix};z\right]\longrightarrow(1-z)^{-\frac{a}{2}}\,{}_2F_1\left[\begin{matrix}\frac{a}{2},-\frac{a}{2}+b\\ \frac{1}{2}+b\end{matrix};\frac{z^2}{4\,(z-1)}\right]$$

❑ **T2138**

$${}_2F_1\left[\begin{matrix}a,1-a\\ c\end{matrix};z\right]\longrightarrow(1-z)^{c-1}\,{}_2F_1\left[\begin{matrix}\frac{c}{2}-\frac{a}{2},\frac{a}{2}+\frac{c}{2}-\frac{1}{2}\\ c\end{matrix};4\,z\,(1-z)\right]$$

❑ **T2139**

$${}_2F_1\left[\begin{matrix}c,-\frac{1}{2}+a-c\\ a\end{matrix};z\right]\longrightarrow\left(\frac{1+\sqrt{1-z}}{2}\right)^{1-a}\,{}_2F_1\left[\begin{matrix}a-2\,c,1-a+2\,c\\ a\end{matrix};\frac{1-\sqrt{1-z}}{2}\right]$$

❑ **T2140**

$${}_2F_1\left[\begin{matrix}a,\frac{1}{2}+a\\ b\end{matrix};\frac{4\,z}{(1+z)^2}\right]\longrightarrow(1+z)^{2\,a}\,{}_2F_1\left[\begin{matrix}2\,a,1+2\,a-b\\ b\end{matrix};z\right]$$

❑ **T2141**

$${}_2F_1\left[\begin{matrix}a,\frac{1}{2}+a\\ b\end{matrix};\frac{4\,z}{(1+z)^2}\right]\longrightarrow\frac{(1+z)^{2\,a}}{1-z}\,{}_2F_1\left[\begin{matrix}-1+2\,a,\frac{1}{2}+a,2\,a-c\\ -\frac{1}{2}+a,c\end{matrix};z\right]$$

❑ **T2163** ([7], Ex 3.8, $q\to 1$)

$${}_2F_1\left[\begin{matrix}a,b\\ c\end{matrix};z\right]\longrightarrow(1-z)^{-b}\,\frac{\Gamma(a-b)\,\Gamma(c)}{\Gamma(a)\,\Gamma(-b+c)}\,{}_2F_1\left[\begin{matrix}b,-a+c\\ 1-a+b\end{matrix};\frac{1}{1-z}\right]$$
$$+(1-z)^{-a}\,\frac{\Gamma(-a+b)\,\Gamma(c)}{\Gamma(b)\,\Gamma(-a+c)}\,{}_2F_1\left[\begin{matrix}a,-b+c\\ 1+a-b\end{matrix};\frac{1}{1-z}\right]$$

❑ **T2191** ([29], (1.8.10))

$${}_2F_1\left[\begin{matrix}a,b\\ c\end{matrix};z\right]\longrightarrow(1-z)^{c-a-b}\,\frac{\Gamma(c)\,\Gamma(a+b-c)}{\Gamma(a)\,\Gamma(b)}\,{}_2F_1\left[\begin{matrix}c-b,c-a\\ 1-a-b+c\end{matrix};1-z\right]$$
$$+\frac{\Gamma(c)\,\Gamma(c-a-b)}{\Gamma(c-b)\,\Gamma(c-a)}\,{}_2F_1\left[\begin{matrix}a,b\\ 1+a+b-c\end{matrix};1-z\right]$$

❑ **T2192**

$${}_2F_1\left[\begin{matrix}a,b\\ c\end{matrix};z\right]\longrightarrow z^{-a}\,\frac{\Gamma(1+a-c)\,\Gamma(1+b-c)}{\Gamma(1-c)\,\Gamma(1+a+b-c)}\,{}_2F_1\left[\begin{matrix}a,1+a-c\\ 1+a+b-c\end{matrix};\frac{z-1}{z}\right]$$
$$-z^{1-c}\,\frac{\Gamma(1+a-c)\,\Gamma(1+b-c)\,\Gamma(-1+c)}{\Gamma(a)\,\Gamma(b)\,\Gamma(1-c)}\,{}_2F_1\left[\begin{matrix}1+a-c,1+b-c\\ 2-c\end{matrix};z\right]$$

6.3 ${}_3F_2$ hypergeometric series

❑ **T3204**

$${}_3F_2\left[\begin{matrix}a,b,c\\d,e\end{matrix};1\right] \longrightarrow \frac{\Gamma(e)\,\Gamma(-a-b-c+d+e)}{\Gamma(-a+e)\,\Gamma(-b-c+d+e)}\,{}_3F_2\left[\begin{matrix}a,-b+d,-c+d\\d,-b-c+d+e\end{matrix};1\right]$$

❑ **T3205** ([29], (2.3.3.7))

$${}_3F_2\left[\begin{matrix}a,b,c\\d,e\end{matrix};1\right] \rightarrow \frac{\Gamma(d)\,\Gamma(e)\,\Gamma(-a-b-c+d+e)}{\Gamma(b)\,\Gamma(-a-b+d+e)\,\Gamma(-b-c+d+e)}{}_3F_2\left[\begin{matrix}-b+d,-b+e,-a-b-c+d+e\\-a-b+d+e,-b-c+d+e\end{matrix};1\right]$$

❑ **T3206**

$${}_3F_2\left[\begin{matrix}a,b,-n\\d,e\end{matrix};1\right] \longrightarrow \frac{(-a-b+d+e)_n}{(e)_n}\,{}_3F_2\left[\begin{matrix}-n,-a+d,-b+d\\d,-a-b+d+e\end{matrix};1\right]$$

where n is a nonnegative integer.

❑ **T3207** ([7], (3.1.1))

$${}_3F_2\left[\begin{matrix}a,b,-n\\d,e\end{matrix};1\right] \longrightarrow \frac{(-b+e)_n}{(e)_n}\,{}_3F_2\left[\begin{matrix}-n,b,-a+d\\d,1+b-e-n\end{matrix};1\right]$$

where n is a nonnegative integer.

❑ **T3231** ([7], Appendix(III.21), $q \to 1$)

$${}_3F_2\left[\begin{matrix}2\,a,2\,b,c\\\frac{1}{2}+a+b,d\end{matrix};1\right] \longrightarrow {}_4F_3\left[\begin{matrix}a,b,c,-c+d\\\frac{1}{2}+a+b,\frac{d}{2},\frac{1+d}{2}\end{matrix};1\right]$$

❑ **T3234** ([29], (2.5.7))

$${}_3F_2\left[\begin{matrix}2\,a,2\,b,a+b\\2\,a+2\,b,\frac{1}{2}+a+b\end{matrix};z\right] \longrightarrow {}_2F_1\left[\begin{matrix}a,b\\\frac{1}{2}+a+b\end{matrix};z\right]^2$$

❑ **T3235** ([7], Ex 3.4, $q \to 1$)

$${}_3F_2\left[\begin{matrix}a,b,-n\\d,2\,b\end{matrix};2\right] \longrightarrow \frac{(-a+d)_n}{(d)_n}\,{}_4F_3\left[\begin{matrix}\frac{a}{2},\frac{1}{2}+\frac{a}{2},\frac{1}{2}-\frac{n}{2},\frac{-n}{2}\\\frac{1}{2}+\frac{a}{2}-\frac{d}{2}-\frac{n}{2},1+\frac{a}{2}-\frac{d}{2}-\frac{n}{2},\frac{1}{2}+b\end{matrix};1\right]$$

❑ **T3236** ([7], (3.4.8), $q \to 1$)

$${}_3F_2\left[\begin{matrix}a,\frac{a}{2}+1,b\\\frac{a}{2},1+a-b\end{matrix};z\right] \longrightarrow \frac{(1-z)}{(1+z)^{1+a}}\,{}_2F_1\left[\begin{matrix}\frac{1}{2}+\frac{a}{2},\frac{1}{2}+\frac{a}{2}\\1+a-b\end{matrix};\frac{4\,z}{(1+z)^2}\right]$$

where the input series is in very-well-poised order.

❑ **T3237** ([7], (3.10.4), $q \to 1$)

$${}_3F_2\left[\begin{matrix}x,y,-n\\b,a\end{matrix};1\right] \longrightarrow \frac{(a-x)_n\,(a-y)_n}{(e)_n\,(a-x-y)_n}{}_6V_5\left[-a-n+x+y;1-a-b-n+x+y,x,y,-n;-1\right]$$

❑ **T3239**

$${}_3F_2\left[\begin{matrix}a,b,c\\1+a-b,1+a-c\end{matrix};z\right] \longrightarrow (1-z)^{-a}\,{}_3F_2\left[\begin{matrix}\frac{a}{2},\frac{1+a}{2},1+a-b-c\\1+a-b,1+a-c\end{matrix};\frac{-4\,z}{(1-z)^2}\right]$$

❑ **T3240** ([7], (3.5.10), $q \to 1$)

$$ {}_3F_2\left[\begin{matrix} c,b,d \\ a,a-b+d \end{matrix};1\right] \longrightarrow \frac{\Gamma(2\,a)\,\Gamma(2\,a-2\,b-c)\,\Gamma(a-b+d)\,\Gamma(a-c+d)}{\Gamma(2\,a-2\,b)\,\Gamma(2\,a-c)\,\Gamma(a+d)\,\Gamma(a-b-c+d)} \times {}_7V_6\left[-\frac{1}{2}+a;b,\frac{c}{2},\frac{1}{2}+\frac{c}{2},\frac{a}{2}-\frac{d}{2},\frac{1}{2}+\frac{a}{2}-\frac{d}{2};1\right] $$

❑ **T3261** ([7], Appendix(III.33), $q \to 1$)

$$ {}_3F_2\left[\begin{matrix} a,b,c \\ d,e \end{matrix};1\right] \longrightarrow \frac{\Gamma(c)\,\Gamma(1+c-d)\,\Gamma(1-a)\,\Gamma(-b-c+e)}{\Gamma(-b+e)\,\Gamma(-c+e)\,\Gamma(1-a+c)\,\Gamma(1-d)}\,{}_3F_2\left[\begin{matrix} c,-a+d,1+c-e \\ 1-a+c,1+b+c-e \end{matrix};1\right] - \frac{\Gamma(-1+d)\,\Gamma(e)\,\Gamma(1+b-d)\,\Gamma(1+c-d)\,\Gamma(1-a)\,\Gamma(-b-c+e)\,\Gamma(1+b+c-e)}{\Gamma(1-d)\,\Gamma(1-d+e)\,\Gamma(b)\,\Gamma(c)\,\Gamma(-a+d)\,\Gamma(-1-b-c+d+e)\,\Gamma(2+b+c-d-e)} \times {}_3F_2\left[\begin{matrix} 1+a-d,1+b-d,1+c-d \\ 2-d,1-d+e \end{matrix};1\right] $$

❑ **T3262** ([29], (4.3.4.2))

$$ {}_3F_2\left[\begin{matrix} a,b,c \\ d,e \end{matrix};1\right] \longrightarrow \frac{\Gamma(e)\,\Gamma(-b-c+e)}{\Gamma(-b+e)\,\Gamma(-c+e)}\,{}_3F_2\left[\begin{matrix} -a+d,b,c \\ d,1+b+c-e \end{matrix};1\right] + \frac{\Gamma(d)\,\Gamma(e)\,\Gamma(b+c-e)\,\Gamma(-a-b-c+d+e)}{\Gamma(-a+d)\,\Gamma(b)\,\Gamma(c)\,\Gamma(-b-c+d+e)}\,{}_3F_2\left[\begin{matrix} -b+e,-c+e,-a-b-c+d+e \\ -b-c+d+e,1-b-c+e \end{matrix};1\right] $$

❑ **T3263** ([7], Appendix(III.33), $q \to 1$)

$$ {}_3F_2\left[\begin{matrix} a,b,c \\ d,e \end{matrix};1\right] \longrightarrow \frac{\Gamma(1-c)\,\Gamma(d)\,\Gamma(-a-b+d)\,\Gamma(1+a-e)}{\Gamma(1+a-c)\,\Gamma(-a+d)\,\Gamma(-b+d)\,\Gamma(1-e)}\,{}_3F_2\left[\begin{matrix} 1+a-d,-c+e,a \\ 1+a+b-d,1+a-c \end{matrix};1\right] + \frac{\Gamma(1-c)\,\Gamma(a+b-d)\,\Gamma(d)\,\Gamma(1+a-e)\,\Gamma(e)\,\Gamma(-a-b-c+d+e)}{\Gamma(a)\,\Gamma(b)\,\Gamma(1-b-c+d)\,\Gamma(1+a+b-d-e)\,\Gamma(-c+e)\,\Gamma(-a-b+d+e)} \times {}_3F_2\left[\begin{matrix} 1-b,-a-b-c+d+e,-b+d \\ 1-a-b+d,1-b-c+d \end{matrix};1\right] $$

❑ **T3264** ([7], Appendix(III.34), $q \to 1$)

$$ {}_3F_2\left[\begin{matrix} a,b,c \\ d,e \end{matrix};1\right] \longrightarrow \frac{\Gamma(1+b-e)\,\Gamma(1+c-e)}{\Gamma(1-e)\,\Gamma(1+b+c-e)}\,{}_3F_2\left[\begin{matrix} -a+d,b,c \\ d,1+b+c-e \end{matrix};1\right] - \frac{\Gamma(d)\,\Gamma(1+a-e)\,\Gamma(1+b-e)\,\Gamma(1+c-e)\,\Gamma(e-1)}{\Gamma(a)\,\Gamma(b)\,\Gamma(c)\,\Gamma(1-e)\,\Gamma(1+d-e)}\,{}_3F_2\left[\begin{matrix} 1+c-e,1+b-e,1+a-e \\ 1+d-e,2-e \end{matrix};1\right] $$

❑ **T3267** ([7], Ex. 3.6, $q \to 1$)

$$ {}_3F_2\left[\begin{matrix} a,b,c \\ d,e \end{matrix};1\right] \to \frac{\Gamma(d)\,\Gamma(1+a-e)\,\Gamma(1+b-e)\,\Gamma(1+c-e)}{\Gamma(-a+d)\,\Gamma(1-e)\,\Gamma(1+a+b-e)\,\Gamma(1+a+c-e)} \times {}_3F_2\left[\begin{matrix} a,1+a-e,1+a+b+c-d-e \\ 1+a+b-e,1+a+c-e \end{matrix};1\right] - \frac{\Gamma(d)\,\Gamma(1+a-e)\,\Gamma(1+b-e)\,\Gamma(1+c-e)\,\Gamma(e-1)}{\Gamma(a)\,\Gamma(b)\,\Gamma(c)\,\Gamma(1-e)\,\Gamma(1+d-e)} \times {}_3F_2\left[\begin{matrix} 1+a-e,1+b-e,1+c-e \\ 2-e,1+d-e \end{matrix};1\right] $$

❑ **T3268** ([7], Ex. 3.6, $q \to 1$)

$$
{}_3F_2\left[\begin{matrix}a,b,c\\ d,e\end{matrix};1\right] \to \frac{\Gamma(a-b)\,\Gamma(d)\,\Gamma(e)\,\Gamma(-a-b-c+d+e)}{\Gamma(a)\,\Gamma(-b+d)\,\Gamma(-b+e)\,\Gamma(-a-c+d+e)} \times {}_3F_2\left[\begin{matrix}b,-a+d,-a+e\\ 1-a+b,-a-c+d+e\end{matrix};1\right]
$$
$$
+ \frac{\Gamma(-a+b)\,\Gamma(d)\,\Gamma(e)\,\Gamma(-a-b-c+d+e)}{\Gamma(b)\,\Gamma(-a+d)\,\Gamma(-a+e)\,\Gamma(-b-c+d+e)} \times {}_3F_2\left[\begin{matrix}a,-b+d,-b+e\\ -b-c+d+e,1+a-b\end{matrix};1\right]
$$

6.4 ${}_4F_3$ hypergeometric series

❑ **T4301** ([29], (4.3.5.1))

$$
{}_4F_3\left[\begin{matrix}a,b,c,-n\\ e,f,1+a+b+c-e-f-n\end{matrix};1\right] \to \frac{(-a+e)_n\,(-a+f)_n}{(e)_n\,(f)_n}
$$
$$
\times\ {}_4F_3\left[\begin{matrix}-n,a,1+a+c-e-f-n,1+a+b-e-f-n\\ 1+a+b+c-e-f-n,1+a-e-n,1+a-f-n\end{matrix};1\right]
$$

where n is a nonnegative integer.

❑ **T4302** ([7], Appendix(III.16), $q \to 1$)

$$
{}_4F_3\left[\begin{matrix}a,b,c,-n\\ e,f,1+a+b+c-e-f-n\end{matrix};1\right] \to \frac{(a)_n\,(-a-b+e+f)_n\,(-a-c+e+f)_n}{(e)_n\,(f)_n\,(-a-b-c+e+f)_n}
$$
$$
\times\ {}_4F_3\left[\begin{matrix}-n,-a+e,-a+f,-a-b-c+e+f\\ -a-b+e+f,-a-c+e+f,1-a-n\end{matrix};1\right]
$$

where n is a nonnegative integer.

❑ **T4303** ([29], (2.4.1.1))

$$
{}_4F_3\left[\begin{matrix}a,b,c,-n\\ e,f,1+a+b+c-e-f-n\end{matrix};1\right] \to \frac{(-a-b+e+f)_n\,(-a-c+e+f)_n}{(-a+e+f)_n\,(-a-b-c+e+f)_n}
$$
$$
\times\ {}_7V_6\left[-1-a+e+f;-a+f,-a+e,b,c,-n;1\right]
$$

where n is a nonnegative integer.

❑ **T4304** ([29], (4.3.6.4))

$$
{}_4F_3\left[\begin{matrix}a,b,c,-n\\ e,f,1+a+b+c-e-f-n\end{matrix};1\right]
$$
$$
\to \frac{\Gamma(-b-c+e+f+n)\,\Gamma(-a-c+e+f+n)\,\Gamma(-a-b+e+f+n)\,\Gamma(e+f+n)}{\Gamma(-c+e+f+n)\,\Gamma(-b+e+f+n)\,\Gamma(-a+e+f+n)\,\Gamma(-a-b-c+e+f+n)}
$$
$$
\times\ {}_7V_6\left[-1+e+f+n;a,b,c,e+n,f+n;1\right]
$$

where n is a nonnegative integer.

❑ **T4306** ([7], Appendix(III.21), $q \to 1$)

$$
{}_4F_3\left[\begin{matrix}a,b,c,d\\ \frac{1}{2}+a+b,\frac{c+d}{2},\frac{1+c+d}{2}\end{matrix};1\right] \longrightarrow {}_3F_2\left[\begin{matrix}2\,a,2\,b,c\\ \frac{1}{2}+a+b,c+d\end{matrix};1\right]
$$

❑ **T4309** ([7], Ex 2.13(i), $q \to 1$)

$$
{}_4F_3\left[\begin{matrix} a, b, c, d \\ 1+a-b, 1+a-c, 1+a-d \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(3+2\,a-2\,b-2\,c-2\,d)\,\Gamma(2+2\,a-b-c-d)}{\Gamma(3-3\,a-2\,b-2\,c-2\,d)\,\Gamma(2+a-b-c-d)}
$$

$$
\times\, {}_7V_6\left[1+2\,a-b-c-d; a/2, \frac{1}{2}+\frac{a}{2}, 1+a-c-d, 1+a-b-d, 1+a-b-c; 1\right]
$$

❑ **T4310** ([7], Ex 2.13(ii), $q \to 1$)

$$
{}_4F_3\left[\begin{matrix} a, b, c, d \\ 1+a-b, 1+a-c, 1+a-d \end{matrix}; -1\right] \longrightarrow \frac{\Gamma(2+2\,a-b-c-d)\,\Gamma(1+\frac{a}{2})}{\Gamma(1+a)\,\Gamma(2+\frac{3\,a}{2}-b-c-d)}
$$

$$
\times\, {}_6V_5\left[1+2\,a-b-c-d; a/2, 1+a-c-d, 1+a-b-d, 1+a-b-c; -1\right]
$$

❑ **T4312** ([7], Ex 3.4, $q \to 1$)

$$
{}_4F_3\left[\begin{matrix} \frac{a}{2}, \frac{1}{2}+\frac{a}{2}, \frac{1}{2}-\frac{n}{2}, \frac{-n}{2} \\ \frac{d}{2}, \frac{1}{2}+\frac{d}{2}, \frac{1}{2}+b \end{matrix}; 1\right] \longrightarrow \frac{(d-a)_n}{(d)_n}\, {}_3F_2\left[\begin{matrix} a, b, -n \\ 1+a-d-n, 2\,b \end{matrix}; 2\right]
$$

❑ **T4331**

$$
{}_4F_3\left[\begin{matrix} a, \frac{a}{2}+1, b, c \\ \frac{a}{2}, 1+a-b, 1+a-c \end{matrix}; z\right] \longrightarrow \frac{(1+z)}{(1-z)^{a+1}}\, {}_3F_2\left[\begin{matrix} \frac{1}{2}+\frac{a}{2}, 1+\frac{a}{2}, 1+a-b-c \\ 1+a-b, 1+a-c \end{matrix}; -\frac{4\,z}{(1-z)^2}\right]
$$

where the input series is in very-well-poised order.

❑ **T4332**

$$
{}_4F_3\left[\begin{matrix} b, x, y, -n \\ a-x, a-y, a+n \end{matrix}; 1\right] \longrightarrow \frac{(a)_n\,(a-x-y)_n}{(a-x)_n\,(a-y)_n} {}_5F_4\left[\begin{matrix} x, y, \frac{a}{2}-\frac{b}{2}, \frac{1}{2}+\frac{a}{2}-\frac{b}{2}, -n \\ a-b, \frac{a}{2}, \frac{1}{2}+\frac{a}{2}, 1-a-n+x+y \end{matrix}; 1\right]
$$

❑ **T4362** ([29], (2.4.4.3))

$$
{}_4F_3\left[\begin{matrix} a, b, c, d \\ e, f, 1+a+b+c+d-e-f \end{matrix}; 1\right] \longrightarrow \frac{\Gamma(-a+e+f)\,\Gamma(-a-b-c+e+f)}{\Gamma(-a-b+e+f)\,\Gamma(-a-c+e+f)}
$$

$$
\times \frac{\Gamma(-a-b-d+e+f)\,\Gamma(-a-c-d+e+f)}{\Gamma(-a-d+e+f)\,\Gamma(-a-b-c-d+e+f)} \times {}_7V_6\left[-1-a+e+f; -a+f, -a+e, b, c, d; 1\right]
$$

$$
- \frac{\Gamma(e)\,\Gamma(a+b+c+d-e-f)\,\Gamma(f)\,\Gamma(-a-b-c+e+f)\,\Gamma(-a-b-d+e+f)}{\Gamma(a)\,\Gamma(b)\,\Gamma(c)\,\Gamma(d)\,\Gamma(-a-b-c-d+e+f)}
$$

$$
\times \frac{\Gamma(-a-c-d+e+f)\,\Gamma(-b-c-d+e+f)}{\Gamma(-a-b-c-d+2\,e+f)\,\Gamma(-a-b-c-d+e+2\,f)}
$$

$$
\times\, {}_4F_3\left[\begin{matrix} -a-b-c+e+f, -a-b-d+e+f, -a-c-d+e+f, -b-c-d+e+f \\ -a-b-c-d+2\,f+f, -a-b-c-d+e+2\,f, 1-a-b-c-d+e+f \end{matrix}; 1\right]
$$

❑ **T4391** ([7], (3.5.7), $q \to 1$)

$$
{}_4F_3\left[\begin{matrix} a, c, d, e \\ b, 2\,a, 1-2\,b+c+d+e \end{matrix}; 1\right] \to \frac{\Gamma(b)\,\Gamma(\frac{1}{2}+b)\,\Gamma(b-{}^{c}2-\frac{d}{2})\,\Gamma(b-\frac{c}{2}-\frac{d}{2})}{\Gamma(b-\frac{c}{2})\,\Gamma(\frac{1}{2}+b-\frac{c}{2})\,\Gamma(b-\frac{d}{2})\,\Gamma(\frac{1}{2}+b-\frac{d}{2})}
$$

$$
\times \frac{\Gamma(b-\frac{c}{2}-\frac{e}{2})\,\Gamma(\frac{1}{2}+b-\frac{c}{2}-\frac{e}{2})\,\Gamma(b-\frac{d}{2}-\frac{e}{2}\,\Gamma(\frac{1}{2}+b-\frac{d}{2}-\frac{e}{2}}{\Gamma(b\frac{e}{2})\,\Gamma(\frac{1}{2}+b-\frac{e}{2}\,\Gamma(b-\frac{c}{2}-\frac{d}{2}-\frac{e}{2})\,\Gamma(\frac{1}{2}+b-\frac{c}{2}-\frac{d}{2}-\frac{e}{2})}
$$

$$\times {}_9V_8\left[-\frac{1}{2}+b; -a+b, \frac{c}{2}, \frac{1}{2}+\frac{c}{2}, \frac{d}{2}, \frac{1}{2}+\frac{d}{2}, \frac{e}{2}, \frac{1}{2}+\frac{e}{2}; 1\right]$$

$$-\frac{\Gamma(\frac{1}{2}+a)\,\Gamma(b)\,\Gamma(b-\frac{c}{2}-\frac{d}{2})\,\Gamma(\frac{1}{2}+b-\frac{c}{2}-\frac{d}{2})\,\Gamma(a+2\,b-c-d-e)\,\Gamma(b-\frac{c}{2}-\frac{e}{2})\,\Gamma(\frac{1}{2}+b-\frac{c}{2}-\frac{e}{2})}{\Gamma(\frac{c}{2})\,\Gamma(\frac{1}{2}+\frac{c}{2})\,\Gamma(\frac{d}{2})\,\Gamma(\frac{1}{2}+\frac{d}{2})\,\Gamma(3\,b-c-d-e)\,\Gamma(b-\frac{c}{2}-\frac{d}{2}-\frac{e}{2})\,\Gamma(\frac{1}{2}+b-\frac{c}{2}-\frac{d}{2}-\frac{e}{2})}$$

$$\times\frac{\Gamma(b-\frac{d}{2}-\frac{e}{2})\,\Gamma(\frac{1}{2}+b-\frac{d}{2}-\frac{e}{2}\,\Gamma(-b+\frac{c}{2}+\frac{d}{2}+\frac{e}{2})\,\Gamma(\frac{1}{2}-b+\frac{c}{2}+\frac{d}{2}+\frac{e}{2}}{\Gamma(a+b-\frac{c}{2}-\frac{d}{2}-\frac{e}{2})\,\Gamma(\frac{1}{2}+a+b-\frac{c}{2}-\frac{d}{2}-\frac{e}{2})\,\Gamma(\frac{e}{2})\,\Gamma(\frac{1}{2}+\frac{e}{2})}$$

$$\times {}_4F_3\left[\begin{matrix}a+2\,b-c-d-e, 2\,b-c-d, 2\,b-c-e, 2\,b-d-e\\ 3\,b-c-d-e, 2\,a+2\,b-c-d-e, 1+2\,b-c-d-e\end{matrix}; 1\right]$$

6.5 ${}_5F_4$ hypergeometric series

❑ **T5401** ([29], (2.4.3.4))

$${}_5F_4\left[\begin{matrix}a, b, c, d, -n\\ 1+a-b, 1+a-c, 1+a-d, -2-2\,a+2\,b+2\,c+2\,d-n\end{matrix}; 1\right]$$

$$\longrightarrow\frac{(2+a-b-c-d)_n\,(3+3\,a-2\,b-2\,c-2\,d)_n}{(2+2\,a-b-c-d)_n\,(3+2\,a-2\,b-2\,c-2\,d)_n}$$

$$\times {}_9V_8[1+2\,a-b-c-d; 1+a-c-d, 1+a-b-d, 1+a-b-c,$$

$$\frac{a}{2}, \frac{1}{2}+\frac{a}{2}, 3+3\,a-2\,b-2\,c-2\,d+n, -n; 1]$$

where n is a nonnegative integer.

❑ **T5402** ([7], Appendix(III.26), $q \to 1$)

$${}_5F_4\left[\begin{matrix}-n, b, c, d, e\\ 1-b-n, 1-c-n, 1-d-n, -2+2\,b+2\,c+2\,d+e+2\,n\end{matrix}; 1\right]$$

$$\longrightarrow\frac{(2-b-c-d-e-2\,n)_n\,(3-2\,b-2\,c-2\,d-3\,n)_n}{(2-b-c-d-2\,n)_n\,(3-2\,b-2\,c-2\,d-e-3\,n)_n}$$

$$\times {}_9V_8[1-b-c-d-2\,n; 1-c-d-n, 1-b-d-n, 1-b-c-n,$$

$$\frac{-n}{2}, \frac{1}{2}-\frac{n}{2}, e, 3-2\,b-2\,c-2\,d-e-3\,n; 1]$$

where n is a nonnegative integer.

❑ **T5403**

$${}_5F_4\left[\begin{matrix}x, y, a, \frac{1}{2}+a, -n\\ 2\,a, b, \frac{1}{2}+b, 1-2\,b-n+x+y\end{matrix}; 1\right]\longrightarrow\frac{(2\,b-x)_n\,(2\,b-y)_n}{(2\,b)_n\,(2\,b-x-y)_n}{}_4F_3\left[\begin{matrix}-2\,a+2\,b, x, y, -n\\ -2\,b-x, -2\,b-y, 2\,b+n\end{matrix}; 1\right]$$

6.6 ${}_6F_5$ hypergeometric series

❑ **T6501** ([29], (2.4.3.3))

$${}_6F_5\left[\begin{matrix}a, 1+\frac{a}{2}, b, c, d, -n\\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, -1-2\,a+2\,b+2\,c+2\,d-n\end{matrix}; 1\right]$$

$$\longrightarrow\frac{(2+3\,a-2\,b-2\,c-2\,d)_n\,(1+a-b-c-d)_n}{(2+2\,a-2\,b-2\,c-2\,d)_n\,(2+2\,a-b-c-d)_n}$$

$$\times\, {}_9V_8[1+2\,a-b-c-d;1+a-c-d,1+a-b-d,1+a-b-c,$$
$$1+\frac{a}{2},\frac{1}{2}+\frac{a}{2},2+3\,a-2\,b-2\,c-2\,d+n,-n;1]$$

❑ **T6531** ([29], (2.4.3.5))

$${}_6F_5\left[\begin{matrix}a,1+\frac{a}{2},b,c,d,-n\\ \frac{a}{2},1+a-b,1+a-c,1+a-d,-2\,a+2\,b+2\,c+2\,d-n\end{matrix};1\right]$$
$$\longrightarrow \frac{(1+3\,a-2\,b-2\,c-2\,d+2\,n)}{(1+3\,a-2\,b-2\,c-2\,d)}\,\frac{(a-b-c-d)_n\,(1+3\,a-2\,b-2\,c-2\,d)_n}{(2+2\,a-b-c-d)_n\,(1+2\,a-2\,b-2\,c-2\,d)_n}$$
$$\times\, {}_9V_8[1+2\,a-b-c-d;1+a-c-d,1+a-b-d,1+a-b-c,$$
$$\frac{1}{2}+\frac{a}{2},1+\frac{a}{2},1+3\,a-2\,b-2\,c-2\,d+n,-n;1]$$

❑ **T6532** ([7], Ex 2.13(ii), $q\to 1$)

$${}_6F_5\left[\begin{matrix}a,1+\frac{a}{2},b,c,d,1+2\,a-2\,b-c-d\\ \frac{a}{2},1+a-b,1+a-c,1+a-d,-a+2\,b+c+d\end{matrix};-1\right]$$
$$\longrightarrow \frac{\Gamma(1+2\,b)\,\Gamma(1+a-b)}{\Gamma(1+b)\,\Gamma(1+a)}\,{}_4F_3\left[\begin{matrix}2\,b,-a+2\,b+c,-a+2\,b+d,1+a-c-d\\ 1+a-c,1+a-d,-a+2\,b+c+d\end{matrix};-1\right]$$

where the input series is in very-well-poised order.

❑ **T6533** ([7], (3.10.4), $q\to 1$)

$${}_6F_5\left[\begin{matrix}a,1+\frac{a}{2},b,x,y,-n\\ \frac{a}{2},1+a-b,1+a-x,1+a-y,1+a+n\end{matrix};-1\right]$$
$$\longrightarrow \frac{(1+a)_n\,(1+a-x-y)_n}{(1+a-x)_n\,(1+a-y)_n}\,{}_3F_2\left[\begin{matrix}-n,x,y\\ -a-n+x+y,1+a-b\end{matrix};1\right]$$

where the input series is in very-well-poised order.

6.7 $_7F_6$ hypergeometric series

❑ **T7631** ([29], (2.4.1.1))

$${}_7F_6\left[\begin{matrix}a,1+\frac{a}{2},b,c,d,e,f\\ \frac{a}{2},1+a-b,1+a-c,1+a-d,1+a-e,1+a-f\end{matrix};1\right]$$
$$\to \frac{\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(1+a-f)\,\Gamma(1+a-d-e-f)}{\Gamma(1+a)\,\Gamma(1+a-d-e)\,\Gamma(1+a-d-f)\,\Gamma(1+a-e-f)}\,{}_4F_3\left[\begin{matrix}1+a-b-c,d,e,f\\ 1+a-b,1+a-c,-a+d+e+f\end{matrix};1\right]$$

where the input series is in very-well-poised order.

❑ **T7632** ([29], (2.4.1.1))

$${}_7F_6\left[\begin{matrix}a,1+\frac{a}{2},b,c,d,e,-n\\ \frac{a}{2},1+a-b,1+a-c,1+a-d,1+a-e,1+a+n\end{matrix};1\right]$$
$$\longrightarrow \frac{(1+a)_n\,(1+a-d-e)_n}{(1+a-d)_n\,(1+a-e)_n}\,{}_4F_3\left[\begin{matrix}1+a-b-c,d,e,-n\\ 1+a-b,1+a-c,-a+d+e-n\end{matrix};1\right]$$

where the input series is in very-well-poised order and n is a nonnegative integer.

❑ **T7633** ([29], (4.3.6.4))

$$ {}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, d, e, 1+a-e+n \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, e-n \end{matrix}; 1\right] $$

$$ \to \frac{\Gamma(1+a-d)\,\Gamma(1+a-c)\,\Gamma(1+a-b)\,\Gamma(1+a-b-c-d)}{\Gamma(1+a-c-d)\,\Gamma(1+a-b-d)\,\Gamma(1+a-b-c)\,\Gamma(1+a)}\,{}_4F_3\left[\begin{matrix} b, c, d, -n \\ 1+a-e, -a+b+c+d, e-n \end{matrix}; 1\right] $$

where the input series is in very-well-poised order and n is a nonnegative integer.

❑ **T7634**

$$ {}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, d, e, f \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, 1+a-f \end{matrix}; 1\right] $$

$$ \longrightarrow \frac{\Gamma(1+a-e)\,\Gamma(1+a-f)\,\Gamma(2+2\,a-b-c-d)\,\Gamma(2+2\,a-b-c-d-e-f)}{\Gamma(1+a)\,\Gamma(1+a-e-f)\,\Gamma(2+2\,a-b-c-d-e)\,\Gamma(2+2\,a-b-c-d-f)} $$

$$ \times\, {}_7V_6[1+2\,a-b-c-d; 1+a-c-d, 1+a-b-d, 1+a-b-c, e, f; 1] $$

where the input series is in very-well-poised order.

❑ **T7635**

$$ {}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, d, e, f \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, 1+a-f \end{matrix}; 1\right] $$

$$ \longrightarrow \frac{\Gamma(1+a-c)\,\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(1+a-f)}{\Gamma(1+a)\,\Gamma(b)\,\Gamma(2+2\,a-b-d-e-f)\,\Gamma(2+2\,a-b-c-e-f)} $$

$$ \times \frac{\Gamma(3+3\,a-2\,b-c-d-e-f)\,\Gamma(2+2*a-b-c-d-e)}{\Gamma(2+2\,a-b-c-d-f)\,\Gamma(2+2\,a-b-c-d-e)}\,{}_7V_6[2+3\,a-2\,b-c-d-e-f; $$

$$ 1+a-b-c, 1+a-b-d, 1+a-b-e, 1+a-b-e, 2+2\,a-b-c-d-e-f; 1] $$

where the input series is in very-well-poised order.

❑ **T7636** ([7], (3.5.10), $q \to 1$)

$$ {}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, \frac{1}{2}+c, d, \frac{1}{2}+d \\ \frac{a}{2}, 1+a-b, 1+a-c, \frac{1}{2}+a-c, 1+a-d, \frac{1}{2}+a-d \end{matrix}; 1\right] $$

$$ \longrightarrow \frac{\Gamma(1+2\,a-2\,b)\,\Gamma(1+2\,a-2\,c)\,\Gamma(1+2\,a-2\,d)\,\Gamma(1+2\,a-b-2\,c-2\,d)}{\Gamma(1+2\,a)\,\Gamma(1+2\,a-2\,b-2\,c)\,\Gamma(1+2\,a-b-2\,d)\,\Gamma(1+2\,a-2\,c-2\,d)} $$

$$ \times\, {}_3F_2\left[\begin{matrix} 2\,c, b, \frac{1}{2}+a-2\,d \\ 1+2\,a-b-2\,d, \frac{1}{2}+a \end{matrix}; 1\right] $$

where the input series is in very-well-poised order.

❑ **T7637** ([7], Ex 2.13(i), $q \to 1$)

$$ {}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, \frac{1}{2}+b, c, d, 1+2\,a-2\,b-c-d \\ \frac{a}{2}, 1+a-b, \frac{1}{2}+a-b, 1+a-c, 1+a-d, -a+2\,b+c+d \end{matrix}; 1\right] $$

$$ \longrightarrow \frac{\Gamma(1+a-2\,b)\,\Gamma(1+2\,a-2\,b)}{\Gamma(1+a)\,\Gamma(1+2\,a-4\,b)}\,{}_3F_2\left[\begin{matrix} 2\,b, -a+2\,b+c, -a+2\,b+d, 1+a-c-d \\ 1+a-c, 1+a-d, -a+2\,b+c+d \end{matrix}; 1\right] $$

where the input series is in very-well-poised order.

❑ **T7691** ([29], (2.4.4.3))

$$
{}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, d, e, f \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, 1+a-f \end{matrix}; 1\right]
$$

$$
\to \frac{\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(1+a-f)\,\Gamma(1+a-d-e-f)}{\Gamma(1+a)\,\Gamma(1+a-d-e)\,\Gamma(1+a-d-f)\,\Gamma(1+a-e-f)}\,{}_4F_3\left[\begin{matrix} 1+a-b-c, d, e, f \\ 1+a-b, 1+a-c, -a+d+e+f \end{matrix}; 1\right]
$$

$$
+\frac{\Gamma(1+a-b)\,\Gamma(1+a-c)\,\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(1+a-f)\,\Gamma(2+2\,a-b-c-d-e-f)}{\Gamma(1+a)\,\Gamma(1+a-b-c)\,\Gamma(d)\,\Gamma(e)\,\Gamma(f)\,\Gamma(2+2\,a-b-d-e-f)}
$$

$$
\times\frac{\Gamma(-1-a+d+e+f)}{\Gamma(2+2\,a-c-d-e-f)}\,{}_4F_3\left[\begin{matrix} 1+a-d-e, 1+a-d-f, 1+a-e-f, 2+2\,a-b-c-d-e-f \\ 2+2\,a-b-d-e-f, 2+2\,a-c-d-e-f, 2+a-d-e-f \end{matrix}; 1\right]
$$

where the input series is in very-well-poised order.

❑ **T7692** ([29], (4.3.7.8))

$$
{}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, d, e, f \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, 1+a-f \end{matrix}; 1\right]
$$

$$
\to -\frac{\Gamma(a-b)\,\Gamma(1+a-c)\,\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(1+a-f)\,\Gamma(-a+b+d)\,\Gamma(-a+b+e)}{\Gamma(1+a)\,\Gamma(-a+b)\,\Gamma(1+b-c)\,\Gamma(1+b-d)\,\Gamma(1+b-e)\,\Gamma(1+b-f)\,\Gamma(d)}
$$

$$
\times\frac{\Gamma(-a+b+f)\,\Gamma(-a+d+e+f)\,\Gamma(a+1-d-e-f)\,\Gamma(1-c)\,\Gamma(1-a+2\,b)}{\Gamma(e)\,\Gamma(f)\,\Gamma(1+a-b-c)\,\Gamma(-2\,a+b+d+e+f)\,\Gamma(1+2\,a-b-d-e-f)}
$$

$$
\times\,{}_7V_6\,[-a+2\,b; b, -a+b+c, -a+b+d, -a+b+e, -a+b+f; 1]
$$

$$
+\frac{\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(1+a-f)\,\Gamma(1+a-d-e-f)\,\Gamma(1-c)\,\Gamma(1-c+e+f)}{\Gamma(1+a)\,\Gamma(1+a-d-e)\,\Gamma(1+a-d-f)\,\Gamma(1+a-e-f)\,\Gamma(1-c+e)}
$$

$$
\times\frac{\Gamma(-a+b+e)\,\Gamma(-a+b+f)}{\Gamma(1-c+f)\,\Gamma(-a+b+e+f)}\,{}_7V_6\,[-c+e+f; 1+a-b-c, 1+a-c-d, -a+e+f, e, f; 1]
$$

where the input series is in very-well-poised order.

❑ **T7693** ([7], Ex 2.15, $q \to 1$)

$$
{}_7F_6\left[\begin{matrix} 2\,a, a+1, a+b, a+c, a+d, a+e, a+f \\ a, 1+a-b, 1+a-c, 1+a-d, 1+a-e, 1+a-f \end{matrix}; 1\right]
$$

$$
\longrightarrow -\frac{\Gamma(a-b)\,\Gamma(1+2\,b)\,\Gamma(-b+c)\,\Gamma(b+c)\,\Gamma(1-a-d)\,\Gamma(1+a-d)\,\Gamma(1-a-e)}{\Gamma(1+2\,a)\,\Gamma(-a+b)\,\Gamma(-a+c)\,\Gamma(a+c)\,\Gamma(1-b-d)\,\Gamma(1+b-d)\,\Gamma(1-b-e)}
$$

$$
\times\frac{\Gamma(1+a-e)\,\Gamma(1-a-f)\,\Gamma(1+a-f)}{\Gamma(1+b-e)\,\Gamma(1-b-f)\,\Gamma(1+b-f)}\,{}_7V_6\,[2\,b; a+b, b+c, b+d, b+e, b+f; 1]
$$

$$
-\frac{\Gamma(a-c)\,\Gamma(b-c)\,\Gamma(b+c)\,\Gamma(1+2\,c)\,\Gamma(1-a-d)\,\Gamma(1+a-d)\,\Gamma(1-a-e)}{\Gamma(1+2\,a)\,\Gamma(-a+b)\,\Gamma(a+b)\,\Gamma(-a+c)\,\Gamma(1-c-d)\,\Gamma(1+c-d)\,\Gamma(1-c-e)}
$$

$$
\times\frac{\Gamma(1+a-e)\,\Gamma(1-a-f)\,\Gamma(1+a-f)}{\Gamma(1+c-e)\,\Gamma(1-c-f)\,\Gamma(1+c-f)}\,{}_7V_6\,[2\,c; a+c, b+c, c+d, c+e, c+f; 1]
$$

where the input series is in very-well-poised order.

❑ **T7694** ([29], (4.3.7.8))

$$
{}_7F_6\left[\begin{matrix} a, 1+\frac{a}{2}, b, c, d, e, f \\ \frac{a}{2}, 1+a-b, 1+a-c, 1+a-d, 1+a-e, 1+a-f \end{matrix}; 1\right]
$$

$$
\longrightarrow \frac{\Gamma(1+a-b)\,\Gamma(1-d)\,\Gamma(1+a-e)\,\Gamma(-a+c+e)\,\Gamma(1+a-f)\,\Gamma(1+a-b-e-f)}{\Gamma(1+a)\,\Gamma(-a+c)\,\Gamma(1+a-b-e)\,\Gamma(1-d+e)\,\Gamma(1+a-b-f)\,\Gamma(1+a-e-f)}
$$

$$
\times \frac{\Gamma(-a+c+f)\,\Gamma(1-d+e+f)}{\Gamma(1-d+f)\,\Gamma(-a+c+e+f)}\,{}_7V_6\,[-d+e+f; 1+a-b-d, -a+e+f, 1+a-c-d, e, f; 1]
$$

$$
+\frac{\Gamma(1+a-b)\,\Gamma(1+a-c)\,\Gamma(1-d)\,\Gamma(1+a-d)\,\Gamma(1+a-e)\,\Gamma(-a+c+e)\,\Gamma(1+a-f)}{\Gamma(1+a)\,\Gamma(b)\,\Gamma(1-b+c)\,\Gamma(1+a-c-d)\,\Gamma(2+a-b-d-e)\,\Gamma(e)}
$$

$$
\times \frac{\Gamma(3+2\,a-2\,b-d-e-f)\,\Gamma(2+2\,a-b-c-d-e-f)\,\Gamma(-a+c+f)\,\Gamma(-1-a+b+e+f)}{\Gamma(2+a-b-d-f)\,\Gamma(2+2\,a-b-c-e-f)\,\Gamma(2+2\,a-b-d-e-f)\,\Gamma(f)\,\Gamma(-1-2\,a+b+c+e+f)}
$$

$$
\times\,{}_7V_6\,[2+2\,a-2\,b-d-e-f; 1+a-b-d, 1-b, 2+2\,a-b-c-e-f-g, 1+a-b-f, 1+a-b-e; 1]
$$

where the input series is in very-well-poised order.

6.8 ${}_8F_7$ hypergeometric series

❑ **T8731**

$$
{}_7F_6\left[\begin{matrix} 2\,a+n, a+\frac{n}{2}+1, c, d, e, \frac{1}{2}+a+n, 1+4\,a-c-d-e+n, -n \\ a+\frac{n}{2}, 1+2\,a+n-c, 1+2\,a+n-d, 1+2\,a+n-e, \frac{1}{2}+a, -2\,a+c+d+e, 1+2\,a+2\,n \end{matrix}; -1\right]
$$

$$
\longrightarrow \frac{(1+2\,a-c)_n\,(1+2\,a-d)_n\,(1+2\,a-e)_n\,(1+2\,a-c-d-e)_n}{(1+2\,a)_n\,(1+2\,a-c-d)_n\,(1+2\,a-c-e)_n(1+2\,a-d-e)_n}
$$

$$
\times\,{}_{11}V_{10}\left[a; -n, \frac{c}{2}, \frac{1}{2}+\frac{c}{2}, \frac{d}{2}, \frac{1}{2}+\frac{d}{2}, \frac{e}{2}, \frac{1}{2}+\frac{e}{2}, \frac{1}{2}+2\,a-\frac{c}{2}-\frac{d}{2}-\frac{e}{2}+\frac{n}{2}, 1+2\,a-\frac{c}{2}-\frac{d}{2}-\frac{e}{2}+\frac{n}{2}; 1\right]
$$

where the input series is in very-well-poised order.

❑ **T8732**

$$
{}_7F_6\left[\begin{matrix} 2\,a-e, a-\frac{e}{2}+1, \frac{1}{2}+a-e, c, d, e, 1+4\,a-c-d-e+n, -n \\ a-\frac{e}{2}, \frac{1}{2}+a, 1+2\,a-e-c, 1+2\,a-e-d, 1+2\,a-2\,e, -2\,a+c+d-n, 1+2\,a+n-e \end{matrix}; -1\right]
$$

$$
\longrightarrow \frac{(1+2\,a-c)_n\,(1+2\,a-d)_n\,(1+2\,a-e)_n\,(1+2\,a-c-d-e)_n}{(1+2\,a)_n\,(1+2\,a-c-d)_n\,(1+2\,a-c-e)_n(1+2\,a-d-e)_n}
$$

$$
\times\,{}_{11}V_{10}\left[a; e, \frac{c}{2}, \frac{1}{2}+\frac{c}{2}, \frac{d}{2}, \frac{1}{2}+\frac{d}{2}, \frac{1}{2}+2\,a-\frac{c}{2}-\frac{d}{2}-\frac{e}{2}+\frac{n}{2}, 1+2\,a-\frac{c}{2}-\frac{d}{2}-\frac{e}{2}+\frac{n}{2}, \frac{1}{2}-\frac{n}{2}, \frac{-n}{2}; 1\right]
$$

where the input series is in very-well-poised order.

The usage of the functions `HypTransfList` and `HypTransfPrint` is similar to `HypSumList` and `HypSumPrint`.

Chapter 7

Contiguous relations

`HypContig, HypContigList`

Two hypergeometric series ${}_rF_s\left[\begin{smallmatrix}(a)\\(b)\end{smallmatrix};z\right]$ and ${}_rF_s\left[\begin{smallmatrix}(a')\\(b')\end{smallmatrix};z\right]$ are said to be *contiguous* when all their parameters are equal except one pair, and this pair of parameters differs only by unity, for example:

$${}_0F_2\left[\begin{matrix}a\\b_1,b_2+1\end{matrix};z\right] \qquad \text{is contiguous to} \qquad {}_0F_2\left[\begin{matrix}a\\b_1,b_2\end{matrix};z\right].$$

Similarly, any two hypergeometric series are said to be *associated* when their parameters differ by integers only.
Thus, ${}_2F_3\left[\begin{matrix}a_1,a_2\\b_1,b_2,b_3\end{matrix};z\right]$ and ${}_2F_3\left[\begin{matrix}a_1+m,a_2+n\\b_1+p,b_2+q,b_3+r\end{matrix};z\right]$ are associated, for integer values of m,n,p,q and r.

From now on (and to simplify), the term "contiguous" also includes the "associated" notion.

The package contains a large number of contiguous formulas:

```
- C01, C02,
- C14, C15, C16, C17, C18, C19,
  C20, C21, C22, C23, C24, C25,
- C26, C27, C30, C31, C32, C33,
  C34, C35, C36, C40, C41, C42, C43, C44,
  C45, C46, C49, C50, C51, C52,
- C53,
  C54, C55, C56, C57,
  C58,
- C59, C60, C61, C62, C63,
  C64, C65, C66, C67, C68, C69, C70, C71, C72, C73,
  C74, C75, C76, C77, C78, C79, C80, C81, C82, C83, C84, C85, C86, C87, C88,
  C89, C90, C91, C92, C93, C94, C95, C96, C97, C98,
  C99, C100, C101, C102, C103,
- C104, C110,
  C105, C112,
  C106, C107, C114, C115,
  C108,
  C118,
  C109, C120,
- C111,
  C113,
  C116, C117,
  C119,
  C121.
```

The usage of the function `HypContig` is: `HypContig(shyp,nb,[pos1, pos2, ...]])` with:

- `shyp`, a hypergeometric series (in standard notation);
- `nb`, the numbering of the formula (see above);
- the fact that the parameters of hypergeometric series are symmetric implies that a given formula may have several ways of applying, according to the position of the parameters. The variables `posX` are the positions of the "*special*" (upper or/and lower) parameters. [Let us remark that some formulas are totally symmetric and the position of parameters doesn't modify the result].

Several formulas could be factorised, but we prefer to keep them in expanded form to make their handling easy (each term can be manipulated individually).

7.1 Two-term relations

❑ **C01**:

$$ {}_rF_s\left[\begin{matrix}(A)\\(B)\end{matrix};z\right] \longrightarrow 1 + z\,\frac{\prod_{i=1}^{r} A_i}{\prod_{i=1}^{s} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1,(A+1)\\2,(B+1)\end{matrix};z\right] $$

❑ **C02** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix}1,(A)\\(B)\end{matrix};z\right] \longrightarrow \frac{1}{z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix}1,(A-1)\\(B-1)\end{matrix};z\right] - \frac{1}{z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)} $$

7.2 Three-term relations with one parameter

❑ **C14** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix}a,(A)\\(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a-1,(A)\\(B)\end{matrix};z\right] + z\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s} B_i}\,{}_rF_s\left[\begin{matrix}a,(A+1)\\(B+1)\end{matrix};z\right] $$

❑ **C15** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix}a,(A)\\(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a+1,(A)\\(B)\end{matrix};z\right] - z\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s} B_i}\,{}_rF_s\left[\begin{matrix}a+1,(A+1)\\(B+1)\end{matrix};z\right] $$

❑ **C16** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix}a,(A)\\(B)\end{matrix};z\right] \longrightarrow \frac{1}{z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix}a,(A-1)\\(B-1)\end{matrix};z\right] - \frac{1}{z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix}a-1,(A-1)\\(B-1)\end{matrix};z\right] $$

❑ **C17** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ (B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a-1,(A) \\ (B) \end{matrix};z\right] + z\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s} B_i}\,{}_rF_s\left[\begin{matrix} a,(A+1) \\ (B+1) \end{matrix};z\right] $$

❑ **C18** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ (B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a+1,(A) \\ (B) \end{matrix};z\right] - z\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s} B_i}\,{}_rF_s\left[\begin{matrix} a+1,(A+1) \\ (B+1) \end{matrix};z\right] $$

❑ **C19** with `pos[up]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ (B) \end{matrix};z\right] \longrightarrow \frac{1}{z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix} a,(A-1) \\ (B-1) \end{matrix};z\right] - \frac{1}{z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix} a-1,(A-1) \\ (B-1) \end{matrix};z\right] $$

❑ **C20** with `pos[low]=1`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ b,(B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} (A) \\ b+1,(B) \end{matrix};z\right] + \frac{z}{b\,(1+b)}\,\frac{\prod_{i=1}^{r} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_rF_s\left[\begin{matrix} (A+1) \\ b+2,(B+1) \end{matrix};z\right] $$

❑ **C21** with `pos[low]=1`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ b,(B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} (A) \\ b-1,(B) \end{matrix};z\right] - \frac{z}{(b-1)\,b}\,\frac{\prod_{i=1}^{r} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_rF_s\left[\begin{matrix} (A+1) \\ b+1,(B+1) \end{matrix};z\right] $$

❑ **C22** with `pos[low]=1`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix} (A) \\ b,(B) \end{matrix};z\right] \quad\longrightarrow\quad & \frac{(b-1)\,(b-2)}{z}\,\frac{\prod_{i=1}^{s-1} B_i}{\prod_{i=1}^{r} A_i}\,{}_rF_s\left[\begin{matrix} (A-1) \\ b-2,(B-1) \end{matrix};z\right] \\ - \quad & \frac{(b-1)\,(b-2)}{z}\,\frac{\prod_{i=1}^{s-1} B_i}{\prod_{i=1}^{r} A_i}\,{}_rF_s\left[\begin{matrix} (A-1) \\ b-1,(B-1) \end{matrix};z\right] \end{aligned} $$

❑ **C23** with `pos[low]=1`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ b,(B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} (A) \\ b+1,(B) \end{matrix};z\right] + \frac{z}{b\,(1+b)}\,\frac{\prod_{i=1}^{r} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_rF_s\left[\begin{matrix} (A+1) \\ b+2,(B+1) \end{matrix};z\right] $$

❑ **C24** with `pos[low]=1`:

$$ {}_rF_s\left[\begin{matrix}(A)\\ b,(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}(A)\\ b-1,(B)\end{matrix};z\right] - \frac{z}{(b-1)\,b}\,\frac{\prod_{i=1}^{r} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_rF_s\left[\begin{matrix}(A+1)\\ b+1,(B+1)\end{matrix};z\right] $$

❑ **C25** with `pos[low]=1`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix}(A)\\ b,(B)\end{matrix};z\right] \quad\longrightarrow\quad & \frac{(b-1)\,(b-2)}{z}\,\frac{\prod_{i=1}^{s-1} B_i}{\prod_{i=1}^{r} A_i}\,{}_rF_s\left[\begin{matrix}(A-1)\\ b-2,(B-1)\end{matrix};z\right]\\ - \quad & \frac{(b-1)\,(b-2)}{z}\,\frac{\prod_{i=1}^{s-1} B_i}{\prod_{i=1}^{r} A_i}\,{}_rF_s\left[\begin{matrix}(A-1)\\ b-1,(B-1)\end{matrix};z\right] \end{aligned} $$

7.3 Three-term relations with two parameters

❑ **C26** with `pos1[up]=1, pos2[up]=2`:

$$ {}_rF_s\left[\begin{matrix}a,b,(A)\\ (B)\end{matrix};z\right] \longrightarrow \frac{b}{b-a}\,{}_rF_s\left[\begin{matrix}a,b+1,(A)\\ (B)\end{matrix};z\right] + \frac{a}{a-b}\,{}_rF_s\left[\begin{matrix}a+1,b,(A)\\ (B)\end{matrix};z\right] $$

❑ **C27** with `pos1[up]=1, pos2[up]=2`:

$$ {}_rF_s\left[\begin{matrix}a,b,(A)\\ (B)\end{matrix};z\right] \longrightarrow \frac{a-b-1}{a-1}\,{}_rF_s\left[\begin{matrix}a-1,b,(A)\\ (B)\end{matrix};z\right] + \frac{b}{a-1}\,{}_rF_s\left[\begin{matrix}a-1,b+1,(A)\\ (B)\end{matrix};z\right] $$

❑ **C30** with `pos1[up]=1, pos2[up]=2`:

$$ {}_rF_s\left[\begin{matrix}a,b,(A)\\ (B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a-1,b+1,(A)\\ (B)\end{matrix};z\right] + (1-a+b)\,z\,\frac{\prod_{i=1}^{r-2} A_i}{\prod_{i=1}^{s} B_i}\,{}_rF_s\left[\begin{matrix}a,b+1,(A+1)\\ (B+1)\end{matrix};z\right] $$

❑ **C31** with `pos1[up]=1, pos2[up]=2`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix}a,b,(A)\\ (B)\end{matrix};z\right] \quad\longrightarrow\quad & \frac{1}{(b-a)\,z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-2}(A_i-1)}\,{}_rF_s\left[\begin{matrix}a,b-1,(A-1)\\ (B-1)\end{matrix};z\right]\\ - \quad & \frac{1}{(b-a)\,z}\,\frac{\prod_{i=1}^{s}(B_i-1)}{\prod_{i=1}^{r-2}(A_i-1)}\,{}_rF_s\left[\begin{matrix}a-1,b,(A-1)\\ (B-1)\end{matrix};z\right] \end{aligned} $$

❑ **C32** with `pos1[up]=1, pos2[up]=2`:

$$ {}_rF_s\left[\begin{matrix}a,b,(A)\\ (B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a-1,b-1,(A)\\ (B)\end{matrix};z\right] + (a+b-1)\,z\,\frac{\prod_{i=1}^{r-2} A_i}{\prod_{i=1}^{s} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a,b,a+b,(A+1)\\ -1+a+b,(B+1)\end{matrix};z\right] $$

❑ **C33** with `pos1[up]=1`, `pos2[up]=2`:

$$ {}_rF_s\left[\begin{matrix} a,b,(A) \\ (B) \end{matrix};z\right] \to {}_rF_s\left[\begin{matrix} a+1,b+1,(A) \\ (B) \end{matrix};z\right] - (1+a+b)z\frac{\prod_{i=1}^{r-2} A_i}{\prod_{i=1}^{s} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} a+1,b+1,a+b+2,(A+1) \\ a+b+1,(B+1) \end{matrix};z\right] $$

❑ **C34** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \longrightarrow \frac{b-1}{b-1-a}\,{}_rF_s\left[\begin{matrix} a,(A) \\ b-1,(B) \end{matrix};z\right] + \frac{a}{1+a-b}\,{}_rF_s\left[\begin{matrix} a+1,(A) \\ b,(B) \end{matrix};z\right] $$

❑ **C35** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \longrightarrow \frac{b-1}{a-1}\,{}_rF_s\left[\begin{matrix} a-1,(A) \\ b-1,(B) \end{matrix};z\right] + \frac{a-b}{a-1}\,{}_rF_s\left[\begin{matrix} a-1,(A) \\ b,(B) \end{matrix};z\right] $$

❑ **C36** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \longrightarrow \frac{b-a}{b}\,{}_rF_s\left[\begin{matrix} a,(A) \\ b+1,(B) \end{matrix};z\right] + \frac{a}{b}\,{}_rF_s\left[\begin{matrix} a+1,(A) \\ b+1,(B) \end{matrix};z\right] $$

❑ **C40** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a-1,(A) \\ b-1,(B) \end{matrix};z\right] + \frac{(b-a)\,z}{(b-1)\,b}\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_rF_s\left[\begin{matrix} a,(A+1) \\ b+1,(B+1) \end{matrix};z\right] $$

❑ **C41** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a+1,(A) \\ b+1,(B) \end{matrix};z\right] - \frac{(b-a)\,z}{b\,(1+b)}\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_rF_s\left[\begin{matrix} a+1,(A+1) \\ b+2,(B+1) \end{matrix};z\right] $$

❑ **C42** with `pos1[up]=1`, `pos2[dw]=1`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \quad &\longrightarrow \quad \frac{(b-2)\,(b-1)}{(b-a-1)\,z}\,\frac{\prod_{i=1}^{s-1}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix} a,(A-1) \\ b-1,(B-1) \end{matrix};z\right] \\ &- \quad \frac{(b-2)\,(b-1)}{(b-a-1)\,z}\,\frac{\prod_{i=1}^{s-1}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_rF_s\left[\begin{matrix} a-1,(A-1) \\ b-2,(B-1) \end{matrix};z\right] \end{aligned} $$

❑ **C43** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ b,(B) \end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a-1,(A) \\ b+1,(B) \end{matrix};z\right] + \frac{(a+b)\,z}{b\,(1+b)}\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} a,1+a+b,(A+1) \\ b+2,a+b,(B+1) \end{matrix};z\right] $$

❑ **C44** with `pos1[up]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}a,(A)\\ b,(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a+1,(A)\\ b-1,(B)\end{matrix};z\right] - \frac{(a+b)\,z}{(b-1)\,b}\frac{\prod_{i=1}^{r-1}A_i}{\prod_{i=1}^{s-1}B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1+a,a+b+1,(A+1)\\ b+1,a+b,(B+1)\end{matrix};z\right] $$

❑ **C45** with `pos1[dw]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}(A)\\ a,b,(B)\end{matrix};z\right] \longrightarrow \frac{a-1}{a-b}\,{}_rF_s\left[\begin{matrix}(A)\\ -1+a,b,(B)\end{matrix};z\right] + \frac{b-1}{(b-a)}\,{}_rF_s\left[\begin{matrix}(A)\\ a,-1+b,(B)\end{matrix};z\right] $$

❑ **C46** with `pos1[dw]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}(A)\\ a,b,(B)\end{matrix};z\right] \longrightarrow \frac{b-1}{a}\,{}_rF_s\left[\begin{matrix}(A)\\ a+1,b-1,(B)\end{matrix};z\right] + \frac{1+a-b}{a}\,{}_rF_s\left[\begin{matrix}(A)\\ a+1,b,(B)\end{matrix};z\right] $$

❑ **C49** with `pos1[dw]=1`, `pos2[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}(A)\\ a,b,(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix}(A)\\ a+1,b-1,(B)\end{matrix};z\right] + \frac{(b-a-1)\,z}{a\,(1+a)\,(b-1)\,b}\frac{\prod_{i=1}^{r}A_i}{\prod_{i=1}^{s-2}B_i}\,{}_rF_s\left[\begin{matrix}(A+1)\\ a+2,b+1,(B+!)\end{matrix};z\right] $$

❑ **C50** with `pos1[dw]=1`, `pos2[dw]=1`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix}(A)\\ a,b,(B)\end{matrix};z\right] &\longrightarrow \frac{(a-2)\,(a-1)\,(b-2)\,(b-1)}{(b-a)\,z}\frac{\prod_{i=1}^{s-2}(B_i-1)}{\prod_{i=1}^{r}(A_i-1)}\,{}_rF_s\left[\begin{matrix}(A-1)\\ -2+a,b-1,(B-1)\end{matrix};z\right] \\ &- \frac{(a-2)\,(a-1)\,(b-2)\,(b-1)}{(b-a)\,z}\frac{\prod_{i=1}^{s-2}(B_i-1)}{\prod_{i=1}^{r}(A_i-1)}\,{}_rF_s\left[\begin{matrix}(A-1)\\ a-1,b-2,(B-1)\end{matrix};z\right] \end{aligned} $$

❑ **C51** with `pos1[dw]=1`, `pos2[dw]=1`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix}(A)\\ a,b,(B)\end{matrix};z\right] &\longrightarrow {}_rF_s\left[\begin{matrix}(A)\\ a+1,b+1,(B)\end{matrix};z\right] \\ &+ \frac{(1+a+b)\,z}{a\,(1+a)\,b\,(1+b)}\frac{\prod_{i=1}^{r}A_i}{\prod_{i=1}^{s-2}B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a+b+2,(A+1)\\ a+2,b+2,a+b+1,(B+1)\end{matrix};z\right] \end{aligned} $$

❑ **C52** with `pos1[dw]=1`, `pos2[dw]=1`:

$$ \begin{aligned} {}_rF_s\left[\begin{matrix}(A)\\ a,b,(B)\end{matrix};z\right] &\longrightarrow {}_rF_s\left[\begin{matrix}(A)\\ a-1,b-1,(B)\end{matrix};z\right] \\ &- \frac{(a+b-1)\,z}{(a-1)\,a\,(b-1)\,b}\frac{\prod_{i=1}^{r}A_i}{\prod_{i=1}^{s-2}B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a+b,(A+1)\\ a+1,1+b,b-a+1,(B+1)\end{matrix};z\right] \end{aligned} $$

7.4 Three-term relations with three parameters

❑ **C53** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`:

$$
{}_rF_s\left[\begin{matrix}a,b,c,(A)\\(B)\end{matrix};z\right] \longrightarrow \frac{b\,(c-a-1)}{(b-a)\,(c-1)}\,{}_rF_s\left[\begin{matrix}a,b+1,c-1,(A)\\(B)\end{matrix};z\right] + \frac{a\,(c-b-1)}{(a-b)\,(c-1)}\,{}_rF_s\left[\begin{matrix}a+1,b,c-1,(A)\\(B)\end{matrix};z\right]
$$

❑ **C54** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`:

$$
{}_rF_s\left[\begin{matrix}a,b,(A)\\c,(B)\end{matrix};z\right] \longrightarrow \frac{b\,(c-a)}{(b-a)\,c}\,{}_rF_s\left[\begin{matrix}a,b+1,(A)\\c+1,(B)\end{matrix};z\right] + \frac{a\,(c-b)}{(a-b)\,c)}\,{}_rF_s\left[\begin{matrix}a+1,b,(A)\\c+1,(B)\end{matrix};z\right]
$$

❑ **C55** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`:

$$
{}_rF_s\left[\begin{matrix}a,b,(A)\\c,(B)\end{matrix};z\right] \longrightarrow \frac{(1-a+b)\,(c-1)}{(a-1)\,(1+b-c)}\,{}_rF_s\left[\begin{matrix}a-1,b,(A)\\c-1,(B)\end{matrix};z\right] + \frac{b\,(c-a)}{(a-1)\,(c-b-1))}\,{}_rF_s\left[\begin{matrix}a-1,b+1,(A)\\c,(B)\end{matrix};z\right]
$$

❑ **C56** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a,(A)\\b,c,(B)\end{matrix};z\right] \longrightarrow \frac{(b-1)\,(c-a)}{(b-a-1)\,c}\,{}_rF_s\left[\begin{matrix}a,(A)\\b-1,c+1,(B)\end{matrix};z\right] + \frac{a\,(1-b+c)}{(1+a-b)\,c}\,{}_rF_s\left[\begin{matrix}a+1,(A)\\b,c+1,(B)\end{matrix};z\right]
$$

❑ **C57** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a,(A)\\b,c,(B)\end{matrix};z\right] \longrightarrow \frac{(b-1)\,(c-a)}{(a-1)\,(c-b)}\,{}_rF_s\left[\begin{matrix}a-1,(A)\\b-1,c,(B)\end{matrix};z\right] + \frac{(b-a)\,(c-1)}{(a-1)\,(b-c)}\,{}_rF_s\left[\begin{matrix}a-1,(A)\\b,c-1,(B)\end{matrix};z\right]
$$

❑ **C58** with `pos1[dw]=1`, `pos2[dw]=2`, `pos3[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}(A)\\a,b,c,(B)\end{matrix};z\right] \longrightarrow \frac{(a-1)\,(1-b+c)}{(a-b)\,c}\,{}_rF_s\left[\begin{matrix}(A)\\a-1,b,c+1,(B)\end{matrix};z\right] + \frac{(b-1)\,(1-a+c)}{(b-a)\,c}\,{}_rF_s\left[\begin{matrix}(A)\\a,b-1,c+1,(B)\end{matrix};z\right]
$$

7.5 Three-term relations with four parameters

❑ **C59** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`:

$$
{}_rF_s\left[\begin{matrix}a,2-a+c,b,c-b,(A)\\(B)\end{matrix};z\right] \longrightarrow \frac{(a-b-1)\,(1-a-b+c)}{(a-1)\,(1-a+c)}\,{}_rF_s\left[\begin{matrix}a-1,1-a+c,b,c-b,(A)\\(B)\end{matrix};z\right] + \frac{b\,(c-b)}{(a-1)\,(1-a+c)}\,{}_rF_s\left[\begin{matrix}a-1,1-a+c,b+1,1-b+c,(A)\\(B)\end{matrix};z\right]
$$

❑ **C60** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`:

$$
{}_rF_s\left[\begin{matrix}a, c-a, b, c-b, (A)\\ (B)\end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)}{(a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix}a+1, 1-a+c, b, c-b, (A)\\ (B)\end{matrix}; z\right] - \frac{b\,(c-b)}{(a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix}a, c-a, b+1, 1-b+c, (A)\\ (B)\end{matrix}; z\right]
$$

❑ **C61** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`:

$$
{}_rF_s\left[\begin{matrix}a, c-a, b, c-b, (A)\\ (B)\end{matrix}; z\right] \longrightarrow \frac{a\,b}{(c-a-1)\,(c-b-1)}\,{}_rF_s\left[\begin{matrix}a+1, c-a-1, b+1, -1-b+c, (A)\\ (B)\end{matrix}; z\right] + \frac{(c-1)\,(-1-a-b+c)}{(c-a-1)\,(c-b-1)}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1+\frac{c-1}{2}, a, -1-a+c, b, c-b-1, (A)\\ \frac{c-1}{2}, (B)\end{matrix}; z\right]
$$

❑ **C62** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`:

$$
{}_rF_s\left[\begin{matrix}a, c-a, b, 2-b+c, (A)\\ (B)\end{matrix}; z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a+1, 1-a+c, b-1, 1+c-b, (A)\\ (B)\end{matrix}; z\right] + (a-b+1)\,(1+c)\,(c-a-b+1)\,z\,\frac{\prod_{i=1}^{r-4} A_i}{\prod_{i=1}^{s} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a+1, 1-a+c, b, 2-b+c, c+2, (A+1)\\ c+1, (B+1)\end{matrix}; z\right]
$$

❑ **C63** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`, and `d` (additional parameter at the right hand side of the relation):

$$
{}_rF_s\left[\begin{matrix}a, 2-a+c, b, c-b, (A)\\ (B)\end{matrix}; z\right] \longrightarrow \frac{b\,(c-b)\,(a-d-1)\,(1-a+c-d)}{(a-1)\,(1-a+c)\,(b-d)\,(c-b-d)}\,{}_rF_s\left[\begin{matrix}a-1, 1-a+c, b+1, 1-b+c, (A)\\ (B)\end{matrix}; z\right] + \frac{(1-a+b)\,(1-a-b+c)\,(c-d)\,d}{(a-1)\,(1-a+c)\,(b-d)\,(c-b-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix}a-1, 1-a+c, b, -b+c, d+1, 1+c-d, (A)\\ d, c-d, (B)\end{matrix}; z\right]
$$

❑ **C64** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$
{}_rF_s\left[\begin{matrix}a, 1-a+c, b, (A)\\ c-b, (B)\end{matrix}; z\right] \longrightarrow \frac{(a-b-1)\,(c-a-b)}{(a-1)\,(c-a)}\,{}_rF_s\left[\begin{matrix}a-1, c-a, b, (A)\\ c-b, (B)\end{matrix}; z\right] + \frac{b\,(c-b-1)}{(a-1)\,(c-a)}\,{}_rF_s\left[\begin{matrix}a-1, c-a, b+1, (A)\\ -1-b+c, (B)\end{matrix}; z\right]
$$

❑ **C65** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$
{}_rF_s\left[\begin{matrix}a, c-a, b, (A)\\ 1-b+c, (B)\end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)}{(b-1)\,(1-b+c)}\,{}_rF_s\left[\begin{matrix}a+1, 1-a+c, b-1, (A)\\ 2-b+c, (B)\end{matrix}; z\right] - \frac{(1+a-b)\,(1-a-b+c)}{(b-1)\,(1-b+c)}\,{}_rF_s\left[\begin{matrix}a, c-a, b+1, (A)\\ 2-b+c, (B)\end{matrix}; z\right]
$$

❑ **C66** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}a, c-a, b, (A)\\ 1-b+c, (B)\end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)}{(a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix}a+1, 1-a+c, b, (A)\\ 1-b+c, (B)\end{matrix}; z\right] - \frac{b\,(c-b)}{(a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix}a, c-a, b+1, (A)\\ c-b, (B)\end{matrix}; z\right] $$

❑ **C67** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}a, 1-a+c, b, (A)\\ c-b, (B)\end{matrix}; z\right] \longrightarrow \frac{a\,b}{(c-a)\,(c-b)}\,{}_rF_s\left[\begin{matrix}a+1, c-a, b+1, (A)\\ 1-b+c, (B)\end{matrix}; z\right] + \frac{c\,(c-a-b)}{(c-a)\,(c-b)}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1+\frac{c}{2}, a, c-a, b, (A)\\ \frac{c}{2}, 1-b+c, (B)\end{matrix}; z\right] $$

❑ **C68** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}a, c-a, b, (A)\\ 1-b+c, (B)\end{matrix}; z\right] \longrightarrow \frac{(c-a)\,(c-b)}{(a-1)\,(b-1)}\,{}_rF_s\left[\begin{matrix}a-1, 1-a+c, b-1, (A)\\ c-b, (B)\end{matrix}; z\right] - \frac{(c-1)\,(1-a-b+c)}{(a-1)\,(b-1)}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1+\frac{c-1}{2}, -1+a, c-a, b-1, (A)\\ \frac{c-1}{2}, 1-b+c, (B)\end{matrix}; z\right] $$

❑ **C69** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}a, c-a, b, (A)\\ 1-b+c, (B)\end{matrix}; z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a+1, 1-a+c, b-1, (A)\\ 2-b+c, (B)\end{matrix}; z\right] + \frac{(1+a-b)\,(1+c)\,(1-a-b+c)\,z}{(1-b+c)\,(2-b+c)}\,\frac{\prod_{i=1}^{r-3} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a+1, 1-a+c, b, c+2, (A+1)\\ 3-b+c, c+1, (B+1)\end{matrix}; z\right] $$

❑ **C70** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$ {}_rF_s\left[\begin{matrix}a, 1-a+c, b, (A)\\ c-b, (B)\end{matrix}; z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a-1, c-a, b+1, (A)\\ -1-b+c, (B)\end{matrix}; z\right] - \frac{(a-b-1)\,c\,(c-a-b)\,z}{(c-b-1)\,(c-b)}\,\frac{\prod_{i=1}^{r-3} A_i}{\prod_{i=1}^{s-1} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a, 1-a+c, b+1, c+1, (A+1)\\ 1-b+c, c, (B+1)\end{matrix}; z\right] $$

❑ **C71** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, and `d` (additional parameter at the right hand side of the relation):

$$ {}_rF_s\left[\begin{matrix}a, 1-a+c, b, (A)\\ c-b, (B)\end{matrix}; z\right] \longrightarrow \frac{b\,(c-b-1)\,(a-d-1)\,(c-a-d)}{(a-1)\,(c-a)\,(b-d)\,(c-b-1-d)}\,{}_rF_s\left[\begin{matrix}a-1, c-a, b+1, (A)\\ -1-b+c, (B)\end{matrix}; z\right] + \frac{(1-a+b)\,(c-a-b)\,(c-d-1)\,d}{(a-1)\,(c-a)\,(b-d)\,(c-b-1-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix}a-1, c-a, b, d+1, c-d, (A)\\ c-b, d, c-d-1, (B)\end{matrix}; z\right] $$

❑ **C72** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, and `d` (additional parameter at the right hand side of the relation):

$$
{}_rF_s\left[\begin{matrix}a, c-a, b, (A)\\ 1-b+c, (B)\end{matrix};z\right] \longrightarrow \frac{a\,(c-a)\,(b-d-1)\,(1-b+c-d)}{(b-1)\,(1-b+c)\,(a-d)\,(c-a-d)}\,{}_rF_s\left[\begin{matrix}a+1, 1-a+c, b-1, (A)\\ 2-b+c, (B)\end{matrix};z\right]
$$
$$
-\frac{(b-a-1)\,(1-a-b+c)\,(c-d)\,d}{(b-1)\,(1-b+c)\,(a-d)\,(c-a-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix}a, c-a, b-1, d+1, 1+c-d, (A)\\ 2-b+c, d, c-d, (B)\end{matrix};z\right]
$$

❑ **C73** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`:

$$
{}_rF_s\left[\begin{matrix}a, b, a+b, (A)\\ b-a+1, (B)\end{matrix};z\right] \longrightarrow \frac{1}{(a+b-1)\,z}\,\frac{\prod_{i=1}^{s-1}(B_i-1)}{\prod_{i=1}^{r-3}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a, b, (A-1)\\ (B-1)\end{matrix};z\right]
$$
$$
-\frac{1}{(a+b-1)\,z}\,\frac{\prod_{i=1}^{s-1}(B_i-1)}{\prod_{i=1}^{r-3}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a-1, b-1, (A-1)\\ (B-1)\end{matrix};z\right]
$$

❑ **C74** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a, b, (A)\\ c-a, c-b, (B)\end{matrix};z\right] \longrightarrow \frac{(a-b-1)\,(c-a-b)}{(a-1)\,(c-a)}\,{}_rF_s\left[\begin{matrix}a-1, b, (A)\\ 1-a+c, -b+c, (B)\end{matrix};z\right]
$$
$$
+\frac{b\,(c-b-1)}{(a-1)\,(c-a)}\,{}_rF_s\left[\begin{matrix}a-1, b-1, (A)\\ 1-a+c, c-b-1, (B)\end{matrix};z\right]
$$

❑ **C75** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a, b, (A)\\ c-a, c-b, (B)\end{matrix};z\right] \longrightarrow \frac{a\,(c-a-1)}{(a-b)\,(-1-a-b+c)}\,{}_rF_s\left[\begin{matrix}a+1, b, (A)\\ c-a-1, -b+c, (B)\end{matrix};z\right]
$$
$$
-\frac{b\,(c-b-1)}{(a-b)\,(-1-a-b+c)}\,{}_rF_s\left[\begin{matrix}a, b+1, (A)\\ c-a, -1-b+c, (B)\end{matrix};z\right]
$$

❑ **C76** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a, b, (A)\\ c-a, c-b, (B)\end{matrix};z\right] \longrightarrow \frac{a\,b}{(c-a)\,(c-b)}\,{}_rF_s\left[\begin{matrix}a+1, b+1, (A)\\ 1-a+c, 1-b+c, (B)\end{matrix};z\right]
$$
$$
+\frac{c\,(c-a-b)}{(c-a)\,(c-b)}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1+\frac{c}{2}, a, b, (A)\\ \frac{c}{2}, 1-a+c, 1-b+c, (B)\end{matrix};z\right]
$$

❑ **C77** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a, b, (A)\\ c-a, c-b, (B)\end{matrix};z\right] \longrightarrow \frac{(c-a-1)\,(c-b-1)}{(a-1)\,(b-1)}\,{}_rF_s\left[\begin{matrix}a-1, b-1, (A)\\ c-a-1, c-b-1, (B)\end{matrix};z\right]
$$
$$
-\frac{(c-2)\,(c-a-b)}{(a-1)\,(b-1)}\,{}_{r+1}F_{s+1}\left[\begin{matrix}1+\frac{c-2}{2}, -1+a, b-1, (A)\\ \frac{c-2}{2}, c-a, c-b, (B)\end{matrix};z\right]
$$

❑ **C78** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$ {}_rF_s\left[\begin{matrix} a, b, (A) \\ c-a, c-b, (B) \end{matrix}; z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a+1, b-1, (A) \\ c-a-1, 1-b+c, (B) \end{matrix}; z\right] $$

$$ + \frac{(1+a-b)\,c\,(c-a-b)\,z}{(c-a-1)\,(c-a)\,(c-b)\,(1-b+c)}\,\frac{\prod_{i=1}^{r-2} A_i}{\prod_{i=1}^{s-2} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} a+1, b, c+1, (A+1) \\ 1-a+c, 2-b+c, c, (B+1) \end{matrix}; z\right] $$

❑ **C79** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`, and `d` (additional parameter at the right hand side of the relation):

$$ {}_rF_s\left[\begin{matrix} a, b, (A) \\ c-a, c-b, (B) \end{matrix}; z\right] \longrightarrow \frac{b\,(c-b-1)\,(a-d-1)\,(c-a-d)}{(a-1)\,(c-a)\,(b-d)\,(c-b-1-d)}\,{}_rF_s\left[\begin{matrix} a-1, b+1, (A) \\ 1-a+c, c-b-1, (B) \end{matrix}; z\right] $$

$$ + \frac{(1-a+b)\,(c-a-b)\,(c-d-1)\,d}{(a-1)\,(c-a)\,(b-d)\,(c-b-1-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix} a-1, b, d+1, c-d, (A) \\ 1-a+c, c-b, d, c-d-1, (B) \end{matrix}; z\right] $$

❑ **C80** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$ {}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b, c-b, (B) \end{matrix}; z\right] \longrightarrow \frac{(b-1)\,(c-b-1)}{(a-1)\,(c-a-1)}\,{}_rF_s\left[\begin{matrix} a-1, c-a-1, (A) \\ b-1, c-b-1, (B) \end{matrix}; z\right] $$

$$ + \frac{(a-b)\,(c-a-b)}{(a-1)\,(c-a-1)}\,{}_rF_s\left[\begin{matrix} a-1, b, c-a-b, (A) \\ b, -b+c, (B) \end{matrix}; z\right] $$

❑ **C81** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$ {}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b, c-b, (B) \end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)}{b\,(c-b)}\,{}_rF_s\left[\begin{matrix} a+1, 1-a+c, (A) \\ b+1, 1-b+c, (B) \end{matrix}; z\right] $$

$$ - \frac{(a-b)\,(c-a-b)}{b\,(c-b)}\,{}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b+1, 1-b+c, (B) \end{matrix}; z\right] $$

❑ **C82** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$ {}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b, 2-b+c, (B) \end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)}{(1+a-b)\,(1-a-b+c)}\,{}_rF_s\left[\begin{matrix} a+1, 1-a+c, (A) \\ b, 2-b+c, (B) \end{matrix}; z\right] $$

$$ - \frac{(b-1)\,(1-b+c)}{(1+a-b)\,(1-a-b+c)}\,{}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b-1, 1-b+c, (B) \end{matrix}; z\right] $$

❑ **C83** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$ {}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b, c-b, (B) \end{matrix}; z\right] \longrightarrow \frac{a\,(b-1)}{(c-a-1)\,(c-b)}\,{}_rF_s\left[\begin{matrix} a+1, -1-a+c, (A) \\ b-1, 1-b+c, (B) \end{matrix}; z\right] $$

$$ + \frac{(c-1)\,(c-a-b)}{(c-a-1)\,(c-b)}\,{}_{r+1}F_{s+1}\left[\begin{matrix} 1+\frac{c-1}{2}, a, -1-a+c, (A) \\ \frac{c-1}{2}, b, 1-b+c, (B) \end{matrix}; z\right] $$

❑ **C84** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$ {}_rF_s\left[\begin{matrix} a, c-a, (A) \\ b, c-b, (B) \end{matrix}; z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a+1, -1-a+c, (A) \\ b+1, 1-b+c, (B) \end{matrix}; z\right] $$

$$ + \frac{(a-b)\,(1+c)\,(c-a-b)\,z}{b\,(1+b)\,(c-b)\,(1-b+c)}\,\frac{\prod_{i=1}^{r-2} A_i}{\prod_{i=1}^{s-2} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} a+1, 1-a+c, c+2, (A+1) \\ b+2, 2-b+c, c+1, (B+1) \end{matrix}; z\right] $$

❑ **C85** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}a, c-a, (A)\\ b, c-b, (B)\end{matrix}; z\right] \longrightarrow {}_rF_s\left[\begin{matrix}a-1, c-a-1, (A)\\ b-1, -1-b+c, (B)\end{matrix}; z\right]
$$

$$
-\frac{(a-b)\,(c-1)\,(c-a-b)\,z}{(b-1)\,b\,(c-b-1)\,(c-b)}\,\frac{\prod\limits_{i=1}^{r-2} A_i}{\prod\limits_{i=1}^{s-2} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix}a, c-a, c, (A+1)\\ 1+b, 1-b+c, c-1, (B+1)\end{matrix}; z\right]
$$

❑ **C86** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`, and `d` (additional parameter at the right hand side of the relation):

$$
{}_rF_s\left[\begin{matrix}a, c-a, (A)\\ b, c-b, (B)\end{matrix}; z\right] \longrightarrow \frac{(b-1)\,(c-b-1)\,(a-d-1)\,(c-a-1-d)}{(a-1)\,(c-a-1)\,(b-d-1)\,(c-b-1-d)}\,{}_rF_s\left[\begin{matrix}a-1, c-a-1, (A)\\ b-1, c-b-1, (B)\end{matrix}; z\right]
$$

$$
+\frac{(b-a)\,(c-a-b)\,(c-d-2)\,d}{(a-1)\,(c-a-1)\,(b-d-1)\,(c-b-1-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix}a-1, c-a-1, d+1, c-d-1, (A)\\ b, c-b, d, c-d-2, (B)\end{matrix}; z\right]
$$

❑ **C87** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`, and `d` (additional parameter at the right hand side of the relation):

$$
{}_rF_s\left[\begin{matrix}a, c-a, (A)\\ b, c-b, (B)\end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)\,(b-d)\,(c-b-d)}{b\,(c-b)\,(a-d)\,(c-a-d)}\,{}_rF_s\left[\begin{matrix}a+1, 1-a+c, (A)\\ b+1, 1-b+c, (B)\end{matrix}; z\right]
$$

$$
-\frac{(b-a)\,(c-a-b)\,(c-d)\,d}{b\,(c-b)\,(a-d)\,(c-a-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix}a, c-a, d+1, 1+c-d, (A)\\ b+1, 1-b+c, d, c-d, (B)\end{matrix}; z\right]
$$

❑ **C88** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`,

$$
{}_rF_s\left[\begin{matrix}a, a+b, (A)\\ b+1, b-a+1, (B)\end{matrix}; z\right] \longrightarrow \frac{(b-1)\,b}{(a+b-1)\,z}\,\frac{\prod\limits_{i=1}^{s-2}(B_i-1)}{\prod\limits_{i=1}^{r-2}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a, (A-1)\\ b-1, (B-1)\end{matrix}; z\right]
$$

$$
-\frac{(b-1)\,b}{(a+b-1)\,z}\,\frac{\prod\limits_{i=1}^{s-2}(B_i-1)}{\prod\limits_{i=1}^{r-2}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a-1, (A-1)\\ b, (B-1)\end{matrix}; z\right]
$$

❑ **C89** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}a, (A)\\ c-a, b, 1-b+c, (B)\end{matrix}; z\right] \longrightarrow \frac{(b-1)\,(c-b)}{(a-1)\,(c-a)}\,{}_rF_s\left[\begin{matrix}a-1, (A)\\ 1-a+c, b, 1-b+c, (B)\end{matrix}; z\right]
$$

$$
+\frac{(a-b)\,(1-a-b+c)}{(a-1)\,(c-a)}\,{}_rF_s\left[\begin{matrix}a-1, (A)\\ 1-a+c, b, 1-b+c, (B)\end{matrix}; z\right]
$$

❑ **C90** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}a, (A)\\ 1-a+c, b, c-b, (B)\end{matrix}; z\right] \longrightarrow \frac{a\,(c-a)}{b\,(c-b)}\,{}_rF_s\left[\begin{matrix}a+1, (A)\\ c-a, b+1, 1-b+c, (B)\end{matrix}; z\right]
$$

$$
-\frac{(a-b)\,(c-a-b)}{b\,(c-b)}\,{}_rF_s\left[\begin{matrix}a, (A)\\ 1-a+c, b+1, 1-b+c, (B)\end{matrix}; z\right]
$$

❑ **C91** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix} a,(A) \\ c-a,b,1-b+c,(B)\end{matrix};z\right] \longrightarrow \frac{a\,(c-a-1)}{(1+a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix} a+1,(A) \\ c-a-1,b,1-b+c,(B)\end{matrix};z\right] - \frac{(b-1)\,(c-b)}{(1+a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix} a,(A) \\ c-a,b-1,c-b,(B)\end{matrix};z\right]
$$

❑ **C92** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix} a,(A) \\ c-a,b,1-b+c,(B)\end{matrix};z\right] \longrightarrow \frac{a\,(b-1)}{(c-a)\,(1-b+c)}\,{}_rF_s\left[\begin{matrix} a+1,(A) \\ 1-a+c,b-1,2-b+c,(B)\end{matrix};z\right] + \frac{c\,(1-a-b+c)}{(c-a)\,(1-b+c)}\,{}_{r+1}F_{s+1}\left[\begin{matrix} 1+\frac{c}{2},a,(A) \\ \frac{c}{2},1-a+c,b,2-b+c,(B)\end{matrix};z\right]
$$

❑ **C93** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix} a,(A) \\ 1-a+c,b,c-b,(B)\end{matrix};z\right] \longrightarrow \frac{(c-a)\,(c-b-1)}{(a-1)\,b}\,{}_rF_s\left[\begin{matrix} a-1,(A) \\ c-a,b+1,-1-b+c,(B)\end{matrix};z\right] - \frac{(c-1)\,(c-a-b)}{(a-1)\,b}\,{}_{r+1}F_{s+1}\left[\begin{matrix} 1+\frac{c-1}{2},-1+a,(A) \\ \frac{c-1}{2},1-a+c,b+1,c-b,(B)\end{matrix};z\right]
$$

❑ **C94** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix} a,(A) \\ 1-a+c,b,c-b,(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a+1,(A) \\ c-a,b+1,-1-b+c,(B)\end{matrix};z\right]
$$

$$
+\frac{(a-b)\,(1+c)\,(c-a-b)\,z}{b\,(1+b)\,(c-a)\,(1-a+c)\,(c-b)\,(1-b+c)}\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s-3} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} a+1,c+2,(A+1) \\ 2-a+c,b+2,2-b+c,c+1,(B+1)\end{matrix};z\right]
$$

❑ **C95** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$
{}_rF_s\left[\begin{matrix} a,(A) \\ c-a,b,1-b+c,(B)\end{matrix};z\right] \longrightarrow {}_rF_s\left[\begin{matrix} a-1,(A) \\ 1-a+c,b-1,-b+c,(B)\end{matrix};z\right]
$$

$$
-\frac{(a-b)\,c\,(1-a-b+c)\,z}{(b-1)\,b\,(c-a)\,(1-a+c)\,(c-b)\,(1-b+c)}\,\frac{\prod_{i=1}^{r-1} A_i}{\prod_{i=1}^{s-3} B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} a,c+1,(A+1) \\ 2-a+c,b+1,2-b+c,c,(B+1)\end{matrix};z\right]
$$

❑ **C96** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`, and `d` (additional parameter at the right hand side of the relation):

$$
{}_rF_s\left[\begin{matrix} a,(A) \\ c-a,b,1-b+c,(B)\end{matrix};z\right] \to \frac{(b-1)\,(c-b)\,(a-d-1)\,(c-a-d)}{(a-1)\,(c-a)\,(b-d-1)\,(c-b-d)}\,{}_rF_s\left[\begin{matrix} a-1,(A) \\ 1-a+c,b-1,c-b,(B)\end{matrix};z\right] + \frac{(b-a)\,(1-a-b+c)\,(c-d-1)\,d}{(a-1)\,(c-a)\,(b-d-1)\,(c-b-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix} a-1,d+1,c-d,(A) \\ 1-a+c,b,1-b+c,d,c-d-1,(B)\end{matrix};z\right]
$$

❑ **C97** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`, and `d` (additional parameter at the right hand side of the relation):

$$ {}_rF_s\left[\begin{matrix} a,(A) \\ 1-a+c,b,c-b,(B)\end{matrix};z\right] \longrightarrow \frac{a\,(c-a)\,(b-d)\,(c-b-d)}{b\,(c-b)\,(a-d)\,(c-a-d)}\,{}_rF_s\left[\begin{matrix} a+1,(A) \\ c-a,b+1,1-b+c,(B)\end{matrix};z\right] - \frac{(b-a)\,(c-a-b)\,(c-d)\,d}{b\,(c-b)\,(a-d)\,(c-a-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix} a,d+1,1+c-d,(A) \\ 1-a+c,b+1,1-b+c,d,c-d,(B)\end{matrix};z\right] $$

❑ **C98** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`:

$$ {}_rF_s\left[\begin{matrix} a+b,(A) \\ a+1,b+1,b-a+1,(B)\end{matrix};z\right] \longrightarrow \frac{(a-1)\,a\,(b-1)\,b}{(a+b-1)\,z}\,\frac{\prod_{i=1}^{s-3}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix} (A-1) \\ a-1,b-1,(B-1)\end{matrix};z\right] - \frac{(a-1)\,a\,(b-1)\,b}{(a+b-1)\,z}\,\frac{\prod_{i=1}^{s-3}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix} (A-1) \\ a,b,(B-1)\end{matrix};z\right] $$

❑ **C99** with `pos1[dw]=1`, `pos2[dw]=2`, `pos3[dw]=3`, `pos4[dw]=4`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ a,c-a,b,2-b+c,(B)\end{matrix};z\right] \rightarrow \frac{(b-1)\,(1-b+c)}{a\,(c-a)}\,{}_rF_s\left[\begin{matrix} (A) \\ a+1,1-a+c,b-1,1-b+c,(B)\end{matrix};z\right] + \frac{(1+a-b)\,(1-a-b+c)}{a\,(c-a)}\,{}_rF_s\left[\begin{matrix} (A) \\ a+1,1-a+c,b,2-b+c,(B)\end{matrix};z\right] $$

❑ **C100** with `pos1[dw]=1`, `pos2[dw]=2`, `pos3[dw]=3`, `pos4[dw]=4`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ a,c-a,b,c-b,(B)\end{matrix};z\right] \longrightarrow \frac{(a-1)\,(c-a-1)}{(a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix} (A) \\ a-1,c-a-1,b,-b+c,(B)\end{matrix};z\right] - \frac{(b-1)\,(c-b-1)}{(a-b)\,(c-a-b)}\,{}_rF_s\left[\begin{matrix} (A) \\ a,c-a,b-1,-1-b+c,(B)\end{matrix};z\right] $$

❑ **C101** with `pos1[dw]=1`, `pos2[dw]=2`, `pos3[dw]=3`, `pos4[dw]=4`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ a,c-a,b,c-b,(B)\end{matrix};z\right] \longrightarrow \frac{(a-1)\,(b-1)}{(c-a)\,(c-b)}\,{}_rF_s\left[\begin{matrix} (A) \\ a-1,1-a+c,b-1,1-b+c,(B)\end{matrix};z\right] + \frac{(c-1)\,(1-a-b+c)}{(c-a)\,(c-b)}\,{}_{r+1}F_{s+1}\left[\begin{matrix} 1+\frac{c-1}{2},(A) \\ \frac{c-1}{2},a,1-a+c,b,1-b+c,(B)\end{matrix};z\right] $$

❑ **C102** with `pos1[dw]=1`, `pos2[dw]=2`, `pos3[dw]=3`, `pos4[dw]=4`:

$$ {}_rF_s\left[\begin{matrix} (A) \\ a,2-a+c,b,c-b,(B)\end{matrix};z\right] \rightarrow {}_rF_s\left[\begin{matrix} (A) \\ a-1,1-a+c,b+1,1-b+c,(B)\end{matrix};z\right] + \frac{(a-b-1)\,(1+c)\,(1-a-b+c)\,z}{(a-1)\,a\,b\,(1+b)\,(1-a+c)\,(2-a+c)\,(c-b)\,(1-b+c)} \times \frac{\prod_{i=1}^{r}A_i}{\prod_{i=1}^{s-4}B_i}\,{}_{r+1}F_{s+1}\left[\begin{matrix} c+2,(A+1) \\ a-1,3-a+c,b+2,2-b+c,c+1,(B+1)\end{matrix};z\right] $$

❑ **C103** with `pos1[dw]=1`, `pos2[dw]=2`, `pos3[dw]=3`, `pos4[dw]=4`, and `d` (additional parameter at the right hand side of the relation):

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}(A)\\ a,c-a,b,2-b+c,(B)\end{matrix};z\right]\\
&\longrightarrow \frac{(b-1)\,(1-b+c)\,(a-d)\,(c-a-d)}{a\,(c-a)\,(b-d-1)\,(1-b+c-d)}\,{}_rF_s\left[\begin{matrix}(A)\\ a+1,1-a+c,b-1,1-b+c,(B)\end{matrix};z\right]\\
&+\frac{(b-a-1)\,(1-a-b+c)\,(c-d)\,d}{a\,(c-a)\,(b-d-1)\,(1-b+c-d)}\,{}_{r+2}F_{s+2}\left[\begin{matrix}d+1,1+c-d,(A)\\ 1+a,1-a+c,b,2-b+c,d,c-d,(B)\end{matrix};z\right]
\end{aligned}
$$

7.6 Three-term relations with six parameters

❑ **C104** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`, `pos5[up]=5`, `pos6[dw]=1`:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}a,c-a,b,c-b,c,(A)\\ c-1,(B)\end{matrix};z\right]\\
&\longrightarrow \frac{1}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-1}(B_i-1)}{\prod_{i=1}^{r-5}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a-1,c-a-1,b,-b+c,(A-1)\\ (B-1)\end{matrix};z\right]\\
&- \frac{1}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-1}(B_i-1)}{\prod_{i=1}^{r-5}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,c-a,b-1,-1-b+c,(A-1)\\ (B-1)\end{matrix};z\right]
\end{aligned}
$$

❑ **C110** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`, `pos5[up]=5`, `pos6[dw]=1`:

$$
\begin{aligned}
{}_rF_s\left[\begin{matrix}1+\frac{c}{2},a,c-a,b,c-b,(A)\\ \frac{c}{2},(B)\end{matrix};z\right] &\longrightarrow \frac{(c-a)\,(c-b)}{c\,(c-a-b)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,1-a+c,b,1-b+c,(A)\\ (B)\end{matrix};z\right]\\
&- \frac{a\,b}{c\,(c-a-b)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a+1,c-a,b+1,c-b,(A)\\ (B)\end{matrix};z\right]
\end{aligned}
$$

❑ **C105** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`, `pos5[dw]=1`, `pos6[dw]=2`:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}a,c-a,b,c,(A)\\ 1-b+c,-1+c,(B)\end{matrix};z\right]\\
&\longrightarrow \frac{(c-b-1)\,(c-b)}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-2}(B_i-1)}{\prod_{i=1}^{r-4}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a-1,c-a-1,b,(A-1)\\ c-b-1,(B-1)\end{matrix};z\right]\\
&- \frac{(c-b-1)\,(c-b)}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-2}(B_i-1)}{\prod_{i=1}^{r-4}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,c-a,b-1,(A-1)\\ c-b,(B-1)\end{matrix};z\right]
\end{aligned}
$$

❑ **C112** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`, `pos5[dw]=1`, `pos6[dw]=2`:

$$
{}_rF_s\left[\begin{matrix}1+\frac{c}{2},a,c-a,b,(A)\\ \frac{c}{2},1-b+c,(B)\end{matrix};z\right] \longrightarrow \frac{(c-a)\,(c-b)}{c\,(c-a-b)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,1-a+c,b,(A)\\ c-b,(B)\end{matrix};z\right] - \frac{a\,b}{c\,(c-a-b)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a+1,c-a,b+1,(A)\\ 1-b+c,(B)\end{matrix};z\right]
$$

❑ **C106** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, `pos5[dw]=2`, `pos6[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}a,b,c,(A)\\ 1-a+c,1-b+c,-1+c,(B)\end{matrix};z\right]
$$

$$
\longrightarrow \frac{(c-a-1)\,(c-a)\,(c-b-1)\,(c-b)}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-3}(B_i-1)}{\prod_{i=1}^{r-3}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a-1,b,(A-1)\\ -a+c,c-b-1,(B-1)\end{matrix};z\right]
$$

$$
- \frac{(c-a-1)\,(c-a)\,(c-b-1)\,(c-b)}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-3}(B_i-1)}{\prod_{i=1}^{r-3}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,b-1,(A-1)\\ -1-a+c,c-b,(B-1)\end{matrix};z\right]
$$

❑ **C107** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, `pos5[dw]=2`, `pos6[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}a,c-a,c,(A)\\ b+1,1-b+c,-1+c,(B)\end{matrix};z\right]
$$

$$
\longrightarrow \frac{(b-1)\,b\,(c-b-1)\,(c-b)}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-3}(B_i-1)}{\prod_{i=1}^{r-3}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}-1+a,c-a-1,(A-1)\\ b-1,c-b-1,(B-1)\end{matrix};z\right]
$$

$$
- \frac{(b-1)\,b\,(c-b-1)\,(c-b)}{(a-b)\,(c-1)\,(c-a-b)\,z}\,\frac{\prod_{i=1}^{s-3}(B_i-1)}{\prod_{i=1}^{r-3}(A_i-1)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,c-a,(A-1)\\ b,c-b,(B-1)\end{matrix};z\right]
$$

❑ **C114** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, `pos5[dw]=2`, `pos6[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}1+\frac{c}{2},a,b,(A)\\ \frac{c}{2},1-a+c,1-b+c,(B)\end{matrix};z\right] \longrightarrow \frac{(c-a)\,(c-b)}{c\,(c-a-b)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,b,(A)\\ c-a,-b+c,(B)\end{matrix};z\right] - \frac{a\,b}{c\,(c-a-b)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a+1,b+1,(A)\\ 1-a+c,1-b+c,(B)\end{matrix};z\right]
$$

❑ **C115** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, `pos5[dw]=2`, `pos6[dw]=3`:

$$
{}_rF_s\left[\begin{matrix}1+\frac{c}{2},a,c-a,(A)\\ \frac{c}{2},b,2-b+c,(B)\end{matrix};z\right] \longrightarrow \frac{(c-a)\,(1-b+c)}{c\,(1-a-b+c)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a,1-a+c,(A)\\ b,1-b+c,(B)\end{matrix};z\right] - \frac{a\,(b-1)}{c\,(1-a-b+c)}\,{}_{r-1}F_{s-1}\left[\begin{matrix}a+1,c-a,(A)\\ 1+b,2-b+c,(B)\end{matrix};z\right]
$$

❑ **C108** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`, `pos5[dw]=3`, `pos6[dw]=4`:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix} a, c, (A) \\ 1-a+c, b, 2-b+c, -1+c, (B) \end{matrix}; z\right] \\
\longrightarrow\quad & \frac{(b-2)\,(b-1)\,(c-a-1)\,(c-a)\,(c-b)\,(1-b+c)}{(1+a-b)\,(c-1)\,(1-a-b+c)\,z} \\
& \times \frac{\prod_{i=1}^{s-4}(B_i-1)}{\prod_{i=1}^{r-2}(A_i-1)}\; {}_{r-1}F_{s-1}\left[\begin{matrix} a-1, (A-1) \\ c-a, b-2, c-b, (B-1) \end{matrix}; z\right] \\
- \quad & \frac{(b-2)\,(b-1)\,(c-a-1)\,(c-a)\,(c-b)\,(1-a+b)}{(1+a-b)\,(c-1)\,(1-a-b+c)\,z} \\
& \times \frac{\prod_{i=1}^{s-4}(B_i-1)}{\prod_{i=1}^{r-2}(A_i-1)}\; {}_{r-1}F_{s-1}\left[\begin{matrix} a, (A-1) \\ c-a-1, b-1, 1-b+c, (B-1) \end{matrix}; z\right]
\end{aligned}
$$

❑ **C118** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`, `pos5[dw]=3`, `pos6[dw]=4`:

$$
\begin{aligned}
{}_rF_s\left[\begin{matrix} 1+\frac{c}{2}, a, (A) \\ \frac{c}{2}, 1-a+c, b, 2-b+c, (B) \end{matrix}; z\right] \quad\longrightarrow\quad & \frac{(c-a)\,(1-b+c)}{c\,(1-a-b+c)}\; {}_{r-1}F_{s-1}\left[\begin{matrix} a, (A) \\ c-a, b, 1-b+c, (B) \end{matrix}; z\right] \\
- \quad & \frac{a\,(b-1)}{c\,(1-a-b+c)}\; {}_{r-1}F_{s-1}\left[\begin{matrix} a+1, (A) \\ 1-a+c, b-1, 2-b+c, (B) \end{matrix}; z\right]
\end{aligned}
$$

❑ **C109** with `pos1[up]=1`, `pos2[dw]=1`, `pos3[dw]=2`, `pos4[dw]=3`, `pos5[dw]=4`, `pos6[dw]=5`:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix} c, (A) \\ a, 2-a+c, b, 2-b+c, -1+c, (B) \end{matrix}; z\right] \\
\longrightarrow\quad & \frac{(a-2)\,(a-1)\,(b-2)\,(b-1)\,(c-a)\,(1-a+c)\,(c-b)\,(1-b+c)}{(a-b)\,(c-1)\,(2-a-b+c)\,z} \\
& \times \frac{\prod_{i=1}^{s-5}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\; {}_{r-1}F_{s-1}\left[\begin{matrix} (A-1) \\ a-1, 1-a+c, b-2, c-b, (B-1) \end{matrix}; z\right] \\
- \quad & \frac{(a-2)\,(a-1)\,(b-2)\,(b-1)\,(c-a)\,(1-a+c)\,(c-b)\,(1-b+c)}{(a-b)\,(c-1)\,(2-a-b+c)\,z} \\
& \times \frac{\prod_{i=1}^{s-5}(B_i-1)}{\prod_{i=1}^{r-1}(A_i-1)}\; {}_{r-1}F_{s-1}\left[\begin{matrix} (A-1) \\ a-2, c-a, b-1, 1-b+c, (B-1) \end{matrix}; z\right]
\end{aligned}
$$

❑ **C120** with pos1[up]=1, pos2[dw]=1, pos3[dw]=2, pos4[dw]=3, pos5[dw]=4, pos6[dw]=5:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}1+\frac{c}{2},(A)\\ \frac{c}{2},a,2-a+c,b,2-b+c,(B)\end{matrix};z\right]\\
\longrightarrow\ &\frac{(1-a+c)\,(1-b+c)}{c\,(2-a-b+c)}\;{}_{r-1}F_{s-1}\left[\begin{matrix}(A)\\ a,1-a+c,b,1-b+c,(B)\end{matrix};z\right]\\
-\ &\frac{(a-1)\,(b-1)}{c\,(2-a-b+c)}\;{}_{r-1}F_{s-1}\left[\begin{matrix}(A)\\ a-1,2-a+c,b-1,2-b+c,(B)\end{matrix};z\right]
\end{aligned}
$$

7.7 Three-term relations with eight parameters

❑ **C111** with pos1[up]=1, pos2[up]=2, pos3[up]=3, pos4[up]=4, pos5[up]=5, pos6[up]=6, pos7[dw]=1, pos8[dw]=2:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}a,c-a,b,c-b,d+1,1+c-d,(A)\\ d,c-d,(B)\end{matrix};z\right]\\
\longrightarrow\ &\frac{a\,(c-a)\,(b-d)\,(c-b-d)}{(b-a)\,(c-a-b)\,(c-d)\,d}\;{}_{r-2}F_{s-2}\left[\begin{matrix}a+1,1-a+c,b,-b+c,(A)\\ (B)\end{matrix};z\right]\\
-\ &\frac{b\,(c-b)\,(a-d)\,(c-a-d)}{(b-a)\,(c-a-b)\,(c-d)\,d}\;{}_{r-2}F_{s-2}\left[\begin{matrix}a,c-a,b+1,1-b+c,(A)\\ (B)\end{matrix};z\right]
\end{aligned}
$$

❑ **C113** with pos1[up]=1, pos2[up]=2, pos3[up]=3, pos4[up]=4, pos5[up]=5, pos6[dw]=1, pos7[dw]=2, pos8[dw]=3:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}a,c-a,b,d+1,1+c-d,(A)\\ 1-b+c,d,c-d,(B)\end{matrix};z\right]\\
\longrightarrow\ &\frac{a\,(c-a)\,(b-d)\,(c-b-d)}{(b-a)\,(c-a-b)\,(c-d)\,d}\;{}_{r-2}F_{s-2}\left[\begin{matrix}a+1,1-a+c,b,(A)\\ 1-b+c,(B)\end{matrix};z\right]\\
-\ &\frac{b\,(c-b)\,(a-d)\,(c-a-d)}{(b-a)\,(c-a-b)\,(c-d)\,d}\;{}_{r-2}F_{s-2}\left[\begin{matrix}a,c-a,b+1,(A)\\ c-b,(B)\end{matrix};z\right]
\end{aligned}
$$

❑ **C116** with pos1[up]=1, pos2[up]=2, pos3[up]=3, pos4[up]=4, pos5[dw]=1, pos6[dw]=2, pos7[dw]=3, pos8[dw]=4:

$$
\begin{aligned}
&{}_rF_s\left[\begin{matrix}a,b,d+1,1+c-d,(A)\\ 1-a+c,1-b+c,d,c-d,(B)\end{matrix};z\right]\\
\longrightarrow\ &\frac{a\,(c-a)\,(b-d)\,(c-b-d)}{(b-a)\,(c-a-b)\,(c-d)\,d}\;{}_{r-2}F_{s-2}\left[\begin{matrix}a+1,b,(A)\\ c-a,1-b+c,(B)\end{matrix};z\right]\\
-\ &\frac{b\,(c-b)\,(a-d)\,(c-a-d)}{(b-a)\,(c-a-b)\,(c-d)\,d}\;{}_{r-2}F_{s-2}\left[\begin{matrix}a,b+1,(A)\\ 1-a+c,c-b,(B)\end{matrix};z\right]
\end{aligned}
$$

❑ **C117** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[up]=4`, `pos5[dw]=1`, `pos6[dw]=2`, `pos7[dw]=3`, `pos8[dw]=4`:

$$
\begin{aligned}
& {}_rF_s\left[\begin{matrix} a, c-a, d+1, 1+c-d, (A) \\ b, 2-b+c, d, c-d, (B) \end{matrix}; z\right] \\
\longrightarrow\; & \frac{a\,(c-a)\,(b-d-1)\,(1-b+c-d)}{(b-a-1)\,(1-a-b+c)\,(c-d)\,d}\, {}_{r-2}F_{s-2}\left[\begin{matrix} a+1, 1-a+c, (A) \\ b, 2-b+c, (B) \end{matrix}; z\right] \\
- \; & \frac{(b-1)\,(1-b+c)\,(a-d)\,(c-a-d)}{(b-a-1)\,(1-a-b+c)\,(c-d)\,d}\, {}_{r-2}F_{s-2}\left[\begin{matrix} a, c-a, (A) \\ -1+b, 1-b+c, (B) \end{matrix}; z\right]
\end{aligned}
$$

❑ **C119** with `pos1[up]=1`, `pos2[up]=2`, `pos3[up]=3`, `pos4[dw]=1`, `pos5[dw]=2`, `pos6[dw]=3`, `pos7[dw]=4`, `pos8[dw]=5`:

$$
\begin{aligned}
& {}_rF_s\left[\begin{matrix} a, d+1, 1+c-d, (A) \\ 1-a+c, b, 2-b+c, d, c-d, (B) \end{matrix}; z\right] \\
\longrightarrow\; & \frac{a\,(c-a)\,(b-d-1)\,(1-b+c-d)}{(b-a-1)\,(1-a-b+c)\,(c-d)\,d}\, {}_{r-2}F_{s-2}\left[\begin{matrix} a+1, (A) \\ c-a, b, 2-b+c, (B) \end{matrix}; z\right] \\
- \; & \frac{(b-1)\,(1-b+c)\,(a-d)\,(c-a-d)}{(b-a-1)\,(1-a-b+c)\,(c-d)\,d}\, {}_{r-2}F_{s-2}\left[\begin{matrix} a, (A) \\ 1-a+c, -1+b, 1-b+c, (B) \end{matrix}; z\right]
\end{aligned}
$$

❑ **C121** with `pos1[up]=1`, `pos2[up]=2`, `pos3[dw]=1`, `pos4[dw]=2`, `pos5[dw]=3`, `pos6[dw]=4`, `pos7[dw]=5`, `pos8[dw]=6`:

$$
\begin{aligned}
& {}_rF_s\left[\begin{matrix} d+1, 1+c-d, (A) \\ a, 2-a+c, b, 2-b+c, d, c-d, (B) \end{matrix}; z\right] \\
\rightarrow\; & \frac{(a-1)\,(1-a+c)\,(b-d-1)\,(1-b+c-d)}{(b-a)\,(2-a-b+c)\,(c-d)\,d}\, {}_{r-2}F_{s-2}\left[\begin{matrix} (A) \\ a-1, 1-a+c, b, 2-b+c, (B) \end{matrix}; z\right] \\
- \; & \frac{(b-1)\,(1-b+c)\,(a-d-1)\,(1-a-c+d)}{(b-a)\,(2-a-b+c)\,(c-d)\,d}\, {}_{r-2}F_{s-2}\left[\begin{matrix} (A) \\ a, 2-a+c, -1+b, 1-b+c, (B) \end{matrix}; z\right]
\end{aligned}
$$

For example, see the proof of theorem 4 in [18], in which a contigous relation is applied several times, and combined with summation formulas, to prove a summation. See also [22] for a proof of a transformation formula using contiguous relation.

Chapter 8

Formal limits of hypergeometric expressions

Lim

The function Lim allows to do *formal limits of hypergeometric expressions*. In particular, the confluence principle is implemented:

$$\lim_{b\to+\infty} {}_2F_1\left[\begin{matrix}a,b\\c\end{matrix};z/b\right] = {}_1F_1\left[\begin{matrix}a\\c\end{matrix};z\right]$$

Proof:

$${}_2F_1\left[\begin{matrix}a,b\\c\end{matrix};z/b\right] = \sum_k \frac{(a)_k\,(b)_k}{(c)_k\,(1)_k}\,\frac{z^k}{b^k} = \sum_k \frac{(a)_k}{(c)_k\,(1)_k}\,z^k\,\frac{(b)_k}{b^k}$$

Let b tends to $+\infty$. So, the last expression tends to

$$\sum_k \frac{(a)_k}{(c)_k\,(1)_k}\,z^k = {}_1F_1\left[\begin{matrix}a\\c\end{matrix};z\right] .$$

```
> eg := HYP[[a,b,-n],[c,1+a+b-c-n],1] = RF[-a+c,n]*RF[-b+c,n]/RF[c,n]/RF[-a-b+c,n];

                                                 RF[c - a, n] RF[- b + c, n]
  eg := HYP[[a, b, - n], [c, 1 + a + b - c - n], 1] = ---------------------------
                                                 RF[c, n] RF[- a - b + c, n]

> Lim( eg, b=infinity );

                                 GAMMA(- a + c + n) GAMMA(c)
              HYP[[a, - n], [c], 1] = ---------------------------
                                 GAMMA(- a + c) GAMMA(c + n)

> #confluent principle
> Lim( HYP[[a,2*b],[c],z/b], b=infinity );

                           HYP[[a], [c], 2 z]
```

```
> #obviously
> expr := RF[-n,r]/RF[1+a+b-c-n,r];

                                      RF[- n, r]
                       expr := ------------------------
                                 RF[1 + a + b - c - n, r]

> Lim( expr, n=infinity );

                                      1

> # Slater, Appendix (III.24)
> egbis := HypSumPrint( HYP[[a,1-a,c],[d,1+2*c-d],1], 34 );

          egbis := HYP[[a, 1 - a, c], [d, 1 + 2 c - d], 1] =

                (1 - 2 c)                              /
            Pi 2          GAMMA(d) GAMMA(1 + 2 c - d)  /
                                                      /

            (GAMMA(1/2 a + 1/2 + c - 1/2 d) GAMMA(1/2 a + 1/2 d)

             GAMMA(1 - 1/2 a + c - 1/2 d) GAMMA(1/2 - 1/2 a + 1/2 d))

> Lim( egbis, c=infinity );

                                          (1 - d)   1/2
                                         2        Pi    GAMMA(d)
  HYP[[a, 1 - a], [d], 1/2] = -----------------------------------------------
                              GAMMA(1/2 a + 1/2 d) GAMMA(1/2 - 1/2 a + 1/2 d)

> Lim( HYP[[a1,a2],[b1,b2,lbd*v],lbd*z], lbd=infinity );

                         HYP[[a1, a2], [b1, b2], z/v]

> Lim( GAMMA(a+m)/GAMMA(b+m), m=infinity );

                                            (a - b)
                             limit         m
                             m -> infinity

> exprbis := RF[2*a,k] * RF[a/2+k/4,n]
> * RF[a/2+k/4+1/2,n] * RF[a/2+k/4+1/4,n] * RF[a/2+k/4+3/4,n]
> / ( RF[a/2,n] * RF[a/2+1/2,n] * RF[a/2+1/4,n] * RF[a/2+3/4,n] ):
> Lim( exprbis, n=infinity );

                                             k   k
                              limit         4   n
                              n -> infinity
```

Chapter 9

Polynomial, rational and hypergeometric solutions of recurrences

From the algorithmic point of view, polynomial solutions can be found by indeterminate coefficients, rational solutions can be found by Abramov's algorithm and hypergeometric solutions can be computed thans to Petkovsek's algorithm.

`GenRec, PolySolRec, RatioSolRec, HypSolRec`

These functions are used to find all polynomial, rational and hypergeometric solutions of linear recurrence equations (see definition 8) with *polynomial coefficients*:

$$a_k(n)\,U(n+k) + \cdots + a_1(n)\,U(n+1) + a_0(n)\,U(n) = f(n) \qquad \text{with } a_i(n), f(n) \in \mathbb{K}[n] \tag{9.1}$$

If there is no solution, the return value is "`no solution`".

9.1 Polynomial solutions of recurrences

To find polynomial solutions of a recurrence (9.1), it is natural to proceed in two steps:

- find an upper bound d for the possible degrees of polynomial solutions of the recurrence [3] ;
- given d, compute all polynomial solutions having degree at most d (using the method of undetermined coefficients).

Example 5

$$3\,U(n+2) - n\,U(n+1) + (n-1)\,U(n) = 0 \;, \tag{9.2}$$

The process to determine a degree bound d for polynomial solutions of this equation, returns $d = 2$. So, once $a\,n^2 + b\,n + c$ has been substitued for $U(n)$ into equation (9.2), it is easy to explicite a, b and c:

$$3\,(a\,(n+2)^2 + b\,(n+2) + c) \;-\; n\,(a\,(n+1)^2 + b\,(n+1) + c) \;+\; (n-1)\,(a\,n^2 + b\,n + c) = 0 \;.$$

After expanding the last expression, then solving this resulting linear equation:

$$\begin{aligned} & (11\,a+b)\,n+12\,a+6\,b+2\,c=0\ , \\ \Longrightarrow\quad & b=-11\,a\ ,\qquad c=27\,a\ , \end{aligned}$$

we get the general solution $U(n)=a\,n^2-11\,a\,n+27\,a=a\,(n^2-11\,n+27)$.
Finally, polynomial solutions of (9.2) are ***up to a constant factor****:*

$$U(n)=n^2-11\,n+27\ .$$

```
> PolySolRec( 3*U(n+2)-n*U(n+1)+(n-1)*U(n), U(n) );

                                           2
                        _x2 (27 - 11 n + n )

> `HYPERG/Verbose` := 1:
> PolySolRec( 3*U(n+2)-n*U(n+1)+(n-1)*U(n)=3, U(n) );

 Degree bound of polynomial solutions: 2

                                                   2
                     3/2 + 27 _x2 - 11 _x2 n + _x2 n

> PolySolRec( n*U(n+1)-(n+8)*U(n), U(n) );

 Degree bound of polynomial solutions: 8

        _x8 n (n + 7) (n + 6) (n + 5) (n + 4) (n + 3) (n + 2) (n + 1)
```

The output is a polynomial in n with constant coefficients `_x0`, `_x1`, `_x2`, ...

It is possible to precise initial conditions to have an explicit solution, but if insufficient boundary conditions are given, `PolySolRec` uses symbolic names (see above) as default values.

```
> PolySolRec( 3*U(n+2)-n*U(n+1)+(n-1)*U(n), U(n), {U(0)=27} );

 Degree bound of polynomial solutions: 2

                                         2
                            27 - 11 n + n

> PolySolRec( r(n+2)-2*r(n+1)+r(n)=2, r(n), {r(0)=-1,r(1)=m} );

 Degree bound of polynomial solutions: 5

                                         2
                            - 1 + m n + n
```

This first algorithm is used by the two following functions `RatioSolRec` and `HypSolRec` to compute rational and hypergeometric solutions.

9.2 Rational solutions of recurrences

This section presents an algorithm due to *Abramov* that computes rational solutions $P(n)/Q(n)$ of an equation of type (9.1). The idea of is to first construct a polynomial divisible by the denominator of any rational solution of a linear recurrence equation.

Let $U(n) = \dfrac{P(n)}{Q(n)}$ be a rational solution of the recurrence (9.1):

- The *Abramov* algorithm determines the denominator $Q(n)$, see [1].
- By substituting $U(n)$ in (9.1), we get,

$$a_k(n)\,\frac{P(n+k)}{Q(n+k)} + \cdots + a_1(n)\,\frac{P(n+1)}{Q(n+1)} + a_0(n)\,\frac{P(n)}{Q(n)} = f(n)\ .$$

  Then, we have to search a polynomial solution $P(n)$ of the following recurrence equation:

$$a_k(n)P(n+k)\prod_{\substack{i=0\\ i\neq k}}^{k} Q(n+i)+\cdots+a_1(n)P(n+1)\prod_{\substack{i=0\\ i\neq 1}}^{k} Q(n+i)+a_0(n)P(n)\prod_{\substack{i=0\\ i\neq 0}}^{k} Q(n+i) = f(n)\prod_{i=0}^{k} Q(n+i)\ .$$

  This can be done easily by the algorithm presented in section **9.1**.

```
> RatioSolRec( 2*n*(n+3)*U(n)-(n+1)*(3*n+8)*U(n+1)+(n+2)^2*U(n+2), U(n) );

                                      _x0
                                      ---
                                       n

> RatioSolRec( (n+4)*(2*n+1)*(n+2)*U(n+3)-(2*n+3)*(n+3)*(n+1)*U(n+2)
>          +n*(n+2)*(2*n-3)*U(n+1)-(n-1)*(2*n-1)*(n+1)*U(n)=0, U(n) );

                                  _x1 (2 n - 3)
                              1/2 -------------
                                        2
                                       n  - 1

> RatioSolRec( (n+4)*U(n+2)+U(n+1)-(n+1)*U(n), U(n), {U(0)=2} );

                                       4
                                ---------------
                                (n + 2) (n + 1)

> `HYPERG/Verbose` := 1:
> RatioSolRec( (n+4)*U(n+2)+U(n+1)-(n+1)*U(n), U(n), {U(1)=1} );

 Denominator of rational solutions: (n+2)*(n+1)

 Degree bound of polynomial solutions: 1

                                       6
                                ---------------
                                (n + 2) (n + 1)
```

9.3 Hypergeometric solutions of recurrences

The following process applies only to homogeneous recurrences ($f(n)=0$):

$$a_k(n)\,U(n+k)+\cdots+a_1(n)\,U(n+1)+a_0(n)\,U(n)=0 \qquad \text{with } a_i(n)\in\mathbb{K}[n] \tag{9.3}$$

Note that non-homogeneous equations can also be considered after homogenization (by increasing the order by one).

Assume that $S(n)$ is a hypergeometric solution of (9.3). Then there is a rational sequence $R(n)$ such that $S(n+1)=R(n)\,S(n)$, ..., $S(n+i)=R(n+i-1)\,\cdots\,R(n)\,S(n)$. Substituting this into (9.3), and cancelling $S(n)$, we get a non-linear equation in R: (*Ricatti equation*)

$$a_k(n)\,R(n+k-1)\,\cdots\,R(n)+\cdots+a_1(n)\,R(n+1)\,R(n)+a_0(n)=0\ . \tag{9.4}$$

Let

$$R(n)=\frac{S(n+1)}{S(n)}=z\,\frac{A(n)}{B(n)}\,\frac{C(n+1)}{C(n)} \tag{9.5}$$

with

$$\begin{aligned} \gcd(A(n),B(n+k)) &= 1 \qquad \text{for all } k\in\mathbb{N}\ ,\\ \gcd(A(n),C(n)) &= 1\ ,\\ \gcd(B(n),C(n+1)) &= 1\ ,\end{aligned}$$

A, B, C, monic polynomials,

be the normal form (called Gosper-Petkovšek form, [23]) for $R(n)$.

Then,

$$\begin{aligned} & z^k\,a_k(n)\,A(n+k-1)\,A(n+k-2)\cdots A(n)\,C(n+k)\\ & +z^{(k-1)}\,B(n+k-1)\,A(n+k-2)\cdots A(n)\,C(n+k-1)\\ & +\cdots+a_0(n)\,B(n+k-1)\cdots B(n)\,C(n)=0\ .\end{aligned}$$

From this last recurrence, the following algorithm (proposed in [23]) finds all solutions $R(n)$ of (9.4), and also all hypergeometric solutions of (9.3).

Algorithm 2 (HYPER, Petkovšek)
For all monic factors $A(n)$ of $a_0(n)$ and $B(n)$ of $a_k(n-k+1)$, do

- let $M_i(n)=a_i(n)\displaystyle\prod_{j=0}^{i-1}A(n+j)\prod_{j=i}^{k-1}B(n+j)$ for $i=0,1,\cdots,k$;
- $p=max\{\deg M_i(n)$ for $i=0,1,\cdots,k\}$;
- let α_i be the coefficient of n^p in $M_i(n)$ for $i=0,1,\cdots,k$;
- for all non-zero $z\in\mathbb{K}$ such that $\displaystyle\sum_{i=0}^{k}\alpha_i\,z^i=0$,

  if the recurrence

  $$\sum_{i=0}^{k}z^i\,M_i(n)\,C(n+i)=0$$

has a non-zero polynomial solution $C(n)$ then,

$$R(n) = z\,\frac{A(n)}{B(n)}\,\frac{C(n+1)}{C(n)}$$

return a non-zero solution $S(n)$ of $S(n+1) = R(n)\,S(n)$.

□

This algorithm returns a generating set (not necessarily linearly independent) of hypergeometric solutions of (9.3).

Note that HypSolRec returns a set of quotients $U(n+1)/U(n)$ rather than solutions $U(n)$ themselves. For example, if the set is $\{\alpha_n, \beta_n\}$ (with $\alpha_n = \frac{a_{n+1}}{a_n}$ and $\beta_n = \frac{b_{n+1}}{b_n}$), then the general solution is a linear combination of a_n and b_n.

```
> HypSolRec( (n-1)*y(n+2)-(n^2+3*n-2)*y(n+1)+2*n*(n+1)*y(n)=0, y(n) );

                              {2, n + 1}

> HypSolRec( a(n+2)-199*a(n+1)+9900*a(n), a(n) );

                              {99, 100}

> # number of involutions of a set with n elements
> HypSolRec( a(n)=a(n-1)+(n-1)*a(n-2), a(n) );

                                  {}

> HypSolRec(a(n+2) - (2*n + 1)*a(n+1) + (n^2-u)*a(n) , a(n));

                               1/2        1/2
                         {n + u   , n - u   }

> `HYPERG/Verbose` := 1:
> HypSolRec( -4*(2*m+3)*(2*m+1)*U(m)+(m+3)*(m+2)*U(m+1), U(m) );

 Searching hypergeometric solutions with 1 and 1

 Degree bound of polynomial solutions: 0
 ....
 Searching hypergeometric solutions with m+3/2 and m+3

 Degree bound of polynomial solutions: 0

 Searching hypergeometric solutions with m+3/2 and (m+3)*(m+2)

                          (m + 3/2) (m + 1/2)
                      {16 -------------------}
                            (m + 3) (m + 2)
```

This last exemple shows that there isn't any non-zero hypergeometric solution.

9.4 Generate p-order recurrences

Suppose you want a recurrence whose solution space contains $n!$ and 2^n. The algorithm to obtain a such recurrence is:

- compute the "Casoratian" determinant of $y(n)$, $n!$ and 2^n, and set it equal to 0 ;

$$\begin{vmatrix} y(n) & n! & 2^n \\ y(n+1) & (n+1)! & 2^{n+1} \\ y(n+2) & (n+2)! & 2^{n+2} \end{vmatrix} = 0 \ ,$$

- and divide it by the product $n! \times 2^n$;
- this gives a 2^{nd} order linear recurrence with polynomial coefficients satisfied by $n!$ and 2^n:

$$(1-n)\,y(n+2) + (n^2+3\,n-2)\,y(n+1) - 2\,n\,(n+1)\,y(n) = 0$$

This means that $n!$ and 2^n are both solutions of the last recurrence.

So, it is easy to generalize this algorithm to make a p-order homogeneous recurrence such that s_1, s_2, ..., s_p are its solutions.

```
> rec := GenRec( {n!,2^n}, y(n) );

                                   2
  rec := 2 n (n + 1) y(n) + (- n  - 3 n + 2) y(n + 1) + (n - 1) y(n + 2) = 0

> HypSolRec( rec, y(n) );

                                   {2, n + 1}

> recbis := GenRec( {k,k^2,k^3}, U(k) );

  recbis := - 2 (k + 3) (k + 2) (k + 1) U(k) + 6 k (k + 3) (k + 2) U(k + 1)

       - 6 k (k + 3) (k + 1) U(k + 2) + 2 k (k + 2) (k + 1) U(k + 3) = 0

> PolySolRec( recbis, U(k) );

                                              2
                           k (_x1 + _x2 k + _x3 k )
```

Chapter 10

Gosper and Zeilberger algorithms

Gosper, HypToRec, Prove, SummandToRec, SumToRec

10.1 Gosper's algorithm

The Gosper algorithm [9] deals with the question of finding an *upward* antidifference $T(k)$ for given $U(k)$, i.e. a sequence $T(k)$ for which

$$\boldsymbol{T(k+1) - T(k) = U(k)} \ ,$$

in the particular case when $U(k)$ is a hypergeometric term. In that case, $\frac{T(k)}{T(k-1)}$ is a rational function with respect to k.

```
> T := Gosper( k^2, k );

                                          2        3
                      T := _x0 + 1/6 k - 1/2 k  + 1/3 k

> U := subs(k=k+1,T) - T;

                             2               3        2        3
           U := 1/6 - 1/2 (k + 1)  + 1/3 (k + 1)  + 1/2 k  - 1/3 k

> simplify( U );

                                        2
                                       k

> Gosper( k*k!, k );

                                       k!

> Gosper( (k/2)!, k );
Error, (in Gosper) (1/2*k)!, is not hypergeometric in, k
```

```
> T1 := Gosper( binomial(n,k), n );

                                   (- n + k) n!
                        T1 := - -------------------
                                (k + 1) k! (n - k)!

> convert( T1, binomial );

                           (n - k) binomial(n, k)
                           ----------------------
                                   k + 1

> T2 := Gosper( (-1)^(k+1)*(4*k+1)*(2*k)!/(k!*4^k*(2*k-1)*(k+1)!), k );

                                        (k + 1)
                                (2 k)! (-1)        (k + 1)
                    T2 := - 2 --------------------------
                                                    k
                                 (k + 1)! (2 k - 1) 4  k!

> T3 := Gosper( (-1)^k*binomial(n,k), n );
                                             k
                                 (- n + k) (-1)  n!
                        T3 := - -------------------
                                (k + 1) k! (n - k)!

> convert( T3, binomial );

                                  k
                        (n - k) (-1)  binomial(n, k)
                        ----------------------------
                                   k + 1

> convert( Gosper( binomial(1/2,m-j+1)*binomial(1/2,m+j), j ), binomial );

                   2          2
  (j - 1) (m + 2 m  + j - 2 j ) binomial(1/2, m - j + 1) binomial(1/2, m + j)
  ---------------------------------------------------------------------------
                                 m (1 + 2 m)

> Gosper(1/k,k);

                                   FAIL
```

Remark: Gosper's algorithm is in the computational heart of Zeilberger's fast algorithm described in the next section.

10.2 Zeilberger's algorithm

10.2.1 Description

The (fast) Zeilberger algorithm [31] deals with the definite summation of hypergeometric terms. It is a direct application of the Gosper algorithm. It finds a linear homogeneous recurrence relation for the sum

$$s(n) := \sum_k F(n,k) \ ,$$

where $F(n,k)$ is a hypergeometric term in n and k.

The first step is finding a recurrence relation for the summand $F(n,k)$. It is an extension of the Gosper algorithm which finds polynomials $a_j(n)$ and $G(n,k)$ such that:

$$\sum_{j=0}^{J} \boldsymbol{a_j(n)\, F(n+j,k) = G(n,k+1) - G(n,k)} \;, \tag{10.1}$$

with $G(n,k) = R(n,k)\,F(n,k)$ where $R(n,k)$ is a rational function called "certificate".

Summing (10.1) over all integers k, we get

$$\sum_{j=0}^{J} a_j(n)\, s(n+j) = G(n,\infty) - G(n,-\infty) \;, \tag{10.2}$$

and checking that $G(n,\pm\infty) = 0$, the recurrence over the initial sum $s(n)$ is proved:

$$\boldsymbol{a_0(n)\, s(n) + a_1(n)\, s(n+1) + \cdots + a_J(n)\, s(n+J) = 0} \;. \tag{10.3}$$

The existence of a such recurrence equation for $s(n)$ is assured, under the hypothesis that $F(n,k)$ is a proper hypergeometric term, i.e.

$$F(n,k) = P(n,k)\, \frac{\prod_{s=1}^{pp} (a_s\, n + b_s\, k + c_s)!}{\prod_{t=1}^{qq} (u_t\, n + v_t\, k + w_t)!}\, w^n\, z^k \;,$$

where P is a polynomial, w and z are complex numbers, a_s, b_s, u_t, v_t are integer parameters, and c_i, w_i are complex parameters. (see the fundamental theorem 4.4.1 in [24].)

```
> SummandToRec( binomial(n,k), k, n, F, G, 'cert' );

              - 2 F(n, k) + F(n + 1, k) = G(n, k + 1) - G(n, k)

> cert;

                                     k
                                -----------
                                - n - 1 + k

> SumToRec( Sum(binomial(n,k),k), n, A );

                           2 A(n) - A(1 + n) = 0

> SumToRec( Sum(binomial(n,k)^2,k), n, A );

                  (- 4 n - 2) A(n) + (1 + n) A(1 + n) = 0

> SumToRec( Sum(binomial(n,k)^3,k), n, A );

             2               2                              2
  - 8 (n + 1)  A(n) + (- 7 n  - 21 n - 16) A(n + 1) + (n + 2)  A(n + 2) = 0

> # Motzkin numbers
> SumToRec( Sum((-1)^(k+1)*binomial(1/2,k)*binomial(k,n+2-k)
                       *2^(2*k-n-3)*3^(n+2-k),k), n, A );

        (3 + 3 n) A(n) + (2 n + 5) A(1 + n) + (- 4 - n) A(2 + n) = 0
```

```
> `HYPERG/Verbose` := 1:
> HypToRec( HYP[[a,b,-n],[d,e],z], a, A, 'certificat' );

 Searching recurrence of order 1

 Searching recurrence of order 2

 Searching recurrence of order 3
                                                2       2
(- e + 1 + a) (a + 1 - d) A(a) + (z - 4 - 3 a  + z a  - 7 a - z b n - z b + z n

     - d e + 2 d - a z b + a z n + 2 z a + 2 e + 2 d a + 2 e a) A(a + 1)

     - (a + 1) (e + 3 z - 5 + 2 z a - 3 a + d - z b + z n) A(a + 2)

     + (a + 2) (a + 1) (z - 1) A(a + 3) = 0

> certificat;

                           (d + k - 1) (e + k - 1) k
                           -------------------------
                                       a
```

10.2.2 Automatic proof

The Prove function allows to obtain automatic proof:

```
> Prove( (-1)^k*binomial(n,k),k,n, 'filename=Proof' );

                     Creating automatic proof in file Proof

> !more Proof
AUTOMATIC PROOF

Let F(n,k) be given by:
                                  k
                              (-1)  binomial(n, k)

and let SUM(n) be the sum of F(n,k) with respect to k.

Then SUM(n) satisfies the following linear recurrence equation: (I)

                      SUM(n) n + (- n - 2) SUM(n + 1) = 0

PROOF: We cleverly construct G(n,k):

                                       k
                         (k + 1) k (-1)  binomial(n, k)
                         -----------------------------
                                   n + 1 - k

with the motive that

              F(n) n + (- n - 2) F(n + 1) = G(n, k + 1) - G(n, k)
```

```
(check!)

and the identity (I) follows upon summing with respect to k.

> Prove(binomial(x,k)*binomial(y,n-k),k,n,'filename=Myproof','mode=latex');

                    Creating automatic proof in file Myproof
```

Part III

Routines

This part describes the main routines available in the **HYPERG** package. Each function description has the following format: after a one-line description showing how the function can be called, we find informations about the expected arguments. Then, we give a brief exposition about the function, and finally we provide some examples of the use of the function. Some related functions or topics finish the description.

All of the descriptions that are printed in this part can also be produced on-line in *Maple* session by the commande `help(HYPERG[function])` or `?HYPERG,function`.

The first chapter is devoted to present the `HYPERG` package. Then, all main routines are divided into nine groups according their feature:

- the basic objects (hypergeometric term, hypergeometric series, recurrences) ;
- the rules for manipulating sums ;
- the rules for manipulating factorial expressions ;
- the summation formulas ;
- the transformation formulas ;
- the contiguous relations ;
- the objects for computing limits of hypergeometric expressions ;
- the objects for finding polynomial, rational and hypergeometric solutions of recurrences ;
- the objects for applying the Gosper and Zeilberger algorithms.

Note that all routines are still available after having loaded the `HYPERG` package.

Chapter 11

Presentation of the package HYPERG

```
HELP FOR: A package for hypergeometric functions.

CALLING SEQUENCE:
  HYPERG[<function>](args)
  <function>(args)

SYNOPSIS:
- This package provides functions to manipulate hypergeometric functions.

- Whenever there is a conflict between  a  function  name  in  HYPERG  and
  another   name   used  in  the  same  session,   use  the  long  form
  HYPERG[<function>].

- The available objects are:

     HYP               VHYP              RF

- To get more information on  Hypergeometric Series and Rising Factorials,
  type ?HYPERG,HYP (or ?HYPERG[HYP]),  ?HYPERG,VHYP (or ?HYPERG[VHYP]) and
  ?HYPERG,RF (or ?HYPERG[RF]).

- The available functions are:

     AddParam          CheckRec          FirstTerms        Homog
     HypConverg        HypDiff           HypEval           HypOrder
     HypPermBoth       HypPermLow        HypPermUp         HypSimplify
     HypToVHyp         HypType           HypergToRec       IsHYP
     IsHomog           IsHyperg          IsRF              IsVHYP
     Lim               MapApply          MapList           RecOrder
     RfEval            ShiftRec          SimplifyRec       SubsRec
     SumToHyp          Time              VHypToHyp

- The rules for manipulating factorial expressions are:

     BaseSplit         Ext1              Ext2              Inv
     Linear1           Linear2           Neg1              Neg2
     Split             Trans
```

```
- The rules for manipulating hypergeometric series are:

     HypContig         HypContigPrint    HypSum            HypSumList
     HypSumPrint       HypTransf         HypTransfList     HypTransfPrint

- The functions for applying the Gosper and Zeilberger algorithms:

     Gosper            HypToRec          Prove             SumToRec
     SummandToRec

- The functions for finding solutions of recurrences:

     GenRec            HypSolRec         PolySolRec        RatioSolRec

- The available q-objects are:

     QHYP              WHYP              QRF               QBIN

- The functions for manipulating q-series:

     IsQBIN            IsQHYP            IsQRF             IsWHYP
     QBinEval          QHypEval          QHypOrder         QHypToWHyp
     QHypType          QRfEval           WHypToQHyp

- The functions for finding solutions of q-recurrences:

     HypSolQRec        PolySolQRec       RatioSolQRec

- For help with a particular function, type either  ?HYPERG[<function>] or
  ?HYPERG,<function>   where <function>   is  one  from  the  above  list.

EXAMPLES:
> with(HYPERG):
> mysum := Sum( binomial(M,k)*binomial(N,R-k), k ):
> expr := SumToHyp( mysum );

             expr := binomial(N, R) HYP[[- R, - M], [N - R + 1], 1]

> expr := HypSum( expr, 1 );   # Vandermonde's theorem
                               - R, must be <= 0.

                              binomial(N, R) RF[N + 1, M]
                      expr := ---------------------------
                                    RF[N - R + 1, M]

> expr := RfEval( expr );

                   binomial(N, R) GAMMA(N + 1 + M) GAMMA(N - R + 1)
            expr := ------------------------------------------------
                          GAMMA(N + 1) GAMMA(N - R + 1 + M)

> simplify( expr );
```

```
          GAMMA(N + 1 + M)
---------------------------------
GAMMA(R + 1) GAMMA(N - R + 1 + M)
```

Chapter 12

Basic objects

12.1 Hypergeometric term

```
FUNCTION: IsHyperg - test if is a hypergeometric term

CALLING SEQUENCE:
  IsHyperg(s,n)
  HYPERG[IsHyperg](s,n)

PARAMETERS:
  s - a term (considered as a sequence)
  n - the index of the sequence

SYNOPSIS:
- The IsHyperg function tests whether a sequence is  hypergeometric in  n.

- Whenever there  is  a  conflict  between  the function name IsHyperg and
  another name used in the same session, use the form  HYPERG['IsHyperg'].

EXAMPLES:
> with(HYPERG):
> IsHyperg( 2^p*p!, p );

                                     true

> IsHyperg( sin(n), n );
                                    false

SEE ALSO: HYPERG,   HYP,  IsHYP,  VHYP,  IsVHYP,   RF,  IsRF,  HypergToRec
```

```
FUNCTION: HypergToRec - return a linear first-order homogeneous recurrence
                      - satisfied by a hypergeometric term

CALLING SEQUENCE:
  HypergToRec(term, u(n))
  HYPERG[HypergToRec](term, u(n))
```

```
PARAMETERS:
  term - a hypergeometric term
  u,n  - the name and the index of a sequence

SYNOPSIS:
- By definition of hypergeometric term, it is trivial to find a recurrence
  relation satisfied by a hypergeometric term:
          U(n+1)/U(n) = P(n)/Q(n) where P and Q are polynomials in n,
  so,     P(n) * U(n) - Q(n) * U(n+1) = 0.

- The  hypergeometric  term  must  be  a  product  of  rational  functions
  factorials,    rising  factorials,    binomial  coefficients    and
  power functions.

- The output is a  linear  first-order  homogeneous  recurrence  satisfied
  by the hypergeometric term.

- Whenever there is a conflict between the function name  HypergToRec  and
  another   name   used   in   the   same   session, use  the  long  form
  HYPERG['HypergToRec'].

EXAMPLES:
> with(HYPERG):
> HypergToRec( binomial(3*k,k), T(k) );

         3 (3 k + 1) (3 k + 2) T(k) - 2 (2 k + 1) (k + 1) T(k + 1) = 0

> HypergToRec( (-1)^n * RF[n,3] * 2^n / (n+b), U(n) );

             - 2 (n + 3) (n + b) U(n) - (n + 1 + b) n U(n + 1) = 0

SEE ALSO: HYPERG, IsHyperg
```

12.2 Hypergeometric series

x

```
FUNCTION: HYP - how generalized HYPergeometric series are coded

SYNOPSIS:
- HYP is an indexed name: HYP[ l1, l2, z ] with

     - l1, the list of upper parameters,
     - l2, the list of lower parameters,
     - z, the evaluation point.

EXAMPLES:
> with(HYPERG):
> H := HYP[[a,b,c],[d,e],z]:
> IsHYP( H );

                                      true

SEE ALSO: HYPERG,   IsHYP,   VHYP,    RF,  HypConverg,  HypDiff,  HypEval,
```

```
          HypOrder, HypPerm, HypSimplify,  HypToVHyp,   AddParam
```

```
FUNCTION: IsHYP - test if is a hypergeometric series

CALLING SEQUENCE:
  IsHYP(shyp)
  HYPERG[IsHYP](shyp)

PARAMETERS:
  shyp - one term

SYNOPSIS:
- The  IsHYP  function  tests  whether a term is  a hypergeometric series.

- If a lower parameter is a negative or null integer, the function gives a
  warning.

- Whenever  there  is  a  conflict  between  the function name  IsHYP  and
  another  name  used in the same session,  use the form  HYPERG['IsHYP'].

EXAMPLES:
> with(HYPERG):
> IsHYP( HYP[[a,b,c],[d,e],z] );

                                     true

> IsHYP( HYP[[a,2,c],[d,-3],z] );

       warning HYP: denominator values must not be nonpositive integers

                                     true

SEE ALSO: HYPERG,  HYP, VHYP, RF,  HypConverg, HypDiff, HypEval, HypOrder,
          HypPerm, HypSimplify,    HypToVHyp,  AddParam, IsVHYP, IsRF
```

```
FUNCTION: HypConverg - test if is a convergent hypergeometric series

CALLING SEQUENCE:
  HypConverg( shyp )
  HYPERG[HypConverg]( shyp )

PARAMETERS:
  shyp - a hypergeometric series in standard notation (HYP or VHYP)

SYNOPSIS:
- The  HypConverg  function  tests  whether  a  hypergeometric  series  is
  convergent. The  return  value  is  "true" or "false".

- When it is imprecis,  the function prints restrictions  in order for the
  series to be convergent.
```

```
- Whenever there is a conflict between the function name HypConverg
  and another name used in the same session, use the long form
  HYPERG['HypConverg'].

EXAMPLES:
> with(HYPERG):
> HypConverg( HYP[[a,b],[c],z] );
# abs(z) must be less than 1
# or z must be egal to 1 and 0 < c-a-b
# or z must be egal to -1 and -1 < c-a-b

                                    true

> HypConverg( HYP[[1,3],[4],z] );
# abs(z) must be less than 1
# or z must be egal to -1

                                    true

> HypConverg( HYP[[a],[b,c],z] );

                                    true

> HypConverg( HYP[[a,b,c],[d],z] );
# z must be egal to 0

                                    true

SEE ALSO: HYPERG, IsHYP, IsVHYP, HypDiff, HypEval, HypOrder, HypType
```

```
FUNCTION: HypEval - Rule that transforms a HYP[] into a Sum()

CALLING SEQUENCE:
  HypEval( expr )
  HYPERG[HypEval]( expr )

  HypEval( expr, tsi )
  HYPERG[HypEval]( expr, tsi )

PARAMETERS:
  expr - any expression
  tsi  - a name, the summation index

SYNOPSIS:
- The HypEval function expands hypergeometric series into sums. It takes
  in input any expression involving HYP (generalized HYPergeometric
  series) and VHYP (HYPergeometric series in Very-well-poised order).

- The optional second argument (tsi) allows you to choose the name of the
  summation index. The default value is k.

- Whenever there is a conflict between the function name HypEval and
  another name used in the same session, use the form HYPERG['HypEval'].
```

```
EXAMPLES:
> with(HYPERG):
> HypEval( c * d! * HYP[[a,b],[c],z] );

                          /infinity                    \
                          | -----                     k|
                          |  \      RF[a, k] RF[b, k] z |
                     c d! |   )     --------------------|
                          |  /          RF[c, k] k!     |
                          | -----                       |
                          \ k = 0                       /

> HypEval( term * VHYP[a,[e,f],z^2] );

           /infinity                                           \
           | -----                                          2 k|
           |  \      RF[a, k] RF[1/2 a + 1, k] RF[e, k] RF[f, k] (z ) |
      term |   )     -------------------------------------------------|
           |  /      RF[1/2 a, k] RF[1 + a - e, k] RF[1 + a - f, k] k!|
           | -----                                                 |
           \ k = 0                                                 /

> #special case:
> HypEval( HYP[[],[],z] );

                                     exp(z)

SEE ALSO: HYPERG,  HYP,  IsHYP,  VHYP,  IsVHYP,  AddParam,  HypContig,
          HypConverg, HypDiff, HypOrder, HypPerm, HypSimplify, HypType
```

12.3 Rising factorial

```
FUNCTION: RF - how Rising Factorials are coded

SYNOPSIS:
- RF is an indexed name:  RF[a, k]  where a and k are names (or numerical',
  values).

- The mathematical notation is (a)_k.

EXAMPLES:
> with(HYPERG):
> rf := RF[a,4];

                                 rf := RF[a, 4]

> RfEval( rf );

                              a (a + 1) (a + 2) (a + 3)
```

SEE ALSO: HYPERG, IsRF, RfEval, HYP, VHYP

FUNCTION: IsRF - test if is a rising factorial

CALLING SEQUENCE:
  IsRF(rf)
  HYPERG[IsRF](rf)

PARAMETERS:
  rf - one term

SYNOPSIS:
- The IsRF function tests whether a term is a rising factorial.

- Whenever there is a conflict between the function name IsRF and another name used in the same session, use the long form HYPERG['IsRF'].

EXAMPLES:
```
> with(HYPERG):
> IsRF( RF[a,b] );

                                       true
> IsRF( RF[a,-3] );

                                       true

> IsRF( RF[a,3.2] );

            warning RF: the second argument is not an integer, 3.2

                                      false
```

SEE ALSO: HYPERG, RF, RfEval, IsHYP, IsVHYP

FUNCTION: RfEval - Rule that evals rising factorials

CALLING SEQUENCE:
  RfEval(expr)
  HYPERG[RfEval](expr)

PARAMETERS:
  expr - any expression

SYNOPSIS:
- The RfEval function evals (if possible) rising factorial (RF[]).

- Whenever there is a conflict between the function name RfEval and another name used in the same session, use the form HYPERG['RfEval'].

EXAMPLES:
```
> with(HYPERG):
```

```
> RfEval( RF[a,k] );

                                    GAMMA(a + k)
                                    ------------
                                      GAMMA(a)

> RfEval( RF[b,-3] * RF[a,4] );

                             a (a + 1) (a + 2) (a + 3)
                             -------------------------
                              (b - 3) (b - 2) (b - 1)

> RfEval( RF[4,-4] );  # 1/((4-1)*(4-2)*(4-3)*(4-4))
Error, (in HYPERG/RfEval/basic) division by zero

> RfEval( RF[4,3] );   # 4*(4+1)*(4+2) = 120

                                        120
> RfEval( RF[0,k] );

                                      RF[0, k]

> RfEval( RF[0,0] );

                                         1

> RfEval( RF[0,-3] );

                                        -1/6

> RfEval( RF[0,5] );

                                         0

SEE ALSO: HYPERG,  RF, IsRF, HYP, IsHYP, VHYP, IsVHYP, HypEval, HypSimplify
```

12.4 Manipulation of hypergeometric series

```
FUNCTION: AddParam - add a parameter to a [basic] hypergeometric series
                   - in standard notation

CALLING SEQUENCE:
  AddParam( shyp, p )
  HYPERG[AddParam]( shyp, p )

PARAMETERS:
  shyp - a [basic] hypergeometric series in standard notation
  p    - a parameter

SYNOPSIS:
- Add  the parameter  p  to the list of upper parameters,  and to the list
```

```
  of lower parameters.

- p must be a nonnegative integer.

- Whenever there is a conflict between  the  function  name  AddParam  and
  another name used in the same session, use the form  HYPERG['AddParam'].

EXAMPLES:
> with(HYPERG):
> AddParam( HYP[[a,b],[d],1], c );

                         c, must be an integer > 0.

                         HYP[[c, a, b], [c, d], 1]

SEE ALSO: HYPERG,  IsHYP, HypConverg, HypDiff, HypEval, HypOrder, HypPerm,
          HypSimplify, IsQHYP, QHypEval, QHypOrder
```

```
FUNCTION: HypSimplify - Apply simplification rules
                      - to [basic] hypergeometric series

CALLING SEQUENCE:
  HypSimplify( expr )
  HYPERG[HypSimplify]( expr )

  HypSimplify( expr, 'factor' )
  HYPERG[HypSimplify]( expr, 'factor' )

  HypSimplify( expr, 'combpow' )
  HYPERG[HypSimplify]( expr, 'combpow' )

PARAMETERS:
  expr - any expression

SYNOPSIS:
- The  HypSimplify  function  cancels equal upper and lower parameters  of
  ordinary (and basic)  hypergeometric  series.  Moreover,  this  function
  expands all upper and lower parameters.

- It  is   also   used   to  simplify  any [q] hypergeometric  expression.

- The  'combpow'  option  is used  to combine  terms  with  powers  in the
  evaluation point of hypergeometric [q]-series.

- The  'factor'  option   is   used   to   factorize   evaluation    point
  of hypergeometric [q]-series.

- Whenever  there is a conflict  between  the  function  name  HypSimplify
  and another  name  used  in  the  same  session,  use  the  long  form
  HYPERG['HypSimplify'].

EXAMPLES:
> with(HYPERG):
```

```
> HypSimplify( QHYP[[a,b],[b],q,z] );

                              QHYP[[a], [], q, z]

> HypSimplify( HYP[[a,b,c],[c],z] + HYP[[a,b,d],[d],z] );

                              2 HYP[[a, b], [], z]

> HypSimplify( HYP[[a,b,c],[d,c],1/4*(-4*x+4*x^2+1)/(x*(x-1))],'factor' );

                                                      2
                                             (2 x - 1)
                         HYP[[a, b], [d], 1/4 ----------]
                                              x (x - 1)

> hyp := HYP[[a, b],[d],1/4*((2*x-1)^2)^(1/2)/x/(x-1)];

                                                            2 1/2
                                                 ((2 x - 1) )
                   hyp := HYP[[a, b], [d], 1/4 ---------------]
                                                    x (x - 1)

> HypSimplify( hyp, 'combpow' );

                                              2 x - 1
                         HYP[[a, b], [d], 1/4 ---------]
                                             x (x - 1)

SEE ALSO: HYPERG,    IsHYP,  AddParam,   HypConverg,   HypDiff,   HypEval,
          HypOrder,  HypPerm
```

X

```
FUNCTION: HypPermBoth - Rules for permuting parameters
          HypPermUp   - in [basic] hypergeometric series
          HypPermLow

CALLING SEQUENCE:
  HypPermUp( shyp, p )
  HYPERG[HypPermUp]( shyp, p )

  HypPermLow( shyp, p )
  HYPERG[HypPermLow]( shyp, p )

  HypPermBoth( shyp, p )
  HYPERG[HypPermBoth]( shyp, p )

PARAMETERS:
  shyp - a [basic] hypergeometric series in standard notation
  p    - a permutation

SYNOPSIS:
- p must be a list of positive numbers forming a permutation.

- The effect is that the new (upper/lower) parameter at position i  is the
```

```
  old parameter form position p[i].

- For the   HypPermBoth  function, the hypergeometric series must have one
  more upper parameter than it has lower parameters.

- Whenever there is a conflict  between the function name   HypPermXX  and
  another name used in the same session, use the form HYPERG['HypPermXX'].

EXAMPLES:
> with(HYPERG):
> HypPermLow( HYP[[a,b,d,c],[e,f,g],z], [2,3,1] );

                     HYP[[a, b, d, c], [f, g, e], z]

> HypPermBoth( QHYP[[a,b,c],[d,e],q,z], [2,1] );

                       QHYP[[a, c, b], [e, d], q, z]

SEE ALSO: HYPERG,  HYP,      AddParam,  HypConverg, HypDiff,  HypSimplify,
          HypEval, HypOrder, HypType,  QHYP, QHypEval, QHypOrder, QHypType
```

```
FUNCTION: HypDiff - Differentiation of a hypergeometric series

CALLING SEQUENCE:
  HypDiff( shyp, var )
  HYPERG[HypDiff]( shyp, var )

PARAMETERS:
  shyp - a hypergeometric series in standard notation (HYP or VHYP)
  var  - a name

SYNOPSIS:
- The  HypDiff function computes the derivative of a hypergeometric series
  with respect to var.

- Whenever there is a conflict  between  the  function  name  HypDiff  and
  another name used in the same session,  use the form  HYPERG['HypDiff'].

EXAMPLES:
> with(HYPERG):
> HypDiff( HYP[[a,b,c],[d,e],z], z );

             a b c HYP[[a + 1, b + 1, c + 1], [d + 1, e + 1], z]
             ---------------------------------------------------
                                     d e

> HypDiff( HYP[[a,b],[c],3*z^4], z );

                     3                                      4
                    z  a b HYP[[a + 1, b + 1], [c + 1], 3 z ]
                 12 -----------------------------------------
                                        c
```

```
SEE ALSO: HYPERG, IsHYP, IsVHYP, HypConverg, HypEval, HypOrder, HypType
```

```
FUNCTION: FirstTerms - extracts first terms of a summation

CALLING SEQUENCE:
  FirstTerms( s, n )
  HYPERG[FirstTerms]( s, n )

PARAMETERS:
  s - a Sum  -or- a HYPergeometric series
  n - a positive integer

SYNOPSIS:
- The  FirstTerms  function  extracts  the 'n' first terms of a summation.

- Whenever there is a conflict  between the function name  FirstTerms  and
  another name use in the same session, use the form HYPERG['FirstTerms'].

EXAMPLES:
> with(HYPERG):
> h1 := Sum(a[i],i=5..N):
> FirstTerms( h1, 3);

                                          /  N      \
                                          |-----    |
                                          | \       |
                    a[5] + a[6] + a[7] +  |  )   a[i]|
                                          | /       |
                                          |-----    |
                                          \i = 8    /

> h2 := HYP[[a,2],[3],z]:
> FirstTerms( h2, 4);

                                                                       2
       RF[a, 0] RF[2, 0]   RF[a, 1] RF[2, 1] z       RF[a, 2] RF[2, 2] z
       ----------------- + ------------------- + 1/2 --------------------
            RF[3, 0]             RF[3, 1]                  RF[3, 2]

                                      /infinity                   \
                                  3   | -----                    k|
                RF[a, 3] RF[2, 3] z   |  \      RF[a, k] RF[2, k] z |
           + 1/6 -------------------- + |   )     --------------------|
                      RF[3, 3]        |  /          RF[3, k] k!    |
                                      | -----                      |
                                      \ k = 4                      /

SEE ALSO: HYPERG, HYP, IsHYP, HypConverg, HypEval, Sum
```

12.5 Very-well-poised hypergeometric series

```
FUNCTION: VHYP - how HYPergeometric series (in Very-well-poised) are coded

SYNOPSIS:
- VHYP is an indexed name: HYP[ a, l, z ] with

     - a, the first upper parameter the hypergeometric series,
     - l, a list of parameters,
     - z, the evaluation point.

- VHYP[ a, [p_1,...,p_r], z ] is equivalent to
  HYP[ [a, 1+a/2, p_1, ..., p_r], [1/2,1+a-p_1, ..., 1+a-p_r], z]

EXAMPLES:
> with(HYPERG):
> v := VHYP[a,[3,b-1],z]:
> VHypToHyp( v );

         HYP[[a, 1/2 a + 1, 3, b - 1], [1/2 a, - 2 + a, 2 + a - b], z]

SEE ALSO: HYPERG,    IsVHYP,  HYP,  RF,  HypConverg,  HypEval,
          VHyptoHyp, HypToVHyp
```

x

```
FUNCTION: IsVHYP - test if is a hypergeometric series
                 - in very-well-poised order

CALLING SEQUENCE:
  IsVHYP(hyp)
  HYPERG[IsVHYP](hyp)

PARAMETERS:
  hyp - one term

SYNOPSIS:
- The  IsVHYP  function tests whether a term is a hypergeometric series in
  'very-well-poised' order.

- If a lower parameter is a negative or null integer, the function gives a
  warning.

- Whenever there  is  a  conflict  between the  function name  IsVHYP  and
  another name used  in the same session,  use the form  HYPERG['IsVHYP'].

EXAMPLES:
> with(HYPERG):
> v1 := VHYP[a,[b,c],z];

                            v1 := VHYP[a, [b, c], z]

> IsVHYP( v1 );

                                      true

> v2 := VHYP[-3,[2,4,-3],z];
```

```
                         v2 := VHYP[-3, [2, 4, -3], z]

> IsVHYP( v2 );

                                    false

SEE ALSO: HYPERG,  VHYP, HYP, RF, HypConverg, HypDiff, HypEval, VHypToHyp,
          HypToVHyp, IsHYP, IsRF
```

```
FUNCTION: HypOrder - Order parameters of a hypergeometric series

CALLING SEQUENCE:
  HypOrder( shyp )
  HYPERG[HypOrder]( shyp )

PARAMETERS:
  shyp - a hypergeometric series in standard notation

SYNOPSIS:
- The   HypOrder   function   is  used  to  order  the  parameters  of a
  hypergeometric  series   in   'well-poised',   'very-well-poised',   and
  'nearly-well poised order'.

- Whenever there is a conflict  between the  function  name  HypOrder  and
  another name used in the same session, use the form  HYPERG['HypOrder'].

EXAMPLES:
> with(HYPERG):
> # well-poised order
> HypOrder( HYP[[a,b,c,1],[a,a+1-c,a+1-b],z] );

                HYP[[a, b, c, 1], [a + 1 - b, a + 1 - c, a], z]

> # very-well-poised order
> HypOrder( HYP[[-n,b,1+a/2,a],[a+1-b,a/2,a+1+n],z] );

         HYP[[a, 1 + 1/2 a, - n, b], [1/2 a, n + a + 1, a + 1 - b], z]

> # nearly-poised order
> HypOrder( HYP[[x+1,-n-1/2,-n],[x-2*n,3/2],1] );

               HYP[[- n, - n - 1/2, x + 1], [3/2, x - 2 n], 1]

SEE ALSO: HYPERG, IsHYP, IsVHYP,  HypConverg,  HypEval,  HypPerm, HypType,
          AddParam
```

```
FUNCTION: HypType - Print the type of a hypergeometric series

CALLING SEQUENCE:
  HypType( shyp )
```

```
  HYPERG[HypType]( shyp )

PARAMETERS:
  shyp - a hypergeometric series in standard notation (HYP or VHYP)

SYNOPSIS:
- The  HypType  function  is used to know  the  type  of  a hypergeometric
  series   ('well-poised', 'very-well-poised', 'nearly-well poised order',
  or 'ordinay').

- The  function  prints  the  type  of  the  series, and  the return value
  is always NULL.

- Whenever there is a conflict  between  the  function  name  HypType  and
  another name used in the same session,  use the form  HYPERG['HypType'].

EXAMPLES:
> with(HYPERG):
> # very-well poised order
> HypType( HYP[[-n,b,1+a/2,a],[a+1-b,a/2,a+1+n],z] );

                   Very-well poised hypergeometric series

> # well-poised order
> HypType( HYP[[a,b,c,1],[a,a+1-c,a+1-b],z] );

                    Well-poised hypergeometric series

> # nearly-poised order
> HypType( HYP[[x+1,-n-1/2,-n],[x-2*n,3/2],1] );

              Nearly-poised (first kind) hypergeometric series

> # very-well poised order
> HypType( VHYP[a,[b,2+c,d],z] );

                   Very-well poised hypergeometric series

SEE ALSO: HYPERG,  IsHYP,  IsVHYP, HypConverg, HypEval, HypOrder, HypPerm,
          AddParam
```

```
FUNCTION: HypToVHyp - convert hypergeometric series
                    - into hypergeometric series in very-well-poised order

CALLING SEQUENCE:
  HypToVHyp( shyp )
  HYPERG[HypToVHyp]( shyp )

PARAMETERS:
  shyp - a hypergeometric series in standard notation

SYNOPSIS:
- This  function  tries to convert  (if possible)  a hypergeometric series
```

```
  into a hypergeometric series in well-poised order.

- If the convertion is impossible, the return value of the function is the
  initial hypergeometric series.

- Whenever there is a  conflict  between the function name  HypToVHyp  and
  another name used in the same session, use the form HYPERG['HypToVHyp'].

EXAMPLES:
> with(HYPERG):
> HypToVHyp( HYP[[a,a/2+1,b,c],[a/2,1+a-b,1+a-c],z] );

                              VHYP[a, [b, c], z]

> HypToVHyp( HYP[[a,b,c],[d,e],z] );

            warning: not a Very-well poised hypergeometric series

                          HYP[[a, b, c], [d, e], z]

SEE ALSO: HYPERG, HYP, IsHYP, VHYP, IsVHYP, VHypToHyp, HypConverg, HypEval
```

```
FUNCTION: VHypToHyp - convert hypergeometric series in very-well-poised order
                    - into hypergeometric series in standard notation

CALLING SEQUENCE:
  VHypToHyp( vhyp )
  HYPERG[VHypToHyp]( vhyp )

PARAMETERS:
  vhyp - a hypergeometric series in very-well-poised order

SYNOPSIS:
- This function converts  a hypergeometric  series  in  'very-well-poised'
  order    into    a    hypergeometric    series    in    standard    notation.

- Whenever there is a  conflict  between the function name  VHypToHyp  and
  another name used in the same session, use the form HYPERG['VHypToHyp'].

EXAMPLES:
> with(HYPERG):
> VHypToHyp( VHYP[a,[],z] );

                       HYP[[a, 1/2 a + 1], [1/2 a], z]

> VHypToHyp( VHYP[x+1,[b,c],z] );

   HYP[[x + 1, 1/2 x + 3/2, b, c], [1/2 x + 1/2, 2 + x - b, 2 + x - c], z]

SEE ALSO: HYPERG, VHYP, IsVHYP, HYP, IsHYP, HypToVHyp, HypConverg, HypEval
```

12.6 Manipulation of hypergeometric expressions

```
FUNCTIONS: MapList & MapApply
           - Functions for controlled application of rules and functions

CALLING SEQUENCE:
  MapList( expr )
  HYPERG[MapList]( expr )

  MapList( expr, level )
  HYPERG[MapList]( expr, level )

  MapApply( fun, pos, expr, seq_args )
  HYPERG[MapApply]( fun, pos, expr, seq_args )

PARAMETERS:
  expr     - any expression
  level    - an optional positive integer
  fun      - a function
  pos      - a set of list of integers
  seq_args - an optional sequence of parameters

SYNOPSIS:
- The  MapList  function  returns   a  set  of  subexpressions  of  'expr'
  together with  their respective positions in 'expr'.  You can choose the
  depth in the analysis expression with 'level'  (the default value is 1).

- This helps to apply  rules or functions  to specific subexpressions with
  MapApply (see below).

- Let 'seq_args' be the  following  sequence  a_1, a_2, a_3, ..., a_n  and
  let 'pos' be the following set {p_1,p_2,...,p_k}  where p_i is a list of
  integers.  Let 'pi' be [e1,e2,...,eS] (for example), then  the  MapApply
  function  applies  the 'fun' function  to  the  following  subexpression
  op(e1,op(e2,...,op(eS,expr)...))  with the  optional arguments a_1, a_2,
  a_3, ... and a_n.

- Whenever  there  is  a conflict  between a function name   MapApply   or
  MapList  and  another  name used  in the same session, use the long form
  HYPERG['MapApply'] or HYPERG['MapList'].

EXAMPLES:
> with(HYPERG):
> expr := RF[a,3] * HYP[[a,b,c],[d,e],z] * RF[b,-2]:
> MapList( expr );

     {[RF[a, 3], [1]], [HYP[[a, b, c], [d, e], z], [2]], [RF[b, -2], [3]]}

> MapApply( RfEval, {[1],[3]}, expr );

                     a (a + 1) (a + 2) HYP[[a, b, c], [d, e], z]
                     -------------------------------------------
```

```
                                (b - 2) (b - 1)

> exprbis := HYP[[a,b],[e],1] + HYP[[a,c],[d],1]:
> MapList( exprbis );

           {[HYP[[a, b], [e], 1], [1]], [HYP[[a, c], [d], 1], [2]]}

> # apply Vandermonde's theorem (S2101) to the second term
> res := MapApply( HypSum, {[2]}, exprbis, 1 );

                          c, must be an integer <= 0

                                                RF[d - a, - c]
                  res := HYP[[a, b], [e], 1] + --------------
                                                  RF[d, - c]

> MapList( res, 2 );

                                                                 1
 {[HYP[[a, b], [e], 1], [1]], [[RF[d-a,-c], [2, 1]], [--------, [2, 2]]]}
                                                             RF[d,-c]

> MapApply( RfEval, {[2,2]}, res );

                                        RF[d - a, - c] GAMMA(d)
                 HYP[[a, b], [e], 1] + -----------------------
                                              GAMMA(d - c)

SEE ALSO: HYPERG, HypEval, RfEval
```

12.7 Manipulation of recurrence relations

```
FUNCTION: RecOrder - computes the order of a recurrence equation

CALLING SEQUENCE:
  RecOrder( rec, u(n) )
  HYPERG[RecOrder]( rec, u(n) )

PARAMETERS:
  rec - a linear recurrence equation with polynomial coefficients
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The  RecOrder  function  takes  as input  a  recurrence  equation  'rec'
  and a sequence 'u(n)'. It computes the order of the recurrence equation.

- Whenever there is a conflict between  the  function  name  RecOrder  and
  another name use in the same session, use  the  form  HYPERG['RecOrder'].

EXAMPLES:
> with(HYPERG):
> RecOrder( U(n-1)+n*U(n+1)+3*n*U(n+2)=2*n, U(n) );
```

```
                                               3

SEE ALSO: HYPERG,   SubsRec,  CheckRec,  SimplifyRec,  ShiftRec,
          Ishomog,  Homog,    GenRec,    HypergToRec
```

```
FUNCTION: SubsRec - substitutes (in a recurrence relation)
                  - the sequence by an expression

CALLING SEQUENCE:

  SubsRec( rec, u(n), expr )
  HYPERG[SubsRec]( rec, u(n), expr )

PARAMETERS:
  rec  - a linear recurrence equation with polynomial coefficients
  u,n  - the name and the index of the recurrence
  expr - an expression

SYNOPSIS:
- The  SubsRec  function  substitutes  (in the  recurrence  relation)  the
  sequence 'u(n)' by the expression 'expr'.

- The 'expr' expression must not involve the sequence u(n).

- Whenever there is a conflict  between  the  function  name  SubsRec  and
  another name use in the same session, use  the  form  HYPERG['SubsRec'].

EXAMPLES:
> with(HYPERG):
> rec := SubsRec( (n+2)*U(n+1)-n*U(n)=n, U(n), V(n-1)*n ):
> SimplifyRec( rec, V(n) );

                                                 2
                    (n + 2) V(n) (n + 1) - n  V(n - 1) = n

> SubsRec( (n+2)*U(n+1)-n*U(n)=n, U(n), 2^n );

                                   n      n
                               n 2  + 4 2  = n

SEE ALSO: HYPERG,   RecOrder,   CheckRec,   SimplifyRec,   ShiftRec,
          Ishomog,  Homog,      GenRec,     HypergToRec
```

```
FUNCTION: CheckRec - tests if an expression satisfies
                   - a recurrence relation

CALLING SEQUENCE:

  CheckRec( rec, u(n), expr )
  HYPERG[CheckRec]( rec, u(n), expr )
```

```
PARAMETERS:
  rec  - a linear recurrence equation with polynomial coefficients
  u,n  - the name and the index of the recurrence
  expr - an expression

SYNOPSIS:
- The  CheckRec  function  tests  if an expression  'expr'  satisfies  the
  recurrence  equation  'rec'.  The  return  value  is  "true" or "false".

- Whenever there is a conflict  between the  function  name  CheckRec  and
  another name use in the same session,  use the form  HYPERG['CheckRec'].

EXAMPLES:
> with(HYPERG):
> CheckRec( (n+2)*U(n+1)-n*U(n)=n, U(n), 2^n );

                                      false

> CheckRec( C(n)-2*C(n+1)+C(n+2)=0, C(n), n );

                                       true

SEE ALSO: HYPERG,     RecOrder,   SubsRec,   SimplifyRec,   ShiftRec
          Ishomog,    Homog,      GenRec,    HypergToRec
```

```
FUNCTION: SimplifyRec - simplifies a recurrence equation

CALLING SEQUENCE:
  SimplifyRec( rec, u(n) )
  HYPERG[SimplifyRec]( rec, u(n) )

  SimplifyRec( rec, u(n), 'factor' )
  HYPERG[SimplifyRec]( rec, u(n), 'factor' )

PARAMETERS:
  rec - a linear recurrence equation with polynomial coefficients
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The  SimplifyRec  function simplifies the recurrence equation.  It takes
  as input a recurrence equation 'rec' and a sequence 'u(n)'  and  makes a
  recurrence relation simpler: combines equal recurrence terms and factors
  the polynomial coefficients.

- The   'factor'   option   is   used  to  delete  factors  not  dependent
  of the sequence 'u(n)'.

- Whenever there is a conflict between the function name  SimplifyRec  and
  another name use in the same session, use the form HYPERG['SimplifyRec'].

EXAMPLES:
> with(HYPERG):
```

```
> SimplifyRec( U(n)+U(n+2)+n*U(n)=3*n, U(n) );

                        (n + 1) U(n) + U(n + 2) = 3 n

> SimplifyRec( (-1+4*n)*(n-2)*C(n)+(n-2)*C(n+2)=-3*n^2+3*n+6,
                                                      C(n), 'factor');

                   (- 1 + 4 n) C(n) + C(n + 2) = - 3 n - 3,

SEE ALSO: HYPERG, RecOrder, CheckRec, ShiftRec, Ishomog, Homog, GenRec,
          HypergToRec
```

```
FUNCTION: ShiftRec - shifts a recurrence equation

CALLING SEQUENCE:
  ShiftRec( rec, u(n) )
  HYPERG[ShiftRec]( rec, u(n) )

  ShiftRec( rec, u(n), shift )
  HYPERG[ShiftRec]( rec, u(n), shift )

PARAMETERS:
  rec   - a linear recurrence equation with polynomial coefficients
  u,n   - the name and the index of the recurrence
  shift - an integer

SYNOPSIS:
- The  ShiftRec  function  shifts the recurrence equation.  It returns the
  recurrence with variable 'n' replaced by variable 'n + shift'.

- The  optional  third argument  indicates  the  shift value.   It  can be
  a positive or negative value  (for a forward or a backward shift).   The
  default value is 1.

- Whenever there is a conflict  between the  function  name  ShiftRec  and
  another name use in the same session,  use the form  HYPERG['ShiftRec'].

EXAMPLES:
> with(HYPERG):
> ShiftRec( (n+2)*U(n+1)-n*U(n)=n, U(n) );

                (- n - 1) U(n + 1) + (n + 3) U(n + 2) = n + 1

> ShiftRec( U(n-1)+n*U(n+1)+3*n*U(n+2)=2*n, U(n), 3 );

          U(n + 2) + (n + 3) U(n + 4) + (3 n + 9) U(n + 5) = 2 n + 6

SEE ALSO: HYPERG,    RecOrder,   SubsRec,   CheckRec,   SimplifyRec,
          Ishomog,   Homog,      GenRec,    HypergToRec
```

```
FUNCTION: IsHomog - tests if a recurrence equation is homogeneous
```

```
CALLING SEQUENCE:
  IsHomog( rec, u(n) )
  HYPERG[IsHomog]( rec, u(n) )

PARAMETERS:
  rec - a linear recurrence equation with polynomial coefficients
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The IsHomog function tests if a recurrence equation is homogeneous.  The
  return value is "true" or "false".

- Whenever there is a  conflict  between the  function  name  IsHomog  and
  another name used in the same session,  use the form  HYPERG['IsHomog'].

EXAMPLES:
> with(HYPERG):
> IsHomog( 2*S(n)-S(n+1)=0, S(n) );

                                    true

> IsHomog( (n+2)*T(n+1)-n*T(n)=n, T(n) );

                                   false

SEE ALSO: HYPERG,    Homog,   RecOrder,  CheckRec,  SubsRec,  SimplifyRec,
          ShiftRec,  GenRec,  HypergToRec
```

```
FUNCTION: Homog - Homogenizes a recurrence equation

CALLING SEQUENCE:
  Homog( rec, u(n) )
  HYPERG[Homog]( rec, u(n) )

PARAMETERS:
  rec - a linear recurrence equation with polynomial coefficients
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The  Homog  function homogenizes the recurrence equation.

- Whenever there  is  a  conflict  between the  function  name  Homog  and
  another name  used  in the same session,  use the form  HYPERG['Homog'].

EXAMPLES:
> with(HYPERG):
> Homog( (n+2)*U(n+1)-n*U(n)=n, U(n) );

                                   2
         n (n + 1) U(n) - 2 (n + 1)  U(n + 1) + n (n + 3) U(n + 2) = 0

SEE ALSO: HYPERG,  Ishomog,  RecOrder,  CheckRec,  SubsRec,  ShiftRec
```

`SimplifyRec`, `GenRec`, `HypergToRec`

Chapter 13

Transforming sums into hypergeometric notation

x

```
FUNCTION: SumToHyp - convert summations into hypergeometric series

CALLING SEQUENCE:
  SumToHyp( expr )
  HYPERG[SumToHyp]( expr )

PARAMETERS:
  expr - a summation (Sum)

SYNOPSIS:
- The  SumToHyp  function converts (if possible) a Sum into hypergeometric
  standard    notation    (precisely    into   an    expression    involving
  a hypergeometric series with real or complex parameters).

- It also applies  the  HypOrder  function  to order the parameters of the
  hypergeometric    series    in    'well-poised',    'very-well-poised'    or
  'nearly-poised' order.

- Whenever there is  a  conflict  between the function name  SumToHyp  and
  another name used in the same session,  use the form HYPERG['SumToHyp'].

EXAMPLES:
> with(HYPERG):
> SumToHyp( Sum((-1)^k*binomial(2*n,k)^2,k=0..infinity) );

                        HYP[[- 2 n, - 2 n], [1], -1]

> SumToHyp( Sum((k^2+4)/(k+2),k) );

            2 HYP[[1 - 2 I, 2, 1 + 2 I, 1], [- 2 I, 3, 2 I], 1]

SEE ALSO: HYPERG, HYP, IsHYP, HypConverg, HypEval
```

Chapter 14

Manipulating factorial expressions

x

```
FUNCTIONS: BaseSplit    Ext1         Ext2         Inv
           Linear1      Linear2      Neg1         Neg2
           Split        Trans
           - Rules to handle any factorial expression (Gamma function,
           - and Rising Factorial)

CALLING SEQUENCE:
  BaseSplit( expr, split )        HYPERG[BaseSplit]( expr, split )
  Ext1( expr, top_ext )           HYPERG[Ext1]( expr, top_ext )
  Ext2( expr, bottom_ext )        HYPERG[Ext2]( expr, bottom_ext )
  Inv( expr )                     HYPERG[Inv]( expr )
  Linear1( expr )                 HYPERG[Linear1]( expr )
  Linear2( expr )                 HYPERG[Linear2]( expr )
  Neg1( expr )                    HYPERG[Neg1]( expr )
  Neg2( expr )                    HYPERG[Neg2]( expr )
  Trans( expr )                   HYPERG[Trans]( expr )
  Split( expr, bottom_split )     HYPERG[Split]( expr, bottom_split )

PARAMETERS:
  expr                                     - any (factorial) expression
  split, top_ext, bottom_ext, bottom_split - an integer (or a name)

SYNOPSIS:
- These functions apply basic rules:

  x BaseSplit( expr, split )
    o RF[a,n]  = split^n * product(RF[(a+k)/split,n/split], k=0..split-1);
    o GAMMA(a) = split^(a-1/2) * (2*Pi)^((1-split)/2)
                               * product(GAMMA((a+k)/split),k=0..split-1);

  x Ext1( expr, top_ext )
    o RF[a,n] = GAMMA(a+n)/GAMMA(a)               if top_ext = infinity
    o RF[a,n] = RF[a,top_ext+n]/RF[a+n,top_ext]   if top_ext in an integer

  x Ext2( expr, bottom_ext )
    o RF[a,n]  = RF[a-bottom_ext,bottom_ext+n]/RF[a-bottom_ext,bottom_ext]
    o GAMMA(a) = GAMMA(a-bottom_ext)*RF[a-bottom_ext,bottom_ext]
```

```
  x Inv( expr )
    o GAMMA(z) = Pi/sin(Pi*z)/GAMMA(1-z)

  x Linear1( expr )
    o RF[a,n] = a*RF[a+1,n-1]
    o GAMMA(a) = (a-1)*GAMMA(a-1)

  x Linear2( expr )
    o RF[a,n] = (a+n-1)*RF[a,n-1]

  x Neg1( expr )
    o RF[a,n] = 1/RF[a+n,-n]

  x Neg2( expr )
    o RF[a,n] = (-1)^n/RF[1-a,-n]

  x Trans( expr )
    o RF[a,n] = (-1)^n*RF[1-n-a,n]

  x Split( expr, bottom_split )
    o RF[a,n]  = RF[a,bottom_split]*RF[a+bottom_split,n-bottom_split]
    o GAMMA(a) = GAMMA(a+bottom_split)/RF[a,bottom_split]

- Whenever there is a conflict between a function name XX and another name
  used in the same session, use the long form HYPERG['XX']

EXAMPLES:
> with(HYPERG):
> BaseSplit( GAMMA(2*c), 3 );

           (2 c - 1/2)
          3            GAMMA(2/3 c) GAMMA(2/3 c + 1/3) GAMMA(2/3 c + 2/3)
      1/2 ---------------------------------------------------------------
                                      Pi

> Neg1( 3/RF[a,n] );

                             3 RF[a + n, - n]

> Split( a*GAMMA(a)*RF[a,b]*GAMMA(b), m );

                  a GAMMA(a + m) GAMMA(b + m) RF[a + m, b - m]
                  --------------------------------------------
                                   RF[b, m]

SEE ALSO: HYPERG, HypEval, HypSimplify, RfEval
```

Chapter 15

Applying summation formulas

```
FUNCTION: HypSum - Summation formula in form of rules

CALLING SEQUENCE:
  HypSum( shyp, nb )
  HYPERG[HypSum]( shyp, nb )

PARAMETERS:
  shyp - a hypergeometric series in standard notation
  nb   - a formula number

SYNOPSIS:
- The  HypSum  function  applies  a summation formula  on a hypergeometric
  series.

- Each  summation  formula  has its own number  (see details in the user's
  reference manual).

- Several  formulas  has  'special'  parameters,  so  it  can be useful to
  modify the parameters' position with the functions HypPermLow, HypPermUp
  or HypPermBoth before apply HypSum.

- The   HypSum   function   also   displays  the  condition(s)  about  the
  evaluation point and/or the parameters of hypergeometric series in order
  for the formula to be valid.

- Whenever there  is  a  conflict  between the function  name  HypSum  and
  another name used in the same session,  use the form   HYPERG['HypSum'].

EXAMPLES:
> with(HYPERG):
> #Gauss's theorem
> HypSum( HYP[[a,b],[c],z], 3 );

                             z, must be equal to, 1

                             a + b, must be < to, c
```

```
                          GAMMA[c] GAMMA[c - a - b]
                          -------------------------
                          GAMMA[c - a] GAMMA[c - b]

SEE ALSO: HYPERG, HYP, IsHYP, HypSumList, HypSumPrint, HypPerm, HypConverg
          HypContig,   HypContigPrint,    HypTransf,   HypTransfList,
          HypTransfPrint,   MapList,      MapApply
```

```
FUNCTION: HypSumList - gives a list of applicable summation formulas

CALLING SEQUENCE:
  HypSumList( shyp )
  HYPERG[HypSumList]( shyp )

PARAMETERS:
  shyp - a hypergeometric series in standard notation

SYNOPSIS:
- The  HypSumList  function  returns   (for a given hypergeometric series)
  a list of applicable summation formulas.

- See the user's reference manual for details on formula numbering.

- Whenever  there is a conflict between the function name  HypSumList  and
  another  name   used   in   the   same  session,    use  the  long  form
  HYPERG['HypSumList'].

EXAMPLES:
> with(HYPERG):
> HypSumList( HYP[[a,b],[d],1] );

  [[S2101], [HypPermUp, [2, 1], S2101], [S2103], [HypPermUp, [2, 1], S2103],

   [S2105], [HypPermUp, [2, 1], S2105]]

SEE ALSO: HYPERG,     HYP,    IsHYP,   HypSum,   HypSumPrint,   HypContig,
          HypContigPrint,    HypConverg,    HypTransf,     HypTransfList,
          HypTransfPrint
```

```
FUNCTION: HypSumPrint - Print a summation formula in form of an equation

CALLING SEQUENCE:
  HypSumPrint( shyp, nb )
  HYPERG[HypSumPrint]( shyp, nb )

PARAMETERS:
  shyp - a hypergeometric series in standard notation
  nb   - a formula number

SYNOPSIS:
```

- The HypSumPrint function prints a summation formula in form of an equation.

- Each summation formula has its own number (see the user's reference manual). It is the same numbering as that in HypSum(shyp, nb).

- Whenever there is a conflict between the function name HypSumPrint and another name used in the same session, use the long form HYPERG['HypSumPrint'].

```
EXAMPLES:
> with(HYPERG):
> HypSumPrint( HYP[[a,b,c],[d,e],z], 2 );

                              z, must be equal to, 1

                            a, must be an integer <= 0

                            1 + a - b, must be = to, d

                            1 + a - c, must be = to, e

          HYP[[a, b, c], [d, e], z] =

              RF[1, - a] RF[b, - 1/2 a] RF[c, - 1/2 a] RF[b + c, - a]
              -------------------------------------------------------
              RF[1, - 1/2 a] RF[b, - a] RF[c, - a] RF[b + c, - 1/2 a]
```

SEE ALSO: HYPERG, HYP, IsHYP, HypSum, HypSumList, HypContig, HypContigPrint, HypConverg, HypTransf, HypTransfList, HypTransfPrint

Chapter 16

Applying transformation formulas

```
FUNCTION: HypTransf - Transformation formula in form of rules

CALLING SEQUENCE:
  HypTransf( shyp, nb )
  HYPERG[HypTransf]( shyp, nb )

PARAMETERS:
  shyp - a hypergeometric series in standard notation
  nb   - a formula number

SYNOPSIS:
- The    HypTransf    function   applies   a   transformation   formula
  on a hypergeometric series.

- Each  transformation  formula  has  its  own  number (see details in the
  user's reference manual).

- Several  formulas  has  'special'  parameters,  so  it  can be useful to
  modify the parameters' position with the functions HypPermLow, HypPermUp
  or HypPermBoth before apply HypTransf.

- The  HypTransf  function   also  displays  the  condition(s)  about  the
  evaluation point and/or the parameters of hypergeometric series in order
  for the formula to be valid.

- Whenever there is  a  conflict between the function name  HypTransf  and
  another name used in the same session, use the form HYPERG['HypTransf'].

EXAMPLES:
> with(HYPERG):
> HypTransf( HYP[[a,b,c],[d,e],z], 7 );

                          c, must be an integer <= 0

                            z, must be equal to, 1

           RF[- b + e, - c] HYP[[c, b, d - a], [d, 1 + b - e + c], 1]
```

```
                    -----------------------------------------------------------
                                             RF[e, - c]

SEE ALSO: HYPERG, HYP, IsHYP,  HypTransfList,  HypTransfPrint,  HypContig,
          HypContigPrint,   HypConverg,   HypPerm,   HypSum,   HypSumList,
          HypSumPrint,      MapList,      MapApply
```

```
FUNCTION: HypTransfList - gives a list of applicable transform. formulas

CALLING SEQUENCE:
  HypTransfList( shyp )
  HYPERG[HypTransfList]( shyp )

PARAMETERS:
  shyp - a hypergeometric series in standard notation

SYNOPSIS:
- The  HypTransfList  function returns  (for a given hypergeometric series)
  a list of applicable transformation formulas.

- See the user's reference manual for details on formula numbering.

- Whenever there is a conflict between the function name HypTransfList and
  another  name   used   in   the   same  session,    use  the  long  form
  HYPERG['HypTransfList'].

EXAMPLES:
> with(HYPERG):
> HypTransfList( HYP[[1,1],[2],z] );

[[T2103], [HypPermUp, [2, 1], T2103], [T2104], [HypPermUp, [2, 1], T2104],

 [T2136], [HypPermUp, [2, 1], T2136], [T2137], [HypPermUp, [2, 1], T2137]]

SEE ALSO: HYPERG,   HYP,  IsHYP,   HypTransf,  HypTransfPrint,  HypContig,
          HypContigPrint, HypConverg, HypSum,  HypSumList,     HypSumPrint
```

```
FUNCTION: HypTransfPrint - Print a transformation formula
                         - in   form   of   an   equation

CALLING SEQUENCE:
  HypTransfPrint( shyp, nb )
  HYPERG[HypTransfPrint]( shyp, nb )

PARAMETERS:
  shyp - a hypergeometric series in standard notation
  nb   - a formula number

SYNOPSIS:
- The  HypTransfPrint  function prints a  transformation formula  in  form
  of an equation.
```

- Each transformation formula has its own number (see details in the user's reference manual). It is the same numbering as that in HypTransf(shyp, nb).

- Whenever there is a conflict between the function name HypTransfPrint and another name used in the same session, use the long form HYPERG['HypTransfPrint'].

EXAMPLES:

```
> with(HYPERG):
> HypTransfPrint( HYP[[a,b,c],[d,e],z], 4 );

                             z, must be equal to, 1

        HYP[[a, b, c], [d, e], z] = GAMMA(e) GAMMA(- a - b - c + d + e)

            HYP[[a, d - b, d - c], [d, - b - c + d + e], 1]/

            (GAMMA(- a + e) GAMMA(- b - c + d + e))
```

SEE ALSO: HYPERG, HYP, IsHYP, HypTransf, HypTransfList, HypContig, HypContigPrint, HypConverg, HypSum, HypSumList, HypSumPrint

Chapter 17

Applying contiguous formulas

```
FUNCTION: HypContig - Contiguous formula in form of rules

CALLING SEQUENCE:
  HypContig( shyp, nb )
  HYPERG[HypContig]( shyp, nb )

  HypContig( shyp, nb, pos_1, ..., pos_i )
  HYPERG[HypContig]( shyp, nb, pos_1, ..., pos_i )

PARAMETERS:
  shyp       - a hypergeometric series in standard notation
  nb         - a formula number
  pos_j, ... - upper/lower parameter positions

SYNOPSIS:
- This function applies a  contiguous  formula on a hypergeometric series.

- Each  contiguous formula  has its own number  (see details in the user's
  reference manual).

- The variables  pos_i  are  the positions of the  'special' (upper or/and
  lower) parameters.

- Whenever there is a conflict  between the function  name  HypContig  and
  another name used in the same session, use the form HYPERG['HypContig'].

EXAMPLES:
> with(HYPERG):
> HypContig( HYP[[a,b,c],[d,e],z], 26, 1,3 );

       c HYP[[a, c + 1, b], [d, e], z]   a HYP[[a + 1, c, b], [d, e], z]
       ------------------------------- + -------------------------------
                    c - a                             a - c

SEE ALSO: HYPERG,  HYP,  IsHYP,      HypContigPrint,  HypConverg,  HypSum,
          HypSumPrint,   HypTransf,  HypTransfPrint,  MapList,  MapApply
```

```
FUNCTION: HypContigPrint - Print a contiguous formula
                         - in  form  of  an  equation

CALLING SEQUENCE:
  HypContigPrint( shyp, nb )
  HYPERG[HypContigPrint]( shyp, nb )

  HypContigPrint( shyp, nb, pos_1, ..., pos_i )
  HYPERG[HypContigPrint]( shyp, nb, pos_1, ..., pos_i )

PARAMETERS:
  shyp       - a hypergeometric series in standard notation
  nb         - a formula number
  pos_j, ... - upper/lower parameter positions

SYNOPSIS:
- The  HypContigPrint  function prints a contiguous  formula in form of an`,
  equation.

- Each  contiguous formula  has its own number  (see details in the user's
  reference manual).

- The variables  pos_i  are  the positions of the  'special' (upper or/and
  lower) parameters.

- Whenever there is a conflict  between the function  name  HypContigPrint
  and another name used in the same session, use the long form
  HYPERG['HypContigPrint'].

EXAMPLES:
> with(HYPERG):
> HypContigPrint(HYP[[a,b],[c],z], 21, 1);

      HYP[[a, b], [c], z] =

                                  z a b HYP[[a + 1, b + 1], [c + 1], z]
          HYP[[a, b], [c - 1], z] - -------------------------------------
                                                (c - 1) c

SEE ALSO: HYPERG,     HYP,  IsHYP,  HypContig,       HypConverg,   HypSum,
          HypTransf,  HypSumPrint,  HypTransfPrint,  MapList,  MapApply
```

Chapter 18

Computing formal limits of hypergeometric expressions

```
FUNCTION: Lim - compute formal limit of hypergeometric expressions

CALLING SEQUENCE:
  Lim( f, x=a )
  HYPERG[Lim]( f, x=a )

PARAMETERS:
  f - an algebraic expression
  x - a name
  a - an algebraic expression (limit point, possibly + or -infinity)

SYNOPSIS:
- The  Lim  function attempts to  compute  the  limiting value of  f  as x
  approaches a.

- If  Lim  cannot find a closed form for the limit, then the prettyprinter
  displays  the limit function  using  a two-dimensional format  (with  an
  asympotic estimate for f).

- Whenever  there  is  a  conflict  between the function  name   Lim   and
  another name used in the same session, use the long form  HYPERG['Lim'].

EXAMPLES:
> with(HYPERG):
> Lim( HYP[[a,b,-n],[c,1+a+b-c-n],1], b=infinity );

                          HYP[[a, - n], [c], 1]

> Lim( HYP[[a1,a2,lbd*u],[b1,b2],z/lbd], lbd=infinity );

                        HYP[[a1, a2], [b1, b2], u z]

> Lim( GAMMA(a+m)/GAMMA(b+m), m=infinity );

                                         (a - b)
```

```
                    limit          m
                    m -> infinity
```

SEE ALSO: HYPERG, HYP, IsHYP, HypEval, RfEval

Chapter 19

Finding polynomial, rational and hypergeometric solutions of recurrences

x

```
FUNCTION: PolySolRec - linear recurrence equation solver
                     - polynomial solutions

CALLING SEQUENCE:
  PolySolRec(eqn, u(n))
  HYPERG[PolySolRec](eq, u(n))

  PolySolRec(eqn, u(n), inits)
  HYPERG[PolySolRec](eq, u(n), inits)

PARAMETERS:
  eqn     - a linear recurrence equation with polynomial coefficients
  u,n     - the name and the index of the recurrence
  inits   - a set of initial conditions

SYNOPSIS:
- The function PolySolRec finds all polynomial solutions of the recurrence
  relation.

- The  first  argument  should be  a  single  recurrence  equation  with
  polynomial coefficients.  This recurrence isn't necessarily homogeneous.

  Any expressions  which is not an equation will be understood to be equal
  to zero.

- The second argument indicates the sequence that PolySolRec  should solve
  for.  A  sequence   is   represented   by   a   name   and   an   index.

- The output is  a polynom in  n  with constant coefficients _x0, _x1, ...

- The optional  inits  argument is a  set  of initial conditions. Each one
  may be specified in the following way: u(a)=v.  If insufficient boundary
```

```
  conditions are given,  PolySolRec uses symbolic names as default values.

- If PolySolRec is unable to compute a polynomial solution, it returns the
  message 'No polynomial solution'. This means that there is  no solution.

- Whenever there is a conflict between the  function  name  PolySolRec and
  another name used in the same session, use the form HYPERG['PolySolRec'].

EXAMPLES:
> with(HYPERG):
> PolySolRec( 3*U(n+2)-2*U(n+1)-U(n)=-1, U(n) );

                             _x0 - 1/4 n

> PolySolRec( n*U(n+1)-(n+8)*U(n), U(n) );

       _x8 n (n + 7) (n + 6) (n + 5) (n + 4) (n + 3) (n + 2) (1 + n)

> PolySolRec( r(n+2)-2*r(n+1)+r(n)=2, r(n), {r(0)=-1,r(1)=m} );

                                           2
                             - 1 + m n + n

SEE ALSO: HYPERG,  SimplifyRec,  ShiftRec,   SubsRec,  CheckRec,   GenRec,
          RatioSolRec,  HypSolRec
```

```
FUNCTION: RatioSolRec - linear recurrence equation solver
                      - rational solutions

CALLING SEQUENCE:
  RatioSolRec(eqn, u(n))
  HYPERG[RatioSolRec](eq, u(n))

  RatioSolRec(eqn, u(n), inits)
  HYPERG[RatioSolRec](eq, u(n), inits)

PARAMETERS:
  eqn   - a linear recurrence equation with polynomial coefficients
  u,n   - the name and the index of the recurrence
  inits - a set of initial conditions

SYNOPSIS:
- The function RatioSolRec  finds all rational solutions of the recurrence
  relation.

- The  first  argument  should  be   a  single  recurrence  equation  with
  polynomial coefficients.  This recurrence isn't necessarily homogeneous.

  Any expressions  which is not an equation will be understood to be equal
  to zero.

- The second argument  indicates  the sequence   that  RatioSolRec  should
  solve  for.   A  sequence  is  represented  by  a  name  and  an  index.
```

```
- The output is a rational function in  n  with constant coefficients _x0,
  _x1, _x2, _x3, ...

- The optional  third  argument is a  set  of initial conditions. Each one
  may be specified in the following way: u(a)=v.  If insufficient boundary
  conditions are given, RatioSolRec uses symbolic names as default values.

- If  RatioSolRec  is unable to compute  a rational solution,   it returns
  the  message  'No rational solution'.  This  means  that  there  is   no
  solution.

- Whenever there is a conflict between the function name  RatioSolRec  and
  another   name   used   in   the   same  session,   use  the  long  form
  HYPERG['RatioSolRec'].

EXAMPLES:
> with(HYPERG):
> RatioSolRec( (n+4)*U(n+2)+U(n+1)-(n+1)*U(n), U(n) );

                                    _x0
                              ---------------
                              (n + 2) (n + 1)

> RatioSolRec( (n+4)*U(n+2)+U(n+1)-(n+1)*U(n), U(n), {U(1)=1} );

                                     6
                              ---------------
                              (n + 2) (n + 1)

SEE ALSO: HYPERG,  SimplifyRec,  ShiftRec,   SubsRec,  CheckRec,   GenRec,
          PolySolRec,   HypSolRec
```

```
FUNCTION: HypSolRec - linear recurrence equation solver
                    - hypergeometric solutions

CALLING SEQUENCE:
  HypSolRec(eqn, u(n))
  HYPERG[HypSolRec](eq, u(n))

PARAMETERS:
  eqn - a linear recurrence equation with polynomial coefficients
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The function   HypSolRec   finds  all  hypergeometric  solutions  of the
  recurrence relation.

- The  first  argument  should  be   a  single  recurrence  equation  with
  polynomial  coefficients. Any expressions  which is not an equation will
  be understood to be equal to zero.

  This recurrence must be homogeneous.
```

- The second argument indicates the sequence that HypSolRec should solve for. (A sequence is represented by a name and an index).

- The output is a generating set (not necessarily linearly independent) of hypergeometric solutions (with constant coefficients _x0, _x1, ...) of the recurrence relation. These solutions u(n) are given by their rational representations u(n+1)/u(n).

- Whenever there is a conflict between the function name HypSolRec and another name used in the same session, use the form HYPERG['HypSolRec'].

EXAMPLES:
```
> with(HYPERG):
> HypSolRec( n*(1+n)*U(2+n)-2*n*(1+k+n)*U(1+n)+(k+n)*(1+k+n)*U(n), U(n) );

                (k + n) (_x0 + _x1 (1 + n))  1 + k + n  k + n
               {---------------------------, ---------, -----}
                      n (_x0 + _x1 n)            n      n - 1
```

SEE ALSO: HYPERG, SimplifyRec, ShiftRec, SubsRec, CheckRec, GenRec, PolySolRec, RatioSolRec

---

FUNCTION: GenRec - generate a recurrence

CALLING SEQUENCE:
```
  GenRec( sols, u(n) )
  HYPERG[GenRec]( sols, u(n) )
```

PARAMETERS:
  sols - a set of 'p' terms {s1, s2, ..., sp}
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The function GenRec generates a p-order homogeneous recurrence satisfied by u(n) such that s1, s2, ..., and sp are its solutions.

- The s1, s2, ..., sp terms must be products of rational functions factorials, rising factorials, binomial coefficients and power functions.

- The output is a recurrence in n with polynomial coefficients.

- If GenRec is unable to compute a recurrence, it returns 0 = 0.

- Whenever there is a conflict between the function name GenRec and another name used in the same session, use the form HYPERG['GenRec'].

EXAMPLES:
```
> with(HYPERG):
> GenRec( {2^n,n}, V(n) );

            2 V(n) n + (- 3 n + 2) V(n + 1) + (n - 1) V(n + 2) = 0
```

```
> GenRec( {k,k^2/(k+1)/(k-1)}, U(k) );

                                      2
     - (k - 1) (k + 2) (k + 1) (k  + 3 k + 3) U(k)

                 2                2
          + 2 k  (k + 1) (k + 2)  U(k + 1)

                                     2
          - k (k + 3) (k + 1) (k  + k + 1) U(k + 2) = 0
```

SEE ALSO: HYPERG, RecOrder, SubsRec, CheckRec, ShiftRec, SimplifyRec, PolySolRec, RatioSolRec, HypSolRec

Chapter 20

Using the Gosper and Zeilberger algorithms

x

```
FUNCTION: Gosper - Gosper's algorithm for summation

CALLING SEQUENCE:
  Gosper(f, k)
  HYPERG[Gosper](f, k)

PARAMETERS:
  f - an expression
  k - a name

SYNOPSIS:
- This function is an  implementation of  Gosper's algorithm,  it computes
  an upward antidifference of a the expression f.

- An expression g is called 'antidifference' of f if
  f = subs(k=k+1,g) - g

- Whenever  there is a conflict  between  the  function  name  Gosper  and
  another name used in the same session, use  the  form  HYPERG['Gosper'].

EXAMPLES:
> with(HYPERG):
> Gosper( binomial(n,k), n );

                              (n - k) n!
                          -------------------
                          (k + 1) k! (n - k)!

> Gosper(1/k,k);

                                 FAIL

SEE ALSO: HYPERG, HypToRec, Prove, SummandToRec, SumToRec
```

x

```
FUNCTION: SummandToRec - Zeilberger's algorithm

CALLING SEQUENCE:
  SummandToRec(s, k, n, F, G, 'cert')
  HYPERG[SummandToRec](s, k, n, F, G, 'cert')

  SummandToRec(s, k, n, F, G, 'maxorder=m_o', 'cert')
  HYPERG[SummandToRec](s, k, n, F, G, 'maxorder=m_o', 'cert')

  SummandToRec(s, k, n, F, G, 'recorder=r_o', 'cert')
  HYPERG[SummandToRec](s, k, n, F, G, 'recorder=r_o', 'cert')

PARAMETERS:
  s        - a summand
  k, n     - names
  F,G,     - names
  m_o, r_o - integers
  cert     - an optional name (formal parameter)

SYNOPSIS:
- This function is an implementation of Zeilberger's algorithm,  computing
  a recurrence equation for the summand 's'.

- The required condition is that the summand is hypergeometric in n and k.

- The  output  'rec(n)'  is  a linear recurrence  equation with polynomial
  coefficients such that:

                   rec(n) F(n,k) = G(n, k + 1) - G(n, k),

  where G(n,k)=cert(n,k)*F(n,k).

- The optional name  cert(n,k)  is called the certificate.

- The output  rec(n)  is  a recurrence satisfied by s(n).
  The order of the recurrence is:
     o either less or equals to m_o (where the default value of m_o is 6),
     o either equals to r_o.

- If  this  algorithm  can't  deliver  a  such recurrence, then it returns
  the message 'No recurrence equation of order XXX'.

- Whenever  there  is a conflict between the  function  name  SummandToRec
  and  another  name  used  in  the  same  session,  use  the  long  form
  HYPERG['SummandToRec'].

EXAMPLES:
> with(HYPERG):
> SummandToRec(k*binomial(2*n+1,2*k+1),k,n,F,G,'certificate');

 - 4 n (2 n + 1) F(n, k) + n (2 n - 1) F(n + 1, k) = G(n, k + 1) - G(n, k)

> certificate;
```

```
                                      2
         (2 k + 1) (k - 1) (- 6 n  - 1 - 7 n + 4 n k + 2 k)
         --------------------------------------------------
                   (- n - 1 + k) (- 2 n - 1 + 2 k)
```

SEE ALSO: HYPERG, Gosper, HypToRec, Prove, SumToRec

x

```
FUNCTION: SumToRec - Zeilberger's algorithm

CALLING SEQUENCE:
  SumToRec(a, n, s, 'cert')
  HYPERG[SumToRec](a, n, s, 'cert')

  SumToRec(a, n, s, 'maxorder=m_o', 'cert')
  HYPERG[SumToRec](a, n, s, 'maxorder=m_o', 'cert')

  SumToRec(a, n, s, 'recorder=r_o', 'cert')
  HYPERG[SumToRec](a, n, s, 'recorder=r_o', 'cert')

PARAMETERS:
  a        - a summation
  n,       - a name
  s,       - a name
  m_o, r_o - integers
  cert     - an optional name (formal parameter)

SYNOPSIS:
- This function is an implementation of Zeilberger's algorithm,  computing
  a recurrence equation for the sum

                             -----
                              \
                 s(n) := a :=  )   F(n, k)
                              /
                             -----
                               k

- The  required  condition  is  that F(n,k) is  hypergeometric in n and k.

- The  output  'rec(n)'  is  a linear recurrence  equation with polynomial
  coefficients such that:

                   rec(n) F(n,k) = G(n, k + 1) - G(n, k),

  where G(n,k)=cert(n,k)*F(n,k).

- The optional name 'cert(n,k)' is called the certificate.

- The output  rec(n)  is  a recurrence satisfied by s(n).
  The order of the recurrence is:
     o either less or equals to m_o (where the default value of m_o is 6),
     o either equals to r_o.
```

```
- If  this  algorithm  can't  deliver  a  such recurrence, then it returns
  the message 'No recurrence equation of order XXX'.

- Whenever  there is a conflict between the function name   SumToRec   and
  another name used in the same session, use the form  HYPERG['SumToRec'].

EXAMPLES:
> with(HYPERG):
> SumToRec( Sum(binomial(n,k),k), n, A );

                            2 A(n) - A(n + 1) = 0

> SumToRec( Sum(binomial(n,k),k), n, A, 'recorder=2' );

                      - 2 A(n) - A(n + 1) + A(n + 2) = 0

> SumToRec( Sum(binomial(n,k),k), n, A, 'recorder=3', 'certificate' );

                2 A(n) - 7 A(n + 1) + A(n + 2) + A(n + 3) = 0

> certificate;

                                     2                2
                       (5 + 10 n + 3 n  - n k + 2 k - k ) k
                     - ------------------------------------
                        (n + 3 - k) (n + 2 - k) (n + 1 - k)

> SumToRec( Sum(binomial(n,k)*binomial(a,k)*binomial(b,k),k), n, A );

        - 2 (n + 2) (n + 1) A(n) - (n + 2) (a + b - 9 - 5 n) A(n + 1)

                               2
             + (2 b - 19 n - 4 n  + n b - a b + n a + 2 a - 23) A(n + 2)

                     2
             + (n + 3)  A(n + 3) = 0

SEE ALSO: HYPERG, Gosper, HypToRec, Prove, SummandToRec
):

#print(INTERFACE_HELP('insert', 'topic'=`HYPERG/SumToRec
#                     'helpfile'=`.
```

```
FUNCTION: HypToRec - Zeilberger's algorithm

CALLING SEQUENCE:
  HypToRec(shyp, n, s, 'cert')
  HYPERG[HypToRec](shyp, n, s, 'cert')

  HypToRec(shyp, n, s, 'maxorder=m_o', 'cert')
  HYPERG[HypToRec](shyp, n, s, 'maxorder=m_o', 'cert')

  HypToRec(shyp, n, s, 'recorder=r_o', 'cert')
```

```
  HYPERG[HypToRec](shyp, n, s, 'recorder=r_o', 'cert')

PARAMETERS:
  shyp     - a standard hypergeometric series
  n        - a name
  s        - a name
  m_o, r_o - integers
  cert     - an optional name (formal parameter)

SYNOPSIS:
- This function is an implementation of Zeilberger's algorithm,  computing
  a recurrence  equation  for the  standard  hypergeometric  series  shyp.

- The output 'rec(n)'  is  a linear recurrence  equation  with  polynomial
  coefficients.  The  optional  name  'cert'  is  called  the certificate.

- The output  rec(n)  is  a recurrence satisfied by the series shyp.`,
  The order of the recurrence is:
     o either less or equals to m_o (where the default value of m_o is 6),
     o either equals to r_o.

- If  this  algorithm  can't  deliver  a  such recurrence, then it returns
  the message 'No recurrence equation of order XXX'.`,

- Whenever there is a conflict  between the function name   HypToRec   and
  another name used in the same session, use the form  HYPERG['HypToRec'].

EXAMPLES:
> with(HYPERG):
> HypToRec( HYP[[a,b],[d],1], a, A );

               (d - 1 - a) A(a) + (b - d + 1 + a) A(a + 1) = 0

SEE ALSO: HYPERG, Gosper, Prove, SummandToRec, SumToRec
```

```
FUNCTION: Prove - Zeilberger's algorithm (automatic proof)

CALLING SEQUENCE:
  Prove(F, k, n)
  HYPERG[Prove](F, k, n)

  Prove(F, k, n, 'filename=fname')
  HYPERG[Prove](F, k, n, 'filename=fname')

  Prove(F, k, n, 'mode=out')
  HYPERG[Prove](F, k, n, 'mode=out')

PARAMETERS:
  F     - a summand
  k,    - a name
  n,    - a name
  fname - a (optional) name
  out   - a (optional) name
```

```
SYNOPSIS:
- This  function  is  similar of  'SumToRec' function  but  writes a proof
  in a txt or latex file.

- The first option is the output file name,  (by default, 'HYPERG.proof').

- The other option is the file type: txt or latex (the default one is txt)

- This function is an implementation of Zeilberger's algorithm,  computing
  a recurrence equation for the sum

                                 -----
                                  \
                  SUM(n) := a :=   )   F(n, k)
                                  /
                                 -----
                                   k

- The  required  condition  is  that F(n,k) is  hypergeometric in n and k.

- Whenever  there is a  conflict  between  the  function  name  Prove  and
  another name used in the same session,  use  the  form  HYPERG['Prove'].

EXAMPLES:
> with(HYPERG):
> Prove(binomial(n,k),k,n);

                 Creating automatic proof in file HYPERG.proof

> Prove(binomial(x,k)*binomial(y,n-k),k,n,'Myproof');

                    Creating automatic proof in file Myproof

SEE ALSO: HYPERG, Gosper, HypToRec, SummandToRec, SumToRec
```

Chapter 21

Miscellaneous functions

```
FUNCTION: Time - gives time information

CALLING SEQUENCE:
  Time(f, 't', arg_1, ..., arg_k)
  HYPERG[Time](f, 't', arg_1, ..., arg_k)

PARAMETERS:
  f                  - any name of function
  t                  - any name of variable (formal parameter)
  arg_1, ..., arg_k  - optional extra arguments

SYNOPSIS:
- The Time function runs f(arg_1,...,arg_k) and saves in the parameter 't'
  the computation time.

- The return value of Time(f,'t',arg_1,...,arg_k)  is  f(arg_1,...,arg_k).

- For composite functions, transform it into a single function in order to
  use Time.

- Whenever there  is  a  conflict  between  the  function  name  Time  and
  another name used in the same session, use the long form HYPERG['Time'].

EXAMPLES:
> with(HYPERG):
> Time( HypSolRec, 't', y(n) = (n-1)*y(n-1) + (n-1)*y(n-2), y(n) );

                                   {n + 1}

> t;

                                     .167

SEE ALSO: HYPERG, HYP, RF
```

```
> Time( PolySolRec, 't', n*U(n+1)-(n+100)*U(n)=0,U(n) );

_x100 n (n + 99) (n + 98) (n + 97) (n + 96) (n + 95) (n + 94) (n + 93) (n + 92)

    (n + 91) (n + 90) (n + 89) (n + 88) (n + 87) (n + 86) (n + 85) (n + 84)

    (n + 83) (n + 82) (n + 81) (n + 80) (n + 79) (n + 78) (n + 77) (n + 76)

    (n + 75) (n + 74) (n + 73) (n + 72) (n + 71) (n + 70) (n + 69) (n + 68)

    (n + 67) (n + 66) (n + 65) (n + 64) (n + 63) (n + 62) (n + 61) (n + 60)

    (n + 59) (n + 58) (n + 57) (n + 56) (n + 55) (n + 54) (n + 53) (n + 52)

    (n + 51) (n + 50) (n + 49) (n + 48) (n + 47) (n + 46) (n + 45) (n + 44)

    (n + 43) (n + 42) (n + 41) (n + 40) (n + 39) (n + 38) (n + 37) (n + 36)

    (n + 35) (n + 34) (n + 33) (n + 32) (n + 31) (n + 30) (n + 29) (n + 28)

    (n + 27) (n + 26) (n + 25) (n + 24) (n + 23) (n + 22) (n + 21) (n + 20)

    (n + 19) (n + 18) (n + 17) (n + 16) (n + 15) (n + 14) (n + 13) (n + 12)

    (n + 11) (n + 10) (n + 9) (n + 8) (n + 7) (n + 6) (n + 5) (n + 4) (n + 3)

    (n + 2) (n + 1)

> t;

                                   116.300
```

Part IV

Examples

Chapter 22

Hypergeometric series and manipulation

22.1 Vandermonde sum

```
> s := Sum( binomial(M,k)*binomial(N,R-k), k );

                        -----
                         \
                  s :=    )   binomial(M, k) binomial(N, R - k)
                         /
                        -----
                          k

> expr := SumToHyp( s );

             expr := binomial(N, R) HYP[[- R, - M], [N - R + 1], 1]

> expr := HypSum( expr, 1 );

                        - R, must be an integer <= 0.

                             binomial(N, R) RF[N + 1, M]
                    expr := ---------------------------
                                   RF[N - R + 1, M]

> expr := RfEval( expr );

                 binomial(N, R) GAMMA(N + 1 + M) GAMMA(N - R + 1)
          expr := ------------------------------------------------
                         GAMMA(N + 1) GAMMA(N - R + 1 + M)

> simplify( expr, GAMMA );

                               GAMMA(N + 1 + M)
                        ---------------------------------
                        GAMMA(R + 1) GAMMA(N - R + 1 + M)
```

So, we have an evaluation of the *Vandermonde* sum:

$$\sum_k \binom{M}{k} \binom{N}{R-k} = \frac{\Gamma(1+N+M)}{\Gamma(M+N+1-R)\,\Gamma(R+1)} \ .$$

22.2 Orthogonal polynomials

- Orthogonal *Chebyshev* polynomials (first type): $T_n^{(1)}(z) = {}_2F_1\left[\begin{matrix}-n, n\\ \frac{1}{2}\end{matrix}; \frac{(1-z)}{2}\right]$, satisfy the following recurrence:

```
> HypToRec( HYP[[-n,n],[1/2],(1-z)/2], n, CH );     # Chebyshev polynomial Type I

                   - CH(n) + 2 CH(n + 1) z - CH(n + 2) = 0
```

$$T_n^{(1)}(z) - 2\,z\,T_{n+1}^{(1)}(z) + T_{n+2}^{(1)}(z) = 0.$$

- Orthogonal *Hermite* polynomials

```
> HypToRec( (2*x)^n*HYP[[-n/2,-(n-1)/2],[],-1/x^2], n, S );  # Hermite polynomial

                 (2 n + 2) S(n) - 2 x S(n + 1) + S(n + 2) = 0
```

- Orthogonal *Legendre* or *spherical* polynomials
  We want to check that the two following different expressions of *Legendre* polynomials are identical:

$$P_n(x) = \frac{1}{2^n}\sum_k \binom{n}{k}^2 (x-1)^k\,(x+1)^(n-k)\ , \qquad Q_n(x) = {}_2F_1\left[\begin{matrix}-n, n+1\\ 1\end{matrix}; \frac{1-x}{2}\right]\ .$$

```
> P := 1/2^n*Sum(binomial(n,k)^2*(x-1)^k*(x+1)^(n-k),k):
> P := SumToHyp( P );

                                      n                     x - 1
                P := (1/2 x + 1/2)  HYP[[- n, - n], [1], -----]
                                                            x + 1

> Q := HYP[[-n,n+1],[1],(1-x)/2];

                    Q := HYP[[- n, n + 1], [1], 1/2 - 1/2 x]

> Qt := HypTransf(Q,4);

                                n                     1/2 - 1/2 x
               (1/2 x + 1/2)  HYP[[- n, - n], [1], - -----------]
                                                     1/2 x + 1/2

> HypSimplify( Qt );

                                    n                     x - 1
                   (1/2 x + 1/2)  HYP[[- n, - n], [1], -----]
                                                          x + 1
```

- A limit relation between hypergeometric orthogonal polynomials
  The continous dual Han polynomials can be found from the Wilson polynomials defined by

$$W_n(x) = \frac{{}_4F_3\left[\begin{matrix}-n,n+a+b+c+d-1,a+ix,a-ix\\a+b,a+c,a+d\end{matrix};1\right]}{(a+b)_n\,(a+c)_n\,(a+d)_n}$$

by dividing by $(a+d)_n$ and letting $d \to \infty$:

$$\lim_{d\longrightarrow\infty}\frac{W_n(x)}{(a+d)_n} = \frac{{}_3F_2\left[\begin{matrix}-n,,a+ix,a-ix\\a+b,a+c\end{matrix};1\right]}{(a+b)_n\,(a+c)_n}\ , \tag{22.1}$$

$$= S_n(x)\ , \tag{22.2}$$

```
> Lim( HYP[[-n,n+a+b+c+d-1,a+I*x,a-I*x],[a+b,a+c,a+d],1]
      * RF[a+b,n] * RF[a+c,n] * RF[a+d,n] / RF[a+d,n],   d=infinity);

GAMMA(a + b + n) GAMMA(a + c + n)

    HYP[[- n, a + I x, a - I x], [a + b, a + c], 1]/(GAMMA(a + b) GAMMA(a + c))
```

22.3 Ramanujan hypergeometric identity

$$1-\left(\frac{1}{2}\right)^3+\left(\frac{1/2\;3/2}{2}\right)^3+\cdots = {}_3F_2\left[\begin{matrix}1/2,1/2,1/2\\1,1\end{matrix};-1\right]$$

$$= \left({}_2F_1\left[\begin{matrix}1/4,1/4\\1\end{matrix};-1\right]\right)^2$$

$$= \frac{\Gamma(\frac{9}{8})^2}{\Gamma(\frac{5}{4})^2\,\Gamma(\frac{7}{8})^2}\ .$$

```
> H := HYP[[1/2,1/2,1/2],[1,1],-1];

                    H := HYP[[1/2, 1/2, 1/2], [1, 1], -1]

> Ht := HypTransf( H, 34 );

                                                  2
                     Ht := HYP[[1/4, 1/4], [1], -1]

> HypSum( Ht, 4 );

                               1/2              2
                         (2 + 2   ) GAMMA(3/4)
                     1/4 ----------------------
                                           4
                               GAMMA(7/8)
```

Chapter 23

Automatic proofs of identities

This chapter gives several examples for proving, discovering hypergeometric identities (applying formulas on series, solving recurrences,...).

23.1 Recurrences

∘ The following identities can be proved by use of Zeilberger's algorithm.

- $\sum_k \binom{n}{k} = 2^n$,
- $\sum_{k=0}^{n} \binom{n+k}{k} 2^{-k} = 2^n$,
- $\sum_k \binom{n}{k} \binom{b}{k} = \frac{(n+b)!}{n!\, b!}$,
- $\sum_{i=0}^{l} \binom{k}{i} \binom{l}{i} \binom{n+k+l-i}{k+l} = \binom{n+k}{k} \binom{n+l}{l}$,
- $k = l \qquad \sum_{i=0}^{k} \binom{k}{i}^2 \binom{n+2\,k-i}{2\,k} = \binom{n+k}{k}^2$.

For example (the last one),

```
> s := Sum( binomial(k,i)^2*binomial(n+2*k-i,2*k)/binomial(n+k,k)^2, i );

                    -----               2
                     \     binomial(k, i)  binomial(n + 2 k - i, 2 k)
              s :=    )    ----------------------------------------
                     /                                    2
                    -----              binomial(n + k, k)
                      i

> SumToRec( s, k, S );

                              S(k) - S(k + 1) = 0

> S(0) := eval( subs(i=0, k=0, binomial(k,i)^2*binomial(n+2*k-i,2*k)/binomial(n+k,k)^2 ) );

                                   S(0) := 1
```

$$\begin{cases} S(k) - S(k+1) &= 0 \ , \\ S(0) &= 1 \ . \end{cases}$$

Finally: $S(k+1) = S(k) = \cdots = S(0) = 1$. The identity is proved.

Here is an another example of Zeilberger's algorithm:

```
> id := HypTransfPrint( HYP[[a,b],[c],z], 3 );

                                          (c - a - b)
  id := HYP[[a, b], [c], z] = (1 - z)            HYP[[c - a, c - b], [c], z]

> HypToRec( op(1,id), a, L );

           (c - a - 1) L(a) + (z b - z + 2 - c - z a + 2 a) L(a + 1)

                + (z - 1) (a + 1) L(a + 2) = 0

> HypToRec( op(2,id), a, R );

         (- c + a + 1) R(a) + (- z b + z - 2 + c + z a - 2 a) R(a + 1)

              - (z - 1) (a + 1) R(a + 2) = 0
```

The two sides of this **T2103** transformation formula satisfy the same recurrence of second order, this proves its validity. This verification gives an extra argument for `HYPERG`'s tools.

∘ We consider the following *double* sum studied in [20]:

$$\forall\, m, n \in \mathbb{N}\ , \quad \sum_{j,k} (-1)^{j+k} \binom{j+k}{j} \binom{r}{j} \binom{n}{k} \binom{m+n-j-k}{m-j} = \binom{n+r}{n} \binom{m-r}{m-n}\ ,$$

We apply the following elementary inversion formula:

$$\sum_j (-1)^j \binom{r}{j} f(j) = g(r) \Longleftrightarrow f(r) = \sum_j (-1)^j \binom{r}{j} g(j)\ ,$$

We obtain:

$$\sum_k (-1)^k \binom{n}{k} \binom{r+k}{r} \binom{m+n-r-k}{m-r} = \sum_j (-1)^j \binom{r}{j} \binom{n+j}{n} \binom{m-j}{m-n}\ ,$$

It is easy to check that both members of the identity satisfy the same recurrence and that enought initial conditions are identical:

```
> E := Sum( (-1)^k*binomial(n,k)*binomial(r+k,k)*binomial(m+n-r-k,m-r), k );

        -----
         \        k
 E :=     )   (-1)  binomial(n, k) binomial(r + k, k) binomial(m + n - r - k, m - r)
         /
        -----
          k

> SumToRec( E, n, L );

  - (m + n + 2) (- m + n) L(n) + (2 n + 3) (2 r - m) L(n + 1) + (n + 2)  L(n + 2) = 0

> F := Sum( (-1)^j*binomial(r,j)*binomial(n+j,j)*binomial(m-j,m-n), j );

        -----
         \       j
   F :=   )   (-1)  binomial(r, j) binomial(n + j, j) binomial(m - j, m - n)
         /
        -----
          j

> SumToRec( F, n, R );

  - (m + n + 2) (- m + n) R(n) + (2 n + 3) (2 r - m) R(n + 1) + (n + 2)  R(n + 2) = 0
```

○ $${}_2F_1\left[\begin{matrix}a, 1-a\\ d\end{matrix}; \frac{1}{2}\right] = 2^{1-d}\sqrt{\pi}\,\frac{\Gamma(d)}{\Gamma(\frac{a+d}{2})\,\Gamma(\frac{1-a+d}{2})}\ ,$$

```
> left := HYP[[a,1-a,c],[d,1+2*c-d],1]:
> Lim( left, c=infinity );

                         HYP[[a, 1 - a], [d], 1/2]

> right := HypSum( HYP[[a,1-a,c],[d,1+2*c-d],1], 34 );

                      (1 - 2 c)                              /
          right := Pi 2          GAMMA(d) GAMMA(1 + 2 c - d)  /  (
                                                             /

              GAMMA(1/2 a + 1/2 + c - 1/2 d) GAMMA(1/2 a + 1/2 d)

              GAMMA(1 - 1/2 a + c - 1/2 d) GAMMA(1/2 - 1/2 a + 1/2 d))

> Lim( right, c=infinity );

                          (1 - d)   1/2
                         2        Pi    GAMMA(d)
                ---------------------------------------------
                GAMMA(1/2 a + 1/2 d) GAMMA(1/2 - 1/2 a + 1/2 d)
```

∘ $$\sum_k \binom{m}{k}\binom{n}{r-k} = \frac{(-1)^r\,\Gamma(-n-m+r)}{\Gamma(r+1)\,\Gamma(-n-m)}\ ,$$

```
> mysum := Sum( binomial(m,k)*binomial(n,r-k), k );

                         -----
                          \
                mysum :=   )   binomial(m, k) binomial(n, r - k)
                          /
                         -----
                           k

> myrec := SumToRec( mysum, r, S );

               myrec := (- n - m + r) S(r) + (r + 1) S(r + 1) = 0

> mysol := HypSolRec( myrec, S(r) );     # hypergeometric solutions of recurrence

                                      - n - m + r
                          mysol := {- -----------}
                                         r + 1

> myres := product(op(1,mysol),r=0..r-1);

                                          r
                                      (-1)  GAMMA(- n - m + r)
                      myres := ---------------------------
                                   GAMMA(r + 1) GAMMA(- n - m)
```

∘ $$\sum_{j=0}^{n}\binom{n}{j}^3 = \sum_{k=0}^{n}\binom{n}{k}^2\binom{2\,(n-k)}{n}\ .$$

```
> left := Sum( binomial(n,j)^3, j );

                                 -----
                                  \                  3
                         left :=   )   binomial(n, j)
                                  /
                                 -----
                                   j

> SumToRec( left, n, A );

              2              2                               2
   - 8 (n + 1)  A(n) + (- 7 n  - 21 n - 16) A(n + 1) + (n + 2)  A(n + 2) = 0

> right := Sum( binomial(n,k)^2*binomial(2*(n-k),n), k );

                       -----
                        \                  2
              right :=   )   binomial(n, k)  binomial(2 n - 2 k, n)
                        /
                       -----
                         k

> SumToRec( right, n, B );
              2              2                               2
   - 8 (n + 1)  B(n) + (- 7 n  - 21 n - 16) B(n + 1) + (n + 2)  B(n + 2) = 0
```

Moreover, initial conditions are $A(0) = B(0) = 1$ and $A(1) = B(1) = 2$.

- $$\sum_{k=0}^{n} \binom{n}{k} 5^{(k-1)/2} = \left(\sqrt{5}+1\right)^n ,$$

```
> rec := SumToRec( Sum(binomial(n,k)*5^((k-1)/2),k), n, A );

                                    1/2
                     rec := (- 5    - 1) A(n) + A(n + 1) = 0

> # hypergeometric solutions of recurrence
> sol := HypSolRec( rec, A(n) );

                                         1/2
                               sol := {5    + 1}

> product(op(sol),k=0..n-1);

                                   1/2     n
                                 (5    + 1)
```

- $$\sum_{k=0}^{n} k \binom{n}{k}^2 = (2\,n-1)\binom{2n-2}{n-1} ,$$

```
> mysum := Sum( (-1)^k * binomial(n,k)^2 / binomial(2*n,k) * binomial(2*n,n), k );

                     -----      k              2
                      \     (-1)  binomial(n, k)  binomial(2 n, n)
            mysum :=   )    ------------------------------------
                      /              binomial(2 n, k)
                     -----
                       k

> myrec := SumToRec( mysum, n, S );

                myrec := (n + 1) S(n) + (- n - 1) S(n + 1) = 0

> PolySolRec( myrec, S(n) );

                                       _x0

> S(0) := eval( subs( n=0, k=0, (-1)^k*binomial(n,k)^2/binomial(2*n,k)*binomial(2*n,n) ) );

                                 S(0) := 1
```

So,

$$\begin{cases} S(n) = \texttt{constant} , \\ S(0) = 1 , \\ S(n) = 1 . \end{cases}$$

∘ $$\sum_{i=0}^{k}(-1)^i \binom{k}{i}\binom{n+i}{k} = (-1)^k ,$$

```
> SumToRec( Sum((-1)^i * binomial(k,i) * binomial(n+i,k),i), k, S );

                                               (n + i - k) i
             [(k + 1) S(k) + (k + 1) S(k + 1) = 0, -------------]
                                                - k - 1 + i

> S(0) := eval( subs( k=0, i=0, (-1)^i*binomial(k,i)*binomial(n+i,k) ) );

                                S(0) := 1
```

So,

$$\begin{cases} (k+1)\,S(k) + (k+1)\,S(k+1) = 0 , \\ S(k) + S(k+1) = 0 , \\ S(k) = -S(k-1) = S(k-2) = \cdots = (-1)^k\, S(0) = (-1)^k . \end{cases}$$

∘ $$\sum_{i=0}^{k}\binom{n}{i}\binom{m}{k-i} = \binom{m+n}{k} ,$$ *Vandermonde's Identity,*

```
> SumToRec( Sum(binomial(n,i) * binomial(m,k-i) / binomial(m+n,k),i), k, S );

                                       (m - k + i) i
              [S(k) - S(k + 1) = 0, - -------------------------]
                                      (- k - 1 + i) (m + n - k)

> S(0) := eval( subs( k=0, i=0, binomial(n,i)*binomial(m,k-i)/binomial(m+n,k) ) );

                                S(0) := 1
```

$$\begin{cases} S(k) - S(k+1) = 0 , \\ S(k) = S(k-1) = \cdots = S(0) = 1 . \end{cases}$$

∘ $$\sum_{0\leq k\leq n/2} (-1)^k \binom{n-k}{k} 2^{n-2\,k} = n+1\ ,$$

```
> rec := SumToRec( Sum((-1)^k * binomial(n-k,k) * 2^(n-2*k),k), n, S );

                 rec := (n + 2) S(n) + (- n - 1) S(n + 1) = 0

> PolySolRec( rec, S(n) );

                                 (n + 1) _x0

> S(0) := (-1)^0 * binomial(0-0,0) * 2^(0-2*0);

                                  S(0) := 1
```

$$\begin{cases} S(n) = (n+1)\times \texttt{constant}\ , & S(0) = 1\ , \\ S(n) = n+1\ . \end{cases}$$

∘ $$\sum_{k=0}^{n} \binom{n}{k}^2 x^k = \sum_{l=0}^{n} \binom{n}{l} \binom{2\,n-l}{n} (x-1)^l\ ,$$

```
> SumToRec( Sum(binomial(n,k)^2*x^k,k), n, S, 'cert1' );

          2
   (x - 1)  (n + 1) S(n) - (1 + x) (2 n + 3) S(n + 1) + (n + 2) S(n + 2) = 0

> cert1;

                2      2                                                2
  (n + 1) (- 4 n  + x n  + 4 n k - 2 k x n - 13 n + 4 x n - 4 k x - 10 + k  x

          2              2   /           2          2
       - k  + 4 x + 6 k) k   /  ((n + 2 - k)  (n + 1 - k) )
                            /

> SumToRec( Sum(binomial(n,l) * binomial(2*n-l,n) * (x-1)^l,l), n, T, 'cert2' );

          2
   (x - 1)  (n + 1) T(n) - (1 + x) (2 n + 3) T(n + 1) + (n + 2) T(n + 2) = 0

> cert2;

       3        2      3       2                                   2         2
 - (x n  + 5 x n  - 9 n  - 39 n  - 54 n + 8 x n + 4 x - 24 - 2 x n  l + 12 n  l

                                                  2        2      2      2
     - 6 x n l + 33 n l - 4 l x + 22 l - 4 n l  + x n l  - 5 l  + x l ) l

                     /               2               2
    (- 2 n - 1 + l)  /  ((- n - 2 + l)  (- n - 1 + l) )
                    /
```

These two recurrences are the same, and $S(0) = T(0) = 1$, $S(1) = T(1) = 1 + x$. It is enough to prove the identity (even the two certificates are differents).

- $\displaystyle\sum_{j=0}^{k} \binom{k}{j}^2 \binom{n+2\,k-j}{2\,k} = \binom{n+k}{k}^2$, *Li-Jen-Shu,*

```
> mysum := Sum( binomial(k,j)^2 * binomial(n+2*k-j,2*k) / binomial(n+k,k)^2, j );

                  -----              2
                   \    binomial(k, j)  binomial(n + 2 k - j, 2 k)
         mysum :=   )   ----------------------------------------
                   /                                 2
                  -----             binomial(n + k, k)
                    j

> SumToRec( mysum, k, S );

                              S(k) - S(k + 1) = 0

> S(0) := eval( subs( k=0, j=0, binomial(k,j)^2*binomial(n+2*k-j,2*k)/binomial(n+k,k)^2 ) );

                                   S(0) := 1
```

- $\displaystyle\sum_{s=0}^{2\,m}(-1)^s \binom{2\,m}{s}^3 = (-1)^m \frac{(3\,m)!}{(m!)^3}$,

```
> SumToRec( Sum((-1)^s*binomial(2*m,s)^3/(-1)^m/(3*m)!*(m!)^3,s), m, M, 'cert' );

                              M(m) - M(m + 1) = 0

> cert;

               5         4         3         2                        4
  - 1/6 (448 m  + 1760 m  + 2728 m  + 2084 m  + 784 m + 116 - 624 s m

                 3         2                            2  3        2  2
       - 1932 s m  - 2214 m  s - 1113 m s - 207 s + 348 s  m  + 792 s  m

                2        2       3  2        3         3      4        4   3
       + 594 m s  + 147 s  - 90 s  m  - 132 s  m - 48 s  + 9 s  m + 6 s ) s

         /                                     3              3
        /  ((3 m + 2) (3 m + 1) (2 m + 1 - s)  (2 m + 2 - s) )
       /
```

Remark: note that it is the specialization $a = b = c = m$ of the following equality:

$$\sum_s (-1)^s \binom{b+c}{b+s} \binom{c+a}{c+s} \binom{a+b}{a+s} = \frac{(a+b+c)!}{a!\,b!\,c!} \ .$$

- $$\sum_{j=0}^{d} \binom{d}{j} \binom{d-r-j-1}{r-j} = \frac{2^r}{r!} \prod_{i=0}^{r-1} d-1-2\,i \; ,$$

```
> rec := SumToRec( Sum(binomial(d,j)*binomial(d-r-j-1,r-j),j), r, R );

             rec := (- 2 d + 4 r + 2) R(r) + (r + 1) R(r + 1) = 0

> sol := HypSolRec( rec, R(r) );

                                   - 1/2 d + r + 1/2
                       sol := {- 4 -----------------}
                                         r + 1

> normal( Product(op(sol),r=0..r-1) );

                          r - 1
                         --------'
                        '  |  |        - d + 2 r + 1
                           |  |    - 2 -------------
                           |  |            r + 1
                           |  |
                          r = 0
```

- $$(2^n+1)\,(2^n-1)/3 = \sum_k \binom{2\,n+1}{n-1-3\,k} \; ,$$

```
> rec := SumToRec(Sum(binomial(2*n+1,n-1-3*k),k),n,R);

                 rec := 4 R(n) - 5 R(n + 1) + R(n + 2) = 0

> left := (2^n+1)*(2^n-1)/3:
> CheckRec(rec,R(n),left);

                                    true
```

- $${}_4F_3\left[\begin{matrix} \frac{3}{2}+\frac{n}{5}, \frac{2}{3}, -n, 2+2\,n \\ \frac{11}{6}+n, \frac{4}{3}, \frac{1}{2}+\frac{n}{5} \end{matrix}; \frac{2}{27}\right] = \frac{(5/2)_n\,(11/6)_n}{(3/2)_n\,(7/2)_n} \; ,$$

```
> H := HYP[[3/2+n/5,2/3,-n,2+2*n],[11/6+n,4/3,1/2+n/5],2/27]:
> HypToRec( H, n, L );

        (2 n + 5) (6 n + 11) L(n) - 3 (7 + 2 n) (2 n + 3) L(n + 1) = 0

> HypergToRec( RF[5/2,n]*RF[11/6,n]/RF[3/2,n]/RF[7/2,n], R(n) );

        (2 n + 5) (6 n + 11) R(n) - 3 (7 + 2 n) (2 n + 3) R(n + 1) = 0

> HypEval( subs(n=0, HYP[[3/2+n/5,2/3,-n,2+2*n],[11/6+n,4/3,1/2+n/5],2/27] ) );

                                     1

> RfEval( subs(n=0, RF[5/2,n]*RF[11/6,n]/RF[3/2,n]/RF[7/2,n] ) );

                                     1
```

◦ Numerical evaluation of several hypergeometric series:

$${}_2F_1\left[\begin{matrix}0.1, 0.2\\ 0.4\end{matrix}; 1\right]$$

```
> hyp := HYP[[0.1,0.2],[0.4],1]:
> HypSum( hyp, 3 );  # Gauss's theorem

                                  1.536533111
```

$${}_2F_1\left[\begin{matrix}-1, -0.5\\ 3\end{matrix}; 1\right]$$

```
> hyp  := HYP[[-1,-1/2],[3],1]:
> hyps := HypSum(hyp,5);

                                        RF[5/2, 2]
                               hyps := 4 ----------
                                         RF[5, 2]

> RfEval( hyps );

                                      7/6
```

We can also compute the first terms:

```
> f := FirstTerms( HYP[[-1,-1/2],[3],1], 5 );

                    RF[-1, 0] RF[-1/2, 0]   RF[-1, 1] RF[-1/2, 1]
               f := --------------------- + ---------------------
                          RF[3, 0]                RF[3, 1]

                      /infinity                      \
                      | -----                        |
                      |  \      RF[-1, k] RF[-1/2, k]|
                    + |   )     ---------------------|
                      |  /           RF[3, k] k!     |
                      | -----                        |
                      \ k = 5                        /

> RfEval( f );

                  /           /  k - 1            \                 \
                  |           | --------'         |                 |
                  |           |'  |  |            |                 |
                  |           |   |  |    (- 1 + i)| GAMMA(- 1/2 + k)|
                  |infinity   |   |  |            |                 |
                  | -----     |   |  |            |                 |
                  |  \        \  i = 0            /                 |
          7/6 +   |   )     - --------------------------------------|
                  |  /                 1/2                          |
                  | -----            Pi    GAMMA(3 + k) k!          |
                  \ k = 5                                           /
# the sum is zero because of the product.
```

23.2 Catalan numbers

Some interesting identities about Catalan numbers: see [28]

- Let C_k the k^{th} *Catalan* number, $\frac{1}{k+1}\binom{2\,k}{k}$,

```
> C:=proc(k)
> binomial(2*k,k)/(k+1);
> end;

C := proc(k) binomial(2*k,k)/(k+1) end
```

- $\sum_k \binom{n-1}{2\,k} C_k\, 2^{n-2\,k-1} = C_n$,

```
> SumToRec( Sum(binomial(n-1,2*k)*C(k)*2^(n-2*k-1),k), n, S );

                    (2 + 4 n) S(n) + (- n - 2) S(n + 1) = 0

> HypergToRec( C(n), T(n) );

                     (2 + 4 n) T(n) - (n + 2) T(n + 1) = 0
```

- $\sum_k \binom{n-1}{2\,k} \binom{n-2\,k-1}{n-k-i} C_k = \frac{\binom{n}{i}\binom{n}{i-1}}{n}$,

```
> f := Sum( binomial(n-1,2*k)*binomial(n-2*k-1,n-k-i)*C(k), k );

       -----
        \    binomial(n - 1, 2 k) binomial(n - 2 k - 1, n - k - i) binomial(2 k, k)
f :=     )   ----------------------------------------------------------------------
        /                                   k + 1
       -----
         k

> SumToRec( f, n, S );

            - n (n + 1) S(n) + (i - n - 1) (i - 2 - n) S(n + 1) = 0

> HypergToRec( binomial(n,i)*binomial(n,i-1)/n, T(n) );

             n (n + 1) T(n) - (i - n - 1) (i - 2 - n) T(n + 1) = 0
```

- $\displaystyle\sum_k \binom{n-1}{2\,k} C_k \left[\binom{n-2\,k-1}{i-k-1} - \binom{n-2\,k-1}{i-k}\right] = \frac{\binom{n+1}{i}\binom{n-1}{i-1}}{\binom{i+1}{2}}\,(i-\frac{n}{2})$,

```
> e1 := binomial(n-2*k-1,i-k-1)-binomial(n-2*k-1,i-k):
> e2 := binomial(n+1,i)*binomial(n-1,i-1)/binomial(i+1,2)*(i-n/2):
> g := Sum( binomial(n-1,2*k)*C(k)*e1/e2, k );

              -----
               \
        g :=    )   binomial(n - 1, 2 k) binomial(2 k, k)
               /
              -----
                k

             (binomial(n - 2 k - 1, i - k - 1) - binomial(n - 2 k - 1, i - k))

             binomial(i + 1, 2)/

             ((k + 1) binomial(n + 1, i) binomial(n - 1, i - 1) (i - 1/2 n))

> SumToRec( g, n, S );

                            - S(n) + S(n + 1) = 0
```

- $\displaystyle\sum_k \binom{2\,m}{2\,k} C_k\, C_{m-k} = C_m\, C_{m+1}$,

```
> h := Sum( binomial(2*m,2*k)*C(k)*C(m-k), k );

             -----
              \    binomial(2 m, 2 k) binomial(2 k, k) binomial(2 m - 2 k, m - k)
       h :=    )   --------------------------------------------------------------
              /                          (k + 1) (m - k + 1)
             -----
               k

> SumToRec( h, m, S );

              - 4 (2 m + 3) (2 m + 1) S(m) + (m + 3) (m + 2) S(m + 1) = 0

> HypergToRec( C(m)*C(m+1), REC(m) );

             4 (2 m + 3) (2 m + 1) REC(m) - (m + 3) (m + 2) REC(m + 1) = 0
```

- $\sum_k \binom{2\,m-1}{2\,k} C_k\, C_{m-k} = \frac{1}{2} C_m\, C_{m+1}$,

```
> p := Sum( binomial(2*m-1,2*k)*C(k)*C(m-k), k );

          -----
           \     binomial(2 m - 1, 2 k) binomial(2 k, k) binomial(2 m - 2 k, m - k)
    p :=    )    ----------------------------------------------------------------
           /                            (k + 1) (m - k + 1)
          -----
            k

> SumToRec( p, m, S );

  - 4 (2 m + 3) (2 m + 1) (m + 1) S(m) + (m + 3) (m + 2) (m + 1) S(m + 1) = 0
```

23.3 Fibonacci numbers

The well-known *Fibonacci* numbers are defined by $F_0 = 0, F_1 = 1$, and $F_{n+2} = F_{n+1} + F_n$:

```
> sols := HypSolRec( F(n+2)=F(n+1)+F(n), F(n) );

                                     1/2                1/2
                 sols := {1/2 + 1/2 5    , 1/2 - 1/2 5    }

> s1 := product(op(1,sols),n=0..n-1);

                                        1/2 n
                         s1 := (1/2 + 1/2 5    )

> s2 := product(op(2,sols),n=0..n-1);

                                        1/2 n
                         s2 := (1/2 - 1/2 5    )
```

The general solution is a linear combinaison of s_1 and s_2: $F_n = \lambda\, s_1(n) + \mu\, s_2(n)$.
With initial conditions, we find:

$$\begin{cases} n = 0\ , & \lambda + \mu = 0\ , \\ n = 1\ , & \lambda\,\frac{1+\sqrt{5}}{2} + \mu\,\frac{1-\sqrt{5}}{2} = 1\ , \end{cases}$$

$$\text{So,}\qquad F_n = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n \right)\ .$$

We can also easily verify that:

$$\sum_{k=0}^{n} \binom{n-k}{k} = F_n\ ,$$

```
> SumToRec( Sum(binomial(n-k,k),k), n, F );

                    - F(n) - F(n + 1) + F(n + 2) = 0
```

Part V

q-analog

Chapter 24

Basic hypergeometric series

`QHYP, QRF & QBIN: IsQHYP, IsQRF, IsQBIN, QHypEval, QRfEval, QBinEval`
`AddParam, HypPermBoth, HypPermLow, HypPermUp`
`WHYP: QHypType, QHypOrder, IsWHYP, QHypToWHyp, WHypToQHyp`
`MapList, MapApply`
`PolySolQRec, RatioSolQRec, HypSolQRec`

24.1 Basics

The basic objects are:

- the q-rising factorial `QRF[a,q,k]` (q-analog of the Pochhammer symbol):

$$(a;q)_k := \begin{cases} (1-a)\,(1-a\,q)\ldots(1-a\,q^{k-1}) & k=1,2,\ldots, \\ 1 & k=0, \\ [(1-a\,q^{-1})\,(1-a\,q^{-2})\ldots(1-a\,q^{k})]^{-1} & k=-1,-2,\ldots, \qquad a\neq q,q^2,\ldots,q^{-k}; \end{cases}$$

  and the compact Gasper and Rahman notation:

$$(a_1,a_2,\cdots,a_r;q)_k := (a_1;q)_k\,(a_2;q)_k\cdots(a_r;q)_k\ ,$$

  We have $(a;q)_k = \dfrac{(a;q)_\infty}{(a\,q^k;q)_\infty}$;

- the q-binomial coefficient `QBIN[n,m,q]`:

$$\begin{bmatrix} n \\ m \end{bmatrix}_q := \begin{cases} (q;q)_n\,{(q;q)_m}^{-1}\,{(q;q)_{n-m}}^{-1} & \text{if } 0\leq m\leq n\ , \\ 0 & \text{otherwise.} \end{cases}$$

  The polynomials $\begin{bmatrix} n \\ m \end{bmatrix}_q$ were first studied by Gauss and have come to be known as Gaussian polynomials (see [4]). The polynomial $\begin{bmatrix} n \\ m \end{bmatrix}_q$ is a polynomial of degree $m\,(n-m)$ in q that satisfies the following relations:

$$\begin{aligned} \begin{bmatrix} n \\ 0 \end{bmatrix}_q &= \begin{bmatrix} n \\ n \end{bmatrix}_q = 1 \\ \begin{bmatrix} n \\ m \end{bmatrix}_q &= \begin{bmatrix} n \\ n-m \end{bmatrix}_q \\ \begin{bmatrix} n \\ m \end{bmatrix}_q &= \begin{bmatrix} n-1 \\ m-1 \end{bmatrix}_q + q^m \begin{bmatrix} n-1 \\ m \end{bmatrix}_q \end{aligned}$$

- and also the basic hypergeometric series QHYP[[a_1,...,a_r],[b_1,...,b_s],q,z]:

$$ {}_r\phi_s \left[\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} ; q, z \right] , $$

(see notation 2 page 158).

Definition 9 *A series $\sum_{k\geq 0} t_k$ is q-**hypergeometric** if the ratio of two consecutive terms is a rational function of q^k:*

$$ \frac{t_{k+1}}{t_k} = \frac{P(q^k)}{Q(q^k)} , \tag{24.1} $$

where P and Q are polynomials in q^k. (Note that q should be contained in the coefficient field which should be of characteristic 0.)

Obviously, a q-hypergeometric term t_k satisfy a first-order linear homogeneous recurrence relation:

$$ Q(q^k)\, t_{k+1} - P(q^k)\, t_k = 0 . $$

Notation 2 *The standard form of a basic hypergeometric series is*

$$ {}_r\phi_s \left[\begin{matrix} a_1, \ldots, a_r \\ b_1, \ldots, b_s \end{matrix} ; q, z \right] := \sum_{k=0}^{\infty} \frac{(a_1, \ldots, a_r; q)_k}{(q, b_1, \ldots, b_s; q)_k} z^k \left[(-1)^k q^{\binom{k}{2}} \right]^{1+s-r} , \tag{24.2} $$

where z is called "evaluation point", q is called the "base" and r, s are respectively the number of "upper" and "lower" parameters of the series. In order to have a well-defined series, we require that $\forall i = 1, \ldots, s, \quad b_i \neq 1, q^{-1}, q^{-2}, \ldots$.

The basic functions are IsQHYP, IsQRF and QHypEval, QRfEval. The last two functions are rules that transform a QHYP (respectively a QRF) into a sum (respectively q-factorial expression).

```
> IsQRF( QRF[a,q,b] );

                                  true

> IsQHYP( QHYP[[a,b,c],[1,e],q,z] );

           warning QHYP: denominator values must be different of 1

                                  true
```

The first lower parameter of this basic hypergeometric series is equal to 1, so its evaluation is impossible.

```
> IsQHYP( QHYP[[a,b],[c],q,z] );

                                  true

> QRfEval( QRF[a,q,3] );

                                                 2
                       (1 - a) (1 - a q) (1 - a q )

> QHypEval( QHYP[[a],[c],q,z] );
```

```
infinity
 -----                 k      k  (1/2 k (k - 1))
  \      QRF[a, q, k] z  (-1)  q
   )     --------------------------------------
  /             QRF[c, q, k] QRF[q, q, k]
 -----
 k = 0
```

In the following special case, the factor $\left[(-1)^k\, q^{\binom{k}{2}}\right]^{1+s-r}$ has disappeared because $1+s-r=0$.

```
> QHypEval( QHYP[[a,b],[c],q,z] );
```

```
infinity
 -----                           k
  \      QRF[a, q, k] QRF[b, q, k] z
   )     ----------------------------
  /        QRF[c, q, k] QRF[q, q, k]
 -----
 k = 0
```

To manipulate q-series, `AddParam` allows to add a parameter to upper and lower parameters:

```
> AddParam( QHYP[[b,c],[d],q,z], 3 );
```

```
QHYP[[3, b, b], [3, d], q, z]
```

The functions `HypPermLow`, `HypPermUp` and `HypPermBoth` allow to permute parameters in hypergeometric series, the usage is `HypPermXX(shyp,p)` with:

- `shyp`, a basic hypergeometric series (in standard notation);
- `p`, a list of positive numbers forming a permutation.

The effect is that the new (upper or lower) parameter at position `i` is the old parameter from position `p[i]`.

```
> HypPermLow( QHYP[[a, b, c, d], [e, f, g], q, z], [2,3,1] );
```

```
QHYP[[1, b, b, d], [f, g, e], q, z]
```

24.2 Others q-hypergeometric series

Definition 10 *The basic hypergeometric series*

$$ {}_{r+1}\phi_r \left[\begin{matrix} a_1, a_2, \ldots, a_{r+1} \\ b_1, \ldots, b_r \end{matrix} ; q, z \right] \tag{24.3} $$

*is called **well-poised** if its parameters satisfy the relation*

$$ q\, a_1 = a_2\, b_1 = a_3\, b_2 = \ldots = a_{r+1}\, b_r \ . $$

Definition 11 *The q-series (24.3) is called **nearly-poised** if all but one of the above pairs of parameters (regarding q as the first lower parameter) have the same product. We distinguish two cases:*

- *a series is called a **nearly-poised series of the first kind** if*

$$ q\, a_1 \neq a_2\, b_1 = a_3\, b_2 = \ldots = a_{r+1}\, b_r \ , $$

- *and it is called a* ***nearly-poised series of the second kind*** *if*

$$q\, a_1 = a_2\, b_1 = a_3\, b_2 = \ldots \neq a_{r+1}\, b_r \ .$$

Definition 12 *If the q-series (24.3) is well-poised and if, in addition,* $a_2 = q\, a_1^{\frac{1}{2}}$, $a_3 = -q\, a_1^{\frac{1}{2}}$, *then it is called a* ***very-well-poised*** *series.*

The object `WHYP[a,[list],z]` represents a q-HYPergeometric series in Very-well-poised order: precisely, `WHYP[a,[b,c],q,z]` corresponds to:
`QHYP[[a, q*a^(1/2), -q*a^(1/2), b, c],[a^(1/2), -a^(1/2), q*a/b, q*a/c],q,z]`.
You can imagine what are the features of the following functions: `IsWHYP`, `WHypToQHyp`, `QHypToWHyp`, `QHypOrder`, `QHypType` and `QHypEval`.

```
> IsWHYP( WHYP[a,[b,c,q*a,e],q,z] );

            warning WHYP: denominator values must be different of 1

                                     true
```

THE IMPORTANT POINT IS THAT ALGORITHMS STUDIED IN PREVIOUS CHAPTERS EXTEND TO THE q-CASE.

24.3 Solutions of q-recurrences

The `PolySolQRec`, `RatioSolQRec` and `HypSolQRec` functions are used to find all polynomial, rational and q-hypergeometric solutions of linear q-recurrence equations with *polynomial coefficients*:

$$a_k(n)\, U(q^k\, n) + \cdots + a_1(n)\, U(q\, n) + a_0(n)\, U(n) = f(n) \qquad \text{with } a_i(n), f(n) \in \mathbb{K}[n] \qquad (24.4)$$

If there is no solution, the return value is "`no solution`".

```
> PolySolQRec( U(q^2*n)-2*U(q*n)+U(n)=(q^6-2*q^3+1)*n^3+(q^5-2*q^3+q)*q*n^2, U(n) );

                                        2  2    3
                                 _x0 + q  n  + n

> RatioSolQRec( q^3*(q*n+1)*y(q^2*n)-2*q^2*(n+1)*y(q*n)+(n+q)*y(n)
                               =(q^6-2*q^3+1)*n^2 + (q^5-2*q^3+q)*n, y(n) );

                                           2
                                    q n + n
                                    --------
                                      n + q

> HypSolQRec( (q^2*n-1)*y(q^2*n)+(q^2*(1+q)*n^2-q*n)*y(q*n)+q^2*n^3*y(n)=0, y(n) );

                                           2
                                        n q
                                 {- n, - -------}
                                         q n - 1
```

Chapter 25

Routines

25.1 Basic hypergeometric series

```
FUNCTION: QHYP - how generalized QHYPergeometric series are coded

SYNOPSIS:
- QHYP is an indexed name: HYP[ l1, l2, q, z ] with

     - l1, the list of upper parameters,
     - l2, the list of lower parameters,
     - q, the base,
     - z, the evaluation point.

EXAMPLES:
> with(HYPERG):
> qhyp := QHYP[[a,b,c],[d,e],q,z]:
> IsQHYP( qhyp );

                                      true

SEE ALSO: HYPERG,  IsQHYP,  WHYP, IsWHYP,  QRF, QBIN, QHypEval, QHypOrder,
          HypPerm, QHypToWHyp, AddParam
```

```
FUNCTION: IsQHYP - test if is a basic hypergeometric series

CALLING SEQUENCE:
  IsQHYP(qhyp)
  HYPERG[IsQHYP](qhyp)

PARAMETERS:
  qhyp - one term

SYNOPSIS:
- The  IsQHYP  function  tests  whether a term is a  basic  hypergeometric
  series.

- If a lower parameter is egal to 1,  then  the function gives a  warning.
```

```
- Whenever there  is  a  conflict  between  the function name  IsQHYP  and
  another name  used in the same session,  use the form  HYPERG['IsQHYP'].

EXAMPLES:
> with(HYPERG):
> IsQHYP( QHYP[[a,b,c],[d,e],q,z] );

                                     true

> IsQHYP( QHYP[[a,q,c],[d,1],q,z] );

          warning QHYP: denominator values must be different of 1

                                     true

SEE ALSO: HYPERG,    QHYP,       QRF,  QBIN,  WHYP,  QHypEval,  QHypOrder,
          HypPerm,   QHypToVHyp, AddParam,    IsQRF, IsQBIN,    IsWHYP
```

```
FUNCTION: QHypEval - Rule that transforms a QHYP[] into a Sum()

CALLING SEQUENCE:
  QHypEval( expr )
  HYPERG[QHypEval]( expr )

  QHypEval( expr, tsi )`,
  HYPERG[QHypEval]( expr, tsi )`,

PARAMETERS:
  expr - any expression
  tsi  - a name, the summation index

SYNOPSIS:
- The  QHypEval  function expands  basic hypergeometric series  into sums.
  It takes in  input  any expression  involving  QHYP (generalized basic
  HYPergeometric  series)  and  WHYP (basic HYPergeometric series  in
  Very-Well-poised order).

- The optional second argument (tsi)  allows you to choose the name of the
  summation index. The default value is k.`,

- Whenever  there is a conflict  between the  function  name  QHypEval and
  another name used in the same session, use the form  HYPERG['QHypEval'].

EXAMPLES:
> with(HYPERG):
> QHypEval( QHYP[[a,b],[e,f,g],q,z] );

       infinity
        -----                            k      k 2   (1/2 k (k - 1)) 2
         \      QRF[a, q, k] QRF[b, q, k] z  ((-1) )  (q                )
          )     ---------------------------------------------------------
         /         QRF[e, q, k] QRF[f, q, k] QRF[g, q, k] QRF[q, q, k]
```

```
        -----
        k = 0

SEE ALSO: HYPERG,  QHYP,  IsQHYP,   WHYP,  IsWHyp,  AddParam, HypSimplify,
          QHypOrder,      HypPerm,  QHypType‘
```

25.2 q-rising factorial

```
FUNCTION: QRF - how Q-Rising Factorials are coded

SYNOPSIS:
- QRF is an indexed name:   QRF[a, q, k]   where  a, q  and  k  are  names
  (or numerical values).

- The mathematical notation is (a;q)_k.

EXAMPLES:
> with(HYPERG):
> qrf := QRF[a,q,k]:
> QRfEval( qrf );

                             k - 1
                            --------’
                           ’  |  |            i
                              |  |    (1 - a q )
                              |  |
                              |  |
                             i = 0

SEE ALSO: HYPERG, IsQRF, QRfEval, QBIN, QHYP, WHYP
```

```
FUNCTION: IsQRF - test if is a q-rising factorial

CALLING SEQUENCE:
  IsQRF(qrf)
  HYPERG[IsQRF](qrf)

PARAMETERS:
  qrf - one term

SYNOPSIS:
- The IsQRF function tests whether a term is a q-rising factorial.

- Whenever there is  a  conflict  between  the  function  name  IsQRF  and
  another name used in the same session, use the long form HYPERG['IsQRF'].

EXAMPLES:
> with(HYPERG):
> IsQRF( QRF[a] );
```

```
                                   false

> IsQRF( QRF[a,q,5] );

                                    true

SEE ALSO: HYPERG, QRF, QRfEval, QBIN, QBinEval, IsQBin, IsQHYP, IsWHYP
```

```
FUNCTION: QRfEval - Rule that evals q-rising factorials

CALLING SEQUENCE:
  QRfEval( expr )
  HYPERG[QRfEval]( expr )

PARAMETERS:
  expr - any expression

SYNOPSIS:
- The  QRfEval  function evals (if possible) q-rising factorial (QRF[]).

- Whenever there is  a conflict  between  the  function  name  QRfEval and
  another name used in the same session, use the  form  HYPERG['QRfEval'].

EXAMPLES:
> with(HYPERG):
> QRfEval( QRF[a,q,3] );

                                                    2
                          (1 - a) (1 - a q) (1 - a q )

> QRfEval( QRF[q,q,4] );

                                  2        3        4
                       (1 - q) (1 - q ) (1 - q ) (1 - q )

> QRfEval( QRF[a,q,-3] );

                                     1
                       -------------------------------
                                /      a \ /      a \
                       (1 - a/q) |1 - ----| |1 - ----|
                                |      2 | |      3 |
                                \     q  / \     q  /

> QRfEval( QRF[q,q,-3] );
Error, (in HYPERG/QRfEval/basic) division by zero

SEE ALSO: HYPERG, QRF, IsQRf,  QBIN, IsQBin, QBinEval, QHYP, IsQHYP, WHYP,
          IsWhyp, QHypEval
```

25.3 q-binomial coefficient

```
FUNCTION: QBIN - how Q-binomial coefficients are coded

SYNOPSIS:
- QBIN is an indexed name:  QBIN[n, m, q]  where  n, m  and  q  are  names
  (or numerical values).

- q-binomial coefficients are also called Gaussian polynomials.

EXAMPLES:
> with(HYPERG):
> qbin := QBIN[n,m,q]:
> QBinEval( qbin );

                                   QRF[q, q, n]
                           -----------------------------
                           QRF[q, q, m] QRF[q, q, n - m]

SEE ALSO: HYPERG, IsQBIN, QBinEval, QRF, QHYP, WHYP
```

```
FUNCTION: IsQBIN - test if is a q-binomial coefficient

CALLING SEQUENCE:
  IsQBIN(qbin)
  HYPERG[IsQBIN](qbin)

PARAMETERS:
  qbin - one term

SYNOPSIS:
- The  IsQBIN  function tests whether a term is  a q-binomial coefficient.

- Whenever there is a  conflict  between  the  function  name  IsQBIN  and
  another name used in the same session, use  the  form  HYPERG['IsQBIN'].

EXAMPLES:
> with(HYPERG):
> IsQBIN( QBIN[a,b,c,d] );

                                     false

> IsQBIN( QBIN[a,k,q] );

                                     true

SEE ALSO: HYPERG, QBIN, QRF, QBinEval, QRfEval, IsQHYP, IsWHYP, IsQRF
```

```
FUNCTION: QBinEval - Rule that evals q-binomial coefficients
```

```
CALLING SEQUENCE:
  QBinEval( expr )
  HYPERG[QBinEval]( expr )

PARAMETERS:
  expr - any expression

SYNOPSIS:
- The QBinEval function evals (if possible)  q-binomial coefficient QBIN[].

- Whenever there is  a conflict  between  the  function  name  QBinEval and
  another name used in the same session, use the  form  HYPERG['QBinEval'].

EXAMPLES:
> with(HYPERG):
> QBinEval( QBIN[n,m,q] );

                                    QRF[q, q, n]
                               -----------------------------
                               QRF[q, q, m] QRF[q, q, n - m]

> QBinEval( QBIN[5,3,q] );

                                    QRF[q, q, 5]
                                -------------------------
                                QRF[q, q, 3] QRF[q, q, 2]

> QBinEval( QBIN[3,5,q] );

                                         0

> QBinEval( QBIN[3,3,q] );

                                         1

> QBinEval( QBIN[n,n,q] );

                                         1

> QBinEval( QBIN[0,0,q] );

                                         1

> QBinEval( QBIN[k,0,q] );

                                         1

SEE ALSO: HYPERG, QBIN, IsQBin,  QRF, IsQRF, QRfEval,  QHYP, IsQHYP, WHYP,
          IsWHYP, QHypEval
```

25.4 Very-well-poised basic hypergeometric series

x

```
FUNCTION: WHYP - how HYPergeometric series (in Very-well-poised) are coded

SYNOPSIS:
- WHYP is an indexed name: HYP[ a, l, z ] with

     - a, the first upper parameter the hypergeometric series,
     - l, a list of parameters,
     - q, the base.
     - z, the evaluation point.

- WHYP[ a, [p_1,...,p_r], q, z ] is equivalent to
  QHYP[ [a, q*a^(1/2), -q*a^(1/2), p_1, ..., p_r],
         [a^(1/2), -a^(1/2), q*a/p_1, ..., q*a/p_r],q,z]

EXAMPLES:
> with(HYPERG):
> w := WHYP[a,[n,q],q,z]:
> WHypToQHyp( w );

                   1/2       1/2           1/2     1/2  q a
        QHYP[[a, q a    , - q a    , n, q], [a    , - a    , ---, a], q, z]
                                                             n

SEE ALSO: HYPERG,     IsWHYP,   QHYP,   IsQHYP,   QRF,   QBIN,   QHypEval,
          WHyptoQHyp, QHypToWHyp
```

```
FUNCTION: IsWHYP - test if is a basic hypergeometric series
                 - in very-well-poised order

CALLING SEQUENCE:
  IsWHYP(hyp)
  HYPERG[IsWHYP](hyp)

PARAMETERS:
  hyp - one term

SYNOPSIS:
- The  IsWHYP  function  tests  whether a term  is  a basic hypergeometric
  series in very-well-poised' order.

- If a lower parameter is egal to 1,  then  the function gives a  warning.

- Whenever there  is  a  conflict  between the  function name  IsWHYP  and
  another name used  in the same session,  use the form  HYPERG['IsWHYP'].

EXAMPLES:
> with(HYPERG):
> IsWHYP( WHYP[a,[b,c,q*a,e],q,z] );

            warning WHYP: denominator values must be different of 1

                                     true
```

```
SEE ALSO: HYPERG,  WHYP,  QHYP,   QRF,   QBIN,  QHypEval,  WHypToQHyp,
          QHypToWHyp,    IsQHYP,  IsQRF,  IsQBIN
```

```
FUNCTION: QHypOrder - Order parameters of a basic hypergeometric series

CALLING SEQUENCE:
  QHypOrder( sqhyp )
  HYPERG[QHypOrder]( sqhyp )

PARAMETERS:
  sqhyp - a basic hypergeometric series in standard notation

SYNOPSIS:
- The  QHypOrder  function is used  to  order  the  parameters  of a basic
  hypergeometric  series   in   'well-poised',   'very-well-poised',   and
  'nearly-well poised order'.

- Whenever there is a conflict between the  function  name  QHypOrder  and
  another name used in the same session, use the form HYPERG['QHypOrder'].

EXAMPLES:
> with(HYPERG):
> #very-well poised order
> qhyp := QHypOrder( QHYP[[q^(-n),b,q*sqrt(a),a,-q*sqrt(a)],
>                 [a*q/b,-sqrt(a),sqrt(a),a*q^(n+1)],q,z] );

qhyp :=

            1/2       1/2      (- n)    1/2     1/2  a q     (n + 1)
QHYP[[a, q a   , - q a   , b, q     ],[a   , - a   , ---, a q       ],q,z]
                                                      b

> QHypToWHyp( qhyp );

                                       (- n)
                             WHYP[a, [b, q     ], q, z]

SEE ALSO: HYPERG, IsQHYP, IsWHYP, QHypEval, HypPerm, QHypType, AddParam
```

```
FUNCTION: QHypType - Print the type of a basic hypergeometric series

CALLING SEQUENCE:
  QHypType( sqhyp )
  HYPERG[HQypType]( sqhyp )

PARAMETERS:
  sqhyp - a hypergeometric q-series in standard notation (QHYP or WHYP)

SYNOPSIS:
- The  QHypType  function  is used to know  the type  of  a hypergeometric
  q-series ('well-poised', 'very-well-poised', 'nearly-well poised order',
```

```
  or 'ordinay').

- The  function  prints  the  type  of  the  series, and  the return value
  is always NULL.

- Whenever there is a conflict between  the  function  name  QHypType  and
  another name used in the same session, use the form  HYPERG['QHypType'].

EXAMPLES:
> with(HYPERG):
> # very-well poised order
> QHypType( QHYP[[q^(-n),b,q*sqrt(a),a,-q*sqrt(a)],
>                [a*q/b,-sqrt(a),sqrt(a),a*q^(n+1)],q,z] );

                    Very-well poised hypergeometric q-series

> # well-poised order
> QHypType( QHYP[[b,a,c,q],[a,a*q/c,a*q/b],q,z] );

                      Well-poised hypergeometric q-series

SEE ALSO: HYPERG, IsQHYP, IsWHYP, QHypEval, QHypOrder, HypPerm, AddParam
```

```
FUNCTION: QHypToWHyp
          - convert basic hypergeometric series
          - into basic hypergeometric series in very-well-poised order

CALLING SEQUENCE:
  QHypToWHyp( sqhyp )
  HYPERG[QHypToWHyp]( sqhyp )

PARAMETERS:
  sqhyp - a basic hypergeometric series in standard notation

SYNOPSIS:
- This function tries to convert  (if possible)  a q-hypergeometric series
  into a basic hypergeometric series in well-poised order.

- If the convertion is impossible, the return value of the function is the
  initial basic hypergeometric series.

- Whenever there is a conflict  between the function name  QHypToWHyp  and
  another name used in the same session, use the form HYPERG['QHypToWHyp'].

EXAMPLES:
> with(HYPERG):
> qhyp := QHYP[ [q^(-n),b,q*sqrt(a),a,-q*sqrt(a)],
       [a*q/b,-sqrt(a),sqrt(a),a*q^(n+1)], q, z]:
> QHypToWHyp( qhyp );

                                          (- n)
                           WHYP[a, [b, q     ], q, z]
```

```
SEE ALSO: HYPERG, QHYP, IsQHYP, WHYP, IsWHYP, WHypToQHyp, QHypEval
```

```
FUNCTION: WHypToQHyp
          - convert basic hypergeometric series in very-well-poised order
          - into basic hypergeometric series in standard notation

CALLING SEQUENCE:
  WHypToQHyp( whyp )
  HYPERG[WHypToQHyp]( whyp )

PARAMETERS:
  whyp - a basic hypergeometric series in very-well-poised order

SYNOPSIS:
- This function converts a  q-hypergeometric series  in 'very-well-poised'
  order  into  a  basic  hypergeometric  series   in  standard   notation.

- Whenever there is a conflict  between the function name  WHypToQHyp  and
  another name used in the same session, use the form HYPERG['WHypToQHyp'].

EXAMPLES:
> with(HYPERG):
> w := WHYP[a,[b,q^(-n)],q,z]:
> WHypToQHyp( w );

          1/2       1/2      (- n)      1/2     1/2  q a   (1 + n)
QHYP[[a, q a   , - q a   , b, q     ],[a    , - a    , ---, q        a],q,z]
                                                      b

SEE ALSO: HYPERG, WHYP, IsWHYP, QHYP, IsQHYP, QHypToWHyp, QHypEval
```

25.5 Solutions of q-recurrences

```
FUNCTION: PolySolQRec - linear q-recurrence equation solver
                      - polynomial solutions

CALLING SEQUENCE:
  PolySolQRec(eqn, u(n))
  HYPERG[PolySolQRec](eq, u(n))

PARAMETERS:
  eqn     - a linear q-recurrence equation with polynomial coefficients
  u,n     - the name and the index of the recurrence

SYNOPSIS:
- The  function  PolySolQRec  finds  all  polynomial  solutions   of   the
  q-recurrence relation.

- The  first  argument  should be  a  single  q-recurrence  equation  with
  polynomial coefficients.  This recurrence isn't necessarily homogeneous.
```

```
  Any expressions  which is not an equation will be understood to be equal
  to zero.

- The second argument indicates the sequence that PolySolQRec should solve
  for. A  sequence   is   represented  by   a   name   and  an   index.

- The output is  a polynom in  n  with constant coefficients _x0, _x1, ...

- If PolySolQRec is unable to compute a polynomial solution, it returns the
  message 'No polynomial solution'. This means that there is  no solution.

- Whenever there is a conflict between the function  name  PolySolQRec and
  another   name   used   in   the   same  session,   use  the  long  form
  HYPERG['PolySolQRec'].

EXAMPLES:
> with(HYPERG):
> PolySolQRec(   U(q^2*n)-2*U(q*n)+U(n)
              = (q^6-2*q^3+1)*n^3+(q^5-2*q^3+q)*q*n^2, U(n) );

                                      2  2    3
                                _x0 + q  n  + n

SEE ALSO: HYPERG, GenQRec, RatioSolQRec, HypSolQRec
```

```
FUNCTION: RatioSolQRec - linear q-recurrence equation solver
                       - rational solutions

CALLING SEQUENCE:
  RatioSolQRec(eqn, u(n))
  HYPERG[RatioSolQRec](eq, u(n))

PARAMETERS:
  eqn   - a linear q-recurrence equation with polynomial coefficients
  u,n   - the name and the index of the recurrence
  inits - a set of initial conditions

SYNOPSIS:
- The  function   RatioSolQRec   finds  all  rational  solutions   of  the
  q-recurrence relation.

- The  first  argument  should be  a  single  q-recurrence  equation  with
  polynomial coefficients.  This recurrence isn't necessarily homogeneous.

  Any expressions  which is not an equation will be understood to be equal
  to zero.

- The second argument indicates  the sequence   that  RatioSolQRec  should
  solve  for.   A  sequence  is  represented  by  a  name  and  an  index.

- The output is a rational function in  n  with constant coefficients _x0,
  _x1, _x2, _x3, ...
```

```
- If  RatioSolQRec  is unable to compute  a rational solution,  it returns
  the  message  'No rational solution'.  This  means  that  there  is   no
  solution.

- Whenever there is a conflict between the function name  RatioSolQRec and
  another   name   used   in   the   same  session,   use  the long  form
  HYPERG['RatioSolQRec'].

EXAMPLES:
> with(HYPERG):
> RatioSolQRec(   q^3*(q*n+1)*y(q^2*n)-2*q^2*(n+1)*y(q*n)+(n+q)*y(n)
>                = (q^6-2*q^3+1)*n^2+(q^5-2*q^3+q)*n, y(n) );

                                        2
                                 q n + n
                                 --------
                                   n + q

SEE ALSO: HYPERG, GenQRec, PolySolQRec, HypSolQRec
```

```
FUNCTION: HypSolQRec - linear q-recurrence equation solver
                     - hypergeometric solutions

CALLING SEQUENCE:
  HypSolQRec(eqn, u(n))
  HYPERG[HypSolQRec](eq, u(n))

PARAMETERS:
  eqn - a linear q-recurrence equation with polynomial coefficients
  u,n - the name and the index of the recurrence

SYNOPSIS:
- The function   HypSolQRec  finds  all  hypergeometric  solutions  of the
  q-recurrence relation.

- The  first  argument  should be  a  single  q-recurrence  equation  with
  polynomial  coefficients. Any expressions  which is not an equation will
  be understood to be equal to zero.

  This recurrence must be homogeneous.

- The second argument indicates the sequence  that HypSolQRec should solve
  for.  (A  sequence   is   represented  by   a   name   and   an  index).

- The output is a generating set (not necessarily linearly independent) of
  hypergeometric solutions  (with constant coefficients _x0, _x1, ...)  of
  the  recurrence  relation.  These  solutions  u(n)  are  given  by their
  rational representations u(n+1)/u(n).

- Whenever there is a conflict between the function  name  HypSolQRec  and
  another name used in the same session, use the form HYPERG['HypSolQRec'].
```

```
EXAMPLES:
> with(HYPERG):
> rec := x*y(q^3*x) - q^3*x^2*y(q^2*x) - (x^2+q)*y(q*x)+q*x*(x^2+q)*y(x):
> HypSolQRec( rec, y(x) );

                                   {q x}

> HypSolQRec( y(q^2*x)-(1+q)*y(q*x)+q*(1-q*x^2)*y(x)=0, y(x) );

                             1/2          1/2
                       {1 - q    x, 1 + q    x}

SEE ALSO: HYPERG, GenQRec, PolySolQRec, RatioSolQRec
```

```
FUNCTION: GenQRec - generate a q-recurrence

CALLING SEQUENCE:
  GenQRec( sols, u(n) )
  HYPERG[GenQRec]( sols, u(n) )

PARAMETERS:
  sols - a set of 'p' terms {s1, s2, ..., sp}
  u,n  - the name and the index of the recurrence

SYNOPSIS:
- The function  GenQRec   generates  a  p-order  homogeneous  q-recurrence
  satisfied by  u(n)  such  that  s1, s2, ..., and sp  are  its solutions.

- A q-recurrence is a relation between u(n) and successive q-shifts u(q*n)
  u(q^2*n), ...

- The  output is  a  q-recurrence  in  n   with  polynomial  coefficients.

- If   GenQRec   is unable  to  compute  a  recurrence, it returns  0 = 0.

- Whenever there is  a  conflict  between  the  function name  GenQRec and
  another name used in the same session, use the  form  HYPERG['GenQRec'].

EXAMPLES:
> with(HYPERG):
> GenQRec( {n*q,1}, y(n) );

                      2        2                 2
          (q - 1) y(n q ) + (- q  + 1) y(n q) + (q  - q) y(n) = 0

> GenQRec( {x^2*q,x}, y(x) );

                     2           3                    4    3
        q (q - 1) y(x q ) + q (- q  + q) y(x q) + q (q  - q ) y(x) = 0

SEE ALSO: HYPERG, QHYP, PolySolQRec, RatioSolQRec, HypSolQRec
```

Notation

– Symbols –

Index

List of functions

Bibliography

[1] S. A. Abramov, *Rational solutions of linear differential and q-difference equations with polynomial coefficients*, Proceedings ISSAC '95, ACM Press, New York, (1995), 285-289.

[2] S. A. Abramov, M. Petkovšek, *Finding all q-hypergeometric solutions of q-difference equations*, Proceedings FPSAC '95, Université de Marne-la-vallée, (1995), 1–10.

[3] S. A. Abramov, M. Bronstein, M. Petkovšek, *On polynomial solutions of linear operator equations.*

[4] G. E. Andrews, *The theory of partitions*, Encyclopedia of Mathematics and its Applications, Addison-Wesley (1976).

[5] P. Cartier, *Démonstration "automatique" d'identités et fonctions hypergéométriques [d'après D. Zeilberger]*, Séminaire Bourbaki, **746** (1991-92).

[6] S. B. Ekhad, J. E. Majewicz, *A short WZ-style proof of Abel's identity*, Electronic Journal of Combinatorics, **R16**, (1996).

[7] G. Gasper, M. Rahman, *Basic hypergeometric series*, Encyclopedia of Mathematics and its applications, Cambridge University Press, **35** (1990).

[8] G. Gasper, M. Rahman, *Errata to "Basic hypergeometric series*, Electronic version available at `http://www.math.nwu.edu/preprints/gasper/bhserrata`, (1998).

[9] R. W. Gosper, Jr, *Decision procedure for indefinite hypergeometric summation*, Proceedings of the national academy of sciences of the United States of America, **75** (1978), 40–42.

[10] R. L. Graham, D. E. Knuth, O. Patashnik, *Concrete mathematics*, A foundation for computer science, Addison-Wesley, (1989).

[11] W. Koepf, *Summation in Maple*, Maple Technical Newsletter, **3** (2), (1996), 26–32.

[12] W. Koepf, *Hypergeometric summation, an algorithmic approach to summation and special function identities*, Vieweg, Braunschweig/Wiesbaden, (1998).

[13] T. H. Koornwinder, *Identities of nonterminating series by Zeilberger's algorithm*, (1998).

[14] C. Krattenthaler, *A systematic list of two- and three-term contiguous relations for basic hypergeometric series*, unpublished manuscript, (1993).

[15] C. Krattenthaler, *`HYP`, Manual for a Mathematica package for handling hypergeometric series*, 160 pages, (1994).

[16] C. Krattenthaler, *`HYPQ`, Manual for a Mathematica package for handling basic hypergeometric series*, 193 pages, (1994).

[17] C. Krattenthaler, *HYP and HYPQ - Mathematica packages for the manipulation of binomial sums and hypergeometric series, respectively q-binomial sums and basic hypergeometric series*, Journal of symbolic computation, **20** (1995), 737–744.

[18] C. Krattenthaler, *A new proof of the M-R-R conjecture - including a generalization*, Journal of Difference Equations and Applications.

[19] Y. L. Luke, *The special functions and their approximations*, Academic Press, New York, (1969).

[20] P. Paule, M. Schorn, *A Mathematica version of Zeilberger's algorithm for proving binomial coefficient identities*, RISC-Linz Report Series, **93-36** (1993), J. Kepler University of Linz.

[21] P. Paule, *Short and easy computer proofs of the Rogers-Ramanujan identities and of identities of similar type*, The Electronic Journal of Combinatorics, **R10**, (1994).

[22] P. Paule, *A classical hypergeometric proof of an important transformation formula found by J.-B. Baillon and R. E. Bruck*, Theory and applications of nonlinear operators of accretive and monotone type, A.G. Karsatos (ed.), Lecture notes in Pure and applied mathematics, **178**, Marcel Dekker, (1996).

[23] M. Petkovšek, *Hypergeometric solutions of linear recurrences with polynomial coefficients*, Journal of symbolic computation, **14** (1992), 243–264.

[24] M. Petkovšek, H. S. Wilf, D. Zeilberger, *A = B*, A.K. Peters, Wellesley, Massachusetts, (1996). (programs are available at the WorldWideWeb site `http://www.cis.upenn.edu/~wilf/AeqB.html`).

[25] M. Petkovšek, B. Salvy, *Finding all hypergeometric solutions of linear differential equations*, Institut National de Recherche en Informatique et en Automatique, Research Report **1907** (1993).

[26] B. Salvy, P. Zimmermann, *Gfun: a maple package for the manipulation of generating and holonomic functions in one variable*, ACM Transactions on Mathematical Software, **20**, 2, (1994), 163–177.

[27] W. W. Sawyer, *Prelude to Mathematics*, Baltimore, Penguin, (1955).

[28] R. Simion, D. Ullman, *On the structure of the lattice of noncrossing partitions*, Discrete Mathematics, **98-3** (1991), 193–206.

[29] L. J. Slater, *Generalized hypergeometric functions*, Cambridge University Press, Cambridge, (1966).

[30] D. Zeilberger, *A fast algorithm for proving terminating hypergeometric identities*, Discrete Mathematics, **80** (1990), 207–211.

[31] D. Zeilberger, *Three recitations on holonomic systems and hypergeometric series*, Séminaire Lotharingien de Combinatoire, **B24a** (1990).

[32] D. Zeilberger, *A holonomic systems approach to special function identities*, Journal of computational and applied mathematics, **32** (1990), 321–368.

[33] D. Zeilberger, *The method of creative telescoping*, Journal of symbolic computation, **11** (1991), 195–204.