
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](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(/mcbn) 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
;   Modified  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
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 <> 0,-1,-2,... )
C          x --- Argument
C  Output: HG --- M(a,b,x)
C
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 $b=-n \rightarrow M(a,b,x)$ is undefined (13.1.3)

IF (B.EQ.0.0D0.OR.B.EQ.-ABS(INT(B))) THEN

if (debug) WRITE(*,*) 'b=-n $\rightarrow M(a,b,x)$ is undefined'

HG=1.0D+300

C--- case where $a=0$ or $x=0 \rightarrow M(a,b,x)=1$ (13.1.2)

ELSE IF (A.EQ.0.0D0.OR.X.EQ.0.0D0) THEN

if (debug) WRITE(*,*) 'a=0 or x=0 $\rightarrow M(a,b,x) = 1$ '

HG=1.0D0

C--- case where $a=-1 \rightarrow M(a,b,x)=1-x/b$ (13.6.27)

ELSE IF (A.EQ.-1.0D0) THEN

if (debug) WRITE(*,*)

. 'a=-1 $\rightarrow M(a,b,x) =$ polynomial of degree 1'

HG=1.0D0-X/B

C--- case where $a=b \rightarrow M(a,b,x) = \exp(x)$ (13.6.12)

ELSE IF (A.EQ.B) THEN

if (debug) WRITE(*,*) 'a=b $\rightarrow M(a,b,x) = \exp(x)$ '

HG=DEXP(X)

C--- case where $a=b+1 \rightarrow M(a,b,x) = \{1+x/b\} \exp(x)$ (13.6.2)

ELSE IF (A-B.EQ.1.0D0) THEN

if (debug) WRITE(*,*) 'a=b+1 $\rightarrow M(a,b,x) = \{1+x/b\} \exp(x)$ '

HG=(1.0D0+X/B)*DEXP(X)

C--- case where $a=1$ and $b=2 \rightarrow M(a,b,x) = \{\exp(x)-1\}/x$ (13.6.14)

ELSE IF (A.EQ.1.0D0.AND.B.EQ.2.0D0) THEN

if (debug) WRITE(*,*) 'a=1 and b=2 $\rightarrow M(a,b,x) = \{\exp(x)-1\}/x$ '

HG=(DEXP(X)-1.0D0)/X

C--- case $a=-m \rightarrow$ 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 $\rightarrow M(a,b,x) =$ 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 133 times

2) [confluent_hypergeometric_series_M.f](#), downloaded 141 times
