Computer Analysis of Hypergeometric Series: a Project of an Instruction Set Duplicating Operations of the Factorization Method

A.W.Niukkanen, O.S.Paramonova

Vernadsky Institute (GEOKHI),
RAS, Kossygin st. 19, 117975 Moscow, Russia;
e - mail: elkor@geokhi.msk.su; niukkanen@tula.net

The operator factorization method (see, e.g., [1]) greatly facilitating the study of multiple, including simple, hypergeometric series is the main object of our further interest.

The main goal of the paper is to outline a project of a universal "formula sythesizer" in the theory of hypergeometric series. The main idea of the project is to convert the basic operations of the operator factorization method into a complete set of commands serving us as a symbolic manipulation superstructure over a computer algebra system. Presently we do not try to carry out a program implementation of **this** *part of the project (***other** *parts are well underway; see Sec. 1.5 and 4). Our only intention is to present, explicitly, a complete list of* **the main operations** *inherent in the factorization method.*

The research has been supported by Russian Foundation for Basic Research, grant 01-01-00380.

1 Computer analysis and the factorization method

1.1. *Computer algebra or computer analysis?* Being related to mathematical physics by origin and to a great number of problems in a variety of sciences, by application, the hypergeometric series do have most direct relation to mathematical analysis, by the methods used for their study. Computer – aided approach to the study of hypergeometric series relates, obviously, to *computer analysis* rather than to *computer algebra.*

Analysis is the heart of mathematics and the concept of function is the heart of analysis. Functions in pure mathematics are the immaterial entities which are deprived of all properties except those endowed at will of mathematician. As to applied analysis the hypergeometric series serve as a universal substitute for what we call a function. In contrast to their "pure analogs" they show a fantastic abundance of properties endowed by their explicit structure. Therefore any algorithm efficient enough to tackle multiple hypergeometric series of arbitrarily complicated structure can be looked upon as a versatile solver capable to handle with almost any problem relating to functions of practical interest.

1.2. *The central idea of the operator method* is connected with an introduction of a new simple differential operation of *"Ω-multiplication"* $w = u * v$ over the functions $u = u(x_1, \dots, x_N)$ and $v = v(x_1, \dots, x_N)$:

$$\langle u * v | x_1, \dots, x_N\rangle = u\,(d/ds_1, \dots, d/ds_N)\; v(x_1 s_1, \dots, x_N s_N)\,|_{\forall s_n = 0} \tag{1}$$

The fundamental importance of the Ω-multiplication is that it allows any series having complicated structure to be directly expressed through simpler series thus permitting us to use the properties of the simple series to analyze any property of the initial complicated series. Let ${}^N F[A; x_1, \dots, x_N]$, ${}^N F[B; x_1, \dots, x_N]$ and ${}^N F[A, B; x_1, \dots, x_N]$ be the series of the power functions $x_1^{i_1} \dots x_N^{i_N}/(i_1! \dots i_N!)$ with coefficients $A(i_1, \dots, i_N)$, $B(i_1, \dots, i_N)$ and $A(i_1, \dots, i_N)B(i_1, \dots, i_N)$, respectively. The $i_n (n = 1, \dots, N)$ are summation indices of the series ($i_n = 0, 1, 2, \dots$).

The general factorization formula for the series ${}^N F[A, B]$ reads

$${}^F N[A, B; x_1, \dots, x_N] = \langle {}^N F[A] *^N F[B] | x_1, \dots, x_N\rangle \tag{2}$$

In short, $F[A \times B] = F[A] * F[B]$. This formula conveys the *property of* Ω *- representablility* of multiplication operation over coeffifients of an *arbitrary power series*. It shows also that the factorization method can be looked upon as a *disguised form of algebraization of the theory of hypergeometric series*.

1.3. *Conceptual basics of the method.* First, any hypergeometric series is expressed only through hypergeometric series (*closure property*). No necessity in employment of any other auxiliary representation is arisen.

Second, the functional relation $f * f_1 = f * f_2$ will be called Ω-equivalent to the relation $f_1(x_1, \dots, x_N)$=$f_2(x_1, \dots, x_N)$. The concept of Ω*-equivalence* allows *classes* of Ω-equivalent relations to be introduced. In each class a *simplest relation* which will be called a *proto-relation* can be chosen. Having proved the proto-relation we thus *prove all formulas* belonging to the class.

Third, using Ω-multiplication Ω-equivalent operators $F_1 \Leftrightarrow F_2$ can be introduced. They defined by

$$F_1\,(d/ds, s)\,\Psi(xs)\big|_{s=0} = F_2\,(d/ds, s)\,\Psi(xs)\big|_{s=0}, \tag{3}$$

where Ψ is an arbitrary function. Note that F_1 and F_2 are not necessarily identical to one another. The possibility to substitute F_1 for F_2 in an Ω-product is an inportant technical expedient of the method.

Fourth, by analogy with arithmetically identical expressions the algebraic expressions connected by finite number of arithmetic operations and Ω-multiplication operations will be called Ω*-identical expressions*. Transformation of an expression to an Ω-identical form is another important technical expedient.

1.4. *Three approaches to the use of factorization method.* Altogether we can see three approaches. *The first one is to derive all formulas manually* being dispensed completely with the necessity of working with computer. Even in this case the factorization method offers great advantages over traditional methods. *The second approach implies the manual derivation of basis sets of formulas* with subsequent application of symbolic computer programs for exaustive search of all different combinations of the basis formulas. *The third way consists in full–scale computerization of all operations* inherent in the factorization method which would open up the possibility for computer–aided derivation of any formula relating to the theory of hypergeometric functions. One of the goals of the present paper is to substantiate, theoretically, the feasibility of the third approach.

1.5. *An experience in development of symbolic computer programs.* All programs developed so far [2, 3] are based on the second (intermediate) approach. These programs utilize not the factorization method *per se*, but the result of its application to a certain class of formulas with the aim to derive manually, without using a computer, a set of basic relationships playing the role of building blocks for a chosen class of formulas. Applying such a "bounded–universal" procedure to one or another of several score classes of formulas making up the backbone of the theory of hypergeometric series one can obtain, in principle, sufficiently complete computerized version of the theory indirectly connected with the factorization method. The initial steps in this direction covering the transformation theory of hypergeometric series are made, with sufficient completeness, in the programs announced in [2, 3].

The notable success of this approach was corroborated by the computer–aided derivation [4] of an important reduction formula given by Gelfand at al. [5] and its numerous non-trivial generalizations.

1.6. *An outline of the project.* The accumulated experience suggests the desirability of expanding the programs' potentialities, in the spirit of the third approach (see Section 1.4).

The central part of the core (CPC) of the projected program complex is conceived to perform, directly, all operations of the operator factorization method thus placing at the user's disposal a sort of a universal interactive "formula synthesizer".

The peripheral part of the core (PPC) is planned to consist of macro-commands implementing the sets of basic relations obtained with the help of the CPC. Many of these relations have been already obtained manually [4, 6]. Very large sets of relations can be obtained with the help of the PPC in an automatic mode.

If the relations presenting little interest for derivation of new formulas are consided to be valuable on their own they will be placed at date base surrounding (DBS). The DBS will play the role of an information reference system. The DBS is expected to be of moderate size for the main bulk of relations will be generated in PPC.

1.7. *How the method would work in the program.* The functionality of the method in the program would not differ much from the work using hand and pencil. There are two general schemes for carrying out calculations with the help of operator factorization method.

The first scheme consists of four steps.

Step 1 (analysis) breaks up a series into an Ω-product of simpler series. **Step 2 (simple series transformations)** utilizes the known properties of the simpler series for transformation of factorized terms. **Step 3 (auxiliary transformations)** uses a finite set of auxiliary identities converting the resultant expression into a form allowing application of a (possible new) factirization formula. **Step 4 (synthesis)** transforms the operator expression into an algebraic form with the help of a suitable factorization formula.

The second scheme is based on the concept of Ω-equivalence (see Section 1.3). Introducing a simple relation we can multiply its both sides by an Ω-factor thus obtaining a new relation.

Despite the seemingly exotic nature of these approaches, they prove in practice to be quite simple, universal and effective, and thus fairly suitable for the role of a superstructure over one of the existing analytical manipulation systems.

2 Instruction set underlying the central part of the core (CPC) of the proposed program complex

The instruction set duplicates the main operations of the factorization method. Each *definition* of a formula giving us a project of a future entry of the instruction set is supplied with *capital* boldface label. Sometimes mnemonic synonyms of the instructions names are indicated in parentheses. The numbers attached to the labels of kindred formulas are to be subsituted, in course of the program implementation, by exact specific indications of the commands formats.

The *references* to the definitions will be denoted by *lower case* boldface labels. The whole set of formulas given below constitutes a project of a specialized algorithmic language. This language may give an origin to a program complex aimed at a full-scale computerization of a substantial part of applied mathematical analysis. For convenience, all formulas are subdivided into several classes.

Sometimes, if a formula written for the case of one variable can be extended directly to the case of several variables we do not present the latter explicitly.

For notation see Refs. [1] and [13].

2.1. *Factorization formulas*

FACT1. Factorization of the series $F[\mathbf{d}; x]$

$$F[\mathbf{d}_1\,;\, x_1\, d(s)]\, F[\mathbf{d}_2; x_2 s]|_{s=0} = F[\mathbf{d}_1, \mathbf{d}_2\,;\, x_1\, x_2]\,, \qquad d(s) = d/ds \tag{4}$$

FACT2. Factorization of the series in one variable containing compound parameter

$$F[\mathbf{d}_1\,;\, x_1\, d(s)]\, F[\mathbf{d}_2; x_2 s^m]|_{s=0} = F[<\mathbf{d}_1 \mid m>, \mathbf{d}_2\,;\, x_1^m x_2] \tag{5}$$

FACT3. General factorization of the series ${}^N F$

$$\begin{aligned} &{}^N F[L_1,\, L_2\,;\, x_1,\, \ldots\,, x_N] \\ &\quad =^N F\left[L_1\,;\, d(s_1),\, \ldots\,, d(s_N)\right]{}^N F[L_2\,;\, x_1 s_1,\, \ldots\,, x_N s_N]\,\big|_{\forall s_n = 0} \end{aligned} \tag{6}$$

FACT4. Special factorization of the series ${}^N F$

$$\begin{aligned} &{}^N F[<\mathbf{d} \mid m_1,\, \ldots\,, m_N>,\, L\,;\, x_1,\, \ldots\,, x_N] \\ &\quad = F\left[\mathbf{d}\,;\, d(s)\right]\,{}^N F[L\,;\, x_1 s^{m_1},\, \ldots\,, x_N s^{m_N}]\,\big|_{s=0} \end{aligned} \tag{7}$$

FACT5. Factorization of ${}^N F$ containing the glueing operator $F[\mathbf{d}_0\,;\, x d^m(s)]$

$$\begin{aligned} &F[\mathbf{d}_0\,;\, x d^m(s)]\, F[\mathbf{d}_1\,;\, x_1 s^m] \cdots F[\mathbf{d}_N\,;\, x_N s^m]|_{s=0} \\ &\quad = F\left[\mathbf{d}_0, \frac{1}{m}, \ldots, \frac{m-1}{m}\ :\ \mathbf{d}_1\,;\, \ldots\,;\, \mathbf{d}_N\,;\, x x_1 m^m, \ldots, x x_N m^m\right] \end{aligned} \tag{8}$$

FACT6. Factorization of multiple series containing constant arguments

$$\Omega =^{N+P} F[\langle \mathbf{d}|m_1, \dots, m_{N+P}\rangle, \dots : x_1\frac{d}{ds_1}, \dots, x_N\frac{d}{ds_N}, u_1, \dots, u_P]$$

$$\times^{N+Q}F[\langle \mathbf{d}'|l_1, \dots, l_{N+Q}\rangle, \dots : y_1 s_1, \dots, y_N s_N, v_1, \dots, v_Q]\,|_{\forall s_n=0} \tag{9}$$

$$=^{N+P+Q} F[\langle \mathbf{d}|m_1, \dots, m_N, m_{N+1}, \dots, m_{N+P}, \underbrace{0, \dots, 0}_{Q}\rangle,$$

$$\langle \mathbf{d}'|l_1, \dots, l_N, \overbrace{0, \dots, 0}^{P}, l_{N+1}, \dots, l_{N+Q}\rangle, \dots :$$

$$x_1y_1, \dots, x_Ny_N, u_1, \dots, u_P, v_1, \dots, v_Q] \tag{10}$$

2.2. *General properties of the Ω-multiplication operation*
OMEGA 1. Commutativity property $u * v = v * u$ (**COMM**)

$$u(d(s))v(xs)|_{s=0} = v(d(s))u(xs)|_{s=0} \tag{11}$$

OMEGA 2. Coupling rule (**COUP**)

$$u(d(s))v(xs)|_{s=0} = u(xd(s))v(s)|_{s=0} \tag{12}$$

OMEGA 3. Associativity property (**ASSOC**)

$$< w * (u * v)|x >=< (w * u) * v|x > \tag{13}$$

OMEGA 4. $\exp(x)$ plays the role of Ω-unit (**OMUN**) that is $\exp * f = f * \exp = f$, or

$$\exp(d(s))f(xs)|_{s=0} = f(d(s))\exp(xs)_{s=0} = f(x) \tag{14}$$

OMEGA 5. The Ω-"unitarity" can be interpreted as "renaming" (s for x) property (**REN**)

$$\exp xd(s)f(s)|_{s=0} = f(x) = f(x)|_{s\Rightarrow x} \tag{15}$$

2.3. *Ω-equivalent operators*
The case of **EQUIV1**. an arbitrary operator multiplied by power function

$$F(d(s))s^n\,\Psi(s)|_{s=0} = F^{(n)}(d(s))\,\Psi(s)|_{s=0} \tag{16}$$

The case of **EQUIV2**. an arbitrary operator multiplied by exponential function

$$F(d(s))e^{xs}\,\Psi(s)|_{s=0} = F(d(s)+x)\,\Psi(s)|_{s=0} \tag{17}$$

The case of **EQUIV3**. the binomial operator multiplied by exponential function

$$F_0^1[a; d(s)]e^{xs}\,\Psi(s)|_{s=0} = (1-x)^{-a}F_0^1\left[a; d(s)/1-x\right]\,\Psi(s)\Big|_{s=0} \tag{18}$$

2.4. *Relationships containing operators (without setting differentiation variable to zero)*
OPER1. Shift operator identity (**SHIFT**)

$$\exp(u\,d(x))f(x) = f(x+u) \tag{19}$$

OPER2. Similarity transformation (F and f are arbitrary functions, A is an arbitrary operator) (**SIMIL**)

$$f^{-1}\,F(A)\,f = F(f^{-1}A\,f\,) = F(A + f^{-1}[A, f\,]\,) \tag{20}$$

OPER3. Operator argument displacement formula (**DISP**)

$$\exp(-vx)F[d(x)]\exp(vx) = F[d(x) + v] \tag{21}$$

OPER4. Applying of a differential operator to $\exp(x)$ (**OPEXP**)

$$F(d(x))\,\exp(ux) = F(u)\,\exp(ux) \tag{22}$$

OPER5. Generalized Leibnitz rule (**LEIB**)

$$F(d(x))\,f_1(x)f_2(x) = F(d(x_1) + d(x_2))f_1(x_1)f_2(x_2)|_{x_1=x_2=x} \tag{23}$$

OPER6. Differentiation of simple hypergeometric series (**DIFHYP**)

$$d^n(x)\,F[\mathbf{d};\,ux] = u^n(\mathbf{d}, n)\,F[\mathbf{d} + n;\,ux] \tag{24}$$

2.5. *Elementary reduction formulas*
RED1. Reduction of the exponential series F_0^0 (**REDEXP**)

$$F_0^0[*//*; x] = \exp(x) \tag{25}$$

RED2. Reduction of the binomial series F_0^1 (**REDBIN**)

$$F_0^1[a//*; x] \equiv F[a; x] = (1 - x)^{-a} \tag{26}$$

RED3. Reduction of an infinite geometrical progression (**GEOINF**)

$$1 + x + x^2 + \cdots \equiv F_0^1[1; x] = (1 - x)^{-1} \tag{27}$$

RED4. Reduction of the finite geometrical progression (**GEOFIN**)

$$1 + x + x^2 + \cdots + x^N = (1 - x^{N+1})/(1 - x) \tag{28}$$

RED5. Reduction of the series ${}^N F$ with empty glueing set

$${}^N F[*//*\,:\,\mathbf{d}_1\,;\ldots;\,\mathbf{d}_N;\,x_1,\ldots,x_N] = F[\mathbf{d}_1;\,x_1]\cdots F[\mathbf{d}_N;\,x_N] \tag{29}$$

RED6. Reduction of the series ${}^N F$ with empty individual sets

$${}^N F[\mathbf{d}\,:\,*\,;\ldots;\,*\,;\,x_1,\ldots,x_N] = F[\mathbf{d}\,;\,x_1 + \cdots + x_N] \tag{30}$$

2.6. *Auxiliary algebraic identities*
ALG1 Gauss-Legendre multiplication formula for the Pochhammer symbol (**MULT(m)**)

$$(\alpha, m\,i)\;=\;m^{m\,i}\left(\frac{\alpha}{m}, i\right)\left(\frac{\alpha+1}{m}, i\right)\cdots\left(\frac{\alpha+m-1}{m}, i\right) \tag{31}$$

ALG2. Inversion formula for Pochhammer symbol (**INVER**)

$$(a,\,-I) = (-1)^I\,(1-a,\,I)^{-1} \tag{32}$$

ALG3. Cancellation or, *vica versa*, introduction of equal parameters in numerator and denominator of a series (**CANC, INTRO**)

$$F[\mathbf{d};\,x] = F[a,\,\mathbf{d}//a;\,x] \tag{33}$$

ALG4. Vertical transfer of parameters (**VERT**)

$${}^N F[\langle a|m_1,\ldots,m_N\rangle,\,L;\,\mathbf{x}] ={}^N F[L//\langle 1-a|\bar{m}_1,\ldots,\bar{m}_N\rangle;\,(-1)^{\mathbf{m}}\,\mathbf{x}]\,, \tag{34}$$

$\mathbf{x} = [x_1,\ldots,x_N], \quad (-1)^{\mathbf{m}}\,\mathbf{x} = [(-1)^{m_1}x_1,\ldots,(-1)^{m_N}x_N]$

ALG5. Any series ${}^N F$ is symmetric with respect to simultaneous permutation of arguments $x_i \rightleftharpoons x_j$, individual sets of parameters $\mathbf{d}_i \rightleftharpoons \mathbf{d}_j$ and all corresponding spectral numbers $m_i \rightleftharpoons m_i$, $l_i \rightleftharpoons l_j$, etc. (**PERM**)

ALG6. Uniformization of the argument of the binomial series $F_0^1[a;\,x+u]$ (**UNIF**)

$$F_0^1[a;\,x+u] = (1-x-u)^{-a} = (1-u)^{-a}\,F_0^1\,[a;\,x/(1-u)] \tag{35}$$

ALG7. Factorization of geometrical progression $F_0^1[1;\,x]$ into a product of two progressions (**PROG**)

$$F_0^1[1;\,x] = \left(\sum_{r=0}^{N-1} x^r\right)\,F_0^1[1;\,x^N] \tag{36}$$

ALG8. Decomposition of e^x into a sum of N series $F_{N-1}^0(x^N)$ (**DEXP(N)**)

$$e^x = \sum_{r=0}^{N-1}\frac{x^r}{r!}\,F\left[\begin{matrix}1\,;\,x^N\\ \langle 1+r|N\rangle\end{matrix}\right] = \sum_{r=0}^{N-1}\frac{x^r}{r!}\,F\left[\begin{matrix}*\,; \quad (x/N)^N\\ \frac{1+r}{N},\ldots,\frac{N-1}{N},\frac{N+1}{N},\ldots,\frac{N+r}{N}\end{matrix}\right] \tag{37}$$

ALG9. Addition formula for binimial series (**ADDBIN**)

$$F_0^1[a;\,x_1+x_2] = \sum_{n=0}^{\infty}\frac{(a,n)}{n!}\,x_1^n\,F_0^1[a+n;\,x_1]\,x_2^n\,F_0^1[a+n;\,x_2] \tag{38}$$

3 Examples suggestive of functionality specifics of the proposed CPC commands

3.1. *An elementary example* can be seen from comparison between relationships **prog** and **dexp(N)** which prove to be Ω-equivalent one to another! The elementary formula **prog** follows from the formulas **geofin** and **geoinf**. Thus it is just the **prog** plays the role of proto-relation (see Sec. 1.3). Applying the operator $F_1^0[*//1;\,z\,d(x)]|_{x=0}$ to the both parts of **prog** we use the operations **fact1, canc** and **redexp** in the left-hand part and the operations **equiv1, difhyp, fact2, mult(m)** and **canc** in the right-hand part. Then the **dexp(N)** follows immediately. More general relations belonging to this class can be obtained in analogous way if we apply $F[\mathbf{d};\, zd(x)]|_{x=0}$, instead of the F_1^0, to the both parts of **prog**.

3.2. *Already published examples.* In fact, we are delivered from the necessity of giving many examples of how the method could work in practice. Numerous examples of the kind are given in the already published papers [1], [7]-[13]. Many simple examples illustrating application of operations (4) - (38) are given is Ref. [1]. The same operations were emlpoyed for derivation of new recurrence relations [7]. Some new generating functions for the Laguerre polynomials were presented in Ref. [8]. More general generating functions, as well as a complete set of Meixner-type formulas and a new class of Lagrangean polynomials were introduced in [9]. Very important special transformations of the Appel F_4 and the Horn H_1 and G_2 functions were obtained in [10]. A new approach to derivation and generalization of involved Burchnall and Chaundy expansions playing a particularly important role in the theory of double hypergeometric series was found in [11]. A sophisticated analysis of many particular problems originated from contemplation of a classical relation between Bessel functions was given in [12]. A heavy use of the operations (4) - (38) was made in [13] for analysis of linearization relations and addition formulas including a generaliztion of an important Koornwinder formula of the Jacobi polynomials. Special attention has been given in Ref. [13] to the details of the new technology of analytical transformations based on the operations (4) - (38).

3.3. *An additional example.* The references given in Sec. 3.2 relate mostly to the main scheme involving the four steps mentioned in Sec. 1.7. The second scheme based on the concept of Ω-equivalency was paid lesser attention in the above examples and needs therefore a little bit more substantiation. To this end we introduce the notation

$$u_1 = 1 - z + \xi_1 z, \quad u_2 = 1 - z + \xi_2 z$$

and consider the elementary relationship

$$\mathcal{L} \equiv F_0^1[c; (1-\xi_1)(1-\xi_2)z] = (1-z)^c (u_1 u_2)^{-c} F_0^1[c;\, \xi_1\xi_2 z/u_1 u_2] \equiv \mathcal{R}. \tag{39}$$

which is readily verified by using the reduction rule **redbin**. We then transform the proto-relation (39) to an Ω-identical form facilitating transition to Ω-equivalent relations. Using twice the operation **fact1** we get the preliminary Ω-identical (see Sec. 1.3) transformation

$$F_0^1\left[c; \frac{\xi_1\xi_2 z}{u_1 u_2}\right] = \prod\nolimits_{n=1}^{2} F_0^1\left[c; \frac{\xi_n(1-z)}{u_n}\frac{d}{ds_n}\right] F_1^0\left[\begin{matrix}*; \\ c\end{matrix}\ \frac{s_1 s_1 z}{(1-z)^2}\right]\Bigg|_{s_1=s_2=0} \tag{40}$$

To simplify dependence on ξ_1, ξ_2 we multiply eq. (40) by $u_1^{-c}u_2^{-c}$ (see eq. (39)), transform the both operator series F_0^1 with the help of **redbin**, allow for definitions of u_1, u_2, make some elementary algebraic manipulations, use, inversely, the **redbin** and employ **equiv2**. Thus we have

$$u_n^{-c} F_0^1\left[c; \frac{\xi_n(1-z)}{u_n}\frac{d}{ds_n}\right] = (1-z)^{-c} F_0^1\left[c;\, \xi_n\left(\frac{d}{ds_n} + \frac{z}{z-1}\right)\right]$$
$$\Leftrightarrow (1-z)^{-c} F_0^1[c; \xi_n d(s_n)] \exp[z s_n (z-1)^{-1}]. \tag{41}$$

Inserting (40) and (41) into (39) we finally have the desired Ω-identical representation of

$\mathcal{R}$:

$$\mathcal{R} = (1-z)^{-c}\, F^1_0[c;\, \xi_1 d(s_1)] F^1_0[c;\, \xi_2 d(s_2)] \\ \times \exp\left(\frac{zs_1}{z-1}\right) \exp\left(\frac{zs_2}{z-1}\right) F^0_1 \left[\begin{matrix} *\,; & \dfrac{s_1 s_2 z}{(1-z)^2} \\ c & \end{matrix}\right]\Bigg|_{s_1=s_2=0} . \tag{42}$$

With the help of **redbin** the left-hand side $\mathcal{L}$ can be writhen as:

$$\mathcal{L} = \sum\nolimits_{n=0}^{\infty} \frac{(c,n}{n!} F^1_0[-n, \xi_1] F^1_0[-n, \xi_2]\, z^n . \tag{43}$$

Then we apply the operator

$$F^0_1[*//c;\, x_1 d(\xi_1)] F^0_1[*//c;\, x_2 d(\xi_2)]\,|_{\xi_1=\xi_2=0}$$

to the both sides of the identity $\mathcal{L} = \mathcal{R}$ where $\mathcal{L}$ and $\mathcal{R}$ are given by eqs. (42) and (43) respectevely. In case of the $\mathcal{L}$ the only operation **fact1** is needed. In case of the $\mathcal{R}$ we apply **fact1, canc, redexp** and **ren**, consecutively. The result

$$\sum\nolimits_{n=0}^{\infty} \frac{(c,n)}{n!} F^1_1 \left[\begin{matrix} -n\,; & x_1 \\ c & \end{matrix}\right] F^1_1 \left[\begin{matrix} -n\,; & x_2 \\ c & \end{matrix}\right] z^n \\ = (1-z)^{-c} \exp\left[\frac{(x_1+x_2)z}{z-1}\right] F^0_1 \left[\begin{matrix} *\,; & \dfrac{x_1 x_2 z}{(1-z)^2} \\ c & \end{matrix}\right] \tag{44}$$

is equivalent to the Hille-Hardi bilinear generating function for Laguerre polynomials (see [14], vol.1).

Applying to the both sides of eq. (44) the operator product

$$F^1_0[a_1;\, \xi_1 d(x_1)] F^1_0[a_2;\, \xi_2 d(x_2)]|_{x_1=x_2} ,$$

using twice **fact1** on the left and **equiv3** (see eq. (41)) and **fact1** on the right we obtain the apparently new bilinear generating functions for Gaussian polynomials $F[-n, a//c;\, \xi]$:

$$\sum\nolimits_{n=0}^{\infty} \frac{(c,n)}{n!} F^2_1 \left[\begin{matrix} -n, a_1\,; & \xi_1 \\ c & \end{matrix}\right] F^2_1 \left[\begin{matrix} -n, a_2\,; & \xi_2 \\ c & \end{matrix}\right] z^n \\ = (1-z)^{a_1+a_2-c}\, u_1^{-a_1} u_2^{-a_2} F[a_1, a_2//c; \xi_1\xi_2 z/u_1u_2] . \tag{45}$$

An attempt to apply eq.(45) to the Gegenbauer polynomials $C^\lambda_n(x)$ may seem to make no sense whatever because any of the known hypergeometric representations of $C^\lambda_n(x)$ in the form of F^2_1 contains *two* parameters dependent on n whereas each of the F^2_1 in eq. (45) contains only *one* such parameter. Being sure that the list of representations for the $C^\lambda_n(x)$ given in literature is not complete we looked into a question of how many different formulas for the $C^\lambda_n(x)$ may exist. We used the linear ($x \to x^{-1}$, $x \to 1-x$, $x \to x/(x-1)$) and quadratic transformations (see [14], vol. 2 and the Sec. 4.2 below) conserving the polynomial structure of the transformed functions. We found altogether 18 different representations. The formula

$$C^\lambda_n(x) = \frac{(2\lambda, n)}{n!} [x + (x^2-1)^{1/2}]^n\, F^2_1 \left[\begin{matrix} -n, \lambda\,; & \dfrac{2(x^2-1)^{1/2}}{x+(x^2-1)^{1/2}} \\ 2\lambda & \end{matrix}\right] \tag{46}$$

which is incidentally absent in literature is of prime interest for applications.

This formula is remarkable for that the dependence on the order n of the polynjmial $F[-n,\lambda//2\lambda]$ in the definition (46) in the same as in the case of the polynomials $F[-n//\alpha+1]$ occuring in the definition of the Laguerre polynomials. The structural similarity of the $C_n^{\lambda}(x)$ and L_n^{α} shows that solution of the problems where dependence on n plays an essential role would have, formally, much in common for the $C_n^{\lambda}(x)$ and L_n^{α} despite the fact that the Gegenbauer polynomials are a natural particular case of the Jacobi polynomials which have nothing to do with the Laguerre polynomials. Beyond the factorization method this formal observation would hardly be of any significance. On the contrary, within the factorization method the simple observation gives us a powerful tool for obtaining new interesting results. For example, letting $a_1 = a_2 = \lambda$, $c = 2\lambda$ in eq. (45) and using eq. (46) to express the resultant F_1^2 polynomials through $C_n^{\lambda}(x)$ we can readily obtain a seemingly new bilinear generating function for the Gegenbauer polynomials. Derivation of bilateral, for the L_n^{α} and $C_n^{\lambda}(x)$, generating function can be also performed with ease.

4 Examples of macro-commands constituiting the peripheral part of the core (PPC) of the proposed program complex

As was already mentioned above (see Sec. 1.5 and 1.6) along with the universal set of "low-level" derivation rules (4) - (38) we are going to use the generators of formula classes consisting of a few specialized "high-level" basic relationships. In distinction to the CPC operations many of the PPC macro-commands have been already programmed [2, 3] and applied for analysis of multiple hypergeometric series [4, 15].

We first give typical instances of macro-commands and then present some examples of using these macro-commands (all necessary definitions and notation are given in the refs. [1, 13]).

4.1. *Linear transformations*

LIN(K). Linear transformation connecting two series having the Kummer type (1//1) with respect to x_0 (the L^* symbolizes the coefficients independent of summation index i_0):

$$F\left[\begin{matrix}\langle\nu_1|1,\mathbf{m}_1\rangle\,,\ L^*\,;\ x_0,\mathbf{x}\\ \langle\nu_0|1,\mathbf{m}_0\rangle\end{matrix}\right] = e^{x_0}\,F\left[\begin{matrix}\langle\nu_{0\bar{1}}|1,\mathbf{m}_{0\bar{1}}\rangle,\langle\nu_1|0,\mathbf{m}_1\rangle,\ L^*\,;\ -x_0,\mathbf{x}\\ \langle\nu_0|1,\mathbf{m}_0\rangle\ \,,\langle\nu_{0\bar{1}}|0,\mathbf{m}_{0\bar{1}}\rangle\end{matrix}\right] \tag{47}$$

LIN(G). Three linear transformations linking the series having the Gauss type (2//1) with respect to x_0. For each series we use a canonical representation [1, 13], where all spectral numbers connected with x_0 are equal to 1. The three transformations change, consecutively, the first (G01), the second (G02) and the both (G00) numerator parameters. The symbol following G in (GOQ) is the number of argument.

$$F\left[\begin{matrix}\langle\nu_1|1,\mathbf{m}_1\rangle\,,\ \langle\nu_2|1,\mathbf{m}_2\rangle\,,\ L^*\,;\ x_0,\mathbf{x}\\ \langle\nu_0|1,\mathbf{m}_0\rangle\end{matrix}\right] =$$

LIN(G01) $= L_1^0 F \equiv (1-x_0)^{-\nu_2} \times$

$$\times F\left[\begin{array}{l}\langle\nu_{0\bar1}|1,\mathbf{m}_{0\bar1}\rangle\,,\ \langle\nu_2|1,\mathbf{m}_2\rangle\,,\ \langle\nu_1|0,\mathbf{m}_1\rangle\,,\ L^*\,;\ \dfrac{x_0}{x_0-1},\dfrac{\mathbf{x}}{(1-x_0)^{\mathbf{m}_2}}\\ \langle\nu_0|1,\mathbf{m}_0\rangle\,,\ \ \langle\nu_{0\bar1}|0,\mathbf{m}_{0\bar1}\rangle\end{array}\right] \tag{48}$$

LIN(G02) $= L_2^0 F \equiv (1-x_0)^{-\nu_1} \times$

$$\times F\left[\begin{array}{l}\langle\nu_1|1,\mathbf{m}_1\rangle\,,\ \langle\nu_{0\bar2}|1,\mathbf{m}_{0\bar2}\rangle\,,\ \langle\nu_2|0,\mathbf{m}_2\rangle\,,\ L^*\,;\ \dfrac{x_0}{x_0-1},\dfrac{\mathbf{x}}{(1-x_0)^{\mathbf{m}_1}}\\ \langle\nu_0|1,\mathbf{m}_0\rangle\,,\ \ \langle\nu_{0\bar2}|0,\mathbf{m}_{0\bar2}\rangle\end{array}\right] \tag{49}$$

LIN(G00) $= L_0^0 F \equiv (1-x_0)^{\nu_{0\bar1\bar2}} \times$

$$\times F\left[\begin{array}{l}\langle\nu_{0\bar1}|1,\mathbf{m}_{0\bar1}\rangle,\ \langle\nu_{0\bar2}|1,\mathbf{m}_{0\bar2}\rangle,\ \langle\nu_1|0,\mathbf{m}_1\rangle\,,\langle\nu_2|0,\mathbf{m}_2\rangle\,,\ L^*\,;\ x_0,\mathbf{X}\\ \ \langle\nu_0|1,\mathbf{m}_0\rangle,\ \ \ \langle\nu_{0\bar1}|0,\mathbf{m}_{0\bar1}\rangle,\ \langle\nu_{0\bar2}|0,\mathbf{m}_{0\bar2}\rangle\end{array}\right], \tag{50}$$

$$\mathbf{X} = \mathbf{x}(1-x_0)^{-\mathbf{m}_{\bar012}}\,.$$

4.2. *Quadratic transformations*
The "classical" theory of quadratic transfornations even in case of simple series lacks simplicity and transparency of structure to say nothing of multiple case. The relationships which follow are applicable to *any multiple series* satisfying some necessary conditions. Moreover instead of 9 functions we can confine ourselves, at the first step, but to 3 functions:

$$F_1 \equiv F_1[\langle\nu_1|1,\mathbf{m}_1\rangle,\langle\nu_1+1/2|1,\mathbf{m}_1\rangle,L^*//\langle\nu_0|1,\mathbf{m}_0\rangle;x_0,\mathbf{x}], \tag{51}$$

$$F_2 \equiv F_2[\langle\nu_1|1,\mathbf{m}_1\rangle,\langle\nu_2|1,\mathbf{m}_2\rangle,L^*//\langle1+\nu_{1\bar2}|1,\mathbf{m}_{1\bar2}\rangle;x_0,\mathbf{x}], \tag{52}$$

$$F_3 \equiv F_3[\langle\nu_1|1,\mathbf{m}_1\rangle,\langle\nu_2|1,\mathbf{m}_2\rangle,L^*//\langle2\nu_2|1,2\mathbf{m}_2\rangle;x_0,\mathbf{x}]. \tag{53}$$

Just these functions occur in the following three basic quadratic transformations:
QUAD32. The transformation relating F_3 to F_2 is

$$F_3 = [(2/(2-x_0)]^{\nu_1}F_1\left[\begin{array}{l}\langle\frac{\nu_1}{2}|1,\frac{\mathbf{m}_1}{2}\rangle,\langle\frac{\nu_1+1}{2}|1,\frac{\mathbf{m}_1}{2}\rangle,L^*;x_{13},\mathbf{x}_{13}\\ \langle\nu_2+1/2|1,\mathbf{m}_2\rangle\end{array}\right] \tag{54}$$

$x_{13} = x_0^2(2-x_0)^{-2}$, $\mathbf{x}_{13} = 4^{\mathbf{m}_{1\bar2}}\,\mathbf{x}\,(2-x_0)^{-\mathbf{m}_1}$.
QUAD21. The transformation linking F_2 and F_1 has the form

$$F_2 = (1+x_0)^{-\nu_1}F_1\left[\begin{array}{l}\langle\frac{\nu_1}{2}|1,\frac{\mathbf{m}_1}{2}\rangle,\langle\frac{\nu_1+1}{2}|1,\frac{\mathbf{m}_1}{2}\rangle,\langle\nu_2|0,\mathbf{m}_2\rangle,L^*;x_{12},\mathbf{x}_{12}\\ \langle1+\nu_1-\nu_2|1,\mathbf{m}_1-\mathbf{m}_2\rangle\end{array}\right] \tag{55}$$

$x_{12} = 4x_0(1+x_0)^{-2}$, $\mathbf{x}_{12} = 2^{\mathbf{m}_1}\mathbf{x}(1+x_0)^{-\mathbf{m}_1}$.
QUAD32. The transformation expressing F_3 through F_2 is

$$F_3 = [2/(1+\sqrt{1-x_0})]^{2\nu_1}F_2\left[\begin{array}{l}\langle\nu_1|1,\mathbf{m}_1\rangle,\langle\nu_{1\bar2}+\frac12|1,\mathbf{m}_{1\bar2}\rangle,L^*;x_{23},\mathbf{x}_{23}\\ \langle\nu_2+1/2|1,\mathbf{m}_2\rangle\langle\nu_{1\bar2}+\frac12|0,\mathbf{m}_{1\bar2}\rangle\end{array}\right] \tag{56}$$

$x_{23} = (1-\sqrt{1-x_0})^2(1+\sqrt{1-x_0})^{-2}$, $\mathbf{x}_{23} = 4^{\mathbf{m}_{1\bar2}}\mathbf{x}(1+\sqrt{1-x_0})^{-2\mathbf{m}_1}$.
All other quadratic transformations follow from eqs. (54)-(56) by using the three operations **lin(G)** for F_1, F_2, F_3 occuring in (54)-(56). This adds 6 new functions. Letting

$N = 0$ we thus obtain a complete systematic set of quadratic transformations of the Gauss function F_1^2.

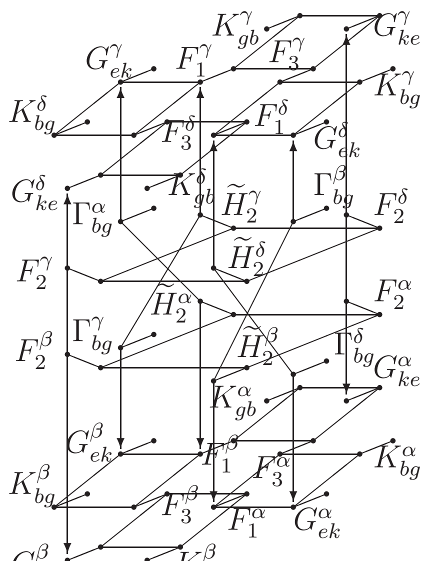
4.3. *Examples.* Many interesting examples can be found in ref. [4] where the macro-commands **lin(G)** were applied to Gelfand functions on grassmanians $G_{2,4}$ and $G_{3,6}$. These functions depend on three and four variables, respectively. The linear transformations permitted us to use a new algorithm of finding reducible cases of these functions. The idea of the algorithm lies in transforming the functions to the form allowing one out of 6 *elementary* self-explanatory reduction rules to be used. The list of the elementary reductions is given in ref. [4].

To conclude with, we give, without going into details, the results of computer analysis of the special Appell function $F_4[a_1, a_2, a_1, b_2; x_1\,, x_2]$. We first used a representation of the general F_4 function through a complete series of the third order [10] and then, confining ourselves to the special case, we obtained two different expressions of the general F_4 in the form of the following non-Hornian functions:

$$K_{gb} = F\begin{bmatrix} \alpha : a_1, a_1' \,;\, a_2 \,;\, x_1, x_2 \\ \beta : \quad b_1 \quad ;\, * \end{bmatrix}, \tag{57}$$

$$\Gamma_{bg} = F\begin{bmatrix} \langle\alpha_1|1,\bar{1}\rangle\langle\alpha_2|\bar{1},1\rangle : a_1 \,;\, a_2, a_2' \,;\, x_1, x_2 \\ : * \,;\quad b_2 \end{bmatrix}. \tag{58}$$

The processing of these functions consisted in using all possible linear commands **lin(G)** along with an auxiliary bilinear transformation applicable to the functions containing an "indefinite" parameter $\langle 0|1,\bar{1}\rangle$. The process performed in automatic mode gave us the following 5 functions:



$$G_{ek} = F\begin{bmatrix} \alpha_1, \alpha_2 : * \,;\, a_2 \,;\, x_1, x_2 \\ \beta \quad : * \,;\, b_2 \end{bmatrix} \tag{59}$$

$$F_1 = F\begin{bmatrix} \alpha : a_1 \,;\, a_2 \,;\, x_1, x_2 \\ \beta : * \,;\, * \end{bmatrix} \tag{60}$$

$$F_2 = F\begin{bmatrix} \alpha : a_1 \,;\, a_2 \,;\, x_1, x_2 \\ * : b_1 \,;\, b_2 \end{bmatrix} \tag{61}$$

$$F_3 = F\begin{bmatrix} * : a_1, a_1' \,;\, a_2, a_2' \,;\, x_1, x_2 \\ \beta : \quad * \quad ; \quad * \end{bmatrix} \tag{62}$$

$$\widetilde{H}_2 = F\begin{bmatrix} \langle\alpha|\bar{1},1\rangle : a_1, a_1' \,;\, a_2 \,;\, x_1, x_2 \\ : \quad * \quad ;\, b_2 \end{bmatrix} \tag{63}$$

The result of all transformation is represented at the diagram. Black nodes at the ends of short segments denote arguments of the double series. Long lines symbolize linear transformations. For more details see Ref.[15].

5 Concluding remarks

The program implementation of the "central part of the core" (see Sec. 2) would allow us, instead of using a "manual", with pen and paper, calculation technique, to perform all analytical transformations with the help of computer in an interactive mode, in this way relieving the researcher of the tedious copying of cumbersome formulas and placing at his disposal a universal "formula synthesizer" of a sort.

An addition of new macro-commands to the "peripheral part of the core" (see Sec. 4) would give us an access to hundreds and thousands of new relationships, whose publication in the traditional form of books and periodicals would hardly have been practicable.

An inappropriately cumbersome user interface may well happen to become a substantial practical obstacle to the effective program realization of the global approach. A detailed look at the work with the formulas' "screen images" and sophisticated investigation of different variants of its organization seems to be obligatory condition for the effective man–machine formula interface. We hope that recourse to "semantics-oriented" tools, like XML, OpenMath, etc., may help us to solve the problem.

References

[1] A.W.Niukkanen A new theory of multiple hypergeometric series and its prospects for computer algebra programming (in Russian), *Fundamentalnaya i prikladnaya matematika*, 1999, **5**, 1–29.

[2] O.S.Paramonova, A.W.Niukkanen, Analytical transformations of hypergeometric series (in Russian), *Programmirovanie*, 1998, No 6, 25–26.

[3] O.S.Paramonova, A.W.Niukkanen, New algorithms of finding reduction formulas of multiple hypergeometric series (in Russian), *Programmirovanie*, 2000, No 1, 62–63.

[4] A.W.Niukkanen, O.S.Paramonova, Computer generation of complicated transformations for multiple hypergeometric series, *Comput. Phys. Commun.*, 2000, **126**, 141–148.

[5] I.M. Gel'fand, M.I. Graev, V.S. Retakh, General hypergeometric equations systems and hypergeometric type series (in Russian), *Uspekhi Matem. Nauk*, 1992, **47**, 3–82.

[6] A. W. Niukkanen, Transformation theory of multiple hypergeometric series and computer aided symbolic calculations, *Proceedings of the 9th Conference on Computational modelling and computing in physics*, Dubna, 1997, 219–223.

[7] A.W.Niukkanen, Factorization method and recurrence relations (in Russian), *Integral transforms and special functions.* Computer center, 1997, **1**, No 2, 10-15.

[8] A. W. Niukkanen, A new method in the theory of the hypergeometric series (in Russian), *Uspekhi Matematicheskikh Nauk*, 1988, **43**, 191–193.

[9] A. W. Niukkanen, A new approach to the theory of the hypergeometric series (in Russian), *Russian Matematicheskie zametki*, 1991, **50**, 65–73.

[10] A.W.Niukkanen, Factorization method and special transformations of the functions F_4, H_1 and G_2 (in Russian), *Uspekhi Matem. Nauk*, 1999, **54**, 169–170.

[11] A.W.Niukkanen, New algoritms for the hypergeometric series and the Burchnall and Chaundy expansions (in Russian), *Programmirovanie*, 2000, No 2, 67–69.

[12] A.W.Niukkanen, I.V.Perevozchikov, V.A.Lurie, A generalization of a classical relation between $J_{\nu+n}(z), J_\nu(z)$ and $J_{\nu-1}(z)$, *Fractional calculus and applied analysis*, 2000, **3**, No 2, 119-132.

[13] A.W.Niukkanen, Operator factorization method and addition formulas for hypergeometric functions, *Integral transforms and special functions*, 2001, **11**, 25-48.

[14] A. Erdélyi, W. Magnus, F. Oberhettinger, F. Tricomi, *Higer transcendental functions*, (McGraw-Hill, N.-Y., 1953).

[15] O.S.Paramonova, A.W.Niukkanen, Computer analysis of transformation formulas of the Appell and the Horn functions, submitted to *Programmirovanie.*