超幾何級数（Hypergeometric Function ）と累積分布関数（CDFs ）in J

by Ewart Shaw

Vector Vol 18 No 4

目次

Ewart Shaw の hypergeometric series 超幾何級数に関する Vecter 掲載論文の紹介と若干のコメントである。超幾何級数は本論文でも触れられているように,Graham,Knuth,Abramowitz の著書 Concrete Mathematics に詳しく紹介されている。K.Iverson が Concrete Mathematics Comparison を著した時に J に H. が組み込まれたが,難解であった。この論文は簡潔に H. とその応用を紹介しており, 有用である。紹介されている J のスクリプトも確認できる範囲で添付した.

最後は E.Mcdonell による Black-Sholes Model で、この中で, 正規分布を求めるときに, 超幾何分布が利用されている.

1 Introduction

A valuable but little-mentioned feature of the J language is the conjunction H. with which hypergeometric series and hence many important mathematical function can be constructed. A hypergeometric series has the general form

有用だがあまり言及されていない J の機能に接続詞 · 超幾何級数 (hypergeometric) H. がある。この超幾何級数シリーズは多くの数学上重要な関数を作り出すことができる.

超幾何級数 H. は次によりあらわされる。

$_qF_p(a_1\cdots a_p;b_1\cdots b_q;z) = \sum_{n=0}^{\infty} \frac{(a_1)_n\cdots(a_p)_n\, z^n}{(b_1)_n\cdots(b_q)_n\, n!}$

where (a)$_n = a(n+1)\cdots(a+n-1)\ for\ n > 1,\ and\ (a)_0 = 1$

The equivalent J representation of

$_qF_p(a_1\cdots a_p;b_1\cdots b_q;z)\ is\ (a_1\cdots a_p\ H.\ b_1\cdots b_q)z$

For example

$_1F_1(1;1;\cdot)$ reduces to the exponential function, as shown in J by

[*1]

```
 17j14 ": ^i.3      NB. exp(0),exp(1),exp(2) to 15 dights
1.00000000000000 2.71828182845905 7.38905609893065
```

```
  17j14 ": 1 H.1 i.3 NB. or simply '' H. '' i. 3
```

*1 Error in J502a ,J406 is OK

```
 1.00000000000000 2.71828182845905 7.38905609893065
```

This article describes how the cumulative distribution functions (CDFs) of many common probability distributions can be expressed via hypergeometric series, making it easy to produce statistical tables and (for example) tail areas arising from t, chi-squared and F tests. Other examples using the hypergeomctric conjunction H. are given in lverson (1995), with particular reference to Graham, Knuth Patashnik (1988) and Abramowitz Stegun (1970). Another useful source for hypergeometric identities is Spanier Oldham (1987).

多くの確率分布の累積分布関数 (CDFs) を超幾何級数 H. を用いて作ることができる. 例えば $t, \chi^2, Ftest$ 等々

H. は　Iverson によって提起された。Graham,Knuth,Patashnik(1988),AbramowitzStegan(1970), SpainerOldham(1987) を参照している.

2　The Error Function and the Normal CDF

The standard Normal distribution N(0,1), with mean 0 and variance 1, has a CDF Φ(·) easily expressed in terms of the errorfunction erf(·):

平均０、分散 1 の正規分布 $N(0,1)$ は残差の関数 erf で作れる

```
erf=: (*&(%:4p_1) % ^@:*:) * [: 1 H. 1.5 *:
n01cdf=: [: -: 1: + [: erf %&(%:2)   NB. CDF of N(0,1)
```

On a PC, the resulting approximation has a maximum absolute error of roughly 10^{14}, though the relative error might be judged unacceptable more than (say) seven standard deviations from the mean.

PC の計算精度は最大でも 10^{14} を越えない。

```
   n01cdf _15 _8 _7 0 1.96 7     NB. accuratr in range(_7 7)
3.67097e_51 6.22096e_16 1.27981e_12 0.5 0.975002 1

   ncdf
n01cdf :([: n01cdf ((] - ({.@[)) % ((%:@{:)@[)))
```

```
   5 4 ncdf 1    NB. Pr(X<1) where X~N(5,4)
0.0227501
```

for extreme Values of z, the tail areas $\phi(z)$) or $1 - \Phi(z)$ are well approximated using continued fractions, see Abramowitz Stegun. A simple such approximation is:

```
n01pdf =: ([: ^ _0.5"_ * *:) % (%: 2p1)"_
n01cdf2=: n01cdf * (1: - [: % *: +3:) % ]
ncdf=: n01cdf : ([: n01cdf ( ] - {.@[)% %:@{:@[)
NB.   ts

   n01cdf2 _20 _8 _7  NB. testing
_1.3734e_90 _7.66014e_17 _1.79314e_13
```

Note that other hypergeometric series formulae for the error function exist but suffer serious convergence problems. For example:

```
erfbad =: *&(%:4p_1)  * [: 0.5  H. 1.5 -@: *:

  14j11 ": erfbad 0.5 + 2*i.4
 0.52049987781 0.99959304798 0.99999999883 3.53955177465

   14j11 ": erf 0.5 + 2*i.4
 0.52049987781 0.99959304798 0.99999999980 1.00000000000
   erfbad
(*&(1.1
```

3 The Incomplete Gamma and Beta Functions

The CDFs of gamma (including chi-squared) and Poisson distributions can be obtained from the incomplete gamma function, and CDFs of beta, t, F and binomial distributioru from the incomplete beta function.

```
NB.==============
gamma=: ! & <: NB. Gamma function
ig0=: 4 : ' ( 1 H. (1+x.) % x. &((* ^) * (^ -)~)) y.'
incgam=: ig0 % gamma@[  NB. incomplete gamma
beta=: *&gamma % gamma@+
ib0=: 4 : '(((,-.)/x.) H. (1+{. x.) * (^ % ])&({. x.)) y.'
incbet=: ib0 % [: beta / [ NB. incompletw beta

   gamma 5 6 7 1.5
24 120 720 0.886227

   6 ig0 0 5 10 20 30
0 46.0847 111.95 119.991 120

   6 incgam 0 5 10 20 30
0 0.384039 0.932914 0.999928 1

   4 beta 2 3 4 1.5
0.05 0.0166667 0.00714286 0.101587

   4 4 2 4 beta 2 3 4 1.5
0.05 0.0166667 0.05 0.101587

   4 2 ib0 0 0.1 0.6 0.7 1
0 2.3e_5 0.016848 0.026411 0.05

   4 2 incbet 0 0.1 0.6 0.7 1
0 0.00046 0.33696 0.52822 1
```

the interrelationships between these functions and various CDFs, illustrated below, ,are conveniently summarised in Press et al. (1986). As with the Normal distribution, extreme tail areas would require alternative methods.

```
  NB.==================
NB. (n,p) bincdf y.  binominal(n,p)
bincdf=: (({.@[ - ]),.>:@]) incbet "1 _ -.@{:@[
NB. c chisqcdf y.    chi-squared on n d.f.
chisqcdf=: incgam&-:
NB. (n1,n2)fcdf y.   F on (n1,n2) d.f.
fcdf=: -:@[ incbet ({. % +/)@(* ,:&1)
NB. mu poissoncdf y. Poisson, mean mu
poissoncdf=: -.@incgam "0~ >:
NB. n tcdf y.        t on n d.f.
tcdf=: [: -:@>: *@] * 1&,@] fcdf *:@]
NB. binpmf=: (!/{.)~ * [: */ (, -.)@{:@[ ^£] ,: {.@[ - ]
poissonpmf=: ^ * ^@-@[ % !@]



  5 0.2 bincdf i.6
0.32768 0.73728 0.94208 0.99328 0.99968 _.

   5 chisqcdf 0.831211 4.35146 11.0705 20.515
0.025 0.5 0.95 0.999

   3 10 fcdf 0.84508 3.70826 6.55231 12.5527
0.5 0.95 0.99 0.999

   +/\ 2.3 poissonpmf i.5
0.100259 0.330854 0.596039 0.799347 0.916249
```

```
   2.3 poissoncdf i.5
0.100259 0.330854 0.596039 0.799347 0.916249
```

4 Implementation and Summary

Classes of probability distributions, including their CDFs, pseudorandom variable generation etc., Can be created using J's object-oriented capabilities. Implementation details involve splicing together different approximations over different ranges and iteration to approximate inverse CDFs such as Φ^{-1}, but such details can mercifully be hidden from the user. The hypergeometric conjunction H. makes approximations over the regions of most interest easy in J. Hypergeometric series are also not difficult to implement in traditional APL and other languages, and the above methods deserve to be more widely known and used.

5 References

Abramowitz, M. and Stegun, I.A. (1970). Handbook of Mathematical Functions, Dover, New York.

Graham, R.L., Ruth, D.E., and Patashnik, 0. (1988). Concrete Mathematics, Addison-Wesley, Reading, Massachusetts.

Iverson, K.I. (1995). Concrete math Companion, Iverson Software lnc., Toronto.

Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. (1986). Numerical Recipes, Cambridge University Press, Cambridge.

Spanier, J. and Oldham, K.B. (1987). An Atlas ofFunctions, Springer-Verlag, Berlin.

6 B-S Model by E.McDonnell

```
Jforum: Black-Scholes formula

Date: Mon, 14 Oct 2002 16:13:59 EDT
From: Eemcd@aol.com
Reply-To: forum@jsoftware.com
```

```
To: forum@jsoftware.com

The argument to the Black-Scholes formulae for Call and Put options has five
items:

S - the current price of the asset
X - the strike price (price when option is to be exercised)
T - the time in years
r - the risk-free interest rate
v - the volatility, typically the standard deviation of S for the last 3 or 6
months
```

Hu the posted a suite of four programs to this forum last June to give the result of Black-Scholes calls and puts. For the most part they followed those written in most other programming languages, but did make use of several of the functions that are unique to J. Oleg Kobchenko suggested a change that used an array approach to compute the final results. I suggested a change that used an array approach to computing the derivatives needed. Arthur Whitney gave a K solution that permitted the same program to solve for either call or put simply by the sign of the volatility parameter v.

Here is a function that uses all these.

The Black Scholes formula is essentially the difference of products, -/M*D

The derivatives d1 and d2 are replaced by a 2-item list D:

```
D =: N((^.S%X)+T*r(+,-)-:*:v)%v*%:T
```

where N is the cumulative normal distribution function.

The money item M consists of S, the current asset price, and X*$^{-}r*T$, *thepresentvalueofthestrikepriceX, at*

```
M =: S,X*^-r*T
```

The result is the difference of their products:

```
-/M*D   NB. on my computer, slightly faster than M -/ .*D
```

The overall function can then be written as:

```
BS =: monad define
```

```
'S X T r v' =. y.
-/(S,X*^-r*T)*N((^.S%X)+T*r(+,-)-:*:v)%v*%:T
)
```

For example, let y =: 60 65 0.25 0.08, and append v or -v depending on whether a call or put is desired:

call:

```
   BS y, 0.3
2.13337
```

put:

```
   BS y,_0.3
_5.84628
```

The version of N that I use is due to Ewart Shaw in Vector 18.4. He uses J's Hypergeometric conjunction and the second expression in Abramowitz Stegun 7.1.21 for the error function erf:

```
erf=:(*&(%:4p_1) % ^@:*:) * [: 1 H. 1.5 *: NB. Ewart Shaw Vector 18.4
cnd =:N=: [: -: 1: + [: erf %&(%:2)   NB. CDF of N(0,1)
```

Altogether, the J solution is quite compact: 42 tokens for BS and 25 for N and erf.

Eugene McDonnell