Subject: Confluent Hypergeometric Function of the First Kind
Posted by noahh.schwartz on Wed, 20 Feb 2008 15:44:49 GMT
View Forum Message <> Reply to Message

Hello everyone,

I am looking for the Confluent Hypergeometric Function of the First Kind in the IDL Math Library but it does not seem to be implemented!

I would like to use a function similar to the Hypergeometric1F1[a, b, z] of Mathematica [http://reference.wolfram.com/mathematica/ref/Hypergeometric1F1.html].

I have not found what I was looking for, and so decided to try to code it my self... [sigh...]. Beeing a fresh beginner in IDL this is a hard task!

Would anybody know how to code an infinite series expansion like the Hypergeometric1F1?

Thank you in advance for your time!
Noah

Subject: Re: Confluent Hypergeometric Function of the First Kind
Posted by Spon on Fri, 22 Feb 2008 12:46:29 GMT
View Forum Message <> Reply to Message

On Feb 22, 10:36 am, noahh.schwa...@gmail.com wrote:
> On 21 fév, 17:26, Dan Larson <dlar...@aecom.yu.edu> wrote:
>
>
>
>> On Feb 21, 9:30 am, Spon <christoph.b...@gmail.com> wrote:
>
>>> On Feb 20, 3:44 pm, noahh.schwa...@gmail.com wrote:
>
>>>> Hello everyone,
>
>>>> I am looking for the ConfluentHypergeometricFunctionof the First
>>>> Kind in theIDLMath Library but it does not seem to be implemented!
>
>>>> I would like to use afunctionsimilar to the Hypergeometric1F1[a, b,
>>>> z] of Mathematica [http://reference.wolfram.com/mathematica/ref/
>>>> Hypergeometric1F1.html].
>
>>>> I have not found what I was looking for, and so decided to try to code
>>>> it my self... [sigh...]. Beeing a fresh beginner inIDLthis is a hard

>>>> task!
>
>>>> Would anybody know how to code an infinite series expansion like the
>>>> Hypergeometric1F1?
>
>>>> Thank you in advance for your time!
>>>> Noah
>
>>> Noah,
>
>>> here's my attempt. It accepts only scalar inputs for A and B, while Z
>>> can be a vector. I've tested it for the examples on the mathematica
>>> site and it seems to give correct results, and works correctly for
>>> complex input too as far as I can tell. 'Precision' is an input
>>> variable to specify how close two successive iterations have to be
>>> before thefunctionassumes they are the same and aborts the while
>>> loop. Default is 7 (i.e. stop when results differ by 10^-7 or less).
>>> If you're finding this programme is running very slow, try decreasing
>>> the precision (I was surprised how fast it runs despite the while
>>> loop, actually!)
>
>>> Ideally the input parameters should all be double precision before you
>>> make the call to the funcion, but thefunctionconverts them if
>>> they're not.
>
>>> If you want all your inputs to be vectors (not just Z), I'm sure it
>>> can be done, but it'd be a bit more complicated. :-)
>
>>> Take care,
>>> Chris
>
>>> FUNCTIONHYPERGEOMETRICONEFONE, A, B, Z, $
>>> PRECISION = Precision, $
>>> K = K ; K is an output parameter to count No. of WHILE loops
>>> performed.
>
>>> ; References:
>>> ; http://reference.wolfram.com/mathematica/ref/Hypergeometric1 F1.html
>>> ; http://en.wikipedia.org/wiki/Confluent_hypergeometric_functi on
>
>>> IF N_PARAMS() NE 3 THEN MESSAGE, 'Must input A, B & Z as 3 input
>>> parameters.'
>>> IF N_ELEMENTS(A) GT 1 THEN MESSAGE, 'Variable A must be a scalar.'
>>> IF N_ELEMENTS(B) GT 1 THEN MESSAGE, 'Variable B must be a scalar.'
>
>>> A *= 1.0D ; Double precision or double complex scalar
>>> B *= 1.0D ; Double precision or double complex scalar
>>> Z *= 1.0D ; Double precision or double complex scalar or vector

>>> IF N_ELEMENTS(Precision) EQ 0 THEN $
>>>   Precision = 7L ELSE $
>>>   Precision = (LONG(Precision))[0]
>>> Cutoff = 10D^(-1D * Precision) > (MACHAR()).EPS ; Cutoff can't be
>>> smaller than machine accuracy!
>
>>> K = 0L
>>> ThisResult = REPLICATE(0D, N_ELEMENTS(Z))
>>> WHILE (N_ELEMENTS(LastResult) EQ 0) || (MAX(ABS(LastResult -
>>> ThisResult)) GT Cutoff) DO BEGIN
>>>   LastResult = ThisResult
>>>   AK = GAMMA(A + K) / GAMMA(A) ; Define (A)k
>>>   BK = GAMMA(B + K) / GAMMA(B) ; Define (B)k
>>>   F = (AK * Z^K) / (BK * FACTORIAL(K)) ; Evaluatefunction.
>>>   ThisResult = LastResult + F
>>>   K += 1
>>> ENDWHILE ; Until result is good to Precision
>
>>>   ; Error if not enough while loops to give accurate results.
>>> IF K LE 1 THEN MESSAGE, 'Functionfailed. Try greater precision.'
>
>>> RETURN, ThisResult
>>> END- Hide quoted text -
>
>>> - Show quoted text -
>
>> Noah,
>
>> Here is my implementation, both of the series expansion of thehypergeometricfunctionand the integral representation. Depending on
>> the parameters, I have found that one may be more stable than the
>> other. Both of these are based on Arfken and Weber, Mathematical
>> Methods for Physicists.
>
>> best,
>> dan
>
>> ; chss
>> ; confluenthypergeometricseries solution
>> ; calculates the solution to the differential equation:
>> ;   xy''(x) + (c - x)y'(x) - ay(x) = 0
>> ; (see Afken and Weber, p. 801-2)
>> ;
>> ; inputs:
>> ;   n: number of terms to calculate. Due limits inIDLarchitecture n
>> must be between 1 and 170.
>> ;   a, c: vector of constants - see above. They MUST have the same
>> number of elelments

```
>>  ;   x: input value
>>  ;
>>  ; outputs:
>>  ;   y: output vector
>>  ;
>>  ; Dan Larson, 2007.10.11
>
>>  functionchss, n, a, c, x
>>      y = DBLARR((SIZE(a))[1])
>>      ; first term of series expansion is 1, so we initialize the output
>>      y[*] = 1
>
>>      FOR i = 0, (SIZE(a))[1]-1 DO BEGIN
>>          ; initialize constant series products
>>          a_p = 1
>>          c_p = 1
>>          FOR j = 0, n DO BEGIN
>
>>            a_p *= (a[i] + j)
>>            c_p *= (c[i] + j)
>>            d = FACTORIAL(j + 1)
>
>>            y[i] += (a_p/c_p)*(x^(j + 1))/d
>
>>          ENDFOR
>>      ENDFOR
>>      RETURN, y
>>  END
>
>>  ; chins
>>  ; confluenthypergeometricintegral solution
>>  ; calculates the solution to the differential equation:
>>  ;   xy''(x) + (c - x)y'(x) - ay(x) = 0
>>  ; (see Afken and Weber, p. 801-2)
>>  ;
>>  ; inputs:
>>  ;   a, c: vector constants - see above
>>  ;   x: input value
>>  ;
>>  ; outputs:
>>  ;   y: output vector
>>  ;
>>  ;Dan larson, 2007.10.11
>
>>  Functionchins, a, c, x
>>      ; curve point resolution, change this if your results look like
>>  crap
>>      b = 1000
```

```
>>     ; initialize t vector
>>     t = DINDGEN(b + 1)/b
>>     ; initialize vector of curve heights
>>     h = DBLARR(b + 1)
>>     ; initialize output
>>     y=dblarr(n_elements(c))
>
>>     g1 = GAMMA(c)
>>     g2 = GAMMA(a) * GAMMA(c - a)
>
>>     FOR i = 0, (SIZE(a))[1]-1 DO BEGIN
>>        FOR j = 0, b DO BEGIN
>>          h[j] = exp(x*t[j]) * ((t[j])^(a[i] - 1.0)) * ((1.0 -
>>  t[j])^(c[i] - a[i] - 1.0))
>>          ENDFOR
>
>>        y[i] = (g1[i]/g2[i]) * int_tabulated(t, h, /double)
>>      ENDFOR
>
>>     RETURN, y
>>  END
>
>>  ; chcmp
>>  ; compares the results between the integral and series form of CHF
>>  ;
>>  ; inputs:
>>  ;   a, c: constants
>>  ;   x: input value
>>  ;
>>  ; output:
>>  ;   none
>>  ;  (solution is printed to screen)
>
>>  PRO chcmp, a, c, x, epsilon
>>     y1 = chins(a, c, x)
>
>>     lastVal = 0.0000
>
>>     FOR n = 1, 170 DO BEGIN
>>        y2 = chss( n, a, c, x)
>
>>        IF ( ABS(y1 - y2)/y1 LT epsilon) THEN BEGIN
>>          print, "Number of necessary terms for ep = ", epsilon, " n =
>>  ", n
>>          BREAK
>>        ENDIF
>
>>        IF( ABS(lastVal - y2)/y2 EQ 0) THEN BEGIN
```

```
>>          print, "Series converged before matching integral!"
>>          BREAK
>>        ENDIF
>
>>        lastVal = y2
>>      ENDFOR
>
>>      IF n EQ 169 THEN PRINT, "Equations did not meet!"
>
>>      RETURN
>>  END
>
>  Thank you all for your answers. It really made my day!
>  I've tested the 3 methods for the "Mathematica" example (i.e. for
>  1F1(1,2,[-5..5]))
>  HYPERGEOMETRICONEFONE, CHSS and CHINS seem to work fine for this
>  example. HYPERGEOMETRICONEFONE seems a bit faster than the other ones.
>  Unfortunately I am more interested in evaluating something like 1F1(-.
>  75,1,40) :
>
>  IDL> print,hypergeometriconefone(-0.75D,1D,-40D)
>              NaN
>
>  IDL> print, chins([-0.75D],[1D],-40D)
>              NaN
>
>  IDL> print, chss(169D,[-0.75D],[1D],-40D)
>         17.478776
>
>  The mathematica website gives [http://functions.wolfram.com/
>   webMathematica/FunctionEvaluation.jsp?name=Hypergeometric1F1 ]:

>
>  Thanks,
>  Noah
```

Phew, you're really pushing my little programme to the limit here! ;-)

If you put this line:
Print, [ThisResult, K]
into my programme just before the line with ENDWHILE on it, you'll see
that the function actually gets quite close to the answer before it
succumbs to floating overflow. On my machine, the results of the last
few iterations are:

```
      17.462850       96.000000
      17.015058       97.000000
      17.196383       98.000000
      17.123695       99.000000
```

If you try "print, hypergeometriconefone(-0.75D,1D,-40D, prec=1)", you should get similar results.

I suspect IDL won't really help you out here, at least until someone invents 128-bit quadruple-precision data type :-(
Or, unless you use a better algorithm, I guess. chss seems to be that better algorithm in this case.

Actually, it would help to replace this:
    F = (AK * Z^K) / (BK * FACTORIAL(K)) ; Evaluate function.

with:
    F = (AK / BK) * (Z^K / FACTORIAL(K)) ; Evaluate function.

Though I suspect it's just putting off the inevitable by a few more iterations :-(
Also, there should've been some sort of protection for preventing k from going over 169, which is the limit at which FACTORIAL can work reliably.

I'll try and amend the programme and repost if you're having difficulty piecing my garbled lines together into your copy.

Subject: Re: Confluent Hypergeometric Function of the First Kind
Posted by Spon on Fri, 22 Feb 2008 13:16:22 GMT
View Forum Message <> Reply to Message

On Feb 22, 12:46 pm, Spon <christoph.b...@gmail.com> wrote:
> On Feb 22, 10:36 am, noahh.schwa...@gmail.com wrote:
>
>
>
>> On 21 fév, 17:26, Dan Larson <dlar...@aecom.yu.edu> wrote:
>
>>> On Feb 21, 9:30 am, Spon <christoph.b...@gmail.com> wrote:
>
>>>> On Feb 20, 3:44 pm, noahh.schwa...@gmail.com wrote:
>
>>>> > Hello everyone,
>
>>>> > I am looking for the ConfluentHypergeometricFunctionof the First
>>>> > Kind in theIDLMath Library but it does not seem to be implemented!
>
>>>> > I would like to use afunctionsimilar to the Hypergeometric1F1[a, b,
>>>> > z] of Mathematica [http://reference.wolfram.com/mathematica/ref/
>>>> > Hypergeometric1F1.html].

>
>>>> > I have not found what I was looking for, and so decided to try to code
>>>> > it my self... [sigh...]. Beeing a fresh beginner inIDLthis is a hard
>>>> > task!
>
>>>> > Would anybody know how to code an infinite series expansion like the
>>>> > Hypergeometric1F1?
>
>>>> > Thank you in advance for your time!
>>>> > Noah
>
>>>> Noah,
>
>>>> here's my attempt. It accepts only scalar inputs for A and B, while Z
>>>> can be a vector. I've tested it for the examples on the mathematica
>>>> site and it seems to give correct results, and works correctly for
>>>> complex input too as far as I can tell. 'Precision' is an input
>>>> variable to specify how close two successive iterations have to be
>>>> before thefunctionassumes they are the same and aborts the while
>>>> loop. Default is 7 (i.e. stop when results differ by 10^-7 or less).
>>>> If you're finding this programme is running very slow, try decreasing
>>>> the precision (I was surprised how fast it runs despite the while
>>>> loop, actually!)
>
>>>> Ideally the input parameters should all be double precision before you
>>>> make the call to the funcion, but thefunctionconverts them if
>>>> they're not.
>
>>>> If you want all your inputs to be vectors (not just Z), I'm sure it
>>>> can be done, but it'd be a bit more complicated. :-)
>
>>>> Take care,
>>>> Chris
>
>>>> FUNCTIONHYPERGEOMETRICONEFONE, A, B, Z, $
>>>> PRECISION = Precision, $
>>>> K = K ; K is an output parameter to count No. of WHILE loops
>>>> performed.
>
>>>> ; References:
>>>> ; http://reference.wolfram.com/mathematica/ref/Hypergeometric1 F1.html
>>>> ; http://en.wikipedia.org/wiki/Confluent_hypergeometric_functi on
>
>>>> IF N_PARAMS() NE 3 THEN MESSAGE, 'Must input A, B & Z as 3 input
>>>> parameters.'
>>>> IF N_ELEMENTS(A) GT 1 THEN MESSAGE, 'Variable A must be a scalar.'
>>>> IF N_ELEMENTS(B) GT 1 THEN MESSAGE, 'Variable B must be a scalar.'
>

>>>> A *= 1.0D ; Double precision or double complex scalar
>>>> B *= 1.0D ; Double precision or double complex scalar
>>>> Z *= 1.0D ; Double precision or double complex scalar or vector
>>>> IF N_ELEMENTS(Precision) EQ 0 THEN $
>>>>   Precision = 7L ELSE $
>>>>   Precision = (LONG(Precision))[0]
>>>> Cutoff = 10D^(-1D * Precision) > (MACHAR()).EPS ; Cutoff can't be
>>>> smaller than machine accuracy!
>
>>>> K = 0L
>>>> ThisResult = REPLICATE(0D, N_ELEMENTS(Z))
>>>> WHILE (N_ELEMENTS(LastResult) EQ 0) || (MAX(ABS(LastResult -
>>>> ThisResult)) GT Cutoff) DO BEGIN
>>>>   LastResult = ThisResult
>>>>   AK = GAMMA(A + K) / GAMMA(A) ; Define (A)k
>>>>   BK = GAMMA(B + K) / GAMMA(B) ; Define (B)k
>>>>   F = (AK * Z^K) / (BK * FACTORIAL(K)) ; Evaluatefunction.
>>>>   ThisResult = LastResult + F
>>>>   K += 1
>>>> ENDWHILE ; Until result is good to Precision
>
>>>>   ; Error if not enough while loops to give accurate results.
>>>> IF K LE 1 THEN MESSAGE, 'Functionfailed. Try greater precision.'
>
>>>> RETURN, ThisResult
>>>> END- Hide quoted text -
>
>>>> - Show quoted text -
>
>>> Noah,
>
>>> Here is my implementation, both of the series expansion of thehypergeometricfunctionand
the integral representation. Depending on
>>> the parameters, I have found that one may be more stable than the
>>> other. Both of these are based on Arfken and Weber, Mathematical
>>> Methods for Physicists.
>
>>> best,
>>> dan
>
>>> ; chss
>>> ; confluenthypergeometricseries solution
>>> ; calculates the solution to the differential equation:
>>> ;   xy''(x) + (c - x)y'(x) - ay(x) = 0
>>> ; (see Afken and Weber, p. 801-2)
>>> ;
>>> ; inputs:
>>> ;   n: number of terms to calculate. Due limits inIDLarchitecture n

```
>>>  must be between 1 and 170.
>>>  ;   a, c: vector of constants - see above. They MUST have the same
>>>  number of elelments
>>>  ;   x: input value
>>>  ;
>>>  ; outputs:
>>>  ;   y: output vector
>>>  ;
>>>  ; Dan Larson, 2007.10.11
>
>>>  functionchss, n, a, c, x
>>>     y = DBLARR((SIZE(a))[1])
>>>     ; first term of series expansion is 1, so we initialize the output
>>>     y[*] = 1
>
>>>     FOR i = 0, (SIZE(a))[1]-1 DO BEGIN
>>>        ; initialize constant series products
>>>        a_p = 1
>>>        c_p = 1
>>>        FOR j = 0, n DO BEGIN
>
>>>          a_p *= (a[i] + j)
>>>          c_p *= (c[i] + j)
>>>          d = FACTORIAL(j + 1)
>
>>>          y[i] += (a_p/c_p)*(x^(j + 1))/d
>
>>>        ENDFOR
>>>     ENDFOR
>>>     RETURN, y
>>>  END
>
>>>  ; chins
>>>  ; confluenthypergeometricintegral solution
>>>  ; calculates the solution to the differential equation:
>>>  ;   xy''(x) + (c - x)y'(x) - ay(x) = 0
>>>  ; (see Afken and Weber, p. 801-2)
>>>  ;
>>>  ; inputs:
>>>  ;   a, c: vector constants - see above
>>>  ;   x: input value
>>>  ;
>>>  ; outputs:
>>>  ;   y: output vector
>>>  ;
>>>  ;Dan larson, 2007.10.11
>
>>>  Functionchins, a, c, x
```

```
>>>     ; curve point resolution, change this if your results look like
>>>  crap
>>>     b = 1000
>>>     ; initialize t vector
>>>     t = DINDGEN(b + 1)/b
>>>     ; initialize vector of curve heights
>>>     h = DBLARR(b + 1)
>>>     ; initialize output
>>>     y=dblarr(n_elements(c))
>
>>>     g1 = GAMMA(c)
>>>     g2 = GAMMA(a) * GAMMA(c - a)
>
>>>     FOR i = 0, (SIZE(a))[1]-1 DO BEGIN
>>>        FOR j = 0, b DO BEGIN
>>>          h[j] = exp(x*t[j]) * ((t[j])^(a[i] - 1.0)) * ((1.0 -
>>>  t[j])^(c[i] - a[i] - 1.0))
>>>          ENDFOR
>
>>>        y[i] = (g1[i]/g2[i]) * int_tabulated(t, h, /double)
>>>      ENDFOR
>
>>>     RETURN, y
>>>  END
>
>>>  ; chcmp
>>>  ; compares the results between the integral and series form of CHF
>>>  ;
>>>  ; inputs:
>>>  ;   a, c: constants
>>>  ;   x: input value
>>>  ;
>>>  ; output:
>>>  ;   none
>>>  ;  (solution is printed to screen)
>
>>>  PRO chcmp, a, c, x, epsilon
>>>     y1 = chins(a, c, x)
>
>>>     lastVal = 0.0000
>
>>>     FOR n = 1, 170 DO BEGIN
>>>        y2 = chss( n, a, c, x)
>
>>>        IF ( ABS(y1 - y2)/y1 LT epsilon) THEN BEGIN
>>>          print, "Number of necessary terms for ep = ", epsilon, " n =
>>>  ", n
>>>          BREAK
```

```
>>>       ENDIF
>
>>>       IF( ABS(lastVal - y2)/y2 EQ 0) THEN BEGIN
>>>         print, "Series converged before matching integral!"
>>>         BREAK
>>>       ENDIF
>
>>>       lastVal = y2
>>>     ENDFOR
>
>>>     IF n EQ 169 THEN PRINT, "Equations did not meet!"
>
>>>     RETURN
>>>  END
>
>>  Thank you all for your answers. It really made my day!
>>  I've tested the 3 methods for the "Mathematica" example (i.e. for
>>  1F1(1,2,[-5..5]))
>>  HYPERGEOMETRICONEFONE, CHSS and CHINS seem to work fine for this
>>  example. HYPERGEOMETRICONEFONE seems a bit faster than the other ones.
>>  Unfortunately I am more interested in evaluating something like 1F1(-.
>>  75,1,40) :
>
>>  IDL> print,hypergeometriconefone(-0.75D,1D,-40D)
>>              NaN
>
>>  IDL> print, chins([-0.75D],[1D],-40D)
>>              NaN
>
>>  IDL> print, chss(169D,[-0.75D],[1D],-40D)
>>         17.478776
>
>>  The mathematica website gives [http://functions.wolfram.com/
>>   webMathematica/FunctionEvaluation.jsp?name=Hypergeometric1F1 ]:

>
>>  Thanks,
>>  Noah
>
>  Phew, you're really pushing my little programme to the limit here! ;-)
>
>  If you put this line:
>  Print, [ThisResult, K]
>  into my programme just before the line with ENDWHILE on it, you'll see
>  that the function actually gets quite close to the answer before it
>  succumbs to floating overflow. On my machine, the results of the last
>  few iterations are:
>         17.462850       96.000000
```

> 17.015058 97.000000
> 17.196383 98.000000
> 17.123695 99.000000
>
> If you try "print, hypergeometriconefone(-0.75D,1D,-40D, prec=1)", you
> should get similar results.
>
> I suspect IDL won't really help you out here, at least until someone
> invents 128-bit quadruple-precision data type :-(
> Or, unless you use a better algorithm, I guess. chss seems to be that
> better algorithm in this case.
>
> Actually, it would help to replace this:
> F = (AK * Z^K) / (BK * FACTORIAL(K)) ; Evaluate function.
>
> with:
> F = (AK / BK) * (Z^K / FACTORIAL(K)) ; Evaluate function.
>
> Though I suspect it's just putting off the inevitable by a few more
> iterations :-(
> Also, there should've been some sort of protection for preventing k
> from going over 169, which is the limit at which FACTORIAL can work
> reliably.
>
> I'll try and amend the programme and repost if you're having
> difficulty piecing my garbled lines together into your copy.

The more I stare at this, the more I realise that my guess at
available precision is WAY off the mark :-(
I'll go stare at the 'Sky is falling' page for another while, shall
I? :-P
I need to make sure the precision is determined by the significant
figures, not just an absolute.

Anyway, for now if you get NaN results, just try dropping PRECISION
below 5.
If you want acres of spam, call the function with the debug keyword
set.

e.g.:
Test = HYPERGEOMETRICONEFONE(-0.75d,1d,-40d, prec = 5, k = k, /DEBUG)
Print, 'IDL thinks the result is ' + STRTRIM(Test, 2) + ' and got
there in ' + STRTRIM(k, 2) + ' iterations.'
IDL> IDL thinks the result is 17.145252 and got there in 109
iterations.

As an aside: a) Yes, I know I should learn format codes and do this
properly ;-) and

b) How does IDL work out the linebreak if I use commas instead of plus signs?

Presumably by using "+" I'm joining the whole string together before it gets passed to the PRINT command, whereas PRINT has to figure out what to do with the different inputs if I use "," - but what criteria does it use?

Good luck!
Chris

```
FUNCTION HYPERGEOMETRICONEFONE, A, B, Z, $
 DEBUG = Debug, $
 PRECISION = Precision, $
 K = K ; K is an output parameter to count No. of WHILE loops performed.

DoDebug = KEYWORD_SET(Debug)
IF ~DoDebug THEN ON_ERROR, 2

 ; Mathematica defines Hypergeometric1F1 here:
 ;  http://reference.wolfram.com/mathematica/ref/Hypergeometric1 F1.html
 ;  http://en.wikipedia.org/wiki/Confluent_hypergeometric_functi on
IF N_PARAMS() NE 3 THEN MESSAGE, 'Must input A, B & Z as 3 input parameters.'
IF N_ELEMENTS(A) GT 1 THEN MESSAGE, 'Variable A must be a scalar.'
IF N_ELEMENTS(B) GT 1 THEN MESSAGE, 'Variable B must be a scalar.'

A *= 1.0D ; Double precision or double complex scalar
B *= 1.0D ; Double precision or double complex scalar
Z *= 1.0D ; Double precision or double complex vector
IF N_ELEMENTS(Precision) EQ 0 THEN $
  Precision = 5L ELSE $
    Precision = (LONG(Precision))[0] < 5
Cutoff = 10D^(-1D * Precision) > 16D * (MACHAR()).EPS

K = 0L
ThisResult = REPLICATE(0D, N_ELEMENTS(Z))
 ; Logical OR (||) used so this construct works even when LastResult not defined.
 ; (This doesn't work with bitwise OR)
WHILE (N_ELEMENTS(LastResult) EQ 0) || (MAX(ABS(LastResult - ThisResult)) GT Cutoff) DO BEGIN
    LastResult = ThisResult
    AK = GAMMA(A + K) / GAMMA(A) ; Define (A)k
    BK = GAMMA(B + K) / GAMMA(B) ; Define (B)k
    F = (AK / BK) * (Z^K / FACTORIAL(K)) ; Evaluate function.
    ThisResult = LastResult + F
```

```
    K += 1
    IF K GT 169 THEN MESSAGE, 'Failure to converge.'
    IF DoDebug THEN Print, [ThisResult, K]
ENDWHILE ; Until result is good to Precision

 ; Error if not enough while loops to give accurate results.
IF K LE 1 THEN MESSAGE, 'Function failed. Try greater precision.'

RETURN, ThisResult
END
```

Subject: Re: Confluent Hypergeometric Function of the First Kind
Posted by Dan Larson on Fri, 22 Feb 2008 15:30:08 GMT
View Forum Message <> Reply to Message

On Feb 22, 8:16 am, Spon <christoph.b...@gmail.com> wrote:
> On Feb 22, 12:46 pm, Spon <christoph.b...@gmail.com> wrote:
>
>
>
>
>
>> On Feb 22, 10:36 am, noahh.schwa...@gmail.com wrote:
>
>>> On 21 fév, 17:26, Dan Larson <dlar...@aecom.yu.edu> wrote:
>
>>>> On Feb 21, 9:30 am, Spon <christoph.b...@gmail.com> wrote:
>
>>>> > On Feb 20, 3:44 pm, noahh.schwa...@gmail.com wrote:
>
>>>> > > Hello everyone,
>
>>>> > > I am looking for the ConfluentHypergeometricFunctionof the First
>>>> > > Kind in theIDLMath Library but it does not seem to be implemented!
>
>>>> > > I would like to use afunctionsimilar to the Hypergeometric1F1[a, b,
>>>> > > z] of Mathematica [http://reference.wolfram.com/mathematica/ref/
>>>> > > Hypergeometric1F1.html].
>
>>>> > > I have not found what I was looking for, and so decided to try to code
>>>> > > it my self... [sigh...]. Beeing a fresh beginner inIDLthis is a hard
>>>> > > task!
>
>>>> > > Would anybody know how to code an infinite series expansion like the
>>>> > > Hypergeometric1F1?
>
>>>> > > Thank you in advance for your time!

>>>> > > Noah
>
>>>> > Noah,
>
>>>> > here's my attempt. It accepts only scalar inputs for A and B, while Z
>>>> > can be a vector. I've tested it for the examples on the mathematica
>>>> > site and it seems to give correct results, and works correctly for
>>>> > complex input too as far as I can tell. 'Precision' is an input
>>>> > variable to specify how close two successive iterations have to be
>>>> > before thefunctionassumes they are the same and aborts the while
>>>> > loop. Default is 7 (i.e. stop when results differ by 10^-7 or less).
>>>> > If you're finding this programme is running very slow, try decreasing
>>>> > the precision (I was surprised how fast it runs despite the while
>>>> > loop, actually!)
>
>>>> > Ideally the input parameters should all be double precision before you
>>>> > make the call to the funcion, but thefunctionconverts them if
>>>> > they're not.
>
>>>> > If you want all your inputs to be vectors (not just Z), I'm sure it
>>>> > can be done, but it'd be a bit more complicated. :-)
>
>>>> > Take care,
>>>> > Chris
>
>>>> >FUNCTIONHYPERGEOMETRICONEFONE, A, B, Z, $
>>>> > PRECISION = Precision, $
>>>> > K = K ; K is an output parameter to count No. of WHILE loops
>>>> > performed.
>
>>>> > ; References:
>>>> > ; http://reference.wolfram.com/mathematica/ref/Hypergeometric1 F1.html
>>>> > ; http://en.wikipedia.org/wiki/Confluent_hypergeometric_functi on
>
>>>> > IF N_PARAMS() NE 3 THEN MESSAGE, 'Must input A, B & Z as 3 input
>>>> > parameters.'
>>>> > IF N_ELEMENTS(A) GT 1 THEN MESSAGE, 'Variable A must be a scalar.'
>>>> > IF N_ELEMENTS(B) GT 1 THEN MESSAGE, 'Variable B must be a scalar.'
>
>>>> > A *= 1.0D ; Double precision or double complex scalar
>>>> > B *= 1.0D ; Double precision or double complex scalar
>>>> > Z *= 1.0D ; Double precision or double complex scalar or vector
>>>> > IF N_ELEMENTS(Precision) EQ 0 THEN $
>>>> > Precision = 7L ELSE $
>>>> > Precision = (LONG(Precision))[0]
>>>> > Cutoff = 10D^(-1D * Precision) > (MACHAR()).EPS ; Cutoff can't be
>>>> > smaller than machine accuracy!
>

>>>> > K = 0L
>>>> > ThisResult = REPLICATE(0D, N_ELEMENTS(Z))
>>>> > WHILE (N_ELEMENTS(LastResult) EQ 0) || (MAX(ABS(LastResult -
>>>> > ThisResult)) GT Cutoff) DO BEGIN
>>>> > LastResult = ThisResult
>>>> > AK = GAMMA(A + K) / GAMMA(A) ; Define (A)k
>>>> > BK = GAMMA(B + K) / GAMMA(B) ; Define (B)k
>>>> > F = (AK * Z^K) / (BK * FACTORIAL(K)) ; Evaluatefunction.
>>>> > ThisResult = LastResult + F
>>>> > K += 1
>>>> > ENDWHILE ; Until result is good to Precision
>
>>>> > ; Error if not enough while loops to give accurate results.
>>>> > IF K LE 1 THEN MESSAGE, 'Functionfailed. Try greater precision.'
>
>>>> > RETURN, ThisResult
>>>> > END- Hide quoted text -
>
>>>> > - Show quoted text -
>
>>>> Noah,
>
>>>> Here is my implementation, both of the series expansion of thehypergeometricfunctionand the integral representation. Depending on
>>>> the parameters, I have found that one may be more stable than the
>>>> other. Both of these are based on Arfken and Weber, Mathematical
>>>> Methods for Physicists.
>
>>>> best,
>>>> dan
>
>>>> ; chss
>>>> ; confluenthypergeometricseries solution
>>>> ; calculates the solution to the differential equation:
>>>> ; xy''(x) + (c - x)y'(x) - ay(x) = 0
>>>> ; (see Afken and Weber, p. 801-2)
>>>> ;
>>>> ; inputs:
>>>> ; n: number of terms to calculate. Due limits inIDLarchitecture n
>>>> must be between 1 and 170.
>>>> ; a, c: vector of constants - see above. They MUST have the same
>>>> number of elelments
>>>> ; x: input value
>>>> ;
>>>> ; outputs:
>>>> ; y: output vector
>>>> ;
>>>> ; Dan Larson, 2007.10.11

```
>
>>>>  functionchss, n, a, c, x
>>>>     y = DBLARR((SIZE(a))[1])
>>>>     ; first term of series expansion is 1, so we initialize the output
>>>>     y[*] = 1
>
>>>>     FOR i = 0, (SIZE(a))[1]-1 DO BEGIN
>>>>       ; initialize constant series products
>>>>       a_p = 1
>>>>       c_p = 1
>>>>       FOR j = 0, n DO BEGIN
>
>>>>         a_p *= (a[i] + j)
>>>>         c_p *= (c[i] + j)
>>>>         d = FACTORIAL(j + 1)
>
>>>>         y[i] += (a_p/c_p)*(x^(j + 1))/d
>
>>>>       ENDFOR
>>>>     ENDFOR
>>>>     RETURN, y
>>>>  END
>
>>>>  ; chins
>>>>  ; confluenthypergeometricintegral solution
>>>>  ; calculates the solution to the differential equation:
>>>>  ;   xy''(x) + (c - x)y'(x) - ay(x) = 0
>>>>  ; (see Afken and Weber, p. 801-2)
>>>>  ;
>>>>  ; inputs:
>>>>  ;   a, c: vector constants - see above
>>>>  ;   x: input value
>>>>  ;
>>>>  ; outputs:
>>>>  ;   y: output vector
>>>>  ;
>>>>  ;Dan larson, 2007.10.11
>
>>>>  Functionchins, a, c, x
>>>>     ; curve point resolution, change this if your results look like
>>>>  crap
>>>>     b = 1000
>>>>     ; initialize t vector
>>>>     t = DINDGEN(b + 1)/b
>>>>     ; initialize vector of curve heights
>>>>     h = DBLARR(b + 1)
>>>>     ; initialize output
>>>>     y=dblarr(n_elements(c))
```

```
>
>>>>     g1 = GAMMA(c)
>>>>     g2 = GAMMA(a) * GAMMA(c - a)
>
>>>>     FOR i = 0, (SIZE(a))[1]-1 DO BEGIN
>>>>       FOR j = 0, b DO BEGIN
>>>>         h[j] = exp(x*t[j]) * ((t[j])^(a[i] - 1.0)) * ((1.0 -
>>>>  t[j])^(c[i] - a[i] - 1.0))
>>>>         ENDFOR
>
>>>>       y[i] = (g1[i]/g2[i]) * int_tabulated(t, h, /double)
>>>>      ENDFOR
>
>>>>     RETURN, y
>>>>  END
>
>>>>  ; chcmp
>>>>  ; compares the results between the integral and series form of CHF
>>>>  ;
>>>>  ; inputs:
>>>>  ;   a, c: constants
>>>>  ;   x: input value
>>>>  ;
>>>>  ; output:
>>>>  ;   none
>>>>  ;  (solution is printed to screen)
>
>>>>  PRO chcmp, a, c, x, epsilon
>>>>     y1 = chins(a, c, x)
>
>>>>     lastVal = 0.0000
>
>>>>     FOR n = 1, 170 DO BEGIN
>>>>       y2 = chss( n, a, c, x)
>
>>>>       IF ( ABS(y1 - y2)/y1 LT epsilon) THEN BEGIN
>>>>         print, "Number of necessary terms for ep = ", epsilon, " n =
>>>>  ", n
>>>>         BREAK
>>>>       ENDIF
>
>>>>       IF( ABS(lastVal - y2)/y2 EQ 0) THEN BEGIN
>>>>         print, "Series converged before matching integral!"
>>>>         BREAK
>>>>       ENDIF
>
>>>>       lastVal = y2
>>>>     ENDFOR
```

>
>>>>   IF n EQ 169 THEN PRINT, "Equations did not meet!"
>
>>>>   RETURN
>>>> END
>
>>> Thank you all for your answers. It really made my day!
>>> I've tested the 3 methods for the "Mathematica" example (i.e. for
>>> 1F1(1,2,[-5..5]))
>>> HYPERGEOMETRICONEFONE, CHSS and CHINS seem to work fine for this
>>> example. HYPERGEOMETRICONEFONE seems a bit faster than the other ones.
>>> Unfortunately I am more interested in evaluating something like 1F1(-.
>>> 75,1,40) :
>
>>> IDL> print,hypergeometriconefone(-0.75D,1D,-40D)
>>>   NaN
>
>>> IDL> print, chins([-0.75D],[1D],-40D)
>>>   NaN
>
>>> IDL> print, chss(169D,[-0.75D],[1D],-40D)
>>>   17.478776
>
>>> The mathematica website gives [http://functions.wolfram.com/
>>>   webMathematica/FunctionEvaluation.jsp?name=Hypergeometric1F1]:

>
>>> Thanks,
>>> Noah
>
>> Phew, you're really pushing my little programme to the limit here! ;-)
>
>> If you put this line:
>> Print, [ThisResult, K]
>> into my programme just before the line with ENDWHILE on it, you'll see
>> that the function actually gets quite close to the answer before it
>> succumbs to floating overflow. On my machine, the results of the last
>> few iterations are:
>>   17.462850   96.000000
>>   17.015058   97.000000
>>   17.196383   98.000000
>>   17.123695   99.000000
>
>> If you try "print, hypergeometriconefone(-0.75D,1D,-40D, prec=1)", you
>> should get similar results.
>
>> I suspect IDL won't really help you out here, at least until someone
>> invents 128-bit quadruple-precision data type :-(

>> Or, unless you use a better algorithm, I guess. chss seems to be that
>> better algorithm in this case.
>
>> Actually, it would help to replace this:
>> F = (AK * Z^K) / (BK * FACTORIAL(K)) ; Evaluate function.
>
>> with:
>> F = (AK / BK) * (Z^K / FACTORIAL(K)) ; Evaluate function.
>
>> Though I suspect it's just putting off the inevitable by a few more
>> iterations :-(
>> Also, there should've been some sort of protection for preventing k
>> from going over 169, which is the limit at which FACTORIAL can work
>> reliably.
>
>> I'll try and amend the programme and repost if you're having
>> difficulty piecing my garbled lines together into your copy.
>
> The more I stare at this, the more I realise that my guess at
> available precision is WAY off the mark :-(
> I'll go stare at the 'Sky is falling' page for another while, shall
> I? :-P
> I need to make sure the precision is determined by the significant
> figures, not just an absolute.
>
> Anyway, for now if you get NaN results, just try dropping PRECISION
> below 5.
> If you want acres of spam, call the function with the debug keyword
> set.
>
> e.g.:
> Test =
>
> read more >>- Hide quoted text -
>
> - Show quoted text -...

Noah, Chris,

The fact that the series expansions (chss) works better for some paramters is what led me to use both implementations. You may not need as many terms (169) as you are using. I usually use about 20 - 50 and see no change in the result after that. I am a little lazier than Chris, so I didn't put a lot of effort into determining why the integral representation was failing. If someone pieces that together, I would like to hear about it.

- dan

Subject: Re: Confluent Hypergeometric Function of the First Kind
Posted by Matthias Cuntz on Wed, 05 Mar 2008 22:01:23 GMT
View Forum Message <> Reply to Message

noahh.schwartz@gmail.com wrote:
> Hello everyone,
>
> I am looking for the Confluent Hypergeometric Function of the First
> Kind in the IDL Math Library but it does not seem to be implemented!
>
> I would like to use a function similar to the Hypergeometric1F1[a, b,
> z] of Mathematica [http://reference.wolfram.com/mathematica/ref/
> Hypergeometric1F1.html].
>
> I have not found what I was looking for, and so decided to try to code
> it my self... [sigh...]. Beeing a fresh beginner in IDL this is a hard
> task!
>
> Would anybody know how to code an infinite series expansion like the
> Hypergeometric1F1?
>
> Thank you in advance for your time!
> Noah

Dear Noah,

I am a bit late but just came back to the office.

I am using a Fortran-77 program which comes from the book: "Computation
of Special Functions" of Zhang & Jin:
http://jin.ece.uiuc.edu/routines/routines.html

I very simple changed the program mchgm.for so that it works with arrays.
I compile it locally and then call it with spawn in mc_kummer.pro
The IDL routine looks a bit cluttered because I use it on different
machines.

BTW, mc_kummer uses mc_tmp() that produces a unique temporay file -
replace by whatever you want. It also uses mc_setup(/mcbin) that tells
it where it finds the executable - replace with your a.out.

Cheers
Matthias

```
;+
; NAME:
;       MC_KUMMER
;
```

```
; PURPOSE:
;       Computes the Kummer or confluent hypergeometric function
;
; CATEGORY:
;       Math.
;
; CALLING SEQUENCE:
;       res =  mc_kummer(a,b,x)
;
; INPUTS:
;       a:    parameter array
;       b:    parameter array
;       x:    variable array
;
; OUTPUTS:
;       res:  confluent hypergeometric series
;
; EXAMPLE:
;       IDL> den = mc_kummer(-0.5d*ka , 0.5d , -0.5d*peclet_l)
;
; PROCEDURE:
;       mc_kummer calls the external fortran program
;       confluent_hypergeometric_series_M to compute the Kummer function
;
; REFERENCE:
;       Weisstein E.W. (2004b) Hypergeometric function.
;       In: MathWorld - A Wolfram Web Resource,
;       Website: http://mathworld.wolfram.com/HypergeometricFunction.html
;
; MODIFICATION HISTORY:
;       Written by: JO, Sep 2004
;       Modfied     MC, Feb 2005 - a, b, x arrays
;                   MC, Jun 2007 - undef
;-
FUNCTION MC_KUMMER, a, b, x, undef=undef
  COMPILE_OPT idl2,
  ON_ERROR, 2
  ; ---
  os = ([1,2,3,4,5])[(where([ 'darwin', 'win32', 'linux', 'osf', 'aix' ] eq strlowcase(!version.os)))[0]]
  ;
  if n_elements(undef) eq 0 then undef = -9999.
  aa = a
  bb = b
  xx = x
  iiundef = where(aa eq undef or bb eq undef or xx eq undef, undefcount)
  if undefcount gt 0 then begin
     aa[iiundef] = 0.5
     bb[iiundef] = 0.5
```

```
   cc[iiundef] = 0.5
endif
;
tempfile = mc_tmp()
reportfile = mc_tmp()
;---
n = n_elements(x)
result = ''
;---
get_lun, unit
openw, unit, tempfile
printf, unit, n
for i=0, n-1 do printf, unit, a[i], b[i], x[i]
free_lun, unit
; --- Run C program
newpath=mc_setup(/mcbin)
case os of
   1: begin
      ccode = newpath+'confluent_hypergeometric_series_M.Mac'
      if not file_test(ccode) then $
        message,'No executable found for this platform: '+ccode
      spawn, ccode+' < ' + tempfile + ' > ' + reportfile, summary, /sh
   end
   2: begin
      ccode=newpath+'confluent_hypergeometric_series_M.exe'
      if not file_test(ccode) then $
        message,'No executable found for this platform: '+ccode
      spawn, ccode+' < ' + tempfile + ' > ' + reportfile, summary, /log_output
   end
   3: begin
      spawn, 'uname -a', uname, /sh
      uname = strmid(uname,0,6)
      if uname eq 'obelix' then $
        ccode = newpath+'confluent_hypergeometric_series_M.Linux.obelix' $
      else if uname eq 'asterix' then $
        ccode = newpath+'confluent_hypergeometric_series_M.Linux.asterix' $
      else $
        ccode = newpath+'confluent_hypergeometric_series_M.Linux'
      if not file_test(ccode) then $
        message,'No executable found for this platform: '+ccode
      spawn, ccode+' < ' + tempfile + ' > ' + reportfile, summary, /sh
   end
   4: begin
      ccode = newpath+'confluent_hypergeometric_series_M.DEC'
      if not file_test(ccode) then $
        message,'No executable found for this platform: '+ccode
      spawn, ccode+' < ' + tempfile + ' > ' + reportfile, summary, /sh
   end
```

```
      5: begin
         ccode = newpath+'confluent_hypergeometric_series_M.IBM'
         if not file_test(ccode) then $
           message,'No executable found for this platform: '+ccode
         spawn, ccode+' < ' + tempfile + ' > ' + reportfile, summary, /sh
      end
  endcase
  ; ---
  res = dblarr(n)
  get_lun, unit
  openr, unit, reportfile
  for i=0, n-1 do begin
     readf, unit, result
     res[i] = double(result)
  endfor
  free_lun, unit
  ; ---
  file_delete, tempfile, reportfile
  ;
  if undefcount gt 0 then res[iiundef] = undef
  ;
  if n gt 1 then return, res else return, res[0]
  ;
END
;-

      PROGRAM MCHGM
C
 C=============================================================
C       Purpose: This program computes the confluent
C                hypergeometric function M(a,b,x) using
C                subroutine CHGM
C       Input  : a  --- Parameter
C                b  --- Parameter ( b &lt;&gt; 0,-1,-2,... )
C                x  --- Argument
C       Output:  HG --- M(a,b,x)
C       Example:
C                  a       b       x         M(a,b,x)
C                 ----------------------------------------
C                  1.5     2.0    20.0     .1208527185D+09
C                  4.5     2.0    20.0     .1103561117D+12
C                 -1.5     2.0    20.0     .1004836854D+05
C                 -4.5     2.0    20.0    -.3936045244D+03
C                  1.5     2.0    50.0     .8231906643D+21
C                  4.5     2.0    50.0     .9310512715D+25
C                 -1.5     2.0    50.0     .2998660728D+16
C                 -4.5     2.0    50.0    -.1806547113D+13
C
```

```
 C=======================================================
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      integer i, n
      LOGICAL debug
      PARAMETER (debug = .FALSE.)
C
      if (debug) then
        WRITE(*,*) 'Please enter a, b and x '
        READ(*,*) A,B,X
        WRITE(*,*) '    a       b       x          M(a,b,x)'
        WRITE(*,*) ' ----------------------------------------'
        CALL CHGM(A,B,X,HG)
        WRITE(*,'(1X,F5.1,3X,F5.1,3X,F5.1,D20.10)') A,B,X,HG
      else
        read(*,*) n
        do i=1, n
          READ(*,*) A,B,X
          CALL CHGM(A,B,X,HG)
          WRITE(*,*) HG
        end do
      endif
C
      END PROGRAM MCHGM
C
 C////////////////////////////////////////////////////////// //
 C////////////////////////////////////////////////////////// //
 C////////////////////////////////////////////////////////// //
 C////////////////////////////////////////////////////////// //
 C////////////////////////////////////////////////////////// //
C
      SUBROUTINE CHGM(A,B,X,HG)
C
 C=======================================================
C       Purpose: Compute confluent hypergeometric function
C                M(a,b,x)
C       Input  : a  --- Parameter
C                b  --- Parameter ( b &lt;&gt; 0,-1,-2,... )
C                x  --- Argument
C       Output:  HG --- M(a,b,x)

 C=======================================================
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      LOGICAL debug
      PARAMETER (debug = .FALSE.)
C
      PI=3.141592653589793D0
```

```
      A0=A
      A1=A
      X0=X
      HG=0.0D0
C
C---  SPECIAL CASES
C
C--- case where b=-n --> M(a,b,x) is undefined (13.1.3)
      IF (B.EQ.0.0D0.OR.B.EQ.-ABS(INT(B))) THEN
        if (debug) WRITE(*,*) 'b=-n --> M(a,b,x) is undefined'
        HG=1.0D+300
C--- case where a=0 or x=0 --> M(a,b,x)=1 (13.1.2)
      ELSE IF (A.EQ.0.0D0.OR.X.EQ.0.0D0) THEN
        if (debug) WRITE(*,*) 'a=0 or x=0 --> M(a,b,x) = 1'
        HG=1.0D0
C--- case where a=-1 --> M(a,b,x)=1-x/b (13.6.27)
      ELSE IF (A.EQ.-1.0D0) THEN
        if (debug) WRITE(*,*)
     .    'a=-1 --> M(a,b,x) = polynomial of degree 1'
        HG=1.0D0-X/B
C--- case where a=b --> M(a,b,x) = exp(x) (13.6.12)
      ELSE IF (A.EQ.B) THEN
        if (debug) WRITE(*,*) 'a=b --> M(a,b,x) = exp(x)'
        HG=DEXP(X)
C--- case where a=b+1 -->  M(a,b,x) = {1+x/b}*exp(x) (13.6.2)
      ELSE IF (A-B.EQ.1.0D0) THEN
        if (debug) WRITE(*,*) 'a=b+1 --> M(a,b,x) = {1+x/b}*exp(x)'
        HG=(1.0D0+X/B)*DEXP(X)
C--- case where a=1 and b=2 --> M(a,b,x) = {exp(x)-1}/x (13.6.14)
      ELSE IF (A.EQ.1.0D0.AND.B.EQ.2.0D0) THEN
        if (debug) WRITE(*,*) 'a=1 and b=2 --> M(a,b,x) = {exp(x)-1}/x'
        HG=(DEXP(X)-1.0D0)/X
C--- case a=-m --> polynomial of degree m (13.1.3 and 13.6.9)
      ELSE IF (A.EQ.INT(A).AND.A.LT.0.0D0) THEN
        if (debug) WRITE(*,*)
     .    'a=-m --> M(a,b,x) = polynomial of degree m'
        M=INT(-A)
        R=1.0D0
        HG=1.0D0
        DO K=1,M
          R=R*(A+K-1.0D0)/K/(B+K-1.0D0)*X
          HG=HG+R
        ENDDO
      ENDIF
      IF (HG.NE.0.0D0) RETURN
C
C---  GENERAL CASE
C
```

```
C--- case x<0 --> M(a,b,x) = exp(x)*M(a,b-a,-x) (13.1.27)
      IF (X.LT.0.0D0) THEN
        if (debug) WRITE(*,*) 'x<0 --> M(a,b,x) = exp(x)*M(b-a,b,-x)'
        A=B-A
        A0=A
        X=DABS(X)
      ENDIF
C--- from now on we try to evaluate M(A,B,X) even if (A,X) is not equal to (A1,X0)
C--- case ??????????
      IF (A.LT.2.0D0) NL=0
      IF (A.GE.2.0D0) THEN
        NL=1
        LA=INT(A)
        A=A-LA-1.0D0
      ENDIF
C--- loop on ??????????
      DO N=0,NL
        IF (A0.GE.2.0D0) A=A+1.0D0
        IF (X.LE.30.0D0+DABS(B).OR.A.LT.0.0D0) THEN
C--- use classical formula 13.1.2 for small x and small a
          if (debug) WRITE(*,*)
     .     'classic formula for small x with up to 500 maximum terms'
C         HG=1.0D0
          HG=1.0D0*DEXP(X0)
          RG=1.0D0
          J=0
          DO WHILE (DABS(RG/HG).GE.1.0D-15 .OR. J .LT. 500)
            J=J+1
            RG=RG*(A+J-1.0D0)/(J*(B+J-1.0D0))*X
C             HG=HG+RG
            HG=HG+RG*DEXP(X0)
          ENDDO
        ELSE
C--- computation with asymptotic formula 13.5.1 up to 8 digits
          if (debug) WRITE(*,*)
     .     'asymtpotic formula for big x up to 8 digits'
          CALL GAMMA(A,TA)
          CALL GAMMA(B,TB)
          XG=B-A
          CALL GAMMA(XG,TBA)
          SUM1=1.0D0
          SUM2=1.0D0
          R1=1.0D0
          R2=1.0D0
          DO I=1,8
            R1=-R1*(A+I-1.0D0)*(A-B+I)/(X*I)
            R2=-R2*(B-A+I-1.0D0)*(A-I)/(X*I)
            SUM1=SUM1+R1
```

```
          SUM2=SUM2+R2
        ENDDO
C       HG1=TB/TBA*X**(-A)*DCOS(PI*A)*SUM1
        HG1=TB/TBA*X**(-A)*DCOS(PI*A)*SUM1*EXP(X0)
C       HG2=TB/TA*DEXP(X)*X**(A-B)*SUM2
        HG2=TB/TA*DEXP(X+X0)*X**(A-B)*SUM2
        HG=HG1+HG2
      ENDIF
      IF (N.EQ.0) Y0=HG
      IF (N.EQ.1) Y1=HG
    ENDDO
    IF (A0.GE.2.0D0) THEN
      DO I=1,LA-1
        HG=((2.0D0*A-B+X)*Y1+(B-A)*Y0)/A
        Y0=Y1
        Y1=HG
        A=A+1.0D0
      ENDDO
    ENDIF
C  IF (X0.LT.0.0D0) HG=HG*DEXP(X0)
    IF (X0.GE.0.0D0) HG=HG*DEXP(-X0)
    A=A1
    X=X0
    RETURN
    END SUBROUTINE CHGM
C
 C/////////////////////////////////////////////////////////// //
 C/////////////////////////////////////////////////////////// //
 C/////////////////////////////////////////////////////////// //
 C/////////////////////////////////////////////////////////// //
 C/////////////////////////////////////////////////////////// //
C
    SUBROUTINE GAMMA(X,GA)
C
 C=========================================================
C
C
 C=========================================================
C
    IMPLICIT DOUBLE PRECISION (A-H,O-Z)
    DIMENSION G(26)
    PI=3.141592653589793D0
    IF (X.EQ.INT(X)) THEN
      IF (X.GT.0.0D0) THEN
```

```
        GA=1.0D0
        M1=X-1
        DO K=2,M1
          GA=GA*K
        ENDDO
      ELSE
        GA=1.0D+300
      ENDIF
    ELSE
      IF (DABS(X).GT.1.0D0) THEN
        Z=DABS(X)
        M=INT(Z)
        R=1.0D0
        DO K=1,M
          R=R*(Z-K)
        ENDDO
        Z=Z-M
      ELSE
        Z=X
      ENDIF
      DATA G /1.0D0,0.5772156649015329D0,
     &   -0.6558780715202538D0, -0.420026350340952D-1,
     &   0.1665386113822915D0,-.421977345555443D-1,
     &   -.96219715278770D-2, .72189432466630D-2,
     &   -.11651675918591D-2, -.2152416741149D-3,
     &   .1280502823882D-3, -.201348547807D-4,
     &   -.12504934821D-5, .11330272320D-5,
     &   -.2056338417D-6, .61160950D-8,
     &   .50020075D-8, -.11812746D-8,
     &   .1043427D-9, .77823D-11,
     &   -.36968D-11, .51D-12,
     &   -.206D-13, -.54D-14, .14D-14, .1D-15/
      GR=G(26)
      DO K=25,1,-1
        GR=GR*Z+G(K)
      ENDDO
      GA=1.0D0/(GR*Z)
      IF (DABS(X).GT.1.0D0) THEN
        GA=GA*R
        IF (X.LT.0.0D0) GA=-PI/(X*GA*DSIN(PI*X))
      ENDIF
    ENDIF
    RETURN
    END SUBROUTINE GAMMA
```

File Attachments

1) mc_kummer.pro, downloaded 109 times
2) confluent_hypergeometric_series_M.f, downloaded 116 times