The four double-hypergeometric Appell functions, a complete implementation in a computer algebra system

Edgardo S. Cheb-Terrab
Physics, Differential Equations and Mathematical Functions, Maplesoft

Abstract:
The four multi-parameter Appell functions, AppellF1, AppellF2, AppellF3 and AppellF4 are doubly hypergeometric functions that include as particular cases the 2F1 hypergeometric and some cases of the MeijerG function, and with them most of the known functions of mathematical physics. Appell functions have been popping up with increasing frequency in applications in quantum mechanics, molecular physics, and general relativity. In this talk, a full implementation of these functions in the Maple computer algebra system, including, for the first time, their numerical evaluation over the whole complex plane, is presented, with details about the symbolic and numerical strategies used.

Appell Functions (symbolic)

The main references:

- P. Appel, J.Kamke de Feriet, "Fonctions hypergeometriques et Hyperspheriques", 1926
- 24 papers in the literature, ranging from 1882 to 2015

Definition and Symmetries

The definition of the four Appell series and the corresponding domains of convergence can be seen through the FunctionAdvisor. For example, for AppellF1,

\[
\begin{align*}
F_1(a, b_1, b_2, c, z_1, z_2) &= \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \frac{(a)_{k_1} (b_1)_{k_1} (b_2)_{k_2} (c)_{k_1} z_1^{k_1} z_2^{k_2}}{(k_1!)^{k_1} (k_2!)^{k_2}}, \ |z| < 1 \\
&< 1 \land |z_2| < 1
\end{align*}
\] (1.1.1)

From this definition, by swapping the AppellF1 variables subscripted with the numbers 1 and 2, the function remains the same; hence

\[
\begin{align*}
F_1(a, b_2, b_1, c, z_2, z_1) &= F_1(a, b_1, b_2, c, z_1, z_2)
\end{align*}
\] (1.1.2)
Analogously, the definition and symmetry of AppellF2 are

\[ F_2(a, b_1, b_2, c_1, c_2, z_1, z_2) = \sum_{k_1 = 0}^{\infty} \sum_{k_2 = 0}^{\infty} \frac{(a)_{k_1} + (b_1)_{k_1} (b_2)_{k_2}}{(c_1)_{k_1} (c_2)_{k_2} k_1! k_2!} z_1^{k_1} z_2^{k_2}, \quad |z_1| + |z_2| < 1 \]  

(1.1.3)

> FunctionAdvisor (symmetries, AppellF2)

\[ [F_2(a, b_1, b_2, c_1, c_2, z_1, z_2)] \]  

(1.1.4)

The cases of AppellF3 and AppellF4 are more general in that, from their definition, besides the symmetry under a swap of the subscripted variables, these two functions have additional symmetries under exchange of positions of the function's parameters

> FunctionAdvisor (definition, AppellF3)

\[ F_3(a_1, a_2, b_1, b_2, c, z_1, z_2) = \sum_{k_1 = 0}^{\infty} \sum_{k_2 = 0}^{\infty} \frac{(a_1)_{k_1} (a_2)_{k_2} (b_1)_{k_1} (b_2)_{k_2}}{(c)_{k_1 + k_2} k_1! k_2!} z_1^{k_1} z_2^{k_2}, \quad |z_1| < 1 \wedge |z_2| < 1 \]  

(1.1.5)

> FunctionAdvisor (symmetries, AppellF3)

\[ [F_3(a_1, a_2, b_1, b_2, c, z_1, z_2)] \]  

(1.1.6)

> FunctionAdvisor (definition, AppellF4)

\[ F_4(a, b_1, c_2, z_1, z_2) = \sum_{k_1 = 0}^{\infty} \sum_{k_2 = 0}^{\infty} \frac{(a)_{k_1} (b_1)_{k_1 + k_2} z_1^{k_1} z_2^{k_2}}{(c_1)_{k_1} (c_2)_{k_2} k_1! k_2! \sqrt{|z_1|}}, \quad |z_1| + |z_2| < 1 \]  

(1.1.7)

> FunctionAdvisor (symmetries, AppellF4)

\[ [F_4(a, b_1, c_2, z_1, z_2)] \]  

(1.1.8)

**Polynomial and Singular Cases**

- From these four definitions, the Appell functions are singular (division by zero) when the \( c \) parameters entering the \texttt{pochhammer} functions in the denominators of these series are non-positive integers: these \texttt{pochhammer} functions will be equal to zero when the summation indices
of these series are bigger than the absolute values of the $c$ parameters.

- For an analogous reason, when the $a$ and/or $b$ parameters entering the pochhammer functions in the numerators of the series are non-positive integers, the series will truncate and the Appell functions will be polynomial.

- When both the pochhammers in the numerators and denominators have non-positive integer arguments, the Appell functions are polynomial when the absolute values of the non-positive integers in the numerators are smaller than or equal to the absolute values of the non-positive integers in the denominators, and singular otherwise.

- The combinatorial of all these conditions can also be consulted using the FunctionAdvisor. For example, for AppellF1, the singular cases happen when any of the following conditions hold

$$\text{FunctionAdvisor}(\text{singularities, AppellF1})$$

\[
\begin{align*}
F_1(a, b_1, b_2, c, z_1, z_2), & \\
& (c::\mathbb{Z}(0, -) \land a::\mathbb{Z}(0, -) \land b_1::(-\mathbb{Z}(0, -)) \land a < c) \lor (c::
\mathbb{Z}(0, -) \land a::\mathbb{Z}(0, -) \land b_2::(-\mathbb{Z}(0, -)) \land a < c) \\
& \lor (c::\mathbb{Z}(0, -) \land a::\mathbb{Z}(0, -) \land b_1::
\mathbb{Z}(0, -) \land a < c \land b_1 + b_2 < c) \lor (c::\mathbb{Z}(0, -) \land a::(-\mathbb{Z}(0, -)) \land b_1::(-\mathbb{Z}(0, -)) \land a:< c) \\
& \lor (c::\mathbb{Z}(0, -) \land a::(-\mathbb{Z}(0, -)) \land b_1::\mathbb{Z}(0, -) \land b_2::\mathbb{Z}(0, -) \land b_1 + b_2 < c) \]
\end{align*}
\]

$$\begin{array}{c}
> \text{FunctionAdvisor}(\text{singularities, AppellF1}) \\
\{F_1(a, b_1, b_2, c, z_1, z_2), (c::\mathbb{Z}(0, -) \land a::\mathbb{Z}(0, -) \land b_1::(-\mathbb{Z}(0, -)) \land a < c) \lor (c::\mathbb{Z}(0, -) \land a::\mathbb{Z}(0, -) \land b_2::(-\mathbb{Z}(0, -)) \land a < c) \lor (c::\mathbb{Z}(0, -) \land a::\mathbb{Z}(0, -) \land b_1::\mathbb{Z}(0, -) \land a < c \land b_1 + b_2 < c) \lor (c::\mathbb{Z}(0, -) \land a::(-\mathbb{Z}(0, -)) \land b_1::(-\mathbb{Z}(0, -)) \land a < c) \lor (c::\mathbb{Z}(0, -) \land a::(-\mathbb{Z}(0, -)) \land b_1::\mathbb{Z}(0, -) \land b_2::\mathbb{Z}(0, -) \land b_1 + b_2 < c) \}
\end{array}$$

$$%\text{AppellF1} \left( \frac{1}{2}, \frac{3}{2}, \frac{1}{4}, -2, z_1, z_2 \right)$$

$$%\text{AppellF1} \left( \frac{1}{2}, \frac{3}{2}, \frac{1}{4}, -2, z_1, z_2 \right)$$

$$\text{value}(%)$$

Error, (in \text{AppellF1:-Singularities}) numeric exception: division by zero

A polynomial example: $b_1$ and $b_2$ are negative integers

$$\begin{array}{c}
> \text{AppellF1} \left( \frac{3}{2}, -2, -1, \frac{1}{2}, z_1, z_2 \right) \\
\{F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \frac{(a)_{k_1} + (b_1)_{k_1} (b_2)_{k_2} z_1^{k_1} z_2^{k_2}}{(c)_{k_1 + k_2} k_1! k_2!} \} \\
\end{array}$$

$$\begin{array}{c}
> \text{AppellF1} \left( \frac{3}{2}, -2, -1, \frac{1}{2}, z_1, z_2 \right) \\
\{F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \frac{(a)_{k_1} + (b_1)_{k_1} (b_2)_{k_2} z_1^{k_1} z_2^{k_2}}{(c)_{k_1 + k_2} k_1! k_2!} \} \\
\end{array}$$

$$\begin{array}{c}
> \text{simplify}(\text{series}((1.2.3), z_1)) \\
1 - 3 z_2 + (-6 + 10 z_2) z_1 + (5 - 7 z_2) z_1^2 + O(z_1^6) \\
\end{array}$$

> Single Power Series with Hypergeometric Coefficients

$$\begin{array}{c}
> \text{FunctionAdvisor}(\text{definition, AppellF1}(a, b_1, b_2, c, z_1, z_2)) \\
\{F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \frac{(a)_{k_1} + (b_1)_{k_1} (b_2)_{k_2} z_1^{k_1} z_2^{k_2}}{(c)_{k_1 + k_2} k_1! k_2!} \} \\
\end{array}$$

1.2.1
\[
< 1 \land |z_2| < 1
\]

> \text{subsindets} (1.3.1), specfunc (pochhammer), expand

\[
F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \frac{(a + \_k1) \_k2 \, (b_1) \_k1 \, (b_2) \_k2 \, z_1^{-k1} z_2^{-k2}}{(c + \_k1) \_k2 \, (c) \_k1 \, k1! \, k2!} \, |z_1| < 1 \land |z_2| < 1
\] (1.3.2)

> expand ((1.3.2))

\[
F_1(a, b_1, b_2, c, z_1, z_2)
\]

\[
= \sum_{k_1=0}^{\infty} \frac{(a) \_k1 \, (b_1) \_k1 \, z_1^{-k1}}{(c) \_k1 \, k1!} \left( \sum_{k_2=0}^{\infty} \frac{(a + \_k1) \_k2 \, (b_2) \_k2 \, z_2^{-k2}}{k2!} \right)
\] (1.3.3)

\[
< 1 \land |z_2| < 1
\]

> value (%) assuming abs \((z_2) < 1\)

\[
F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1=0}^{\infty} \frac{(a) \_k1 \, (b_1) \_k1 \, z_1^{-k1}}{(c) \_k1 \, k1!} \, {}_2F_1(b_2, a + \_k1; c + \_k1; z_2)
\] (1.3.4)

\[
|z_1| < 1 \land |z_2| < 1
\]

> FunctionAdvisor \((\text{sum_form}, \text{AppellF1}(a, b_1, b_2, c, z_1, z_2))\)

\[
F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{(a) \_m + n \, (b_1) \_m \, (b_2) \_n \, z_1^{-m} z_2^{-n}}{(c) \_m + n \, m! \, n!} \, |z_1| < 1 \land |z_2| < 1
\] (1.3.5)

\[
< 1 \], \hspace{1cm} F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1=0}^{\infty} \frac{(a) \_k1 \, (b_1) \_k1 \, z_1^{-k1}}{(c) \_k1 \, k1!} \, {}_2F_1(a + k, b_2; c + k; z_2) \, |z_1|< 1 \], \hspace{1cm} F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_2=0}^{\infty} \frac{(a) \_k2 \, (b_2) \_k2 \, z_2^{-k2}}{(c) \_k2 \, k2!} \, {}_2F_1(a + k, b_1; c + k; z_1) \, |z_2|
As indicated in the formulas above, for these two Appell functions the domain of convergence of the single sum with hypergeometric coefficients is larger than the domain of convergence of the double series, because the hypergeometric coefficient in the single sum - say the one in $z_2$ - analytically extends the series with regards to the other variable - say $z_1$ - entering the hypergeometric coefficient. Hence, for $\text{AppellF1}$ and $\text{AppellF3}$, the case where one of the two variables, $z_1$ or $z_2$, is equal to 1, is convergent only when the corresponding hypergeometric coefficient in the single sum form is convergent. For instance, for $\text{AppellF1}$ the convergent case at $z_1 = 1$ requires that $0 < -\text{Re}( -c + a + b_1 )$.

The situation is different for $\text{AppellF2}$ and $\text{AppellF4}$, where the domain of convergence with regards to the two variables $z_1$ and $z_2$ is entangled, i.e. it intrinsically depends on a combination of the two variables:

\[
FunctionAdvisor(\text{sum\_form}, \text{AppellF2})
\]

\[
F_2(a, b_1, b_2, c_1, c_2, z_1, z_2) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{(a)_m (b_1)_n (b_2)_n z_1^m z_2^n}{(c_1)_m (c_2)_n m! n!}, |z_2| + |z_1| < 1
\]
\[
\sum_{k=0}^{\infty} \frac{(a)_k (b_2)_k}{(c_2)_k k!} \binom{a + k}{b_1} \binom{b + k}{c_1} z_1^k \left(1 + \frac{z_2}{z_1}\right)^k, |z_2| + |z_1| < 1
\]

> FunctionAdvisor(sum_form, AppellF4)

\[
F_4(a, b, c_1, c_2, z_1, z_2) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{(a)_m (b)_n}{(c_1)_m (c_2)_n m! n!} \left(1 + \frac{z_1}{z_2}\right)^m \left(1 + \frac{z_2}{z_1}\right)^n, |z_2| + |z_1| < 1
\]

(1.3.8)

\[
F_4(a, b, c_1, c_2, z_1, z_2) = \sum_{k=0}^{\infty} \frac{(a)_k (b)_k}{(c_1)_k k!} \binom{a + k}{b_1} \binom{b + k}{c_1} z_1^k \left(1 + \frac{z_2}{z_1}\right)^k, |z_2| + |z_1| < 1
\]

so the hypergeometric coefficient in one variable in the single sum form does not extend the domain of convergence of the double sum but for particular cases, and from the formulas above one cannot conclude about the value of the function when one of \(z_1\) or \(z_2\) is equal to 1 unless the other one is exactly equal to 0.

### Analytic Extension from the Appell Series to the Appell Functions

In the literature, the Appell series are analytically extended by integral representations in terms of Eulerian double integrals. With the exception of \texttt{AppellF4}, one of the two iterated integrals can always be computed resulting in a single integral with hypergeometric integrand. For example, for \texttt{AppellF3}

> FunctionAdvisor(integral_form, AppellF3)

\[
F_3(a_1, a_2, b_1, b_2, c, z_1, z_2)
\]

(1.4.1)

\[
\Gamma(c) \left\{ \begin{array}{l}
\frac{1}{(1 - u)^{-1 + \frac{a}{b_1}} z_1^{-\frac{1}{b_1}} F_1(a_2, b_2, c - b_1; z_2 u)} \ \text{d}u
\end{array} \right.
\]

\[
= \frac{\Gamma(b_1) \Gamma(c - b_1)}{\Gamma(b_1) \Gamma(c - b_1)}, 0 < R(b_1) \land 0
\]
In the case of AppellF4, single integral representation exists only for particular values of the function's parameters, for example two cases are
These integral representations are the starting point for the derivation of many of the identities known for the four Appell functions.

**Euler-Type and Contiguity Identities**

For the purpose of numerically evaluating the four Appell functions over the whole complex plane, instead of numerically evaluating the integral representations, it is simpler, when possible, to evaluate the function using identities. For example, with the exception of AppellF3, the Appell functions admit identities analogous to Euler identities for the hypergeometric function.
These Euler-type identities, as well as contiguity identities for the four Appell functions, are visible using the FunctionAdvisor or directly from the function. For AppellF4, for instance, provided that none of \(a, b, a - b, c_2 - a\) is a non-positive integer,

\[
\text{AppellF4}(a, b, c_1, c_2, z_1, z_2) = \text{AppellF4:-Transformations} \ "\text{Euler}\" \{1\}(a, b, c_1, c_2, z_1, z_2); \quad F_4(a, b, c_1, c_2, z_1, z_2)
\]

\[
\Gamma(c_2) \Gamma(b) \left(\frac{-z_2}{z_2}\right)^{-a} F_4(a, a - c_2 + 1, a - b + 1, c_1, \frac{1}{z_2}, \frac{z_1}{z_2}) + \frac{\Gamma(c_2) \Gamma(a) \left(\frac{-z_2}{z_2}\right)^{-b} F_4(b, 1 + b - c_2, b - a + 1, c_1, \frac{1}{z_2}, \frac{z_1}{z_2})}{\Gamma(c_2 - b)}
\]

This identity can be used to evaluate AppellF4 at \(z_1 = 1\) over the whole complex plane since, in that case, the two variables of the Appell Functions on right-hand side become equal, and that is a special case of AppellF4, expressible in terms of hypergeometric 4F3 functions

\[
\text{eval}(1.5.1, z_1 = 1); \quad F_4(a, b, c_1, c_2, 1, z_2) = \frac{1}{\Gamma(c_2 - a) \Gamma(b)} \left(\Gamma(c_2) \Gamma(b - a) \left(\frac{-z_2}{z_2}\right)^{-a} F_3(a, a - c_2 + 1, a - b + 1, c_1, \frac{1}{z_2}, \frac{z_1}{z_2})\right) + \frac{1}{\Gamma(c_2 - b) \Gamma(a)} \left(\Gamma(c_2) \Gamma(a - b) \left(\frac{-z_2}{z_2}\right)^{-b} F_3(b, 1 + b - c_2, b - a + 1, c_1, \frac{1}{z_2}, \frac{z_1}{z_2})\right)
\]

A contiguity transformation for AppellF4

\[
\text{AppellF4}(a, b, c_1, c_2, z_1, z_2) = \text{AppellF4:-Transformations} \ "\text{Contiguity}\" \{2\}(a, b, c_1, c_2, z_1, z_2); \quad F_4(a, b, c_1, c_2, z_1, z_2) = F_4(b, a + n, c_1, c_2, z_1, z_2)
\]

\[
\frac{b z_1 \left(\sum_{k=1}^{n} F_4(a + k, b + 1, c_1 + 1, c_2, z_1, z_2)\right)}{c_1}
\]
\[
- \frac{b z_2 \left( \sum_{k=1}^{\infty} F_4 \left( a + k, b + 1, c_1, 1 + c_2, z_1, z_2 \right) \right)}{c_2}
\]

### Appell Differential Equations

Each of the four Appell functions satisfy a linear system of partial differential equations, for example for \( \text{AppellF1} \):

\[
\begin{align*}
& f(a, b_1, b_2, c, z_1, z_2) = F_1(a, b_1, b_2, c, z_1, z_2), \quad \frac{\partial^2}{\partial z_1^2} f(a, b_1, b_2, c, z_1, z_2) = \\
& z_2 \left( \frac{\partial^2}{\partial z_1^2} f(a, b_1, b_2, c, z_1, z_2) \right) - z_1 \\
& \quad \left( -a - b_1 - 1 \right) z_1 + c \left( \frac{\partial}{\partial z_1} f(a, b_1, b_2, c, z_1, z_2) \right) \quad \frac{\partial^2}{\partial z_1 \partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad a b_1 \quad \frac{\partial^2}{\partial z_1^2} f(a, b_1, b_2, c, z_1, z_2) \quad z_1 \left( z_1 - 1 \right) \\
& b_1 z_2 \left( \frac{\partial}{\partial z_1} f(a, b_1, b_2, c, z_1, z_2) \right) - z_1 \left( z_1 - 1 \right) \\
& \quad \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad a b_1 \quad \frac{\partial^2}{\partial z_1 \partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad z_2 \left( z_1 - 1 \right) \\
& b_2 \left( \frac{\partial}{\partial z_1} f(a, b_1, b_2, c, z_1, z_2) \right) \\
& \quad \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad a b_2 \quad \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \quad z_2 \left( z_2 - 1 \right)
\end{align*}
\]

By using differential algebra techniques, this PDE system, as well as the ones corresponding to each of the other Appell functions, can be transformed into an equivalent PDE system where one of the equations is a linear ODE in
In the case of \textit{AppellF1}, this linear ODE is of third order and can be computed as follows

\[
F1(z_1, z_2) = \text{AppellF1}(a, \, b_1, \, b_2, \, c, \, z_1, \, z_2);
\]

\[
F1(z_1, z_2) = F_1(a, \, b_1, \, b_2, \, c, \, z_1, \, z_2);
\]

\[
\frac{\partial^3}{\partial z_2^3} F1(z_1, z_2) = \frac{1}{z_2^2} \frac{1}{(z_2 - 1)} \left( \frac{\partial^2}{\partial z_2^2} F1(z_1, z_2) \right) + \left( -3 \right) z_1 - c - b_2 - 2 \right) z_2 + z_1 \left( c - b_1 + 1 \right) \left( \frac{\partial}{\partial z_2} F1(z_1, z_2) \right) + \left( \frac{\partial}{\partial z_2} F1(z_1, z_2) \right) a b_2
\]

This is a linear ODE with four regular singularities, one of which is located at \( z_2 = z_1 \).

When applying the same procedure to the other Appell functions, the result is a fourth order linear ODE with singularities of increasing complexity. The singularities of those fourth order linear ODES behind AppellF2, AppellF3 and AppellF4 can be viewed directly using the \textbf{Singularities} command of the \textbf{MathematicalFunctions:-Evalf} package; for instance for \textit{AppellF4} the singularities of the underlying ODE are

\[
F2(\, a, \, b_1, \, b_2, \, c_1, \, c_2, \, z_1, \, z_2) \]

\[
F2(\, a, \, b_1, \, b_2, \, c_1, \, c_2, \, z_1, \, z_2) \]
\[-9) b_2 - a^2 + (c_1 - 9) a + 4 c_1 - 14 \right) z_2^2 + \left( \left( -z_1 + 2 \right) a + \left( -z_1 + 2 \right) c_2 
+ \left(b_1 - 2\right) z_1 - c_1 + 4 \right) b_2 - \left(z_1 - 2 \right) \left( c_2 + 2 \right) a + \left(b_1 - 3\right) z_1 - c_1 
+ 6 \right) c_2 + \left( 2 b_1 - 4 \right) z_1 - 2 c_1 + 8 \right) z_2 + c_2 \left( z_1 - 1 \right) \left( 1 + c_2 \right) \right) \left( \frac{\partial^{2}}{\partial z_2^2} \right) 
F^2(z_1, z_2) - \left( \left( 2 a - c_1 + 2 \right) b_2 + 2 \left( a + 1 \right) \left(a - c_1 + 2 \right) \right) z_2 + \left( z_1 
- 2 \right) a + \left(-b_1 + 1 \right) z_1 + c_1 - 2 \right) c_2 \right) \left( \frac{\partial}{\partial z_2} F^2(z_1, z_2) \right) + F^2(z_1, z_2) a b_2 \left(a 
- c_1 + 1 \right) \right) \left( b_2 + 1 \right) \right] 
\]

\textbf{MathematicalFunctions}\:-\textbf{Evalf}\:-\textbf{Singularities} (\textbf{AppellF4}(a, b, c_1, c_2, z_1, z_2)); 
\[0, \frac{\left(z_1 - 1 \right) \left(a + b - c_1 + 1 \right) \left(a + b - c_1 - 2 c_2 + 3 \right)}{\left(c_1 - 1 - b + a \right) \left(-c_1 + 1 - b + a \right)}, z_1 + 1 - 2 \sqrt{z_1}, z_1 + 1 \] 
\[+ 2 \sqrt{z_1}, \infty + \infty \right] \quad (1.6.8) \]

\section*{Putting all together}

\[> \text{FunctionAdvisor} (\text{AppellF1}); \]

\section*{AppellF1}

\subsection*{describe}

\textit{AppellF1} = Appell 2-variable hypergeometric function F1

\subsection*{definition}

\[F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k_1 = 0}^{\infty} \sum_{k_2 = 0}^{\infty} \frac{\left(a\right)_{k_1 + k_2} \left(b_1\right)_{k_1} \left(b_2\right)_{k_2} z_1^{k_1} z_2^{k_2}}{\left(c\right)_{k_1 + k_2} \left(-k_1 \right)! \left(-k_2 \right)!} \quad |z_1| < 1 \wedge |z_2| < 1 \]

\subsection*{classify function}

\textit{Appell}

\subsection*{symmetries}
\[ F_1(a, b_2, b_1, c, z_2, z_1) = F_1(a, b_1, b_2, c, z_1, z_2) \]

\textbf{plot}

\textit{Appell 2-variable hypergeometric function} $F1$

\[ F_1 \left( \frac{-13}{2}, 5, 12, \frac{17}{9}, -\frac{3}{10}, x \right) \]

\[ F_1 \left( \frac{-16}{5}, -\frac{5}{2}, \frac{7}{2}, 1, x, \frac{3}{10} \right) \]

\[ F_1 \left( \frac{16}{5} + 2I, -3, \frac{51}{10}, \frac{17}{10}, -2 + \frac{1}{10}, z \right) \]

\[ F_1 \left( \frac{1}{5} - \frac{2I}{5}, 5, 7, -\frac{3I}{4}, -\frac{7}{10}, z \right) \]

\textbf{singularities}
\[ F_1(a, b_1, b_2, c, z_1, z_2) = (c: \mathbb{Z}^{0,-}) \land a::\mathbb{Z}^{0,-} \land b_1::(- \mathbb{Z}^{0,-}) \land a < c \land b_2::(- \mathbb{Z}^{0,-}) \land a < c \land b_1 + b_2 < c \lor (c::\mathbb{Z}^{0,-}) \land a::(- \mathbb{Z}^{0,-}) \land b_2::\mathbb{Z}^{0,-} \land b_1::(- \mathbb{Z}^{0,-}) \land b_2::\mathbb{Z}^{0,-} \land b_1::\mathbb{Z}^{0,-} \land b_2::\mathbb{Z}^{0,-} \land b_1 + b_2 < c) \]

### branch points

\[ F_1(a, b_1, b_2, c, z_1, z_2) = (c::(- \mathbb{Z}^{0,-}) \land b_1::(- \mathbb{Z}^{0,-}) \land z_1 \in [1, \infty + \infty 1]) \lor (c::(- \mathbb{Z}^{0,-}) \land b_2::(- \mathbb{Z}^{0,-}) \land z_2 \in [1, \infty + \infty 1]) \]

### branch cuts

\[ F_1(a, b_1, b_2, c, z_1, z_2) = (a::(- \mathbb{Z}^{0,-}) \land b_1::(- \mathbb{Z}^{0,-}) \land 1 < z_1 \lor (a::(- \mathbb{Z}^{0,-}) \land b_2::(- \mathbb{Z}^{0,-}) \land 1 < z_2) \]

### special values
\[ F_1(a, b_1, b_2, c, z_1, z_2) = 1 \quad \text{for } z_1 = 0 \land z_2 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = 1 \quad a = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = 1 \quad b_1 = 0 \land b_2 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_2; c; z_2) \quad \text{for } z_1 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_2; c; z_2) \quad b_1 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_1; c; z_1) \quad z_2 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_1; c; z_1) \quad b_2 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_1; c; 1) \quad z_1 = 1 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_1; c; 1) \quad b_2 = 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_1; c; 1) \quad z_2 = 1 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _2F_1(a, b_1; c; z_1) \quad z_1 = z_2 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _3F_2 \left( b_1; \frac{a}{2}, \frac{a}{2} + \frac{1}{2}; c; \frac{a}{2} + \frac{1}{2}; \text{for } z_1 = -z_2 \land b_1 = b_2 \right) \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = _1F_0(b_1; z_1) \cdot _1F_0(b_2; z_2) \quad c = a \land a \neq 0 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = \frac{z_1}{z_2} _2F_1(1, a; c; z_1) - z_2 _2F_1(1, a; c; z_2) \quad \text{for } b_1 = 1 \land b_2 = 1 \land z_1 \neq z_2 \]
\[ F_1(a, b_1, b_2, c, z_1, z_2) = (1 - z_1)^{-b_1} (1 - z_2)^{-b_2} F_1 \left( c - a, b_1, b_2, c, z_1, z_2 \right) \]

\[ z_1 \neq 1 \land z_2 \neq 1 \]
\[ \land (a::Z^{(0,-)}) \]
\[ \lor (b_1::Z^{(0,-)}) \]
\[ \lor (b_2::Z^{(0,-)}) \]
\[ \lor (1 < z_1 \land 1 < z_2) \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = (1 - z_1)^{-a} F_1 \left( a, c - b_1 - b_2, b_1, b_2, c, z_1, z_2 \right) \]

\[ z_1 \neq 1 \land (a::Z^{(0,-)}) \]
\[ \lor (b_1::Z^{(0,-)}) \]
\[ \lor (b_2::Z^{(0,-)}) \]
\[ \lor (1 < z_1 \land 1 < z_2) \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = (1 - z_1)^{c - a - b_1} (1 - z_2)^{-b_2} F_1 \left( c - a, c - b_1 - b_2, b_1, b_2, z_1, z_2 \right) \]

\[ z_2 \neq 1 \land z_1 \neq 1 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = \frac{1}{(z_1 - z_2)(-1 + a)} (-1 + c) \left( F_1(-1 + a, b_1, b_2 - 1, -1 + c, z_1, z_2) - F_1(-1 + a, b_1 - 1, b_2, -1 + c, z_1, z_2) \right) \]

\[ z_1 \neq z_2 \land a \neq 1 \]
\[ \land c \neq 1 \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = \frac{(a)_n F_1(n + a, b_1, b_2, c, z_1, z_2)}{(a - c + 1)_n} \]
\[ - \left( \sum_{k=1}^{n} \frac{1}{(a - c + 1)_k} (-1)^k \binom{n}{k} (1 - c)_k F_1(a, b_1, b_2, c - k, z_1, z_2) \right) \]

\[ c \neq 1 \land (\{a - c + 1\}::(\}
\[ \lor \neg Z^{(0,-)}) \lor n \]
\[ \leq |a - c + 1|) \]

\[ F_1(a, b_1, b_2, c, z_1, z_2) \]
\[ = \frac{1}{-b_1 - b_2 + a} \left( a F_1(a + 1, b_1, b_2, c, z_1, z_2) - F_1(a, b_1 + 1, b_2, c, z_1, z_2) \right) \]
\[ a \neq b_1 + b_2 \]
\[
\sum_{m=0}^{\infty} \frac{(a)_{m+n} (b_1)_m (b_2)_n z_1^m z_2^n}{(c)_{m+n} m! n!} \]

\[
F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k=0}^{\infty} \frac{1}{(c)_k k!} \left( (a)_k \left( b_1 \right)_k F_1(a + k, b_2; c + k; z_1^k) + b_1 \right) \]

\[
F_1(a, b_1, b_2, c, z_1, z_2) = \sum_{k=0}^{\infty} \frac{1}{(c)_k k!} \left( (a)_k \left( b_2 \right)_k F_1(a + k, b_1; c + k; z_2^k) + b_2 \right) \]
\[ \text{series} \left( F_1(a, b_1, b_2, c, z_1, z_2), z_1, 4 \right) = _2 F_1(a, b_2; c, z_2) \]
\[ + \frac{a b_1}{c} _2 F_1(b_2, a + 1; 1 + c; z_2) z_1 \]
\[ + \frac{1}{2} \frac{a b_1 (a + 1) (b_1 + 1)}{c (1 + c)} _2 F_1(b_2, a + 2; 2 + c; z_2) z_1^2 \]
\[ + \frac{1}{6} \frac{a b_1 (a + 1) (b_1 + 1) (a + 2) (b_1 + 2)}{c (1 + c) (2 + c)} _2 F_1(b_2, a + 3; 3 + c; z_2) z_1^3 \]
\[ + O(z_1^4) \]

\[ \text{series} \left( F_1(a, b_1, b_2, c, z_1, z_2), z_2, 4 \right) = _2 F_1(a, b_1; c, z_1) \]
\[ + \frac{a b_2}{c} _2 F_1(b_1, a + 1; 1 + c; z_1) z_2 \]
\[ + \frac{1}{2} \frac{a b_2 (a + 1) (b_2 + 1)}{c (1 + c)} _2 F_1(b_1, a + 2; 2 + c; z_1) z_2^2 \]
\[ + \frac{1}{6} \frac{a b_2 (a + 1) (b_2 + 1) (a + 2) (b_2 + 2)}{c (1 + c) (2 + c)} _2 F_1(b_1, a + 3; 3 + c; z_1) z_2^3 \]
\[ + O(z_2^4) \]

integral form
\[ F_1(a, b_1, b_2, c, z_1, z_2) = \begin{cases} \Gamma(c) \end{cases} \]

\[
\begin{aligned}
0 &< \Re(b_1) \land 0 < -\Re(-c + b_1) \\
&= \frac{1}{-c + 1 + b_1} \left( (1 - u)^{-1 + b_1} (-u z_1 + 1)^{-c + a} \right) \\
&\quad \left( \Gamma(b_1) \Gamma(c - a + b_1) \right)
\end{aligned}
\]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = \begin{cases} \Gamma(c) \end{cases} \]

\[
\begin{aligned}
0 &< \Re(b_2) \land 0 < -\Re(-c + b_2) \\
&= \frac{1}{-c + 1 + b_2} \left( (1 - u)^{b_2 - 1} (-z_2 u + 1)^{-c + a} \right) \\
&\quad \left( \Gamma(b_2) \Gamma(c - a + b_2) \right)
\end{aligned}
\]

\[ F_1(a, b_1, b_2, c, z_1, z_2) = \frac{1}{\Gamma(a) \Gamma(c - a)} \begin{cases} \Gamma(c) \end{cases} \]

\[
\begin{aligned}
0 &< \Re(a) \land 0 < -\Re(-c + a) \\
&= \frac{1}{u^a - 1} \left( \Gamma(c) \right)
\end{aligned}
\]
**differentiation rule**

\[
\frac{\partial}{\partial z_1} F_1(a, b_1, b_2, c, z_1, z_2) = \frac{ab_1 F_1(a + 1, b_1, b_2, 1 + c, z_1, z_2)}{c}
\]

\[
\frac{\partial^n}{\partial z_1^n} F_1(a, b_1, b_2, c, z_1, z_2) = \frac{(a) \left(\frac{b_1}{n}\right) F_1(n + a, n + b_1, b_2, n + c, z_1, z_2)}{(c)_n}
\]

\[
\frac{\partial}{\partial z_2} F_1(a, b_1, b_2, c, z_1, z_2) = \frac{ab_2 F_1(a + 1, b_1, b_2, 1 + 1 + c, z_1, z_2)}{c}
\]

\[
\frac{\partial^n}{\partial z_2^n} F_1(a, b_1, b_2, c, z_1, z_2) = \frac{(a) \left(\frac{b_2}{n}\right) F_1(n + a, b_1, n + b_2, n + c, z_1, z_2)}{(c)_n}
\]
\[
\frac{\partial^2}{\partial z_j^2} f(a, b_1, b_2, c, z_1, z_2) = \frac{1}{z_j} \left( z_2 \right) \\
- \frac{1}{z_j} \left( z_2 \right) \\
+ \left( \left( -a - b_j - 1 \right) z_j + c \right) \left( \frac{\partial}{\partial z_1} f(a, b_1, b_2, c, z_1, z_2) \right) \bigg/ \left( z_1 (z_1 - 1) \right) \\
- \left( b_j z_2 \left( \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \right) \right) \bigg/ \left( z_1 (z_1 - 1) \right) \\
- (f(a, b_1, b_2, c, z_1, z_2) a b_j) \bigg/ \left( z_1 (z_1 - 1) \right) \\
\frac{\partial^2}{\partial z_j^2} f(a, b_1, b_2, c, z_1, z_2) = \frac{1}{z_j} \left( z_2 \right) \\
- \frac{1}{z_j} \left( z_2 \right) \\
+ \left( \left( -a - b_j - 1 \right) z_j + c \right) \left( \frac{\partial}{\partial z_1} f(a, b_1, b_2, c, z_1, z_2) \right) \bigg/ \left( z_1 (z_1 - 1) \right) \\
- \left( b_j z_2 \left( \frac{\partial}{\partial z_2} f(a, b_1, b_2, c, z_1, z_2) \right) \right) \bigg/ \left( z_1 (z_1 - 1) \right) \\
- (f(a, b_1, b_2, c, z_1, z_2) a b_j) \bigg/ \left( z_1 (z_1 - 1) \right)
\]

\[f(a, b_1, b_2, c, z_1, z_2) = F_1(a, b_1, b_2, c, z_1, z_2)\]

\[\text{Problem: some formulas in the literature are wrong or miss the conditions indicating when are they valid (exchange with the}\]
From the five formulas above three are wrong in that they are valid only sometimes. These problems became evident after we developed the numerical evaluation. The following is an excerpt of an exchange we had with the Mathematics Editor of the DLMF (Digital Library of Mathematical Functions)

$ \texttt{restart;}$

This is Appell F1

From the symmetry of this function, swap $b l \leftrightarrow b 2$ and $z 1 \leftrightarrow z 2$ does not change the value of the function

According to NIST page 414, 16.16.1, with no restrictions, if $c = b 1 + b 2$ we have

So, applying 16.16.1 to F1 and SF1 we get

\begin{align*}
\text{e1} & := \text{formula(op(F1))} \\
\text{e2} & := \text{formula(op(SF1))}
\end{align*}
and because $F_1 = SF_1$, we expect that $e_1 = e_2$.

Give some values to the parameters ... and recall we are evaluating only hypergeometric functions

$$\text{val} := [a, b_1, b_2, c, z_1, z_2] = \left[ \frac{17}{5}, \frac{37}{10}, -\frac{11}{5}, \frac{3}{2}, 2 - I, 2 + 4I \right]$$

$$\text{val} := \left[ a = \frac{17}{5}, b_1 = \frac{37}{10}, b_2 = -\frac{11}{5}, c = \frac{3}{2}, z_1 = 2 - I, z_2 = 2 + 4I \right]$$  \hspace{1cm} (1.8.6)

$$\text{evalf} \left( \text{subs} \left( \text{val}, e_1 \right) \right) = 13.79273401 - 9.715912626I$$ \hspace{1cm} (1.8.7)

$$\text{evalf} \left( \text{subs} \left( \text{val}, e_2 \right) \right) = -16.86942635 - 0.2468272677I$$ \hspace{1cm} (1.8.8)

In other words, "the special value of $F_1$ is unexpectedly different from the special value of $SF_1$ even when these two are the same function". What happened?

We were expecting $e_1 = e_2$ so that

$$\text{zero} := e_1 - e_2 = 0$$

$$\text{zero} := (1 - z_2)^{-a} \text{$_2F_1$} \left( a, b_1; b_1 + b_2; \frac{z_1 - z_2}{1 - z_2} \right) - (1 - z_1)^{-a} \text{$_2F_1$} \left( a, b_2; b_1 + b_2; \frac{z_2 - z_1}{1 - z_1} \right) = 0$$  \hspace{1cm} (1.8.9)

We know at this point that $\text{zero}$ is not actually 0 but

$$\text{evalf} \left( \text{subs} \left( \text{zero}, \text{val}, \text{zero} \right) \right) = 30.66216036 - 9.469085358I = 0.$$  \hspace{1cm} (1.8.10)

Change variables from $\{z_1, z_2\}$ to some $\{x, y\}$

$$x = 1 - z_1, y = 1 - z_2$$  \hspace{1cm} (1.8.11)

$$\text{solve} \left( \% \right) = \{z_1 = -x + 1, z_2 = -y + 1\}$$  \hspace{1cm} (1.8.12)

Substitute this in $\text{zero}$

$$\text{subs} \left( \% \right) = y^{-a} \text{$_2F_1$} \left( a, b_1; b_1 + b_2; \frac{-x + y}{y} \right) - x^{-a} \text{$_2F_1$} \left( a, b_2; b_1 + b_2; \frac{-y + x}{x} \right) = 0$$  \hspace{1cm} (1.8.13)

Now substitute $y = -x \cdot z + x$ in order to have a simpler argument for the $2F_1$ function

$$\text{subs} \left( y = -x \cdot z + x, \% \right) = (-x + z + x)^{-a} \text{$_2F_1$} \left( a, b_1; b_1 + b_2; \frac{x z}{-x z + x} \right) - x^{-a} \text{$_2F_1$} \left( a, b_2; b_1 + b_2; z \right) = 0$$  \hspace{1cm} (1.8.14)

Isolate the simplest $2F_1$

$$\text{isolate} \left( \text{normal} \left( \% \right), \text{$_2F_1$} \left( a, b_2; b_1 + b_2; z \right) \right) = \frac{(-x + z) x^{-a} \text{$_2F_1$} \left( a, b_1; b_1 + b_2; \frac{z}{z - 1} \right)}{x^{-a}}$$  \hspace{1cm} (1.8.15)

Collect $x$ in the power
> \((-xz + x) = x \cdot (1 - z)\)

\[-xz + x = x(1 - z)\]  
(1.8.16)

> \(ee := \text{subs)((1.8.16), (1.8.15))\)

\[ee := \binom{2F_1(a, b2; b1 + b2; z)}{x^a} \frac{(x(1 - z))^{-a}}{2F_1\left(a, b1; b1 + b2; \frac{z}{z - 1}\right)}\]  
(1.8.17)

And by inspection we arrived at something we can check in NIST: this is formula 15.8.1 on page 390, provided that we can split

\[(x(1 - z))^{-a} = (1 - z)^{-a}x^{-a}\]

The formula for distributing powers, however, involves an exponential not present in the above and that is not always equal to 1:

> \((AB)^\alpha = A^\alpha B^\alpha e^{2\pi i \frac{\pi - \arg(A) - \arg(B)}{2\pi}}\)

\[\frac{2\pi a}{2\pi} \frac{\pi - \arg(A) - \arg(B)}{2\pi}\]  
(1.8.18)

In our case, the dummies \(A, B, \alpha\) are

> \(\text{subs}(A = x, B = (1 - z), \alpha = -a, (1.8.18))\)

\[(x(1 - z))^{-a} = x^{-a} (1 - z)^{-a} e^{2\pi i}\]  
(1.8.19)

So, substitute this value in \(ee\)

> \(ee1 := \text{subs}(1.8.19), ee\)

\[ee1 := \binom{2F_1(a, b2; b1 + b2; z)}{1 - z2} \frac{(1 - z)^{-a}}{2F_1\left(a, b1; b1 + b2; \frac{z}{z - 1}\right)}\]  
(1.8.20)

So "For 16.13.1 in NIST-DLMF to be correct, the exponential above should be equal to 1"

Let's check the value of the exponential above for the values we used in (1.8.6). For that purpose express \(x, y\) and \(z\) in terms of the original \(z1\) and \(z2\) variables of the Appell function

> \(x = 1 - z1, y = 1 - z2, y = -xz + x\)

\[x = 1 - z1, y = 1 - z2, y = -xz + x\]  
(1.8.21)

> \(\text{solve}(%); \{x, y, z\}\)

\[\{x = 1 - z1, y = 1 - z2, z = \frac{z1 - z2}{-1 + z1}\}\]  
(1.8.22)

> \(\text{subs}((1.8.22), ee1)\)

> \(ee2 := \text{simplify}(%\rangle\)

\[ee2 := \binom{2F_1\left(a, b2; b1 + b2; \frac{z1 - z2}{-1 + z1}\right)}{x^a} \frac{\pi - \arg(1 - z1) - \arg\left(-1 + z2 \right)}{2\pi} \frac{\pi - \arg(1 - z1) - \arg\left(-1 + z2 \right)}{2\pi}\]  
(1.8.23)
Check now the value of this exponential, if the NIST textbook formula were correct, this value should be equal to 1

$$E := \text{indets}((1.8.23), '\exp(anything'))[1]$$

$$E := e^{-2 \pi a \frac{\pi - \text{arg}(1 - z1) - \text{arg}\left(\frac{-1 + z2}{-1 + z1}\right)}{2 \pi}}$$  

(1.8.24)

$$> \text{subs}(\text{val}, E)$$

$$e^{-\frac{341 \pi}{5} \frac{\pi - \text{arg}(-1 + 1) - \text{arg}\left(\frac{-3}{2} + \frac{51}{2}\right)}{2 \pi}}$$  

(1.8.25)

$$> \text{evalf}(\%)$$

$$-0.8090169945 + 0.5877852522 \ i$$  

(1.8.26)

It is not equal to 1. We conclude that $e1 \neq e2$, the formula NIST 16.13.1 is not correct for these values, and in fact we see this value (1.8.26) is precisely the ratio between $e1$ and $e2$ when evaluated at the numbers provided in (1.8.6)

$$> e1$$

$$(1 - z2)^{-a} \ {}_2 F_1\left(a, b1; b1 + b2; \frac{z1 - z2}{1 - z2}\right)$$  

(1.8.27)

$$> e2$$

$$(1 - z1)^{-a} \ {}_2 F_1\left(a, b2; b1 + b2; \frac{z2 - z1}{1 - z1}\right)$$  

(1.8.28)

$$> \text{subs}(\text{val}, \frac{e1}{e2})$$

$$\frac{\cdot \ {}_2 F_1\left(\frac{17}{5}, \frac{37}{10}; \frac{3}{2}; \frac{20}{17} + \frac{51}{17}\right)(-1 + 1)^{17/5}}{(-1 - 41)^{17/5} \ {}_2 F_1\left(\frac{17}{5}, \frac{-11}{5}; \frac{3}{2}; \frac{5}{2} - \frac{51}{2}\right)}$$  

(1.8.29)

$$> \text{evalf}(\%)$$

$$-0.8090169934 + 0.5877852558 \ i$$  

(1.8.30)

In conclusion, formula 16.13.1 on page 413 of the DLMF misses stating that the formula is valid only when the exponential is equal to 1

$$> E = 1$$

$$e^{-2 \pi a \frac{\pi - \text{arg}(1 - z1) - \text{arg}\left(\frac{-1 + z2}{-1 + z1}\right)}{2 \pi}} = 1$$  

(1.8.31)

A similar situation happens with the other two hypergeometric special values of AppellF2 and AppellF3, and in fact with several Appell formulas we got from the literature. Detecting the incorrect formulas in the literature and deriving a correction for them was the main source of slowdown in the project.
Appell Functions (numeric)

Goals

- Compute these Appell functions over the whole complex plane

- Considering that this is a research problem, implement different methods and flexible optional arguments to allow for:
  
a) comparison between methods (both performance and correctness),

  b) investigation of a single method in different circumstances.

- Develop a computational structure that can be reused with other special functions (abstract code and provide the main options), and that could also be translated to C (so: only one numerical implementation, not 100 special function numerical implementations)

Limitation: the Maple original evalf command does not accept optional arguments

The cost of numerically evaluating an Appell function

- If it is a special hypergeometric case, then between 1 to 2 hypergeometric functions

- Next simplest case (series/recurrence below) 3 to 4 hypergeometric functions plus adding somewhat large formulas that involve only arithmetic operations up to 20,000 times (frequently less than 100 times)

- Next simplest case: the formulas themselves are power series with hypergeometric function coefficients; these cases frequently converge rapidly but may involve the numerical evaluation of up to hundreds of hypergeometric functions to get the value of a single Appell function.

Strategy for the numerical evaluation of Appell functions (or other functions ...)

The numerical evaluation flows orderly according to:

1) check whether it is a singular case
2) check whether it is a special value
3) compute the value using a series derived from a recurrence related to the underlying ODE
4) perform an sum using an infinite sum formula, checking for convergence
5) perform the numerical integration of the ODE underlying the given Appell function
6) perform a sequence of concatenated Taylor series expansions

Examples

```
> restart; with(MathematicalFunctions, Evalf); with(Evalf);
`Evalf`
```

(2.1.1.1)
\{Add, Evalb, Zoom, QuadrantNumbers, Singularities, GenerateRecurrence, PairwiseSummation\}

\[\%\text{AppellF1}\left(-4, -\frac{6}{5} + 1, -\frac{3}{10}, -3, \frac{1}{2}, 1 + \frac{3}{5}\right)\]

\[F_1\left(-4, -\frac{6}{5} + 1, -\frac{3}{10}, -3, \frac{1}{2}, 1 + \frac{3}{5}\right)\]

\[\text{Evalf(}\%\text{)}\]

case: abs(z1) < 1, abs(z2) > 1; swapping parameters and z1

\[\rightarrow\text{ Numerical evaluation of AppellF1(-4., -.3000000000*I, -1.200000000+1.*I,}\]

\[-3., 1.+ .6000000000*I, .5000000000)\]

\[\rightarrow\text{ AppellF1, checking singular cases}\]

\[\leftarrow\text{ AppellF1, singular case detected}\]

\[\text{CPU time elapsed during evaluation: .7e-2 seconds}\]

\[\text{Float(}\infty\text{) + Float(}\infty\text{) I}\]

\[\text{AppellF4}\left(-4, -\frac{6}{5} + 1, -\frac{3}{10}, \frac{1}{2}, \frac{1}{2}, 1 + \frac{3}{5}\right)\]

\[F_4\left(-4, -\frac{6}{5} + 1, -\frac{3}{10}, \frac{1}{2}, \frac{1}{2}, 1 + \frac{3}{5}\right)\]

\[\text{Evalf(}\%\text{)}\]

case: abs(z1) < 1, abs(z2) > 1; swapping parameters and z1

\[\rightarrow\text{ AppellF4, checking singular cases}\]

\[\rightarrow\text{ AppellF4, trying special values}\]

\[\leftarrow\text{ special values of AppellF4 successful}\]

\[\text{CPU time elapsed during evaluation: .67e-1 seconds}\]

\[\text{F3 := AppellF3}\left(-\frac{4}{3}, -\frac{6}{5} + 1, -\frac{3}{10}, \frac{1}{2}, \frac{1}{2}, 1 + \frac{3}{5}, \frac{1}{10}\right)\]

\[F_3 := F_3\left(\frac{1}{2}, -\frac{4}{3}, -\frac{6}{5} + 1, -\frac{3}{10}, \frac{1}{2}, \frac{1}{2}, 1 + \frac{3}{5}\right)\]

\[\text{Evalf(F3)}\]

case: abs(z1) < 1, abs(z2) > 1; swapping parameters and z1

\[\rightarrow\text{ Numerical evaluation of AppellF3(-1.333333333, .5000000000*I, -3.0000000000*I, -1.200000000+1.*I, .5000000000, 1.+ .6000000000*I, .1000000000)}\]

\[\rightarrow\text{ AppellF3, checking singular cases}\]

\[\rightarrow\text{ AppellF3, trying series based on recurrence}\]

\[\text{CPU time elapsed during evaluation: .118 seconds}\]

\[0.5978088090 + 0.5841674146 I\]

\[\text{Evalf(F3, formulas)}\]

case: abs(z1) < 1, abs(z2) > 1; swapping parameters and z1

\[\rightarrow\text{ Numerical evaluation of AppellF3(-1.333333333, .5000000000*I, -3.0000000000*I, -1.200000000+1.*I, .5000000000, 1.+ .6000000000*I, .1000000000)}\]
.5000000000*I, -.3000000000*I, -1.200000000+1.*I, .5000000000+1.*I, 1.+.6000000000*I, .1000000000)

-> AppellF3, checking singular cases
-> AppellF3, trying formulas
case 1: abs(z2) < 1, using single sum with
hypergeometric coefficients
-> entering Add with: 'formula' = (-1)^k*
pochhammer(-1.333333333,k)*pochhammer(.5000000000*I,k)*
pochhammer(-.3000000000*I,k)*pochhammer(-1.200000000+1.*I, k)/k!/pochhammer(-.5000000000*I+k,k)/pochhammer(.5000000000, 2*k)*hypergeom([-1.333333333+k, -.3000000000+k], [-.5000000000+k,k], .5000000000+2*k)*hypergeom(.5000000000*I+k, -1.200000000+1.*I+k), [.5000000000+2*k], [.1000000000+1.*I+k], .5000000000+1.*I+k), .1000000000+1.*I+k]^k
<- exiting Add with .597808808973+.584167414632*I; after adding 6 terms

CPU time elapsed during evaluation: .263 seconds

0.5978088090 + 0.5841674146 I

(2.1.1.8)

> Evalf(F3, differentialequation)
Error, (in MathematicalFunctions:-Evalf) invalid input: too many and/or wrong type of arguments passed to EvalfAppell:-Evalf; first unused type is differentialequation

> Evalf(F3, allowswapping = false, taylor, plot, time, quiet)
> evalf(%appellf3(2.,1.,1.,-1.500000000, 10.58823529, 60., .4000000000-.2000000000*I))
> -> AppellF3, checking singular cases
> -> AppellF3, trying series based on recurrence
> 
> CPU time elapsed during evaluation: .2e-2 seconds
> 0.1720188468 - 0.06454452081 I

Do not allow swapping, it gets tackled solving the differential equation
> evalf(F33, allowswapping = false)
> -> Numerical evaluation of AppellF3(1., 2., -1.500000000, 1., 10.58823529, .4000000000-.2000000000*I, 60.)
> -> AppellF3, checking singular cases
> -> AppellF3, trying special values
> -> AppellF3, trying formulas
AppellF3, trying the numerical integration of the related ODE
computing initial conditions: all of AppellF3, AppellF3', AppellF3'' and AppellF3''' at z2 = 0.6717514421

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(2., 1., 1., -1.500000000, 10.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(3., 1., 2., -1.500000000, 11.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(4., 1., 3., -1.500000000, 12.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(5., 1., 4., -1.500000000, 13.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(6., 1., 5., -1.500000000, 14.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(7., 1., 6., -1.500000000, 15.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(8., 1., 7., -1.500000000, 16.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(9., 1., 8., -1.500000000, 17.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(10., 1., 9., -1.500000000, 18.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(11., 1., 10., -1.500000000, 19.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(12., 1., 11., -1.500000000, 20.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(13., 1., 12., -1.500000000, 21.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(14., 1., 13., -1.500000000, 22.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(15., 1., 14., -1.500000000, 23.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(16., 1., 15., -1.500000000, 24.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(17., 1., 16., -1.500000000, 25.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(18., 1., 17., -1.500000000, 26.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(19., 1., 18., -1.500000000, 27.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

case: abs(z1) < abs(z2) < 1; swapping parameters and z1 <-> z2

Numerical evaluation of AppellF3(20., 1., 19., -1.500000000, 28.58823529, 0.6717514421, 0.4000000000 -.2000000000*I)

AppellF3, checking singular cases
AppellF3, trying special values
AppellF3, trying series based on recurrence

CPU time elapsed during evaluation: 3.825 seconds

Evalf (F33, taylor, allowswapping = false, plot, time, quiet)


\[ \text{CPU time elapsed during evaluation: \(71\times10^{-1}\) seconds} \]

\[-0.1720188468 - 0.06454452081\, I \]

\[ \text{Singularities (F33)} \]

\[ \begin{bmatrix} 0, -\frac{1}{2} + \frac{1}{2}, 1, \infty + \infty \end{bmatrix} \]

\[ \text{evalf (abs (%[2]))} \]

\[ 0.7071067810 \]

\[ \text{Zoom (\([%], \frac{1}{2}\])} \]
Series/recurrence

Derivation of formulas for AppellF2

The underlying ODE is obtained using a differential algebra process on the linear PDE system satisfied by AppellF2

```maple
restart; with(MathematicalFunctions:-Evalf)
{Add, Evalb, Zoom, QuadrantNumbers, Singularities, GenerateRecurrence, PairwiseSummation}

> b1, b2, c1, c2, z1, z2 := b_1, b_2, c_1, c_2, z_1, z_2
    b1, b2, c1, c2, z1, z2 := b_1, b_2, c_1, c_2, z_1, z_2

> F2 := AppellF2(a, b1, b2, c1, c2, z1, z2)
    F2 := F_2(a, b_1, b_2, c_1, c_2, z_1, z_2)
```

How does this work?
HeunPrime(z0) makes sense (this is just a taylor expansion), we will see that $C_0 = \text{Heun}(z_0)$ and $C_1 = \text{HeunPrime}(z_0)$.

The system with initial conditions: at this point we just say that $u(0) = C_0$, $D(u)(0) = C_1$, but as it makes sense (this is just a taylor expansion), we will see that $C_0 = \text{Heun}(z_0)$ and $C_1 = \text{HeunPrime}(z_0)$.

\[ \begin{align*}
\text{ode}_F2 & := \frac{d^4}{dt^4} u(t) = \frac{1}{t^2} \frac{1}{(t-1)(t+z_1-1)} \left( -\left( 2a + 2b_2 - c_1 + 8 \right)^2 \right) \hspace{1cm} (2.1.2.5)
\end{align*} \]

\[ \begin{align*}
& + \left( \left( z_1 - 2 \right) b_2 + \left( z_1 - 2 \right) a + \left( z_1 - 2 \right) c_2 + \left( -b_1 + 5 \right) z_1 + c_1 - 10 \right) t
\end{align*} \]

\[ \begin{align*}
& - \left( 2c_2 + 2 \right) \left( z_1 - 1 \right) t \left( \frac{d^3}{dt^3} \ u(t) \right) + \left( -b_2^2 + 4a + 2c_1 - 9 \right) b_2
\end{align*} \]

\[ \begin{align*}
& - a^2 + \left( c_1 - 9 \right) a + 4c_1 - 14 \right)^2 + \left( \left( -z_1 + 2 \right) a + \left( -z_1 + 2 \right) c_2
\end{align*} \]

\[ \begin{align*}
& + \left( b_1 - 2 \right) z_1 - c_1 + 4 \right) b_2 - \left( c_2 + 2 \right) \left( z_1 - 2 \right) a + \left( b_1 - 3 \right) z_1 - c_1
\end{align*} \]

\[ \begin{align*}
& + 6 \right) c_2 + \left( 2b_1 - 4 \right) z_1 - c_1 + 8 \right) t + c_2 \left( c_2 + 1 \right) \left( z_1 - 1 \right) \left( \frac{d^2}{dt^2} \ u(t) \right)
\end{align*} \]

\[ \begin{align*}
& + 2 \left( \frac{z_1 - 2}{2} + \frac{-b_1 + 1}{2} \right) \frac{z_1}{2} + \frac{c_1 - 1}{2} \right) \right) c_2 \left( \frac{d}{dt} \ u(t) \right)
\end{align*} \]

\[ \begin{align*}
& + u(t) a b_2 \left( a - c_1 + 1 \right) \right) \right) \left( b_2 + 1 \right)
\end{align*} \]
\[-a^2 + (c_1 - 9) a + 4 c_1 - 14 \right)^2 + \left( \left( (-z_i + 2) a + (-z_i + 2) c_2 \right) \right) + (b_i - 2) z_i - c_1 + 4 \right) b_2 - (c_2 + 2) (z_i - 2) a + \left( (b_i - 3) z_i - c_i + 6 \right) c_2 + (2 b_i - 4) z_i - 2 c_i + 8 \right) t + c_2 (c_2 + 1) (z_i - 1) \right) \left( \frac{d^2}{dt^2} u(t) \right) \right) - \left(2 \left( a - \frac{c_i}{2} + 1 \right) b_2 + (1 + a) (a - c_i + 2) \right) t + \left( \frac{z_i - 2}{2} a \frac{b_i - c_i}{2} + \frac{c_i}{2} - 1 \right) c_2 \left( \frac{d}{dt} u(t) \right) + u(t) a b_2 (a - c_i + 1) \left( b_2 + 1 \right), u(0) = C0, D(u)(0) = C1, \]
\[D^{(2)}(u)(0) = C2, D^{(3)}(u)(0) = C3 \]

Compute the recurrence now

\[
> C_z0_rec \leftarrow \text{simplify(gfun:-diffeqtorec(sys_z0, u(t), v(n)))} \]

\[
C_z0_rec \leftarrow \left\{ (z_i - 2) n + (a - b_i + 1) z_i - 2 a + c_i - 2) (n + c_2) (n + 1) (n + b_2 + 1) v(n + 1) - (n + 2) (n + 1) (n + c_2 + 1) (n + c_2) (z_i - 1) v(n + 2) + v(n) (n + b_2 + 1) (n + b_2) (a + n) (a + n - c_i + 1), v(0) = C0, v(1) = C1, v(2) = \frac{C2}{2}, v(3) = \frac{C3}{6} \right\} \]

This recurrence can be transformed into a procedure that will "compute the nth coefficient, departing from the previous one"

\[
> U \leftarrow \text{GenerateRecurrence(F2)} \]

\[
U \leftarrow \text{proc}(i) \]

\[
\text{option hfloat, cache;} \]

\[

\[
\end \]

The formula we are exploring here is

\[
> F2 \leftarrow \text{Sum}(U(j) z[2^j], j = 0..\text{infinity}) \]
\[ F_2(a, b_1, b_2, c_1, c_2, z_1, z_2) = \sum_{j=0}^{\infty} U(j) z_2^j \]  

(2.1.2.8)

Let's see

> \( U(0) \)

\[ 2F_1(a, b_1; c_1; z_1) \]  

(2.1.2.9)

> \( U(1) \)

\[ \frac{a b_2 2F_1(b_1, 1 + a; c_1; z_1)}{c_2} \]

(2.1.2.10)

> \( U(2) \)

\[ \frac{a b_2 (1 + a) (b_2 + 1) 2F_1(b_1, 2 + a; c_1; z_1)}{2 c_2 (c_2 + 1)} \]

(2.1.2.11)

> \( U(3) \)

\[ \frac{a b_2 2F_1(b_1, a + 3; c_1; z_1) (b_2 + 2) (b_2 + 1) (2 + a) (1 + a)}{6 c_2 (c_2 + 3 c_2 + 2)} \]

(2.1.2.12)

---

**Numerical integration of an underlying differential equation (ODEs and dsolve/numeric)**

The AppellF1 ODE

> restart;

> \( F1 := \text{AppellF1}(a, b_1, b_2, c, x, z) \)

\[ F1 := F_1(a, b_1, b_2, c, x, z) \]

(2.1.3.1)

> \( \text{AppellF1ODE} := \text{subs}(z1 = x, y = f, \text{diff}(y(z), z, z) = (((a + 2 * b_2 + 4) * z^2 + (a + b_1 - b_2 - 3) * x - c - b_2 - 2) z + x (c - b_1 + 1)) \left( \frac{d^2}{dz^2} f(z) \right) - (b_2 + 1) \left( ((-2 * a - b_2 - 2) z + (a - b_1 + 1) x + c) \left( \frac{d}{dz} f(z) \right) - f(z) \right) a b_2) \)) \]

(2.1.3.2)

A first order ODE system version of this equation: we see \( YP[n] \) representing the nth derivative, plus adding the corresponding auxiliary ODE

> \( \text{DEtools[convertsys]}(\text{AppellF1ODE}, [f(0) = Y0, D(f)(0) = DY0, D^2(f)(0) = DDY0], f(z), z) \)
\[
\begin{align*}
YP_1 &= Y_2, 
YP_2 &= Y_3, 
YP_3 &= \frac{1}{z(z-1)(x-z)} \left( \left( (a + 2b + 4)z^2 + ((-a + b - 2) x - c - b - 2) z + x (c - b + 1) \right) Y_3 - (b + 1) \right) 
\end{align*}
\]

Now we want to be able to perform the evaluation for a complex shift and scale, so we can integrate the DE along any line in the complex plane, so apply a change of co-ordinates:

\[ \text{macro } (S = z0, M = \Delta) ; \]

\[ DE0 := \text{simplify} \left( \text{factor} \left( \text{isolate} \left( \text{PDEtools} \left[ \text{dchange} \right] \left( \{ z = S + t \cdot M, f(z) = F(t) \}, \text{AppellF1ODE}, \{ t, F(t) \} \right), \text{diff} \left( F(t), t, 3 \right) \right) \right) \right) ; \]

\[ DE0 := \frac{d^3}{dt^3} F(t) = - \left( 2 \Delta \left( \frac{\Delta^2 (a + 2b + 4)}{2} \right) \right)
\]

\[ - t \left( ((-2a - 4b - 8)z0 + (x + 1)b2 + (a - b1 + 3)x + c + 2) \Delta \right.
\]

\[ + \left( 2 + \frac{a}{2} + b2 \right) z0^2 + \left( \left( -\frac{1}{2} - \frac{x}{2} \right) b2 + \left( -\frac{3}{2} - \frac{a}{2} + \frac{b1}{2} \right) x - \frac{c}{2} \right.
\]

\[ - 1 \right) z0 + \left( x (c - b1 + 1) \right) \left( \frac{d^2}{dt^2} F(t) \right) + \left( b2 + 1 \right) \Delta \left( \left( t \left( a + \frac{b2}{2} \right. \right.ight.
\]

\[ \left. + 1 \right) \Delta + \left( a + \frac{b2}{2} + 1 \right) z0 + \left( -\frac{a}{2} + \frac{b1}{2} - \frac{1}{2} \right) x - \frac{c}{2} \right) \left( \frac{d}{dt} F(t) \right)
\]

\[ + \left( \Delta F(t) a b2 \right) \right) \right) \right) \left( (t \Delta + z0) \left( t \Delta + z0 - 1 \right) \left( t \Delta - x + z0 \right) \right)
\]

First optimization step

\[ \text{newvars } := \{ s1 = S + t \star M \}; \]

\[ \text{newvars } := \{ s1 = t \Delta + z0 \} \]

\[ DE1 := \text{simplify} \left( DE0, \{ S = s1 - t \star M \} \right); \]

\[ DE1 := \frac{d^3}{dt^3} F(t) = - \frac{1}{s1(s1 - 1)(s1 + x)} \left( \Delta \left( \left( (-a - 2b - 4)s1^2 + ((a - b1 + 3)x + c + b2 + 2)s1 - x (c - b1 + 1) \right) \right)
\]

\[ - \left( b1 + b2 + 3 \right) x + c + b2 + 2 \right) s1 - x (c - b1 + 1) \right) \left( \frac{d^2}{dt^2} F(t) \right) + \left( b2 
\]

\[ + 1 \right) \Delta \left( ((-2a - b2 + 2)s1 + (a - b1 + 1)x + c) \right) \left( \frac{d}{dt} F(t) \right) \]
Next step is performed with this little program

```
> OptimizedSequence := proc(EE)
local Objects1, j, s, s_in_ee, s_used, S, N, Objects, ee,
newvars, tmp_newvars, last_N;

ee := EE;
s_in_ee := indets(ee, 'suffixed(s)');
s_used := {};
newvars := {};

" "

                  "
do
   Objects := remove(u -> u::'~`*~' and nops(indets(u,
   'name')) = 1, sort([op(indets(ee, 'Or(~`+~, ~`*~, ~`^~)'))],
   'length'));
   if Objects = [] then break fi;
   N := nops(Objects);
   S := [`tools/_Qn`('s', [ee, newvars], N)];
tmp_newvars := Objects[1 .. N] =~ S;
   ee := subs(op(tmp_newvars), ee);
   s_used := indets(ee, 'suffixed(s)) minus indets
   (newvars, 'suffixed(s)'), minus s_in_ee;
   newvars := newvars union map((u, NV) -> u = subs
   (NV, u), map(rhs = lhs, tmp_newvars));

                  "
   last_N := max(map(parse, map(substring, indets(newvars,
   'suffixed(s)'), 2..-1)));
   s_used := select(member, [seq(cat(s, j), j=1..last_N)], map
   (lhs, newvars));
   newvars := s_used =~ subs(newvars, s_used);

# redefine the sn to match their number
   S := [`tools/_Qn`('s', [EE], nops(newvars))];
   [op(subs(s_used =~ S, [op(newvars), ee]))];
end:

> OptimizedSequence(DE1)
```

\[
\begin{align*}
-\Delta F(t) \ a \ b^2 \\
\end{align*}
\]

\[
\begin{align*}
s_2 &= \frac{1}{s_1}, s_3 = s_1^2, s_4 = b^2 + 1, s_5 = -s_1 + x, s_6 = s_1 - 1, s_7 = -2 - a - b^2 - 2, \\
s_8 &= -a - 2 - b^2 - 4, s_9 = a - b - 1, s_{10} = c - b^1 + 1, s_{11} = a - b + b^2 \\
+ 3, s_{12} &= -\Delta F(t) \ a \ b^2, s_{13} = \frac{1}{s_5}, s_{14} = \frac{1}{s_6}, s_{15} = s_9 x, s_{16} = s_{11} x, s_{17} \\
= s_7 s_1, s_{18} &= s_8 s_3, s_{19} = -x s_{10}, s_{20} = c + s_{15} + s_{17}, s_{21} = b^2 + c + 2
\end{align*}
\]
\[ + s16, s22 = s21 s1, s23 = s20 \left( \frac{d}{dt} F(t) \right), s24 = s23 + s12, s25 = s18 + s19 \]

\[ + s22, s26 = s4 \Delta s24, s27 = s25 \left( \frac{d^2}{dt^2} F(t) \right), s28 = s27 + s26, s29 = -\Delta s28 s2 s14 s13, \frac{d^3}{dt^3} F(t) = s29 \]

Incorporate now the first equation not handled via OptimizedSequence

\[ > \text{OS} := [s1 = S + t \cdot M, \text{op}(2.1.3.7)] \]

\[ \text{OS} := \left[ \begin{array}{c}
s1 = t \Delta + z0, s2 = \frac{1}{s1}, s3 = sI^2, s4 = b2 + 1, s5 = -s1 + x, s6 = s1 - 1, \\
s7 = -2 a - b2 - 2, s8 = -a - 2 b2 - 4, s9 = a - b1 + 1, s10 = c - b1 + 1, \\
s11 = a - b1 + b2 + 3, s12 = -\Delta F(t) a b2, s13 = \frac{1}{s5}, s14 = \frac{1}{s6}, s15 = s9 x, \\
s16 = s11 x, s17 = s7 s1, s18 = s8 s3, s19 = -x s10, s20 = c + s15 + s17, s21 \\
= b2 + c + 2 + s16, s22 = s21 s1, s23 = s20 \left( \frac{d}{dt} F(t) \right), s24 = s23 + s12, \\
s25 = s18 + s19 + s22, s26 = s4 \Delta s24, s27 = s25 \left( \frac{d^2}{dt^2} F(t) \right), s28 = s27 \\
+ s26, s29 = -\Delta s28 s2 s14 s13, \frac{d^3}{dt^3} F(t) = s29 \end{array} \right] \]

\[ > \text{parameters} := \text{op}(1..2, F1) \]

\[ \text{parameters} := a, b1, b2, c, x \]

\[ > \text{auxiliaryvariables} := \text{indets}([[2.1.3.7]], \text{symbol}) \text{minus} \{\text{parameters}, t\} \text{union} \{z0\} \]

\[ \text{auxiliaryvariables} := \{\Delta, s1, s10, s11, s12, s13, s14, s15, s16, s17, s18, s19, s2, \\
s20, s21, s22, s23, s24, s25, s26, s27, s28, s29, s3, s4, s5, s6, s7, s8, s9, z0\} \]

\[ > \text{auxiliaryvariables} \text{minus} \text{indets} (\text{auxiliaryvariables, suffixed(s) }) \]

\[ \{\Delta, z0\} \]

\[ > \text{riauxiliaryfunctions} := \{ \]

\[ \text{F(t) = Y[1] + I.Y[5],} \]

\[ \text{diff(F(t), t) = Y[2] + I.Y[6],} \]

\[ \text{diff(F(t), t, t) = Y[3] + I.Y[7],} \]

\[ \text{diff(F(t), t, t, t) = YP[3] + I.YP[7]} \}

\[ \text{riauxiliaryfunctions} := \left\{ \begin{array}{c}
\text{F(t) = Y}_1 + I \cdot \text{Y}_5, \frac{d}{dt} \text{F(t) = Y}_2 + I \cdot \text{Y}_6, \frac{d^2}{dt^2} \text{F(t) = Y}_3 \end{array} \right\} \]
\[ +1Y_7, \quad \frac{d^3}{dt^3} F(t) = YP_3 + 1YP_7 \]

\[ \text{riOS} \mathrel{:=} \text{map}(u \rightarrow u = \text{cat}(r, u) + I \cdot \text{cat}(i, u), \{ \text{parameters} \}) \]

\[ \text{risymbols} \mathrel{:=} \{ \Delta = r\Delta + 1i\Delta, a = ra + 1ia, b1 = rb1 + 1ib1, b2 = rb2 + 1ib2, c = rc + 1ic, s1 = rs1 + 1is1, s10 = rs10 + 1is10, s11 = rs11 + 1is11, s12 = rs12 + 1is12, s13 = rs13 + 1is13, s14 = rs14 + 1is14, s15 = rs15 + 1is15, s16 = rs16 + 1is16, s17 = rs17 + 1is17, s18 = rs18 + 1is18, s19 = rs19 + 1is19, s2 = rs2 + 1is2, s20 = rs20 + 1is20, s21 = rs21 + 1is21, s22 = rs22 + 1is22, s23 = rs23 + 1is23, s24 = rs24 + 1is24, s25 = \ldots + 1is25, s26 = rs26 + 1is26, s27 = rs27 + 1is27, s28 = rs28 + 1is28, s29 = rs29 + 1is29, s3 = rs3 + 1is3, s4 = rs4 + 1is4, s5 = rs5 + 1is5, s6 = rs6 + 1is6, s7 = rs7 + 1is7, s8 = rs8 + 1is8, s9 = rs9 + 1is9, x = rx + 1ix, z0 = rz0 + 1iz0 \} \]

\[ \text{riOS} \mathrel{:=} \text{subs}(\text{risymbols union riauxiliaryfunctions}, \text{OS}) \]

\[ \text{riOS} \mathrel{:=} \left[ rs1 + 1is1 = t \cdot (r\Delta + 1i\Delta) + rz0 + 1iz0, rs2 + 1is2 \right] \]

\[ = \frac{1}{rs1 + 1is1}, rs3 + 1is3 = (rs1 + 1is1)^2, rs4 + 1is4 = rb2 + 1ib2 + 1, \]

\[ rs5 + 1is5 = -rs1 - 1is1 + rx + 1ix, rs6 + 1is6 = rs1 + 1is1 - 1, rs7 + 1is7 = -2ra - 21ia - rb2 - 1ib2 - 2, rs8 + 1is8 = -ra - 1ia - 2 rb2 \]

\[ -21ib2 - 4, rs9 + 1is9 = ra + 1ia - rb1 - 1ib1 + 1, rs10 + 1is10 = rc + 1ic - rb1 - 1ib1 + 1, rs11 + 1is11 = ra + 1ia - rb1 - 1ib1 + rb2 \]

\[ +1ib2 + 3, rs12 + 1is12 = -(r\Delta + 1i\Delta) \cdot (Y_1 + 1Y_5) \cdot (ra + 1ia) \cdot (rb2 + 1ib2), rs13 + 1is13 = \frac{1}{rs5 + 1is5}, rs14 + 1is14 \]

\[ = \frac{1}{rs6 + 1is6}, rs15 + 1is15 = (rs9 + 1is9) \cdot (rx + 1ix), rs16 + 1is16 = (rs11 + 1is11) \cdot (rx + 1ix), rs17 + 1is17 = (rs7 + 1is7) \cdot (rs1 + 1is1), rs18 + 1is18 = (rs8 + 1is8) \cdot (rs3 + 1is3), rs19 + 1is19 = -(rx + 1ix), rs20 + 1is20 = rc + 1ic + rs15 + 1is15 + rs17 + 1is17, rs21 + 1is21 = rb2 + 1ib2 + rc + 1ic + 2 + rs16 + 1is16, rs22 \]
+1 is22 = (rs21 + 1 is21) (rs1 + 1 is1), rs23 + 1 is23 = (rs20
+ 1 is20) (Y2 + 1 Y6), rs24 + 1 is24 = rs23 + 1 is23 + rs12 + 1 is12, rs25
+ 1 is25 = rs18 + 1 is18 + rs19 + 1 is19 + rs22 + 1 is22, rs26 + 1 is26
= (rs4 + 1 is4) (rDelta + 1 iDelta) (rs24 + 1 is24), rs27 + 1 is27 = (rs25
+ 1 is25) (Y3 + 1 Y7), rs28 + 1 is28 = rs27 + 1 is27 + rs26 + 1 is26, rs29
+ 1 is29 = - (rDelta + 1 iDelta) (rs28 + 1 is28) (rs2 + 1 is2) (rs14
+ 1 is14) (rs13 + 1 is13), YP3 + 1 YP7 = rs29 + 1 is29]

> rseq := NULL:
    for i in riOS do
        rseq := rseq, simplify(map(Re, i)), simplify(map(Im, i)) assuming real;
    end do;
    i := 'i';
    rseq;

rs1 = rDelta t + rz0, is1 = iDelta t + iz0, rs2 = \frac{rs1}{is1^2 + rs1^2}, is2 = \frac{rs1}{is1^2 + rs1^2}, rs3 = -\frac{is1^2}{is1^2 + rs1^2}, rs4 = rb2 + 1, is4 = ib2,

rs5 = -rs1 + rx, is5 = -is1 + ix, rs6 = rs1 - 1, is6 = is1, rs7 = -ra - rb2

- 2, is7 = -2 ia - ib2, rs8 = -ra - 2 rb2 - 4, is8 = -ia - 2 ib2, rs9 = ra

- rb1 + 1, is9 = ia - ib1, rs10 = rc - rb1 + 1, is10 = ic - ib1, rs11 = ra

- rb1 + rb2 + 3, is11 = ia - ib1 + ib2, rs12 = ((ia rb2 + ib2 ra) iDelta

+ rDelta (ia ib2 - ra rb2)) Y1 - Y5 ((ia ib2 - ra rb2) iDelta

- rDelta (ia rb2 + ib2 ra)) iDelta

- rDelta (ia rb2 + ib2 ra)) Y1 + Y5 ((ia rb2 + ib2 ra) iDelta

+ rDelta (ia ib2 - ra rb2)), rs13 = \frac{rs5}{is5^2 + rs5^2}, is13 = -\frac{is5}{is5^2 + rs5^2},

rs14 = \frac{rs6}{is6^2 + rs6^2}, is14 = -\frac{is6}{is6^2 + rs6^2}, rs15 = -is9 ix + rs9 rx, is15

= is9 rx + ix rs9, rs16 = -is11 ix + rs11 rx, is16 = is11 rx + ix rs11, rs17 =

-is1 is7 + rs1 rs7, is17 = is1 rs7 + is7 rs1, rs18 = -is3 is8 + rs3 rs8, is18

= is3 rs8 + is8 rs3, rs19 = is10 ix - rs10 rx, is19 = -is10 rx - ix rs10, rs20

= rc + rs15 + rs17, is20 = ic + is15 + is17, rs21 = rb2 + rc + 2 + rs16,

is21 = ib2 + ic + is16, rs22 = -is1 is21 + rs1 rs21, is22 = is1 rs21

+ is21 rs1, rs23 = -is20 Y6 + rs20 Y2, is23 = is20 Y2 + rs20 Y6, rs24 = rs23

+ rs12, is24 = is23 + is12, rs25 = rs18 + rs19 + rs22, is25 = is18 + is19

+ is22, rs26 = rs4 (-iDelta is24 + rDelta rs24) - is4 (iDelta rs24

...
Add equations for primed variables

\[ YP_{equations} \Leftarrow YP_1 = Y_2, YP_5 = Y_6, YP_2 = Y_3, YP_6 = Y_7 \quad (2.1.3.16) \]

\[ \text{RIOS} := rseq, YP_{equations} \]
\[ \text{RIOS} := rs1 = rDelta t + rz0, is1 = iDelta t + iz0, rs2 = \frac{rs1}{is1^2 + rs1^2}, is2 = \quad (2.1.3.17) \]

\[ \text{rs5} = -rs1 + ox, is5 = -is1 + ox, rs6 = rs1 - 1, is6 = is1, rs7 = -2ra - rb2 - 2, is7 = -2ia - ib2, rs8 = -ra - 2rb2 - 4, is8 = -ia - 2ib2, rs9 = ra - rb1 + 1, is9 = ia - ib1, rs10 = rc - rb1 + 1, is10 = ic - ib1, rs11 = ra - rb1 + rb2 + 3, is11 = ia - ib1 + ib2, rs12 = ((ia ib2 + ib2 ra) iDelta + rDelta (ia ib2 - ra rb2)) Y_1 - Y_5 ((ia ib2 - ra rb2) iDelta - rDelta (ia rb2 + ib2 ra)) Y_1 + Y_5 ((ia rb2 + ib2 ra) iDelta + rDelta (ia ib2 - ra rb2)) \]

\[ \text{rs13} = \frac{rs5}{is5^2 + rs5^2}, is13 = -\frac{is5}{is5^2 + rs5^2}, \text{rs14} = -\frac{is6}{is6^2 + rs6^2}, rs15 = -is9 ox + rs9 ox, is15 = is9 ox + rs9 ox, rs16 = -is11 ox + rs11 ox, is16 = is11 ox + ix rs11, rs17 = -is1 ox + rs1 ox, is17 = is1 ox + is1 ox, rs18 = -is3 ox + rs3 ox, is18 = is3 ox + rs3 ox, rs19 = is10 ox - rs10 ox, is19 = -is10 ox - ix rs10, rs20 = rc + rs15 + rs17, is20 = ic + is15 + is17, rs21 = rb2 + rc + 2 + rs16, is21 = ib2 + ic + is16, rs22 = -is1 is21 + rs1 rs21, is22 = is1 rs21 \]
+ is21 rs1, rs23 = -is20 Y_6 + rs20 Y_2, is23 = is20 Y_2 + rs20 Y_6, rs24 = rs23 + rs12, is24 = is23 + is12, rs25 = rs18 + rs19 + rs22, is25 = is18 + is19 + is22, rs26 = rs4 (-iDelta is24 + rDelta rs24) - is4 (iDelta rs24 + 24 rDelta) - (iDelta is24 - rDelta rs24) is4, rs27 = -is25 Y_7 + rs25 Y_3, is27 = is25 Y_3 + rs25 Y_7,
s28 = rs27 + rs26, is28 = is27 + is26, rs29 = (((-is2 rs28 - is28 rs2) is14 - rs14 (is2 is28 - rs2 rs28)) is13 - ((is2 is28 - rs2 rs28) is14 - rs14 (is2 rs28 + is28 rs2)) is13) iDelta - (((is2 is28 - rs2 rs28) is14 - rs14 (is2 rs28 + is28 rs2)) is13) rDelta, is29 = (((-is2 is28 + rs2 rs28) is13 + is28 rs2 + is28 rs2) is13 + ((is2 rs28 + is28 rs2)) rs23) iDelta + rDelta (((is2 rs28 + is28 rs2) is14 + rs14 (is2 rs28 - rs2 rs28)) is13 + ((is2 rs28 - rs2 rs28) is14 - rs14 (is2 rs28 + is28 rs2)) rs13), YP_3 = rs29, YP_7 = is29, YP_1 = Y_2, YP_5 = Y_6, YP_2 = Y_3, YP_6 = Y_7

Next there is a procedure, RIOSToProc, that takes the RIOS and returns the procedure used by dsolve/numeric to numerically evaluate the ODE behind the RIOS (in this case, the AppellF1 function)

```maple
> RIOSToProc := proc( RIOS )
local previous_warnlevel;
try
    previous_warnlevel := interface(warnlevel = 0);
    return parse(sprintf("proc(N,t,Y,YP) %a end", cat(op(map(u -> sprintf("%a := %a;", op(u)), RIOS))))))
finally:
    interface(warnlevel = previous_warnlevel)
end:
end:
```

Create dsolve's procedure from the Real and Imaginary parts of the Optimized Sequence

```maple
> dproc := RIOSToProc([ RIOS ])$$
dproc := proc(N, t, Y, YP)$$
local rs1, is1, rs2, is2, rs3, is3, rs4, is4, rs5, is5, rs6, is6, rs7, is7, rs8, is8, rs9, is9, rs10, is10, rs11, is11, rs12, is12, rs13, is13, rs14, is14, rs15, is15, rs16, is16, rs17, is17, rs18, is18, rs19, is19, rs20, is20, rs21, is21, rs22, is22, rs23, is23, rs24, is24, rs25, is25, rs26, is26, rs27, is27, rs28, is28, rs29, is29;
rs1 := rDelta*t + rz0;
is1 := iDelta*t + iz0;
rs2 := rs1/(is1^2 + rs1^2);
is2 := -is1/(is1^2 + rs1^2);
is3 := -is1^2 + rs1^2;
is3 := 2*is1*rs1;
```
\[\begin{align*}
rs4 & := rb2 + 1; \\
is4 & := ib2; \\
rs5 & := - rs1 + rx; \\
is5 & := - is1 + ix; \\
rs6 & := rs1 - 1; \\
is6 & := is1; \\
rs7 & := - 2* ra - rb2 - 2; \\
is7 & := - 2* ia - ib2; \\
rs8 & := - ra - 2* rb2 - 4; \\
is8 & := - ia - 2* ib2; \\
rs9 & := ra - rb1 + 1; \\
is9 & := ia - ib1; \\
rs10 & := rc - rb1 + 1; \\
is10 & := ic - ib1; \\
rs11 & := ra - rb1 + rb2 + 3; \\
is11 & := ia - ib1 + ib2; \\
rs12 & := ((ia* rb2 + ib2* ra)* iDelta + rDelta* (ia* ib2 - ra* rb2)) * Y[1] - Y[5]* ((ia* ib2 - ra* rb2)* iDelta + rDelta* (ia* rb2 + ib2* ra)); \\
is12 & := ((ia* ib2 - ra* rb2)* iDelta - rDelta* (ia* rb2 + ib2* ra)) * Y[1] + Y[5]* ((ia* rb2 + ib2* ra)* iDelta + rDelta* (ia* ib2 - ra* rb2)); \\
rs13 & := rs5/(is5^2 + rs5^2); \\
is13 & := - is5/(is5^2 + rs5^2); \\
rs14 & := rs6/(is6^2 + rs6^2); \\
is14 & := - is6/(is6^2 + rs6^2); \\
rs15 & := - is9* ix + rs9* rx; \\
is15 & := is9* rx + ix* rs9; \\
rs16 & := - is11* ix + rs11* rx; \\
is16 & := is11* rx + ix* rs11; \\
rs17 & := - is1* is7 + rs1* rs7; \\
is17 & := is1* rs7 + is7* rs1; \\
rs18 & := - is3* is8 + rs3* rs8; \\
is18 & := is3* rs8 + is8* rs3; \\
rs19 & := is10* ix - rs10* rx; \\
is19 & := - is10* rx - ix* rs10; \\
rs20 & := rc + rs15 + rs17; \\
is20 & := ic + is15 + is17; \\
rs21 & := rb2 + rc + 2 + rs16;
\end{align*}\]
Example 1: $E_1 \left( 1, 2, \frac{1}{2}, 1 + I, \frac{5}{6}, \frac{3}{4} + \frac{1}{20} \right)$
1) Pick values for \(a, b1, b2, c, x\)
2) Choose a start point and direction
3) Evaluate the relevant routine, and use dsolve/numeric

\[ F_1(a, b1, b2, c, x, z) \]  
(2.1.3.1.1)

1) Pick values for \(a, b1, b2, c, x\)

\[ z_{\text{val}} := \frac{3}{4} + \frac{i}{20} \]  
(2.1.3.1.2)

\[ \text{evalf}(\text{abs}(z_{\text{val}})) \]  
0.7516648190  
(2.1.3.1.3)

For the purpose of this example, choose a \(z1\) such that \(|z2| < |z1|\)

\[ AF := \%\text{AppellF1}(1, 2, \frac{1}{2}, 1 + i, \frac{5}{6}, z_{\text{val}}) \]  
(2.1.3.1.4)

\[ \text{infolevel}[\text{AppellF1}] := 5 \]  
(2.1.3.1.5)

dsolve/numeric works best with this value of Digits

\[ Digits := \text{trunc}(\text{evalhf}(Digits)) \]  
\( Digits := 15 \)  
(2.1.3.1.6)

\[ AFval := \text{evalf}(AF) \]  
(2.1.3.1.7)

The parameters of \(F1\), and their real (rP) and imaginary (iP) parts

\[ P := \{op(1..-2, F1)\} \approx \{op(1..-2, AF)\} \]  
(2.1.3.1.8)

\[ rP := \text{map}(u \rightarrow \text{cat}(r, \text{lhs}(u))) = \text{Re}(\text{rhs}(u)), P \]  
(2.1.3.1.9)

\[ iP := \text{map}(u \rightarrow \text{cat}(i, \text{lhs}(u))) = \text{Im}(\text{rhs}(u)), P \]  
(2.1.3.1.10)

2) Choose a start point and direction. We aim at computing \(F1\) at \(z = z2 = z_{\text{val}} = \frac{3}{4} + \frac{1}{20}\).
We start with deriving formulas for \( z_0 \) and \( \Delta \) and choose convenient values when possible (there is some arbitrariness in the choice of \( \Delta \) and \( z_0 \))

We have:

\[
Z := z = t \Delta + z_0
\]

\[
Z := z = t \Delta + z_0
\]  \hspace{1cm} (2.1.3.1.11)

\[
\text{subs (risymbols, Z)}
\]

\[
z = t \left( r\Delta + 1 \cdot i\Delta \right) + rz_0 + i iz_0
\]  \hspace{1cm} (2.1.3.1.12)

\[
\text{Re}(z) = \text{(Re(rhs(\%)) assuming real})
\]

\[
\Re(z) = r\Delta \cdot t + r z_0
\]  \hspace{1cm} (2.1.3.1.13)

\[
\text{Im}(z) = \text{(Im(rhs(\%)) assuming real})
\]

\[
\Im(z) = i\Delta \cdot t + i z_0
\]  \hspace{1cm} (2.1.3.1.14)

\[
\text{solve( (2.1.3.1.13), (2.1.3.1.14), \{t, i\Delta\} )}
\]

\[
\begin{align*}
i\Delta &= \frac{r\Delta \cdot (\Im(z) - i z_0)}{\Re(z) - r z_0}, \\
t &= \frac{\Re(z) - r z_0}{r\Delta}
\end{align*}
\]

Then, \( i\Delta \) is different from zero only if \( \Im(z) - i z_0 \neq 0 \), the real variable \( t \) is given by

\[
t = \frac{\Re(z) - r z_0}{r\Delta},
\]

and all of \( r\Delta, rz_0 \) and \( iz_0 \) are arbitrary, which amounts to say that

• "The initial point \( z_0 \) could be anyone provided it is not a singularity, and the value of \( r\Delta \) can be chosen at will - note however that this value will be taken into account when computing initial values at \( z_0 \) for dsolve's procedure"

Although any allowed value for \( z_0 \) and \( \Delta \) are OK and lead to the correct value of \( F_1 \), if the \( \Im(z) \) is small (within the radius of convergence of the \( F_1 \) series) we can always choose \( iz_0 = \Im(z) \) so that \( i\Delta = 0 \), then choose \( r\Delta = 1 \) (or as big as possible?), and \( rz_0 \) as big as possible (ay 3/4 of the radius of convergence) so that the computation of \( z_0 \) is easy and the numerical integration over \( t \) is along a smaller path. In this example the radius of convergence around the origin is \( \text{abs}(z1) = 5/6 \), so

\[
\text{eval((2.1.3.1.15), z0_Delta union \{z = z_val\})}
\]

\[
\left\{ i\Delta = 0, t = \frac{1}{8} \right\}
\]  \hspace{1cm} (2.1.3.1.18)

\[
\text{z0_Delta := z0_Delta union select(has, (2.1.3.1.18), i\Delta)}
\]

\[
\left\{ i\Delta = 0, iz_0 = \frac{1}{20}, r\Delta = 1, rz_0 = \frac{5}{8} \right\}
\]  \hspace{1cm} (2.1.3.1.19)

\[
\text{Delta_val := subs(z0_Delta, r\Delta + i \cdot i\Delta)}
\]  \hspace{1cm} (2.1.3.20)
\[
\Delta_{val} := 1 \quad (2.1.3.1.20)
\]

> \[ t_{val} := \text{select}(\text{has}, (2.1.3.1.18), t)[1] \]

\[
t_{val} := t = \frac{1}{8} \quad (2.1.3.1.21)
\]

> `evalf` (%)

\[
t = 0.125000000000000 \quad (2.1.3.1.22)
\]

So, the value of \( z_0 \) is 1/10, the evaluating point is \( z = 3/4 \), that implies on integrating the \( t \) variable from \( t = 0 \) to \( t = 13/20 \), and the values of \( rz_0, iz_0, r\Delta, i\Delta \) are, respectively

The procedure to use with `dsolve/numeric` is the one we get from `dproc` by substituting the values of all the relevant variables \( rP, iP \) and \( z0\_Delta \\
> \text{thisproc} := \text{subs}(rP, iP, z0\_Delta, \text{eval(dproc)}) \\
\text{thisproc} := \text{proc}(N, t, Y,YP) \quad (2.1.3.1.23) \\
\text{local} rs1, is1, rs2, is2, rs3, is3, rs4, is4, rs5, is5, rs6, is6, rs7, is7, rs8, \\
is8, rs9, is9, rs10, is10, rs11, is11, rs12, is12, rs13, is13, rs14, is14, \\
rs15, is15, rs16, is16, rs17, is17, rs18, is18, rs19, is19, rs20, is20, rs21, \\
is21, rs22, is22, rs23, is23, rs24, is24, rs25, is25, rs26, is26, rs27, is27, \\
rs28, is28, rs29, is29; \\
rs1 := t + 5/8; \\
is1 := 1/20; \\
rs2 := rs1/(is1^2 + rs1^2); \\
is2 := - is1/(is1^2 + rs1^2); \\
rs3 := - is1^2 + rs1^2; \\
is3 := 2*is1*rs1; \\
rs4 := 3/2; \\
is4 := 0; \\
rs5 := - rs1 + 5/6; \\
is5 := - is1; \\
rs6 := rs1 - 1; \\
is6 := is1; \\
rs7 := - 9/2; \\
is7 := 0; \\
rs8 := - 6; \\
is8 := 0; \\
rs9 := 0; \\
is9 := 0; \\
rs10 := 0; \\
is10 := 1; \\
rs11 := 5/2; \\
is11 := 0; \]
\[\begin{align*}
rs12 & := -1/2 \cdot Y[1]; \\
is12 & := -1/2 \cdot Y[5]; \\
rs13 & := rs5 / (is5^2 + rs5^2); \\
is13 & := -is5 / (is5^2 + rs5^2); \\
rs14 & := rs6 / (is6^2 + rs6^2); \\
is14 & := -is6 / (is6^2 + rs6^2); \\
rs15 & := 5/6 \cdot rs9; \\
is15 & := 5/6 \cdot is9; \\
rs16 & := 5/6 \cdot rs11; \\
is16 & := 5/6 \cdot is11; \\
rs17 & := -is1 \cdot is7 + rs1 \cdot rs7; \\
is17 & := is1 \cdot rs7 + is7 \cdot rs1; \\
rs18 & := -is3 \cdot is8 + rs3 \cdot rs8; \\
is18 & := is3 \cdot rs8 + is8 \cdot rs3; \\
rs19 & := -5/6 \cdot rs10; \\
is19 & := -5/6 \cdot is10; \\
rs20 & := 1 + rs15 + rs17; \\
is20 & := 1 + is15 + is17; \\
rs21 & := 7/2 + rs16; \\
is21 & := 1 + is16; \\
rs22 & := -is1 \cdot is21 + rs1 \cdot rs21; \\
is22 & := is1 \cdot rs21 + is21 \cdot rs1; \\
rs23 & := -is20 \cdot Y[6] + rs20 \cdot Y[2]; \\
is23 & := is20 \cdot Y[2] + rs20 \cdot Y[6]; \\
rs24 & := rs23 + rs12; \\
is24 & := is23 + is12; \\
rs25 & := rs18 + rs19 + rs22; \\
is25 & := is18 + is19 + is22; \\
rs26 & := rs4 \cdot rs24 - is4 \cdot is24; \\
is26 & := rs4 \cdot is24 + rs24 \cdot is4; \\
rs27 & := -is25 \cdot Y[7] + rs25 \cdot Y[3]; \\
is27 & := is25 \cdot Y[3] + rs25 \cdot Y[7]; \\
rs28 & := rs27 + rs26; \\
is28 & := is27 + is26; \\
rs29 & := -((is2 \cdot is28 - rs2 \cdot rs28) \cdot is14 - rs14 \cdot (is2 \cdot rs28 + is28 * rs2) \cdot is13 + ((is2 \cdot rs28 + is28 \cdot rs2) \cdot is14 + rs14 \cdot (is2 * is28 - rs28) \cdot rs13) \cdot is12; \\
is29 & := (is2 \cdot rs28 + is28 \cdot rs2) \cdot is14 + rs14 \cdot (is2 \cdot is28 - rs2
\[
\text{Numerical evaluation of AppellF1(1., 2.,} \quad \\
\text{DF} \quad \text{end}
\]

Now we need the values of F1 and its 1st and 2nd derivatives at z0, to be used as starting
point by dsolve. Note however that, after changing variables \( z = S + Mt \), we have

\[
[f(z), \text{diff}(f(z), z), \text{diff}(f(z), z, z)] : \% = \text{PDEtools}[\text{dechange}](\{z = S + t M, \\
f(z) = F(t)\}, \%, [t, F(t)], \text{known = all})
\]

where on the lhs f(z) represent \( F_1(a, b1, b2, c, x, z) \) and on the rhs \( F(t) \) represents
\( F_1(a, b1, b2, c, x, t) \) so to compute \([F(t), \dot{F}(t), \ddot{F}(t)]\) we compute

\[
\text{convert}
\left(
\text{solve}
\left(
\text{convert}
\left(
\left[
\begin{array}{c}
\frac{d}{dz} f(z) = \frac{\Delta}{F(t)} F(t), \quad \frac{\Delta^2}{d^2 z} f(z) = \frac{\Delta}{F(t)} F(t)
\end{array}
\right],
\frac{\Delta^2}{d^2 z}
\right)
\right),
\text{diff}
\right)
\left[
\begin{array}{c}
\frac{d}{dz} f(z) = \frac{\Delta}{F(t)} F(t), \quad \frac{\Delta^2}{d^2 z} f(z) = \frac{\Delta}{F(t)} F(t)
\end{array}
\right]
\]

In this particular example, Delta = 1 so this observation is not relevant.

We then have

\[
DF := \text{subs}(z = z0, \text{Delta = Delta_val}, \left[F1, \text{Delta} \cdot \text{diff}'(F1, z), \Delta^2 \cdot \text{diff}'(F1, z, z)\right])
\]

\[
DF := \left[
\begin{array}{c}
F_1(a, b1, b2, c, x, z0), \quad \frac{d}{dz0} F_1(a, b1, b2, c, x, z0), \quad \frac{d^2}{dz0^2} F_1(a, b1, \text{b2, c, x, z0})
\end{array}
\right]
\]

\[
\text{subs}(z0 = z0_val, \text{eval}(DF, P))
\]

\[
\left[
F_1\left(1, 2, \frac{1}{2}, 1 + I, \frac{5}{6}, \frac{5}{8} + \frac{1}{20}\right), \left(\frac{1}{4} - \frac{1}{4}\right)F_1\left(2, 2, \frac{3}{2}, 2 + I, \frac{5}{6}, \frac{5}{8} + \frac{1}{20}\right), \frac{5}{8} + \frac{1}{20}, \left(\frac{3}{20} - \frac{9}{20}\right)F_1\left(3, 2, \frac{5}{2}, 3 + I, \frac{5}{6}, \frac{5}{8} + \frac{1}{20}\right)
\right]
\]

\[
DFval := \text{evalf}((\text{2.1.3.27}))
\]

\[
\Rightarrow \text{Numerical evaluation of AppellF1(1., 2.,}
\]

\[
\text{)}
\]

\[
\text{)}
\]
\[
.500000000000000, 1.+1. \times I, .833333333333333, \\
.625000000000000+5.00000000000000e-1 \times I)
\]

\[
\rightarrow \text{AppellF1, checking singular cases}
\]

\[
\rightarrow \text{AppellF1, trying special values}
\]

\[
\rightarrow \text{AppellF1, trying series based on recurrence}
\]

\[
\rightarrow \text{Numerical evaluation of AppellF1(2., 2.,}
\]

\[
1.500000000000000, 2.+1. \times I, .833333333333333, \\
.625000000000000+5.00000000000000e-1 \times I)
\]

\[
\rightarrow \text{AppellF1, checking singular cases}
\]

\[
\rightarrow \text{AppellF1, trying special values}
\]

\[
\rightarrow \text{AppellF1, trying series based on recurrence}
\]

\[
\rightarrow \text{Numerical evaluation of AppellF1(3., 2.,}
\]

\[
2.500000000000000, 3.+1. \times I, .833333333333333, \\
.625000000000000+5.00000000000000e-1 \times I)
\]

\[
\rightarrow \text{AppellF1, checking singular cases}
\]

\[
\rightarrow \text{AppellF1, trying special values}
\]

\[
\rightarrow \text{AppellF1, trying series based on recurrence}
\]

\[
\text{DFval} := [-17.9388701408564 - 16.3439832562898 I,}
\]

\[\text{AFval} \approx 23.2785809735956 + 19.6532599179361 I
\]

\[
\text{evalf}[10](\%\% -\%)
\]

\[
4.1 \times 10^{-7} + 3.5 \times 10^{-7} I
\]

**Example 2:** \[
F_1 \left( 1, 2 + I, \frac{3}{2} - I, 1, \frac{18}{25}, \frac{19}{20} \right)
\]

For the purpose of this example, choose a \( z_1 \) such that \( |z_1| < |z_2| \) and to integrate till \( z_2 \) we need to "go over \( z_1 \)", which if course is not possible and we overcome the difficulty by using a \( z_0 \) such that \( \text{Im}(z_0) > 0 \), so we arrive at \( z_2 \) without going over \( z_1 \)

\[
\rightarrow \text{convert([1, 2 + I, 3/2 - \times I* (1/2), 1, .72, .95, rational)}
\]

\[
\rightarrow \text{The F1 to be evaluated is actually the last example in evalf/tst/AppellF1_Table2.tst.}
\]

\[
\rightarrow AF := \%\text{AppellF1(} op(\%))
\]
\[ AF := F_{11} \left( 1, 2 + 1, \frac{3}{2} - \frac{1}{2}, 1, \frac{18}{25}, \frac{19}{20} \right) \]  

\[ z_{\text{val}} := \text{op}(-1, AF) \]

\[ z_{\text{val}} := \frac{19}{20} \]

\[ \text{evalf}(\text{abs}(z_{\text{val}})) \]

\[ 0.950000000000000 \]

\[ \text{evalf}(\text{abs}(\text{op}(-2, AF))) \]

\[ 0.720000000000000 \]

\[ \text{infolevel}[\text{AppellF1}] := 5 \]

\[ \text{infolevel}_{\text{AppellF1}} := 5 \]

\[ \text{Digits} := \text{trunc}(\text{evalhf}(\text{Digits})) \]

\[ \text{Digits} := 15 \]

The series code resolves the issue swapping \( z_2 \) and \( z_2 \)

\[ AFval := \text{evalf}(AF) \]

\[ \text{case: abs}(z1) < \text{abs}(z2) < 1; \text{ swapping parameters and } z_1 \leftrightarrow z_2 \]

\[ \rightarrow \text{Numerical evaluation of AppellF1}(1., 1.500000000000000-.500000000000000*I, 2.+1.*I, 1., .950000000000000, .720000000000000) \]

\[ \rightarrow \text{AppellF1, checking singular cases} \]

\[ \rightarrow \text{AppellF1, trying special values} \]

\[ \text{special values of AppellF1 successful} \]

\[ AFval := 1.11212017081988 - 254.420420227346 I \]

Go now with dsolve/numeric without swapping. The parameters of F1, and their real (rP) and imaginary (iP) parts

\[ F1(a, b1, b2, c, x, z) \]

\[ P := [\text{op}(1..2, F1)] \Rightarrow [\text{op}(1..2, AF)] \]

\[ P := [a = 1, b1 = 2 + 1, b2 = \frac{3}{2} - \frac{1}{2}, c = 1, x = \frac{18}{25}] \]

\[ rP := \text{map}(u \rightarrow \text{cat}(r, \text{lhs}(u)) = \text{Re}(\text{rhs}(u)), P) \]

\[ rP := [ra = 1, rb1 = 2, rb2 = \frac{3}{2}, rc = 1, rx = \frac{18}{25}] \]

\[ iP := \text{map}(u \rightarrow \text{cat}(i, \text{lhs}(u)) = \text{Im}(\text{rhs}(u)), P) \]

\[ iP := [ia = 0, ib1 = 1, ib2 = -\frac{1}{2}, ic = 0, ix = 0] \]

2) Choose a start point and direction. We aim at computing F1 at \( z = z2 = z_{\text{val}} = \frac{3}{4} + \frac{1}{20} \).

We start with deriving formulas for \( z0 \) and Delta and choose convenient values when possible (there is some arbitrariness in the choice of Delta and \( z0 \)).
We have:

\[ Z := z = t \Delta + z_0 \]  
\[ Z := z = t \Delta + z_0 \]  
(2.1.3.2.13)

\[ \text{subs (risymbols, Z)} \]  
\[ z = t (rDelta + 1 iDelta) + rz0 + 1 iz0 \]  
(2.1.3.2.14)

\[ \text{Re}(z) = (\text{Re}(\text{rhs}(\%)) \text{ assuming real}) \]  
\[ \Re(z) = rDelta \text{ t} + rz0 \]  
(2.1.3.2.15)

\[ \text{Im}(z) = (\text{Im}(\text{rhs}(\%)) \text{ assuming real}) \]  
\[ \Im(z) = iDelta \text{ t} + iz0 \]  
(2.1.3.2.16)

\[ \text{solve ( (2.1.3.2.15), (2.1.3.2.16)), \{t, iDelta\}} \]  
\[ \left\{ iDelta = \frac{rDelta \left( \Im(z) - iz0 \right)}{\Re(z) - rz0}, t = \frac{\Re(z) - rz0}{rDelta} \right\} \]  
(2.1.3.2.17)

Then, \( i\Delta \) is different from zero only if \( \Im(z) - iz0 \neq 0 \), the real variable \( t \) is given by

\[ t = \frac{\Re(z) - rz0}{rDelta} \], and all of \( r\Delta \), \( rz0 \) and \( iz0 \) are arbitrary, which amounts to say that

- "The initial point \( z0 \) could be anyone provided it is not a singularity, and the value of \( r\Delta \) can be chosen at will - note however that this value will be taken into account when computing initial values at \( z0 \) for dsolve’s procedure"

So take \( z0 \) within 1/2 of the radius of convergence, with imaginary part

\[ z0_val := \frac{1}{2} \cdot \min(\text{abs}(\text{op}(-2, AF)), \text{abs}(\text{op}(-1, AF))) + i \cdot \frac{1}{2} \cdot \min(\text{abs}(\text{op}(-2, AF)), \text{abs}(\text{op}(-1, AF))) \]

\[ z0_val := \frac{9}{25} + \frac{91}{25} \]  
(2.1.3.2.18)

\[ z0_Delta := \{rDelta = 1, iz0 = \text{Im}(z0_val), rz0 = \text{Re}(z0_val)\} \]

\[ z0_Delta := \left\{ iz0 = \frac{9}{25}, rDelta = 1, rz0 = \frac{9}{25} \right\} \]  
(2.1.3.2.19)

\[ \text{eval (2.1.3.2.17), z0_Delta union \{z = z_val\}} \]  
\[ \left\{ iDelta = -\frac{36}{59}, t = \frac{59}{100} \right\} \]  
(2.1.3.2.20)

\[ z0_Delta := z0_Delta \text{ union select (has, (2.1.3.2.20), iDelta)} \]

\[ z0_Delta := \left\{ iDelta = -\frac{36}{59}, iz0 = \frac{9}{25}, rDelta = 1, rz0 = \frac{9}{25} \right\} \]  
(2.1.3.2.21)

\[ Delta_val := \text{subs(z0_Delta, rDelta + 1 iDelta)} \]

\[ Delta_val := 1 - \frac{361}{59} \]  
(2.1.3.2.22)

\[ t_val := \text{select (has, (2.1.3.2.20), t)[1]} \]

\[ t_val := t = \frac{59}{100} \]  
(2.1.3.2.23)

\[ \text{evalf (\%)} \]  
(2.1.3.2.24)
\[ t = 0.590000000000000 \quad (2.1.3.2.24) \]

So, the value of \( z_0 \) is 1/10, the evaluating point is \( z = 3/4 \), that implies on integrating the \( t \) variable from \( t = 0 \) to \( t = 13/20 \), and the values of \( r_{z_0}, i_{z_0}, r_{\Delta}, i_{\Delta} \) are, respectively:

The procedure to use with dsolve/numeric is the one we get from dproc by substituting the values of all the relevant variables \( r_P, i_P, z_{0_{\Delta}} \)

\[
> \text{thisproc := subs}(r_P, i_P, z_{0_{\Delta}} \text{, eval(dproc)})
\]

\[
\text{thisproc := proc}(N, t, Y, YP)
\]

\[
\text{local } rs1, is1, rs2, is2, rs3, is3, rs4, is4, rs5, is5, rs6, is6, rs7, is7, rs8, is8, rs9, is9, rs10, is10, rs11, is11, rs12, is12, rs13, is13, rs14, is14, rs15, is15, rs16, is16, rs17, is17, rs18, is18, rs19, is19, rs20, is20, rs21, is21, rs22, is22, rs23, is23, rs24, is24, rs25, is25, rs26, is26, rs27, is27, rs28, is28, rs29, is29;
\]

\[
rs1 := t + 9/25;
\]

\[
is1 := -36/59 * t + 9/25;
\]

\[
rs2 := rs1 / (is1^2 + rs1^2);
\]

\[
is2 := -is1 / (is1^2 + rs1^2);
\]

\[
rs3 := -is1^2 + rs1^2;
\]

\[
is3 := 2 * is1 * rs1;
\]

\[
rs4 := 5/2;
\]

\[
is4 := -1/2;
\]

\[
rs5 := -rs1 + 18/25;
\]

\[
is5 := -is1;
\]

\[
rs6 := rs1 - 1;
\]

\[
is6 := rs1;
\]

\[
rs7 := -11/2;
\]

\[
is7 := 1/2;
\]

\[
rs8 := -8;
\]

\[
is8 := 1;
\]

\[
rs9 := 0;
\]

\[
is9 := -1;
\]

\[
rs10 := 0;
\]

\[
is10 := -1;
\]

\[
rs11 := 7/2;
\]

\[
is11 := -3/2;
\]

\[
rs12 := -141/118 * Y[1] - 167/118 * Y[5];
\]

\[
is12 := 167/118 * Y[1] - 141/118 * Y[5];
\]

\[
rs13 := rs5 / (is5^2 + rs5^2);
\]

\[
is13 := -is5 / (is5^2 + rs5^2);
\]

\[
rs14 := rs6 / (is6^2 + rs6^2);
\]
\[ is_{14} := -is_{6}/(is_{6}^2 + rs_{6}^2); \]
\[ rs_{15} := 18/25*rs_{9}; \]
\[ is_{15} := 18/25*is_{9}; \]
\[ rs_{16} := 18/25*rs_{11}; \]
\[ is_{16} := 18/25*is_{11}; \]
\[ rs_{17} := -is_{1}*is_{7} + rs_{1}*rs_{7}; \]
\[ is_{17} := is_{1}*rs_{7} + is_{7}*rs_{1}; \]
\[ rs_{18} := -is_{3}*is_{8} + rs_{3}*rs_{8}; \]
\[ is_{18} := is_{3}*rs_{8} + is_{8}*rs_{3}; \]
\[ rs_{19} := -18/25*rs_{10}; \]
\[ is_{19} := -18/25*is_{10}; \]
\[ rs_{20} := 1+rs_{15}+rs_{17}; \]
\[ is_{20} := is_{15}+is_{17}; \]
\[ rs_{21} := 9/2 + rs_{16}; \]
\[ is_{21} := -1/2 + is_{16}; \]
\[ rs_{22} := -is_{1}*is_{21} + rs_{1}*rs_{21}; \]
\[ is_{22} := is_{1}*rs_{21} + is_{21}*rs_{1}; \]
\[ rs_{23} := -is_{20}*Y[6] + rs_{20}*Y[2]; \]
\[ is_{23} := is_{20}*Y[2] + rs_{20}*Y[6]; \]
\[ rs_{24} := rs_{23} + rs_{12}; \]
\[ is_{24} := is_{23} + is_{12}; \]
\[ rs_{25} := rs_{18} + rs_{19} + rs_{22}; \]
\[ is_{25} := is_{18} + is_{19} + is_{22}; \]
\[ rs_{26} := rs_{4}*(36/59*is_{24} + rs_{24}) - is_{4}*(-36/59*rs_{24} + is_{24}); \]
\[ is_{26} := rs_{4}*(-36/59*rs_{24} + is_{24}) - (-36/59*is_{24} - rs_{24}) \]
\[ *is_{4}; \]
\[ rs_{27} := -is_{25}*Y[7] + rs_{25}*Y[3]; \]
\[ is_{27} := is_{25}*Y[3] + rs_{25}*Y[7]; \]
\[ rs_{28} := rs_{27} + rs_{26}; \]
\[ is_{28} := is_{27} + is_{26}; \]
\[ rs_{29} := -36/59*((-is_{2}*rs_{28} - is_{28}*rs_{2})*is_{14} - rs_{14}*is_{2} \]
\[ *is_{28} - rs_{2}*rs_{28})) *is_{13} + 36/59*((is_{2}*is_{28} - rs_{2}*rs_{28}) \]
\[ *is_{14} - rs_{14}*(is_{2}*rs_{28} + is_{28}*rs_{2})) *rs_{13} - ((is_{2}*is_{28} - rs_{2} \]
\[ *rs_{28})*is_{14} - rs_{14}*(is_{2}*rs_{28} + is_{28}*rs_{2}) *is_{13} + ((is_{2}*rs_{28} \]
\[ + is_{28}*rs_{2})*is_{14} + rs_{14}*(is_{2}*is_{28} - rs_{2}*rs_{28})) *rs_{13}; \]
\[ is_{29} := -36/59*((-is_{2}*is_{28} + rs_{2}*rs_{28})*is_{14} + rs_{14}*(is_{2} \]
\[ *rs_{28} + is_{28}*rs_{2}))*is_{13} - 36/59*((is_{2}*rs_{28} + is_{28}*rs_{2})*is_{14} \]
\[ + rs_{14}*(is_{2}*is_{28} - rs_{2}*rs_{28})) *rs_{13} + ((is_{2}*rs_{28} + is_{28}*rs_{2}) \]
* is14 + rs14 * (is2 * is28 − rs2 * rs28) * is13 + ((is2 * is28 − rs2 * rs28) * is14 − rs14 * (is2 * rs28 + is28 * rs2)) * rs13;
YP[3] := rs29;
YP[7] := is29;
YP[1] := Y[2];
YP[6] := Y[7];

end

Now we need the values of F1 and its 1st and 2nd derivatives at z0, to be used as starting point by dsolve. Note however that, after changing variables z = S + Mt, we have

> [f(z), diff(f(z), z), diff(f(z), z, z)]: % := PDEtools[dchange](\{z = S + t * M, f(z) = F(t)\}, %, [t, F(t)], known = all)

\[
f(z) = F(t), \frac{df}{dz}(z) = \frac{dF(t)}{\Delta}, \frac{d^2f}{dz^2}(z) = \frac{d^2F(t)}{\Delta^2}
\]  \hspace{1cm} (2.1.3.2.26)

where on the lhs f(z) represent \( F_1(a, b1, b2, c, x, z) \) and on the rhs F(t) represents \( F_1(a, b1, b2, c, x, t) \) so to compute \([F(t), \dot{F}(t), \ddot{F}(t)]\) we compute

> convert(solve(convert(\[f(z) = F(t), \frac{df}{dz}(z) = \frac{dF(t)}{\Delta}, \frac{d^2f}{dz^2}(z) = \frac{d^2F(t)}{\Delta^2}\]), D), convert([F(t), \dot{F}(t), \ddot{F}(t)], D), diff)\)

\[
\begin{align*}
F(t) &= f(z), \quad \frac{dF(t)}{dt} = \left( \frac{df}{dz}(z) \right) \Delta, \\
\frac{d^2F(t)}{dt^2} &= \left( \frac{d^2f}{dz^2}(z) \right) \Delta^2 \\
\end{align*}
\]  \hspace{1cm} (2.1.3.2.27)

In this particular example, Delta = 1 so this observation is not relevant.

We then have

> DF := subs(z = z0, Delta = Delta_val, \[F1, Delta \cdot \text{diff}(F1, z), \Delta^2 \cdot \text{diff}(F1, z, z)\])

\[
DF := \left[ F_1(a, b1, b2, c, x, z0), \left( 1 - \frac{361}{59} \right) \frac{d}{dz0} F_1(a, b1, b2, c, x, z0), \right.
\]
\[
\left. \left( \frac{2185}{3481} - \frac{721}{59} \right) \frac{d^2}{dz0^2} F_1(a, b1, b2, c, x, z0) \right]
\]  \hspace{1cm} (2.1.3.2.28)

> subs(z0 = z0_val, eval(DF, P))

Hypergeometric: case: a = c
Hypergeometric: case: a = c
Hypergeometric: case: a = c
Hypergeometric: case: a = c

\[
\begin{align*}
&\int_0^1 F_0 \left( \begin{array}{c} 2 + i; \frac{18}{25} \end{array} \right) F_0 \left( \begin{array}{c} 3 \ 2 \ - \frac{1}{2} \ 25 \end{array} \right) + \frac{9 \ 1}{25} \right), \left( \frac{141}{118} - \frac{1671}{118} \right) \int_0^1 F_0 \left( \begin{array}{c} 2 \end{array} \right) \\
&\int_0^1 F_0 \left( \begin{array}{c} 5 \ 2 \ - \frac{1}{2} \ 25 \end{array} \right) + \frac{9 \ 1}{25} \right), \left( \frac{1697}{6962} - \frac{19238}{3481} \right) \\
\end{align*}
\]  \hspace{1cm} (2.1.3.2.29)
\[ iF_{\frac{1}{2}} \left( 2 + \frac{18}{25} \right) iF_{\frac{7}{2} - \frac{1}{2} \frac{9}{25} + \frac{9}{25}} \]

> \text{DFval} := \text{evalf}(2.1.3.2.29)
\text{DFval} := [-8.14736299180347 + 24.8904906590253 I, 2.70043561611901 (2.1.3.2.30)
\quad + 66.0074590509333 I, 72.6426963778800 + 258.691598570864 I]

> \text{ini} := \text{Array}([\text{Re}(\text{DFval}[1]), \text{Re}(\text{DFval}[2]), \text{Re}(\text{DFval}[3]), 0, \text{Im}(\text{DFval}[1]),
\quad \text{Im}(\text{DFval}[2]), \text{Im}(\text{DFval}[3]), 0], \text{datatype} = \text{float}[8]);
\text{ini} := [-8.14736299180347, 2.70043561611901, 72.6426963778800, 0,
\quad 24.8904906590253, 66.0074590509333, 258.691598570864, 0.]

> \text{dsol} := \text{dsolve(numeric, procedure = thisproc, initial = ini, start = 0, procvars}
\quad = [rF1(t), rDF1(t), rDDF1(t), rdummy(t), iF1(t), iDF1(t), iDDF1(t),
\quad idummy(t))];
\text{dsol} := \text{proc}(x_{\text{rkf45}}) \ldots \text{end} \quad (2.1.3.2.32)

> \text{dsol(subs(t_val, t = 2, 6)}}\[2, 6]\]
\text{[rF1(t) = 1112.12012136804, iF1(t) = -254.420480401349]} \quad (2.1.3.2.33)

> \text{SFloat(rhs(dsol(subs(t_val, t = 2)) + 1. * dsol(subs(t_val, t = 6))]}\]
\text{1112.12021236803662 - 254.42048040134873 I} \quad (2.1.3.2.34)

> \text{AFval}
\text{1112.12017081988 - 254.420420227346 I} \quad (2.1.3.2.35)

> \text{evalf}[10](\%\% -\%)
\quad 0.000041 - 0.0000602 I \quad (2.1.3.2.36)

Construct now the procedure used in the code by replacing a,b1,b2,c,x by p1, p2, p3, p4, p5

Concatenated Taylor series expansions covering the whole complex plane

\textbf{Subproducts}

\textbf{Improvements in the numerical evaluation of hypergeometric functions}

\textbf{Evalf: an organized structure to implement the numerical evaluation of special functions in general}

\textit{Description}

- Evalf is both a command and a package of commands for the \textit{numerical} evaluation of mathematical expressions and functions, \textit{numerical} experimentation, and fast development of \textit{numerical} algorithms, taking advantage of the advanced \textit{symbolic} capabilities of the Maple computer algebra system. This kind of \textit{numerical/symbolic} environment is increasingly
relevant nowadays, when rather complicated mathematical expressions and advanced special functions, as for instance is the case of the Heun and Appell functions, appear more and more in the modeling of problems in science.

- The Evalf environment is also an excellent helper for understanding how numerical algorithms work, placing some of the typical numerical approaches used in the literature at the tip of your fingers, in a flexible and friendly manner.

- The Evalf command allows, among other things, for the indication of different numerical methods to evaluate the mathematical functions involved in an algebraic expression. In this version, Evalf implements optional arguments for the 10 Heun and 4 Appell functions. For anything else Evalf works the same as the standard evalf command. The options implemented for numerical evaluation, generally speaking, are divided into four groups:
  - restrictive options;
  - numerical methods options;
  - management options;
  - information options;

- As a package, Evalf contains the following commands:

  - Add
  - Evalb
  - GenerateRecurrence
  - PairwiseSummation
  - QuadrantNumbers
  - Singularities
  - Zoom

**Brief description of the commands of the Evalf package**

- **Add** accepts a procedure with a formula that depends on one integer parameter, say \( n \), or two of them, and numerically evaluates the formula adding it from \( n = 0 \) until the result converges with the current value of Digits, or the value \( n = 10000 \) Digits is reached.

- **Evalb** works as evalb but can handle boolean expressions involving the functions as And, Or, Not and interpreting the integer types integer, posint, negint, nonposint, nonnegint, even and odd in an extended sense to include floats, for example: 1. is considered of type integer and \(-4\) is considered of type negint and even.

- **GenerateRecurrence** accepts a mathematical function (currently working only with the four Appell functions) and returns a procedure to compute the \( n \)th coefficient of a power series expansion of the given function around the origin as a function of the \( n - 1 \) previous coefficients, where \( n \) depends on the function given.

- **PairwiseSummation** accepts a one-dimensional Array or a procedure of one argument, say \( U \) and performs a pairwise summation of \( U(j) \) for \( j \) from given \( m \) to \( n \), or from the lower to the upper bound of the given Array. Pairwise summation is a technique to add floating-point numbers that significantly reduces the accumulated round-off error if compared to adding the numbers one at a time.

- **QuadrantNumbers** accepts a complex number and returns the quadrant of the complex plane where the number is located, or it accepts a list of numbers, and returns an Array of four lists, corresponding to each quadrant, of random complex numbers that are around the numbers indicated.

- **Singularities** accepts a Heun or Appell function, for instance with numerical values for the function's parameters, and returns the singularities of the linear ODE satisfied by the given
function.

- **Zoom** is used to zoom within the last plot computed using Evalf for concatenated Taylor expansions used when performing numerical computations of Heun or Appell functions.

As usual, you can load the Evalf package using the `with` command, or invoke Evalf commands using the long form, e.g. as in `Evalf:-Add`.

### To be done

- Simplification of Appell functions - use ladder differential operators
- Expressing integrals in terms of Appell functions
- Expressing solutions to differential equations in terms of Appell functions